Iterated Extended Kalman Smoother-Based Variable Splitting for L_1-Regularized State Estimation

Rui Gao, Filip Tronarp, and Simo Särkkä, *Senior Member, IEEE*

***Abstract*—In this paper, we propose a new framework for solving state estimation problems with an additional sparsity-promoting L_1-regularizer term. We first formulate such problems as minimization of the sum of linear or nonlinear quadratic error terms and an extra regularizer, and then present novel algorithms which solve the linear and nonlinear cases. The methods are based on a combination of the iterated extended Kalman smoother and variable splitting techniques such as alternating direction method of multipliers (ADMM). We present a general algorithmic framework for variable splitting methods, where the iterative steps involving minimization of the nonlinear quadratic terms can be computed efficiently by iterated smoothing. Due to the use of state estimation algorithms, the proposed framework has a low per-iteration time complexity, which makes it suitable for solving a large-scale or high-dimensional state estimation problem. We also provide convergence results for the proposed algorithms. The experiments show the promising performance and speed-ups provided by the methods.**

***Index Terms*—State estimation, sparsity, variable splitting, iterated extended Kalman smoother (IEKS), alternating direction method of multipliers (ADMM)**

I. Introduction

STATE estimation problems naturally arise in many signal processing applications including target tracking, smart grids, and robotics [1]–[3]. In conventional Bayesian approaches, the estimation task is cast as a statistical inverse problem for restoring the original time series from imperfect measurements, based on a statistical model for the measurements given the signal together with a statistical model for the signal. In linear Gaussian models, this problem admits a closed-form solution, which can be efficiently implemented by the Kalman (or Rauch–Tung–Striebel) smoother (KS) [2], [4]. For nonlinear Gaussian models we can use various linearization and sigma-point-based methods [2] for approximate inference. In particular, here we use the so-called iterated extended Kalman smoother (IEKS) [5], which is based on analytical linearisation of the nonlinear functions. Although the aforementioned smoothers are often used to estimate dynamic signals, they lack a mechanism to promote sparsity in the signals.

One approach for promoting sparsity is to add an L_1-term to the cost function formulation of the state estimation problem. This approach imposes sparsity on the signal estimate, which is either based on a *synthesis sparse* or an *analysis sparse* signal model. A synthesis sparse model assumes that the signal can be represented as a linear combination of basis vectors, where the coefficients are subject to, for example, an L_1-penalty, thus promoting sparsity. In the past decade, the use of synthesis sparsity for estimating dynamic signals has drawn a lot of attention [6]–[15]. For example, a pseudo-measurement technique was used in the Kalman update equations for encouraging sparse solutions [7]. A method based on sparsity was applied compressive sensing to update Kalman innovations or filtering errors in [8]. Based on synthesis sparsity, the estimation problem has been formulated as an L_1-regularized least square problem in [14]. Nevertheless, the previously mentioned methods typically only consider synthesis sparsity of the signal and assume a linear dynamic system.

On the other hand, analysis sparsity, also called cosparsity, assumes that the signal is not sparse itself, but rather the outcome is sparse or compressible in some transform domain, which leads to the flexibility in the modeling of signals [16]–[20]. Analysis sparse models involving an analysis operator – a popular choice being total variation (TV) – have been very successful in image processing. For example, several algorithms [18], [19] have been developed to train an analysis operator and the trained operators have been used for image denoising. In [21] the authors proposed to use the TV regularizer to improve the quality of image reconstruction. However, these approaches are not ideally suited for reconstructing dynamic signals. In state estimation problems, the available methods for analysis sparse priors are still limited. The main goal of this paper is to introduce these kinds of methods for dynamic state estimation.

Formulating a state estimation problem using synthesis and analysis sparsity leads to a general class of optimization problems, which require minimization of composite functions such as an analysis-L_1-regularized least-squares problems. The difficulties arise from the appearance of the nonsmooth regularizer. There are various batch optimization methods such as proximal gradient method [22], Douglas-Rachford splitting (DRS) [23], [24], Peaceman-Rachford splitting (PRS) [25], [26], the split Bregman method (SBM) [27], the alternating method of multipliers (ADMM) [28], [29], and the first-order primal-dual (FOPD) method for addressing this problem. However, these general methods do not take the inherent temporal nature of the optimization problem into account, which leads to bad computational and memory scaling in large-scale or high-dimensional data. This often renders the existing methods intractable due to their extensive memory and computational requirements.

As a consequence, we propose to combine a Kalman smoother with variable splitting optimization methods, which allows us to account for the temporal nature of the data in order

R. Gao, F. Tronarp and S. Särkkä are with the Department of Electrical Engineering and Automation, Aalto University, Espoo, 02150 Finland. E-mail: {rui.gao, filip.tronarp, simo.sarkka}@aalto.fi).

to speed up the computations. In this paper, we derive novel methods for efficiently estimating dynamic signals with an extra (analysis) L_1-regularized term. The developed algorithms are based on using computationally efficient KS and IEKS for solving the subproblems arising within the steps of the optimization methods. Our experiments demonstrate promising performance of the methods in practical applications. The main contributions are as follows:

i) We formulate the state estimation problem as an optimization problem that is based upon a general sparse model containing analysis or synthesis prior. The formulation accommodates a large class of popular sparsifying regularizers (e.g., synthesis L_1-norm, analysis L_1-norm, total variation norm) for state estimation.
ii) We present novel practical optimization methods, KS-ADMM and IEKS-ADMM, which are based on combining ADMM with KS and IEKS, respectively.
iii) We also prove the convergence of the KS-ADMM method as well as the local convergence of the IEKS-ADMM method.
iv) We generalize our smoother-based approaches to a general class of variable splitting techniques.

The advantage of the proposed approach is that the computational cost per iteration is much less than in the conventional batch solutions. Our approach is computationally superior to the state-of-the-art in a large-scale or high-dimensional state estimation applications.

The rest of the paper is organized as follows. We conclude this section by reviewing variable splitting methods and IEKS. Section II first develops the batch optimization by a classical ADMM method, and then presents a new KS-ADMM method for solving a linear dynamic estimation problem. Furthermore, for the nonlinear case, we present an IEKS-ADMM method in Section III and establish its local convergence properties. Section IV introduces a more general smoother-based variable splitting algorithmic framework. In particular, a general IEKS-based optimization method is formulated. Various experimental results in Section V demonstrate the effectiveness and accuracy in simulated linear and nonlinear state estimation problem. The performance of the algorithm is also illustrated in real-world tomographic reconstruction.

A. Problem Formulation

Consider the dynamic state-space model [1], [2]

$$\begin{aligned} \mathbf{x}_t &= \mathbf{a}_t(\mathbf{x}_{t-1}) + \mathbf{q}_t, \\ \mathbf{y}_t &= \mathbf{h}_t(\mathbf{x}_t) + \mathbf{r}_t, \end{aligned} \tag{1}$$

where $t = 1, \ldots, T$, $\mathbf{x}_t = \begin{bmatrix} x_{1,t} & x_{2,t} & \ldots & x_{n_x,t} \end{bmatrix}^\top \in \mathbb{R}^{n_x}$ denotes an n_x-dimensional state of the system at the time step t, and $\mathbf{y}_t = \begin{bmatrix} y_{1,t} & y_{2,t} & \ldots & y_{n_y,t} \end{bmatrix}^\top \in \mathbb{R}^{n_y}$ is an n_y-dimensional noisy measurement signal, $\mathbf{h}_t : \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ is a measurement function (typically with $n_y \leq n_x$), and $\mathbf{a}_t : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ is a state transition function at time step t. The initial state $\mathbf{x}_1$ is assumed to have mean $\mathbf{m}_1$ and covariance $\mathbf{P}_1$. The errors $\mathbf{q}_t$ and $\mathbf{r}_t$ are assumed to be mutually independent random variables with known positive definite covariance matrices $\mathbf{Q}_t$ and $\mathbf{R}_t$, respectively. The goal is to estimate the state sequence $\mathbf{x}_{1:T} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$ from the noisy measurement sequence $\mathbf{y}_{1:T} = \{\mathbf{y}_1, \ldots, \mathbf{y}_T\}$. In this paper, we focus on estimating $\mathbf{x}_{1:T}$ by minimizing the sum of quadratic error terms and an L_1 sparsity-promoting penalty.

For the sparsity assumption, we add an extra L_1-penalty for the state $\mathbf{x}_t$, and then formulate the optimization problem as

$$\begin{aligned} \mathbf{x}_{1:T}^\star = \arg\min_{\mathbf{x}_{1:T}} \frac{1}{2} \sum_{t=1}^{T} \|\mathbf{y}_t - \mathbf{h}_t(\mathbf{x}_t)\|^2_{\mathbf{R}_t^{-1}} + \lambda \sum_{t=1}^{T} \|\mathbf{\Omega}_t \mathbf{x}_t\|_1 \\ + \frac{1}{2} \sum_{t=2}^{T} \|\mathbf{x}_t - \mathbf{a}_t(\mathbf{x}_{t-1})\|^2_{\mathbf{Q}_t^{-1}} + \frac{1}{2} \|\mathbf{x}_1 - \mathbf{m}_1\|^2_{\mathbf{P}_1^{-1}}, \end{aligned} \tag{2}$$

where $\mathbf{x}_{1:T}^\star$ is the optimal state sequence, $\mathbf{\Omega}_t$ is a linear operator, and λ is a penalty parameter. The formulation (2) encompasses two particular cases: by setting $\mathbf{\Omega}_t$ to a diagonal matrix (e.g., identity matrix $\mathbf{\Omega}_t = \mathbf{I}$), a synthesis sparse model is obtained, which assumes that $\mathbf{x}_{1:T}$ are sparse. Such a case arises frequently in state estimation applications [10]–[12], [15], [30]. Correspondingly, an analysis sparse model is obtained when a more general $\mathbf{\Omega}_t$ is used. For example, total variation (TV) regularization, which is common in tomographic reconstruction, can be obtained by using a finite-difference matrix as $\mathbf{\Omega}_t$.

More generally, $\mathbf{\Omega}_t$ can be a fixed matrix [16], [20], [31] or a learned matrix [17]–[19]. It should be noted that, if the L_1 term is not used (i.e., when $\lambda = 0$) in (2), the objective can be solved by using a linear or non-linear Kalman)smoother [2], [4], [5]. However, when $\lambda > 0$, the smoothing is no longer applicable, and the cost function is non-differentiable.

Since $\|\mathbf{\Omega}_t \mathbf{x}_t\|_1$ does not have a closed-form proximal operator in general, we employ variable splitting technique for solving the resulting optimization problem. As mentioned above, many variable splitting methods can be used to solve (2), such as PRS [26], split SBM [27], ADMM, DRS [23], and FOPD [32]. Especially, ADMM is a popular member of this class. Therefore, we start by presenting algorithms based on ADMM and then extend them to more general variable splitting methods. In the following, we review variable splitting and IEKS methods, before presenting our approach in detail.

B. Variable Splitting

The methods we develop in this paper are based on variable splitting [33], [34]. Consider an unconstrained optimization problem in which the objective function is the sum of two functions

$$\min_{\mathbf{x}} \theta_1(\mathbf{x}) + \theta_2(\mathbf{\Omega}\mathbf{x}), \tag{3}$$

where $\theta_2(\cdot) = \|\cdot\|_1$, and $\mathbf{\Omega}$ is a matrix. Variable splitting refers to the process of introducing an auxiliary constrained variable $\mathbf{w}$ to separate the components in the cost function. More specifically, we impose the constraint $\mathbf{w} = \mathbf{\Omega}\mathbf{x}$, which transforms the original minimization problem (3) into an equivalent constrained minimization problem given by

$$\min_{\mathbf{x},\mathbf{w}} \theta_1(\mathbf{x}) + \theta_2(\mathbf{w}), \quad \text{s.t.} \quad \mathbf{w} = \mathbf{\Omega}\mathbf{x}. \tag{4}$$

The minimization problem (4) can be solved efficiently by classical constrained optimization methods [35]. The rationale of variable splitting is that it may be easier to solve the constrained problem (4) than the unconstrained one (3). PRS, SBM, FOPD, ADMM, and their variants [36] are a few well-known variable splitting methods – see also [37], [38] for a recent historical overview.

ADMM [28] is one of the most popular algorithms for solving (4). ADMM defines an augmented Lagrangian function, and then alternates between the updates of the split variables. Given $\mathbf{x}^{(0)}$, $\mathbf{w}^{(0)}$, and $\boldsymbol{\eta}^{(0)}$, its iterative steps are:

$$\begin{aligned}\mathbf{x}^{(k+1)} &= \arg\min_{\mathbf{x}} \theta_1(\mathbf{x}) + (\boldsymbol{\eta}^{(k)})^\top(\mathbf{w}^{(k)} - \boldsymbol{\Omega}\mathbf{x}) \\ &\quad + \frac{\rho}{2}\|\mathbf{w}^{(k)} - \boldsymbol{\Omega}\mathbf{x}\|^2, \qquad (5a)\\ \mathbf{w}^{(k+1)} &= \arg\min_{\mathbf{w}} \theta_2(\mathbf{w}) + (\boldsymbol{\eta}^{(k)})^\top(\mathbf{w} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)}) \\ &\quad + \frac{\rho}{2}\|\mathbf{w} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)}\|^2, \qquad (5b)\\ \boldsymbol{\eta}^{(k+1)} &= \boldsymbol{\eta}^{(k)} + \rho(\mathbf{w}^{(k+1)} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)}), \qquad (5c)\end{aligned}$$

where $\boldsymbol{\eta}$ is a Lagrange multiplier and ρ is a parameter.

The PRS method [25], [26] is similar to ADMM except that it updates the Lagrange multiplier twice. The typical iterative steps for (3) are

$$\begin{aligned}\mathbf{x}^{(k+1)} &= \arg\min_{\mathbf{x}} \theta_1(\mathbf{x}) + (\boldsymbol{\eta}^{(k)})^\top(\mathbf{w}^{(k)} - \boldsymbol{\Omega}\mathbf{x}) \\ &\quad + \frac{\rho}{2}\|\mathbf{w}^{(k)} - \boldsymbol{\Omega}\mathbf{x}\|^2, \qquad (6a)\\ \boldsymbol{\eta}^{(k+\frac{1}{2})} &= \boldsymbol{\eta}^{(k)} + \alpha\rho(\mathbf{w}^{(k)} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)}), \qquad (6b)\\ \mathbf{w}^{(k+1)} &= \arg\min_{\mathbf{w}} \theta_2(\mathbf{w}) + (\boldsymbol{\eta}^{(k+\frac{1}{2})})^\top(\mathbf{w} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)}) \\ &\quad + \frac{\rho}{2}\|\mathbf{w} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)}\|^2, \qquad (6c)\\ \boldsymbol{\eta}^{(k+1)} &= \boldsymbol{\eta}^{(k+\frac{1}{2})} + \alpha\rho(\mathbf{w}^{(k+1)} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)}), \qquad (6d)\end{aligned}$$

where $\alpha \in (0, 1)$.

In SBM [27], we iterate the steps

$$\begin{aligned}\mathbf{x}^{(k+1)} &= \arg\min_{\mathbf{x}} \theta_1(\mathbf{x}) + \frac{\rho}{2}\|\mathbf{w}^{(k)} - \boldsymbol{\Omega}\mathbf{x} + \boldsymbol{\eta}^{(k)}\|_2^2, \qquad (7a)\\ \mathbf{w}^{(k+1)} &= \arg\min_{\mathbf{w}} \theta_2(\mathbf{w}) + \frac{\rho}{2}\|\mathbf{w} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)} + \boldsymbol{\eta}^{(k)}\|_2^2, \qquad (7b)\end{aligned}$$

M times, and update the extra variable by

$$\boldsymbol{\eta}^{(k+1)} = \boldsymbol{\eta}^{(k)} + (\mathbf{w}^{(k+1)} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)}). \qquad (8)$$

When $M = 1$, this is equivalent to ADMM.

There are also other variable splitting methods which alternate proximal steps for the primal and dual variables. One example is FOPD [32], where the kth iteration consists of the following subproblems

$$\begin{aligned}\mathbf{x}^{(k+1)} &= \arg\min_{\mathbf{x}} \theta_1(\mathbf{x}) + \frac{1}{2\rho}\|\mathbf{x} - (\mathbf{x}^{(k)} - \rho\boldsymbol{\Omega}^\top\mathbf{w}^{(k)})\|^2, \qquad (9a)\\ \hat{\mathbf{x}}^{(k+1)} &= \mathbf{x}^{(k+1)} + \tau(\mathbf{x}^{(k+1)} - \hat{\mathbf{x}}^{(k)}), \qquad (9b)\\ \mathbf{w}^{(k+1)} &= \arg\min_{\mathbf{w}} \theta_2^*(\mathbf{w}) + \frac{1}{2\gamma}\|\mathbf{w} - (\mathbf{w}^{(k)} + \gamma\boldsymbol{\Omega}\hat{\mathbf{x}}^{(k+1)})\|^2, \qquad (9c)\end{aligned}$$

where $\theta_2^*(\mathbf{w}) = \langle \boldsymbol{\Omega}\mathbf{x}, \mathbf{w}\rangle - \theta_2(\boldsymbol{\Omega}\mathbf{x})$, and $\tau and \gamma$ are parameters.

All these variable splitting algorithms provide simple ways to construct efficient iterative algorithms that offer simpler inner subproblems. However, the subproblems such as (5a), (6a), (6a) and (9a) remain computationally expensive, as they involve large matrix-vector products when the dimensionality of $\mathbf{x}$ is large. We circumvent this problem by combining variable splitting with KS and IEKS.

C. The Iterated Extended Kalman Smoother

IEKS [5] is an approximative algorithm for solving nonlinear optimal smoothing problems. However, it can also be seen [5] as an efficient implementation of the Gauss–Newton algorithm for solving the problem

$$\begin{aligned}\mathbf{x}_{1:T}^\star = \arg\min_{\mathbf{x}_{1:T}} &\frac{1}{2}\sum_{t=1}^{T}\|\mathbf{y}_t - \mathbf{h}_t(\mathbf{x}_t)\|^2_{\mathbf{R}_t^{-1}} \\ &+ \frac{1}{2}\sum_{t=2}^{T}\|\mathbf{x}_t - \mathbf{a}_t(\mathbf{x}_{t-1})\|^2_{\mathbf{Q}_t^{-1}} + \frac{1}{2}\|\mathbf{x}_1 - \mathbf{m}_1\|^2_{\mathbf{P}_1^{-1}}.\end{aligned} \qquad (10)$$

That is, it produces the maximum a posteriori (MAP) estimate of the trajectory. The IEKS method works by alternating between linearisation of $\mathbf{a}_t$ and $\mathbf{h}_t$ around a previous estimate $\mathbf{x}_{1:T}^{(i)}$, as follows:

$$\begin{aligned}\mathbf{a}_t(\mathbf{x}_{t-1}) &\approx \mathbf{a}_t(\mathbf{x}_{t-1}^{(i)}) + \mathbf{J}_{a_t}(\mathbf{x}_{t-1}^{(i)})(\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^{(i)}), \qquad (11a)\\ \mathbf{h}_t(\mathbf{x}_t) &\approx \mathbf{h}_t(\mathbf{x}_t^{(i)}) + \mathbf{J}_{h_t}(\mathbf{x}_t^{(i)})(\mathbf{x}_t - \mathbf{x}_t^{(i)}), \qquad (11b)\end{aligned}$$

and solving the linearized problem

$$\begin{aligned}&\mathbf{x}_{1:T}^{(i+1)} = \\ &\arg\min_{\mathbf{x}_{1:T}} \frac{1}{2}\sum_{t=1}^{T}\|\mathbf{y}_t - \mathbf{h}_t(\mathbf{x}_t^{(i)}) - \mathbf{J}_{h_t}(\mathbf{x}_t^{(i)})(\mathbf{x}_t - \mathbf{x}_t^{(i)})\|^2_{\mathbf{R}_t^{-1}} \\ &+ \frac{1}{2}\sum_{t=2}^{T}\|\mathbf{x}_t - \mathbf{a}_t(\mathbf{x}_{t-1}^{(i)}) - \mathbf{J}_{a_t}(\mathbf{x}_{t-1}^{(i)})\|^2_{\mathbf{Q}_t^{-1}}.\end{aligned} \qquad (12)$$

where $\mathbf{J}_\phi$ is the Jacobian of $\phi(\mathbf{x})$. The solution of (12) can in turn be efficiently obtained by the Rauch–Tung–Striebel (RTS) smoother [4], which first computes the filtering mean and covariances $\mathbf{m}_{1:T}$ and $\mathbf{P}_{1:T}$, by alternating between prediction

$$\begin{aligned}\mathbf{m}_t^- &= \mathbf{a}_t(\mathbf{x}_{t-1}^{(i)}) + \mathbf{J}_{a_t}(\mathbf{x}_{t-1}^{(i)})(\mathbf{m}_{t-1} - \mathbf{x}_{t-1}^{(i)}), \qquad (13a)\\ \mathbf{P}_t^- &= \mathbf{J}_{a_t}(\mathbf{x}_{t-1}^{(i)})\mathbf{P}_{t-1}[\mathbf{J}_{a_t}(\mathbf{x}_{t-1}^{(i)})]^\top + \mathbf{Q}_t, \qquad (13b)\end{aligned}$$

and update

$$\begin{aligned}\mathbf{S}_t &= \mathbf{J}_{h_t}(\mathbf{x}_t^{(i)})\mathbf{P}_t^-[\mathbf{J}_{h_t}(\mathbf{x}_t^{(i)})]^\top + \mathbf{R}_t, \qquad (14a)\\ \mathbf{K}_t &= \mathbf{P}_t^-[\mathbf{J}_{h_t}(\mathbf{x}_t^{(i)})]^\top[\mathbf{S}_t]^{-1}, \qquad (14b)\\ \mathbf{m}_t &= \mathbf{m}_t^- + \mathbf{K}_t\Big(\mathbf{y}_t - \mathbf{h}_t(\mathbf{x}_t^{(i)}) - \mathbf{J}_{h_t}(\mathbf{x}_t^{(i)})(\mathbf{m}_t^- - \mathbf{x}_t^{(i)})\Big), \qquad (14c)\\ \mathbf{P}_t &= \mathbf{P}_t^- - \mathbf{K}_t\mathbf{S}_t[\mathbf{K}_t]^\top, \qquad (14d)\end{aligned}$$

where $\mathbf{S}_t$ and $\mathbf{K}_t$ are the innovation covariance matrix and the Kalman gain at the time step t, respectively. The filtering

means $\mathbf{m}_t$ and covariances $\mathbf{P}_t$ are then corrected in a backwards (smoothing) pass

$$\mathbf{G}_t = \mathbf{P}_t[\mathbf{J}_{a_t}(\mathbf{x}_{t-1}^{(i)})]^\top[\mathbf{P}_{t+1}^-]^{-1}, \tag{15a}$$

$$\mathbf{m}_t = \mathbf{m}_t + \mathbf{G}_t\Big(\mathbf{m}_{t+1}^s - \mathbf{m}_{t+1}^-\Big), \tag{15b}$$

$$\mathbf{P}_t^s = \mathbf{P}_t + \mathbf{G}_t\Big(\mathbf{P}_{t+1}^s - \mathbf{P}_{t+1}^-\Big)[\mathbf{G}_t]^\top. \tag{15c}$$

Now setting $\mathbf{x}_t^{(i+1)} = \mathbf{m}_t^s$ gives the solution to (12). When the functions $\mathbf{a}_t$ and $\mathbf{h}_t$ are linear, the above iteration converges in a single step. This algorithm is the classical RTS smoother or more briefly KS [4].

In this paper, we use the KS and IEKS algorithms as efficient methods for solving generalized versions of the optimization problems given in (10), which arise within the steps of variable splitting.

II. Linear State Estimation by KS-ADMM

In this section, we present the KS-ADMM algorithm which is a novel algorithm for solving L_1-regularized linear Gaussian state estimation problems. In particular, Section II-A describes the batch solution by ADMM. Then, by defining an artificial measurement noise and a pseudo-measurement, we formulate the KS-ADMM algorithm to solve the primal variable update in Section II-B.

A. Batch Optimization

Let us assume that the state transition function $\mathbf{a}_t$ and the measurement function $\mathbf{h}_t$ are linear, denoted by

$$\mathbf{a}_t(\mathbf{x}_{t-1}) = \mathbf{A}_t\mathbf{x}_{t-1}, \quad \mathbf{h}_t(\mathbf{x}_t) = \mathbf{H}_t\mathbf{x}_t, \tag{16}$$

where $\mathbf{A}_t$ and $\mathbf{H}_t$ are the transition matrix and the measurement matrix, respectively. In order to reduce this problem to (3), we stack the entire state sequence into a vector, which transforms the objective into a batch optimization problem. Thus, we define the following variables

$$\mathbf{x} = \mathrm{vec}(\mathbf{x}_1, \ldots, \mathbf{x}_T), \tag{17a}$$

$$\mathbf{y} = \mathrm{vec}(\mathbf{y}_1, \ldots, \mathbf{y}_T), \tag{17b}$$

$$\mathbf{m} = \mathrm{vec}(\mathbf{m}_1, \mathbf{0}, \ldots, \mathbf{0}), \tag{17c}$$

$$\mathbf{H} = \mathrm{blkdiag}(\mathbf{H}_1, \ldots, \mathbf{H}_T), \tag{17d}$$

$$\mathbf{Q} = \mathrm{blkdiag}(\mathbf{P}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_T), \tag{17e}$$

$$\mathbf{R} = \mathrm{blkdiag}(\mathbf{R}_1, \ldots, \mathbf{R}_T), \tag{17f}$$

$$\mathbf{\Omega} = \mathrm{blkdiag}(\mathbf{\Omega}_1, \ldots, \mathbf{\Omega}_T), \tag{17g}$$

$$\mathbf{\Psi} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & & \\ -\mathbf{A}_2 & \mathbf{I} & \ddots & \\ & \ddots & \ddots & \mathbf{0} \\ & & -\mathbf{A}_T & \mathbf{I} \end{pmatrix}. \tag{17h}$$

The optimization problem introduced in Section I-A can now be reformulated as the following batch optimization problem

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}} \frac{1}{2}\left\|\mathbf{y} - \mathbf{H}\mathbf{x}\right\|_{\mathbf{R}^{-1}}^2 + \frac{1}{2}\left\|\mathbf{\Psi}\mathbf{x} - \mathbf{m}\right\|_{\mathbf{Q}^{-1}}^2 + \lambda\left\|\mathbf{\Omega}\mathbf{x}\right\|_1, \tag{18}$$

which in turn can be seen to be a special case of (3). Here, our algorithm for solving (18) builds upon the batch ADMM [28].

To derive an ADMM algorithm for (18), we introduce an auxiliary variable $\mathbf{w} = \mathrm{vec}(\mathbf{w}_1, \ldots, \mathbf{w}_T)$ and a linear equality constraint $\mathbf{w} = \mathbf{\Omega}\mathbf{x}$. The resulting equality-constrained problem is formulated mathematically as

$$\begin{aligned} &\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_{\mathbf{R}^{-1}}^2 + \frac{1}{2}\|\mathbf{\Psi}\mathbf{x} - \mathbf{m}\|_{\mathbf{Q}^{-1}}^2 + \lambda\|\mathbf{w}\|_1 \\ &\text{s.t. } \mathbf{w} = \mathbf{\Omega}\mathbf{x}. \end{aligned} \tag{19}$$

The main objective here is to find a stationary point $(\mathbf{x}^\star, \mathbf{w}^\star, \boldsymbol{\eta}^\star)$ of the augmented Lagrangian function associated with (19) as the function

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \mathbf{w}; \boldsymbol{\eta}) &\triangleq \frac{1}{2}\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_{\mathbf{R}^{-1}}^2 + \lambda\|\mathbf{w}\|_1 \\ &+ \frac{1}{2}\|\mathbf{\Psi}\mathbf{x} - \mathbf{m}\|_{\mathbf{Q}^{-1}}^2 + \boldsymbol{\eta}^\top(\mathbf{w} - \mathbf{\Omega}\mathbf{x}) + \frac{\rho}{2}\|\mathbf{w} - \mathbf{\Omega}\mathbf{x}\|^2, \end{aligned} \tag{20}$$

where $\boldsymbol{\eta} \in \mathbb{R}^{TP}$ is the dual variable and ρ is a penalty parameter. As described in Section I-B, at each iteration of ADMM we perform the updates

$$\mathbf{x}^{(k+1)} = \arg\min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}), \tag{21a}$$

$$\mathbf{w}^{(k+1)} = \arg\min_{\mathbf{w}} \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}; \boldsymbol{\eta}^{(k)}), \tag{21b}$$

$$\boldsymbol{\eta}^{(k+1)} = \boldsymbol{\eta}^{(k)} + \rho(\mathbf{w}^{(k+1)} - \mathbf{\Omega}\mathbf{x}^{(k+1)}). \tag{21c}$$

The update for the primal sequence $\mathbf{x}$ is equivalent to the quadratic optimization problem given by

$$\begin{aligned} \mathbf{x}^{(k+1)} = \arg\min_{\mathbf{x}} &\frac{1}{2}\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_{\mathbf{R}^{-1}}^2 + \frac{1}{2}\|\mathbf{\Psi}\mathbf{x} - \mathbf{m}\|_{\mathbf{Q}^{-1}}^2 \\ &+ \frac{\rho}{2}\|\mathbf{w} - \mathbf{\Omega}\mathbf{x} + \boldsymbol{\eta}/\rho\|^2, \end{aligned} \tag{22}$$

which has the closed-form solution

$$\begin{aligned} \mathbf{x}^{(k+1)} &= \left[\mathbf{H}^\top\mathbf{R}^{-1}\mathbf{H} + \mathbf{\Psi}^\top\mathbf{Q}^{-1}\mathbf{\Psi} + \rho\,\mathbf{\Omega}^\top\mathbf{\Omega}\right]^{-1} \\ &\times \left[\mathbf{H}^\top\mathbf{R}^{-1}\mathbf{y} + \mathbf{\Psi}^\top\mathbf{Q}^{-1}\mathbf{m} + \rho\,\mathbf{\Omega}^\top(\mathbf{w}^{(k)} + \boldsymbol{\eta}^{(k)}/\rho)\right]. \end{aligned} \tag{23}$$

For the dual sequence $\mathbf{w}$, the iteration in (21b) can be solved by [39]

$$\mathbf{w}^{(k+1)} = \mathbf{e}^{(k)} - \mathrm{sgn}(\mathbf{e}^{(k)}) \circ \max(|\mathbf{e}^{(k)}| - \lambda/\rho, 0), \tag{24}$$

where $\mathbf{e}^{(k)} = \mathbf{\Omega}\mathbf{x}^{(k+1)} + \boldsymbol{\eta}^{(k)}/\rho$.

While the optimization problem (22) can be solved in closed-form, direct solution is computationally demanding, especially when the number of time points or the dimensionality of the state is large. However, the problem can be recognized to be a special case of optimization problems where the iterations can be solved by KS (see Section I-C) provided that we add pseudo-measurements to the problem. In the following, we present the resulting algorithm.

B. The KS-ADMM Solver

The proposed KS-ADMM solver is described in Algorithm 1. To extend the batch ADMM to KS-ADMM, we first

define an artificial measurement noise $\mathbf{\Sigma}_t = \mathbf{I}/\rho$ and a pseudo-measurement $\mathbf{z}_t = \mathbf{w}_t + \boldsymbol{\eta}_t/\rho$, and then rewrite (22) as

$$\begin{aligned}\min_{\mathbf{x}_{1:T}} \frac{1}{2}\sum_{t=1}^{T} \|\mathbf{y}_t - \mathbf{H}_t\mathbf{x}_t\|^2_{\mathbf{R}_t^{-1}} + \frac{1}{2}\sum_{t=1}^{T} \|\mathbf{z}_t - \mathbf{\Omega}_t\mathbf{x}_t\|^2_{\mathbf{\Sigma}_t^{-1}} \\ + \frac{1}{2}\sum_{t=2}^{T} \|\mathbf{x}_t - \mathbf{A}_t\mathbf{x}_{t-1}\|^2_{\mathbf{Q}_t^{-1}} + \frac{1}{2}\|\mathbf{x}_1 - \mathbf{m}_1\|^2_{\mathbf{P}_1^{-1}}.\end{aligned} \tag{25}$$

The solution to (25) can then be computed by running KS on the state estimation problem

$$\mathbf{x}_t = \mathbf{A}_t\mathbf{x}_{t-1} + \mathbf{q}_t, \tag{26a}$$
$$\mathbf{y}_t = \mathbf{H}_t\mathbf{x}_t + \mathbf{r}_t, \tag{26b}$$
$$\mathbf{z}_t = \mathbf{\Omega}_t\mathbf{x}_t + \boldsymbol{\sigma}_t. \tag{26c}$$

Here, $\boldsymbol{\sigma}_t$ is an independent random variable with covariance $\mathbf{\Sigma}_t$. The KS-based solution can be described as a four stage recursive process: prediction, $\mathbf{y}_t$-update, $\mathbf{z}_t$-update, and the RTS smoother which should be performed for $t = 1, \ldots, T$. First, the prediction step is given by

$$\mathbf{m}_t^- = \mathbf{A}_t\mathbf{m}_{t-1}, \tag{27a}$$
$$\mathbf{P}_t^- = \mathbf{A}_t\,\mathbf{P}_{t-1}\,\mathbf{A}_t^\top + \mathbf{Q}_t, \tag{27b}$$

where $\mathbf{m}_t^-$ and $\mathbf{P}_t^-$ are the predicted mean and covariance at the time t. Secondly, the update steps for $\mathbf{y}_t$ are given by

$$\mathbf{S}_t^y = \mathbf{H}_t\,\mathbf{P}_t^-\,\mathbf{H}_t^\top + \mathbf{R}_t, \tag{28a}$$
$$\mathbf{K}_t^y = \mathbf{P}_t^-\,\mathbf{H}_t^\top\,[\mathbf{S}_t^y]^{-1}, \tag{28b}$$
$$\mathbf{m}_t^y = \mathbf{m}_t^- + \mathbf{K}_t^y[\mathbf{y}_t - \mathbf{H}_t\,\mathbf{m}_t^-], \tag{28c}$$
$$\mathbf{P}_t^y = \mathbf{P}_t^- - \mathbf{K}_t^y\,\mathbf{S}_t^y\,[\mathbf{K}_t^y]^\top. \tag{28d}$$

Thirdly, the update steps for $\mathbf{z}_t$ are

$$\mathbf{S}_t^z = \mathbf{\Omega}_t\,\mathbf{P}_t^y\,\mathbf{\Omega}_t^\top + \mathbf{\Sigma_t}, \tag{29a}$$
$$\mathbf{K}_t^z = \mathbf{P}_t^-\,\mathbf{\Omega}_t^\top[\mathbf{S}_t^z]^{-1}, \tag{29b}$$
$$\mathbf{m}_t = \mathbf{m}_t^y + \mathbf{K}_t^z[\mathbf{z}_t - \mathbf{\Omega_t}\mathbf{m}_t^y], \tag{29c}$$
$$\mathbf{P}_t = \mathbf{P}_t^y - \mathbf{K}_t^z\,\mathbf{S}_t^z\,[\mathbf{K}_t^z]^\top. \tag{29d}$$

Here, $\mathbf{S}_t^y$ and $\mathbf{S}_t^z$, $\mathbf{K}_t^y$ and $\mathbf{K}_t^z$, $\mathbf{m}_t^y$ and $\mathbf{m}_t$, $\mathbf{P}_t^y$ and $\mathbf{P}_t$ are the innovation covariances, gain matrices, means, and covariances for the variables $\mathbf{y}_t$ and $\mathbf{z}_t$ at the time step t, respectively. Finally, we run a RTS smoother [4] for $t = T-1, \ldots, 1$, which has the steps

$$\mathbf{G}_t = \mathbf{P}_t\,\mathbf{A}_{t+1}^\top\,[\mathbf{P}_{t+1}^-]^{-1}, \tag{30a}$$
$$\mathbf{m}_t^s = \mathbf{P}_t + \mathbf{G}_t\,[\mathbf{m}_{t+1}^s - \mathbf{m}_{t+1}^-], \tag{30b}$$
$$\mathbf{P}_t^s = \mathbf{P}_t + \mathbf{G}_t\,[\mathbf{P}_{t+1}^s - \mathbf{P}_{t+1}^-]\,\mathbf{G}_t^\top, \tag{30c}$$

where $\mathbf{m}_T^s = \mathbf{m}_T$ and $\mathbf{P}_T^s = \mathbf{P}_T$ (see [2] for more details). This gives the update for $\mathbf{x}_{1:T}$ as:

$$\mathbf{x}_{1:T}^{(k+1)} = \mathbf{m}_{1:T}^s. \tag{31}$$

The remaining updates for $t = 1, \ldots, T$ are given as

$$\mathbf{w}_t^{(k+1)} = \mathbf{e}_t^{(k)} - \mathrm{sgn}(\mathbf{e}_t^{(k)}) \circ \max(|\mathbf{e}_t^{(k)}| - \lambda/\rho, 0), \tag{32}$$

where $\mathbf{e}_t^{(k)} = \mathbf{\Omega}_t\mathbf{x}_t^{(k+1)} + \boldsymbol{\eta}_t^{(k)}/\rho$, and

$$\boldsymbol{\eta}_t^{(k+1)} = \boldsymbol{\eta}_t^{(k)} + \rho(\mathbf{w}_t^{(k+1)} - \mathbf{\Omega}_t\mathbf{x}_t^{(k+1)}). \tag{33}$$

Algorithm 1: KS-ADMM

Input: $\mathbf{y}_t$, $\mathbf{H}_t$, $\mathbf{A}_t$, $\mathbf{Q}_t$, $\mathbf{R}_t$, $\mathbf{\Omega}_t$, $t = 1, \ldots, T$; parameters λ and ρ; $\mathbf{m}_1$ and $\mathbf{P}_1$.

Output: $\mathbf{x}_{1:T}$.

while *not convergent* **do**
- run the Kalman filter using (27), (28), and (29);
- run the Kalman smoother by using (30);
- compute $\mathbf{x}_{1:T}$ by (31);
- compute $\mathbf{w}_{1:T}$ by (32);
- compute $\boldsymbol{\eta}_{1:T}$ by (33);

end

It is useful to note that in Algorithm 1, the covariances and gains are independent of the iteration number and thus can be pre-computed outside the ADMM iterations. Furthermore, when the model is time-independent, we can often use stationary Kalman filters and smoothers instead of their general counterparts which can further be used to speed up the computations.

C. Convergence of KS-ADMM

In this section, we discuss the convergence of KS-ADMM. If the system (26) is detectable [40], then the objective function (20) is convex. The traditional convergence results for ADMM such as in [28], [41] then ensure that the objective globally converges to the stationary (optimal) point $(\mathbf{x}_{1:T}^\star, \mathbf{w}_{1:T}^\star, \boldsymbol{\eta}_{1:T}^\star)$. The result is given in the following.

Theorem 1 (Convergence of KS-ADMM)**.** *Assume that the system* (26) *is detectable [40]. Then, for a constant ρ, the sequence $\{\mathbf{x}_{1:T}^{(k)}, \mathbf{w}_{1:T}^{(k)}, \boldsymbol{\eta}_{1:T}^{(k)}\}$ generated by Algorithm 1 from any starting point $\{\mathbf{x}_{1:T}^{(0)}, \mathbf{w}_{1:T}^{(0)}, \boldsymbol{\eta}_{1:T}^{(0)}\}$ converges to the stationary point $\{\mathbf{x}_{1:T}^\star, \mathbf{w}_{1:T}^\star, \boldsymbol{\eta}_{1:T}^\star\}$ of* (20)*.*

Proof. Due to the detectability assumption, the objective function is convex, and thus the result follows from the classical ADMM convergence proof [28], [41]. ■

III. Nonlinear State Estimation by IEKS-ADMM

When $\mathbf{a}_t$ and $\mathbf{h}_t$ are nonlinear, the $\mathbf{x}$ subproblem arising in the ADMM iteration cannot be solved in closed form. In the following, we first present a batch solution of the nonlinear case based on a Gauss–Newton (GN) iteration and then show how it can be efficiently implemented using IEKS.

A. Batch Optimization

Let us now consider the case where the state transition function $\mathbf{a}_t$ and the measurement function $\mathbf{h}_t$ in (1) are nonlinear. We now proceed to rewrite the optimization (2) in batch form by defining the following variables

$$\mathbf{a}(\mathbf{x}) = \mathrm{vec}(\mathbf{x}_1, \mathbf{x}_2 - \mathbf{a}_2(\mathbf{x}_1), \ldots, \mathbf{x}_T - \mathbf{a}_T(\mathbf{x}_{T-1})), \tag{34a}$$
$$\mathbf{h}(\mathbf{x}) = \mathrm{vec}(\mathbf{h}_1(\mathbf{x}_1), \ldots, \mathbf{h}_T(\mathbf{x}_T)). \tag{34b}$$

Note that the variables $\mathbf{x}$, $\mathbf{y}$, $\mathbf{m}$, $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{\Omega}$ have the same definitions as (17). Using these variables, the $\mathbf{x}$ subproblem can be naturally transformed into

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}} \frac{1}{2}\left\|\mathbf{y}-\mathbf{h}(\mathbf{x})\right\|^2_{\mathbf{R}^{-1}} + \frac{1}{2}\left\|\mathbf{m}-\mathbf{a}(\mathbf{x})\right\|^2_{\mathbf{Q}^{-1}} + \lambda\left\|\mathbf{\Omega x}\right\|_1, \tag{35}$$

which is also a special case of (3), similarly to the linear case.

Following the ADMM, we define the augmented Lagrangian function associated with (35) as:

$$\mathcal{L}(\mathbf{x},\mathbf{w};\boldsymbol{\eta}) \triangleq \frac{1}{2}\left\|\mathbf{y}-\mathbf{h}(\mathbf{x})\right\|^2_{\mathbf{R}^{-1}} + \lambda\|\mathbf{w}\|_1 + \frac{1}{2}\left\|\mathbf{m}-\mathbf{a}(\mathbf{x})\right\|^2_{\mathbf{Q}^{-1}} + \boldsymbol{\eta}^\top(\mathbf{w}-\mathbf{\Omega x}) + \frac{\rho}{2}\|\mathbf{w}-\mathbf{\Omega x}\|^2. \tag{36}$$

Since the nonlinear batch solution is based on ADMM, the iteration steps of $\mathbf{w}$ and $\boldsymbol{\eta}$ are the same with the linear case (see Equations (24) and (21c)). Here, we focus on introducing the solution of the primal variable $\mathbf{x}$.

When updating $\mathbf{x}$, the objective is no longer a quadratic function. However, the optimization problem can be solved with GN [42]. Here, the $\mathbf{x}$ subproblem is rewritten as

$$\min_{\mathbf{x}} f(\mathbf{x}), \tag{37}$$

where

$$f(\mathbf{x}) = \frac{1}{2}\|\mathbf{R}^{-\frac{1}{2}}(\mathbf{y}-\mathbf{h}(\mathbf{x}))\|^2 + \frac{1}{2}\|\mathbf{Q}^{-\frac{1}{2}}(\mathbf{m}-\mathbf{a}(\mathbf{x}))\|^2 + \frac{\rho}{2}\|\mathbf{w}-\mathbf{\Omega x}+\boldsymbol{\eta}/\rho\|^2.$$

Then, the gradient of $f(\mathbf{x})$ is given by

$$\nabla f(\mathbf{x}) = \begin{bmatrix}\mathbf{R}^{-\frac{1}{2}}\mathbf{J}_h(\mathbf{x})\\ \mathbf{Q}^{-\frac{1}{2}}\mathbf{J}_a(\mathbf{x})\\ \rho^{\frac{1}{2}}\mathbf{\Omega}\end{bmatrix}^\top \begin{bmatrix}\mathbf{R}^{-\frac{1}{2}}(\mathbf{h}(\mathbf{x})-\mathbf{y})\\ \mathbf{Q}^{-\frac{1}{2}}(\mathbf{a}(\mathbf{x})-\mathbf{m})\\ \rho^{\frac{1}{2}}(\mathbf{\Omega x}-\mathbf{w}-\boldsymbol{\eta}/\rho)\end{bmatrix}, \tag{38}$$

where

$$\mathbf{J}_h(\mathbf{x}) = \text{blkdiag}(\mathbf{J}_{h_1},\ldots,\mathbf{J}_{h_T}),$$

$$\mathbf{J}_a(\mathbf{x}) = \begin{pmatrix}\mathbf{I} & \mathbf{0} & & \\ -\mathbf{J}_{a_2} & \mathbf{I} & \ddots & \\ & \ddots & \ddots & \mathbf{0}\\ & & -\mathbf{J}_{a_T} & \mathbf{I}\end{pmatrix},$$

and the Hessian is $\nabla^2 f(\mathbf{x}) = \mathbf{J}^\top\mathbf{J}(\mathbf{x}) + \mathbf{H}(\mathbf{x})$, where

$$\mathbf{J}^\top\mathbf{J}(\mathbf{x}) = \mathbf{J}_h^\top\mathbf{R}^{-1}\mathbf{J}_h(\mathbf{x}) + \mathbf{J}_a^\top\mathbf{Q}^{-1}\mathbf{J}_a(\mathbf{x}) + \rho\mathbf{\Omega}^\top\mathbf{\Omega},$$

$$[\mathbf{H}(\mathbf{x})]_{ij} = \frac{1}{2}(\mathbf{h}(\mathbf{x})-\mathbf{y})^\top\mathbf{R}^{-1}\frac{\partial^2\mathbf{h}(\mathbf{x})}{\partial x_i\,\partial x_j} + \frac{1}{2}(\mathbf{a}(\mathbf{x})-\mathbf{m})^\top\mathbf{Q}^{-1}\frac{\partial^2\mathbf{a}(\mathbf{x})}{\partial x_i\,\partial x_j}.$$

In GN, avoiding the trouble of computing the residual $[\mathbf{H}(\mathbf{x})]_{ij}$, we use the approximation $\nabla^2 f(\mathbf{x}) \approx \mathbf{J}^\top\mathbf{J}(\mathbf{x})$ to replace $\nabla^2 f(\mathbf{x})$, which means $[\mathbf{H}(\mathbf{x})]_{ij}$ is assumed to be small enough. Thus, the primal variable in $\mathbf{x}$ iteration is updated by:

$$\mathbf{x}^\star = \left[\mathbf{J}_h^\top\mathbf{R}^{-1}\mathbf{J}_h(\mathbf{x}) + \mathbf{J}_a^\top\mathbf{Q}^{-1}\mathbf{J}_a(\mathbf{x}) + \rho\,\mathbf{\Omega}^\top\mathbf{\Omega}\right]^{-1} \times \left[\mathbf{J}_h^\top\mathbf{R}^{-1}(\mathbf{y}-\mathbf{h}(\mathbf{x})) + \mathbf{J}_a^\top\mathbf{Q}^{-1}(\mathbf{m}-\mathbf{a}(\mathbf{x})) + \rho\,\mathbf{\Omega}^\top(\mathbf{w}^{(k)}+\boldsymbol{\eta}^{(k)}/\rho)\right]. \tag{40}$$

The iterations can stopped after a maximum number of iterations $i_{\max}$ or if the condition $\|\mathbf{x}^{(i+1)}-\mathbf{x}^{(i)}\|_2 \le \varepsilon$ is satisfied, where ε is an error tolerance. If ε is small enough, then it means that the above algorithm has (almost) converged. The rest of the ADMM updates can be implemented similarly to the linear Gaussian case.

B. The IEKS-ADMM Solver

We now move on to consider the IEKS-ADMM solver. As discussed in Section I-C, IEKS can be seen as an efficient implementation of the GN method, which inspires us to derive an efficient implementation of the batch ADMM.

Now, we rewrite the $\mathbf{x}$ subproblem (37) as

$$\min_{\mathbf{x}_{1:T}} \frac{1}{2}\sum_{t=1}^{T}\|\mathbf{y}_t-\mathbf{h}_t(\mathbf{x}_t)\|^2_{\mathbf{R}_t^{-1}} + \frac{1}{2}\sum_{t=1}^{T}\|\mathbf{z}_t-\mathbf{\Omega}_t\mathbf{x}_t\|^2_{\mathbf{\Sigma}_t^{-1}} + \frac{1}{2}\sum_{t=2}^{T}\|\mathbf{x}_t-\mathbf{a}_t(\mathbf{x}_{t-1})\|^2_{\mathbf{Q}_t^{-1}} + \frac{1}{2}\|\mathbf{x}_1-\mathbf{m}_1\|^2_{\mathbf{P}_1^{-1}}. \tag{41}$$

In a modest scale (e.g., $T \approx 10^3$), $\mathbf{x}_{1:T}$ can be directly computed by (40) although its computations scale as $\mathcal{O}(T^3)$. When T is large, the batch ADMM will have high memory and computational requirements. In this case, the use of IEKS becomes beneficial due to its linear computational scaling. In this paper, the proposed method incorporates IEKS into ADMM to design the IEKS-ADMM algorithm for solving the nonlinear case.

In the IEKS algorithm, the Gaussian smoother is run several times with $\mathbf{a}_t$ and $\mathbf{h}_t$ and their Jacobians are evaluated at the previous (inner loop) iteration. The detailed iteration steps of IEKS-ADMM are described in Algorithm 2. In particular, following the prediction steps (13) in Section I-C, the update steps for $\mathbf{y}_t$ are given by

$$\mathbf{S}_t^y = \mathbf{J}_{h_t}(\mathbf{x}_t^{(i)})\,\mathbf{P}_t^-\,[\mathbf{J}_{h_t}(\mathbf{x}_t^{(i)})]^\top + \mathbf{R}_t, \tag{42a}$$

$$\mathbf{K}_t^y = \mathbf{P}_t^-\,[\mathbf{J}_{h_t}(\mathbf{x}_t^{(i)})]^\top\,[\mathbf{S}_t^y]^{-1}, \tag{42b}$$

$$\mathbf{m}_t^y = \mathbf{m}_t^- + \mathbf{K}_t^y[\mathbf{y}_t - \mathbf{h}_t(\mathbf{x}_t^{(i)}) - \mathbf{J}_{h_t}(\mathbf{x}_t^{(i)})(\mathbf{m}_t^- - \mathbf{x}_t^{(i)})], \tag{42c}$$

$$\mathbf{P}_t^y = \mathbf{P}_t^- - \mathbf{K}_t^y\,\mathbf{S}_t^y\,[\mathbf{K}_t^y]^\top, \tag{42d}$$

and for the pseudo-measurement $\mathbf{z}_t$, the update steps are the same as in the linear case. They are given in (29).

Additionally, the RTS smoother steps are also described in Section I-C. We can then obtain the solution as $\mathbf{x}_{1:T}^{(k+1)} = \mathbf{m}_{1:T}^s$. Note that the updates on $\mathbf{w}_{1:T}$ and $\boldsymbol{\eta}_{1:T}$ can be implemented by (32) and (33), respectively.

C. Convergence of IEKS-ADMM

In this section, our aim is to prove the convergence of the IEKS-ADMM algorithm. Although we can much rely on existing convergence results, unfortunately, when $\mathbf{a}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$ are nonlinear, the traditional convergence analysis [28], [43]–[45] does not work as such. In particular, Jacobian matrices $\mathbf{J}_a$, $\mathbf{J}_h$ and $\mathbf{\Omega}$ in this paper are possibly rank-deficient, which is not covered by the existing convergence

Algorithm 2: IEKS-ADMM

Input: $\mathbf{y}_t$, $\mathbf{h}_t$, $\mathbf{a}_t$, $\mathbf{Q}_t$, $\mathbf{R}_t$, $\mathbf{\Omega}_t$, $t = 1, \ldots, T$; λ and ρ; $\mathbf{m}_1$ and $\mathbf{P}_1$.

Output: $\mathbf{x}_{1:T}$

while *not convergent* **do**
- compute $\mathbf{x}_{1:T}$ by using the IEKS;
- compute $\mathbf{w}_{1:T}$ by (32);
- compute $\boldsymbol{\eta}_{1:T}$ by (33);

end

results. In the following, we will establish the convergence analysis which also covers this case.

For notational convenience, we define $\theta_1(\mathbf{x}) = \frac{1}{2}\|\mathbf{y} - \mathbf{h}(\mathbf{x})\|^2_{\mathbf{R}^{-1}} + \frac{1}{2}\|\mathbf{m} - \mathbf{a}(\mathbf{x})\|^2_{\mathbf{Q}^{-1}}$ and $\theta_2(\mathbf{w}) = \lambda\|\mathbf{w}\|_1$. The variables $\mathbf{x}$ and $\mathbf{w}$ are two sets of time series, and $\theta_1(\mathbf{x})$ is a non-quadratic, possibly nonconvex function. The corresponding augmented Lagrangian function can be rewritten as

$$\begin{aligned}\mathcal{L}(\mathbf{x}, \mathbf{w}; \boldsymbol{\eta}) = \theta_1(\mathbf{x}) + \theta_2(\mathbf{w}) \\ + \boldsymbol{\eta}^\top(\mathbf{w} - \mathbf{\Omega}\mathbf{x}) + \frac{\rho}{2}\|\mathbf{w} - \mathbf{\Omega}\mathbf{x}\|^2,\end{aligned} \tag{43}$$

where $\mathbf{\Omega}$ can be full-row rank or full-column rank. Thus, the convergence is analyzed in two different cases. We make the following assumptions.

Assumption 1. *The gradient $\nabla\theta_1(\mathbf{x})$ is Lipschitz continuous with constant L_{θ_1}, that is,*

$$\|\nabla\theta_1(\mathbf{x}_1) - \nabla\theta_1(\mathbf{x}_2)\| \leq L_{\theta_1}\|\mathbf{x}_1 - \mathbf{x}_2\|, \forall \mathbf{x}_1, \mathbf{x}_2 \in dom(\theta_1).$$

Assumption 2. *Function $\theta_1(\mathbf{x}) + \theta_2(\mathbf{w})$ is lower bounded and coercive over the feasible set $\{(\mathbf{x}, \mathbf{w}) : \mathbf{w} = \mathbf{\Omega}\mathbf{x}\}$.*

First, we prove that the sequence $\mathcal{L}(\mathbf{x}^k, \mathbf{w}^k; \boldsymbol{\eta}^k)$ is monotonically non-increasing in the following lemma.

Lemma 1 (Nonincreasing sequence)**.** *Let Assumptions 1 and 2 be satisfied and $\{\mathbf{x}^{(k)}, \mathbf{w}^{(k)}, \boldsymbol{\eta}^{(k)}\}$ be the iterative sequence generated by ADMM. Assume that one of two cases is satisfied:*

Case (a): *There exists ρ_0 such that when $\rho > \rho_0$, $\mathbf{x} \mapsto \mathcal{L}(\mathbf{x}, \mathbf{w}; \boldsymbol{\eta})$ is μ_x-strongly convex, that is, $\mathcal{L}(\mathbf{x}, \mathbf{w}; \boldsymbol{\eta})$ satisfies*

$$\begin{aligned}&\mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) - \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) \\ &\geq \langle \nabla\mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}), \mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}\rangle \\ &\quad + \frac{\mu_x}{2}\|\mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}\|^2.\end{aligned} \tag{44}$$

Furthermore, assume that $\rho > \max\left(\frac{2L^2_{\theta_1}}{\kappa_a^2\mu_x}, \rho_0\right)$ and that $\mathbf{\Omega}$ has full row rank with

$$\mathbf{\Omega}\mathbf{\Omega}^\top \succeq \kappa_a^2\mathbf{I}, \quad \kappa_a > 0. \tag{45}$$

Case (b): *$\rho > \frac{L_{\theta_1}}{\kappa_b^2}$, and $\mathbf{\Omega}$ has full-column rank with*

$$\mathbf{\Omega}^\top\mathbf{\Omega} \succeq \kappa_b^2\mathbf{I}, \quad \kappa_b > 0. \tag{46}$$

Then, sequence $\mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)})$ is nonincreasing in k.

Proof. See Appendix A. ∎

Next we prove the convergence of Algorithm 2. For that we need a couple of lemmas which are presented in the following.

Lemma 2 (Convergence of GN)**.** . *Let $\nabla f(\mathbf{x})$ be Lipschitz continuous with constant L_f, $\mathbf{H}(\mathbf{x})$ be bounded by a constant, that is, $\|\mathbf{H}(\mathbf{x})\| \leq \epsilon_h$. If $\mathbf{J}^\top\mathbf{J}(\mathbf{x}^{(i)}) \succeq \mu^2\mathbf{I}$ where $\mu > 0$ is a constant, and $\epsilon_h < \mu^2$, then the sequence $\mathbf{x}^{(i)}$ converges to a local minimum $\mathbf{x}^\star$. In particular, the convergence is quadratic when $\epsilon_h \to 0$, and (at least) linear convergence is obtained when $\epsilon_h < \mu^2$.*

Proof. See Appendix B. ∎

Based on Lemmas 1 and 2, we can now establish the convergence by GN-ADMM in the following lemma.

Lemma 3 (Convergence of GN-ADMM)**.** *Let assumptions of Lemmas 1 and 2 be satisfied. Then, the sequence $\{\mathbf{x}^{(k)}, \mathbf{w}^{(k)}, \boldsymbol{\eta}^{(k)}\}$ generated by GN-ADMM algorithm converges to a local minimum $(\mathbf{x}^\star, \mathbf{w}^\star, \boldsymbol{\eta}^\star)$.*

Proof. By Lemma 1, the sequence $\mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)})$ is nonincreasing in k. By Assumption 2, the sequence $\{\mathbf{x}^{(k)}, \mathbf{w}^{(k)}, \boldsymbol{\eta}^{(k)}\}$ is bounded, because $\mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)})$ is upper bounded by $\mathcal{L}(\mathbf{x}^{(0)}, \mathbf{w}^{(0)}; \boldsymbol{\eta}^{(0)})$ and nonincreasing. It is also lower bounded by

$$\mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) \geq \theta_1(\mathbf{x}^{(k)}) + \theta_2(\mathbf{w}^{(k)}). \tag{47}$$

By Lemma 2, there exists a local minimum $\mathbf{x}^\star$ such that the sequence $\mathbf{x}^{(i)}$ converges to $\mathbf{x}^\star$, which is a local minimum of $\mathbf{x}$ subproblem. The $\mathbf{w}$ subproblem is convex [46] and thus there exists a unique minimum $\mathbf{w}^\star$. We then deduce that the iterative sequence $\{\mathbf{x}^{(k)}, \mathbf{w}^{(k)}, \boldsymbol{\eta}^{(k)}\}$ generated by GN-ADMM converges to $(\mathbf{x}^\star, \mathbf{w}^\star, \boldsymbol{\eta}^\star)$. ∎

The equivalence of GN and IEKS can now be used to show that IEKS-ADMM converges to a local minimum $(\mathbf{x}^\star_{1:T}, \mathbf{w}^\star_{1:T}, \boldsymbol{\eta}^\star_{1:T})$.

Theorem 2 (Convergence of IEKS-ADMM)**.** *If the sequence $\{\mathbf{x}^{(k)}, \mathbf{w}^{(k)}, \boldsymbol{\eta}^{(k)}\}$ generated by GN-ADMM algorithm converges to a local minimum $(\mathbf{x}^\star, \mathbf{w}^\star, \boldsymbol{\eta}^\star)$, then the sequence $\{\mathbf{x}^{(k)}_{1:T}, \mathbf{w}^{(k)}_{1:T}, \boldsymbol{\eta}^{(k)}_{1:T}\}$ generated by IEKS-ADMM algorithm converges to the local minimum $(\mathbf{x}^\star_{1:T}, \mathbf{w}^\star_{1:T}, \boldsymbol{\eta}^\star_{1:T})$.*

Proof. According to [47], the sequence $\{\mathbf{x}^{(k)}, \mathbf{w}^{(k)}, \boldsymbol{\eta}^{(k)}\}$ generated by the GN method and the sequence $\{\mathbf{x}^{(k)}_{1:T}, \mathbf{w}^{(k)}_{1:T}, \boldsymbol{\eta}^{(k)}_{1:T}\}$ generated by IEKS are identical. Based on Lemma 3, we deduce the iterative sequence $\{\mathbf{x}^{(k)}_{1:T}, \mathbf{w}^{(k)}_{1:T}, \boldsymbol{\eta}^{(k)}_{1:T}\}$ generated by IEKS-ADMM is locally convergent to $(\mathbf{x}^\star_{1:T}, \mathbf{w}^\star_{1:T}, \boldsymbol{\eta}^\star_{1:T})$. ∎

IV. Extension to general algorithmic Framework

In this section, we present a general algorithmic framework based on the combination of the extended Kalman smoother and variable splitting. As the smoother solution only applies to the $\mathbf{x}_{1:T}$-subproblem, here we only formulate the corresponding $\mathbf{x}_{1:T}$-subproblem which can be solved with the extended Kalman smoother. The different variants in the proposed framework are distinguished by three different choices: the pseudo-measurement, the pseudo-measurement covariance, and the pseudo-measurement model matrix.

When $\mathbf{a}_t$ and $\mathbf{h}_t$ are linear functions, we have the following general objective function for the $\mathbf{x}_{1:T}$-subproblem:

$$\min_{\mathbf{x}_{1:T}} \frac{1}{2}\sum_{t=1}^{T}\|\mathbf{y}_t-\mathbf{H}_t\mathbf{x}_t\|^2_{\mathbf{R}_t^{-1}}+\frac{1}{2}\sum_{t=1}^{T}\|\mathbf{x}_t-\mathbf{A}_t\mathbf{x}_{t-1}\|^2_{\mathbf{Q}_t^{-1}} + \frac{1}{2}\|\mathbf{x}_1-\mathbf{m}_1\|^2_{\mathbf{P}_1^{-1}}+\frac{1}{2}\sum_{t=1}^{T}\|\mathbf{\Delta}_t-\mathbf{\Theta}_t\mathbf{x}_t\|^2_{\mathbf{\Sigma}_t^{-1}}, \tag{48}$$

and when $\mathbf{a}_t$ and $\mathbf{h}_t$ are nonlinear functions, we have

$$\min_{\mathbf{x}_{1:T}} \frac{1}{2}\sum_{t=1}^{T}\|\mathbf{y}_t-\mathbf{h}_t(\mathbf{x}_t)\|^2_{\mathbf{R}_t^{-1}}+\frac{1}{2}\sum_{t=1}^{T}\|\mathbf{x}_t-\mathbf{a}_t(\mathbf{x}_{t-1})\|^2_{\mathbf{Q}_t^{-1}} + \frac{1}{2}\|\mathbf{x}_1-\mathbf{m}_1\|^2_{\mathbf{P}_1^{-1}}+\frac{1}{2}\sum_{t=1}^{T}\|\mathbf{\Delta}_t-\mathbf{\Theta}_t\mathbf{x}_t\|^2_{\mathbf{\Sigma}_t^{-1}}. \tag{49}$$

In the above objective functions, $\mathbf{\Delta}_t$ is the pseudo-measurement, $\mathbf{\Sigma}_t$ is the pseudo-measurement covariance, and $\mathbf{\Theta}_t$ is the pseudo-measurement model matrix.

As mentioned in Section I-B, various variable splitting such as PRS, SBM, and FOPD can be used to solve the problems (22) and (37). Their KS / IEKS-based counterparts can be obtained by selecting the aforementioned pseudo-measurement model parameters as shown in Table I. The algorithms for solving the optimization problems are then the same as Algorithms 1 and 2 except that the pseudo-measurement updates in Equation (29) are replaced with

$$\mathbf{S}_t^{\delta} = \mathbf{\Theta}_t\,\mathbf{P}^{y}\,\mathbf{\Theta}_t^{\top}+\mathbf{\Sigma}_t, \tag{50a}$$
$$\mathbf{K}_t^{\delta} = \mathbf{P}_t^{-}\,\mathbf{\Theta}_t^{\top}\,[\mathbf{S}_t^{\delta}]^{-1}, \tag{50b}$$
$$\mathbf{m}_t = \mathbf{m}_t^{y}+\mathbf{K}_t^{\delta}\,[\mathbf{\Delta}_t-\mathbf{\Theta}_t\,\mathbf{m}_t^{y}], \tag{50c}$$
$$\mathbf{P}_t = \mathbf{P}_t^{y}-\mathbf{K}_t^{\delta}\,\mathbf{S}_t^{\delta}\,[\mathbf{K}_t^{\delta}]^{\top}, \tag{50d}$$

and the updates of the other variables are performed using the appropriate algorithm (see Section I-B).

V. Numerical Experiments

In the following, we demonstrate the KS and IEKS based variable splitting methods in numerical experiments. We first provide simulated results to study the behavior of the proposed methods with respect to convergence speed and the computational efficiency. We then demonstrate the effectiveness of the methodology in a tomographic reconstruction task.

A. Linear Gaussian Simulation Experiment

Consider a four-dimensional linear tracking model (see, e.g., [2]) where the state contains rectangular coordinates x_1 and x_2, and velocity variables x_3 and x_4. The state of the system at time step t is $\mathbf{x}_t = \begin{bmatrix} x_{1,t} & x_{2,t} & x_{3,t} & x_{4,t}\end{bmatrix}^{\top}$. The transition and measurement model matrices are

$$\mathbf{A}_t = \begin{bmatrix} 1 & 0 & \triangle t & 0 \\ 0 & 1 & 0 & \triangle t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{H}_t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

The matrix $\mathbf{\Omega}_t$ and the covariance for the transition are

$$\mathbf{\Omega}_t = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{Q}_t = q_c \begin{bmatrix} \frac{\Delta t^3}{3} & 0 & \frac{\Delta t^2}{2} & 0 \\ 0 & \frac{\Delta t^3}{3} & 0 & \frac{\Delta t^2}{2} \\ \frac{\Delta t^2}{2} & 0 & \Delta t & 0 \\ 0 & \frac{\Delta t^2}{2} & 0 & \Delta t \end{bmatrix}.$$

with $q_c = 1/2$, $\Delta t = 0.1$, and the measurement noise covariance $\mathbf{R}_t = \sigma^2\mathbf{I}$ with $\sigma = 0.2$. We set the parameters to $\lambda = 1$ and $T = 100$, and run $k_{\max} = 10$ iterations of ADMM. The goal here is to estimate dynamic signals from the noisy measurements $\mathbf{y}_{1:T}$.

In this experiment, we compare the convergence speed (relative error versus iteration number) and the running time (average CPU time versus iteration number) generated by batch versions of PRS [26], SBM [27], FOPD [32], ADMM [28], to the proposed KS-based variable splitting methods. We also evaluate the performance without adding an extra analysis-L_1-regularization term (i.e., $\lambda = 0$) in which case the optimization problem (25) can be solved by KS. Here, the relative error is calculated by

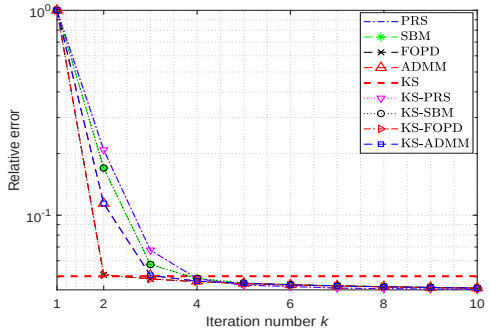
$$\frac{\sum_{t=1}^{T}\|\mathbf{x}_t^{(k)}-\mathbf{x}_t^{\text{true}}\|_2}{\sum_{t=1}^{T}\|\mathbf{x}_t^{\text{true}}\|_2}, \tag{51}$$

where $\mathbf{x}_t^{\text{true}}$ is the ground truth at time step t and $\mathbf{x}_t^{(k)}$ is the kth iterate at time step t. Fig. 1 (a) shows the number of iterations required to solve the estimation problem. After the $k_{\max}$ steps, all the methods have roughly the same relative errors. Fig. 1 (b) shows the average CPU time as function of number of iterations. Note that in order to speed up the KS-based variable splitting methods, we compute the gains $\mathbf{K}_t^{y}$, $\mathbf{K}_t^{z}$ and $\mathbf{G}_t$ only at the first iteration, and use the precomputated matrices in the following iterations. Although PRS and KS-PRS, SBM and KS-SBM, FOPD and KS-FOPD, ADMM and KS-ADMM have the same convergence speed, not surprisingly, KS-SBM, KS-FOPD and KS-ADMM have a lower CPU time. When $T = 100$, KS and the KS-based variable splitting methods have similar CPU time, but KS has a worse relative error. As shown in Fig. 1 (b), running the PRS, SBM, FOPD and ADMM solvers is time-consuming. The KF-based variable splitting methods take about 0.03 seconds to reach 10 iterations, while the classical optimization approaches such as FOPD and ADMM take 7 times longer.

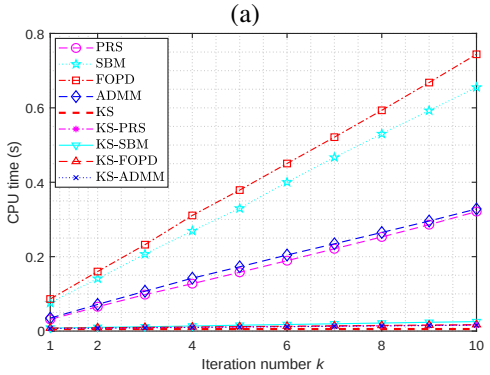
The benefit of our approach is highlighted by the fact that the methods can efficiently solve a large-scale dynamic signal estimation problem with an extra L_1-regularized term. Next, we enlarge the time step count from 10^3 to 10^8, and show the results in Fig. 2. We use 10 iterations for each method, which in practice is enough for convergence. It can be seen that the proposed method significantly outperforms other batch solutions with respect to the consumed CPU time. In particular, the PRS, SBM, FOPD and ADMM solvers with 10^4 time steps take more time than the proposed methods with $T = 10^8$. When $T = 10^5, 10^6, 10^7, 10^8$, the computing operations on PRS, SBM, FOPD, and ADMM run out of memory, and the related results cannot be reported. This is mainly because the KS-based variable splitting methods deal with the objective using recursive computations, which

TABLE I
DIFFERENT CHOICES OF IEKS-BASED VARIABLE SPLITTING ALGORITHMS AT EVERY ITERATION k

Method	Related Quadratic Term	$\boldsymbol{\Theta}_t$	$\boldsymbol{\Delta}_t$	$\boldsymbol{\Sigma}_t$
PRS	$\frac{\rho}{2}\|\mathbf{w}_t - \boldsymbol{\Omega}_t\mathbf{x}_t + \boldsymbol{\eta}_t/\rho\|^2$	$\boldsymbol{\Omega}_t$	$\mathbf{w}_t + \boldsymbol{\eta}_t/\rho$	$\mathbf{I}/\rho$
SBM	$\frac{\rho}{2}\|\mathbf{w}_t - \boldsymbol{\Omega}_t\mathbf{x}_t + \boldsymbol{\eta}_t\|^2$	$\boldsymbol{\Omega}_t$	$\boldsymbol{\eta}_t + \mathbf{w}_t$	$\rho\mathbf{I}$
FOPD	$\frac{1}{2\rho}\|\mathbf{x}_t - (\mathbf{x}_t^{(k)} - \boldsymbol{\Omega}_t^\top\mathbf{w}_t)\|^2$	$\mathbf{I}$	$\mathbf{x}_t^{(k)} - \boldsymbol{\Omega}_t^\top\mathbf{w}_t^{(k)}$	$\rho\mathbf{I}$
ADMM	$\frac{\rho}{2}\|\mathbf{w}_t - \boldsymbol{\Omega}_t\mathbf{x}_t + \boldsymbol{\eta}_t/\rho\|^2$	$\boldsymbol{\Omega}_t$	$\mathbf{w}_t + \boldsymbol{\eta}_t/\rho$	$\mathbf{I}/\rho$



(a)



(b)

Fig. 1. Performance as function of iteration number k with $T = 100$ in the linear experiment: (a) the relative error versus iteration number, with the y-axis in log-scale, (b) the average CPU time versus iteration number.

significantly reduces the computational and memory burden, while the batch optimization methods explicitly deal with large vectors and matrices.

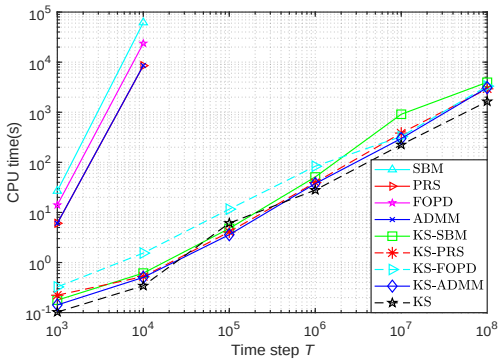


Fig. 2. The average CPU time (seconds) versus time step counts 10^3, 10^4, 10^5, 10^6, 10^7 and 10^8. The axes are in log-scale.

Table III summarizes the running times with different methods when the number of time steps T is varying. The table reports the CPU time in seconds required by each solver after 10 iterations. Not surprisingly, the KF-based variable splitting methods (i.e., KF-PRS, KF-SBM, KF-FOPD, and KF-ADMM) are faster than the batch variable splitting methods. For example, when $T = 10^8$, KF-ADMM takes 4631 seconds to achieve the convergence. The number of iterations of KF-PRS, KF-SBM, KF-FOPD, and KF-ADMM are the same as the number of iterations of the batch optimization methods, while their computing time for every iteration is around 10 times less.

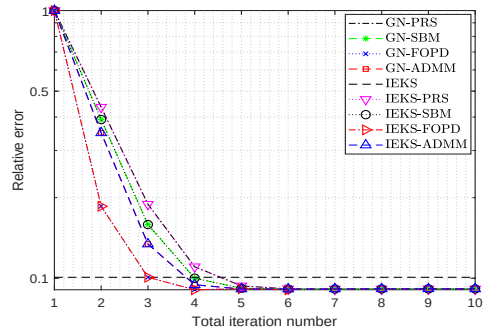
TABLE II
COMPARISON OF AVERAGE CPU TIME (SECONDS) WITH DIFFERENT METHODS IN THE LINEAR CASE.

$\boldsymbol{T}$	$\mathbf{10^3}$	$\mathbf{10^4}$	$\mathbf{10^5}$	$\mathbf{10^6}$	$\mathbf{10^7}$	$\mathbf{10^8}$
PRS	6.06	852	-	-	-	-
SBM	26.79	6178	-	-	-	-
FOPD	14.04	2376	-	-	-	-
ADMM	6.06	857	-	-	-	-
KS	0.16	0.34	6	27	224	1639
KS-PRS	0.22	0.52	29	39	383	2936
KS-SBM	0.17	0.61	31	51	912	3961
KS-FOPD	0.32	1.54	11	83	326	3328
KS-ADMM	0.14	0.50	9	37	298	3096

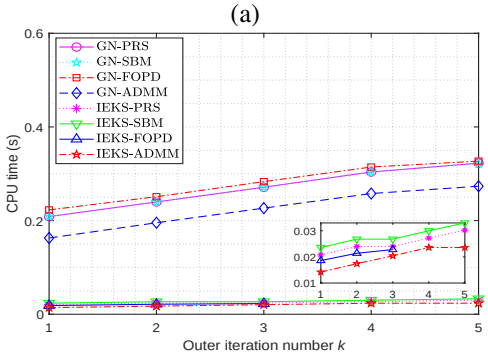
B. Nonlinear Simulation Experiment

In this section, we consider a five-dimensional nonlinear coordinated turn model [1]. We set the parameters to $\lambda = 1$ and $T = 80$, run $k_{\max} = 10$ iterations of all the optimization methods. We compare IEKS [5], GN-PRS and IEKS-PRS, GN-SBM and IEKS-SBM, GN-FOPD, and IEKS-FOPD, GN-ADMM, and IEKS-ADMM by plotting the relative error and the CPU time as functions of the iteration number. Fig. 3 demonstrates the efficiency of our IEKS-based variable splitting methods against GN-PRS, GN-SBM, GN-FOPD and GN-ADMM in the same experiment. The horizontal axis in Fig. 3 (a) describes total iteration number, and the vertical axis gives the relative errors. As can be seen, all the methods give the fast convergence in around 5 iterations.

We are also interested in the performance without adding an extra analysis-L_1-regularized term (i.e., $\lambda = 0$). Since there is no outer iteration in IEKS, we plot the relative error of IEKS after the inner iteration (dashed black line in Fig. 3 (b)). In the average CPU time, IEKS-PRS, IEKS-SBM, IEKS-FOPD, and IEKS-ADMM are clearly superior to batch variable splitting (see Fig. 3 (b)). IEKS-FOPD is the fastest convergent method,



(a)



(b)

Fig. 3. Performance in the coordinated turn model. (a) Relative error versus total iteration number. The y-axis is in log-scale. (b) The average CPU time (seconds) versus outer iteration number k.

needing only 3 iterations. Although the state estimation problem is relatively small scale, GN-PRS, GN-SBM, GN-FOPD and GN-ADMM are still time-consuming.

Additionally, we enlarge the time step count T from 10^2 to 10^7, and plot the results in Fig. 4. When increasing the time step count, our proposed methods significantly outperform the batch methods with respect to CPU time. In particular, the PRS, SBM, FOPD and ADMM solvers with 10^4 time steps take more time than our proposed method with 10^7 time steps. In Table III, we list the CPU time with different methods when T is varying. When $T = 10^5, 10^6, 10^7$, the computer operations on GN–PRS, GN-SBM, GN-FOPD, and GN-ADMM run out of memory.

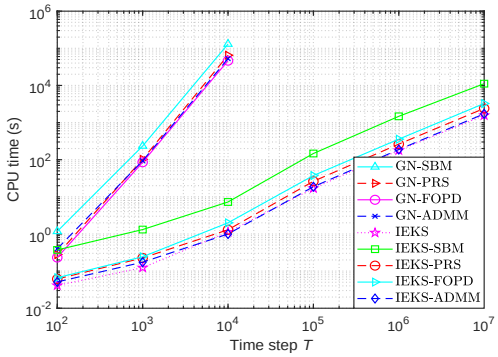


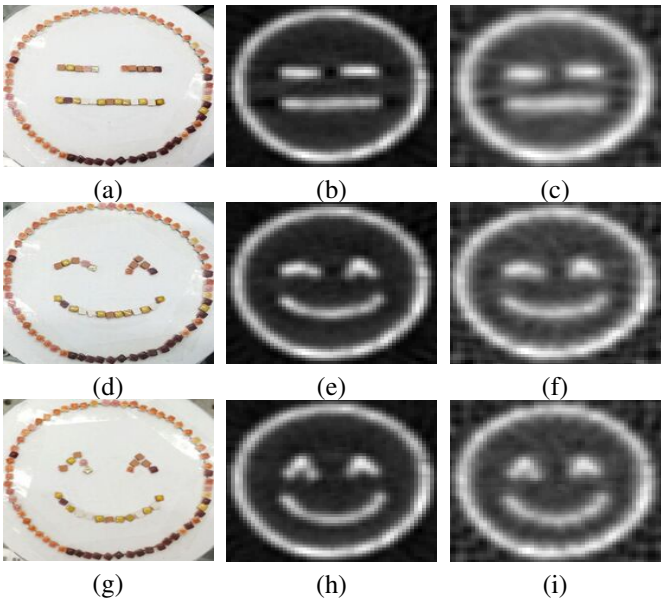
Fig. 4. Average CPU time (seconds) versus time step T is 10^2, 10^3, 10^4, 10^5, 10^6, 10^7 in the nonlinear simulated trajectory. The axes are in log-scale.

TABLE III
COMPARISON OF AVERAGE CPU TIME (SECONDS) WITH DIFFERENT METHODS IN THE NONLINEAR CASE

T	10^2	10^3	10^4	10^5	10^6	10^7
GN-PRS	0.27	102	1303	-	-	-
GN-SBM	1.17	233	5212.7	-	-	-
GN-FOPD	0.23	85	4640	-	-	-
GN-ADMM	0.39	93	1252.6	-	-	-
IEKS	0.04	0.01	1.12	16.92	178	1561
IEKS-PRS	0.06	0.22	1.29	26.21	263	2402
IEKS-SBM	0.37	1.31	7.26	146.81	1473	11155
IEKS-FOPD	0.07	0.24	1.16	37.45	356	3260
IEKS-ADMM	0.05	0.17	1.99	18.72	187	1716

C. Tomographic Reconstruction

In this section, we consider the application of the methodology to X-ray computed tomography imaging [48], [49]. We have a sequence of square X-ray images of size $s \times s$ with $s = 64, 128$, which we are interested in reconstructing from low-dose observations taken from a limited number of angles. These low-dose observations can be modeled by the measurement matrix $\mathbf{H}_t$ which describes line integrals through the object (i.e., Radon transform). We evaluate the performance of the methods on real tomographic X-ray data of an emoji phantom measured at the University of Helsinki [50]. The dataset consists of 33-point time series of the X-ray sinogram of an emoji made of small squared ceramic stones. In the sequence, the emoji transforms from a face with closed eyes and a straight mouth to a face with smiling eyes and mouth.



(a) (b) (c)
(d) (e) (f)
(g) (h) (i)

Fig. 5. Reconstruction results for the emoji motion dataset. Pictures and the reconstructions of emoji motion at time steps $t = 1, 15, 33$ are given in (a), (d), and (g), respectively. Reconstruction from 60 and 30 projections are shown in the middle column images (b), (e), (h) and the third column images (c), (f), (i), respectively.

Fig. 5 shows the CT reconstruction results obtained by KS-ADMM. We set the parameters to $\lambda = 10$, $\rho = 1$, $k_{\max} = 20$, and $n_x = 4096$. The analysis operator $\mathbf{\Omega}_t$ consists of all the vertical and horizontal gradients (one step differences), which

corresponds to so called TV regularization [21]. The number of measurements that correspond to 60 or 30 projections are $n_y = 13020$ and $n_y = 6510$, respectively. Although there is no ground truth to compare the qualitative results, we can observe the visual results from different numbers of projections. When the number of projections is 60, the method provides good reconstruction results with 20 iterations. We see that the 30-projection results suffer from the block artifacts as a consequence of the reduction in dose.

In this experiment we used a stationary Kalman filter and smoother to implement the optimization. We pre-computed all the gains before the iteration, which significantly speeds up the computations in tomographic reconstruction. We report CPU time (seconds) in Table IV. Table IV shows that KS-ADMM achieves significantly lower CPU time than the batch ADMM method. For example, when $n_x = 16384$, ADMM takes three time longer than our proposed method.

TABLE IV
AVERAGE CPU TIME (SECONDS) FOR THE OUTER ITERATION

T	n_x	n_y	ADMM	KF-ADMM
33	16384	13020	3657.58	771.49
33	16384	6510	1422.38	387.26
33	4096	13020	284.83	97.93
33	4096	6510	77.79	20.10

VI. CONCLUSION

In this paper, we have presented two new classes of methods for solving state estimation problems. The estimation problem has been formulated as an (analysis) L_1-regularized optimization problem and the resulting problem has been solved by using the combinations of (iterated extended) Kalman smoother and variable splitting methods such as ADMM. The proposed approaches replace the batch solution for the state-update by using the smoother, which has a lower time-complexity than the batch solution. Furthermore, we have extended the proposed methods to a more general algorithmic framework, where the state-update is computed with the smoother. We have also established (local) convergence results for the novel KS-ADMM and IEKS-ADMM methods.

In two different linear and nonlinear simulated cases, we have presented experimental results that show the efficiency of the smoother-based variable splitting optimization methods, especially when applied to large-scale or high-dimensional L_1-regularized state estimation problems. We also applied the methodology to a real-life tomographic reconstruction problem arising in X-ray-based computed tomography.

APPENDIX A
PROOF OF LEMMA 1

We first prove for **Case (a)**. By the first-order optimality condition of $\mathbf{x}$ subproblem, we have

$$\nabla \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) = 0, \tag{52}$$

which implies that

$$\begin{aligned} \nabla \theta_1(\mathbf{x}^{(k+1)}) &= \boldsymbol{\Omega}^\top(\boldsymbol{\eta}^{(k)} + \rho(\mathbf{w}^{(k)} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)})) \\ &= \boldsymbol{\Omega}^\top \boldsymbol{\eta}^{(k+1)}. \end{aligned} \tag{53}$$

It follows that

$$\|\boldsymbol{\Omega}^\top \boldsymbol{\eta}^{(k+1)} - \boldsymbol{\Omega}^\top \boldsymbol{\eta}^{(k)}\| \leq L_{\theta_1} \|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|. \tag{54}$$

Then, of we assume that $\boldsymbol{\Omega}$ is full-row rank with $\boldsymbol{\Omega}\boldsymbol{\Omega}^\top \succeq \kappa_a^2 \mathbf{I}$, we have

$$\|\boldsymbol{\Omega}^\top \boldsymbol{\eta}^{(k+1)} - \boldsymbol{\Omega}^\top \boldsymbol{\eta}^{(k)}\|^2 \geq \kappa_a^2 \|\boldsymbol{\eta}^{(k+1)} - \boldsymbol{\eta}^{(k)}\|^2. \tag{55}$$

By combining (54) and (55), we get

$$\|\boldsymbol{\eta}^{(k+1)} - \boldsymbol{\eta}^{(k)}\|^2 \leq \frac{L_{\theta_1}^2}{\kappa_a^2} \|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2. \tag{56}$$

Thus, for the $\boldsymbol{\eta}$-subproblem, we can use the primal variable to bound as follows:

$$\begin{aligned} &\mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k+1)}) - \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k)}) \\ &= \langle \boldsymbol{\eta}^{(k+1)}, \mathbf{w}^{(k+1)} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)} \rangle - \langle \boldsymbol{\eta}^{(k)}, \mathbf{w}^{(k+1)} - \boldsymbol{\Omega}\mathbf{x}^{(k+1)} \rangle \\ &= \frac{1}{\rho} \|\boldsymbol{\eta}^{(k+1)} - \boldsymbol{\eta}^{(k)}\|^2 \leq \frac{L_{\theta_1}^2}{\rho \kappa_a^2} \|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2. \end{aligned} \tag{57}$$

Since the $\mathbf{x}$-subproblem is μ_x-strongly convex we have

$$\begin{aligned} &\mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) \\ &\leq \mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) - \frac{\mu_x}{2} \|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2. \end{aligned} \tag{58}$$

Similarly, since the $\mathbf{w}$-subproblem is convex, we have the following inequality:

$$\mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k)}) \leq \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}). \tag{59}$$

Thus, by combining (58), (59) and (61), we obtain:

$$\begin{aligned} &\mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k+1)}) - \mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) \\ &= \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k+1)}) - \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k)}) \\ &\quad + \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k)}) - \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) \\ &\quad + \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) - \mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) \\ &\leq \frac{L_{\theta_1}^2}{\rho \kappa_a^2} \|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2 - \frac{\mu_x}{2} \|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2 \\ &= \left(\frac{L_{\theta_1}^2}{\rho \kappa_a^2} - \frac{\mu_x}{2} \right) \|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2, \end{aligned} \tag{60}$$

which will be negative provided by $\rho > 2L_{\theta_1}^2 / \kappa_a^2 \mu_x$. Thus, when $\rho > \max\left(\frac{2L_{\theta_1}^2}{\kappa_a^2 \mu_x}, \rho_0 \right)$, the result follows.

Next, we prove **Case (b)**. In this case, we do not assume convexity of the $\mathbf{x}$-subproblem. For the $\boldsymbol{\eta}$-subproblem, we obtain

$$\begin{aligned} &\mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k+1)}) - \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k)}) \\ &= \frac{1}{\rho} \|\boldsymbol{\eta}^{(k+1)} - \boldsymbol{\eta}^{(k)}\|^2. \end{aligned} \tag{61}$$

Let $\mathbf{\Omega}$ be full-column rank with $\mathbf{\Omega}^\top\mathbf{\Omega} \succeq \kappa_b^2\mathbf{I}$, which gives

$$\|\mathbf{\Omega}\boldsymbol{x}^{(k+1)} - \mathbf{\Omega}\boldsymbol{x}^{(k)}\|^2 \geq \kappa_b^2\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2. \tag{62}$$

For the $\mathbf{x}$-subproblem we get

$$\begin{aligned}
&\mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) - \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) \\
&= \theta_1(\mathbf{x}^{(k)}) - \theta_1(\mathbf{x}^{(k+1)}) + \langle \boldsymbol{\eta}^{(k)}, \mathbf{\Omega}\mathbf{x}^{(k+1)} - \mathbf{\Omega}\mathbf{x}^{(k)}\rangle \\
&\quad + \langle \rho(\mathbf{w}^{(k)} - \mathbf{\Omega}\mathbf{x}^{(k+1)}), \mathbf{\Omega}\mathbf{x}^{(k+1)} - \mathbf{\Omega}\mathbf{x}^{(k)}\rangle \\
&\quad + \frac{\rho}{2}\|\mathbf{\Omega}\mathbf{x}^{(k+1)} - \mathbf{\Omega}\mathbf{x}^{(k)}\|^2 \\
&\overset{(53)}{=} \theta_1(\mathbf{x}^{(k)}) - \theta_1(\mathbf{x}^{(k+1)}) + \frac{\rho}{2}\|\mathbf{\Omega}\mathbf{x}^{(k+1)} - \mathbf{\Omega}\mathbf{x}^{(k)}\|^2 \\
&\quad + \langle -\nabla\theta_1(\mathbf{x}^{(k+1)}), \mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}\rangle \\
&\geq -\frac{L_{\theta_1}}{2}\|\mathbf{x}^{(k)} - \mathbf{x}^{(k+1)}\|^2 + \frac{\rho}{2}\|\mathbf{\Omega}\mathbf{x}^{(k+1)} - \mathbf{\Omega}\mathbf{x}^{(k)}\|^2 \\
&\overset{(62)}{\geq} \left(\frac{\rho\kappa_b^2}{2} - \frac{L_{\theta_1}}{2}\right)\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2,
\end{aligned} \tag{63}$$

and by combining (59), (61), and (63), we get

$$\begin{aligned}
&\mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k+1)}) - \mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) \\
&= \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k+1)}) - \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k)}) \\
&\quad + \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k+1)}; \boldsymbol{\eta}^{(k)}) - \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) \\
&\quad + \mathcal{L}(\mathbf{x}^{(k+1)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) - \mathcal{L}(\mathbf{x}^{(k)}, \mathbf{w}^{(k)}; \boldsymbol{\eta}^{(k)}) \\
&\leq \frac{L_{\theta_1} - \rho\kappa_b^2}{2}\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2 + \frac{1}{\rho}\|\boldsymbol{\eta}^{(k+1)} - \boldsymbol{\eta}^{(k)}\|^2,
\end{aligned} \tag{64}$$

which will be nonnegative provided that $\rho > \frac{L_{\theta_1}}{\kappa_b^2}$.

APPENDIX B
PROOF OF LEMMA 2

The error between the local iterate $\mathbf{x}^{(i+1)}$ in the update and the minimizer $\mathbf{x}^\star$ satisfies the following recursion:

$$\begin{aligned}
&\|\mathbf{x}^{(i+1)} - \mathbf{x}^\star\| \\
&= \left\|[\mathbf{J}^\top\mathbf{J}(\mathbf{x}^{(i)})]^{-1}\right\| \\
&\quad \times \left\|\mathbf{J}^\top\mathbf{J}(\mathbf{x}^{(i)})(\mathbf{x}^{(i)} - \mathbf{x}^\star) - (\nabla f(\mathbf{x}^{(i)}) - \nabla f(\mathbf{x}^\star))\right\| \\
&\leq \left\|[\mathbf{J}^\top\mathbf{J}(\mathbf{x}^{(i)})]^{-1}\right\| \\
&\quad \times \int_0^1 \left\|\mathbf{J}^\top\mathbf{J}(\mathbf{x}^{(i)}) - (\mathbf{J}^\top\mathbf{J}(\mathbf{x}^\star + \alpha(\mathbf{x}^{(i)} - \mathbf{x}^\star)) \right. \\
&\qquad \left. + \mathbf{H}(\mathbf{x}^\star + \alpha(\mathbf{x}^{(i)} - \mathbf{x}^\star)))\right\| \left\|\mathbf{x}^{(i)} - \mathbf{x}^\star\right\| \mathrm{d}\alpha \\
&\leq \left\|[\mathbf{J}^\top\mathbf{J}(\mathbf{x}^{(i)})]^{-1}\right\| \\
&\quad \times \int_0^1 \left\|\nabla^2 f(\mathbf{x}^{(i)}) - \nabla^2 f(\mathbf{x}^\star + \alpha(\mathbf{x}^{(i)} - \mathbf{x}^\star)) - \mathbf{H}(\mathbf{x}^{(i)})\right\| \\
&\qquad \times \left\|\mathbf{x}^{(i)} - \mathbf{x}^\star\right\| \mathrm{d}\alpha \\
&\leq \frac{L_f}{2}\left\|[\mathbf{J}^\top\mathbf{J}(\mathbf{x}^{(i)})]^{-1}\right\| \left\|\mathbf{x}^{(i)} - \mathbf{x}^\star\right\|^2 \\
&\quad + \left\|[\mathbf{J}^\top\mathbf{J}(\mathbf{x}^{(i)})]^{-1}\mathbf{H}(\mathbf{x}^{(i)})\right\| \left\|\mathbf{x}^{(i)} - \mathbf{x}^\star\right\|.
\end{aligned} \tag{65}$$

Let $\mathbf{H}(\mathbf{x})$ be bounded by ϵ_h, that is, $\|\mathbf{H}(\mathbf{x})\| \leq \epsilon_h$. We conclude that when $\epsilon_h \to 0$, the convergence is quadratic. Now let $\mathbf{J}^\top\mathbf{J}(\mathbf{x}^{(i)}) \succeq \mu^2\mathbf{I}$. Then, linear convergence is obtained when the following condition is satisfied:

$$\begin{aligned}
&\left\|[\mathbf{J}^\top\mathbf{J}(\mathbf{x}^{(i)})]^{-1}\mathbf{H}(\mathbf{x}^{(i)})\right\| \\
&\quad \leq \left\|[\mathbf{J}^\top\mathbf{J}(\mathbf{x}^{(i)})]^{-1}\right\| \|\mathbf{H}(\mathbf{x}^{(i)})\| \leq \frac{\epsilon_h}{\mu^2} < 1.
\end{aligned} \tag{66}$$

REFERENCES

[1] Y. B. Shalom, X. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. Wiley, 2001.

[2] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge, U.K.: Cambridge Univ. Press, Aug. 2013.

[3] E. Mallada, C. Zhao, and S. Low, "Optimal load-side control for frequency regulation in smart grids," *IEEE Trans. Automat. Control*, vol. 62, no. 12, pp. 6294–6309, Dec. 2017.

[4] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic system," *AIAA J.*, vol. 3, no. 8, pp. 1445–1450, Aug. 1965.

[5] B. Bell, "The iterated Kalman smoother as a Gauss-Newton method," *SIAM J. Optim.*, vol. 4, no. 3, pp. 626–636, Aug. 1994.

[6] J. A. Tropp and S. J. Wright, "Computational methods for sparse solution of linear inverse problems," *Proc. IEEE*, vol. 98, no. 6, pp. 948–958, Jun. 2010.

[7] A. Carmi, P. Gurfil, and D. Kanevsky, "Methods for sparse signal recovery using kalman filtering pseudo-measuremennt norms and quasi-norms," *IEEE Trans. Signal Process.*, vol. 58, no. 4, pp. 2405–2409, Apr. 2010.

[8] N. Vaswani, "Kalman filtered compressed sensing," *Proc. IEEE Int. Conf. Image Processing (ICIP)*, pp. 893–896, Oct. 2008.

[9] D. Zachariah, S. Chatterjee, and M. Jansson, "Dynamic iterative pursuit," *IEEE Trans Signal Process*, vol. 60, no. 9, pp. 4967–4972, Sep. 2012.

[10] S. Farahmand, G. Giannakis, and D. Angelosante, "Doubly robust smoothing of dynamical processes via outlier sparsity constraints," *IEEE Trans. Signal Process.*, vol. 59, no. 10, pp. 4529–4543, Oct. 2011.

[11] A. Y. Aravkin, B. M. Bell, J. V. Burke, and G. Pillonetto, "An L1 laplace robust kalman smoother," *IEEE Trans. Autom. Control*, vol. 56, no. 12, pp. 2898–2911, Dec. 2011.

[12] A. Aravkin, J. V. Burke, L. Ljung, A. Lozano, and G. Pillonetto, "Generalized Kalman smoothing: Modeling and algorithms," *Automatica*, vol. 86, pp. 63–86, Dec. 2017.

[13] A. Simonetto and E. Dall'Anese, "Prediction-correction algorithms for time-varying constrained optimization," *IEEE Trans. Signal Process.*, vol. 65, no. 20, pp. 942–952, Oct. 2017.

[14] A. Charles, M. Asif, J. Romberg, and C. Rozell, "Sparsity penalties in dynamical system estimation," in *Proc. 45th Annu. Conf. Inform. Sci. Syst. (CISS)*, no. 1–6, Mar. 2011.

[15] A. S. Charles, A. Balavoine, and C. J. Rozell, "Dynamic filtering of time-varying sparse signals via L1 minimization," *IEEE Trans. Signal Process.*, vol. 64, no. 21, pp. 5644–5656, Nov. 2016.

[16] M. Elad, P. Milanfar, and R. Rubinstein, "Analysis versus synthesis in signal priors," *Inv. Probl.*, vol. 23, no. 3, pp. 947–968, Sep. 2007.

[17] R. Gao, S. A. Vorobyov, and H. Zhao, "Image fusion with cosparse analysis operator," *IEEE Signal Process. Lett.*, vol. 24, no. 7, pp. 943–947, Jul. 2017.

[18] M. Yaghoobi, S. Nam, R. Gribonval, and M. E. Davies, "Constrained overcomplete analysis operator learning for cosparse signal modelling," *IEEE Trans. Signal Process.*, vol. 61, no. 9, pp. 2341–2355, May 2013.

[19] R. Rubinstein, T. Peleg, and M. Elad, "Analysis K-SVD: A dictionary learning algorithm for the analysis sparse model," *IEEE Trans. Signal Process.*, vol. 62, no. 3, pp. 661–677, Feb. 2013.

[20] J. S. Turek, I. Yavneh, and M. Elad, "On MAP and MMSE estimators for the co-sparse analysis model," *Digit. Signal Process.*, vol. 28, pp. 57–74, May 2014.

[21] Y. Hu and M. Jacob, "Higher degree total variation (HDTV) regularization for image recovery," *IEEE Trans. Image Process*, vol. 21, no. 5, pp. 2559–2571, May 2012.

[22] R. Chalasani and J. C. Principe, "Dynamic sparse coding with smoothing proximal gradient method," in *in Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2014, pp. 7188–7192.

[23] J. Eckstein and D. Bertsekas, "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Math. Programming*, vol. 55, pp. 293–318, 1992.

[24] E. Ryu and S. P. Boyd, "A primer on monotone operator methods," *Appl. Comput. Math.*, vol. 15, no. 1, pp. 3–43, 2016.

[25] D. W. Peaceman and H. H. Rachford, "The numerical solution of parabolic and elliptic differential equations," *J. Soc. Indust. Appl. Math.*, vol. 3, no. 1, pp. 28–41, 1955.

[26] B. S. He, H. Liu, Z. R. Wang, and X. M. Yuan, "A strictly contractive PeacemanRachford splitting method for convex programming," *SIAM J. Optim.*, vol. 24, no. 3, pp. 1011–1040, 2014.

[27] T. Goldstein and S. Osher, "The split Bregman method for L1-regularized problems," *SIAM J. Imaging Sci.*, vol. 2, no. 2, pp. 323–343, Apr. 2009.

[28] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning.*, vol. 3, no. 1, pp. 1–122, 2011.

[29] R. Glowinski, "On alternating direction methods of multipliers: A historical perspective," in *Modeling, Simulation and Optimization for Science and Technology*. New York: Springer, 2014, pp. 59–82.

[30] J. Ziniel and P. Schniter, "Dynamic compressive sensing of time-varying signals via approximate message passing," *IEEE Trans. Signal Process.*, vol. 61, no. 21, pp. 5270–5284, Jun. 2013.

[31] L. Shen, M. Papadakis, I. A. Kakadiaris, I. Konstantinidis, D. Kouri, and D. Hoffman, "Image denoising using a tight frame," *IEEE Trans. Image Process*, vol. 15, no. 5, pp. 1254–1263, May 2006.

[32] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *J. Math. Imaging. Vis.*, vol. 40, no. 1, pp. 120–145, May 2011.

[33] R. Courant, "Variational methods for the solution of problems with equilibrium and vibration," *Bull. Amer. Math. Soc.*, vol. 49, pp. 1–23, 1943.

[34] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM J. Imag. Sci.*, vol. 1, no. 3, pp. 248–272, 2008.

[35] S. J. Wright and J. Nocedal, *Numerical Optimization*. Springer Verlag,, 2006.

[36] H. Ouyang, N. He, L. Q. Tran, and A. Gray, "Stochastic alternating direction method of multipliers," *Proc. Int. Conf. Mach. Learn.*, vol. 28, pp. 80–88, Jun. 2013.

[37] E. Ryu and S. P. Boyd, "A primer on monotone operator methods," *Appl. Comput. Math.*, vol. 15, no. 1, pp. 3–43, 2016.

[38] E. Esser, "Applications of Lagrangian-based alternating direction methods and connections to Split-Bregman computat," Appl. Math., Univ. California, Los Angeles, techreport 09–31, 2009.

[39] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, pp. 123–231, 2013.

[40] B. Anderson and J. B. Moore, "Detectability and stabilizability of time-varying discrete-time linear systems," *SIAM Journal on Control and Optimization*, vol. 19, no. 1, pp. 20–32, 1981.

[41] B. S. He, H. Yang, and S. L. Wang, "Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities," *J. Optimiz. Theory App.*, vol. 106, no. 2, pp. 337–356, Aug. 2000.

[42] J. Nocedal and S. J., *Numerical Optimization*. Springer-Verlag, 1999.

[43] M. Hong, M. Razaviyayn, and Z.-Q. Luo, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," *SIAM J. Optim.*, vol. 26, no. 1, pp. 337–364, Jan. 2016.

[44] W. Deng and W. Yin, "On the global and linear convergence of the generalized alternating direction method of multipliers," *J Sci Comput.*, vol. 66, no. 3, pp. 889–916, Mar. 2016.

[45] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *J Sci Comput.*, vol. 78, no. 1, pp. 29–63, Jan. 2019.

[46] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge Univ. Press, 2004.

[47] B. M. Bell and F. W. Cathey, "The iterated Kalman filter update as a Gauss-Newton method," *IEEE Trans. Automat. Control*, vol. 38, no. 2, pp. 294–297, Feb. 1993.

[48] L. Pfister and Y. Bresler, "Tomographic reconstruction with adaptive sparsifying transforms," *Proc. IEEE Int. Conf. Acoust., Speech Signal Process*, pp. 6914–6918, 2014.

[49] T. A. Bubba, M. März, Z. Purisha, M. Lassas, and S. Siltanen, "Shearlet-based regularization in sparse dynamic tomography," *Proc. SPIE*, vol. 10394, p. 103940Y, Aug. 2017.

[50] A. Meaney, Z. Purisha, and S. Siltanen, "Tomographic X-ray data of 3D emoji," *arXiv preprint arXiv:1802.09397*, 2018.