An Iterative Min-Min Optimization Method for Sparse Bayesian Learning

Yasen Wang [1 2] **Junlin Li** [3] **Zuogong Yue** [2 4] **Ye Yuan** [2 4]

Abstract

As a well-known machine learning algorithm, sparse Bayesian learning (SBL) can find sparse representations in linearly probabilistic models by imposing a sparsity-promoting prior on model coefficients. However, classical SBL algorithms lack the essential theoretical guarantees of global convergence. To address this issue, we propose an iterative Min-Min optimization method to solve the marginal likelihood function (MLF) of SBL based on the concave-convex procedure. The method can optimize the hyperparameters related to both the prior and noise level analytically at each iteration by re-expressing MLF using auxiliary functions. Particularly, we demonstrate that the method globally converges to a local minimum or saddle point of MLF. With rigorous theoretical guarantees, the proposed novel SBL algorithm outperforms classical ones in finding sparse representations on simulation and real-world examples, ranging from sparse signal recovery to system identification and kernel regression.

1. Introduction

Sparse Bayesian learning (SBL) originally addresses the issue of obtaining sparse representations over the input space to the corresponding target in supervised learning (Tipping, 2001). Mathematically, we can formalize the problem as finding a sparse solution to the following linearly probabilistic model:

$$\boldsymbol{y} = \boldsymbol{\Phi}(\boldsymbol{x})\boldsymbol{w} + \boldsymbol{\varepsilon}, \tag{1}$$

where $\boldsymbol{y} \in \mathbb{R}^n$ is the output vector, $\boldsymbol{\Phi}(\boldsymbol{x}) \in \mathbb{R}^{n\times m}$ is the overcomplete dictionary matrix composed of basis functions on the input vector $\boldsymbol{x}$, $\boldsymbol{w} \in \mathbb{R}^m$ is the unknown weight vector, and $\boldsymbol{\varepsilon}$ is the uncorrelated Gaussian noise distributed as $\mathcal{N}(0, \lambda \boldsymbol{I}_n)$ with the unknown variance λ. Notably, realizing sparse recovery for $\boldsymbol{w}$ is a fundamental problem in machine learning communities with applications to various science and engineering disciplines, such as signal processing (Donoho, 2006; Kümmerle et al., 2021; Ament & Gomes, 2021a), system identification (Yuan et al., 2019; Sun et al., 2021; 2022; Wang et al., 2023), and regression and classification tasks (Figueiredo, 2003; Ren et al., 2020). Consequently, it is particularly important to design an optimization method with rigorous theoretical guarantees for solving (1) given the sparse constraint on $\boldsymbol{w}$.

To encourage the sparsity of representations, SBL imposes a sparsity-promoting prior $p(\boldsymbol{w} \mid \boldsymbol{\gamma})$ on $\boldsymbol{w}$ to balance model complexity and modeling error (Tipping, 2001; Ament & Gomes, 2021b), where $\boldsymbol{\gamma}$ is the hyperparameter vector. Incorporating the sparsity-promoting prior, the automatic relevance determination framework is naturally implemented to remove irrelevant basis functions (MacKay, 1992; 1995). Following Bayes' rule, combining the prior distribution $p(\boldsymbol{w} \mid \boldsymbol{\gamma})$ and likelihood function $p(\boldsymbol{y} \mid \boldsymbol{w}, \lambda)$ can derive the posterior distribution of $\boldsymbol{w}$ as follows:

$$p(\boldsymbol{w} \mid \boldsymbol{\gamma}, \lambda) \propto p(\boldsymbol{w} \mid \boldsymbol{\gamma}) p(\boldsymbol{y} \mid \boldsymbol{w}, \lambda). \tag{2}$$

Furthermore, the unknown hyperparameters coupled in the posterior distribution can be estimated by maximizing the marginal likelihood function (MLF) as follows:

$$\int p(\boldsymbol{y} \mid \boldsymbol{w}, \lambda) p(\boldsymbol{w} \mid \boldsymbol{\gamma}) p(\boldsymbol{\gamma}) p(\lambda) dw, \tag{3}$$

where $p(\boldsymbol{\gamma})$ and $p(\lambda)$ are non-informative priors imposed on $\boldsymbol{\gamma}$ and λ, respectively. It is also referred to as evidence maximization or type-II maximum likelihood (Neal, 2012; Magris & Iosifidis, 2023). However, current optimization methods used for maximizing the MLF of SBL lack the essential theoretical guarantees of global convergence.

In this paper, we propose a novel optimization method to maximize the MLF of SBL. To this end, we first demonstrate that MLF is the sum of a concave function and a convex function. For such optimization problems, the concave-convex procedure (CCCP) provides an iterative method to

[1]School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan, China [2]State Key Lab of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan, China [3] School of Mathematics and Statistics, Fuyang Normal University, Fuyang, China [4]School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan, China. Correspondence to: Ye Yuan <yye@hust.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

update unknown variables by introducing a latent variable to linearize the concave part (Yuille & Rangarajan, 2001). However, leveraging CCCP to directly solve the MLF of SBL still remains a difficult challenge because the derived subproblem lacks a closed-form solution. To address this issue, we introduce an additional latent variable to re-express the data-dependent term in the subproblem. As such, we propose an iteratively Min-Min optimization method to analytically update the unknown hyperparameters and latent variables in turn conditioned on the current value of the other. In summary, the contributions of this paper are threefold:

1. We propose a novel method to maximize the MLF of SBL, which can optimize the hyperparameters related to both the prior and noise level analytically at each iteration. Additionally, we demonstrate that the proposed optimization method is equivalent to the iteratively reweighted ℓ_2 minimization in the SBL framework. Particularly, the proposed optimization method provides a principled rule to update the regularization and weighting parameters at each iteration, which remains unclear how to obtain such insights to update their values in current ℓ_2 reweighting schemes.

2. Leveraging the Global Convergence Theorem (Luenberger & Ye, 1984), we demonstrate that the proposed optimization method is globally convergent, indicating that the generated series of points converges to a local minimum or saddle point of MLF for any starting point. To the best of our knowledge, we are the first to propose a globally convergent SBL algorithm without fixing the noise level.

3. We demonstrate the proposed novel SBL algorithm on simulation and real-world problems to validate its capability for sparse recovery from overcomplete dictionaries, including sparse signal recovery, system identification, and sparse kernel regression. Benchmarked against classical SBL algorithms, experimental results illustrate its superior performance in finding sparse solutions.

1.1. Related Work

Generally, it is difficult to obtain an estimate of the hyperparameters related to both the prior and noise level by directly maximizing the MLF of SBL (Tipping, 2001). Hence, current studies mainly focus on using iterative optimization methods to address this issue. Here, we briefly review the related work to this paper.

MacKay Updates: In the pioneering work, Tipping (2001) employs the MacKay updates to iteratively estimate the hyperparameters. The basic idea of the MacKay updates is to set the gradient of MLF to zero, and then form the fixed-point updates. While empirically faster to converge, the MacKay updates are unable to guarantee the global convergence (Wipf & Nagarajan, 2007). Additionally, the MacKay updates cannot even ensure an increase in MLF at each iteration.

Expectation-Maximization Algorithm: Except for the MacKay updates, Tipping (2001) also uses the expectation-maximization (EM) algorithm to maximize the MLF of SBL. The EM algorithm is a classical iterative method to find maximum likelihood estimate of MLF, where the probabilistic model typically involves latent variables in addition to unknown hyperparameters. In SBL, the weight vector $\boldsymbol{w}$ is regarded as the latent variable to facilitate the optimization of MLF (Wipf & Rao, 2004b; Zhao et al., 2020). More specifically, in the expectation step, we can conveniently derive the expected log-likelihood with respect to $\boldsymbol{w}$, as it has a closed-form posterior distribution. Accordingly, in the maximization step, we can maximize the expected log-likelihood to find the optimal hyperparameters at the current iteration. Leveraging the EM algorithm, Liu & Rao (2019) further develop a concise SBL method to solve robust principal component analysis problems. Recently, to reduce computational complexity, approximate message passing (AMP) methods are employed to approximate the posterior distribution of $\boldsymbol{w}$ to implement the expectation step (Fang et al., 2016; Al-Shoukairi et al., 2017; Zhang et al., 2023).

Following the discussion in Wipf & Nagarajan (2007); Ament & Gomes (2021b), the EM updates cannot ensure global convergence because they can get trapped in a fixed point instead of a stationary point or local minimum of MLF. Additionally, the EM algorithm has an inherently slow numerical convergence (Wu, 1983).

Variational Inference: In variational inference (VI), the log marginal distribution is decomposed into a lower bound and a Kullback–Leibler (KL) divergence by introducing a variational distribution (Jacobs et al., 2015). Subsequently, VI utilizes the mean-field theory to maximize the lower bound to perform the optimal approximation of the marginal distribution (Blei et al., 2017). Based on VI, Jacobs et al. (2018) develop a computationally efficient SBL algorithm to learn nonlinear autoregressive with exogenous input (NARX) models. Combining VI and Gaussian processes, Jin et al. (2020) present a high precision SBL algorithm to infer the topology of sparse linear networks. Recently, Ray & Szabó (2022) apply VI with Laplace prior slabs to solve high-dimensional sparse linear regression problems. Because the mean-field theory imposes independence assumptions on unknown variables, VI only obtains a distribution close to the joint posterior distribution and thus lacks rigorous theoretical guarantees of global convergence for SBL currently.

Iteratively Reweighted ℓ_1 & ℓ_2 Minimization: Re-expressing the MLF of SBL via auxiliary functions, Wipf &

Nagarajan (2007); Pan et al. (2015) develop an iteratively reweighted ℓ_1 minimization method to optimize the corresponding upper bounding auxiliary cost function. While they demonstrate that the iteratively reweighted ℓ_1 minimization method converges to a local maximum or saddle point of MLF, only a sketch is available for the proof. Additionally, such method needs to set the noise level to be known in theoretical analysis and algorithm design. However, a priori knowledge of the noise level is not available in many cases, and it is application-dependent and potentially sensitive. Using a different upper-bounding auxiliary function, Wipf & Nagarajan (2010) further present an iteratively reweighted ℓ_2 minimization method to optimize the reformulated objective function. The method can either reduce the reformulated objective function or leave it unchanged, but does not fulfill all the technical conditions required to ensure global convergence.

2. Methodology

2.1. Framework of SBL

Basically, SBL infers the weight vector $\boldsymbol{w}$ via its posterior distribution composed of the likelihood and prior functions. In addition, the hyperparameters related to both the prior and noise level, which are coupled in the posterior distribution, are estimated by maximizing the corresponding MLF.

The likelihood function[1] corresponds to (1) is

$$p(\boldsymbol{y} \mid \boldsymbol{w}, \lambda) = (2\pi\lambda)^{-\frac{n}{2}} \exp\left(-\frac{\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{w}\|_2^2}{2\lambda}\right). \tag{4}$$

Note that SBL adopts a hierarchical interpretation of the Student's t-distribution prior to enforce the sparsity of $\boldsymbol{w}$, which is beneficial for the implementation of optimization procedures (MacKay, 1999; Tipping, 2001; Figueiredo, 2003). First, it imposes the Gaussian prior on $\boldsymbol{w}$ as follows:

$$p(\boldsymbol{w} \mid \boldsymbol{\gamma}) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\gamma_i}} \exp\left(-\frac{w_i^2}{2\gamma_i}\right), \tag{5}$$

where w_i is the ith component of $\boldsymbol{w}$, and $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_m]'$ is the hyperparameter vector with γ_i controlling the variance of w_i. To complete the hierarchy, it subsequently defines the Inverse-Gamma distribution over each hyperparameter γ_i as follows:

$$p(\boldsymbol{\gamma}) = \prod_{i=1}^{m} \frac{a_0^{b_0}}{\Gamma(a_0)} \gamma_i^{-a_0-1} \exp\left(-\frac{b_0}{\gamma_i}\right), \tag{6}$$

where $\Gamma(\cdot)$ is the gamma function, a_0 is the shape parameter, and b_0 is the scale parameter. As $p(\boldsymbol{w}) = \int p(\boldsymbol{w}|\boldsymbol{\gamma})p(\boldsymbol{\gamma})d\boldsymbol{\gamma}$ follows the Student's t-distribution, it is equivalent to a two-level hierarchical Bayesian model. Finally, the Inverse-Gamma distribution is also imposed on the remaining hyperparameter λ, severing as a conjugate prior of the likelihood function:

$$p(\lambda) = \frac{a_0^{b_0}}{\Gamma(a_0)} \lambda^{-a_0-1} \exp\left(-\frac{b_0}{\lambda}\right). \tag{7}$$

Generally, a_0 and b_0 are set to small values to generate non-informative priors for $\boldsymbol{\gamma}$ and λ.[2]

For fixed $\boldsymbol{\gamma}$ and λ, combining the likelihood function in (4) and prior distribution in (5) gives the posterior distribution of $\boldsymbol{w}$ as follows:

$$p(\boldsymbol{w} \mid \boldsymbol{\gamma}, \lambda) = \mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}\right), \tag{8}$$

where

$$\boldsymbol{\mu} = \lambda^{-1}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Phi}^T\boldsymbol{y}, \tag{9}$$

$$\boldsymbol{\Sigma} = \lambda^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \boldsymbol{\Gamma}^{-1}, \tag{10}$$

and $\boldsymbol{\Gamma} = \mathrm{diag}[\boldsymbol{\gamma}]$. Based on the *maximum a posteriori* (MAP) principle, the mean $\boldsymbol{\mu}$ is regarded as the estimate of $\boldsymbol{w}$. To estimate $\boldsymbol{\gamma}$ and λ coupled in $\boldsymbol{\mu}$, an essential procedure is to maximize the MLF in (3). Mathematically, this is equivalent to minimizing the negative logarithm of MLF with the form:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\gamma}, \lambda) = &\log|\boldsymbol{\Pi}| + \boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y} + \sum_{i=1}^{m}(2a_0+2)\log\gamma_i \\ &+ \sum_{i=1}^{m}\frac{2b_0}{\gamma_i} + (2a_0+2)\log\lambda + \frac{2b_0}{\lambda},\end{aligned} \tag{11}$$

where $\boldsymbol{\Pi} = \lambda\boldsymbol{I}_n + \boldsymbol{\Phi}\boldsymbol{\Gamma}\boldsymbol{\Phi}^T$.

2.2. Analysis of $\mathcal{L}(\boldsymbol{\gamma}, \lambda)$

The difficulty in minimizing (11) is that it is a non-convex function and λ and $\boldsymbol{\gamma}$ are highly coupled in $\log|\boldsymbol{\Pi}|$ and $\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}$. To develop an efficient algorithm for estimating $\boldsymbol{\gamma}$ and λ, we first demonstrate that $\mathcal{L}(\boldsymbol{\gamma}, \lambda)$ is the sum of a concave function and a convex function. As such, we then design an iterative Min-Min method to optimize (11) based on the CCCP framework.

To start we divide $\mathcal{L}(\boldsymbol{\gamma}, \lambda)$ into the following two parts:

$$\mathcal{L}(\boldsymbol{\gamma}, \lambda) = \mathcal{L}_1(\boldsymbol{\gamma}, \lambda) + \mathcal{L}_2(\boldsymbol{\gamma}, \lambda), \tag{12}$$

where

$$\begin{aligned}\mathcal{L}_1(\boldsymbol{\gamma}, \lambda) = &\log|\boldsymbol{\Pi}| + \sum_{i=1}^{m}(2a_0+2)\log\gamma_i \\ &+ (2a_0+2)\log\lambda,\end{aligned} \tag{13}$$

[1] For the simplicity of notation, $\boldsymbol{\Phi}(\boldsymbol{x})$ is recorded as $\boldsymbol{\Phi}$.

[2] Without the prior knowledge of $\boldsymbol{\gamma}$ and λ, their shape and scale parameters need to be set very small to generate non-informative priors. Hence, we use the same shape and scale parameters for $\boldsymbol{\gamma}$ and λ to simplify notations.

$$\mathcal{L}_2(\boldsymbol{\gamma},\lambda) = \boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y} + \sum_{i=1}^{m}\frac{2b_0}{\gamma_i} + \frac{2b_0}{\lambda}. \tag{14}$$

Lemma 2.1. *$\mathcal{L}_1(\boldsymbol{\gamma},\lambda)$ is strictly concave.*

Proof. As $\log\gamma_i$ and $\log\lambda$ are strictly concave functions, we only need to prove that $\log|\boldsymbol{\Pi}|$ is concave with respect to $\boldsymbol{\gamma}$ and λ. Additionally, $\log|\cdot|$ is concave (see Section 3.1.5 in Boyd & Vandenberghe (2004)). Consequently, we can easily derive the concavity of $\log|\boldsymbol{\Pi}|$ based on the definition of concave functions. □

Lemma 2.2. *$\mathcal{L}_2(\boldsymbol{\gamma},\lambda)$ is strictly convex.*

Proof. It is straightforward to verify the concavity of $\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}$ using Example 3.4 and Section 3.2.2 in Boyd & Vandenberghe (2004) and the strict concavity of $\sum_{i=1}^{m}\frac{2b_0}{\gamma_i}$ and $\frac{2b_0}{\lambda}$ directly. Hence, we can draw the corresponding conclusion. □

2.3. Iterative Min-Min Optimization Method

Based on Lemmas 2.1 and 2.2, we know that $\mathcal{L}(\boldsymbol{\gamma},\lambda)$ is the sum of a concave function $\mathcal{L}_1(\boldsymbol{\gamma},\lambda)$ and a convex function $\mathcal{L}_2(\boldsymbol{\gamma},\lambda)$. Generally, CCCP is a well-known iterative method for solving such optimization problems by linearizing the concave part. For the concave function $\mathcal{L}_1(\boldsymbol{\gamma},\lambda)$, we can re-express it as a minimum over upper-bounding hyperplanes as follows:

$$\mathcal{L}_1(\boldsymbol{\gamma},\lambda) = \min_{\boldsymbol{z}} \langle \boldsymbol{z}, (\boldsymbol{\gamma},\lambda)\rangle - h^*(\boldsymbol{z}), \tag{15}$$

where $h^*(\boldsymbol{z})$ is the concave conjugate of $\mathcal{L}_1(\boldsymbol{\gamma},\lambda)$ with the form:

$$h^*(\boldsymbol{z}) = \min_{\boldsymbol{\gamma},\lambda} \langle \boldsymbol{z}, (\boldsymbol{\gamma},\lambda)\rangle - \mathcal{L}_1(\boldsymbol{\gamma},\lambda). \tag{16}$$

Consequently, leveraging CCCP to minimize (11) gives the following optimization problem:

$$\min_{\boldsymbol{\gamma},\lambda,\boldsymbol{z}} \boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y} + \sum_{i=1}^{m}\frac{2b_0}{\gamma_i} + \frac{2b_0}{\lambda} + \langle \boldsymbol{z}, (\boldsymbol{\gamma},\lambda)\rangle - h^*(\boldsymbol{z}). \tag{17}$$

Subsequently, CCCP updates $\boldsymbol{z}$ and $(\boldsymbol{\gamma},\lambda)$ in turn conditioned on the current value of the other. Given $(\boldsymbol{\gamma}^k,\lambda^k)$ at the kth iteration, the optimal value of $\boldsymbol{z}^k$ equals the slope at the current $(\boldsymbol{\gamma}^k,\lambda^k)$ of $\mathcal{L}_1(\boldsymbol{\gamma},\lambda)$ (Boyd & Vandenberghe, 2004; Wipf & Nagarajan, 2007), resulting in

$$\begin{aligned}\boldsymbol{z}^k =& \nabla_{\boldsymbol{\gamma}^k,\lambda^k}\mathcal{L}_1(\boldsymbol{\gamma},\lambda)\\ =& \Bigg(\text{diag}\left[\boldsymbol{\Phi}^T(\boldsymbol{\Pi}^k)^{-1}\boldsymbol{\Phi}\right] + (2a_0+2)\text{diag}\left[(\boldsymbol{\Gamma}^k)^{-1}\right],\\ & \text{Tr}\left((\boldsymbol{\Pi}^k)^{-1}\right) + \frac{2a_0+2}{\lambda^k}\Bigg),\end{aligned} \tag{18}$$

where $\boldsymbol{\Pi}^k = \lambda^k\boldsymbol{I}_n + \boldsymbol{\Phi}\boldsymbol{\Gamma}^k\boldsymbol{\Phi}^T$ and $\boldsymbol{\Gamma}^k = \text{diag}[\boldsymbol{\gamma}^k]$. Given $\boldsymbol{z}^k$, CCCP needs to solve the following problem to update $(\boldsymbol{\gamma}^{k+1},\lambda^{k+1})$:

$$\min_{\boldsymbol{\gamma},\lambda} \boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y} + \sum_{i=1}^{m}\frac{2b_0}{\gamma_i} + \frac{2b_0}{\lambda} + \left\langle \boldsymbol{z}^k, (\boldsymbol{\gamma},\lambda)\right\rangle. \tag{19}$$

Because $\boldsymbol{\gamma}$ and λ are highly coupled in $\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}$, it is difficult to derive a closed-form solution of $(\boldsymbol{\gamma}^{k+1},\lambda^{k+1})$ by directly minimizing (19). For the Gaussian prior with fixed noise level λ, Wipf & Nagarajan (2007); Pan et al. (2015); Yuan et al. (2023) present an iteratively reweighted ℓ_1 minimization method to estimate $\boldsymbol{\gamma}$ following the CCCP framework. However, obtaining a closed-form solution of $\boldsymbol{\gamma}$ at each iteration still remains challenging in algorithm implementation due to its non-differentiability at the origin. Additionally, λ is not available in many cases, and it is generally application-dependent.

To address such issues, we further re-express $\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}$ by introducing the latent variable $\boldsymbol{\theta}$, which yields

$$\begin{aligned}\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y} &= \frac{1}{\lambda}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\mu}\|_2^2 + \boldsymbol{\mu}^T\boldsymbol{\Gamma}^{-1}\boldsymbol{\mu}\\ &= \min_{\boldsymbol{\theta}} \frac{1}{\lambda}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}\|_2^2 + \boldsymbol{\theta}^T\boldsymbol{\Gamma}^{-1}\boldsymbol{\theta}.\end{aligned} \tag{20}$$

Detailed derivations of (20) can be found in Appendix of Tipping (2001); Pan et al. (2015). Hence, we can re-express (19) as follows:

$$\begin{aligned}\min_{\boldsymbol{\gamma},\lambda}\Bigg(&\min_{\boldsymbol{\theta}}\left(\frac{1}{\lambda}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}\|_2^2 + \boldsymbol{\theta}^T\boldsymbol{\Gamma}^{-1}\boldsymbol{\theta}\right)\\ &+ \left\langle \boldsymbol{z}^k, (\boldsymbol{\gamma},\lambda)\right\rangle + \sum_{i=1}^{m}\frac{2b_0}{\gamma_i} + \frac{2b_0}{\lambda}\Bigg).\end{aligned} \tag{21}$$

To facilitate the optimization process, we first consider minimizing the term related to $\boldsymbol{\theta}$ given $(\boldsymbol{\gamma}^k,\lambda^k)$:

$$\min_{\boldsymbol{\theta}} \frac{1}{\lambda^k}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}\|_2^2 + \boldsymbol{\theta}^T\left(\boldsymbol{\Gamma}^k\right)^{-1}\boldsymbol{\theta}, \tag{22}$$

resulting in the optimal value

$$\boldsymbol{\theta}^k = (\lambda^k)^{-1}\left((\lambda^k)^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \left(\boldsymbol{\Gamma}^k\right)^{-1}\right)^{-1}\boldsymbol{\Phi}^T\boldsymbol{y}. \tag{23}$$

Given $(\boldsymbol{z}^k,\boldsymbol{\theta}^k)$, we can obtain the optimal value of $(\boldsymbol{\gamma}^{k+1},\lambda^{k+1})$ by minimizing

$$\begin{aligned}\mathcal{L}_{\boldsymbol{z}^k,\boldsymbol{\theta}^k}(\boldsymbol{\gamma},\lambda) =& \left\langle \boldsymbol{z}^k, (\boldsymbol{\gamma},\lambda)\right\rangle + \frac{1}{\lambda}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}^k\|_2^2\\ &+ \left(\boldsymbol{\theta}^k\right)^T\boldsymbol{\Gamma}^{-1}\boldsymbol{\theta}^k + \sum_{i=1}^{m}\frac{2b_0}{\gamma_i} + \frac{2b_0}{\lambda}.\end{aligned} \tag{24}$$

Because $\mathcal{L}_{\boldsymbol{z}^k,\boldsymbol{\theta}^k}(\boldsymbol{\gamma},\lambda)$ is a convex function in $(\boldsymbol{\gamma},\lambda)$, the optimal value of $(\boldsymbol{\gamma}^{k+1},\lambda^{k+1})$ can be obtained by letting

its gradient be zero:

$$\gamma_i^{k+1} = \sqrt{\frac{\left(\theta_i^k\right)^2 + 2b_0}{z_i^k}}, \tag{25}$$

$$\lambda^{k+1} = \sqrt{\frac{\|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\theta}^k\|_2^2 + 2b_0}{z_{m+1}^k}}, \tag{26}$$

where θ_i^k and z_i^k are the ith components of $\boldsymbol{\theta}^k$ and $\boldsymbol{z}^k$, respectively. Repeating the above update procedures until convergence, we present an iterative Min-Min optimization method to analytically update $(\boldsymbol{\gamma}, \lambda)$ and $(\boldsymbol{z}, \boldsymbol{\theta})$ in turn conditioned on the current value of the other. Finally, we summarize the proposed novel SBL algorithm in Algorithm 1 below.

Algorithm 1 Iterative Min-Min optimization method

Input: Input vector $\boldsymbol{x}$, output vector $\boldsymbol{y}$, dictionary matrix $\boldsymbol{\Phi}$
Initialize $\boldsymbol{\gamma}^0$ and λ^0
repeat
  Update $\boldsymbol{z}^k$ via (18) conditioned on $\boldsymbol{\gamma}^k$ and λ^k
  Update $\boldsymbol{\theta}^k$ via (23) conditioned on $\boldsymbol{\gamma}^k$ and λ^k
  Update $\boldsymbol{\gamma}^{k+1}$ via (25) conditioned on $\boldsymbol{z}^k$ and $\boldsymbol{\theta}^k$
  Update λ^{k+1} via (26) conditioned on $\boldsymbol{z}^k$ and $\boldsymbol{\theta}^k$
until the algorithm converges
Compute $\boldsymbol{\mu}$ via (9) conditioned on the final $(\boldsymbol{\gamma}, \lambda)$, and adopt it as the estimate of $\boldsymbol{w}$

3. Theoretical Analysis

In this section, we demonstrate that the proposed optimization method is globally convergent, indicating that the generated series of points converges to a local minimum or saddle point of $\mathcal{L}(\boldsymbol{\gamma}, \lambda)$ for any starting point. As the proposed algorithm presents analytical update procedures to optimize $(\boldsymbol{\gamma}, \lambda)$, we can take it as a point-to-point mapping $\mathcal{A}(\cdot)$ that is denoted explicitly by simple mathematical expressions given in Algorithm 1.

Theorem 3.1. *For any starting point* $(\boldsymbol{\gamma}^0, \lambda^0) \in \mathbb{R}_+^{n+1}$*, the sequence* $\{(\boldsymbol{\gamma}^k, \lambda^k) \in \mathbb{R}_+^{n+1}\}$ *generated via* $(\boldsymbol{\gamma}^{k+1}, \lambda^{k+1}) = \mathcal{A}(\boldsymbol{\gamma}^k, \lambda^k)$ *converges monotonically to a local minimum (or saddle point) of* $\mathcal{L}(\boldsymbol{\gamma}, \lambda)$.

The detailed proof of Theorem 3.1 can be found in Appendix A. To the best of our knowledge, this paper for the first time proposes a globally convergent optimization method to estimate $\boldsymbol{\gamma}$ and λ. Particularly, The proposed novel SBL algorithm provides an analytical update rule at each iteration, making it straightforward to be implemented and used.

4. Connection to Iteratively Reweighted ℓ_2 Minimization

In this section, we explore the connection between the proposed iterative optimization method and MAP estimation in $\boldsymbol{w}$ space. We find that the proposed optimization method is equivalent to minimizing the MLF of SBL via iteratively reweighted ℓ_2 minimization method. However, the proposed optimization method is superior to traditional iterative reweighting schemes because it provides an update procedure for both the regularization and weighting parameters at each iteration.

4.1. MAP Estimation in $\boldsymbol{w}$ Space

The canonical form of the MAP estimation for finding maximally sparse representations from overcomplete dictionaries involves solving

$$\min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{w}\|_2^2 + r\|\boldsymbol{w}\|_{\ell_0}, \tag{27}$$

where r is the regularization parameter. However, it is a well-known NP-hard problem (Wipf & Rao, 2004a; Wipf & Nagarajan, 2010; Kümmerle et al., 2021). Hence, it is necessary to make some tractable approximations to replace $\|\boldsymbol{w}\|_{\ell_0}$. To address this issue, the ℓ_1 or ℓ_2 norm is commonly used as a proxy of the ℓ_0 norm, leading to a convex optimization problem:

$$\min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{w}\|_2^2 + r\|\boldsymbol{w}\|_{\ell_1/\ell_2}. \tag{28}$$

However, the ℓ_1 minimization requires more measurements to recover sparse solutions (Chartrand & Staneva, 2008; Wen et al., 2017) and the ℓ_2 minimization cannot generate exactly sparse solutions (Tibshirani, 1996). Consequently, iteratively reweighted ℓ_1 and ℓ_2 minimization algorithms are proposed to alleviate the corresponding problems (Daubechies et al., 2008; Chartrand & Yin, 2008; Candes et al., 2008). In particular, a series of empirical experiments demonstrates that the iteratively reweighted ℓ_2 minimization is superior to other tractable ones (Chartrand & Yin, 2008; Wipf & Nagarajan, 2010).

Employing the iteratively reweighted ℓ_2 minimization yields the following tractable form at the kth iteration:

$$\min_{\boldsymbol{w}} \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{w}\|_2^2 + r\sum_{i=1}^{m} \frac{w_i^2}{\alpha_i^k}, \tag{29}$$

where α_i^k is the weighting parameter. While several heuristic procedures are given to update α_i^k (Chartrand & Yin, 2008; Wipf & Nagarajan, 2010), how to set or update the value of r remains elusive. In the following subsection, we will show that the proposed optimization method can naturally address this issue, as it automatically updates the regularization and weighting parameters at each iteration.

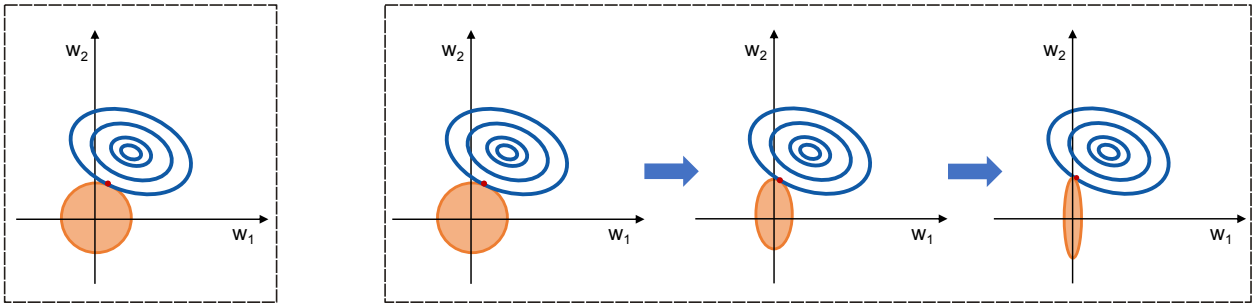

Figure 1. Comparison between the ℓ_2 minimization (left) and the proposed iteratively reweighted ℓ_2 minimization (right) for sparse recovery. Compared with the ℓ_2 minimization, the proposed algorithm will heavily penalize some w_i as the number of iterations increases, compelling them toward zero.

4.2. Iteratively Reweighted ℓ_2 Minimization

Based on MAP, the posterior mean $\boldsymbol{\mu}$ is adopted as the estimate of $\boldsymbol{w}$. Hence, (23) indicates that the latent variable $\boldsymbol{\theta}$ gives the optimal estimate of $\boldsymbol{w}$ at each iteration. Following the optimization procedure of $\boldsymbol{\theta}$, we actually solve a series of reformulated ℓ_2 minimization problems to update $\boldsymbol{w}$. This is summarized in the theorem below.

Theorem 4.1. *The posterior mean vector $\boldsymbol{\mu}$ in (9) derived from the proposed optimization method can be obtained by iteratively solving the following MAP problem:*

$$\min_{\boldsymbol{w}} \|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{w}\|_2^2+\lambda^k\sum_{i=1}^{m}\frac{w_i^2}{\gamma_i^k}, \tag{30}$$

where γ_i^k and λ^k represent weighting and regularization parameters, respectively. Additionally, the proposed optimization method can update γ_i^k and λ^k via (25) and (26) at each iteration, respectively.

Proof. We can draw the conclusion by rearranging (22). □

Theorem 4.1 demonstrates that the proposed optimization method can be interpreted as an iteratively reweighted ℓ_2 minimization method, where the regularization and weighting parameters are automatically updated at each iteration. Such update procedures can be demonstrated to produce a sequence of $(\boldsymbol{\gamma},\lambda)$ that converges to a local minimum or saddle point of $\mathcal{L}(\boldsymbol{\gamma},\lambda)$. We can also consider an alternative form of (30) at each iteration as follows:

$$\min_{\boldsymbol{w}} \|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{w}\|_2^2 \qquad \text{s.t.}\sum_{i=1}^{m}\frac{w_i^2}{\gamma_i^k}\le r, \tag{31}$$

where r is the radius related to λ^k. Figure 1 provides an intuitive explanation of how the proposed optimization method drives some w_i towards zero. As the number of iterations increases, many γ_i^k empirically tend toward zero. Hence, $1/\gamma_i^k$ will heavily penalize the corresponding w_i, compelling it to approach zero as well.

4.3. Additional Constraints on $\boldsymbol{w}$

Expect for the sparsity constraint, we often need to consider additional constraints on $\boldsymbol{w}$ in many cases. For example, nonzero coefficients should be positive in non-negative sparse coding. However, many classical SBL algorithms cannot integrate additional constraints into optimization frameworks (e.g., non-negativity). For the proposed optimization framework, Theorem 4.1 implies that we minimize a convex function at each iteration to update $\boldsymbol{w}$. Consequently, we can easily incorporate the following constraints into the problem formulation:

$$f_i(\boldsymbol{w})\le 0, \quad i=1,\ldots,p, \tag{32}$$
$$h_i(\boldsymbol{w})=0, \quad i=1,\ldots,q, \tag{33}$$

where $f_i(\boldsymbol{w}):\mathbb{R}^m\to\mathbb{R}$ is the convex function and $h_i(\boldsymbol{w}):\mathbb{R}^m\to\mathbb{R}$ is the affine function. With such constraints, we finally derive a convex optimization problem, which can be optimized efficiently (Boyd & Vandenberghe, 2004). Note that while we cannot still guarantee the global convergence of the corresponding algorithm, we provide an efficient method for handling the additional constraints (32) and (33).

5. Experiments

In this section, we demonstrate the performance of the proposed SBL algorithm on a number of examples in various fields, including sparse signal reconstruction, system identification, and sparse kernel regression.[3] Additionally, we compare the proposed novel SBL algorithm with the classical SBL algorithms mentioned previously, including

[3]Source codes are available on GitHub at https://github.com/ArthinYS/MinMinSBL.

Table 1. Probability of successfully recovering sparse signals on the Gaussian-distributed matrix at different SNR levels.

Sparse signal	Method	SNR							
		0 dB	5 dB	10 dB	15 dB	20 dB	25 dB	30 dB	35 dB
± 1 spike signal	**Ours**	**22%**	**74%**	**82%**	**83%**	**83%**	**80%**	**91%**	**100%**
	MacKay_SBL	0	0	0	0	0	0	2%	48%
	EM_SBL	0	0	0	0	0	0	0	7%
	IR_SBL	0	0	0	0	0	0	0	1%
	VI_SBL	0	0	0	0	0	0	0	0
Gaussian signal	**Ours**	**3%**	**14%**	**39%**	**56%**	**63%**	**72%**	**84%**	**94%**
	MacKay_SBL	0	0	0	0	0	0	15%	48%
	EM_SBL	0	0	0	0	0	0	8%	37%
	IR_SBL	0	0	0	0	0	0	2%	3%
	VI_SBL	0	0	0	0	0	0	0	0
Uniformly distributed signal	**Ours**	**9%**	**34%**	**50%**	**63%**	**61%**	**71%**	**87%**	**90%**
	MacKay_SBL	0	0	0	0	0	7%	44%	82%
	EM_SBL	0	0	0	0	0	4%	25%	84%
	IR_SBL	0	0	0	0	0	0	0	7%
	VI_SBL	0	0	0	0	0	0	0	0

MacKay_SBL (MacKay updates), EM_SBL (EM updates), VI_SBL (VI updates), and IR_SBL (iteratively reweighted ℓ_1 minimization updates). To keep a fair comparison, the initial values of $\boldsymbol{\gamma}$ and λ are set to be the same for all the SBL algorithms in all experiments. Overall, experimental results illustrate that the proposed SBL algorithm outperforms the classical ones in finding sparse solutions. As AMP provides an alternative Bayesian approach for sparse recovery, we also compare the proposed SBL algorithm with AMP to demonstrate its superior performance in Appendix B. The experiments are conducted using MATLAB 2022b on the PC with an Apple M1 Pro chip with 10-core CPU and 32 GB RAM.

5.1. Sparse Signal Recovery

Recovering sparse signals with random dictionaries is an important benchmark to evaluate the performance of SBL algorithms (Wipf & Rao, 2004b; Babacan et al., 2009; Zhou et al., 2021). To this end, we generate an $n \times m$ random matrix $\boldsymbol{\Phi}$ and an m-dimensional sparse signal $\boldsymbol{w}$ with k nonzero coefficients at random locations. Particularly, we consider that the nonzero coefficients in $\boldsymbol{w}$ are drawn from three different probability density functions (PDFs): the uniform ± 1 random spike, standardized Gaussian distribution, and uniform distribution on [-1,1]. Finally, we use the Matlab function **awgn** to add Additive White Gaussian Noise on $\boldsymbol{\Phi w}$ to get a resultant signal $\boldsymbol{y}$ with a given signal-to-noise ratio (SNR). In the experiments, we set $n = 60$, $m = 100$, and $k = 4$, and conduct simulation trials with SNR values ranging from 0 to 35 dB in steps of 5 dB. In all cases, we run 100 independent trials to test the performance of all the SBL algorithms. Additionally, a successful trial is recorded if the indices of nonzero elements in the estimated vector $\boldsymbol{w}$ are the same as true indices.

Gaussian-distributed matrix $\boldsymbol{\Phi}$: First, we consider that each component of the matrix $\boldsymbol{\Phi}$ is drawn from a standardized Gaussian distribution. Table 1 records the probability of successfully recovering sparse signals for all the SBL algorithms across different SNR values over 100 runs. For each SNR, the proposed SBL algorithm realizes the highest success probability in recovering sparse signals. When SNR is below 25 dB, only the proposed SBL algorithm has the potential to recover sparse signals. Particularly, the proposed SBL algorithm achieves a 100% recovery rate for the ± 1 spike signal with high SNR (35 dB), while the success probabilities of the others are below 50%. Remarkably, this example demonstrates that the proposed SBL algorithm significantly outperforms the others on sparse signal recovery.

Low-rank matrix $\boldsymbol{\Phi}$: Further, we consider the random matrix $\boldsymbol{\Phi}$ to be a low-rank matrix to test the robustness of all the SBL algorithms. Initially, we generate a random matrix $\boldsymbol{\Phi}$ with each component drawn from a uniform distribution on [0,1]. Subsequently, we utilize the truncated singular value decomposition (SVD) to generate a low-rank matrix with rank r (Falini, 2022). Here, we set r to be 40. Table 2 records the probability of successfully recovering sparse signals for all the SBL algorithms across different SNR values over 100 runs. Similarly, the experimental results validate that the proposed SBL algorithm outperforms the classical ones on the low-rank matrix $\boldsymbol{\Phi}$. In all cases, the success probabilities of the classical SBL algorithms are

Table 2. Probability of successfully recovering sparse signals on the low-rank matrix at different SNR levels.

Sparse signal	Method	SNR							
		0 dB	5 dB	10 dB	15 dB	20 dB	25 dB	30 dB	35 dB
±1 spike signal	**Ours**	**10%**	**41%**	**74%**	**84%**	**91%**	**90%**	**90%**	**93%**
	MacKay_SBL	0	0	0	0	0	0	0	33%
	EM_SBL	0	0	0	0	0	0	0	18%
	IR_SBL	6%	3%	6%	9%	9%	9%	7%	10%
	VI_SBL	0	0	0	0	0	0	0	0
Gaussian signal	**Ours**	2%	**8%**	**30%**	**50%**	**55%**	**66%**	**77%**	**80%**
	MacKay_SBL	0	0	0	0	0	2%	7%	30%
	EM_SBL	0	0	0	0	0	1%	4%	17%
	IR_SBL	**9%**	2%	3%	5%	4%	4%	2%	8%
	VI_SBL	0	0	0	0	0	0	0	0
Uniformly distributed signal	**Ours**	2%	**15%**	**43%**	**61%**	**73%**	**67%**	**82%**	**89%**
	MacKay_SBL	0	0	0	0	0	2%	22%	58%
	EM_SBL	0	0	0	0	0	0	14%	47%
	IR_SBL	**9%**	8%	7%	5%	2%	7%	4%	5%
	VI_SBL	0	0	0	0	0	0	0	0

below 60%. In contrast, the proposed SBL algorithm can recover sparse signals with a success probability over 80% under high SNR conditions.

5.2. Learning the Chaotic Lorenz System

Leveraging machine learning algorithms to discover the governing equations or physical laws of nonlinear dynamical systems from data is essential to understanding such systems for prediction, control, and decision-making (Brunton et al., 2016). Recently, SBL algorithms have been applied to discover linear dynamical systems (Wang et al., 2024), nonlinear state-space systems (Pan et al., 2015), cyber physical systems (Yuan et al., 2019), partial differential equations (Yuan et al., 2023), and stochastic differential equations (Wang et al., 2022). Here, we demonstrate all the SBL algorithms on the canonical chaotic Lorenz system with the following form:

$$\dot{x} = \sigma(y - x), \tag{34}$$

$$\dot{y} = x(\rho - z) - y, \tag{35}$$

$$\dot{z} = xy - \beta z, \tag{36}$$

where $\sigma = 10$, $\rho = 28$, and $\beta = \frac{8}{3}$.

As for the algorithm implementation, we use the MATLAB function **ode45** to solve the system with the initial condition $[x, y, z]^T = [-8, 7, 27]^T$ and obtain data with a time step of 0.001 over the time interval $[0, 65]$. Then, we uniformly sub-sample 65 data points from the collected data. We add the Gaussian noise with mean zero and different variances to derivatives to generate noisy data. The basis functions are composed of polynomials in (x, y, z) up to the fourth order and some sine and cosine functions, including 95 terms totally. Finally, we apply all the SBL algorithms to discover the chaotic Lorenz system from the overcomplete dictionary $\mathbf{\Phi}^{65\times 95}$.

Figure 2 displays the probability of successfully discovering the Lorenz system of all the SBL algorithms across different noise levels (over 100 runs). Similarly, experimental results imply that the proposed SBL algorithm is superior to the classical ones. In all cases, IR_SBL and VI_SBL fail to discover the Lorenz system from the limited data. With the lowest level of noise, the proposed SBL algorithm, MacKay_SBL, and EM_SBL can perfectly discover the Lorenz system from 95 basis functions using only 65 data points. However, the proposed SBL algorithm exhibits overwhelming performance as the noise level increases, implying that it is more robust to noise. Therefore, the proposed SBL algorithm is more likely to discover the underlying mechanisms of physical systems from limited data compared with the others.

5.3. Sparse Kernel Regression

Finally, we apply all the SBL algorithms to kernel regression models on the *Red Wine Quality* dataset, which contains the information concerning wine quality. It includes 1599 samples totally with 1 target variable and 11 features. In the experiment, we split 1599 data points into 1000 for training and 599 for testing. The basis functions include kernel functions and a constant term. Particularly, we consider four different kernel functions: linear, Matérn-3/2, exponential, and Gaussian kernels. The hyperparameters in the

Table 3. Regression result of all the SBL algorithms across the different kernels.

Method	Ours	MacKay_SBL	EM_SBL	IR_SBL	VI_SBL
Exponential kernel	**0.0951**	0.4329	0.9939	0.1006	0.1270
Matérn-3/2 Kernel	**0.0957**	0.1341	1.0048	0.1006	0.1270
Linear kernel	0.0970	**0.0952**	1.0044	0.1058	0.1561
Gaussian kernel	**0.0980**	0.1123	1.0045	0.1004	0.1270

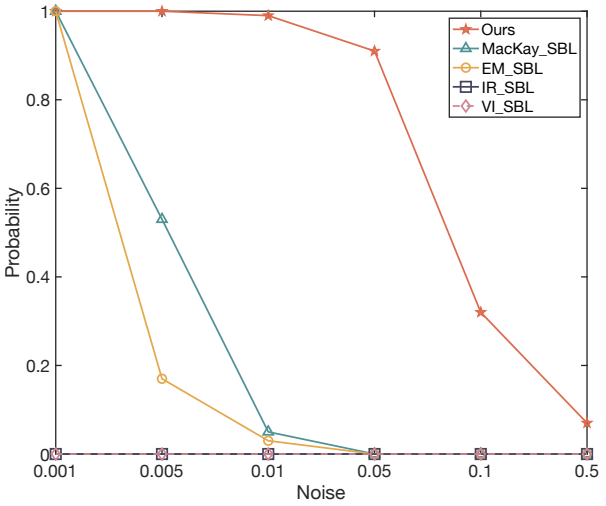

Figure 2. Probability of successfully discovering the Lorenz system at different noise levels.

kernels are set to 1 by default. We also conduct additional experiments to compare all the SBL algorithms on the Gaussian kernel with the different values of hyperparameters in Appendix C.

Here, we define *Sparsity* as the ratio of non-zero elements in the estimated weight vector $\boldsymbol{w}$, and *MRE* as the mean relative error between the ground truth and predictions on the test set. Because the smaller values of *Sparsity* and *MRE* indicate sparser models and more accurate prediction results, respectively, the distance between the point (*Sparsity*, *MRE*) and origin (0,0) can assess the ability of an SBL algorithm regarding balancing model complexity and prediction error in real-world problems. Hence, we can use the following metric to evaluate the performance of the SBL algorithm:

$$d = \sqrt{Sparsity^2 + MRE^2}. \tag{37}$$

Table 3 summarizes the regression result of all the SBL algorithms on the *Red Wine Quality* dataset with the different kernel functions. As observed from the table, the proposed SBL algorithm almost outperforms the classical ones across the different kernel functions. Consequently, the experimental results demonstrate its superior performance to balance model complexity and prediction error in real-world problems.

6. Discussion and Limitation

In this paper, we propose an iterative Min-Min optimization method for minimizing the negative logarithm of MLF $\mathcal{L}(\boldsymbol{\gamma}, \lambda)$. Because $\mathcal{L}(\boldsymbol{\gamma}, \lambda)$ is a non-convex function and $\boldsymbol{\gamma}$ and λ are highly coupled in the data-dependent term $\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}$ and concave term $\mathcal{L}_1(\boldsymbol{\gamma}, \lambda)$, directly minimizing such a function is particularly difficult. To develop an efficient algorithm for estimating $\boldsymbol{\gamma}$ and λ, we demonstrate that $\mathcal{L}(\boldsymbol{\gamma}, \lambda)$ is composed of a concave function and a convex function. While CCCP is designed to solve such optimization problems, the resulting subproblem does not have a closed-form solution in this case. Hence, we use auxiliary functions to re-express not only the concave function $\mathcal{L}_1(\boldsymbol{\gamma}, \lambda)$ but also data-dependent term $\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}$ by introducing latent variables. As such, we can decouple the highly coupled functions and convert the non-convex problem into a Min-Min problem, leading to efficient updates based on the CCCP framework. Leveraging the Global Convergence Theorem, we further demonstrate that the generated sequence of points converges to a local minimum or saddle point of the MLF of SBL. With rigorous theoretical guarantees, experimental results illustrate that the proposed novel SBL algorithm outperforms the classical ones in finding sparse representations in various fields.

We also explore the connection between the proposed optimization algorithm and MAP estimation. Theorem 4.1 demonstrates that the proposed optimization method is equivalent to the iteratively reweighted ℓ_2 minimization. Particularly, the proposed optimization method provides a principled way to update the regularization and weighting parameters at each iteration. In MAP estimation, it remains unclear how to obtain such insights for updating their values.

While we present a simple and analytical update rule for the hyperparameters $\boldsymbol{\gamma}$ and λ, the primary limitation lies in its high computational complexity. At each iteration, the update rule entails the inversion of $\boldsymbol{\Sigma} \in \mathbb{R}^{m\times m}$ and $\boldsymbol{\Pi} \in \mathbb{R}^{n\times n}$. Given $n < m$, we can save the computation cost by re-expressing $\boldsymbol{\Sigma}^{-1}$ using $\boldsymbol{\Pi}^{-1}$ (Wipf & Rao, 2004b). However, it is still hard to deal with the problems with large training sets as computational requirements scale with n^3. Consequently, future work will focus on reducing computation time to make the proposed algorithm applicable to large-scale problems.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant Nos. 92167201, 52205520, and 62303118). The authors are also grateful for the constructive feedback from the reviewers.

Impact Statement

This paper presents a new variant of SBL that improves the performance in finding the sparse solution of linearly probabilistic models. Furthermore, the proposed SBL algorithm, with its simple form and theoretical guarantees, can be applied to various science and engineering disciplines, such as selecting important features from data and furnishing interpretable models. In addition, the proposed optimization framework and detailed proofs have the potential to inspire future research in SBL.

References

Al-Shoukairi, M., Schniter, P., and Rao, B. D. A GAMP-based low complexity sparse Bayesian learning algorithm. *IEEE Transactions on Signal Processing*, 66(2):294–308, 2017.

Ament, S. and Gomes, C. On the optimality of backward regression: Sparse recovery and subset selection. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 5599–5603, 2021a.

Ament, S. E. and Gomes, C. P. Sparse Bayesian learning via stepwise regression. In *International Conference on Machine Learning*, pp. 264–274, 2021b.

Babacan, S. D., Molina, R., and Katsaggelos, A. K. Bayesian compressive sensing using Laplace priors. *IEEE Transactions on Image Processing*, 19(1):53–63, 2009.

Bayati, M. and Montanari, A. The dynamics of message passing on dense graphs, with applications to compressed sensing. *IEEE Transactions on Information Theory*, 57 (2):764–785, 2011.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Boyd, S. P. and Vandenberghe, L. *Convex optimization*. Cambridge University Press, 1st edition, 2004.

Brunton, S. L., Proctor, J. L., and Kutz, J. N. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

Candes, E. J., Wakin, M. B., and Boyd, S. P. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14:877–905, 2008.

Chartrand, R. and Staneva, V. Restricted isometry properties and nonconvex compressive sensing. *Inverse Problems*, 24(3):035020, 2008.

Chartrand, R. and Yin, W. Iteratively reweighted algorithms for compressive sensing. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3869–3872, 2008.

Daubechies, I., DeVore, R., Fornasier, M., and Gunturk, S. Iteratively re-weighted least squares minimization: Proof of faster than linear rate for sparse recovery. In *Annual Conference on Information Sciences and Systems*, pp. 26–29, 2008.

Donoho, D. L. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

Falini, A. A review on the selection criteria for the truncated SVD in data science applications. *Journal of Computational Mathematics and Data Science*, 5:100064, 2022.

Fang, J., Zhang, L., and Li, H. Two-dimensional pattern-coupled sparse Bayesian learning via generalized approximate message passing. *IEEE Transactions on Image Processing*, 25(6):2920–2930, 2016.

Figueiredo, M. A. Adaptive sparseness for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1150–1159, 2003.

Jacobs, W. R., Baldacchino, T., and Anderson, S. R. Sparse Bayesian identification of polynomial NARX models. *IFAC-PapersOnLine*, 48(28):172–177, 2015.

Jacobs, W. R., Baldacchino, T., Dodd, T., and Anderson, S. R. Sparse Bayesian nonlinear system identification using variational inference. *IEEE Transactions on Automatic Control*, 63(12):4172–4187, 2018.

Jin, J., Yuan, Y., and Gonçalves, J. High precision variational Bayesian inference of sparse linear networks. *Automatica*, 118:109017, 2020.

Kümmerle, C., Mayrink Verdun, C., and Stöger, D. Iteratively reweighted least squares for basis pursuit with global linear convergence rate. In *Advances in Neural Information Processing Systems*, pp. 2873–2886, 2021.

Liu, J. and Rao, B. D. Sparse Bayesian learning for robust PCA: Algorithms and analyses. *IEEE Transactions on Signal Processing*, 67(22):5837–5849, 2019.

Luenberger, D. G. and Ye, Y. *Linear and nonlinear programming*. Springer, 2nd edition, 1984.

MacKay, D. J. Bayesian interpolation. *Neural Computation*, 4(3):415–447, 1992.

MacKay, D. J. Probable networks and plausible predictions-a review of practical Bayesian methods for supervised neural networks. *Network: Computation in Neural Systems*, 6(3):469, 1995.

MacKay, D. J. Comparison of approximate methods for handling hyperparameters. *Neural Computation*, 11(5): 1035–1068, 1999.

Magris, M. and Iosifidis, A. Bayesian learning for neural networks: An algorithmic survey. *Artificial Intelligence Review*, pp. 1–51, 2023.

Neal, R. M. *Bayesian learning for neural networks*. Springer Science & Business Media, 1st edition, 2012.

Pan, W., Yuan, Y., Gonçalves, J., and Stan, G.-B. A sparse Bayesian approach to the identification of nonlinear state-space systems. *IEEE Transactions on Automatic Control*, 61(1):182–187, 2015.

Rangan, S. Generalized approximate message passing for estimation with random linear mixing. In *IEEE International Symposium on Information Theory Proceedings*, pp. 2168–2172. IEEE, 2011.

Ray, K. and Szabó, B. Variational Bayes for high-dimensional linear regression with sparse priors. *Journal of the American Statistical Association*, 117(539):1270–1281, 2022.

Ren, S., Zhao, W., and Li, P. Thunder: A fast coordinate selection solver for sparse learning. In *Advances in Neural Information Processing Systems*, pp. 1571–1582, 2020.

Sun, F., Liu, Y., and Sun, H. Physics-informed spline learning for nonlinear dynamics discovery. In *International Joint Conference on Artificial Intelligence*, pp. 2054–2061, 2021.

Sun, L., Huang, D., Sun, H., and Wang, J.-X. Bayesian spline learning for equation discovery of nonlinear dynamics with quantified uncertainty. In *Advances in Neural Information Processing Systems*, pp. 6927–6940, 2022.

Tibshirani, R. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

Tipping, M. E. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

Wang, Y., Fang, H., Jin, J., Ma, G., He, X., Dai, X., Yue, Z., Cheng, C., Zhang, H.-T., Pu, D., et al. Data-driven discovery of stochastic differential equations. *Engineering*, 17:244–252, 2022.

Wang, Y., Cheng, C., Sun, H., Jin, J., and Fang, H. Data augmentation-based statistical inference of diffusion processes. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 33(3), 2023.

Wang, Y., Yuan, Y., Fang, H., and Ding, H. Data-driven discovery of linear dynamical systems from noisy data. *Science China Technological Sciences*, 67(1):121–129, 2024.

Wen, F., Pei, L., Yang, Y., Yu, W., and Liu, P. Efficient and robust recovery of sparse signal and image using generalized nonconvex regularization. *IEEE Transactions on Computational Imaging*, 3(4):566–579, 2017.

Wipf, D. and Nagarajan, S. A new view of automatic relevance determination. In *Advances in Neural Information Processing Systems*, 2007.

Wipf, D. and Nagarajan, S. Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions. *IEEE Journal of Selected Topics in Signal Processing*, 4(2):317–329, 2010.

Wipf, D. and Rao, B. ℓ_0-norm minimization for basis selection. In *Advances in Neural Information Processing Systems*, pp. 1513–1520, 2004a.

Wipf, D. P. and Rao, B. D. Sparse Bayesian learning for basis selection. *IEEE Transactions on Signal Processing*, 52(8):2153–2164, 2004b.

Wu, C. J. On the convergence properties of the EM algorithm. *The Annals of Statistics*, pp. 95–103, 1983.

Yuan, Y., Tang, X., Zhou, W., Pan, W., Li, X., Zhang, H.-T., Ding, H., and Goncalves, J. Data driven discovery of cyber physical systems. *Nature Communications*, 10(1): 4894, 2019.

Yuan, Y., Li, X., Li, L., Jiang, F. J., Tang, X., Zhang, F., Goncalves, J., Voss, H. U., Ding, H., and Kurths, J. Machine discovery of partial differential equations from spatiotemporal data: A sparse Bayesian learning framework. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 33(11), 2023.

Yuille, A. L. and Rangarajan, A. The concave-convex procedure (CCCP). In *Advances in Neural Information Processing Systems*, 2001.

Zhang, X., Fan, P., Hao, L., and Quan, X. Generalized approximate message passing based Bayesian learning detectors for uplink grant-free NOMA. *IEEE Transactions on Vehicular Technology*, 72(11), 2023.

Zhao, L., Gao, W.-J., and Guo, W. Sparse Bayesian learning of delay-Doppler channel for OTFS system. *IEEE Communications Letters*, 24(12):2766–2769, 2020.

Zhou, W., Zhang, H.-T., and Wang, J. An efficient sparse Bayesian learning algorithm based on Gaussian-scale mixtures. *IEEE Transactions on Neural Networks and Learning Systems*, 33(7):3065–3078, 2021.

A. Proof of Theorem 3.1

Proof. As the Global Convergence Theorem gives technical conditions for convergence (Luenberger & Ye, 1984), we divide the proof into three parts to demonstrate that the proposed optimization method follows such conditions.

(i) **For the set of local minima (or saddle points) and $\mathcal{A}(\cdot)$, $\mathcal{L}(\boldsymbol{\gamma},\lambda)$ is a descent function**. Let Ω be the set of points $(\boldsymbol{\gamma},\lambda)$ satisfying $\nabla_{\boldsymbol{\gamma},\lambda}\mathcal{L}(\boldsymbol{\gamma},\lambda)=\mathbf{0}$. First, we show that the slope of $\mathcal{L}_{\boldsymbol{z}^k,\boldsymbol{\theta}^k}(\boldsymbol{\gamma},\lambda)$ is equal to that of $\mathcal{L}(\boldsymbol{\gamma},\lambda)$ at the point $(\boldsymbol{\gamma}^k,\lambda^k)$. Recalling $\boldsymbol{z}^k=\nabla_{\boldsymbol{\gamma}^k,\lambda^k}\mathcal{L}_1(\boldsymbol{\gamma},\lambda)$ and the expression of $\mathcal{L}_{\boldsymbol{z}^k,\boldsymbol{\theta}^k}(\boldsymbol{\gamma},\lambda)$ and $\mathcal{L}(\boldsymbol{\gamma},\lambda)$, we thus only need to prove

$$\nabla_{\boldsymbol{\gamma}^k,\lambda^k}\left(\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}\right)=\nabla_{\boldsymbol{\gamma}^k,\lambda^k}\left(\frac{1}{\lambda}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}^k\|_2^2+\left(\boldsymbol{\theta}^k\right)^T\boldsymbol{\Gamma}^{-1}\boldsymbol{\theta}^k\right). \tag{38}$$

For λ, we can derive

$$\frac{\partial\left(\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}\right)}{\partial\lambda}=\boldsymbol{y}^T\frac{\partial\boldsymbol{\Pi}^{-1}}{\partial\lambda}\boldsymbol{y}=-\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\frac{\partial\boldsymbol{\Pi}}{\partial\lambda}\boldsymbol{\Pi}^{-1}\boldsymbol{y}=-\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{\Pi}^{-1}\boldsymbol{y}. \tag{39}$$

Based on the matrix inversion lemma, we can rewrite $\boldsymbol{\Pi}^{-1}\boldsymbol{y}$ as

$$\boldsymbol{\Pi}^{-1}\boldsymbol{y}=\left(\lambda\boldsymbol{I}_n+\boldsymbol{\Phi}\boldsymbol{\Gamma}\boldsymbol{\Phi}^T\right)^{-1}\boldsymbol{y}=\left(\frac{1}{\lambda}\boldsymbol{I}_n-\frac{1}{\lambda}\boldsymbol{\Phi}\left(\lambda\boldsymbol{\Gamma}^{-1}+\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^T\right)\boldsymbol{y}. \tag{40}$$

Consequently, it follows that

$$\nabla_{\lambda^k}\left(\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}\right)=-\frac{\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}^k\|_2^2}{(\lambda^k)^2}=\nabla_{\lambda^k}\left(\frac{1}{\lambda}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}^k\|_2^2+\left(\boldsymbol{\theta}^k\right)^T\left(\boldsymbol{\Gamma}^k\right)^{-1}\boldsymbol{\theta}^k\right). \tag{41}$$

For $\boldsymbol{\gamma}$, we can derive

$$\frac{\partial(\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y})}{\partial\gamma_i}=\boldsymbol{y}^T\frac{\partial\boldsymbol{\Pi}^{-1}}{\partial\gamma_i}\boldsymbol{y}=-\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\frac{\partial\boldsymbol{\Pi}}{\partial\gamma_i}\boldsymbol{\Pi}^{-1}\boldsymbol{y}=-\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{\Phi}_{[i]}\boldsymbol{\Phi}_{[i]}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}, \tag{42}$$

where $\boldsymbol{\Phi}_{[i]}$ is the ith column of $\boldsymbol{\Phi}$. Hence, we obtain

$$\frac{\partial(\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y})}{\partial\boldsymbol{\gamma}}=-\boldsymbol{\Phi}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}\odot\boldsymbol{\Phi}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}, \tag{43}$$

where $\odot$ is the Hadamard product. Additionally, we can rewrite $\boldsymbol{\Phi}^T$ as

$$\begin{aligned}\boldsymbol{\Phi}^T&=(\lambda\boldsymbol{I}_m+\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{\Gamma})^{-1}\left(\lambda\boldsymbol{I}_m+\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{\Gamma}\right)\boldsymbol{\Phi}^T\\&=\lambda^{-1}\boldsymbol{\Gamma}^{-1}\left(\lambda^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Phi}+\boldsymbol{\Gamma}^{-1}\right)^{-1}\boldsymbol{\Phi}^T\left(\lambda\boldsymbol{I}_n+\boldsymbol{\Phi}\boldsymbol{\Gamma}\boldsymbol{\Phi}^T\right)\\&=\lambda^{-1}\boldsymbol{\Gamma}^{-1}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Pi},\end{aligned} \tag{44}$$

which indicates $\boldsymbol{\Phi}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}=\lambda^{-1}\boldsymbol{\Gamma}^{-1}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Phi}^T\boldsymbol{y}$. Consequently, it follows that

$$\nabla_{\boldsymbol{\gamma}^k}\left(\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}\right)=-\left(\boldsymbol{\Gamma}^k\right)^{-1}\boldsymbol{\theta}^k\odot\left(\boldsymbol{\Gamma}^k\right)^{-1}\boldsymbol{\theta}^k=\nabla_{\boldsymbol{\gamma}^k}\left(\left(\lambda^k\right)^{-1}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}^k\|_2^2+\left(\boldsymbol{\theta}^k\right)^T\boldsymbol{\Gamma}^{-1}\boldsymbol{\theta}^k\right). \tag{45}$$

Based on the above derivation, we demonstrate that $\nabla_{\boldsymbol{\gamma}^k,\lambda^k}\mathcal{L}(\boldsymbol{\gamma},\lambda)=\nabla_{\boldsymbol{\gamma}^k,\lambda^k}\mathcal{L}_{\boldsymbol{z}^k,\boldsymbol{\theta}^k}(\boldsymbol{\gamma},\lambda)$. Moreover, we know that $\nabla_{\boldsymbol{\gamma}^k,\lambda^k}\mathcal{L}(\boldsymbol{\gamma},\lambda)$ is nonzero at the point $\left(\boldsymbol{\gamma}^k,\lambda^k\right)$ outside Ω. Hence, $\mathcal{L}_{\boldsymbol{z}^k,\boldsymbol{\theta}^k}(\boldsymbol{\gamma},\lambda)$ has a minimum elsewhere since it is convex in $(\boldsymbol{\gamma},\lambda)$. For the point $\left(\boldsymbol{\gamma}^k,\lambda^k\right)$ outside Ω, let

$$h_1(\boldsymbol{\gamma},\lambda)=\lambda^{-1}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{\theta}^k\|_2^2+\left(\boldsymbol{\theta}^k\right)^T\boldsymbol{\Gamma}^{-1}\boldsymbol{\theta}^k-\boldsymbol{y}^T\boldsymbol{\Pi}^{-1}\boldsymbol{y}, \tag{46}$$

$$h_2(\boldsymbol{\gamma},\lambda)=\left\langle\boldsymbol{z}^k,(\boldsymbol{\gamma},\lambda)\right\rangle-h^*(\boldsymbol{z}^k)-\mathcal{L}_1(\boldsymbol{\gamma},\lambda). \tag{47}$$

It is notable that $h_1(\boldsymbol{\gamma},\lambda)\geq 0$, $h_2(\boldsymbol{\gamma},\lambda)\geq 0$, and $h_2(\boldsymbol{\gamma},\lambda)$ is convex in $(\boldsymbol{\gamma},\lambda)$. Additionally, $h_1(\boldsymbol{\gamma},\lambda)\geq h_1(\boldsymbol{\gamma}^k,\lambda^k)=0$ and $\nabla_{\boldsymbol{\gamma}^k,\lambda^k}h_2(\boldsymbol{\gamma},\lambda)=0$. Consequently, it follows that

$$\mathcal{L}_{\boldsymbol{z}^k,\boldsymbol{\theta}^k}(\boldsymbol{\gamma},\lambda)-\mathcal{L}(\boldsymbol{\gamma},\lambda)=h_1(\boldsymbol{\gamma},\lambda)+h_2(\boldsymbol{\gamma},\lambda) \tag{48}$$

is nonnegative and achieves the global minimum at the point $\left(\boldsymbol{\gamma}^k, \lambda^k\right)$. However, $\mathcal{L}_{\boldsymbol{z}^k,\boldsymbol{\theta}^k}(\boldsymbol{\gamma}, \lambda)$ reaches the global minimum at the point $\left(\boldsymbol{\gamma}^{k+1}, \lambda^{k+1}\right)$ and the distance between $\mathcal{L}_{\boldsymbol{z}^k,\boldsymbol{\theta}^k}(\boldsymbol{\gamma}, \lambda)$ and $\mathcal{L}(\boldsymbol{\gamma}, \lambda)$ will be increased still further at such a point. Hence, we can derive $\mathcal{L}(\boldsymbol{\gamma}^{k+1}, \lambda^{k+1}) < \mathcal{L}(\boldsymbol{\gamma}^k, \lambda^k)$ for the point $\left(\boldsymbol{\gamma}^k, \lambda^k\right)$ outside Ω. Additionally, for the point $\left(\boldsymbol{\gamma}^k, \lambda^k\right)$ in Ω, we have $\nabla_{\boldsymbol{\gamma}^k,\lambda^k}\mathcal{L}_{\boldsymbol{z}^k,\boldsymbol{\theta}^k}(\boldsymbol{\gamma}, \lambda) = \mathbf{0}$. As such, $\left(\boldsymbol{\gamma}^{k+1}, \lambda^{k+1}\right)$ generated via $\mathcal{A}\left(\boldsymbol{\gamma}^k, \lambda^k\right)$ is equal to $\left(\boldsymbol{\gamma}^k, \lambda^k\right)$, implying $\mathcal{L}(\boldsymbol{\gamma}^{k+1}, \lambda^{k+1}) = \mathcal{L}(\boldsymbol{\gamma}^k, \lambda^k)$. This completes the proof.

(ii) **The sequence $\{(\boldsymbol{\gamma}^k, \lambda^k)\}$ is contained in a compact set**. As

$$\log|\boldsymbol{\Pi}| = \log\left(\lambda\left|\boldsymbol{I}_n + \lambda^{-1}\boldsymbol{\Phi}\boldsymbol{\Gamma}\boldsymbol{\Phi}^T\right|\right) = \log\lambda + \log\left|\boldsymbol{I}_n + \lambda^{-1}\boldsymbol{\Phi}\boldsymbol{\Gamma}\boldsymbol{\Phi}^T\right| \geq \log\lambda, \tag{49}$$

we obtain

$$\mathcal{L}(\boldsymbol{\gamma}, \lambda) \geq \sum_{i=1}^{m}(2a_0 + 2)\log\gamma_i + (2a_0 + 3)\log\lambda. \tag{50}$$

Consequently, if any element of $(\boldsymbol{\gamma}, \lambda)$ is unbounded, $\mathcal{L}(\boldsymbol{\gamma}, \lambda)$ diverges to infinity. Additionally, we know that $\mathcal{L}(\boldsymbol{\gamma}^{k+1}, \lambda^{k+1}) \leq \mathcal{L}(\boldsymbol{\gamma}^k, \lambda^k)$. This implies that there must exists a compact $\mathbb{S} \triangleq \{\boldsymbol{\eta} \in \mathbb{R}_+^{n+1} \mid \|\boldsymbol{\eta}\|_2 \leq r\}$ such that the sequence $\{(\boldsymbol{\gamma}^k, \lambda^k)\}$ is contained in it, where r is the radius.

(iii) **The mapping $\mathcal{A}(\cdot)$ is closed at points outside Ω**. First, the proposed algorithm can be regarded as a continuous function as it presents analytical mathematical expressions to update $(\boldsymbol{\gamma}^{k+1}, \lambda^{k+1})$ from $(\boldsymbol{\gamma}^k, \lambda^k)$ and all elementary functions are continuous. Additionally, $\mathcal{A}(\cdot)$ is a point-to-point mapping. Consequently, the mapping $\mathcal{A}(\cdot)$ is closed at points outside Ω.

Based on the above analyses, the Global Convergence Theorem states that the sequence $\{(\boldsymbol{\gamma}^k, \lambda^k)\}$ converges to a local minimum (or saddle point) of $\mathcal{L}(\boldsymbol{\gamma}, \lambda)$. □

B. Comparison with the Approximate Message Passing Algorithm

Because approximate message passing (AMP) algorithms offer an alternative Bayesian approach to solve sparse recovery problems, we further compare the proposed SBL algorithm with AMP to showcase its performance. The code for implementing AMP is available online at https://sourceforge.net/projects/gampmatlab/. Similarly, we generate an $n \times m$ random matrix $\boldsymbol{\Phi}$ and an m-dimensional sparse signal $\boldsymbol{w}$ with a sparsity rate r. Particularly, we consider that the nonzero coefficients in $\boldsymbol{w}$ are drawn from the standardized Gaussian distribution. Finally, we add the Gaussian noise with mean zero and different variances on $\boldsymbol{\Phi}\boldsymbol{w}$ to get a resultant signal $\boldsymbol{y}$ with a given SNR. In the experiments, we set $n = 64$, $m = 256$, and $r = 0.02$, and conduct simulation trials with SNR values ranging from 0 to 40 dB in steps of 5 dB. In all cases, we run 100 independent trials to test the performance of the proposed SBL algorithm and AMP. Additionally, a successful trial is recorded if the indices of nonzero elements in the estimated vector $\boldsymbol{w}$ are the same as true indices.

Table 4 records the probabilities of the proposed SBL algorithm and AMP in recovering sparse signals across different SNR levels. Because AMP is more reliable under large i.i.d. Gaussian matrices (Bayati & Montanari, 2011; Rangan, 2011), we first consider the random matrix $\boldsymbol{\Phi}$ to be a Gaussian-distributed matrix with each component drawn from the Gaussian distribution with mean zero and variance $\frac{1}{m}$. Table 4 indicates that the proposed SBL algorithm slightly outperforms AMP on the Gaussian-distributed matrix under high SNR conditions. Further, we consider the random matrix $\boldsymbol{\Phi}$ to be a low-rank matrix generated using the truncated SVD to test the robustness of the proposed SBL algorithm and AMP. As observed from Table 4, the experimental results on the low-rank matrix demonstrate that the proposed SBL algorithm is significantly superior to AMP.

Table 4. Probability of successfully recovering sparse signals at different SNR levels.

Dictionary matrix	Method	SNR								
		0 dB	5 dB	10 dB	15 dB	20 dB	25 dB	30 dB	35 dB	40dB
Gaussian-distributed matrix	**Ours**	**3%**	5	**8%**	11%	**9%**	**14%**	**20%**	**13%**	**14%**
	AMP	1%	**6%**	**8%**	**12%**	**9%**	6%	12%	6%	11%
Low-rank matrix	**Ours**	**26%**	**40%**	**45%**	**51%**	**63%**	**50%**	**54%**	**59%**	**48%**
	AMP	1%	0	1%	0	1%	1%	4%	2%	2%

C. Additional Experimental Result on Sparse Kernel Regression

In Section 5.3, we set the hyperparameters in kernels to 1 by default. Here, we further test all the SBL algorithms on the Gaussian kernel function $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\alpha\|\boldsymbol{x} - \boldsymbol{x}'\|^2)$ with the different values of α ranging from 0.05 to 50. Table 5 records the experimental results (i.e., d), which demonstrate that the proposed SBL algorithm outperforms the classical ones when $\alpha > 0.2$. Here, please note that each α corresponds to a different sparse regression problem $\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{w} + \boldsymbol{\varepsilon}$, as $\boldsymbol{\Phi}$ is closely related to α. Hence, we cannot compare the performance of all the SBL algorithms across the different values of α, but should compare their performance on the same value of α. As such, the experimental results indicate that the proposed SBL algorithm outperforms the classical ones over a wide range (i.e., $\alpha > 0.2$).

Table 5. Regression result of all the SBL algorithms across the different values of α.

Method	Ours	MacKay_SBL	EM_SBL	IR_SBL	VI_SBL
$\alpha = 0.05$	0.0964	**0.0938**	1.0044	0.1026	0.1257
$\alpha = 0.1$	0.0976	**0.0943**	1.0043	0.9997	0.1295
$\alpha = 0.2$	0.0950	**0.0944**	1.0043	0.1009	0.1270
$\alpha = 0.4$	**0.0958**	0.0974	1.0044	0.1003	0.1270
$\alpha = 0.6$	**0.0959**	0.1050	1.0044	0.1006	0.1270
$\alpha = 0.8$	**0.0967**	0.1083	1.0045	0.1004	0.1270
$\alpha = 1$	**0.0980**	0.1123	1.0045	0.1004	0.1270
$\alpha = 2$	**0.0981**	0.1428	1.0048	0.1053	0.1270
$\alpha = 3$	**0.1008**	0.1877	1.0052	0.1147	0.1270
$\alpha = 4$	**0.1027**	0.2243	1.0054	0.1279	0.1270
$\alpha = 5$	**0.1071**	0.2601	1.0056	0.1468	0.1270
$\alpha = 10$	**0.1185**	0.4608	1.0064	0.2323	0.1270
$\alpha = 20$	**0.1255**	0.7144	0.9481	0.2531	0.1270
$\alpha = 30$	**0.1255**	0.7405	0.8938	0.2689	0.1270
$\alpha = 40$	**0.1256**	0.7023	0.7830	0.2778	0.1270
$\alpha = 50$	**0.1254**	0.6866	0.7340	0.2782	0.1270