IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 29 July 2024, accepted 14 August 2024, date of publication 22 August 2024, date of current version 2 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3447652

RESEARCH ARTICLE

Continuous-Time Sparse Signal Recovery

TADASHI WADAYAMA, (Member, IEEE), AND AYANO NAKAI-KASAI, (Member, IEEE)
Nagoya Institute of Technology, Nagoya, Aichi 466-8555, Japan

Corresponding author: Tadashi Wadayama (wadayama@nitech.ac.jp)

The work of Tadashi Wadayama was supported by the Japan Society for the Promotion of Science (JSPS) KAKENHI Grant-in-Aid for Scientific Research (A) under Grant JP22H00514. The work of Ayano Nakai-Kasai was supported by the JSPS KAKENHI Grant-in-Aid for Young Scientists under Grant JP23K13334.

ABSTRACT This study investigates a continuous-time method for sparse signal recovery, which is suitable for analog optical circuit implementation. The proposed method is defined by a nonlinear ordinary differential equation (ODE) derived from the gradient flow dynamics of the Lasso objective function. Numerical experiments show that the proposed method certainly finds original sparse vectors within reasonable accuracy. To gain insight into the local convergence properties of the proposed method, a linear approximation around the equilibrium point is applied, yielding a closed-form error evolution ODE. This analysis shows the behavior of convergence to the equilibrium point. In addition, a variational optimization problem is proposed to optimize a time-dependent regularization parameter in order to improve both convergence speed and solution quality. The deep unfolded variational optimization method is introduced as a means of solving this optimization problem, and its effectiveness is validated through numerical experiments.

INDEX TERMS Optical computing, analog computing, optical circuit, sparse signal recovery, Lasso, compressed sensing, gradient flow, deep unfolding, variational optimization.

I. INTRODUCTION

A. BACKGROUND

Optical computing has experienced remarkable advancements in recent years [2]. The Mach–Zehnder Interferometer (MZI) has become foundational for the development of integrated programmable optical circuits, as highlighted by Capmany and Peŕez [3]. The exploration of light-based computing holds significant promise for transcending the constraints inherent in electronic systems. Analog optical computing offers the potential for computation speeds that are orders of magnitude faster than what is currently achievable with traditional electronic systems. Additionally, it is characterized by its notably low power consumption.

Silicon photonics stands as a key enabler in optical computing, harnessing established silicon microfabrication techniques to integrate optical functions onto silicon chips [4], [5]. This technology allows for the creation of compact, energy-efficient optical components. Silicon photonics significantly advances data processing speeds and miniaturization, integrating seamlessly with existing semiconductor technologies and propelling optical computing forward.

The associate editor coordinating the review of this manuscript and approving it for publication was Sukhdev Roy.

Recently, the trend in Moore's law, which asserts that the number of transistors in an integrated circuit will double every 24 months, has not been reflected in reality. A major limiting factor is the power consumption of the integrated digital electric circuit, which hampers the ability to handle large-scale signal processing problems that require high throughput. For instance, base stations (BSs) in wireless network systems [9], [10] need to perform a significant number of signal processing tasks, including multiple-input and multiple-output (MIMO) signal detection and beam forming. In next-generation systems (e.g., beyond 5G or 6G), the limitations of the central processing unit are likely to become a major obstacle to achieving desired specifications. A potential solution to this challenge would be to use an optical computing-based signal processing unit as a specialized processing unit with lower power consumption [6], i.e., *hardware accelerator*.

Recent examples of optical computing include a neural network implemented in the optical domain [13] and an optical Ising machine [14] for combinatorial optimization. Programmable integrated optical circuits [3] are becoming an active research topic in optical computing. In the field of optical numerical computing, optical adders and multipliers have been used to solve ordinary differential equations (ODEs) [12]. Optical computing has the potential

to perform large-scale signal processing tasks with high energy efficiency and extremely fast computation.

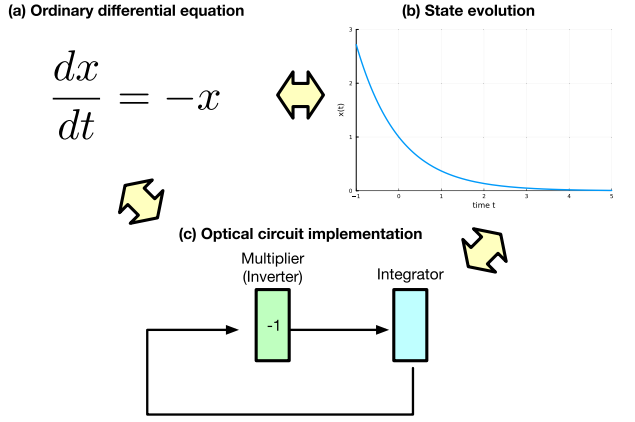


FIGURE 1. Relationship among ODE, state evolution, and optical circuit.

B. CONTINUOUS-TIME METHOD

There have been several studies on continuous-time methods suitable for optical circuit implementation. A pioneering work by Rozell et al. [15] introduced continuous dynamical systems for solving sparse coding. Wang et al. [16] recently presented a continuous-time resistive memory circuit for solving compressed sensing problem. In [17], RRAM (resistive random-access memory)-based analog computing is applied for implementing MIMO precoding problems. A recent work [18] proposed an ODE-based method for MMSE signal detection in MIMO systems and presented a concise analytical formula for the mean squared error. Another example is the gradient flow decoding that is a continuous-time method for decoding LDPC codes [19]. An optical implementation of these methods would provide a certain impact on the wireless communications context.

Figure 1 explains the concept of continuous-time methods for signal processing with a simple example. The ODE depicted in Fig.1(a) defines the system and its continuous-time behavior. A solution of the ODE shown in Fig.1(b) can be seen as a trajectory of the state evolution of the system. The state evolution can be *emulated* by a simple analog optical circuit consisting of a multiplier and an integrator in this case. If the state evolution of the system can solve a certain meaningful signal processing task, the ODE corresponding to the state evolution can be seen as a blueprint of the optical circuit for that task. Despite the promising research direction, there remains a plethora of opportunities for such signal processing methods.

C. CONTRIBUTIONS

The focus of this paper is on a continuous-time method for *sparse signal recovery* [31], [32]. Our work is motivated by growing demand for high-speed and large-scale sparse signal recovery tasks in a compressed sensing context [33]. For instance, grant-based random access schemes [53] require efficient sparse signal recovery algorithms for the detection of active users and channel estimation. Such sparse signal recovery algorithms play a pivotal role in enhancing the efficiency and reliability of grant-based random access systems, especially in scenarios characterized by high user density. Although the advancements of the sparse signal recovery algorithms is significant, all of them are discrete-time algorithms working on a digital computer or a digital circuit. The extension to continuous-time methods suitable for optical circuit implementation is not so straightforward.

Our main contributions of this work are summarized as follows:

- Proposal of a continuous-time method for sparse signal recovery,
- Numerical analysis for assessing the recovery performance,
- Theoretical analysis on convergence behavior,
- Optimization of the system by using deep unfolding.

Our approach is based on the Lasso formulation [34], which can be viewed as a typical regularized least-squares (LS) objective function. The proposed ODE corresponds to the gradient flow based on the Lasso objective function. To gain insight into the local convergence properties of the system, a linear approximation around the equilibrium point is applied, which yields a closed-form error evolution ODE. The analysis tracks system behavior as the solution converges to the equilibrium point. In addition, a variational optimization problem is proposed to optimize a time-dependent regularization parameter in order to improve both convergence speed and solution quality. A deep unfolding-based method is presented for solving the variational problem. The methodology presented in the paper is directly applicable to any type of continuous-time methods.

D. OUTLINE

The method presented in this paper was first introduced in a conference paper [1]. The present paper significantly expands upon the initial presentation, providing a comprehensive explanation of the derivation of the proposed method, an in-depth analysis of its convergence, and detailed studies on the optimization of the system through deep unfolding.

The remainder of the paper is organized as follows: In Section II, the system model and the Lasso formulation used in the paper are introduced. Section III presents the proposed continuous-time method for sparse signal recovery, including numerical demonstrations. Section IV provides an analysis of the local convergence of the proposed method, supported by numerical examples. Section V introduces a parametric ODE with a continuous-time function to be optimized. It presents a deep unfolding-based optimization method for solving a variational optimization problem, aimed at improving convergence behavior and solution quality. Section VI concludes the discussion.

II. PRELIMINARIES

A. NOTATION

The following mathematical notation is used throughout the paper: The symbols $\mathbb{R}$ and $\mathbb{R}_+$ represent the set of real

numbers and the set of positive real numbers, respectively. The one-dimensional Gaussian distribution with mean μ and variance σ^2 is denoted by $\mathcal{N}(\mu, \sigma^2)$. The multivariate Gaussian distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ is represented by $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Assume that a function $f : \mathbb{R} \to \mathbb{R}$ is given. The same function f can be applied to $\boldsymbol{x} \equiv (x_1, x_2, \ldots, x_n)$ as $f(\boldsymbol{x}) = (f(x_1), f(x_2), \ldots, f(x_n))$. The expectation operator is denoted by $\mathsf{E}[\cdot]$. The notation $\mathrm{diag}(\boldsymbol{x})$ denotes the diagonal matrix whose diagonal elements are given by $\boldsymbol{x} \in \mathbb{R}^n$. The matrix exponential $\exp(\boldsymbol{X})(\boldsymbol{X} \in \mathbb{R}^{n\times n})$ is defined by

$$\exp(\boldsymbol{X}) \equiv \sum_{k=0}^{\infty} \frac{1}{k!} \boldsymbol{X}^k. \tag{1}$$

The spectral norm of $\boldsymbol{X} \in \mathbb{R}^{n\times n}$ is denoted by $\|\boldsymbol{X}\|_2$. The notation $[n]$ denotes the set of consecutive integers from 1 to n.

B. SYSTEM MODEL

In this paper, we follow a common system model for compressed sensing [31], [32]. It is assumed that the original sparse signal $\boldsymbol{s} \in \mathbb{R}^n$ follows the Bernoulli-Gaussian distribution, where a non-zero element follows a Bernoulli distribution with probability p and a non-zero element follows a Gaussian distribution $\mathcal{N}(0, 1)$. A linear observation vector $\boldsymbol{y} \in \mathbb{R}^m$ is generated by

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{n}, \tag{2}$$

where $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ is the sensing matrix. The vector $\boldsymbol{n} \in \mathbb{R}^m$ is an additive Gaussian noise vector following $\boldsymbol{n} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$. The goal of the sparse signal recovery problem is to estimate the original sparse vector $\boldsymbol{s}$ from the knowledge of $\boldsymbol{y}$ and $\boldsymbol{A}$ as accurately as possible.

C. LASSO FORMULATION FOR SPARSE SIGNAL RECOVERY

The Lasso objective function is defined as

$$f(\boldsymbol{x}) \equiv \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1, \tag{3}$$

where $\lambda(> 0)$ is the regularization parameter. The L1 regularization term is included in order to promote a sparse solution in the regularized LS estimation. The optimization problem is then defined as

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}). \tag{4}$$

This optimization problem is convex and can be solved by any convex optimization algorithm. A common approach to solving the Lasso problem is to use a proximal gradient descent method, such as ISTA [39].

D. RELATED WORKS

A number of *discrete time* sparse signal recovery algorithms [33] have been developed based on the Lasso formulation [34], [35], [36], [37]. The Iterative Shrinkage Thresholding Algorithm (ISTA) [38], [39] is one of the best-known algorithms for solving the Lasso problem. ISTA is an iterative algorithm comprising two processes: a linear estimation process and a shrinkage process based on a soft shrinkage function. ISTA can be seen as a proximal gradient descent algorithm [40] and can be directly derived from the Lasso formulation. Another powerful iterative algorithm is Approximate Message Passing (AMP) [41], [42] which provides much faster convergence than ISTA. Ma and Ping proposed Orthogonal AMP (OAMP) [43], which can handle various classes of sensing matrices including unitary invariant matrices. Rangan et al. proposed Vector AMP [44] for right-rotationally invariant matrices and provided a theoretical justification for its state evolution.

III. CONTINUOUS-TIME METHOD

A. GRADIENT FLOW FOR SPARSE SIGNAL RECOVERY

In the following argument, we introduce a gradient flow dynamics [45]. Gradient flow is a dynamical system of the form

$$\frac{d\boldsymbol{x}(t)}{dt} = -\nabla f(\boldsymbol{x}(t)), \tag{5}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is called an energy function. The value of the energy $f(\boldsymbol{x}(t))$ decreases monotonically as time t increases. The behavior of many physical systems can be described by an energy minimization process given by the gradient flow.

Figure 2 presents an example of the gradient flow dynamics for a simple convex energy function f defined on two dimensional Euclidean space. The state vector following the ODE (5) gradually approaches the minimum point of f. The gradient flow can be considered as a *continuous-time counterpart of the gradient descent method*.

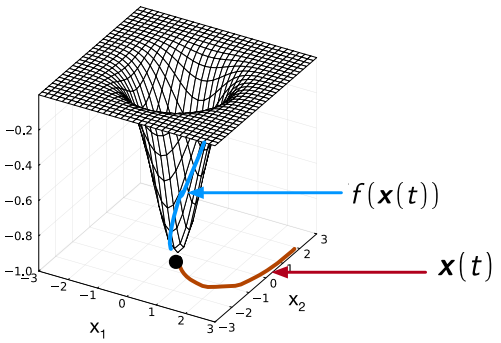


FIGURE 2. An example of gradient flow dynamics.

Although LCA is based on Lasso objective function [15], the Lasso objective function $f(\boldsymbol{x})$ in (3) is not differentiable at $\boldsymbol{x}$ if $\boldsymbol{x}$ includes zero elements. This means that the gradient $\nabla f(\boldsymbol{x})$ is not well-defined at some $\boldsymbol{x}$. Instead of the objective function f defined in (3), we use the differentiable function $g : \mathbb{R}^n \to \mathbb{R}$ defined by

$$g(\boldsymbol{x}) \equiv \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\xi_\alpha(\boldsymbol{x}) \tag{6}$$

as an energy function in the gradient flow. The introduction of the differentiable regularizer makes the convergence analysis

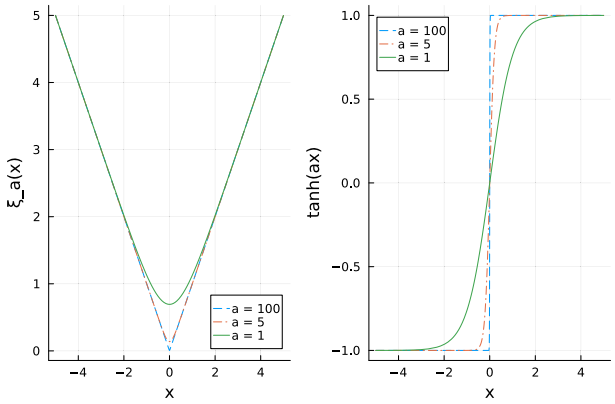


FIGURE 3. Proxy function for $|x|$: Left: $\xi_\alpha(x)$, Right: $\xi'_\alpha(x) = \tanh(\alpha x)$.

in Section IV possible. The function $\xi_\alpha : \mathbb{R} \to \mathbb{R}$ is given by

$$\xi_\alpha(x) \equiv \frac{\log(\exp(\alpha x) + \exp(-\alpha x))}{\alpha}, \tag{7}$$

where $\alpha \in \mathbb{R}_+$ is called the proximity parameter. The function $\xi_\alpha(x)$ can be seen as a proxy function of the absolute value function, i.e., $\xi_\alpha(x) \simeq |x|$ for sufficiently large α. At the limit $\alpha \to \infty$, the function g converges to the function f. The derivative function of ξ_α is given by

$$\xi'_\alpha(x) = \tanh(\alpha x). \tag{8}$$

Figure 3 shows the shapes of $\xi_\alpha(x)$ and $\xi'_\alpha(x) = \tanh(\alpha x)$ for $\alpha = 1, 5, 100$. It can be observed that $\xi_\alpha(x)$ approaches the absolute value function $|x|$ as α increases.

Note that we thus have the gradient of the energy function:

$$\nabla g(\boldsymbol{x}) = \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x}(t) - \boldsymbol{y}) + \lambda \tanh(\alpha \boldsymbol{x}(t)). \tag{9}$$

In this paper, we focus on the gradient flow defined by the following nonlinear ODE:

$$\frac{d\boldsymbol{x}(t)}{dt} = -\left(\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x}(t) - \boldsymbol{y}) + \lambda \tanh(\alpha \boldsymbol{x}(t))\right), \tag{10}$$

$$\boldsymbol{x}(0) = \boldsymbol{x}_0, \tag{11}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^n$ is the state vector at time t, and it can be regarded as the estimate of the original vector $\boldsymbol{s}$ at time t. The variable t represents continuous time in our context. The vector $\boldsymbol{x}_0$ is the initial state providing a boundary condition.

A possible architecture of the analog optical circuit corresponding to the above ODE (10) is presented in Fig.4. The core of the optical circuit shown in Fig.4 is the two matrix-vector product circuit. A programmable MZI-based matrix-vector product circuit [3] can be utilized for implementing this part. The operation of the circuit can be described as follows: The output of the second matrix-vector multiplier is integrated by the optical integrator. These integrated optical signals are then fed back to the input of the first multiplier, creating a closed-loop system that implements the continuous-time dynamics of the ODE. The most challenging aspect of this implementation might be the optical realization of the nonlinear function $\tanh(\cdot)$. However, a hybrid optical-electrical implementation, as demonstrated in [14], can make this task more manageable. In such a hybrid approach, the nonlinear function and the integrator could be implemented in the electrical domain, where these operations are relatively simpler to realize. The optical-to-electrical and electrical-to-optical conversions would need to be carefully designed to maintain the advantages of optical processing.

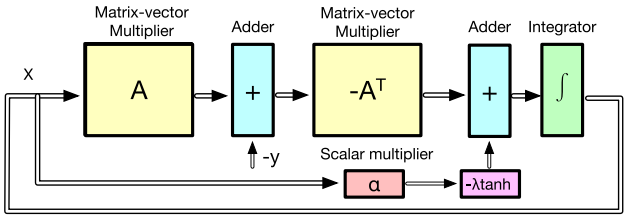


FIGURE 4. Block diagram of analog optical circuit corresponding to ODE (10).

B. NUMERICAL EXPERIMENTS

Figure 5 shows the state trajectories of $\boldsymbol{x}(t)$, which can be considered as the solution of the ODE (10). Each component of $\boldsymbol{x}(t)$ corresponds to a curve in Fig. 5. The Euler method (see Appendix for more details) with $N = 1000$ was used to solve the ODE where N represents the number of bins. The observation vector $\boldsymbol{y}$ was generated randomly according to the system model (2). We can see, in Fig. 5, that the values of many components approach zero and that only a fraction of the curves corresponding to non-zero elements in $\boldsymbol{s}$ deviate from zero. Namely, we can see that the state vector $\boldsymbol{x}(t)$ becomes a sparse vector with time.

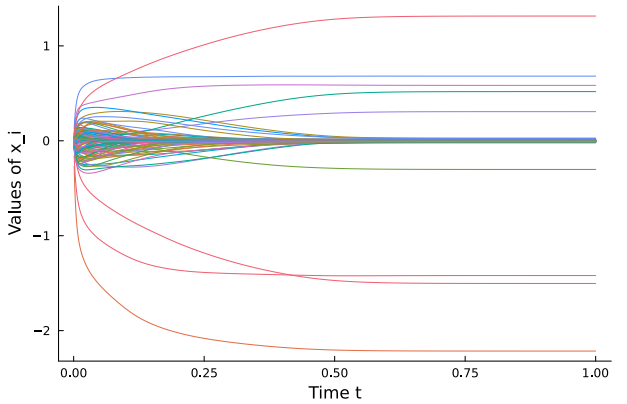


FIGURE 5. A solution of the ODE (10). Each component of $x(t)$ is depicted as a function of time t. The parameters are: $n = 128$, $m = 64$, $p = 0.07$, $\sigma = 0.1$, $\lambda = 3$, $\alpha = 50$, $x_0 = 0$, $T = 1$.

To evaluate the performance and characteristics of our ODE-based sparse signal recovery method (10), numerical experiments were conducted. The parameter settings for the experiments were as follows. The length of the sparse signal $\boldsymbol{s}$ was set to $n = 128$. The length of the observation vector $\boldsymbol{y}$ was $m = 64$. The standard deviation of the Gaussian noises was set to $\sigma = 0.1$. The sensing matrix $\boldsymbol{A}$ was randomly generated. Each element in $\boldsymbol{A}$ follows $\mathcal{N}(0, 1)$. In the following part of this paper, the same $\boldsymbol{A}$ is used for the experiments.

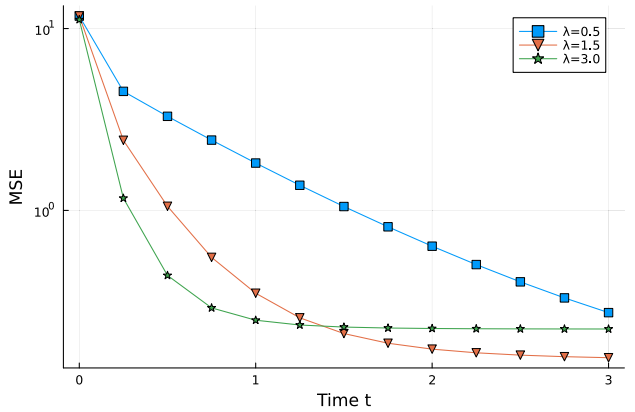


FIGURE 6. Estimates of mean squared error MSE(t) as a function of time t ($n = 128, m = 64, p = 0.1, \sigma = 0.1, \alpha = 50$).

Figure 6 plots the estimates of the mean squared error (MSE) $\mathsf{MSE}(t)$ as a function of time t. The MSE at time t is defined by $\mathsf{E}[\|\boldsymbol{x}(t) - \boldsymbol{s}\|_2^2]$, which can be estimated by

$$\mathsf{MSE}(t) \equiv \mathsf{E}[\|\boldsymbol{x}(t) - \boldsymbol{s}\|_2^2] \simeq \frac{1}{M} \sum_{i \in [M]} \|\boldsymbol{x}_i^{(k)} - \boldsymbol{s}_i\|_2^2, \quad (12)$$

where $\boldsymbol{x}_i^{(k)}$ is the output of the Euler method for the ith trial at index k corresponding to time t. The MSE estimates in Fig. 6 are based on 100 trials of sparse signal recovery processes, i.e., $M = 100$. The Euler method with $N = 5000$ was used for $T = 3$ to estimate $\mathsf{MSE}(t)(0 \le t \le T)$. The parameter settings are nearly the same as in the previous experiment.

We examined three cases: $\lambda = 0.5, 1.5, 3$. From Fig. 6, it is apparent that the MSE curve for $\lambda = 0.5$ has the largest MSE in the range $0 \le t \le 3$; however, the MSE value continues to decrease after $t = 3$. On the other hand, the MSE curve for $\lambda = 3$ shows the fastest convergence, but the value of $\mathsf{MSE}(t)$ saturates to a relatively high constant value in the range $t > 1.0$. The MSE curve for $\lambda = 1.5$ provides a slightly slower convergence compared with the case of $\lambda = 3$, but it shows a lower floor for the MSE values.

These experimental results imply that the regularization constant λ has a strong influence on convergence behavior, similar to discrete time sparse signal recovery algorithms such as ISTA.

IV. LOCAL CONVERGENCE ANALYSIS

A. LINEAR APPROXIMATION

In the previous section, we saw that the convergence behavior of the ODE (10) is highly dependent on the choice of the regularization parameter λ. In this section, the mechanism by which the regularization parameter affects the convergence behavior is examined. The core of the analysis presented here is the use of a linear approximation around the equilibrium point [45] of the ODE to study the local convergence behavior.

In the following analysis, we consider the objective function (6). The gradient of g is thus given by (9). We can now examine the local convergence behavior of the continuous-time dynamical system defined by the ODE (10). The initial vector $\boldsymbol{x}(0) = \boldsymbol{x}_0$ is assumed to be given as a boundary condition.

The equilibrium point $\boldsymbol{x}^*$ of the above ODE is the point satisfying $\nabla g(\boldsymbol{x}) = \boldsymbol{0}$. Thus, the equilibrium point $\boldsymbol{x}^*$ satisfies the equality

$$\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x}^* - \boldsymbol{y}) + \lambda \tanh(\alpha \boldsymbol{x}^*) = \boldsymbol{0}, \quad (13)$$

which is called *equilibrium equality*.

To establish the dynamics of the residual error, we change the coordinate of the ODE according to

$$\boldsymbol{x}(t) = \boldsymbol{x}^* + \boldsymbol{e}(t), \quad (14)$$

where $\boldsymbol{e}(t) \equiv \boldsymbol{x}(t) - \boldsymbol{x}^*$ represents the residual error vector. In the following analysis, $\boldsymbol{e}(t)$ is abbreviated to $\boldsymbol{e}$ for simplicity. Due to the change in the coordinate, the right-hand side of ODE (10) can be transformed to

$$-\left(\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}) + \lambda \tanh(\alpha \boldsymbol{x})\right) \quad (15)$$

$$= -\left(\boldsymbol{A}^T(\boldsymbol{A}(\boldsymbol{x}^* + \boldsymbol{e}) - \boldsymbol{y}) + \lambda \tanh(\alpha(\boldsymbol{x}^* + \boldsymbol{e}))\right) \quad (16)$$

$$= -\left(\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x}^* - \boldsymbol{y}) + \boldsymbol{A}^T\boldsymbol{A}\boldsymbol{e} + \lambda \tanh(\alpha(\boldsymbol{x}^* + \boldsymbol{e}))\right) \quad (17)$$

$$= -\left(\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{e} + \lambda \tanh(\alpha(\boldsymbol{x}^* + \boldsymbol{e})) - \lambda \tanh(\alpha \boldsymbol{x}^*)\right). \quad (18)$$

The last equality is due to the equilibrium equality.

By expanding $\tanh(\alpha \boldsymbol{x})$ using the Taylor expansion and ignoring the higher-order terms, we can obtain the linear approximation around the equilibrium point:

$$\tanh(\alpha \boldsymbol{x}) \simeq \tanh(\alpha \boldsymbol{x}^*) + \boldsymbol{J}(\boldsymbol{x}^*)(\boldsymbol{x} - \boldsymbol{x}^*), \quad (19)$$

where $\boldsymbol{J}(\boldsymbol{x}^*)$ is the Jacobian matrix of $\tanh(\alpha \boldsymbol{x})$ at $\boldsymbol{x}^*$. The Jacobian matrix is given by

$$J(\boldsymbol{x}^*) = \mathrm{diag}\left(\frac{\alpha}{\cosh^2(\alpha x_1^*)}, \frac{\alpha}{\cosh^2(\alpha x_2^*)}, \ldots, \frac{\alpha}{\cosh^2(\alpha x_n^*)}\right), \quad (20)$$

where $\boldsymbol{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)^T$. Note that $\cosh^2(x) > 0$ for any $x \in \mathbb{R}$. This implies that $J(\boldsymbol{x}^*)$ is positive definite for any $\boldsymbol{x}^* \in \mathbb{R}^n$.

By using the linear approximation, we immediately have the following approximation:

$$\lambda \tanh(\alpha(\boldsymbol{x}^* + \boldsymbol{e})) - \lambda \tanh(\alpha \boldsymbol{x}^*) \simeq \lambda J(\boldsymbol{x}^*)\boldsymbol{e}. \quad (21)$$

In the following argument, we will treat this approximated equality as an equality for the sake of simplicity. Substituting this equation into (18), we obtain the linear approximation of the right-hand side of (10):

$$-\left(\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}) + \lambda \tanh(\alpha \boldsymbol{x})\right) = -\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)\right)\boldsymbol{e}. \quad (22)$$

Finally, we have the ODE representing the evolution of the residual error vector:

$$\frac{d\boldsymbol{e}(t)}{dt} = -\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)\right)\boldsymbol{e}(t), \quad (23)$$

$$\boldsymbol{e}(0) = \boldsymbol{e}_0, \tag{24}$$

where $\boldsymbol{e}_0$ represents the initial error vector. This is a linear ODE, which can be solved in a concise form [20]:

$$\boldsymbol{e}(t) = \exp\left(-\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)\right) t\right) \boldsymbol{e}(0). \tag{25}$$

We can expect the above solution to reflect the actual error behavior well if the initial error $\boldsymbol{e}(0)$ is sufficiently close to the zero vector, i.e., in the case where the linear approximation is appropriate.

From the error evolution equation (25), it is clear that the eigenvalues of the matrix $\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)\right) t$ dominate the behavior of the error evolution. Due to the positive definiteness of the Jacobian matrix $\boldsymbol{J}(\boldsymbol{x}^*)$ and the semi-positive definiteness of the Gram matrix $\boldsymbol{A}^T\boldsymbol{A}$, all the eigenvalues of the matrix $\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)\right) t$ are positive real numbers. This means that the time evolution of error vector (25) is *locally asymptotically stable*; that is, $\lim_{t\to\infty} \boldsymbol{e}(t) = \boldsymbol{0}$ for any initial error $\boldsymbol{e}(0)$ sufficiently close to the zero vector.

B. UPPER BOUND ON CONVERGENCE SPEED

From the time evolution or error vector (25), we immediately have the following theorem, under the assumption that the linear approximation is appropriate.

Theorem 1: If the initial vector $\boldsymbol{x}(0)$ is sufficiently close to $\boldsymbol{x}^*$, then the following inequality holds:

$$\frac{\|\boldsymbol{e}(t)\|_2}{\|\boldsymbol{e}(0)\|_2} \leq \exp(-\omega_1 t), \tag{26}$$

where $\{\omega_i\}_{i=1}^n$ is the set of eigenvalues of $\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)$, with the order $0 < \omega_1 \leq \omega_2 \leq \cdots \leq \omega_n$.

(Proof) Taking the norm of both sides of (25), we have an inequality:

$$\|\boldsymbol{e}(t)\|_2 = \| \exp\left(-\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)\right) t\right) \boldsymbol{e}(0)\|_2 \tag{27}$$

$$\leq \| \exp\left(-\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)\right) t\right) \|_2 \|\boldsymbol{e}(0)\|_2, \tag{28}$$

where the matrix norm is the spectral norm. By dividing both sides by $\|\boldsymbol{e}(0)\|_2$, we obtain

$$\frac{\|\boldsymbol{e}(t)\|_2}{\|\boldsymbol{e}(0)\|_2} \leq \| \exp\left(-\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)\right) t\right) \|_2. \tag{29}$$

Note that the above matrix exponential can be rewritten as

$$\exp\left(-\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)\right) t\right) = \boldsymbol{U}\mathrm{diag}(e^{-\omega_1 t}, \ldots, e^{-\omega_n t})\boldsymbol{U}^T, \tag{30}$$

where $\boldsymbol{U}$ is an orthogonal matrix. Since the spectral norm of a semi-positive definite symmetric matrix $\boldsymbol{X}$ coincides with the largest eigenvalue of $\boldsymbol{X}$, we have

$$\| \exp\left(-\left(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)\right) t\right) \|_2 = \exp(-\omega_1 t). \tag{31}$$

Substituting this equality into (29), we obtain the claim of the theorem. □

The above theorem indicates that the term $\exp(-\omega_1 t)$ dominates the convergence speed around an equilibrium point. The average behavior is described by the following corollary, which can be derived directly from Theorem 1.

Corollary 1: If the initial vector $\boldsymbol{x}(0)$ is sufficiently close to $\boldsymbol{x}^*$, then the following inequality holds:

$$\mathsf{E}\left[\frac{\|\boldsymbol{e}(t)\|_2}{\|\boldsymbol{e}(0)\|_2}\right] \leq \mathsf{E}[\exp(-\omega_1 t)]. \tag{32}$$

Let $l_{min}(\boldsymbol{X})$ represent the minimum eigenvalue of a symmetric matrix $\boldsymbol{X}$. When $m < n$, i.e., typical setting for a sparse signal recovery problem, the minimum eigenvalue of $\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)$ satisfies

$$l_{\min}(\boldsymbol{A}^T\boldsymbol{A} + \lambda \boldsymbol{J}(\boldsymbol{x}^*)) \geq l_{\min}(\boldsymbol{A}^T\boldsymbol{A}) + l_{\min}(\lambda \boldsymbol{J}(\boldsymbol{x}^*)) \tag{33}$$

$$= \lambda \min_{i\in[n]} \frac{\alpha}{\cosh^2(\alpha x_i^*)} \tag{34}$$

Note that the condition $m < n$ implies $l_{\min}(\boldsymbol{A}^T\boldsymbol{A}) = 0$. Therefore, increasing the value of λ leads to larger ω_1, which increases the convergence speed.

C. MSE FLOOR

The value on the right-hand side of equation (32) can be reduced by choosing a larger λ. This implies that a larger λ will achieve faster convergence to the equilibrium point. This qualitative argument explains the numerical results shown in Fig. 6; in particular, a larger λ results in a faster decrease of $\mathsf{MSE}(t)$.

On the other hand, from the equilibrium equation (13), we have

$$\|\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x}^* - \boldsymbol{y})\|_2 = \lambda \|\tanh(\alpha \boldsymbol{x}^*)\|_2. \tag{35}$$

Consider a scenario where the noise is negligible. Recall that the original sparse vector $\boldsymbol{s}$ should satisfy

$$\|\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{s} - \boldsymbol{y})\|_2 \simeq 0 \tag{36}$$

under the assumption of negligible noise. In such a case, an increase in λ can lead to an increase in the discrepancy $\|\boldsymbol{x}^* - \boldsymbol{s}\|_2$. From this observation, it can be concluded that a larger λ leads to higher MSE floors, which is indeed observed in Fig. 6.

D. NUMERICAL EXPERIMENTS

The analysis presented in the previous sections is based on a linear approximation around the equilibrium point. Therefore, the accuracy of this approximation should be verified by numerical experiments.

The error norm ratio $\rho(t)$ is defined by

$$\rho(t) \equiv \frac{\|\boldsymbol{e}(t)\|_2}{\|\boldsymbol{e}(0)\|_2} = \frac{\|\boldsymbol{x}(t) - \boldsymbol{x}^*\|_2}{\|\boldsymbol{x}(0) - \boldsymbol{x}^*\|_2}. \tag{37}$$

In the following experiment, we will evaluate the norm ratio $\rho(t)$ for the two cases $\lambda = 1.5$ and $\lambda = 5$.

The parameter settings are nearly the same as in the previous experiment: $n = 128, m = 64, \sigma = 0.1, \alpha = 50$. The equilibrium point $\boldsymbol{x}^*$ is approximated by the value of $\boldsymbol{x}(T)$ where $T = 4$.

We consider two types of initial points, $\boldsymbol{x}(0) = \mathbf{0}$ and

$$\boldsymbol{x}(0) = \hat{\boldsymbol{x}} \equiv \boldsymbol{x}^* + \boldsymbol{\epsilon}, \tag{38}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, 0.1^2\boldsymbol{I})$. Note that $\boldsymbol{x}(0) = \mathbf{0}$ may not be in the region where the linear approximation is valid but $\hat{\boldsymbol{x}}$ can be in such a region because the norm of $\boldsymbol{\epsilon}$ is so small.

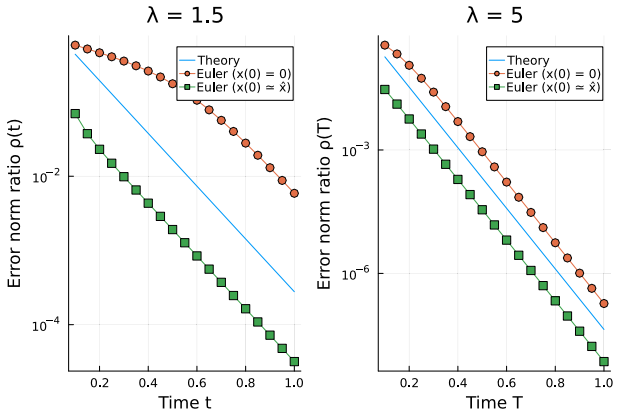


FIGURE 7. Error norm ratio $\rho(t)$ as a function of time t. Left $\lambda = 1.5$, Right $\lambda = 5$ ($n = 128, m = 64, p = 0.1, \sigma = 0.1, \alpha = 50$).

Figure 7 displays the error norm ratio $\rho(t)$ as a function of time t. The solid line labeled "Theory" corresponds to the value of $\exp(-\omega_1 t)$. The left panel shows the case where $\lambda = 1.5$. The error norm ratio curve of $\rho(t)$ with the initial condition $\boldsymbol{x}(0) = \mathbf{0}$ has a shallower slope than that of the curve of $\exp(-\omega_1 t)$ when $t < 0.7$. This may be due to the invalidity of the linear approximation for small t. It can be observed that the curve of $\rho(t)$ with $\boldsymbol{x}_0 = \mathbf{0}$ has almost the same slope as that of $\exp(-\omega_1 t)$ when $t > 0.7$. This can be explained by the fact that the state point $\boldsymbol{x}(t)$ gradually enters the region where the linear approximation is valid. Once the state point enters such a region, the linear approximation theory can be applied. On the other hand, in the case of $\boldsymbol{x}(0) = \hat{\boldsymbol{x}}$, the slope of $\rho(t)$ is almost the same as that of $\exp(-\omega_1 t)$ from the beginning. The result is consistent with the claim of Theorem 1. Since $\hat{\boldsymbol{x}}$ is sufficiently close to $\boldsymbol{x}^*$, the linear approximation is valid from the beginning.

The right panel shows the numerical results for $\lambda = 5$. Nearly the same observations can be made as in the left panel. Compared to the left panel, the value of $\rho(t)$ is much smaller for each t. This observation is consistent with the argument concerning the relationship between λ and the convergence rate in the previous section.

Note that in all the cases, the asymptotic slope of $\rho(t)$ can be accurately estimated by the slope of $\exp(-\omega_1 t)$. This observation supports our claim that the smallest eigenvalue ω_1 determines the convergence rate around an equilibrium point. The numerical results presented in Fig. 7 can be seen as a numerical validation of the argument in the previous sections.

Figure 8 shows $\mathsf{MSE}(\infty)$, $\mathsf{E}[|\lambda \tanh(\alpha \boldsymbol{x}^*)\|_2^2]$, and $\mathsf{E}[\omega_1]$ estimated from 500 trials. For estimating the asymptotic MSE $\mathsf{MSE}(\infty)$, we used $T = 4$ rather than $T = \infty$. The numerical results of $\mathsf{MSE}(\infty)$ indicate that the MSE floor increases as λ increases. We can also observe that the values $\mathsf{E}[|\lambda \tanh(\alpha \boldsymbol{x}^*)\|_2^2]$ (center panel) is an increasing function of λ. This observation is consistent with the argument in Subsection IV-C. The local convergence rate is determined by ω_1. The tendency of $\mathsf{E}[\omega_1]$ is shown in the third panel of Fig. 8. It can be confirmed that a larger λ achieves faster convergence.

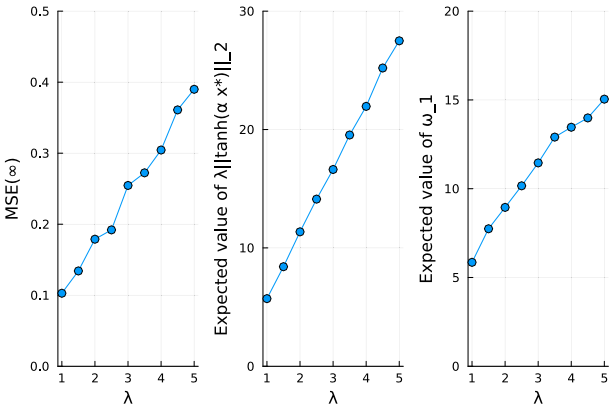


FIGURE 8. Values of $\mathsf{MSE}(\infty)$, $\mathsf{E}[|\lambda \tanh(\alpha x^*)\|_2^2]$, and $\mathsf{E}[\omega_1]$ from left to right ($n = 128, m = 64, p = 0.1, \sigma = 0.1, \alpha = 50$).

Finally, we will experimentally confirm the relationship between α and the MSE convergence behavior. Figure 9 shows the values of $\mathsf{MSE}(\infty)$ and $\mathsf{E}[\omega_1]$ where the value of λ is fixed to 3. From the results of $\mathsf{MSE}(\infty)$, we can say that a larger α results in a smaller MSE floor. The right panel showing $\mathsf{E}[\omega_1]$ indicates that a larger α leads to faster convergence. In summary, α should be large in order to achieve faster convergence and a high-quality solution. We therefore use a relatively large constant $\alpha = 50$ throughout this paper.

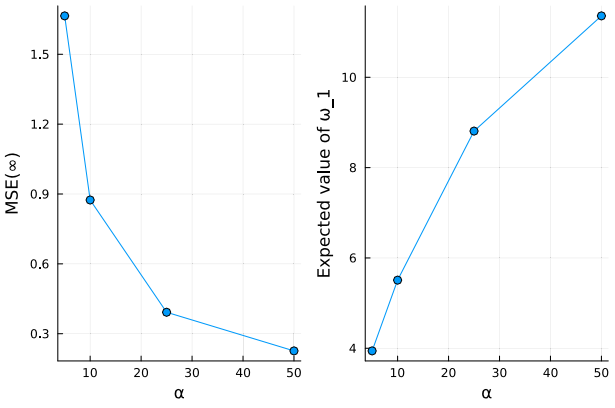


FIGURE 9. Values of $\mathsf{MSE}(\infty)$ and $\mathsf{E}[\omega_1]$ ($n = 128, m = 64, p = 0.1$, $\sigma = 0.1, \lambda = 3$).

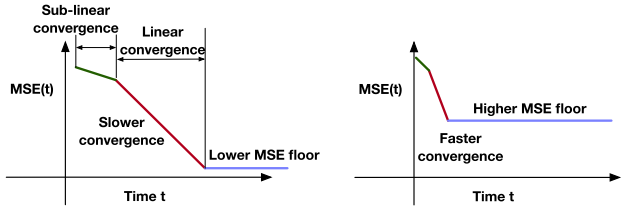
V. DEEP UNFOLDED-VARIATIONAL OPTIMIZATION

A. PARAMETRIC ODE AND ITS OPTIMIZATION

A summary of the theoretical analysis and numerical results presented in the previous section is illustrated in Fig. 10,

which shows the typical behavior of the convergence of $\mathsf{MSE}(t) = \mathsf{E}[\|\boldsymbol{x}(t) - \boldsymbol{s}\|_2^2]$. As discussed earlier, there is a trade-off between the MSE floor value (the asymptotic MSE, $\mathsf{MSE}(\infty)$) and the convergence rate. If we use a large λ, we can expect fast convergence, but we will need to compromise on the quality of the solution, i.e., higher MSE floor. On the other hand, if we use a small λ, we must allow for slow convergence despite of lower MSE floor.

An important consideration is the time period during which sub-linear convergence occurs. If the state point $\boldsymbol{x}(t)$ is close enough to the equilibrium point, the linear approximation can be applied and linear convergence can be expected, as discussed in a previous section. However, if $\boldsymbol{x}(t)$ is not close enough to the equilibrium point, linear convergence cannot be expected. The left panel of Fig. 7 illustrates such a regime, specifically for $0 \leq t \leq 0.6$. As shown in Fig. 10, the linear convergence regime appears after a sub-linear regime.



(a) Small regularizer (b) Large regularizer

FIGURE 10. Typical tendency of the convergence behavior of $\mathsf{MSE}(t)$.

To improve both convergence speed and solution quality, one strategy is to introduce a time-dependent regularization parameter, i.e., varying the value of λ. Starting with a large λ shortens the sub-linear regime and increases the convergence rate. Changing λ judiciously keeps $\boldsymbol{x}(t)$ in the linear convergence regime. By adjusting λ to take on the smallest value at the end of the process, a high-quality solution can be obtained.

This idea is closely related to the *homotopy* or *continuation method*, where the value of the regularization parameter is gradually changed so that the state vector progresses along the solution path. For example, Xiao and Zhang [52] proposed a technique for solving the Lasso problem by using a proximal algorithm, which incorporates a sequence of decreasing values of the regularization parameter. The concept of the continuation method is shown in Fig. 11.

The above idea naturally leads to the following *parametric ODE*:

$$\frac{d\boldsymbol{x}(t)}{dt} = -\left(\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x}(t) - \boldsymbol{y}) + \lambda(t)\tanh(\alpha\boldsymbol{x}(t))\right), \quad (39)$$

where the regularizing constant is replaced with the function $\lambda : \mathbb{R} \to \mathbb{R}$ and $\boldsymbol{x}(0) = \boldsymbol{x}_0$.

In this section, we examine the optimization of $\lambda(t)$ in this parametric ODE. The optimization problem at hand can be formulated as

$$\text{minimize } \mathsf{E}[\|\boldsymbol{x}(T^*) - \boldsymbol{s}\|_2^2] \text{ subject to } \lambda : \mathbb{R} \to \mathbb{R}, \quad (40)$$

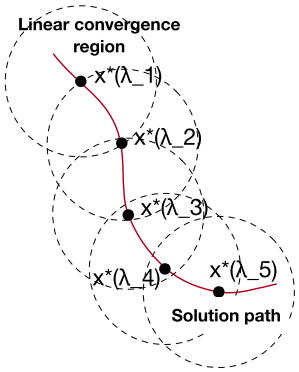


FIGURE 11. Concept of continuation method: $x^*(\lambda_i)(i \in [5])$ represents the equilibrium point when $\lambda = \lambda_i$. Using an appropriately chosen set of parameters $\{\lambda_i\}$, the solution path is fully covered by the linear convergence regions. We can anticipate both faster convergence and a high-quality solution in such a case.

where T^* is the predetermined target time. In essence, we aim to find the optimal continuous-time schedule for λ that produces a high-quality solution at a given time T^*. This optimization problem falls into the category of variational optimization problems. Unfortunately, we cannot expect to derive a compact analytical solution to this problem, and, thus, it must be solved numerically.

B. DEEP UNFOLDED-VARIATIONAL OPTIMIZATION

The development of deep neural networks has also had a significant impact on the design of algorithms for communications and signal processing [21], [22], [23]. Deep unfolding, as described in works such as [26], [29], and [30], can be seen as a highly effective method for improving the convergence of iterative algorithms. Gregor and LeCun introduced the Learned ISTA (LISTA) [26], which uses learnable matrices. LISTA achieves a recovery performance that is much superior to that of the original ISTA. Borgerding et al. also presented variants of AMP and VAMP with learnable capability [27], [28]. Trainable ISTA (TISTA) [30] is a recent learnable sparse signal recovery algorithm with fast convergence. TISTA requires a small number of trainable parameters, which provides a fast and stable training process. It is important to note that algorithms like LISTA operate within a discrete-time framework. Consequently, their methodologies are not directly transferable to the continuous-time sparse signal recovery system examined in this study.

The concept of deep unfolding is simple: Embed trainable parameters in the original iterative algorithm, followed by the unfolding of the signal-flow graph of the original algorithm. The standard supervised training techniques used in deep learning, such as stochastic gradient descent (SGD) [24] and back propagation [25], can then be applied to the unfolded signal-flow graph to optimize the trainable parameters.

In the following, we will introduce the *deep unfolded-variational optimization* (DU-VO) method for the numerical solution of variational optimization problems. The combination of deep unfolding and the Euler method for differential

equation solvers [48] is an active research area in scientific machine learning. However, it should be noted that the technique is not limited to applications within scientific machine learning; rather, it seems to be a powerful tool for optimizing continuous-time systems.

This subsection deals with variational optimization problems in a general context. We will return to the specific optimization problem defined in (40) later.

Assume that we have an ODE

$$\frac{d\boldsymbol{x}(t)}{dt} = h(\boldsymbol{x}(t), u(t), t) \tag{41}$$

and it defines the dynamics of $\boldsymbol{x}(t)$. For a given integrable function F, a *functional* J to be minimized is defined by

$$J[u(t)] \equiv \int_0^T F(\boldsymbol{x}(t), u(t), t)dt. \tag{42}$$

In the following argument, we assume the case where $u(t)$ is one-dimensional, i.e., $u : \mathbb{R} \to \mathbb{R}$, and can control this function. We here consider a variational optimization problem including $\boldsymbol{x}(t)$ defined by an ODE (41):

$$\text{minimize}_{\ u(t)}\ J[u(t)]. \tag{43}$$

Namely, we want to find a one-dimensional function u minimizing the value of the functional J.

This type of variational optimization problem often appears in the field of optimal control [50]. The optimal solution of (43) can be obtained by solving a Hamilton-Jacobi-Bellman (HJB) equation [50]. However, solving the HJB equation is generally difficult because it requires solving a non-linear partial differential equation. In a stochastic context, solving a stochastic HJB equation [50] becomes even more complex and challenging.

The DU-VO method is a numerical approach that aims to solve the variational problem (43). The first step in the implementation of the DU-VO method is to introduce a function approximation using a radial basis function (RBF), which is used to approximate $u(t)$.

Let $\phi : \mathbb{R} \to \mathbb{R}$ be a RBF, which satisfies $\phi(x) = \phi(|x|)$. A continuous function $u(t)$ can be approximated by

$$\tilde{u}(t) \equiv \sum_{i=1}^{S} w_i \phi(t - c_i), \tag{44}$$

where the weight vector $\boldsymbol{w} \equiv (w_1, w_2, \ldots, w_S)^T \in \mathbb{R}^S$ is a trainable parameter that can be updated in an optimization process. The above function approximation is known as the RBF approximation. The shift parameter $\boldsymbol{c} \equiv (c_1, c_2, \ldots, c_S)^T \in \mathbb{R}^S$ is treated as a hyperparameter. Thus, we will use the parametric model of $u(t)$ defined by (44) in the following discussion.

The advantage of using the RBF is that it requires only a small number of trainable parameters to approximate a one-dimensional continuous function, resulting in a stable and efficient learning process.

In order to approximate the solution of (41), we have a number of numerical methods to choose from, such as Runge-Kutta methods; however, for simplicity, the simplest of these, the Euler method, will be used in the following argument. Similarly, for the numerical integration of the functional (42), we will exploit the simplest rectangular method. If necessary, the accuracy of the following DU-VO method can be improved by replacing the ODE solver algorithm and the numerical integration method.

The core of the DU-VO method is the Euler method for numerically solving differential equation (41),

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \eta h(\boldsymbol{x}^{(k)}, \tilde{u}(t^{(k)}), t^{(k)}), \quad k = 0, 1, \ldots, N-1, \tag{45}$$

and the numerical integration step for the functional (42),

$$J^{(k+1)} = J^{(k)} + \eta F(\boldsymbol{x}^{(k)}, \tilde{u}(t^{(k)}), t^{(k)}), \quad k = 0, 1, \ldots, N-1. \tag{46}$$

If the width of the bins is small enough, we can expect that $J^{(N)}$ will become an approximation of $J[\tilde{u}(t)]$. The concept of deep unfolding can be naturally applied to (45) and (46) to optimize the weight vector $\boldsymbol{w}$ that controls the shape of u.

The DU-VO method uses a loss function $J(\boldsymbol{w}) \equiv J^{(N)}$ to approximate the solution of the variational problem (43). The gradient of the loss function with respect to the weight vector $\boldsymbol{w}$, $\nabla J(\boldsymbol{w})$, is evaluated by back propagation. The gradient is then used to update the weight vector $\boldsymbol{w}$. This process can be optimized using common optimizers such as Stochastic Gradient Descent (SGD) or Adam with mini-batch learning. Due to the use of mini-batch learning, we can expect that the weight parameter $\boldsymbol{w}$ can be optimized on *average* across multiple instances, rather than being tailored to a specific instance of a sparse vector. The entire process is summarized in Algorithm 1.

Even if the functional to be optimized contains random variables, the DU-VO method can minimize the expected functional $\mathsf{E}[J(\boldsymbol{w})]$ by using mini-batch training. This is the one of major advantages of the DU-VO method.

C. DU-VO METHOD FOR PROPOSED METHOD

In the following, we consider the parametric ODE (39) and the optimization problem (40) once again. In order to optimize the convergence behavior with respect to $\lambda(t)$, we need a functional of λ to be minimized that is consistent with the optimization problem (40). We here introduce a functional J defined by

$$J[\lambda(t)] \equiv \mathsf{E}\left[\int_0^{T^*} \delta(t - T^*)\|\boldsymbol{x}(t) - \boldsymbol{s}\|^2 dt\right], \tag{47}$$

where $\boldsymbol{s}$ represents the original sparse vector and $\boldsymbol{x}(t)$ in the functional is the solution of the ODE (39). The function δ represents the Dirac's delta function. The functional (47) measures closeness of the solution $\boldsymbol{x}(T^*)$ to the original vector. The time T^* is the predetermined target time. This recursive equation becomes the basis of the following DU-based optimization. The function $\lambda : \mathbb{R} \to \mathbb{R}$ is replaced

Algorithm 1 DU-VO Method

Input: $\boldsymbol{x}_0$ (initial state), $\boldsymbol{w}_0$ (initial weight)
Output: $\boldsymbol{w}$ (optimized weight)

1: Set the initial values: $\boldsymbol{x}^{(0)} \equiv \boldsymbol{x}_0$, $J^{(0)} \equiv 0$, $\boldsymbol{w} = \boldsymbol{w}_0$.
2: **for** $i = 1$ to I **do**
3: **for** $i = 0$ to $N-1$ **do**
4: Compute the Euler step:

$$\boldsymbol{x}^{(k+1)} \equiv \boldsymbol{x}^{(k)} + \eta h(\boldsymbol{x}^{(k)}, \tilde{u}(t^{(k)}), t^{(k)}).$$

5: Compute the numerical integration step:

$$J^{(k+1)} \equiv J^{(k)} + \eta F(\boldsymbol{x}^{(k)}, \tilde{u}(t^{(k)}), t^{(k)}).$$

6: **end for**
7: Compute the gradient of the loss function by using back propagation:

$$\boldsymbol{g} \equiv \nabla J(\boldsymbol{w}).$$

8: The weight parameter $\boldsymbol{w}$ is updated by using $\boldsymbol{g}$ (an SGD optimizer is used).
9: **end for**
10: Output the weight vector $\boldsymbol{w}$.

with a trainable Gaussian RBF approximation [49] defined by

$$\lambda(t) \equiv \sum_{i=1}^{S} w_i \exp(-\beta(t - \Delta i + \theta)^2), \tag{48}$$

where $\{w_i\}_{i=1}^S$ are the trainable weight parameters. The parameters β, Δ, and θ are treated as hyperparameters given before an optimization process.

In an optimization process, we randomly generate multiple mini-batches. A mini-batch is composed of K pairs of vectors, namely $\mathcal{D} \equiv \{(\boldsymbol{s}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{s}_K, \boldsymbol{y}_K)\}$ where $\boldsymbol{s}_i (i \in [K])$ is a sparse random vector following the Bernoulli-Gaussian distribution, and $\boldsymbol{y}_i (i \in [K])$ follows our system model $\boldsymbol{y}_i = \boldsymbol{A}\boldsymbol{s}_i + \boldsymbol{n}_i$. The noise vector $\boldsymbol{n}_i$ follows $\mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$.

The functional J can be approximated as

$$\begin{aligned} J[\lambda(t)] &= \mathsf{E}\left[\int_0^{T^*} \delta(t - T^*)\|\boldsymbol{x}(t) - \boldsymbol{x}^*\|^2 dt\right] \\ &\simeq \frac{1}{K}\sum_{i=1}^{K} \|\boldsymbol{x}_i^{(N)} - \boldsymbol{s}_i\|^2, \end{aligned} \tag{49}$$

where the vector $\boldsymbol{x}_i^{(N)}$ is the state vector corresponding to the data $(\boldsymbol{s}_i, \boldsymbol{y}_i)$. The state vector $\boldsymbol{x}_i^{(k)}$ is evolved according to the Euler recursive equation with the initial condition $\boldsymbol{x}_i(0) = \boldsymbol{0}$.

From this approximation above, it is reasonable to define the loss function:

$$\mathsf{Loss}(\mathcal{D}) \equiv \frac{1}{K}\sum_{i=1}^{K} \|\boldsymbol{x}_i^{(N)} - \boldsymbol{s}_i\|^2, \tag{50}$$

which gives an approximation of MSE as $\mathsf{Loss}(\mathcal{D}) \simeq \mathsf{E}[\mathsf{MSE}(T^*)]$. For each given mini-batch, the trainable parameters $\{w_i\}_{i=1}^S$ are updated based on the gradient of $\mathsf{Loss}(\mathcal{D})$. The optimization process is illustrated in Fig. 12

D. NUMERICAL EXPERIMENTS

In the previous sections, we introduced the DU-based optimization method for solving variational optimization problems. This section presents the results of numerical experiments conducted to evaluate the performance of the proposed method. These experiments were performed using the automatic differentiation mechanism provided by the Flux.jl library [46] in the Julia programming language [47].

The problem setup for the first experiment is similar to the previous ones, with $n = 128, m = 64, p = 0.1, \sigma = 0.1$. The Euler method with $N = 5000$ was used for the optimization process and for estimating the MSE. The following parameters were used for the optimization process. The mini-batch size was set to $K = 10$. The Adam optimizer with a learning rate of 10^{-2} was used. The number of mini-batches, or iterations, used for training was 100. The parameter settings for the RBF approximation of $\lambda(t)$ were $\Delta = 0.25, \beta = 20, S = 20, \theta = 0.5$.

The left panel of Fig. 13 shows the MSE curve of the parametric ODE (39) with the optimized $\lambda(t)$. The MSE curves corresponding to constant regularization parameters $\lambda = 0.5, 1.5, 3.0$ are also shown. As can be seen here, the optimized $\lambda(t)$ attains the smallest MSE value among the four curves at $T^* = 3$. Additionally, the convergence rate, i.e., the slope of the optimized MSE curve, is nearly identical to that of the MSE curve with $\lambda = 0.5$ over the range $2 \leq t \leq 3$. The right panel of Fig. 13 shows the shape of the curve for the optimized $\lambda(t)$. The curve starts at a value of approximately 2.2 and rapidly decreases over the range of $1 \leq t \leq 2$. This result supports the argument in Subsection V-A and suggests that the curve of the optimized $\lambda(t)$ provides an appropriate schedule for decreasing the value of the regularization parameter.

To confirm that the optimized $\lambda(t)$ provides the best MSE among the ODE systems with constant λ, the MSE at $T^* = 3$ was evaluated numerically for several values of λ. The results are plotted in Fig. 14. As shown, the MSE values for the constant λ are above 10^{-1}, while the optimized $\lambda(t)$ achieves an MSE below 10^{-1}. This result thus demonstrates the effectiveness of the proposed method in finding an optimal $\lambda(t)$ that improves the MSE.

We now examine another problem setup that assumes a sparser original signal. Here, the probability is set to $p = 0.05$. The parameters and hyperparameters remain unchanged from the previous experiments, i.e., $n = 128, m = 64, \sigma = 0.1$. The target time is set to $T^* = 2$. The numerical results of this case are shown in Fig. 15. It is clear that the ODE system with the optimized $\lambda(t)$ consistently gives the lowest MSE.

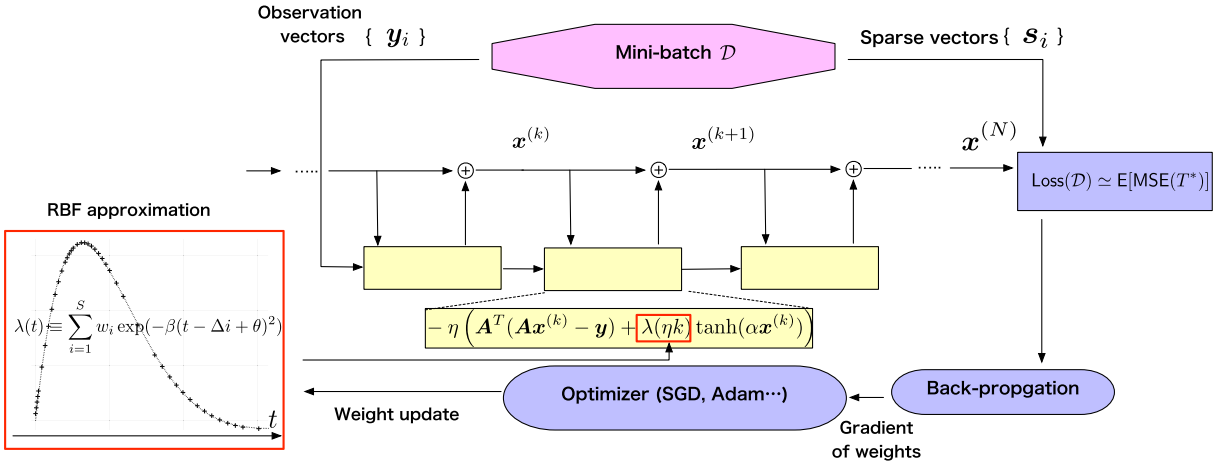


FIGURE 12. Block diagram of Deep Unfolded-Variational Optimization (DU-VO) method for optimizing the parametric ODE (39).

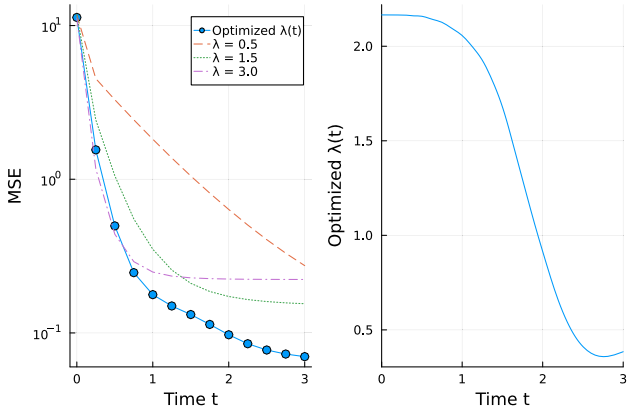


FIGURE 13. Left: MSE as a function of t. Right: optimized $\lambda(t)$. Target time is $T^* = 3$. ($n = 128, m = 64, p = 0.1, \sigma = 0.1, \alpha = 50$).

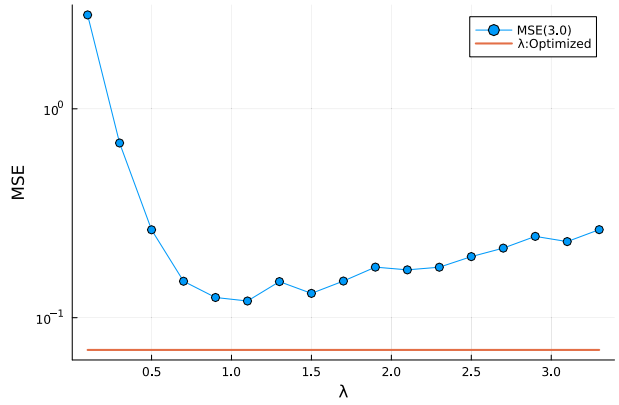


FIGURE 14. MSE at $T^* = 3$ as a function of λ ($n = 128, m = 64, p = 0.1, \sigma = 0.1, \alpha = 50$).

It is also noteworthy that the optimized system consistently achieves close to the optimal value in the range of $0 \leq t \leq 2$. Both the convergence rate and the quality of the solution are improved by the optimization.

In summary, for both cases, the proposed DU-VO method successfully finds an optimized schedule for $\lambda(t)$ that improves the convergence speed and the quality of the solution.

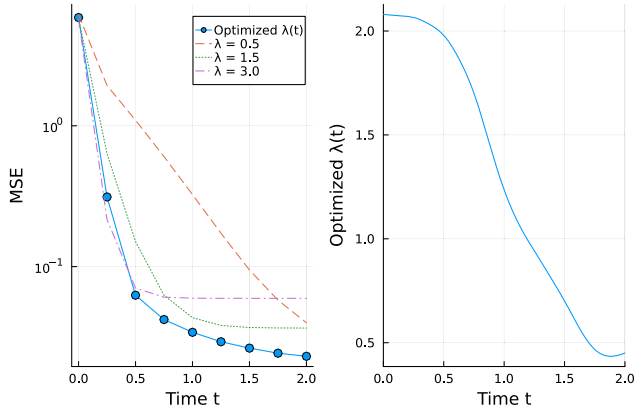


FIGURE 15. Left: MSE as a function of t, Right: Optimized $\lambda(t)$. Target time is $T^* = 2$. ($n = 128, m = 64, p = 0.05, \sigma = 0.1, \alpha = 50$).

VI. CONCLUSION

In this study, we presented a novel approach for solving sparse signal recovery problems using continuous-time method suitable for analog circuits. The method is based on a simple continuous-time gradient flow dynamics of the Lasso objective function. We presented the local convergence analysis of the proposed method using a linear approximation around the equilibrium point. In addition, we introduced a variational optimization problem to optimize the regularization schedule and applied deep unfolding techniques [26], [29] to solve this problem. To the best of our knowledge, this is the first work to propose the use of deep unfolding techniques to solve variational optimization problems.

It should be noted that the methodology proposed in this paper can be applied to various forms of regularized LS problems such as binary quadratic minimization problems.

In this paper, we focused on optical computing paradigms but other analog computing paradigms, such as electronic analog computing based on RRAM [16], [17], might be also

promising for implementing the method presented in the paper. A sequence of research efforts focusing on *ODE-based signal processing*, including studies by [18], [19], and the contributions of this paper, signify a new direction in the field of signal processing.

It should be remarked that the proposed method is evaluated under the ideal condition in this paper, i.e., all the devices are correctly aligned and no noises are involved in the circuit. However, in general, analog computation may be suffered from fabrication errors and system noises. An analog device included in the circuit tends to have parameter errors when it is fabricated. System noises in the circuit may cause non-trivial effect to the system dynamics. Sensitivity analysis and development of mitigation methods for these errors are important issues to be addressed in the future.

APPENDIX

NUMERICAL SOLUTION VIA EULER METHOD

The Euler method is the simplest numerical method for solving simultaneous nonlinear differential equations [20]. Although the convergence order of the Euler method is inferior to that of higher-order methods such as the Runge-Kutta methods [20], the Euler method is simple to use and can provide sufficiently precise solutions if sufficient discretization of the time interval is used. Thus, in this work, we use the Euler method to solve (10).

Consider the ODE in (10). Assume that we need an approximation of the numerical solution of the above ODE in the time interval $0 \leq t \leq T$. This interval is first divided into N bins. The discrete-time ticks $t_k = k\eta (k = 0, 1, \ldots, N)$ define the boundaries of the bins, where the width of a bin, η, is given by $\eta \equiv T/N$. It should be noted that the choice of the bin width η is crucial in order to ensure the stability and the accuracy of the Euler method. A small width leads to a more accurate solution, but requires more computational time. A large width may be computationally efficient but may lead to instability in the solution. Let us define a discretized sample $\boldsymbol{x}^{(k)}$ as $\boldsymbol{x}^{(k)} \equiv \boldsymbol{x}(t_k)$.

By using the Euler method, the solution of (10) can be approximated by the following recursive formula:

$$\begin{aligned} \boldsymbol{x}^{(k+1)} &= \boldsymbol{x}^{(k)} - \eta \nabla g(\boldsymbol{x}^{(k)}) \\ &= \boldsymbol{x}^{(k)} - \eta \left(\boldsymbol{A}^T (\boldsymbol{A} x^{(k)} - \boldsymbol{y}) + \lambda \tanh(\alpha \boldsymbol{x}^{(k)}) \right) \end{aligned} \quad (51)$$

for $k = 0, 1, 2, \ldots, N-1$. The initial value is set to $\boldsymbol{x}^{(0)} = \boldsymbol{x}_0$.

To confirm the accuracy of the Euler method, we conducted an experiment in which we examined the relationship between the MSE and the number of bins N. We first generated a set of sparse signals $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_M$ according to the Bernoulli-Gaussian distribution with $p = 0.1$ and non-zero elements following $\mathcal{N}(0, 1)$. The parameter settings are nearly the same as the settings in the previous experiment; that is, $n = 128, m = 64, \sigma = 0.1, \lambda = 5, \alpha = 50$.

Table 1 shows the estimates of MSE for several different discretizations. Parameter T for the discretizations was set to 4. From Table 1, we can see that the values of the MSE estimates converge as N increases. This result provides evidence that the Euler method works adequately for this ODE. In the following experiments, we use $N = 5000$ to ensure good accuracy and a reasonable computational time.

TABLE 1. Number of bins N and estimated MSE ($T = 4$).

N	MSE(T)
1000	0.503796
2000	0.393782
5000	0.408937
10000	0.408701

ACKNOWLEDGMENT

The authors appreciate the inspiring discussion with Prof. Kazunori Hayashi about optical-based signal processing. They also sincerely thank the anonymous reviewers for their valuable comments, which helped improve the quality of the article. An earlier version of this paper was presented in part at the International Symposium on Information Theory and Its Applications (ISITA) 2022 [1].

REFERENCES

[1] T. Wadayama and A. Nakai-Kasai, "Ordinary differential equation-based sparse signal recovery," in *Proc. Int. Symp. Inf. Theory Its Appl. (ISITA)*, 2022.

[2] N. Stroev and N. G. Berloff, "Analog photonics computing for information processing, inference, and optimization," *Adv. Quantum Technol.*, vol. 6, no. 9, Sep. 2023, Art. no. 2300055.

[3] J. Capmany and D. Peŕez, *Programmable Integrated Photonics*. London, U.K.: Oxford Univ. Press, 2020.

[4] P. Xu and Z. Zhou, "Silicon-based optoelectronics for general-purpose matrix computation: A review," *Adv. Photon.*, vol. 4, no. 4, Jul. 2022, Art. no. 044001.

[5] Q. Cheng, J. Kwon, M. Glick, M. Bahadori, L. P. Carloni, and K. Bergman, "Silicon photonics codesign for deep learning," *Proc. IEEE*, vol. 108, no. 8, pp. 1261–1282, Aug. 2020.

[6] E. Peltonen, M. Bennis, M. Capobianco, M. Debbah, A. Ding, F. Gil-Castiñeira, M. Jurmu, T. Karvonen, M. Kelanti, A. Kliks, T. Leppänen, L. Lovén, T. Mikkonen, A. Rao, S. Samarakoon, K. Seppänen, P. Sroka, S. Tarkoma, and T. Yang, "6G white paper on edge intelligence," Univ. Oulu, Oulu, Finland, Tech. Rep. 8, 2020.

[7] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," in *Proc. 32nd Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2018.

[8] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, and A. Edelman, "Universal differential equations for scientific machine learning," 2020, *arXiv:2001.04385*.

[9] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 186–195, Feb. 2014.

[10] S. Yang and L. Hanzo, "Fifty years of MIMO detection: The road to large-scale MIMOs," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 1941–1988, 4th Quart., 2015.

[11] N. Guo, Y. Huang, T. Mai, S. Patil, C. Cao, M. Seok, S. Sethumadhavan, and Y. Tsividis, "Continuous-time hybrid computation with programmable nonlinearities," in *Proc. 41st Eur. Solid-State Circuits Conf. (ESSCIRC)*, Sep. 2015, pp. 279–282.

[12] L. Lu, J. Wu, T. Wang, and Y. Su, "Compact all-optical differential-equation solver based on silicon microring resonator," *Frontiers Optoelectronics*, vol. 5, no. 1, pp. 99–106, Mar. 2012.

[13] H. Zhang, M. Gu, X. D. Jiang, J. Thompson, H. Cai, S. Paesani, R. Santagati, A. Laing, Y. Zhang, M. H. Yung, Y. Z. Shi, F. K. Muhammad, G. Q. Lo, X. S. Luo, B. Dong, D. L. Kwong, L. C. Kwek, and A. Q. Liu, "An optical neural chip for implementing complex-valued neural network," *Nature Commun.*, vol. 12, no. 1, p. 457, Jan. 2021.

[14] M. Prabhu, C. Roques-Carmes, Y. Shen, N. Harris, L. Jing, J. Carolan, R. Hamerly, T. Baehr-Jones, M. Hochberg, V. Čeperić, J. D. Joannopoulos, D. R. Englund, and M. Soljačić, "Accelerating recurrent Ising machines in photonic integrated circuits," *Optica*, vol. 7, no. 5, p. 551, 2020.
[15] C. J. Rozell, D. H. Johnson, R. G. Baraniuk, and B. A. Olshausen, "Sparse coding via thresholding and local competition in neural circuits," *Neural Comput.*, vol. 20, no. 10, pp. 2526–2563, Oct. 2008.
[16] S. Wang, Y. Luo, P. Zuo, L. Pan, Y. Li, and Z. Sun, "In-memory analog solution of compressed sensing recovery in one step," *Sci. Adv.*, vol. 9, no. 50, Dec. 2023.
[17] P. Zuo, Z. Sun, and R. Huang, "Extremely-fast, energy-efficient massive MIMO precoding with analog RRAM matrix computing," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 7, no. 7, pp. 2335–2339, Jul. 2023.
[18] A. Nakai-Kasai and T. Wadayama, "MMSE signal detection for MIMO systems based on ordinary differential equation," in *Proc. IEEE Global Commun. Conf.*, Dec. 2022, pp. 6176–6181.
[19] T. Wadayama, K. Nakajima, and A. Nakai-Kasai, "Gradient flow decoding for LDPC codes," in *Proc. 12th Int. Symp. Topics Coding (ISTC)*, Brest, France, Sep. 2023.
[20] D. F. Griffiths and D. J. Higham, *Numerical Methods for Ordinary Differential Equations*. London, U.K.: Springer, 2010.
[21] B. Aazhang, B.-P. Paris, and G. C. Orsak, "Neural networks for multiuser detection in code-division multiple-access communications," *IEEE Trans. Commun.*, vol. 40, no. 7, pp. 1212–1222, Jul. 1992.
[22] E. Nachmani, Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning," in *Proc. 54th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2016, pp. 341–346.
[23] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.
[24] Y. A. LeCun, L. Bottou, G. B. Orr, and K. R. Müller, "Efficient backprop," in *Neural Networks: Tricks of the Trade*, G. B. Orr and K. R. Müller, Eds., London, U.k.: Springer, 1998, pp. 9–50.
[25] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.
[26] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, Jun. 2010, pp. 399–406.
[27] M. Borgerding and P. Schniter, "Onsager-corrected deep learning for sparse linear inverse problems," in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Dec. 2016, pp. 227–231.
[28] M. Borgerding, P. Schniter, and S. Rangan, "AMP-inspired deep networks for sparse linear inverse problems," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4293–4308, Aug. 2017.
[29] A. Balatsoukas-Stimming and C. Studer, "Deep unfolding for communications systems: A survey and some new directions," in *Proc. IEEE Int. Workshop Signal Process. Syst. (SiPS)*, Oct. 2019, pp. 266–271.
[30] D. Ito, S. Takabe, and T. Wadayama, "Trainable ISTA for sparse signal recovery," *IEEE Trans. Signal Process.*, vol. 67, no. 12, pp. 3113–3125, Jun. 2019.
[31] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.
[32] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5406–5425, Dec. 2006.
[33] Z. Zhang, Y. Xu, J. Yang, X. Li, and D. Zhang, "A survey of sparse representation: Algorithms and applications," *IEEE Access*, vol. 3, pp. 490–530, 2015.
[34] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Stat. Soc. Ser. B, Stat. Methodol.*, vol. 58, no. 1, pp. 267–288, Jan. 1996.
[35] G. Davis, "Adaptive greedy approximations," *Constructive Approximation*, vol. 13, no. 1, pp. 57–98, Mar. 1997.
[36] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Ann. Statist.*, vol. 32, no. 2, pp. 407–499, Apr. 2004.
[37] T. T. Wu and K. Lange, "Coordinate descent algorithms for lasso penalized regression," *Ann. Appl. Statist.*, vol. 2, no. 1, pp. 224–244, Mar. 2008.
[38] A. Chambolle, R. A. De Vore, N.-Y. Lee, and B. J. Lucier, "Nonlinear wavelet image processing: Variational problems, compression, and noise removal through wavelet shrinkage," *IEEE Trans. Image Process.*, vol. 7, no. 3, pp. 319–335, Mar. 1998.
[39] I. Daubechies, M. Defrise, and C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Commun. Pure Appl. Math.*, vol. 57, no. 11, pp. 1413–1457, Aug. 2004.
[40] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, pp. 123–231, 2014.
[41] Y. Kabashima, "A CDMA multiuser detection algorithm on the basis of belief propagation," *J. Phys. A, Math. Gen.*, vol. 36, pp. 11111–11121, Oct. 2003.
[42] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proc. Nat. Acad. Sci. USA*, vol. 106, no. 45, pp. 18914–18919, Nov. 2009.
[43] J. Ma and L. Ping, "Orthogonal AMP," *IEEE Access*, vol. 5, pp. 2020–2033, 2017.
[44] S. Rangan, P. Schniter, and A. K. Fletcher, "Vector approximate message passing," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 1588–1592.
[45] S. H. Strogatz, *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology Chemistry and Engineering*. Reading, MA, USA: Addison-Wesley, 1994.
[46] M. Innes, "Flux: Elegant machine learning with Julia," *J. Open Source Softw.*, vol. 3, no. 25, p. 602, May 2018.
[47] J. Bezanson, S. Karpinski, V. B. Shah, and A. Edelman, "Julia: A fast dynamic language for technical computing," 2012, *arXiv:1209.5145*.
[48] C. Rackauckas and Q. Nie, "DifferentialEquations.Jl—A performant and feature-rich ecosystem for solving differential equations in Julia," *J. Open Res. Softw.*, vol. 5, no. 1, p. 15, May 2017.
[49] M. D. Buhmann, *Radial Basis Functions: Theory and Implementations*. Cambridge, U.K.: Cambridge Univ. Press, 2003.
[50] J. Yong and X. Y. Zhou, *Stochastic Controls: Hamiltonian Systems and HJB Equations*. London, U.K.: Springer, 1999.
[51] M. Keyanpour and M. Azizsefat, "Numerical solution of optimal control problems by an iterative scheme," *Adv. Model. Optim.*, vol. 13, no. 1, pp. 25–37, 2011.
[52] L. Xiao and T. Zhang, "A proximal-gradient homotopy method for the sparse least-squares problem," *SIAM J. Optim.*, vol. 23, no. 2, pp. 1062–1091, Jan. 2013.
[53] L. Liu, E. G. Larsson, W. Yu, P. Popovski, C. Stefanovic, and E. de Carvalho, "Sparse signal processing for grant-free massive connectivity: A future paradigm for random access protocols in the Internet of Things," *IEEE Signal Process. Mag.*, vol. 35, no. 5, pp. 88–99, Sep. 2018.

TADASHI WADAYAMA (Member, IEEE) was born in Kyoto, Japan, in May 1968. He received the B.E., M.E., and D.E. degrees from Kyoto Institute of Technology, in 1991, 1993, and 1997, respectively. In 1995, he started to work with the Faculty of Computer Science and System Engineering, Okayama Prefectural University, as a Research Associate. From April 1999 to March 2000, he was with the Institute of Experimental Mathematics, Essen University, Germany, as a Visiting Researcher. In 2004, he moved to Nagoya Institute of Technology as an Associate Professor. Since 2010, he has been a Full Professor with Nagoya Institute of Technology. His research interests include coding theory, information theory, and coding and signal processing for digital communication/storage systems. He is a member of IEICE.

AYANO NAKAI-KASAI (Member, IEEE) received the bachelor's degree in engineering, the master's degree in informatics, and the Ph.D. degree in informatics from Kyoto University, Kyoto, Japan, in 2016, 2018, and 2021, respectively. She is currently an Assistant Professor with the Graduate School of Engineering, Nagoya Institute of Technology. Her research interests include signal processing, wireless communications, and machine learning. She is a member of IEICE. She received the Young Researchers' Award from the Institute of Electronics, Information and Communication Engineers, in 2018, and the APSIPA ASC 2019 Best Special Session Paper Nomination Award.