# Efficient, Convergent SENSE MRI Reconstruction for Nonperiodic Boundary Conditions via Tridiagonal Solvers

Mai Le, *Student Member, IEEE*, and Jeffrey A. Fessler, *Fellow, IEEE*

***Abstract*—Undersampling is an effective method for reducing scan acquisition time for MRI. Strategies for accelerated MRI, such as parallel MRI and compressed sensing MRI present challenging image reconstruction problems with nondifferentiable cost functions and computationally demanding operations. Variable splitting (VS) can simplify implementation of difficult image reconstruction problems, such as the combination of parallel MRI and compressed sensing, CS-SENSE-MRI. Combined with augmented Lagrangian (AL) and alternating minimization strategies, variable splitting can yield iterative minimization algorithms with simpler auxiliary variable updates. However, arbitrary variable splitting schemes are not guaranteed to converge. Many variable splitting strategies are combined with periodic boundary conditions. The resultant circulant Hessians enable $\mathcal{O}(n \log n)$ computation but may compromise image accuracy at the spatial boundaries. We propose two methods for CS-SENSE-MRI that use regularization with nonperiodic boundary conditions to prevent wrap-around artifacts. Each algorithm computes one of the resulting variable updates efficiently in $\mathcal{O}(n)$ time using a parallelizable tridiagonal solver. AL-tridiag is a VS method designed to enable efficient computation for nonperiodic boundary conditions. Another proposed algorithm, ADMM-tridiag, uses a similar VS scheme but also ensures convergence to a minimizer of the proposed cost function using the alternating direction method of multipliers (ADMM). AL-tridiag and ADMM-tridiag show speeds competitive with previous VS CS-SENSE-MRI reconstruction algorithm AL-P2. We also apply the tridiagonal VS approach to a simple image inpainting problem.**

***Index Terms*—Alternating direction method of multipliers (ADMM), augmented Lagrangian (AL), denoising, inpainting, Image reconstruction, non-periodic boundaries, parallel magnetic resonance imaging (MRI), tridiagonal solvers, variable splitting.**

## I. Introduction

ACCELERATED scan time benefits many applications of Magnetic Resonance Imaging (MRI), reducing cost and motion blurring. A popular strategy for reducing scan time is to acquire fewer k-space samples. To compensate for the reduced sampling, parallel MRI and Compressed Sensing (CS) are often used. SENSitivity Encoded (SENSE) MRI is a popular framework for parallel MRI [1]. By simultaneously acquiring data from multiple receive coils with spatially varying sensitivities, more data can be collected without additional scan time. SENSE can be combined with CS-inspired techniques to further reduce acquisition time [2]. Irregular undersampling patterns enable higher acceleration but require iterative model-based image reconstruction.

CS MRI reconstructs images assuming image sparsity in some transform domain. To balance adherence of the estimated image to the noisy, undersampled data with the prior assumption of sparsity, we seek to minimize a cost function that describes both. These methods employ regularization with $\ell_1$ norms that promote sparsity but also present computational challenges.

Variable splitting (VS) [3]–[5] is a versatile optimization approach for these cost functions. VS decouples a costly nonlinear optimization into simpler problems via the augmented Lagrangian (AL) framework. VS converts the original cost function into a constrained cost function involving additional auxiliary variables updated with alternating minimization. The AL-P2 algorithm proposed in [4] demonstrated that VS combined with AL can yield significant speed gains over conjugate gradients (CG) and monotone fast iterative shrinkage-thresholding algorithm (MFISTA) [6] for CS-SENSE-MRI. However, AL-P2 lacks a convergence guarantee and uses periodic boundary conditions when applying sparsifying transforms. This assumption leads to a circulant Hessian and an $\mathcal{O}(n \log n)$ FFT-based solution for one of the inner problems, where $n$ is the number of pixels.

The use of periodic boundary conditions is not reasonable for reconstructing 2D axial slices of the brain, when it is surrounded by air at all boundaries. However, when reconstructing an entire 3D volume, it could be undesirable to impose periodic boundary conditions across the top and bottom slices. Non-periodic boundary conditions are also useful in dynamic imaging with acyclic temporal behavior. For example, Dynamic Contrast Enhanced (DCE) images differ significantly in the final frame compared to the initial frame.

Here we present two related variable splitting methods for CS-SENSE-MRI: ADMM-tridiag and AL-tridiag. ADMM-tridiag leverages [7] to ensure convergence. Both proposed algorithms use a regularizer with non-periodic boundary conditions to more

Manuscript received May 2, 2016; revised September 20, 2016; accepted October 21, 2016. Date of publication November 8, 2016; date of current version February 8, 2017. This work was supported in part by the NIH under Grant P01 CA 059827. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. James E. Fowler.

The authors are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: mtle@umich.edu; fessler@umich.edu).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCI.2016.2626999

accurately reflect the reality of the unknown image. Both algorithms incorporate parallelizable tridiagonal solvers that efficiently handle the non-periodic boundary conditions with $\mathcal{O}(n)$ operations. We present numerical results with real *in vivo* data to demonstrate the efficacy of the proposed methods.

We also apply the principles to describe novel VS algorithms for the simpler "special case" of image inpainting using regularization based on a combination of wavelets and anisotropic total variation (TV) with non-periodic boundary conditions.

## II. Problem Formulation

The analysis formulation often used for SENSE MRI reconstruction estimates the unknown image, $\hat{x}$, by seeking the minimizer of a cost function consisting of a datafit term plus a regularizer. Regularization is particularly important for undersampled problems.

Many regularizers have been used for SENSE MRI reconstruction. For CS-SENSE-MRI, sparsifying transforms such as wavelets and first-order finite differences are often used with an $\ell_1$ norm to promote sparsity [2]. This paper focuses on Cartesian SENSE reconstruction with regularizers having non-periodic boundary conditions.

Let $N_c$ denote the number of sensitivity coils, $N_s$ the number of samples received from each coil, and $N_r = N_x N_y$ the number of pixels in the latent image $x$. We formulate regularized SENSE reconstruction as the following optimization problem:

$$\hat{x} = \operatorname*{argmin}_x \frac{1}{2} \|y - \mathbf{F}\mathbf{S}x\|_2^2 + \lambda \|\mathbf{C}_{\mathrm{H}} x\|_1 + \lambda \|\mathbf{C}_{\mathrm{V}} x\|_1 \quad (1)$$

where $y \in \mathbb{C}^{N_c N_s}$ is the undersampled k-space data from all coils, $\mathbf{F} \in \mathbb{C}^{N_c N_s \times N_c N_r}$ is a block diagonal matrix consisting of undersampled DFT matrices, $\mathbf{S} \in \mathbb{C}^{N_c N_r \times N_r}$ is a stack of diagonal matrices containing the sensitivity maps, $\mathbf{C}_{\mathrm{H}} \in \mathbb{R}^{N_r \times N_r}$ and $\mathbf{C}_{\mathrm{V}} \in \mathbb{R}^{N_r \times N_r}$ denote finite differences in the horizontal and vertical directions (equivalent to anisotropic TV), and $\hat{x}$ is the reconstructed image. The regularization parameter $\lambda > 0$ balances adherence to noisy data and prior assumptions that inform choice of regularizers. Section III-D extends (1) to include discrete wavelet transforms. Although we focus on 2D imaging for notational simplicity, the methods generalize readily to 3D problems by adding another term to (1).

We choose $\mathbf{C}_{\mathrm{H}}$ and $\mathbf{C}_{\mathrm{V}}$ to have non-periodic boundary conditions. This is in contrast to many proposed algorithms that use finite differences with periodic boundary conditions, such as AL-P2 [4], RecPF [8], and recMRI [9]. Periodic boundary conditions have been used for computational convenience, despite being physically unnatural. Non-periodic boundary conditions are preferable in most applications [10]–[12]. Differences across the boundaries of medical images do not provide useful information for reconstruction. Furthermore, penalizing differences across spatial boundaries may degrade image quality when the region of interest extends to the boundary. For example, a coronal abdominal image depicts different anatomy at the top and bottom boundaries. Thus this paper focuses on developing methods that are fast yet suitable for non-periodic boundary conditions.

## III. Variable Splitting Methods

The non-differentiable $\ell_1$ norms make (1) a challenging optimization problem. Variable splitting methods like [4], [13], [14] are useful for such problems. Here, we reformulate (1) in an equivalent constrained form using the following novel variable splitting scheme:

$$\hat{\underline{u}} = \operatorname*{argmin}_{\underline{u}} f(\underline{u}) \quad (2)$$

$$\begin{aligned} f(\underline{u}) &= \frac{1}{2} \|y - \mathbf{F} u_2\|^2 + \lambda \|u_0\|_1 + \lambda \|u_1\|_1 \\ \text{s.t. } u_0 &= \mathbf{C}_{\mathrm{H}} x, \qquad u_1 = \mathbf{C}_{\mathrm{V}} u_3, \\ u_2 &= \frac{1}{2}\mathbf{S} u_3 + \frac{1}{2}\mathbf{S} x, \qquad u_3 = x. \end{aligned} \quad (3)$$

For convenience, we group $x$ and the auxiliary variables $u_0, \ldots, u_3$ into one column vector: $\underline{u} \triangleq (u_0, u_1, u_2, u_3, x)$.

This variable splitting scheme intentionally separates the horizontal and vertical finite differences operators, applying them to different auxiliary variables. This separates the tridiagonal structures in the Hessians resulting from AL, permitting decoupled, computationally efficient variable updates further detailed in Section III-C. If the finite difference matrices were combined in the same auxiliary variable as in AL-P2 [4], the resulting Hessians would have a block-tridiagonal with tridiagonal blocks (BTTB) structure that cannot take advantage of an $\mathcal{O}(N_r)$ tridiagonal solver.

### A. Direct AL Approach: AL-Tridiag

Here we detail the algorithm resulting from directly applying AL with alternating minimization to (2). The resulting algorithm, AL-tridiag, does not satisfy the sufficient conditions for convergence in [7], so currently it lacks convergence guarantees. However, it has worked well in all of our experiments, so it is possible that future generalized convergence proofs could be applicable.

We rewrite constrained cost function (2) with a matrix constraint as follows:

$$\min_{\underline{u}} f(\underline{u}) \text{ s.t. } \mathbf{P}\underline{u} = 0 \quad (4)$$

$$\mathbf{P} \triangleq \begin{bmatrix} -\mathbf{I}_{N_r} & 0 & 0 & 0 & \mathbf{C}_{\mathrm{H}} \\ 0 & -\mathbf{I}_{N_r} & 0 & \mathbf{C}_{\mathrm{V}} & 0 \\ 0 & 0 & -\mathbf{I}_{N_r N_c} & \frac{1}{2}\mathbf{S} & \frac{1}{2}\mathbf{S} \\ 0 & 0 & 0 & -\mathbf{I}_{N_r} & \mathbf{I}_{N_r} \end{bmatrix}.$$

The constraint $\mathbf{P}\underline{u} = 0$ enforces (3). The augmented Lagrangian corresponding to (4) is:

$$\mathcal{L}(\underline{u}, \underline{\eta}; \mathbf{M}) = f(\underline{u}) + \frac{1}{2} \left\|\mathbf{P}\underline{u} - \underline{\eta}\right\|_{\mathbf{M}}^2. \quad (5)$$

This formulation introduces dual variables, stacked in a vector: $\underline{\eta} \triangleq (\eta_0, \ldots, \eta_3) \in \mathbb{C}^{(3+N_c)N_r}$. Matrix $\mathbf{M}$ is a positive definite diagonal matrix, consisting of user-selected AL penalty parameters. The diagonal block corresponding to the $i$th segment of $\underline{u}$ is denoted $\mathbf{M}_i$. The choice of $\mathbf{M}$ does not affect the final solution of (5), but it can affect the convergence rate of the resulting algorithm. For many multiplier methods, using positive penalty parameters guarantees convergence to the solution of the original problem that does not involve the penalty parameters [15]. Section III-G discusses heuristics for selecting $\mathbf{M}$.

Ideally, an AL method would update block variables $\underline{u}$ and $\underline{\eta}$ at iteration $n+1$ as follows:

$$\underline{u}^{(n+1)} = \operatorname*{argmin}_{\underline{u}} \mathcal{L}\left(\underline{u}, \underline{\eta}^{(n)}\right) \tag{6}$$

$$\underline{\eta}^{(n+1)} = \underline{\eta}^{(n)} - \mathbf{P}\underline{u}^{(n+1)}. \tag{7}$$

Our proposed AL-tridiag algorithm uses alternating minimization across $u_0, \ldots, u_3, x$ to descend the AL term in (6). Section III-C and the supplement describe the $\underline{u}$ variable updates in more detail. Due to the variable splitting design of (3), each variable update has a direct, closed-form solution with an efficient implementation, e.g., by FFTs or a parallelizable tridiagonal solver.

## *B. ADMM Equivalence for ADMM-Tridiag*

To design a minimization algorithm with convergence guarantees, we reformulate (2) as an instance of the generalized Alternating Direction Method of Multipliers (ADMM) [7], [16]. This formulation allows us to invoke the convergence proof in [7] for our second proposed algorithm, ADMM-tridiag.

We first express the matrix constraint $\mathbf{P}$ in (4) as a product of two matrices, $\mathbf{P} = \mathbf{BA}$, and incorporate the left matrix $\mathbf{B}$ into the following convex cost function:

$$\min_{\underline{u},\underline{v}} \; f(\underline{u}) + g(\underline{v}) \text{ s.t. } \mathbf{A}\underline{u} = \underline{v} \tag{8}$$

$$g(\underline{v}) = \begin{cases} 0, & \mathbf{B}\underline{v} = 0 \\ +\infty, & \mathbf{B}\underline{v} \neq 0. \end{cases} \tag{9}$$

For (8) to satisfy the sufficient conditions for convergence of ADMM in [7], $\mathbf{A}$ must have full rank. For our proposed algorithm, ADMM-tridiag, we design $\mathbf{B}$ and $\mathbf{A}$ as follows:

$$\mathbf{BA}\underline{u} = 0$$

$$\mathbf{B} \triangleq \begin{bmatrix} \mathbf{I}_{N_r} & 0 & 0 & 0 & 0 \\ 0 & \mathbf{I}_{N_r} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{I}_{N_r N_c} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{I}_{N_r} & \mathbf{I}_{N_r} \end{bmatrix}$$

$$\mathbf{A} \triangleq \begin{bmatrix} -\mathbf{I}_{N_r} & 0 & 0 & 0 & \mathbf{C}_\mathrm{H} \\ 0 & -\mathbf{I}_{N_r} & 0 & \mathbf{C}_\mathrm{V} & 0 \\ 0 & 0 & -\mathbf{I}_{N_r N_c} & \frac{1}{2}\mathbf{S} & \frac{1}{2}\mathbf{S} \\ 0 & 0 & 0 & -\mathbf{I}_{N_r} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{I}_{N_r} \end{bmatrix}. \tag{10}$$

For convenience, we describe $\underline{v}$ in terms of its block elements: $\underline{v} \triangleq (v_0, \cdots, v_4)$. The full rank of $\mathbf{A}$, combined with the following alternating minimization framework, satisfies the convergence conditions for ADMM [7]. Thus, (8)–(10) describe an instance of ADMM and guarantees convergence to a minimizer of (8). To handle the constraints of (8), ADMM-tridiag uses the following augmented Lagrangian, similar to Section III-A:

$$\mathcal{L}\left(\underline{u}, \underline{v}, \underline{\eta}; \mathbf{M}\right) = f(\underline{u}) + g(\underline{v}) + \frac{1}{2}\left\|\mathbf{A}\underline{u} - \underline{v} - \underline{\eta}\right\|_{\mathbf{M}}^2. \tag{11}$$

The dual variables $\underline{\eta} \triangleq (\eta_0, \ldots, \eta_4) \in \mathbb{C}^{N_\mathrm{r}(4+N_\mathrm{c})}$ have an additional block-element compared to the AL case. For convenience, we reuse the notation for the dual variables for both algorithms. The matrix $\mathbf{M}$ for ADMM-tridiag is also larger than in the AL case, consisting of blocks $\{\mathbf{M}_i\}_{i=0,\ldots,4}$.

Following [15], [7], ADMM-tridiag alternates between updating $\underline{u}$, $\underline{v}$, and the dual variables $\underline{\eta}$. Examining the $\underline{v}$ update and recalling (9), we see that the role of $\underline{v}$ simplifies greatly by considering the feasible set of $\underline{v}$, $\Omega \triangleq \{\underline{v} \in \mathbb{C}^{(4+N_\mathrm{c})N_\mathrm{r}} : \mathbf{B}\underline{v} = 0\}$, resulting in the following alternating updates:

$$\underline{u}^{(n+1)} \underset{\epsilon_n}{\approx} \operatorname*{argmin}_{\underline{u}} f(\underline{u}) + \frac{1}{2}\left\|\mathbf{A}\underline{u} - \underline{v}^{(n)} - \underline{\eta}^{(n)}\right\|_{\mathbf{M}}^2 \tag{12}$$

$$\underline{v}^{(n+1)} \underset{\zeta_n}{\approx} \operatorname*{argmin}_{\underline{v}\in\Omega} \frac{1}{2}\left\|\mathbf{A}\underline{u}^{(n+1)} - \underline{v} - \underline{\eta}^{(n)}\right\|_{\mathbf{M}}^2 \tag{13}$$

$$\underline{\eta}^{(n+1)} = \underline{\eta}^{(n)} - \left(\mathbf{A}\underline{u}^{(n+1)} - \underline{v}^{(n+1)}\right). \tag{14}$$

Here, we allow for some inexactness in the updates of $\underline{u}$ and $\underline{v}$ at each iteration, $\{\epsilon_n\}$ and $\{\zeta_n\}$, respectively. ADMM convergence holds if the inexactness sequences are summable [3, Theorem 8] [7, Theorem 2.1].

The update for $\underline{v}$ is simple and exact to machine precision. Due to the simple structure of $\mathbf{B}$, we have $v_0, v_1, v_2 = 0$ and $v_4 = -v_3$. The entire update for $\underline{v}$ reduces to a quadratic minimization problem for the auxiliary variable $v_3$ of size $N_r$ with simple, closed-form solution:

$$\begin{aligned} v_3^{(n+1)} = {} & (\mathbf{M}_3 + \mathbf{M}_4)^{-1}\left[\mathbf{M}_3\left(-u_3^{(n+1)} - \eta_3^{(n)}\right)\right. \\ & \left. + \mathbf{M}_4\left(-x^{(n+1)} + \eta_4^{(n)}\right)\right]. \end{aligned} \tag{15}$$

For the joint minimization of $\underline{u}$ in (12), ADMM-tridiag employs alternating minimization, as detailed in Section III-C. If one round of alternating minimization sufficiently approximates the joint minimizer of (12), the iterates of ADMM-tridiag, $\{x^{(n)}\}$, converge to a minimizer of (1), per [7]. This alternating minimization approach is common in other ADMM methods, such as [10], [12].

Compared to AL-tridiag in Section III-A, the direct AL approach to (3), ADMM-tridiag involves one additional variable split. An alternative way to understand the effect of (10) is to describe this ADMM algorithm as the result of applying AL to the following variable splitting scheme for (1):

$$\min_{\underline{u}} \frac{1}{2}\|y - \mathbf{F}u_2\|^2 + \lambda\|u_0\|_1 + \lambda\|u_1\|_1 \tag{16}$$

$$\text{s.t. } u_0 = \mathbf{C}_\mathrm{H}x, \quad u_2 = \frac{1}{2}\mathbf{S}u_3 + \frac{1}{2}\mathbf{S}x,$$

$$u_1 = \mathbf{C}_\mathrm{V}u_3, \quad u_3 = -v_3, \quad v_3 = -x \tag{17}$$

This formulation indirectly enforces $u_3 = x$ through additional auxiliary variable $v_3$. However, written in this form it is less clear that the full-rank condition of [7] is satisfied, whereas that is clear in (8)–(10).

## *C. Variable Updates With Parallelizable Tridiagonal Solvers*

This section describes the block-variable updates for alternating minimization of (12). These variable updates are very similar for both proposed methods, AL-tridiag and ADMM-tridiag, differing only in tuning parameter indices and use of $\underline{v}$ and $\underline{\eta}$ on the right-hand side of these updates. For brevity, we give only the alternating minimization updates for ADMM-tridiag in (12). The variable updates for AL-tridiag in (6) are very similar as shown in Algorithm 2 of the Supplement.[1]

First, we consider the special case where some blocks of $\mathbf{M}$ are constant diagonal matrices. We leave $\mathbf{M}_3$ and $\mathbf{M}_4$ as general positive definite diagonal matrices for reasons explained in Section III-G. Letting $\mathbf{M}_i \triangleq \mu_i \mathbf{I},\ \mu_i > 0$ for $i = 0, 1, 2$ and leveraging the constraint that $v_4 = -v_3$, (12) expands to:

$$\begin{aligned}\underline{u}^{(n+1)} = \underset{x,u_0,\ldots,u_3}{\operatorname{argmin}}\ & \frac{1}{2}\|y - \mathbf{F}u_2\|^2 + \lambda\|u_0\|_1 + \lambda\|u_1\|_1 \\ & + \frac{\mu_0}{2}\left\|-u_0 + \mathbf{C}_{\mathrm{H}}x - \eta_0^{(n)}\right\|^2 \\ & + \frac{\mu_1}{2}\left\|-u_1 + \mathbf{C}_{\mathrm{V}}u_3 - \eta_1^{(n)}\right\|^2 \\ & + \frac{\mu_2}{2}\left\|-u_2 + \frac{1}{2}\mathbf{S}u_3 + \frac{1}{2}\mathbf{S}x - \eta_2^{(n)}\right\|^2 \\ & + \frac{1}{2}\left\|-u_3 - v_3^{(n)} - \eta_3^{(n)}\right\|^2_{\mathbf{M}_3} \\ & + \frac{1}{2}\left\|x + v_3^{(n)} - \eta_4^{(n)}\right\|^2_{\mathbf{M}_4}. \end{aligned} \tag{18}$$

ADMM-tridiag uses alternating minimization to update the blocks of $\underline{u}$. The resulting variable updates are:

$$u_0^{(n+1)} = \operatorname{soft}\left(\mathbf{C}_{\mathrm{H}}x^{(n)} - \eta_0^{(n)}, \frac{\lambda}{\mu_0}\right) \tag{19}$$

$$u_1^{(n+1)} = \operatorname{soft}\left(\mathbf{C}_{\mathrm{V}}u_3^{(n)} - \eta_1^{(n)}, \frac{\lambda}{\mu_1}\right) \tag{20}$$

$$u_2^{(n+1)} = \mathbf{H}_2^{-1}\left(\mathbf{F}'y + \mu_2\left(\frac{1}{2}\mathbf{S}u_3^{(n)} + \frac{1}{2}\mathbf{S}x^{(n)} - \eta_2^{(n)}\right)\right) \tag{21}$$

$$\begin{aligned}u_3^{(n+1)} = \mathbf{H}_3^{-1}\Big(&\mu_1\mathbf{C}_{\mathrm{V}}'\left(u_1^{(n+1)} + \eta_1^{(n)}\right) \\ & + \frac{\mu_2}{2}\mathbf{S}'\left(u_2^{(n+1)} - \frac{1}{2}\mathbf{S}x^{(n)} + \eta_2^{(n)}\right) \\ & + \mathbf{M}_3\left(-v_3^{(n)} - \eta_3^{(n)}\right)\Big)\end{aligned} \tag{22}$$

$$\begin{aligned}x^{(n+1)} = \mathbf{H}_{\mathrm{x}}^{-1}\Big(&\mu_0\mathbf{C}_{\mathrm{H}}'\left(u_0^{(n+1)} + \eta_0^{(n)}\right) \\ & + \frac{\mu_2}{2}\mathbf{S}'\left(u_2^{(n+1)} - \frac{1}{2}\mathbf{S}u_3^{(n+1)} + \eta_2^{(n)}\right) \\ & + \mathbf{M}_4\left(-v_3^{(n)} + \eta_4^{(n)}\right)\Big).\end{aligned} \tag{23}$$

[1]Supplementary material available in the supplementary files/multimedia tab.

**Algorithm 1:** ADMM-Tridiag.

```
 1: Initialize $x$ to square root of sum-of-squares (SoS) of
        zero-filled iFFT images.
 2: Initialize $u_0 = \mathbf{C}_{\mathrm{H}}x$, $u_1 = \mathbf{C}_{\mathrm{V}}u_3$, $u_3 = x$,
        $u_2 = \frac{1}{2}\mathbf{S}u_3 + \frac{1}{2}\mathbf{S}x$, $v_3 = -x$, $\underline{\eta} = \underline{0}$
 3: for $n \le$ total iterations do
 4:     Compute $u_0^{(n+1)}$ via soft-thresholding (19)
 5:     Compute $u_1^{(n+1)}$ via soft-thresholding (20)
 6:     Compute $u_2^{(n+1)}$ via FFTs (21)
 7:     Compute $u_3^{(n+1)}$ via tridiagonal solver (22)
 8:     Compute $x^{(n+1)}$ via tridiagonal solver (23)
 9:     Compute $v_3^{(n+1)}$ via (15)
10:     Compute $\underline{\eta}^{(n+1)}$ via (14)
11: end for
```

The soft-thresholding operator performs element-wise shrinkage for the $\ell_1$ norm using a given threshold $\tau$: $\operatorname{soft}(x,\tau) \triangleq \operatorname{sign}(x)\max(|x| - \tau, 0)$. Thus, (19) and (20) provide simple, direct solutions for updating $u_0$ and $u_1$. The bulk of the computation is "inverting" the following Hessians:

$$\mathbf{H}_2 \triangleq \mathbf{F}'\mathbf{F} + \mu_2\mathbf{I} = \mathbf{Q}\left(\mathbf{\Lambda}_F + \mu_2\mathbf{I}\right)\mathbf{Q}' \tag{24}$$

$$\mathbf{H}_3 \triangleq \mu_1\mathbf{C}_{\mathrm{V}}'\mathbf{C}_{\mathrm{V}} + \frac{\mu_2}{4}\mathbf{S}'\mathbf{S} + \mathbf{M}_3 \tag{25}$$

$$\mathbf{H}_{\mathrm{x}} \triangleq \mu_0\mathbf{C}_{\mathrm{H}}'\mathbf{C}_{\mathrm{H}} + \frac{\mu_2}{4}\mathbf{S}'\mathbf{S} + \mathbf{M}_4. \tag{26}$$

Due to Cartesian undersampling, $\mathbf{H}_2$ is diagonalizable via $N_c$ FFTs, each of which operate efficiently in $\mathcal{O}(N_r \log N_r)$ time. The multi-coil FFT operator is denoted $\mathbf{Q}$. The $x$ update in (23) uses $\mathbf{H}_{\mathrm{x}}$ in (26), a block diagonal matrix with tridiagonal blocks (BDTB) that can be "inverted" in $\mathcal{O}(N_r)$ time via Gaussian elimination. Because it is block diagonal, we parallelize this variable update over the $N_y$ independent blocks. We reformulate the $u_3$ update as another instance of the same BDTB inverse problem through permutation and solve (22) using a tridiagonal solver parallelized over $N_x$ blocks. Each variable update is exact and easy to implement.

We designed the proposed variable splitting in (2) to enable these efficient variable updates. The separation of horizontal and vertical finite differences into $u_0$ and $u_1$ allows for BDTB structures in $\mathbf{H}_3$ and $\mathbf{H}_{\mathrm{x}}$. If the finite difference matrices were combined in the same auxiliary variable as in AL-P2 [4], the resulting Hessian would have a block-tridiagonal with tridiagonal blocks (BTTB) structure and the associated variable update would require a more computationally costly solution [17], [18].

Algorithm 1 summarizes the overall procedure for ADMM-tridiag. All of the variable updates are done in place, so the memory requirements for storing $x, u_0, \ldots, u_3, v_3$, and $\underline{\eta}$ are $8N_{\mathrm{r}}(4 + N_{\mathrm{c}})$ bytes for $N_{\mathrm{r}}(4 + N_{\mathrm{c}})$ complex single-precision values. For AL-tridiag, the $\underline{u}$ updates are very similar.

## *D. Regularization With Finite Differences and Wavelets*

The variable splitting scheme in (17) readily generalizes to combinations of finite difference and orthonormal wavelet regularization. Let $\mathbf{W}$ be an orthonormal wavelet transform (e.g., Haar wavelets). Then the following combined TV / wavelet sparsity cost-function can be manipulated to resemble (1):

$$\frac{1}{2}\|y-\mathbf{FS}x\|_2^2+\lambda_1\|\mathbf{C}_{\mathrm{H}}x\|_1 + \lambda_1\|\mathbf{C}_{\mathrm{V}}x\|_1+\lambda_2\|\mathbf{W}x\|_1 \tag{27}$$

$$=\frac{1}{2}\|y-\mathbf{FS}x\|_2^2+\lambda_1\left\|\tilde{\mathbf{C}}_{\mathrm{H}}x\right\|_1+\lambda_1\left\|\tilde{\mathbf{C}}_{\mathrm{V}}x\right\|_1 \tag{28}$$

$$\tilde{\mathbf{C}}_{\mathrm{H}}\triangleq\begin{bmatrix}\mathbf{C}_{\mathrm{H}}\\ \frac{\alpha_w\lambda_2}{\lambda_1}\mathbf{W}\end{bmatrix};\quad \tilde{\mathbf{C}}_{\mathrm{V}}\triangleq\begin{bmatrix}\mathbf{C}_{\mathrm{V}}\\ \frac{(1-\alpha_w)\lambda_2}{\lambda_1}\mathbf{W}\end{bmatrix}. \tag{29}$$

An additional spatial regularization parameter, $\lambda_2$, controls the weight of the wavelet regularization. Due to the orthonormality of $\mathbf{W}$, we can use this regularizer for both AL-tridiag and ADMM-tridiag, with only minor changes to variable updates (19)–(23). For ADMM-tridiag, the only Hessians affected by introducing wavelets are:

$$\tilde{\mathbf{H}}_3\triangleq\mu_1\mathbf{C}_{\mathrm{V}}'\mathbf{C}_{\mathrm{V}}+\frac{\mu_2}{4}\mathbf{S}'\mathbf{S}+\tilde{\mathbf{M}}_3 \tag{30}$$

$$\tilde{\mathbf{H}}_{\mathrm{x}}\triangleq\mu_0\mathbf{C}_{\mathrm{H}}'\mathbf{C}_{\mathrm{H}}+\frac{\mu_2}{4}\mathbf{S}'\mathbf{S}+\tilde{\mathbf{M}}_4, \tag{31}$$

with positive definite matrices $\tilde{\mathbf{M}}_3=\mathbf{M}_3+\frac{(1-\alpha_w)^2\lambda_2^2}{\lambda_1^2}\mathbf{I}$ and $\tilde{\mathbf{M}}_4=\mathbf{M}_4+\frac{\alpha_w^2\lambda_2^2}{\lambda_1^2}\mathbf{I}$. Hessians $\tilde{\mathbf{H}}_3$ and $\tilde{\mathbf{H}}_{\mathrm{x}}$ are still BDTB, and they can be "inverted" efficiently with a parallelizable tridiagonal solver routine. This wavelet-inclusive variation is featured in experimental results in Section IV-C.

## *E. Special Case: Image Inpainting*

To highlight the value of non-periodic boundary conditions, we examine a specific application of the proposed variable splitting scheme in (2), namely inpainting. Image inpainting fills in image data that is lost or corrupted. Many image inpainting, deblurring, and denoising methods use finite difference regularizers like anisotropic total variation (TV) [11], [19], [20].

Let $\mathbf{D}$ be a binary, diagonal matrix whose nonzero entries denote the set of indices in the inpainting domain. Setting $\mathbf{FS}=\mathbf{D}$ in the CS-SENSE-MRI cost function (1) leads to the following simpler image inpainting problem:

$$\hat{x}=\underset{x}{\operatorname{argmin}}\,\frac{1}{2}\|y-\mathbf{D}x\|_2^2+\lambda\left\|\tilde{\mathbf{C}}_{\mathrm{H}}x\right\|_1+\lambda\left\|\tilde{\mathbf{C}}_{\mathrm{V}}x\right\|_1, \tag{32}$$

where $\tilde{\mathbf{C}}_{\mathrm{H}}$ and $\tilde{\mathbf{C}}_{\mathrm{V}}$ are defined in (29).

For inpainting, we simplify the VS scheme developed for SENSE MRI in (2) to:

$$\underline{\hat{u}}=\underset{\underline{u}}{\operatorname{argmin}}\,f(\underline{u}) \tag{33}$$

$$f(\underline{u})=\frac{1}{2}\left\|y-\mathbf{D}\left(\frac{1}{2}u_2+\frac{1}{2}x\right)\right\|^2+\lambda\|u_0\|_1+\lambda\|u_1\|_1$$

$$\text{s.t. } u_0=\tilde{\mathbf{C}}_{\mathrm{H}}x,\ u_1=\tilde{\mathbf{C}}_{\mathrm{V}}u_2,\ u_2=x.$$

The resulting VS algorithm is a simplification of AL-tridiag, which we denote AL-tridiag-inpaint. Due to the entirely diagonal system matrix, the variable updates consist only of shrinkage and tridiagonal solver updates, eliminating the need for any FFT-based updates. Similarly, we can generalize ADMM-tridiag to the inpainting problem by applying an extra variable splitting, resulting in an additional quadratic minimization problem in each iteration. Section IV-C illustrates the effect of non-periodic boundary conditions for noisy inpainting.

## *F. Comparison With AL-P2*

We compare ADMM-tridiag with AL-P2 [4], a fast VS scheme designed for CS-SENSE-MRI. The original AL-P2 version in [4] used periodic boundary conditions, whereas here we modify it for the non-periodic conditions of (1) and call the modified algorithm AL-P2-NC. The suffix "NC" refers to the non-circulant Hessian we describe in this section. To define AL-P2-NC, we stack the finite difference matrices into a tall matrix, $\mathbf{R}\triangleq[\mathbf{C}_{\mathrm{H}};\mathbf{C}_{\mathrm{V}}]$. Applying the AL-P2 variable splitting scheme to (1) yields the following constrained cost function:

$$\min_{x,u,v,z}\ \|y-\mathbf{F}u\|^2+\lambda\|v\|_1 \tag{34}$$

$$\text{s.t. } u=\mathbf{S}x,\ v=\mathbf{R}u,\ z=x. \tag{35}$$

Applying the augmented Lagrangian to the constrained cost function (34) and using alternating minimization results in variable updates like in [4]:

$$x^{(n+1)}=\mathbf{H}_{\mathrm{x}}^{-1}\left(\mathbf{F}'y+\mu_u\left(\mathbf{S}x^{(n)}+\eta_u{}^{(n)}\right)\right) \tag{36}$$

$$u^{(n+1)}=\mathbf{H}_{\mathrm{u}}^{-1}\left(\mu_u\mathbf{S}'\left(u^{(n)}-\eta_u{}^{(n)}\right)+\mu_z\left(z^{(n)}-\eta_z{}^{(n)}\right)\right) \tag{37}$$

$$v^{(n+1)}=\operatorname{soft}\left(\mathbf{R}z^{(n)}+\eta_v{}^{(n)},\frac{\lambda}{\mu_u}\right) \tag{38}$$

$$z^{(n+1)}=\mathbf{H}_{\mathrm{z}}^{-1}\left(\mu_v\mathbf{R}'\left(v^{(n+1)}-\eta_v{}^{(n)}\right)+\mu_z\left(x^{(n+1)}+\eta_z{}^{(n)}\right)\right). \tag{39}$$

The scalar AL penalty parameters, $\mu_u,\mu_v,\mu_z>0$, do not affect the final solution but do affect convergence rate. The Hessians for $x$ and $u$ are simple to invert and are the same as in ADMM-tridiag:

$$\mathbf{H}_{\mathrm{x}}=\mu_u\mathbf{S}'\mathbf{S}+\mu_z\mathbf{I} \tag{40}$$

$$\mathbf{H}_{\mathrm{u}}=\mathbf{F}'\mathbf{F}+\mu_u\mathbf{I}=\mathbf{Q}(\Lambda_F+\mu_u\mathbf{I})\mathbf{Q}'. \tag{41}$$

The Hessian $\mathbf{H}_{\mathrm{z}}$ for the $z$ update is BTTB (non-circulant) as follows:

$$\mathbf{H}_{\mathrm{z}}=\mu_v\mathbf{R}'\mathbf{R}+\mu_z\mathbf{I}, \tag{42}$$

for which there is no $\mathcal{O}(N_{\mathrm{r}})$ solver. To "invert" $\mathbf{H}_{\mathrm{z}}$ for (39) we applied one iteration of preconditioned gradient descent with a circulant preconditioner. The resulting computation per iteration is essentially identical to that of the original AL-P2 with periodic boundary conditions in [4].

### G. Parameter Selection

For all of the experiments shown in Section IV and in the supplement, we manually chose the spatial regularization parameter, $\lambda$, so that the converged image $x^{(\infty)}$ resembled the true image (for inpainting and CS-SENSE-MRI simulations in the supplement) or the fully-sampled body coil image (for *in vivo* data).

As with other AL algorithms, the tuning parameters $\mathbf{M}_0, \ldots, \mathbf{M}_4 \succeq 0$ do not affect the solution $\hat{x}$, but they can greatly affect the convergence rate. To facilitate comparison with AL-P2-NC, we chose the AL tuning parameters of AL-P2-NC based on the guidelines provided in [4, Eqns. (40)–(41)] as follows:

$$\begin{aligned}\kappa\left(\mathbf{H}_{\mathrm{u}}\right) &= 24;\ \kappa\left(\mathbf{H}_{\mathrm{z}}\right) = 12;\\ \kappa\left(\mathbf{H}_{\mathrm{x}}\right) &= 0.9\,\kappa(\mathbf{S}'\mathbf{S}).\end{aligned} \tag{43}$$

For consistency, we selected the AL tuning parameters of AL-tridiag and ADMM-tridiag with a similar heuristic strategy. For the Hessian of the multi-coil FFT step (21) we selected its associated parameter, $\mu_2$, such that $\kappa\left(\mathbf{H}_2\right) = 24$, exactly as in AL-P2. Our proposed variable splitting (2) results in Hessians in which the regularizer is combined with the sensitivity encoding, so the remaining AL-P2 tuning rules are inapplicable. Instead, we selected the remaining tuning parameters ($\mu_0$, $\mu_1$, $\mathbf{M}_3$, and $\mathbf{M}_4$) to enforce $\kappa\left(\mathbf{H}_3\right) = \kappa\left(\mathbf{H}_{\mathrm{x}}\right) = 12$. Because our remaining Hessians $\mathbf{H}_3$ and $\mathbf{H}_{\mathrm{x}}$ partially consist of $\mathbf{C}_{\mathrm{V}}'\mathbf{C}_{\mathrm{V}}$ and $\mathbf{C}_{\mathrm{H}}'\mathbf{C}_{\mathrm{H}}$, respectively, we choose to enforce the condition number of 12 used for $\mathbf{H}_{\mathrm{z}}$ of AL-P2, which is characterized by the periodic boundary finite differences. We choose to apply this heuristic over the alternative guideline based on $\kappa(\mathbf{S}'\mathbf{S})$ because our choices of $\mathbf{M}_3$ and $\mathbf{M}_4$ in (49)–(50) below make $\mathbf{H}_3$ and $\mathbf{H}_{\mathrm{x}}$ approximately circulant, but far from diagonal.

First we chose the parameters that interact with the thresholding steps, $\mu_0$ and $\mu_1$, based on the maximum value of the initial image, $x_{\max}$, and spatial regularization parameter, $\lambda$:

$$\mu_0 = \mu_1 = \frac{\lambda}{0.02\, x_{\max}}. \tag{44}$$

This sets the threshold of the shrinkage step in (19) and (20) at 2% of the maximum initial image value. This threshold worked well for the noise level of the following simulated and *in vivo* experiments. Recalling the BDTB structures of $\mathbf{H}_3$ (25) and $\mathbf{H}_{\mathrm{x}}$ (26), we designed $\mathbf{M}_3$ and $\mathbf{M}_4$ to enforce the following conditions for scalar $c_3, c_4 > 0$:

$$c_3\mathbf{I} = \frac{\mu_2}{4}\mathbf{S}'\mathbf{S} + \mathbf{M}_3 \tag{45}$$

$$c_4\mathbf{I} = \frac{\mu_2}{4}\mathbf{S}'\mathbf{S} + \mathbf{M}_4. \tag{46}$$

The constant diagonal term results from allowing a spatially varying $\mathbf{M}_3$ and $\mathbf{M}_4$. Therefore, $\mathbf{M}_3$ is higher in regions where the sum-of-squares (SoS) of the sensitivity maps is low and vice versa. Intuitively this results in stronger enforcement of the $u_3 = -v_3$ and $x = -v_3$ constraints in spatial regions where the $u_2 = \frac{1}{2}\mathbf{S}u_3 + \frac{1}{2}\mathbf{S}x$ constraint provides less information. We use [21] for an analytical solution for maximum and minimum eigenvalues of $\mathbf{C}_{\mathrm{H}}'\mathbf{C}_{\mathrm{H}}$ and $\mathbf{C}_{\mathrm{V}}'\mathbf{C}_{\mathrm{V}}$. Due to (45)–(46), $\mathbf{H}_3$ and $\mathbf{H}_{\mathrm{x}}$ are approximately circulant, and eigenvalue analysis of $\mathbf{H}_3$ and $\mathbf{H}_{\mathrm{x}}$ becomes simple:

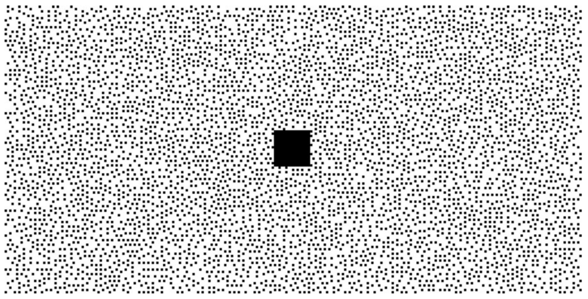


Fig. 1. Retrospective Poisson-disk-based undersampling pattern used for *in vivo* experiments, with reduction factor of 6 and fully sampled central 16 × 16 phase-encodes.

$$\kappa\left(\mathbf{H}_3\right) = \frac{\mu_1\lambda_{\max}\left(\mathbf{C}_{\mathrm{V}}'\mathbf{C}_{\mathrm{V}}\right) + c_3}{\mu_1\lambda_{\min}\left(\mathbf{C}_{\mathrm{V}}'\mathbf{C}_{\mathrm{V}}\right) + c_3} \tag{47}$$

$$\kappa\left(\mathbf{H}_{\mathrm{x}}\right) = \frac{\mu_0\lambda_{\max}\left(\mathbf{C}_{\mathrm{H}}'\mathbf{C}_{\mathrm{H}}\right) + c_4}{\mu_0\lambda_{\min}\left(\mathbf{C}_{\mathrm{H}}'\mathbf{C}_{\mathrm{H}}\right) + c_4}. \tag{48}$$

Let $\hat{c}_3$ and $\hat{c}_x$ be the respective solutions for (47) and (48) for $\kappa\left(\mathbf{H}_3\right) = \kappa\left(\mathbf{H}_{\mathrm{x}}\right) = 12$. Then the values for $\mathbf{M}_3$ and $\mathbf{M}_4$ are as follows:

$$\mathbf{M}_3 \triangleq \max\left(\hat{c}_3\mathbf{I} - \frac{\mu_2}{4}\mathbf{S}'\mathbf{S},\ 10^{-3}\right) \tag{49}$$

$$\mathbf{M}_4 \triangleq \max\left(\hat{c}_4\mathbf{I} - \frac{\mu_2}{4}\mathbf{S}'\mathbf{S},\ 10^{-3}\right). \tag{50}$$

This choice of $\mathbf{M}_3$ and $\mathbf{M}_4$ is informed by the thresholding levels through (44) but also enforces the positive-definite condition for $\mathbf{M}_3$ and $\mathbf{M}_4$. We selected AL-tridiag parameters using the same procedure.

## IV. Results

### A. in vivo Experiment Setup

Following [4], we used a 3D in-vivo volunteer data set acquired from a GE 3T scanner ($T_R = 25$ ms, $T_E = 5.172$ ms, voxel size $= 1 \times 1.35 \times 1$ mm$^3$) with an 8-channel head coil. A corresponding body coil dataset was also acquired for sensitivity map estimation and image quality comparison. The fully-sampled data was 256 × 144 with 128 samples in the read-out direction along $z$. We performed the proposed reconstruction algorithms for two 2D axial slices from retrospectively undersampled data. To promote FFT efficiency, we resampled the data to correspond to an image size of 256 × 128. Fig. 1 shows the Poisson-disk-based undersampling pattern (reduction factor 6) in the $k_x$-$k_y$ phase-encode plane that included the central 16 × 16 phase-encodes, pictured in Fig. 1. This sampling corresponds to one slice of a 3D acquisition with frequency encoding in $k_z$.

We used the central 16 × 16 phase-encodes to generate low resolution images that were then used with the body coil image to estimate smooth sensitivity maps [12], shown in Fig. 9 of the Supplement. The sensitivity values in the air regions were

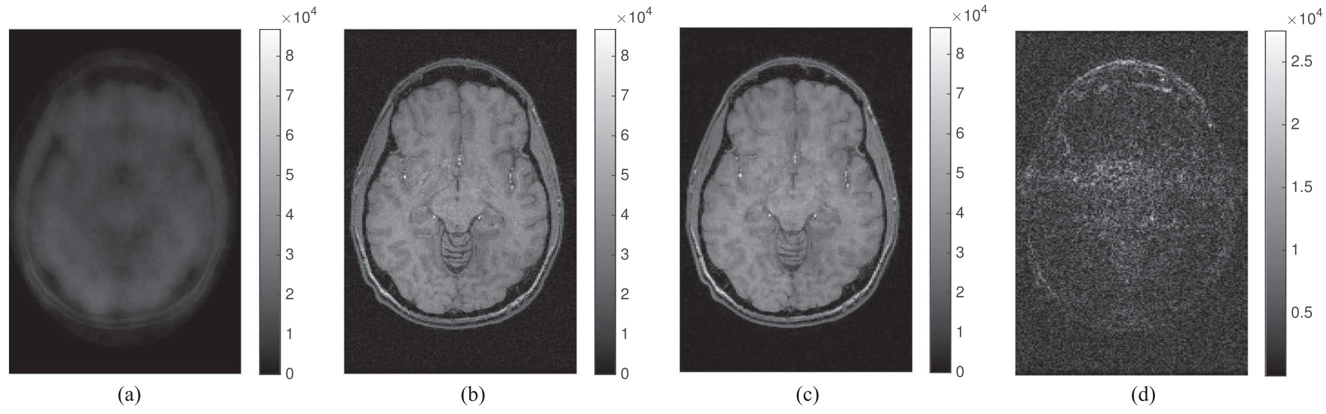


Fig. 2. Axial slice 38 of volunteer data (a) square root of sum-of-squares of the zero-filled iFFT coil images, used as an initial estimate for all algorithms; (b) separately acquired body coil image; (c) $x^{(\infty)}$ calculated by MFISTA; (d) difference between body coil image and $x^{(\infty)}$.

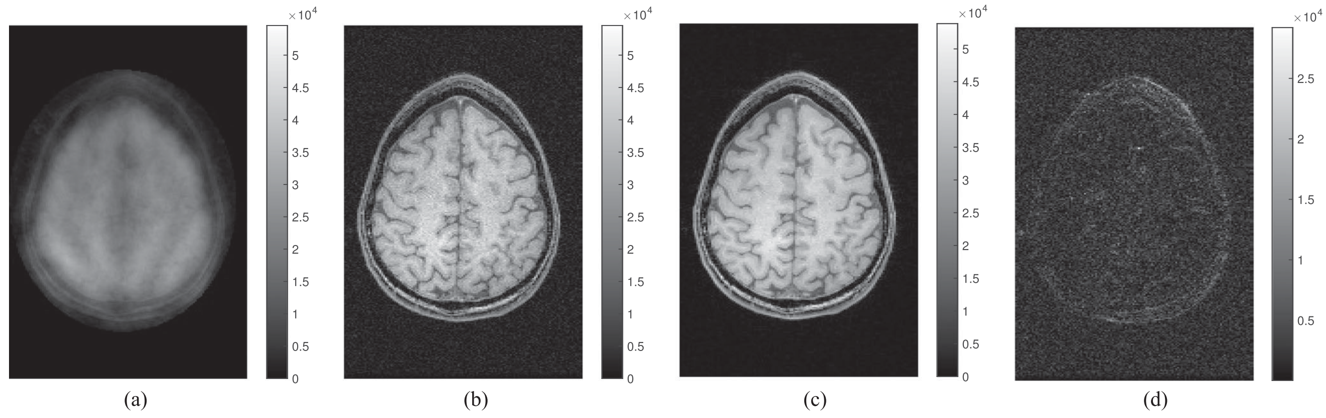


Fig. 3. Axial slice 90 of volunteer data (a) square root of sum-of-squares of the zero-filled iFFT coil images, used as an initial estimate for all algorithms; (b) separately acquired body coil image; (c) $x^{(\infty)}$ calculated by MFISTA; (d) difference between body coil image and $x^{(\infty)}$.

truncated in magnitude to control the maximum value in the sum-of-squares of the sensitivity maps and aid in tuning AL penalty parameters.

Computation was done on a Genuine Intel Xeon CPU E5-2680 with a 2.8 GHz 20 core machine with hyper-threading. The operating system was 64-bit Red Hat 6.7 running gcc version 4.4.7. All algorithms were implemented in Matlab version 8.6 using the image reconstruction toolbox [22], and all algorithms operated on single precision data. We performed parallelization of variable updates in (22) and (23) with a Pthreaded MEX function. The Pthreaded MEX function performed blockwise Gaussian elimination in parallel across each of the tridiagonal blocks of $\mathbf{H}_3$ and $\mathbf{H}_x$. We allocated 20 Pthreads for these operations.

The initial estimate for axial slice 38 was the square-root of the sum-of-squares of the zero-filled iFFT coil images, shown in Fig. 2. Fig. 2 also shows the qualitative similarity between the separately acquired body coil image and the MFISTA solution of (1).

We repeated the experiment with axial slice 90 from the same *in vivo* dataset and using the same sampling pattern. The sensitivity maps estimated for this axial slice are shown in Fig. 9 of the supplement. Fig. 3 shows the initial sum-of-squares estimate, the separately acquired body coil image, and MFISTA solution, and the difference between the body coil and MFISTA solution. For both slice 38 and 90, the converged MFISTA solution shows lower noise than the body coil image.

For both slice 38 and 90, both proposed algorithms reached the solution $x^{(\infty)}$ (to within machine precision), shown in Figs. 2(c) and 3(c). For brevity, Figs. 2(c) and 3(c) do not include $x^{(\infty)}$ for AL-tridiag or ADMM-tridiag, because they are visually indistinguishable from the MFISTA solution. The solutions $x^{(\infty)}$ were also visually similar to the fully sampled SENSE reconstruction without regularization, $\hat{x}_{\mathrm{SENSE}}$. Image comparisons are presented in the supplement.

## B. *Computation Speed Results for in vivo MRI Data*

We quantified the convergence rate of these algorithms using the normalized root mean squared distance (NRMSD) between a given iterate $x^{(n)}$ and the converged solution, $x^{(\infty)}$, in decibels:

$$\mathrm{NRMSD} \triangleq 20\log_{10}\left(\frac{\left\|x^{(n)}-x^{(\infty)}\right\|_2}{\left\|x^{(\infty)}\right\|_2}\right) \tag{51}$$

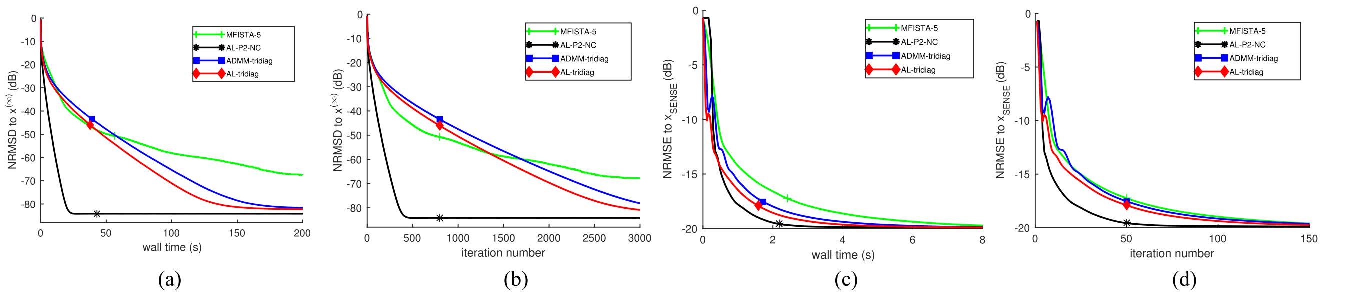


Fig. 4. Axial slice 38: (a, b) NRMSD comparison of AL-tridiag, ADMM-tridiag, MFISTA, and AL-P2-NC to $x^{(\infty)}$; (c, d) NRMSE comparison of AL-tridiag, ADMM-tridiag, MFISTA, and AL-P2-NC to fully sampled SENSE reconstruction.

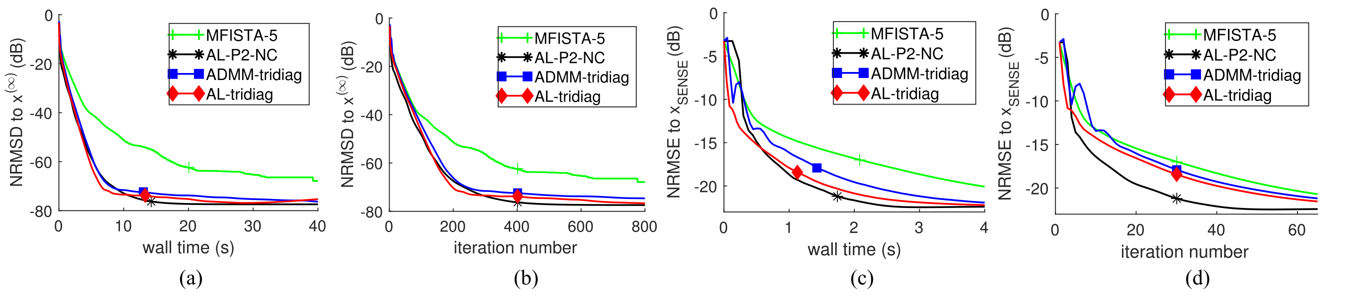


Fig. 5. Axial slice 90: (a, b) NRMSD comparison of AL-tridiag, ADMM-tridiag, MFISTA, and AL-P2-NC to $x^{(\infty)}$; (c, d) NRMSE comparison of AL-tridiag, ADMM-tridiag, MFISTA, and AL-P2-NC to fully sampled SENSE reconstruction.

To generate the solution, $x^{(\infty)}$, we ran MFISTA for 50000 outer iterations with 5 inner NCG iterations. We also calculated the normalized root mean squared error (NRMSE) between a given iterate $x^{(n)}$ and the fully-sampled SENSE reconstruction, $\hat{x}_{\text{SENSE}}$, computed without any regularization.

For computation speed, we measured the wall time of each algorithm. For the AL and ADMM algorithms, we omitted time spent tuning AL penalty parameters and compiling Pthreaded MEX functions. The MFISTA method requires precomputation of the maximum eigenvalue of $\mathbf{S}'\mathbf{F}'\mathbf{F}\mathbf{S}$ via power iteration, which took approximately 4.3 seconds for *in vivo* experiments, whereas all the VS methods avoid this overhead. Computation time excludes time spent computing this maximum eigenvalue. For all algorithms, we omitted the time spent computing the initial sum-of-squares estimate.

As demonstrated in Fig. 4, MFISTA is costlier per iteration than the proposed methods and AL-P2-NC. AL-tridiag converges slightly faster than ADMM-tridiag due to having fewer auxiliary and dual variables to update. AL-P2-NC converged the fastest for this slice.

Fig. 5 shows that axial slice 90 presented a change in relative computation speed toward $x^{(\infty)}$: AL-tridiag and ADMM-tridiag converge faster than AL-P2-NC down to -65 dB NRMSD and up to 500 iterations. In the simulation results shown in the supplement, ADMM-tridiag also converged faster than AL-P2-NC in the early iterations. Overall, the speed of ADMM-tridiag is generally comparable to that of AL-P2-NC. For all AL/ADMM methods, the convergence rate depends on parameter selection; the heuristics used for parameter design in [4] may perform better under some conditions than others. One possible reason for the difference in relative convergence speeds in experiments for slice 38 and slice 90 is the smaller anatomical support in slice 90. Due to the head coil geometry, the smaller head circumference at slice 90 results in a lower signals from the surface coils, which may present a more difficult reconstruction problem.

For these axial slices, we also examined the NRMSE between iterates $x^{(n)}$ and the fully sampled SENSE reconstruction $\hat{x}_{\text{SENSE}}$ without regularization. AL-tridiag and ADMM-tridiag reach the minimum NRMSE after similar amounts of computation as AL-P2-NC, approximately 4 seconds and 60 iterations. By this metric, MFISTA performs slightly worse as a function of computation time and iterations, and would be far worse when one accounts for the overhead of running the power iteration to find the maximum eigenvalue of $\mathbf{S}'\mathbf{F}'\mathbf{F}\mathbf{S}$.

## C. *Image Inpainting*

This section illustrates the benefits of non-periodic boundary conditions and the proposed variable splitting scheme for an inpainting problem. Unlike medical images that often have air at one or more boundaries, natural scenes typically contain useful, distinct information at the boundaries.

To test the effect of AL-tridiag-inpaint, we took a $432 \times 540$ digital photograph using a Samsung SM-G930V camera, randomly discarded 75% of the pixels, and added white Gaussian noise corresponding to 20 dB SNR to the remaining pixels. We used 2D nearest neighbor interpolation to initialize the inpainting estimate, pictured in Fig. 6.

To demonstrate the ease with which the proposed formulation accommodates orthonormal wavelets, we estimated the

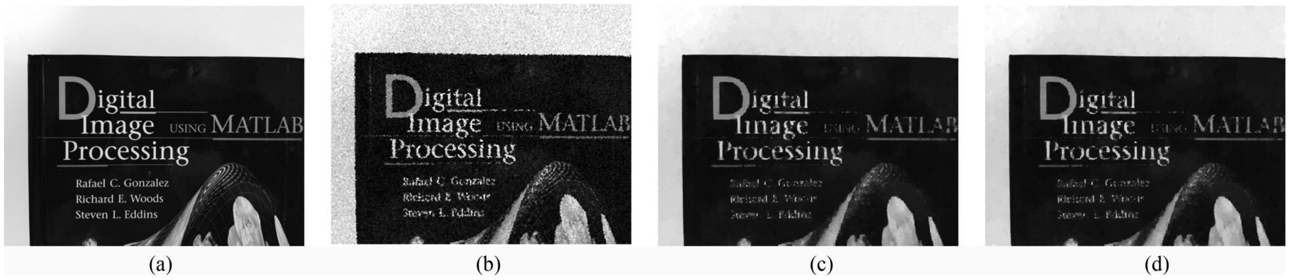


Fig. 6. Inpainting images (a) true image; (b) nearest neighbor interpolation of noisy, partial data (SNR = 20 dB, 75% discarded); (c) inpainting estimate using finite differences with non-periodic boundary conditions and Haar wavelet regularization; (d) inpainting estimate using finite differences with periodic boundary conditions and Haar wavelet regularization. $x_{\mathrm{true}}$.

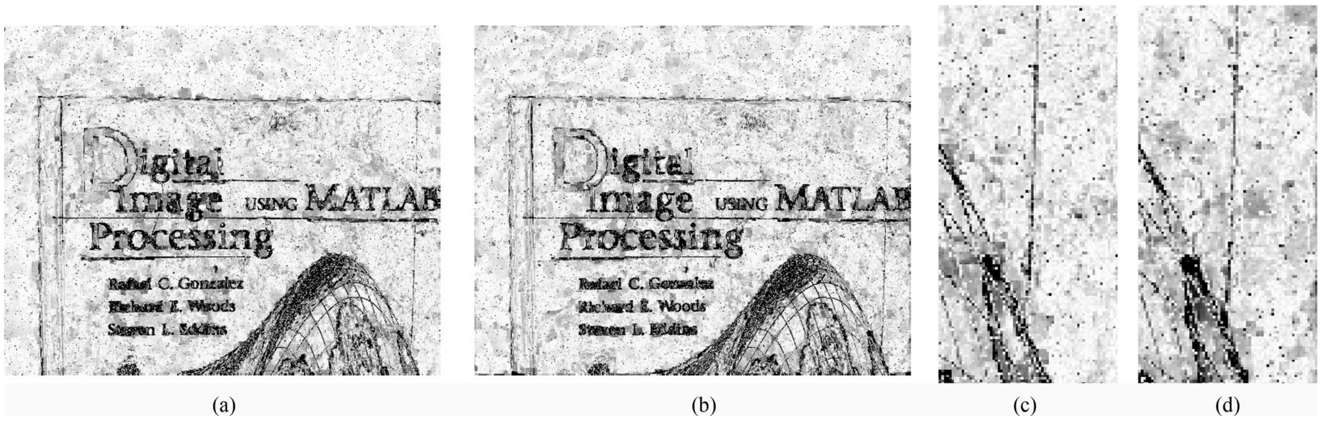


Fig. 7. Absolute difference image between true image and inpainting reconstruction ($\times 10$) (a) using finite differences with non-periodic boundary conditions and Haar wavelets (NRMSE = 0.153); (b) using finite differences with periodic boundary conditions and Haar wavelets (NRMSE = 0.155); (c) corner detail of (a); (d) corner detail of (b).

inpainted image using AL-tridiag-inpaint with the modified regularization operators in (29). We selected regularization parameters $\lambda_1$ and $\lambda_2$ for good image reconstruction quality, and set $\alpha_w = 1$ to limit additional memory usage. We also applied the AL-P2 variable splitting scheme to the inpainting problem, using finite-differences with periodic boundary conditions and Haar wavelets. We show the inpainting images estimated using non-periodic boundary conditions in Fig. 6. All images are displayed on the same grayscale axis as the original image, unless otherwise noted.

Fig. 7 shows the error between the inpainting estimates using non-periodic vs. periodic boundary conditions. The use of periodic boundary conditions results in higher error near the boundaries of the image.

We conducted the inpainting computational speed experiments on the machine described in Section IV-A, and we compute wall time using the same rules as in Section III-G, excluding time spent tuning AL parameters. We measure NRMSD to the MFISTA solution, $x^{(\infty)}$, as a function of wall time. As in the CS-SENSE-MRI experiments, we compare AL-tridiag-inpaint to a variant of AL-P2 to better understand the effect of the proposed variable splitting scheme. We apply the AL-P2 variable splitting scheme to the inpainting problem with non-periodic boundary conditions (32), and we call the resulting AL algorithm AL-P2-NC-inpaint. Similar to AL-P2-NC, it requires an inner iterative variable update due to the non-circulant Hessian. We solve this inner step using one iteration of preconditioned gradient descent with a circulant preconditioner.

Supposing that boundary artifacts are a secondary concern to computational speed, we also compare the speed of AL-tridiag-inpaint to AL-P2-inpaint. AL-P2-inpaint is distinct from AL-P2-NC-inpaint due to its cost function, which uses regularizers with periodic boundary conditions. AL-P2-inpaint is not handicapped by an inner iterative update, because the circulant Hessian can be diagonalized efficiently via FFTs. Though AL-tridiag-inpaint must complete two tridiagonal solver variable updates for each of AL-P2-inpaint's FFT-based variable updates, the $\mathcal{O}(n)$ runtime of the tridiagonal solver and the parallelized implementation result in comparably fast iterations for AL-tridiag-inpaint. The average computation time for each iteration was 0.0543 seconds for AL-P2-inpaint, 0.1193 seconds for AL-P2-NC-inpaint, 0.0583 seconds for AL-tridiag-inpaint, and 0.0745 seconds for ADMM-tridiag-inpaint.

Fig. 8 also demonstrates convergence benefits of the proposed variable splitting scheme. Both AL-tridiag-inpaint and ADMM-tridiag-inpaint were able to reach the same $x^{(\infty)}$ as MFISTA, unlike AL-P2-NC.

## V. Discussion

The ADMM-tridiag algorithm provides a simple way to ensure convergence for variable splitting methods. By examining

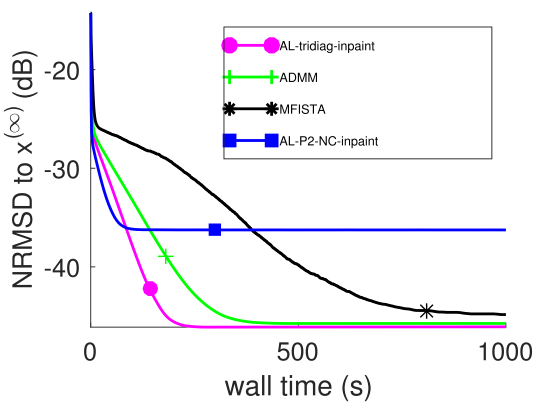


Fig. 8. NRMSD to $x^{(\infty)}$ as a function of elapsed computational time for the proposed inpainting algorithms, AL-tridiag-inpaint and ADMM-tridiag-inpaint, compared with existing methods AL-P2-NC, and MFISTA.

the constraint matrix and designing **B**, we show equivalence between the variable splitting scheme in (8) and ADMM. The additional variable split and variable update led to parallelizability of two of the resulting variable updates. For applications as sensitive as medical diagnosis, an algorithm with convergence guarantees may be preferable to those having unknown convergence properties.

Unlike AL-P2 [4], the proposed algorithm, ADMM-tridiag, has a convergence guarantee and addresses non-periodic boundary conditions, while demonstrating comparable computational speed. Using heuristic parameter tuning based on condition numbers of variable update Hessians, we demonstrated that the speeds of AL-P2-NC and ADMM-tridiag are similar but can vary depending on experimental conditions.

The proposed variable splitting scheme can be useful for a variety of image processing problems, because many natural scenes have non-zero values at the boundaries that do not relate periodically to opposite boundaries. As shown in Section III-E, the proposed method AL-tridiag is readily adapted to image denoising and inpainting problems. Image deblurring is also a good candidate for this proposed variable splitting. Separation of horizontal and vertical differences has also been explored in image segmentation [23]. For MRI reconstruction, the benefit of non-periodic boundary conditions is evident for anatomy that is not entirely surrounded by air. Additional simulation experiments in the supplement illustrate the effect of periodic boundary conditions in reconstructing a sagittal slice of the brain.

To fully benefit from the $\max(N_x, N_y)$ parallelizable tridiagonal updates of AL-tridiag and ADMM-tridiag, one should use a highly parallel computing platform1, such as a GPU.

The algorithms proposed in this work have several limitations. Though the proposed algorithms are designed to facilitate fast computation, the convergence speed is highly dependent on good penalty parameter choice. Though we present some useful heuristics for choosing the AL convergence parameters, the optimal procedure for designing these parameters is a difficult analysis problem and still unknown. (For some simpler ADMM methods, optimal parameter tuning has been analyzed [24].) Moreover, the complexity of convergence parameter design increases with the number of variable splits, and this work is built around an additional separation of horizontal and vertical differences into distinct auxiliary variables. The increased number of tuning parameters introduces another degree of freedom. Using non-scalar penalty parameter matrices $\mathbf{M}_3$ and $\mathbf{M}_4$ further increases the degrees of freedom compared to the simple scalar choice used in most AL methods.

Though the proposed variable splitting scheme can be easily extended to 3D reconstruction problems, this would require introducing two additional variable splits, separating each of the three finite difference directions and introducing a second auxiliary variable proxy for $x$. Though the corresponding variable updates can be quickly computed with shrinkage and the parallelizable tridiagonal solver, this 3D variable splitting scheme could further complicate penalty parameter analysis.

Finally, the variable splitting scheme at the center of the proposed algorithms is applicable only for regularization with first-order finite differences. Though Section III-D shows that the formulation also accommodates orthonormal wavelet penalties combined with finite differences, this variable splitting scheme yields no benefit for other sparsity transforms, e.g., non-orthonormal wavelets and learned dictionaries.

## VI. Summary and Conclusion

This work proposes a variable splitting algorithm for SENSE MRI reconstruction, ADMM-tridiag. The proposed method offers convergence guarantees and efficient variable updates for non-periodic boundary conditions. ADMM-tridiag efficiently handles the non-periodic boundary conditions by separating the finite differences in the horizontal and vertical directions to create easily solvable and parallelizable tridiagonal problems. The method for inducing ADMM equivalence requires only one additional variable split and variable update.

We also presented a simpler variation of this algorithm: AL-tridiag. AL-tridiag was derived from the same variable splitting scheme as ADMM-tridiag, but has a simpler update procedure, resulting in a slight speed increase albeit without any convergence guarantees. We showed a simple relationship between AL-tridiag and ADMM-tridiag and compared their convergence speeds to that of AL-P2-NC and MFISTA. Convergence speed was evaluated in terms of distance to the solution of the proposed cost function (1) as well as to the fully sampled SENSE reconstruction. For retrospectively undersampled *in vivo* data, the proposed algorithms demonstrated comparable convergence speed and produced reconstructed images with good image quality. AL-tridiag was also applied to a noisy image inpainting problem, demonstrating faster convergence speed than AL-P2-NC, and improved image fidelity at the boundaries than AL-P2 with periodic boundary conditions.

All of the proposed algorithms require selection of penalty parameters. We use heuristics determined in [4] to select these AL penalty parameters, although we exploited a more general version of tuning parameters to enable computation of condition numbers for tridiagonal Hessians. Using methods that adapt the parameters as a function of iteration [25] might simplify and accelerate AL methods.

The supplement also describes a fully parallelizable variant of ADMM-tridiag inspired by [16] that converged slower than the proposed methods, as well as additional simulation results and image quality comparisons.

## Acknowledgment

The authors would like to thank S. Ramani for helpful discussions regarding variable splitting methods for SENSE MRI reconstruction.

## References

[1] K. P. Pruessmann, M. Weiger, M. B. Scheidegger, and P. Boesiger, "SENSE: Sensitivity encoding for fast MRI," *Mag. Res. Med.*, vol. 42, no. 5, pp. 952–62, Nov. 1999.
[2] M. Lustig, D. Donoho, and J. M. Pauly, "Sparse MRI: The application of compressed sensing for rapid MR imaging," *Mag. Res. Med.*, vol. 58, no. 6, pp. 1182–95, Dec. 2007.
[3] J. Eckstein and D. P. Bertsekas, "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Math. Programm.*, vol. 55, no. 1–3, pp. 293–318, Apr. 1992.
[4] S. Ramani and J. A. Fessler, "Parallel MR image reconstruction using augmented Lagrangian methods," *IEEE Trans. Med. Imag.*, vol. 30, no. 3, pp. 694–706, Mar. 2011.
[5] Y. Chen, W. Hager, F. Huang, D. Phan, X. Ye, and W. Yin, "Fast algorithms for image reconstruction with application to partially parallel MR imaging," *SIAM J. Imag. Sci.*, vol. 5, no. 1, pp. 90–118, 2012.
[6] A. Beck and M. Teboulle, "Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems," *IEEE Trans. Image Process.*, vol. 18, no. 11, pp. 2419–34, Nov. 2009.
[7] J. Eckstein, "Parallel alternating direction multiplier decomposition of convex programs," *J. Optim. Theory Appl.*, vol. 80, no. 1, pp. 39–62, Jan. 1994.
[8] J. Yang, Y. Zhang, and W. Yin, "A fast alternating direction method for TVL1-L2 signal reconstruction from partial Fourier data," *IEEE J. Sel. Top. Signal Process.*, vol. 4, no. 2, pp. 288–297, Apr. 2010.
[9] Y. Chen, X. Ye, and F. Huang, "A novel method and fast algorithm for MR image reconstruction with significantly under-sampled data," *Inverse Problems. Imag.*, vol. 4, no. 2, pp. 223–240, 2010.
[10] A. Matakos, S. Ramani, and J. A. Fessler, "Accelerated edge-preserving image restoration without boundary artifacts," *IEEE Trans. Image Process.*, vol. 22, no. 5, pp. 2019–29, May 2013.
[11] M. S. C. Almeida and M. A. T. Figueiredo, "Deconvolving images with unknown boundaries using the alternating direction method of multipliers," *IEEE Trans. Image Process.*, vol. 22, no. 8, pp. 3074–86, Aug. 2013.
[12] M. J. Allison, S. Ramani, and J. A. Fessler, "Accelerated regularized estimation of MR coil sensitivities using augmented Lagrangian methods," *IEEE Trans. Med. Imag.*, vol. 32, no. 3, pp. 556–64, Mar. 2013.
[13] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "Fast image recovery using variable splitting and constrained optimization," *IEEE Trans. Image Process.*, vol. 19, no. 9, pp. 2345–56, Sep. 2010.
[14] X. Ye, Y. Chen, and F. Huang, "Computational acceleration for MR image reconstruction in partially parallel imaging," *IEEE Trans. Med. Imag.*, vol. 30, no. 5, pp. 1055–63, May 2011.
[15] D. P. Bertsekas, "Multiplier methods: A survey," *Automatica*, vol. 12, no. 2, pp. 133–45, Mar. 1976.
[16] S. Ramani and J. A. Fessler, "Regularized parallel MRI reconstruction using an alternating direction method of multipliers," in *Proc. IEEE Int. Symp. Biomed. Imag.*, 2011, pp. 385–388.
[17] B. Fischer and J. Modersitzki, "Fast inversion of matrices arising in image processing," *Numer. Algorithms*, vol. 22, no. 1, pp. 1–11, 1999.
[18] P. J. Davis, *Circulant Matrices*. New York, NY, USA: Wiley, 1983.
[19] R. H. Chan, J. Yang, and X. Yuan, "Alternating direction method for image inpainting in wavelet domains," *SIAM J. Imag. Sci.*, vol. 4, no. 3, pp. 807–826, 2011.
[20] J. Dahl, P. C. Hansen, S. H. Jensen, and T. L. Jensen, "Algorithms and software for total variation image reconstruction via first-order methods," *Numer. Algorithms*, vol. 51, no. 1, pp. 67–92, Jan. 2010.
[21] G. Strang, "The discrete cosine transform," *SIAM Rev.*, vol. 41, no. 1, pp. 135–47, 1999.
[22] J. A. Fessler, "Matlab tomography toolbox," 2004. [Online]. Available: http://web.eecs.umich.edu/~fessler/code/
[23] M. Storath and A. Weinmann, "Fast partitioning of vector-valued images," *SIAM J. Imag. Sci.*, vol. 7, no. 3, pp. 1826–52, 2014.
[24] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, "Optimal parameter selection for the alternating direction method of multipliers (ADMM): Quadratic problems," 2013. [Online]. Available: http://www.optimization-online.org/DB_HTML/2013/06/3917.html
[25] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2010.

**Mai Le** (S'12) received the B.Sc. degree in electrical engineering from Stanford University, Stanford, CA, USA, in 2012, and the M.S. degree in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 2013. She is currently working toward the Ph.D. degree in electrical engineering: systems at the University of Michigan. Her current research interests include model based iterative reconstruction methods for MRI.

**Jeffrey A. Fessler** (F'06) received the B.S.E.E. degree from Purdue University, West Lafayette, IN, USA, in 1985. He received the M.S.E.E. degree in 1986, the M.S. degree in statistics in 1989, and the Ph.D. degree in electrical engineering in 1990, all from Stanford University, Stanford, CA, USA. From 1985 to 1988, he was a National Science Foundation Graduate Fellow at Stanford. He has worked at the University of Michigan since then. From 1991 to 1992, he was a Department of Energy Alexander Hollaender Postdoctoral Fellow in the Division of Nuclear Medicine. From 1993 to 1995, he was an Assistant Professor in Nuclear Medicine and the Bioengineering Program. He is currently a Professor in the Departments of Electrical Engineering and Computer Science, Radiology, and Biomedical Engineering. His research interests include statistical aspects of imaging problems, and he has supervised doctoral research in PET, SPECT, X-ray CT, MRI, and optical imaging problems. He became a Fellow of the IEEE in 2006, for contributions to the theory and practice of image reconstruction. He received the Francois Erbsmann award for his IPMI93 presentation, and the Edward Hoffman Medical Imaging Scientist Award in 2013. He has served as an Associate Editor for IEEE TRANSACTIONS ON MEDICAL IMAGING, the IEEE SIGNAL PROCESSING LETTERS, and the IEEE TRANSACTIONS ON IMAGE PROCESSING, and is currently serving as an Associate Editor for the IEEE TRANSACTIONS ON COMPUTATIONAL IMAGING. He has chaired the IEEE T-MI Steering Committee and the ISBI Steering Committee. He was the Co-Chair of the 1997 SPIE conference on Image Reconstruction and Restoration, the Technical Program Co-Chair of the 2002 IEEE International Symposium on Biomedical Imaging (ISBI), and the General Chair of ISBI 2007.

# Efficient, Convergent SENSE MRI Reconstruction for Non-Periodic Boundary Conditions via Tridiagonal Solvers: Supplementary Material

Mai Le, *Student Member, IEEE,* Jeffrey A. Fessler, *Fellow, IEEE*

In this supplementary material for [1], we elaborate on variable updates for AL-tridiag, propose an additional convergent algorithm for SENSE reconstruction using tridiagonal solvers, called ADMM-FP-tridiag, like those proposed in [1], and present additional experimental results. ADMM-FP-tridiag is a fully parallelizable variation of ADMM-tridiag that is amenable to parallelization and has a stronger convergence guarantee because every update is exact (to within computer precision) so no summability conditions are needed.

## I. Variable updates for AL-tridiag

Alternating minimization updates for $\underline{u}$ for ADMM-tridiag corresponding to (12) were presented in Section (III-C) of [1]. Here we explicitly describe the alternating minimization updates for $\underline{u}$ in proposed algorithm AL-tridiag.

$$u_0^{(n+1)} = \text{soft}\left(\mathbf{C}_{\text{H}} x^{(n)} - \eta_0^{(n)}, \frac{\lambda}{\mu_0}\right) \tag{54}$$

$$u_1^{(n+1)} = \text{soft}\left(\mathbf{C}_{\text{V}} u_3^{(n)} - \eta_1^{(n)}, \frac{\lambda}{\mu_1}\right) \tag{55}$$

$$u_2^{(n+1)} = \mathbf{H}_2^{-1}\left(\mathbf{F}' y + \mu_2\left(\frac{1}{2}\mathbf{S} u_3^{(n)} + \frac{1}{2}\mathbf{S} x^{(n)} - \eta_2^{(n)}\right)\right) \tag{56}$$

$$u_3^{(n+1)} = \mathbf{H}_3^{-1}\left(\mu_1 \mathbf{C}_{\text{V}}'\left(u_1^{(n+1)} + \eta_1^{(n)}\right) + \frac{\mu_2}{2}\mathbf{S}'\left(u_2^{(n+1)} - \frac{1}{2}\mathbf{S} x^{(n)} + \eta_2^{(n)}\right) + \mathbf{M}_3\left(x^{(n)} - \eta_3^{(n)}\right)\right) \tag{57}$$

$$x^{(n+1)} = \mathbf{H}_{\text{x}}^{-1}\left(\mu_0 \mathbf{C}_{\text{H}}'\left(u_0^{(n)} + \eta_0^{(n)}\right) + \frac{\mu_2}{2}\mathbf{S}'\left(u_2^{(n+1)} - \frac{1}{2}\mathbf{S} u_3^{(n+1)}) + \eta_2^{(n)}\right) + \mathbf{M}_3\left(u_3^{(n)} + \eta_3^{(n)}\right)\right), \tag{58}$$

where the Hessians $\mathbf{H}_2$ and $\mathbf{H}_3$ are the same as for ADMM-tridiag, defined in Equations (24)-(25). In AL-tridiag, $\mathbf{H}_{\text{x}} \triangleq \mu_0 \mathbf{C}_{\text{H}}' \mathbf{C}_{\text{H}} + \frac{\mu_2}{4}\mathbf{S}'\mathbf{S} + \mathbf{M}_3$. The updates for $u_0$, $u_1$, and $u_2$, (54)-(56) are identical to those in ADMM-tridiag. The updates for $u_3$ and $x$ differ only in the rightmost term containing $\eta_3$ and in the Hessian $\mathbf{H}_{\text{x}}$.

In practice, the implementation differences between AL-tridiag and ADMM-tridiag are further diminished by choosing the same $\mathbf{H}_{\text{x}}$ for both algorithms. Due to the parameter selection technique for ADMM-tridiag described in (III-G), $\mathbf{M}_3 = \mathbf{M}_4$ for ADMM-tridiag, resulting in identical Hessians for both algorithms.

## II. Fully Parallelized ADMM: ADMM-FP-Tridiag

In this supplementary material, we introduce another variation of ADMM-tridiag [1] that expands constraint matrix $\mathbf{P}$ in (4) even further. ADMM-tridiag leverages [2] to ensure convergence via equivalence with the Alternating Direction Method of Multipliers (ADMM). As discussed in (III-B), ADMM-tridiag relies on one cycle of alternating

M. Le and J. A. Fessler are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, 48109 USA (e-mail: {mtle, fessler} @umich.edu).

minimization to solve the joint minimization problem in (12). Though this is a common approach in many ADMM methods, we also investigate a fully parallelized alternative, called ADMM-FP-tridiag, which decomposes the constraint matrix in such a way that the resulting joint minimization problem and the alternating minimization approach are identical. This lessens the degree to which this update is inexact, but does not compensate for other sources of inexact computation, such as machine precision. This algorithm is inspired by the fully parallelized ADMM in [3].

To design a fully parallelized variable update scheme, we define constraint matrices $\mathbf{A}_{\mathrm{FP}}$ and $\mathbf{B}_{\mathrm{FP}}$ as follows:

$$\mathbf{B}_{\mathrm{FP}} \triangleq \begin{bmatrix} \mathbf{I} & \mathbf{I} & 0 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 0 & \mathbf{I} & \mathbf{I} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{I} & \mathbf{I} & \mathbf{I} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{I} & \mathbf{I} \end{bmatrix}, \quad \mathbf{A}_{\mathrm{FP}} \triangleq \begin{bmatrix} -\mathbf{I} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{C}_{\mathrm{H}} \\ 0 & -\mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{C}_{\mathrm{V}} & 0 \\ 0 & 0 & -\mathbf{I} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2}\mathbf{S} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2}\mathbf{S} \\ 0 & 0 & 0 & -\mathbf{I} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{I} \end{bmatrix}. \tag{59}$$

This results in a joint minimization problem for $\underline{u}$ that is entirely decoupled for each block of $\underline{u}$. Tradeoffs include $\underline{v}_{\mathrm{FP}}$ $(\in \mathbb{C}^{7N_r+2N_cN_r})$ that is much larger than $\underline{v}$ in ADMM-tridiag and more non-trivial blocks of $\underline{v}_{\mathrm{FP}}$ to update. Likewise, $\underline{\eta}_{\mathrm{FP}}$ is also larger than in ADMM-tridiag.

Choosing $\mathbf{M}_i \triangleq \mu_i \mathbf{I},\ \mu_i \in \mathbb{R}$ for $i = 0, \ldots, 6$, the expansion of (12) is:

$$\begin{aligned} \underline{u}^{(n+1)} = \underset{x,u_0,u_1,u_{2,3}}{\operatorname{argmin}} \ & \frac{1}{2}\|y - \mathbf{F}u_2\|^2 + \lambda\|u_0\|_1 + \lambda\|u_1\|_1 + \frac{\mu_0}{2}\left\|-u_0^{(n)} - v_0^{(n)} - \eta_0^{(n)}\right\|^2 \\ & + \frac{\mu_1}{2}\left\|\mathbf{C}_{\mathrm{H}}x^{(n)} - v_1^{(n)} - \eta_1^{(n)}\right\|^2 + \frac{\mu_2}{2}\left\|-u_1^{(n)} - v_2^{(n)} - \eta_3^{(n)}\right\|^2 + \frac{\mu_3}{2}\left\|\mathbf{C}_{\mathrm{V}}u_3^{(n)} - v_3^{(n)} - \eta_3^{(n)}\right\|^2 \\ & + \frac{\mu_4}{2}\left\|-u_2^{(n)} - v_4^{(n)} - \eta_4^{(n)}\right\|^2 + \frac{\mu_5}{2}\left\|\frac{1}{2}\mathbf{S}u_3^{(n)} - v_5^{(n)} - \eta_5^{(n)}\right\|^2 + \frac{\mu_6}{2}\left\|\frac{1}{2}\mathbf{S}x^{(n)} - v_6^{(n)} - \eta_6^{(n)}\right\|^2 \\ & + \frac{1}{2}\left\|-u_3^{(n)} - v_7^{(n)} - \eta_7^{(n)}\right\|^2_{\mathbf{M}_7} + \frac{1}{2}\left\|x^{(n)} - v_8^{(n)} - \eta_8^{(n)}\right\|^2_{\mathbf{M}_8}. \end{aligned} \tag{60}$$

The variable updates that result from alternating minimization are:

$$u_0^{(n+1)} = \mathrm{soft}\left(v_0^{(n)} - \eta_0^{(n)}, \frac{\lambda}{\mu_0}\right) \tag{61}$$

$$u_1^{(n+1)} = \mathrm{soft}\left(v_2^{(n)} - \eta_3^{(n)}, \frac{\lambda}{\mu_2}\right) \tag{62}$$

$$u_2^{(n+1)} = \mathbf{H}_{2,\mathrm{FP}}^{-1}\left(\mathbf{F}'y + \mu_4\left(v_4^{(n)} - \eta_4^{(n)}\right)\right) \tag{63}$$

$$u_3^{(n+1)} = \mathbf{H}_{3,\mathrm{FP}}^{-1}\left(\mu_3\mathbf{C}_{\mathrm{V}}'\left(v_3^{(n)} + \eta_3^{(n)}\right) + \frac{\mu_5}{2}\mathbf{S}'\left(v_5^{(n)} + \eta_5^{(n)}\right) + \mu_7\left(-v_7^{(n)} - \eta_7^{(n)}\right)\right) \tag{64}$$

$$x^{(n+1)} = \mathbf{H}_{\mathrm{x,FP}}^{-1}\left(\mu_1\mathbf{C}_{\mathrm{H}}'\left(v_1^{(n)} + \eta_1^{(n)}\right) + \frac{\mu_6}{2}\mathbf{S}'\left(v_6^{(n)} + \eta_6^{(n)}\right) + \mu_8\left(v_8^{(n)} + \eta_8^{(n)}\right)\right). \tag{65}$$

The soft-thresholding operator performs element-wise shrinkage for the $\ell_1$ norm using a given threshold $\tau$: $\mathrm{soft}(x, \tau) \triangleq$

$\operatorname{sign}(x)\max(x-\tau,0)$. The bulk of the computational cost lies in inverting the following Hessians:

$$\mathbf{H}_{2,\mathrm{FP}} \triangleq \mathbf{F}'\mathbf{F} + \mu_4\mathbf{I} = \mathbf{Q}\left(\boldsymbol{\Lambda}_F + \mu_4\mathbf{I}\right)\mathbf{Q}' \tag{66}$$

$$\mathbf{H}_{3,\mathrm{FP}} \triangleq \mu_3\mathbf{C}_{\mathrm{V}}'\mathbf{C}_{\mathrm{V}} + \frac{\mu_5}{4}\mathbf{S}'\mathbf{S} + \mu_7\mathbf{I} \tag{67}$$

$$\mathbf{H}_{\mathrm{x},\mathrm{FP}} \triangleq \mu_1\mathbf{C}_{\mathrm{H}}'\mathbf{C}_{\mathrm{H}} + \frac{\mu_6}{4}\mathbf{S}'\mathbf{S} + \mu_8\mathbf{I}. \tag{68}$$

As in AL-P2, $\mathbf{H}_{2,\mathrm{FP}}$ is diagonalizable via FFTs, which operate efficiently in $\mathcal{O}(n\log n)$ time. The $x$ update in (65) requires inverting $\mathbf{H}_{\mathrm{x},\mathrm{FP}} \triangleq \mu_1\mathbf{C}_{\mathrm{H}}'\mathbf{C}_{\mathrm{H}} + \frac{1}{2}\frac{1}{2}\mu_6\mathbf{S}'\mathbf{S} + \mu_8\mathbf{I}$, a block diagonal matrix with tridiagonal blocks that can be inverted in $\mathcal{O}(n)$ time via Gaussian elimination. Because it is block diagonal, we parallelize this variable update over the $N_y$ blocks, whose computations are independent of one another. As in ADMM-tridiag, we reformulate the $u_3$ update as another instance of the same minimization problem through permutation. The remaining variable updates are exact and easy to implement.

The updates for $\underline{v}_{\mathrm{FP}}$ all simplify to quadratic problems and can be implemented directly as follows:

$$v_0^{(n+1)} = \frac{1}{\mu_0+\mu_1}\left(\mu_0\left(-u_0^{(n+1)} - \eta_0^{(n)}\right) + \mu_1\left(-\mathbf{C}_{\mathrm{H}}x^{(n+1)} + \eta_1^{(n)}\right)\right) \tag{69}$$

$$v_1^{(n+1)} = -\,v_0^{(n)} \tag{70}$$

$$v_2^{(n+1)} = \frac{1}{\mu_2+\mu_3}\left(\mu_2\left(-u_1^{(n+1)} - \eta_2^{(n)}\right) + \mu_3\left(-\mathbf{C}_{\mathrm{V}}u_3^{(n+1)} + \eta_3^{(n)}\right)\right) \tag{71}$$

$$v_3^{(n+1)} = -\,v_2^{(n)} \tag{72}$$

$$\begin{aligned} v_4^{(n+1)} =& \frac{1}{\mu_4\mu_5+\mu_5\mu_6+\mu_4\mu_5}\Bigg[(\mu_5+\mu_6)\left(\mu_4\left(-u_2^{(n+1)} + \eta_2^{(n)}\right) + \mu_6\left(\frac{1}{2}\mathbf{S}x^{(n+1)} - \eta_6^{(n)}\right)\right) \\ & - \mu_6\left(\mu_5\left(\frac{1}{2}\mathbf{S}u_3^{(n+1)} - \eta_5^{(n)}\right) - \mu_6\left(\frac{1}{2}\mathbf{S}x^{(n+1)} - \eta_6^{(n)}\right)\right)\Bigg] \end{aligned} \tag{73}$$

$$\begin{aligned} v_5^{(n+1)} =& \frac{1}{\mu_4\mu_5+\mu_5\mu_6+\mu_4\mu_5}\Bigg[-\mu_6\left(\mu_4\left(-u_2^{(n+1)} + \eta_2^{(n)}\right) + \mu_6\left(\frac{1}{2}\mathbf{S}x^{(n+1)} - \eta_6^{(n)}\right)\right) \\ & (\mu_4+\mu_6)\left(\mu_5\left(\frac{1}{2}\mathbf{S}u_3^{(n+1)} - \eta_5^{(n)}\right) + \mu_6\left(\frac{1}{2}\mathbf{S}x^{(n+1)} - \eta_6^{(n)}\right)\right)\Bigg] \end{aligned} \tag{74}$$

$$v_6^{(n+1)} = -\,v_4^{(n+1)} - v_5^{(n+1)} \tag{75}$$

$$v_7^{(n+1)} = \frac{1}{\mu_7+\mu_8}\left(\mu_7\left(-u_3^{(n+1)} - \eta_7^{(n)}\right) + \mu_8\left(-x^{(n+1)} + \eta_8^{(n)}\right)\right) \tag{76}$$

$$v_8^{(n+1)} = -\,v_7^{(n+1)}. \tag{77}$$

By enforcing constraints arising from the structure of $\mathbf{B}_{\mathrm{FP}}$, the full minimization of $\underline{v}_{\mathrm{FP}}$ reduces to the computation of just five of the nine block variables: $v_0$, $v_2$, $v_4$, $v_5$, and $v_7$. The extraneous four variables can be expressed in terms of the remaining four and omitted from the algorithm altogether.

Finally, the dual variables are updated as follows:

$$\underline{\eta}_{\mathrm{FP}}^{(n+1)} = \underline{\eta}_{\mathrm{FP}}^{(n)} - \left(\mathbf{A}_{FP}\underline{u}^{(n+1)} - \underline{v}_{\mathrm{FP}}^{(n+1)}\right). \tag{78}$$

ADMM-FP-tridiag is highly amenable to parallelization. Each block update for $\underline{u}$ can be done in parallel, as can each block update of $\underline{v}_{\mathrm{FP}}$, as described in 2. This opportunity for parallelization helps to offset some of the additional computational cost incurred from the greater number of variable updates per iteration.

In summary, ADMM-FP-tridiag eliminates alternating minimization across the blocks of $\underline{u}$ and $\underline{v}_{\mathrm{FP}}$ by decoupling each of the four original variable splitting constraints (2). This results in additional auxiliary variables in $\underline{v}_{\mathrm{FP}}$ and dual variables $\underline{\eta}_{\mathrm{FP}}$. However, each block-variable has blocks that can be updated in parallel with one another. This approach also provides exact variable updates (to within numerical precision) by avoiding alternating minimization. The procedure for ADMM-FP-tridiag is summarized in Algorithm 2.

**Algorithm 2** ADMM-FP-tridiag

```
1: Initialize: $u_0, u_1, u_2, u_3, x, v_0, v_2, v_4, v_5, v_7, \underline{\eta}_{\mathrm{FP}}$
2: for $n \leq$ total iterations do
3:     do in parallel
4:         Compute $u_0^{(n+1)}$ using soft-thresholding (61)
5:         Compute $u_1^{(n+1)}$ using soft-thresholding (62)
6:         Compute $u_2^{(n+1)}$ using FFTs (63)
7:         Compute $u_3^{(n+1)}$ using parallelized tridiagonal solver (64)
8:         Compute $x^{(n+1)}$ using parallelized tridiagonal solver (65)
9:     end parfor
10:    do in parallel
11:        Compute $v_0^{(n+1)}$ using (69)
12:        Compute $v_2^{(n+1)}$ using (71)
13:        Compute $v_4^{(n+1)}$ using (73)
14:        Compute $v_5^{(n+1)}$ using (74)
15:        Compute $v_7^{(n+1)}$ using (76)
16:    end parfor
17:    Compute $\underline{\eta}_{\mathrm{FP}}^{(n+1)}$ using (78)
18: end for
```

## A. Parameter Selection for ADMM-FP-tridiag

This proposed algorithm, ADMM-FP-tridiag, has a total of nine AL tuning parameters, substantially more than for ADMM-tridiag. Using the condition number heuristics from [4] on Hessians (67)-(68) leaves six remaining degrees of freedom. This is a result of the two tridiagonal Hessians that include entirely different tuning parameters, unlike the case with ADMM-tridiag:

$$\mathbf{H}_{3,\mathrm{FP}} \triangleq \mu_3 \mathbf{C}_{\mathrm{V}}' \mathbf{C}_{\mathrm{V}} + \frac{\mu_5}{4} \mathbf{S}'\mathbf{S} + \mathbf{M}_7$$
$$\mathbf{H}_{\mathrm{x},\mathrm{FP}} \triangleq \mu_1 \mathbf{C}_{\mathrm{H}}' \mathbf{C}_{\mathrm{H}} + \frac{\mu_6}{4} \mathbf{S}'\mathbf{S} + \mathbf{M}_8.$$

The difficulty of designing these parameters is one of the tradeoffs in choosing the fully parallelized alternative of ADMM-tridiag. Tuning parameters for the following speed comparisons were chosen to enforce the aforementioned Hessian values, leaving some degrees of freedom unexplored.

## B. in vivo Experiment Results for ADMM-FP-tridiag

For the same 3D in-vivo volunteer data set used in [1], we tested the convergence speed of ADMM-FP-tridiag. We applied ADMM-FP-tridiag to the same undersampled data from axial slices 38 and 90 using the same estimated smooth sensitivity maps, shown in Figure 9 and undersampling patterns, shown in Figure 1 of [1]. We initialized ADMM-FP-tridiag with the same zero-filled iFFT image as the other algorithms, shown in Figures 2 and 3 of [1].

The proposed ADMM-FP-tridiag algorithm was implemented with 20 Pthreads allocated to the tridiagonal updates in (64) and (65), but no parallelization was implemented across the blocks of $\underline{u}$ or $\underline{v}$. Here we present the same results as shown in Figures 4 and 5 in [1], with additional convergence speed measurements for ADMM-FP-tridiag. As was the case for AL-tridiag and ADMM-tridiag, time spent tuning AL penalty parameters for ADMM-FP-tridiag was not included in computation time.

The fully parallelizable proposed method, ADMM-FP-tridiag, was not implemented with parallelization across the blocks of $\underline{u}$, $\underline{v}_{\mathrm{FP}}$, or $\underline{\eta}_{\mathrm{FP}}$. For this reason, as well as the substantially larger number of auxiliary and dual variables, ADMM-FP-tridiag performs the slowest of the algorithms in this comparison. It is likely that finely tuning the AL penalty parameters of ADMM-FP-tridiag would result in improved speed, but these speed gains likely will not offset

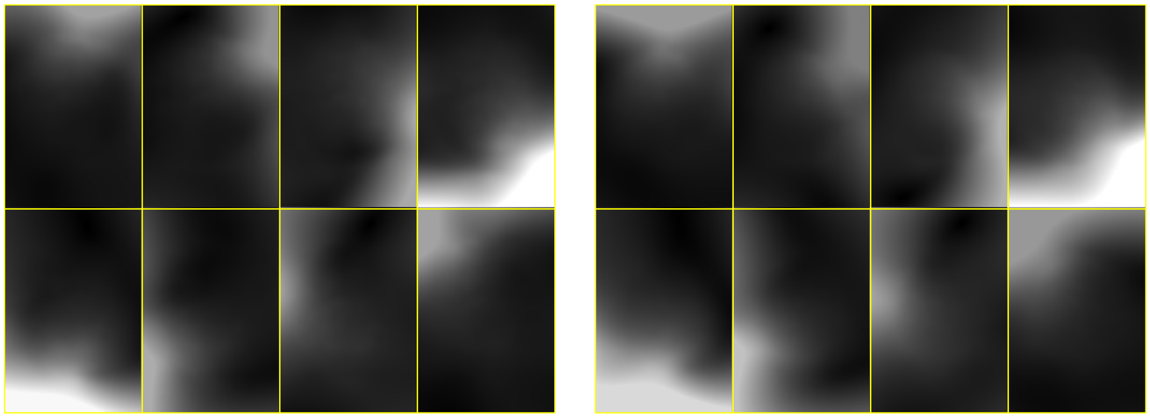


Fig. 9: Magnitudes of the sensitivity maps estimated using central 16 ×16 phase encodes for axial slice 38 (left) and axial slice 90 (right).

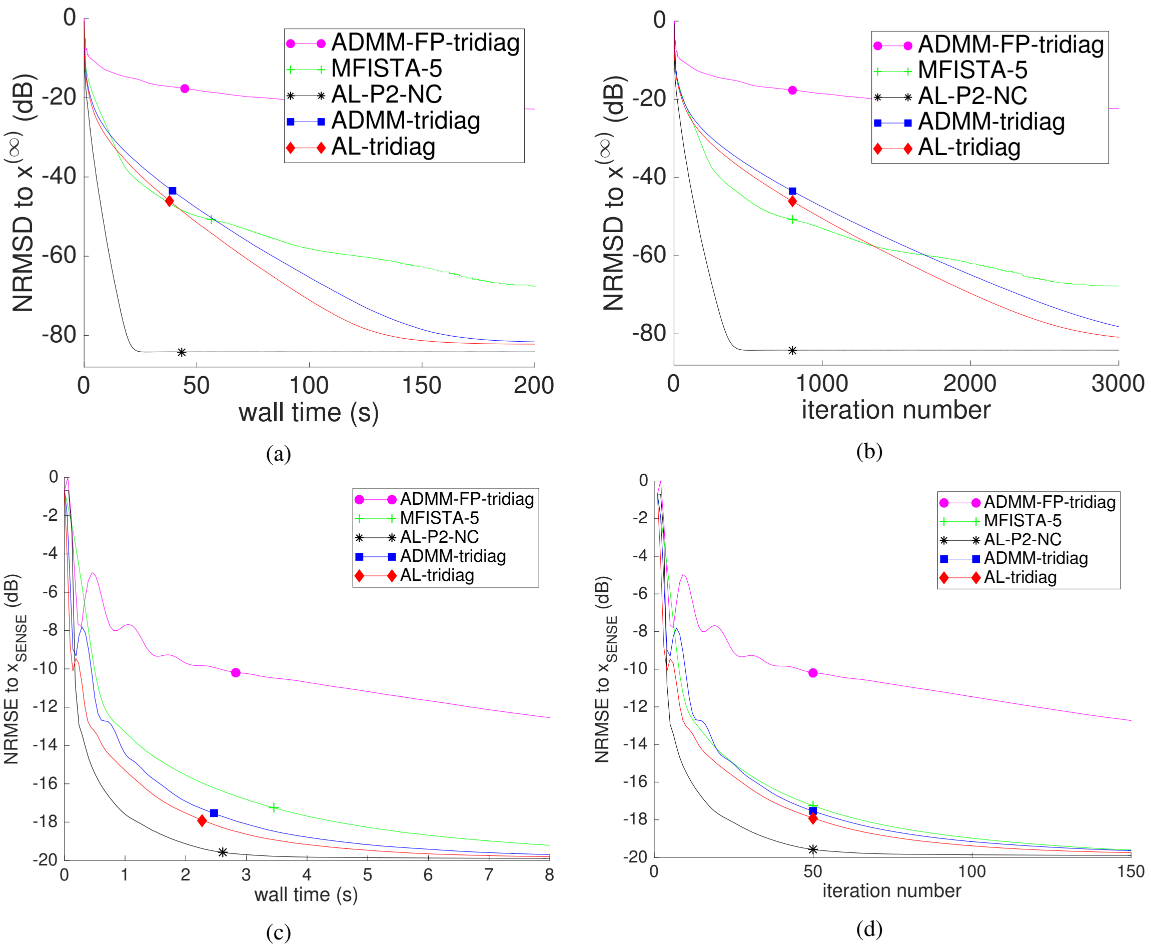


Fig. 10: Axial slice 38: (a, b) NRMSD comparison of proposed algorithms, MFISTA, and AL-P2-NC to $x^{(\infty)}$; (c, d) NRMSE comparison of proposed algorithms, MFISTA, and AL-P2-NC to fully sampled SENSE reconstruction..

the additional bookkeeping required for $\underline{v}_{\text{FP}}$ and $\underline{\eta}_{\text{FP}}$. The use of additional auxiliary variables in ADMM-FP-tridiag may also require more iterations for information to propagate across block elements of $\underline{u}$ and $\underline{v}_{\text{FP}}$.

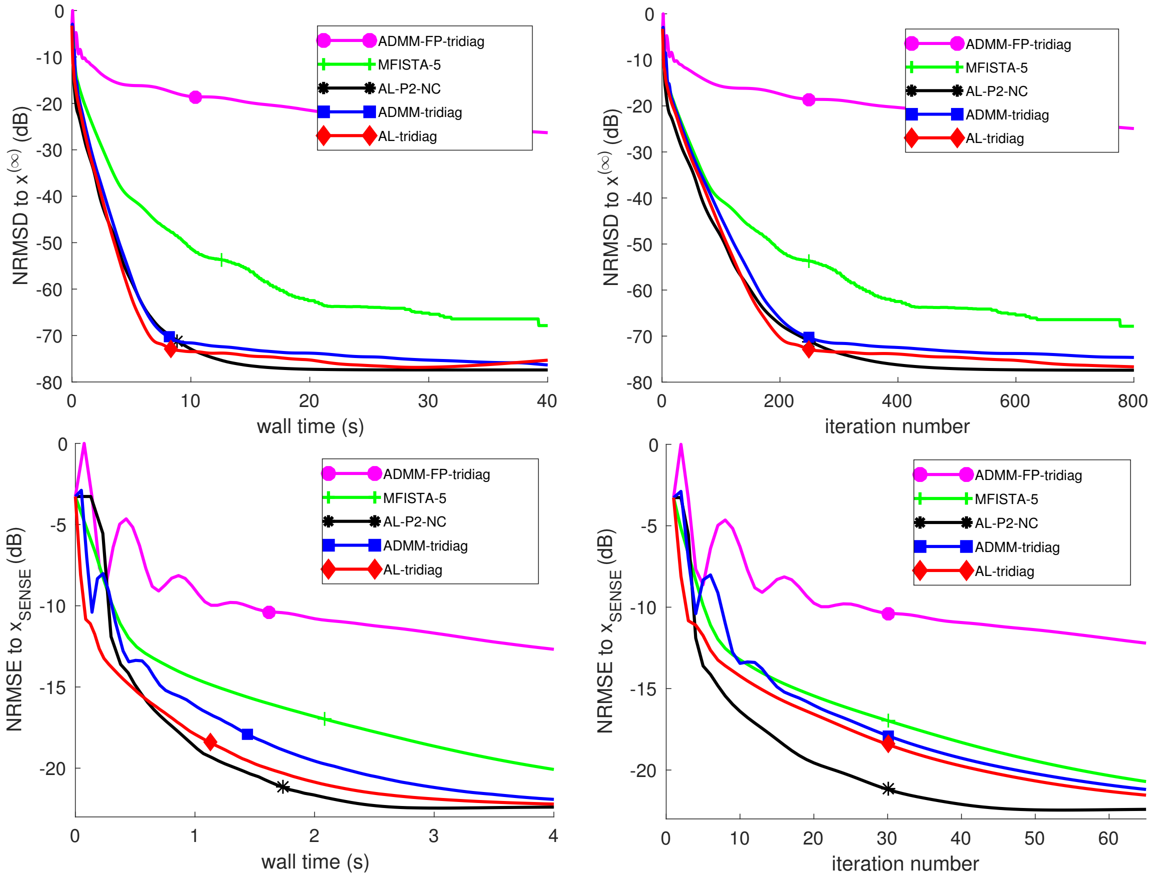


Fig. 11: Axial slice 90: (a,b) NRMSD comparison of proposed algorithms, MFISTA, and AL-P2-NC to $x^{(\infty)}$; (c,d) NRMSE comparison of proposed algorithms, MFISTA, and AL-P2-NC to fully sampled SENSE reconstruction.

For axial slice 90, ADMM-FP-tridiag remains the slowest method. However, due to enforcing equivalence with ADMM, the solution to ADMM-FP-tridiag is within machine precision of $x^*$, despite the much longer convergence time. For this reason, we omit images of the solution to ADMM-FP-tridiag for slices 38 and 90.

## III. Image Quality Comparison for *in vivo* Data

In this section, we compare the image quality of the solution to (1) to the body coil image and the fully sampled SENSE reconstruction $\hat{x}_{\text{SENSE}}$. For both axial slices 38 and 90, all algorithms in the comparison eventually reach the same solution, $x^{(\infty)}$. The reconstructed image from ADMM-tridiag at 5000 iterations, $\hat{x}$ is visually similar to the bodycoil, shown in Figures 12 and 13. ADMM-tridiag is able to reconstruct many of the anatomical details missing in the initial zero-filled iFFT images. The estimated images for AL-tridiag at 5000 iterations is very similar to that of ADMM-tridiag and are not pictured.

Figures 12 and 13 also show the absolute difference of the body coil and reconstructed image, as well as the fully sampled SENSE reconstruction without regularization, $\hat{x}_{\text{SENSE}}$ and the difference between the reconstructed image and SENSE image. The difference image between $\hat{x}$ and $\hat{x}_{\text{SENSE}}$ show that the regions with the highest error are those in which the g-factor is low due to the head coil geometry. The reconstructed image more closely resembles the fully sampled SENSE image than the relatively noisy body coil image.

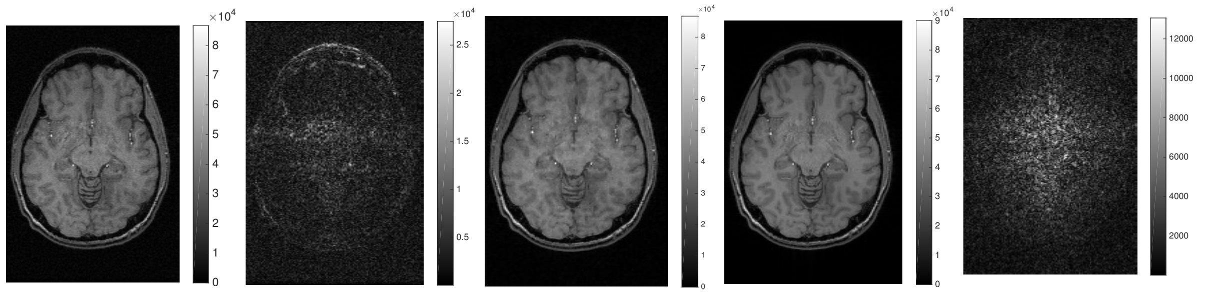


Fig. 12: *in vivo* experiment for axial slice 38. Left to right: (a) body coil image; (b) differences between body coil and ADMM-tridiag; (c) ADMM-tridiag reconstruction; (d) fully sampled SENSE reconstruction; (e) differences between fully sampled SENSE reconstruction and ADMM-tridiag.

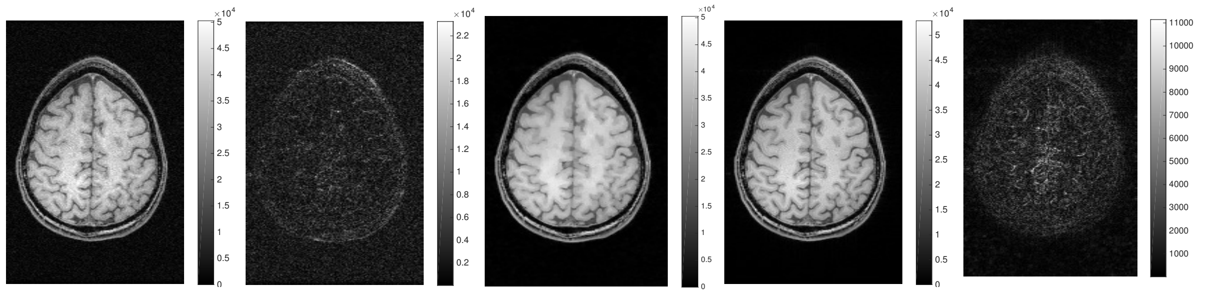


Fig. 13: *in vivo* experiment for axial slice 90. Left to right: (a) body coil image; (b) differences between body coil and ADMM-tridiag; (c) ADMM-tridiag reconstruction; (d) fully sampled SENSE reconstruction; (e) differences between fully sampled SENSE reconstruction and ADMM-tridiag.

## IV. Simulated Data

### A. Axial Slice Reconstruction

We also conducted additional experiments on an undersampled digital phantom to measure reconstruction error. We simulated noisy multi-coil data from a $T_1$-weighted 240 $\times$ 200 BrainWeb image with linear phase. We generated sensitivity maps for a 6-channel head coil array and generated noisy k-space data with SNR of 40. A Poisson-disk based sampling pattern [5] containing the central $16 \times 16$ phase encodes and with an overall undersampling factor of 6 was used for undersampling, as shown in Figure 14.

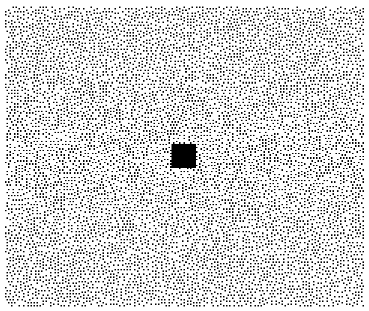
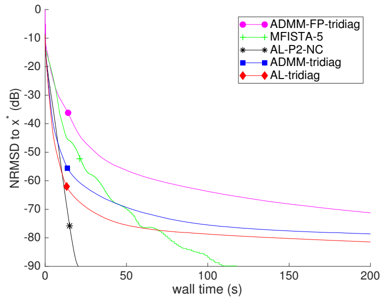


Fig. 14: Poisson-disk-based undersampling pattern used for retrospective undersampling, with reduction factor of 6 and fully sampled central 16$\times$16 phase-encodes.

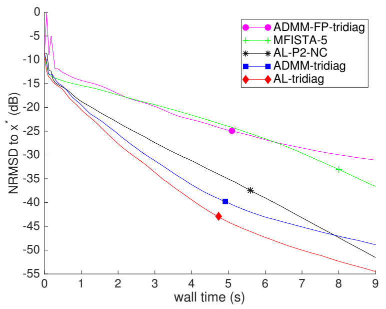
The central phase encodes were included to capture the rich information near the center of k-space. The Poisson disk sampling pattern reduces clustering of sample points in the outer regions of k-space. As with the *in vivo* experimental data, we chose to compare the proposed methods to AL-P2-NC [4] and MFISTA [6]. To measure the convergence rate of these algorithms, we computed the normalized root mean squared distance (NRMSD) between a given iterate $x^{(n)}$ and the converged solution, $x^{(\infty)}$, in terms of decibels (51).

We generated the solution, $x^{(\infty)}$, by running MFISTA for 50000 outer iterations with 5 inner NCG iterations. We solved the inner iterative update for $z$ in AL-P2-NC with preconditioned conjugate gradient using a circulant preconditioner and 1 inner iteration. We also computed the normalized root mean squared error (NRMSE) between a given iterate $x^{(n)}$ and $x_{\text{true}}$, the noiseless BrainWeb image used to generate the synthetic data. Convergence toward to the true image $x_{\text{true}}$ provides useful context for a termination condition for these iterative algorithms.

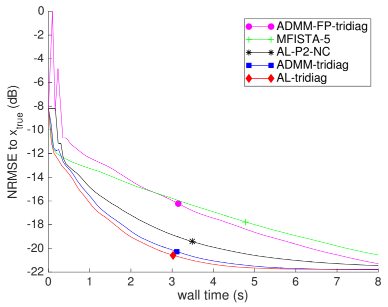
As in the *in vivo* experiments, time spent computing the maximum eigenvalue of $\mathbf{S}'\mathbf{F}'\mathbf{F}\mathbf{S}$ via power iteration, required for MFISTA, was not included in computation time. For these simulations, this computation took 60.4 seconds.



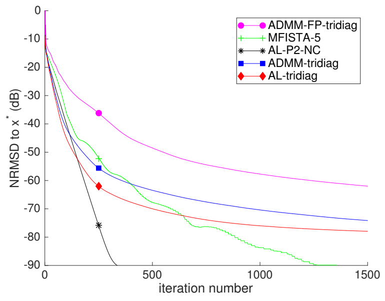
(a) NRMSD to solution $x^{(\infty)}$ as a function of computation time.



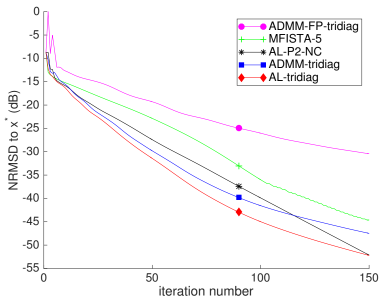
(b) Close up of (a).



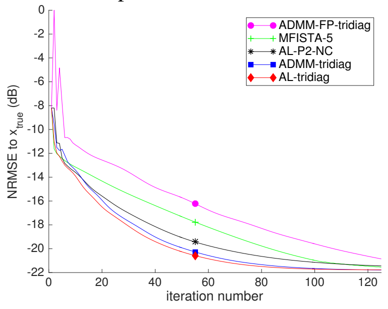
(c) NRMSE to noiseless $x_{\text{true}}$ as a function of computation time.



(d) NRMSD to solution $x^{(\infty)}$ as a function of iteration number.



(e) Close up of (d).



(f) NRMSE to noiseless $x_{\text{true}}$ as a function of iteration number.

Fig. 15: Speed comparison of ADMM-tridiag, AL-tridiag, ADMM-FP-tridiag, MFISTA, and AL-P2-NC for simulated axial data. Top row shows speed as a function of computation time. Bottom row shows speed as a function of iteration. From left to right: NRMSD to $x^{(\infty)}$, a close-up of performance over the first 100 iterations, and NRMSE to $x_{\text{true}}$.

Figure 15 shows that the fully parallelized variant, ADMM-FP-tridiag has significant computational overhead due to the additional auxiliary variables updated in each iteration. In terms of NRMSD to the solution $x^{(\infty)}$, ADMM-tridiag and AL-tridiag outperform AL-P2-NC and other methods in the first hundred iterations, having already achieved -40 dB NRMSD or better. AL-P2-NC eventually overtakes the proposed methods around -50 dB difference to $x^{(\infty)}$. By comparing distance to the noiseless BrainWeb image used for $x_{\text{true}}$, AL-tridiag and ADMM-tridiag are very competitive with AL-P2-NC, reaching similar error levels at similar computation times and iteration numbers. The proposed algorithms show little improvement in NRMSE after 10 seconds of computation or roughly 120 iterations. Further progression of image estimates to $x^{(\infty)}$ do not translate to image quality improvements. In this simulation, ADMM-FP-tridiag compares more favorably in terms of NRMSE convergence than in the *in vivo* experiments.

The blur in the initial estimate is greatly reduced after 5000 iterations of ADMM-tridiag. The reconstructed image differs from the true, fully-sampled, noiseless image, $x_{\text{true}}$, primarily at anatomical edges and centrally located

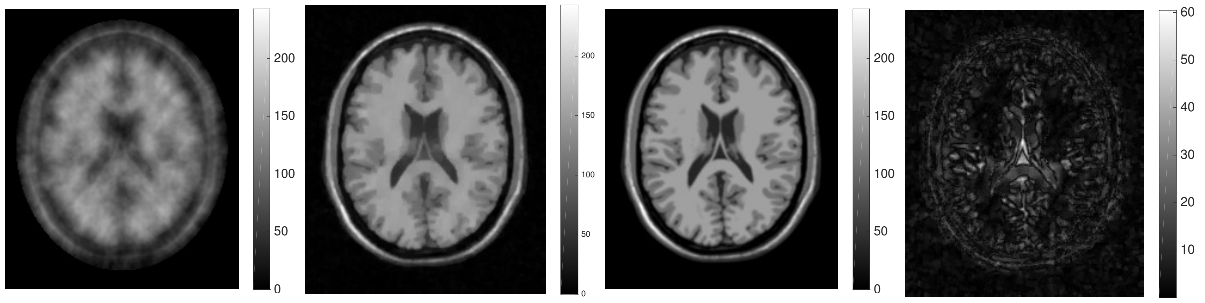


Fig. 16: Left to right: (a) sum-of-squares of the zero-filled iFFT coil images, used as an initial estimate used for all algorithms; (b) ADMM-tridiag solution $\hat{x}$ after 5000 iterations; (c) $x_{\text{true}}$ fully sampled, noiseless image; (d) difference between $\hat{x}$ and $x_{\text{true}}$ after 5000 iterations.

ventricles, as shown in Figure 16. The reconstructed image at 5000 iterations has an NRMSE of $-21.6$ dB.

## *B. Sagittal Slice Reconstruction*

To explicitly demonstrate the effect of non-periodic boundary conditions for MRI, we also reconstructed a 2D sagittal slice from simulated k-space data. We simulated noisy multi-coil data from a $T_1$-weighted 240 $\times$ 200 sagittal BrainWeb image with linear phase. Like in the axial slice experiment, we generated sensitivity maps for a 6-channel head coil array (arranged in 3 rings of 2 coils each) and generated noisy k-space data with SNR of 40.

We used the Poisson disk undersampling pattern shown in Figure 17, which has an overall reduction factor of 2, with the central 16 $\times$ 16 phase encodes fully sampled. Figure 17 also shows the simulated sensitivity maps to demonstrate the head coil geometry used for the sagittal orientation.

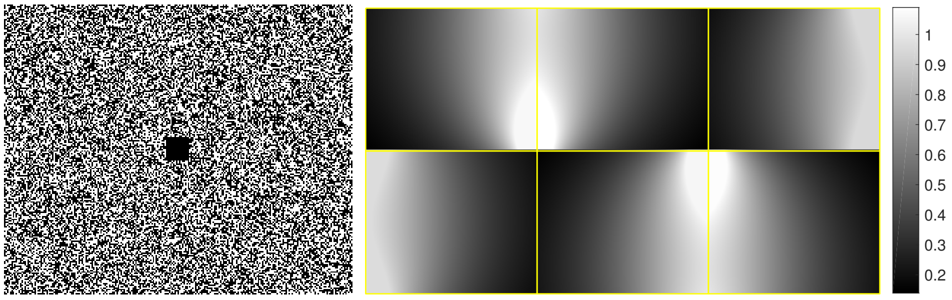


Fig. 17: 256 $\times$ 128 Poisson-disk-based undersampling pattern used for retrospective undersampling in $k_x$-$k_z$ for a sagittal slice, with reduction factor of 2 and fully sampled central 16$\times$16 phase-encodes; (b) simulated sensitivity maps showing 3 rings with two coils each

Figure 18 shows NRMSE as a function of computation time and iteration number for the proposed method and for comparison, AL-P2-NC and MFISTA. For this sagittal slice, AL-tridiag is competitive with AL-P2-NC in quickly achieving low NRMSE.

## V. Variable Splitting Balance Parameter

This section describes possible variations of the proposed algorithms in which the variable splitting scheme does not exhibit the symmetry in (3), namely,

$$\underline{\hat{u}} = \underset{\underline{u}}{\operatorname{argmin}} f(\underline{u}) \tag{79}$$

$$f(\underline{u}) = \frac{1}{2}\|y - \mathbf{F}u_2\|^2 + \lambda\|u_0\|_1 + \lambda\|u_1\|_1$$

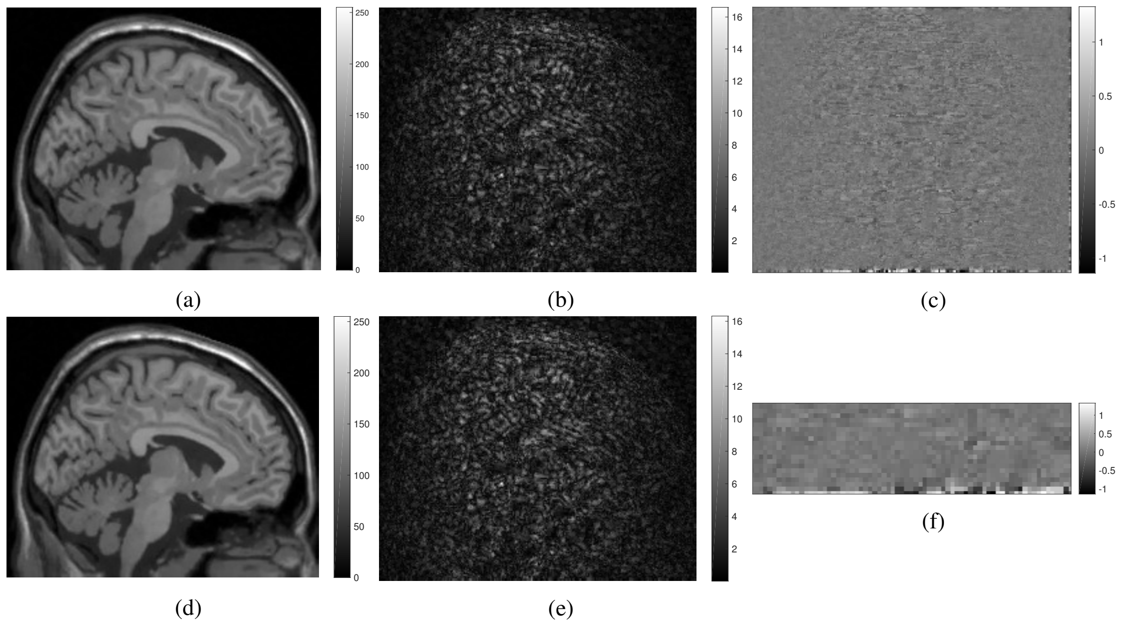
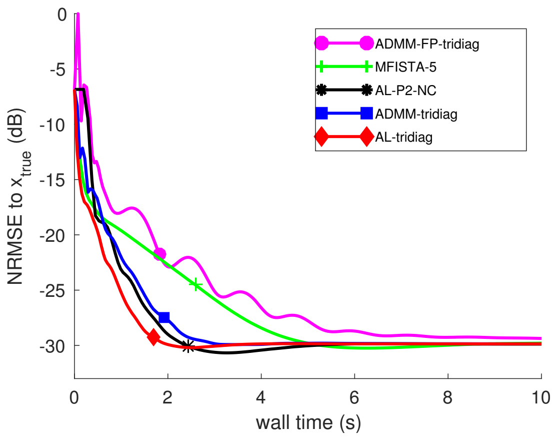
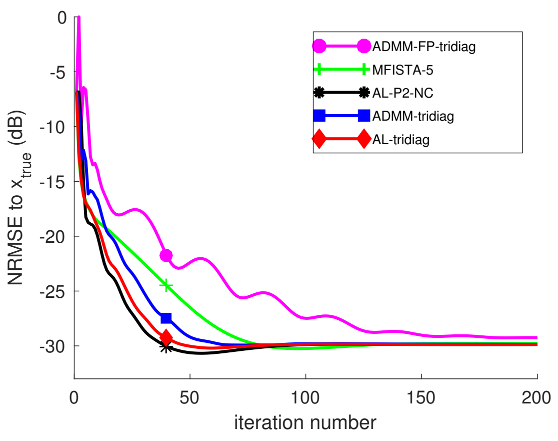


Fig. 18: Reconstructions of a retrospectively undersampled sagittal simulated dataset (a) AL-tridiag reconstruction using non-periodic boundary conditions; (b) absolute difference image between (a) and original, fully sampled image; (c) difference image between (e) and (b) (positive values show areas where circulant boundaries introduced more error); (d) AL-P2 reconstruction using periodic boundary conditions; (e) absolute difference image between (c) and original, fully sampled image; (f) boundary detail of (c)



(a) NRMSE to noiseless, fully-sampled $x_{\text{true}}$ as a function of computation time.



(b) NRMSE to noiseless, fully-sampled $x_{\text{true}}$ as a function of iteration number.

Fig. 19: Speed comparison of ADMM-tridiag, AL-tridiag, ADMM-FP-tridiag, MFISTA, and AL-P2-Nc for simulated sagittal data. For this sagittal slice reconstruction, AL-tridiag is competitive with AL-P2-NC in achieving final NRMSE as a function of computation time (a) and iteration number (b).

$$\begin{aligned} \text{s.t. } u_0 &= \mathbf{C}_{\mathrm{H}} x, & u_1 &= \mathbf{C}_{\mathrm{V}} u_3, \\ u_2 &= (1-\alpha)\mathbf{S} u_3 + \alpha \mathbf{S} x, & u_3 &= x, \end{aligned} \tag{80}$$

in which the variable splitting balance parameter, $\alpha \in [0,1]$. In principle, using different choices of $\alpha$ for AL-tridiag, such as in (79)-(80) could lead to different limits than when $\alpha = \frac{1}{2}$, due to the lack of convergence theory. However, we found empirically that varying $\alpha = \{0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\}$ in AL-tridiag-inpaint resulted in solutions identical to machine precision.

The convergence guarantees of ADMM-tridiag and ADMM-FP-tridiag ensure that the algorithm will converge to the same solution for any $\alpha \in [0,1]$. However, the choice of $\alpha$ may affect the convergence rate and is closely intertwined with the AL parameters.

Unlike the balance parameter for orthonormal wavelets, $\alpha_w$, described in Section III-D, $\alpha$ has a negligible effect on computation time per iteration. Rather, $\alpha$ affects the convergence rate by controlling the connectivity between the auxiliary variables and the extent to which one round of alternating minimization solves (6).

To investigate the effect of $\alpha$ on convergence rate, we conducted timing experiments for AL-tridiag-inpaint for varying $\alpha$. The simpler inpainting problem includes fewer additional parameters that may obfuscate the role of $\alpha$.

For the inpainting problem (33), the resulting Hessians for AL-tridiag-inpaint are:

$$\mathbf{H}_2 = (1-\alpha)^2 \mathbf{D}'\mathbf{D} + \mu_1 \tilde{\mathbf{C}}_{\mathrm{V}}' \tilde{\mathbf{C}}_{\mathrm{V}} + \mathbf{M}_2 \tag{81}$$

$$\mathbf{H}_{\mathrm{x}} = \alpha^2 \mathbf{D}'\mathbf{D} + \mu_0 \tilde{\mathbf{C}}_{\mathrm{H}}' \tilde{\mathbf{C}}_{\mathrm{H}} + \mathbf{M}_2. \tag{82}$$

Because we tune AL parameters $\mu_0$, $\mu_1$, and $\mathbf{M}_2$ to enforce $\kappa(\mathbf{H}_2) = \kappa(\mathbf{H}_{\mathrm{x}})$, this suggests that an even distribution of the influence of the inpainting operator $\mathbf{D}$ across $u_2$ and $x$ also allows for even influence of $\tilde{\mathbf{C}}_{\mathrm{H}}$ and $\tilde{\mathbf{C}}_{\mathrm{V}}$.

Figure 20 shows computation time comparisons to the MFISTA $x^{(\infty)}$ for AL-tridiag-inpaint for varying values of $\alpha$. The clear speed margin for $\alpha = 0.5$ reinforces the specific choice of $\alpha$ in [1].

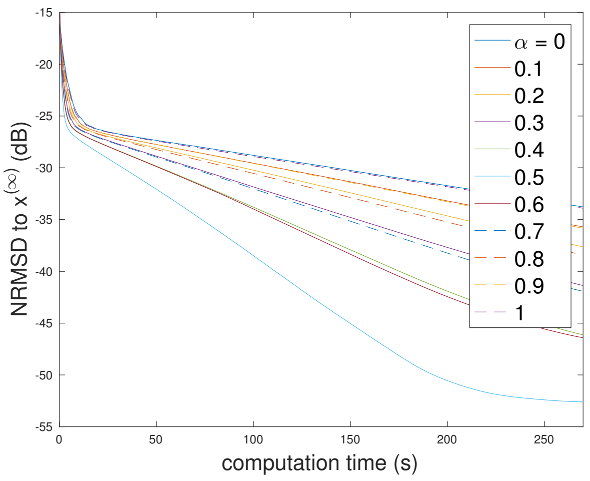


Fig. 20: Convergence of $|x^{(n)} - x^{(\infty)}|$ in dB versus computation time for AL-tridiag-inpaint as a function of balance parameter $\alpha$.

## References

[1] M. Le and J. A. Fessler, "Efficient, convergent SENSE MRI reconstruction for non-periodic boundary conditions via tridiagonal solvers," *IEEE Trans. Computational Imaging*, pp. —–, 2016.

[2] J. Eckstein, "Parallel alternating direction multiplier decomposition of convex programs," *J. Optim. Theory Appl.*, vol. 80, no. 1, pp. 39–62, Jan. 1994.

[3] S. Ramani and J. A. Fessler, "Regularized parallel MRI reconstruction using an alternating direction method of multipliers," in *Proc. IEEE Intl. Symp. Biomed. Imag.*, 2011, pp. 385–8.

[4] ——, "Parallel MR image reconstruction using augmented Lagrangian methods," *IEEE Trans. Med. Imag.*, vol. 30, no. 3, pp. 694–706, Mar. 2011.

[5] D. Dunbar and G. Humphreys, "A spatial data structure for fast Poisson-disk sample generation," *ACM Trans. on Graphics*, vol. 25, no. 3, pp. 503–8, Jul. 2006, sIGGRAPH.

[6] A. Beck and M. Teboulle, “Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems,” *IEEE Trans. Im. Proc.*, vol. 18, no. 11, pp. 2419–34, Nov. 2009.