# Projected Newton method for large-scale Bayesian linear inverse problems

Haibo Li *

**Abstract**

Computing the regularized solution of Bayesian linear inverse problems as well as the corresponding regularization parameter is highly desirable in many applications. This paper proposes a novel iterative method, termed the *Projected Newton method* (PNT), that can simultaneously update the regularization parameter and solution step by step without requiring any high-cost matrix inversions or decompositions. By reformulating the Tikhonov regularization as a constrained minimization problem and writing its Lagrangian function, a Newton-type method coupled with a Krylov subspace method, called the generalized Golub-Kahan bidiagonalization, is employed for the unconstrained Lagrangian function. The resulting PNT algorithm only needs solving a small-scale linear system to get a descent direction of a merit function at each iteration, thus significantly reducing computational overhead. Rigorous convergence results are proved, showing that PNT always converges to the unique regularized solution and the corresponding Lagrangian multiplier. Experimental results on both small and large-scale Bayesian inverse problems demonstrate its excellent convergence property, robustness and efficiency. Given that the most demanding computational tasks in PNT are primarily matrix-vector products, it is particularly well-suited for large-scale problems.

**Keywords** Bayesian inverse problem, Tikhonov regularization, constrained optimization, Newton method, generalized Golub-Kahan bidiagonalization, projected Newton direction

## Contents



*School of Mathematics and Statistics, The University of Melbourne, Parkville, VIC 3010, Australia. haibo.li@unimelb.edu.au

# 1 Introduction

Inverse problems arise in various scientific and engineering fields, where the aim is to recover unknown parameters or functions from noisy observed data. Applications include image reconstruction, computed tomography, medical imaging, geoscience, data assimilation and so on [29, 25, 6, 31, 49]. A linear inverse problem of the discrete form can be written as

$$\boldsymbol{b} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{\epsilon}, \tag{1.1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the underlying quantity to reconstruct, $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ is the discretized forward model matrix, $\boldsymbol{b} \in \mathbb{R}^m$ is the vector of observation with noise $\boldsymbol{\epsilon}$. We assume that the distribution of $\boldsymbol{\epsilon}$ is known, which follows a zero mean Gaussian distribution with positive definite covariance matrix $\boldsymbol{M}$, i.e., $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{M})$. A big challenge for reconstructing a good solution is the ill-posedness of inverse problems, which means that there may be multiple solutions that fit the observation equally well, or the solution is very sensitive with respect to observation perturbation.

To overcome the ill-posedness, regularization is a commonly used technique. From a Bayesian perspective [29, 51], this corresponds to adding a prior distribution of the desired solution to constrain the set of possible solutions to improve stability and uniqueness. By treating $\boldsymbol{x}$ and $\boldsymbol{b}$ as random variables, the observation vector $\boldsymbol{b}$ has a conditional probability density function (pdf)

$$p(\boldsymbol{b}|\boldsymbol{x}) \propto \exp\left(-\frac{1}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}}\right).$$

In order to get a regularized solution, we assume a Gaussian prior about the desired solution with the form $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \alpha^{-1}\boldsymbol{N})$, where $\boldsymbol{N}$ is a positive definite covariance matrix. Then the Bayes' formula leads to

$$p(\boldsymbol{x}|\boldsymbol{b}, \lambda) \propto p(\boldsymbol{x}|\lambda)p(\boldsymbol{b}|\boldsymbol{x}) \propto \exp\left(-\frac{1}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} - \frac{\mu}{2}\|\boldsymbol{x}\|^2_{\boldsymbol{N}^{-1}}\right),$$

where $\|\boldsymbol{x}\|_{\boldsymbol{B}} := (\boldsymbol{x}^\top \boldsymbol{B}\boldsymbol{x})^{1/2}$ is the $\boldsymbol{B}$-norm of $\boldsymbol{x}$ for a positive definite matrix $\boldsymbol{B}$. Maximize the posterior pdf $p(\boldsymbol{x}|\boldsymbol{b}, \lambda)$ leads to the Tikhonov regularization problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \{\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} + \mu\|\boldsymbol{x}\|^2_{\boldsymbol{N}^{-1}}\}, \tag{1.2}$$

where the regularization term $\|\boldsymbol{x}\|^2_{\boldsymbol{N}^{-1}}$ enforces extra structure on the solution that comes from the prior distribution of $\boldsymbol{x}$.

The parameter $\mu$ in the Gaussian prior $\mathcal{N}(\boldsymbol{0}, \mu^{-1}\boldsymbol{N})$ is crucial for obtaining a good regularized solution, which controls the trade-off between the data-fit term and regularization term. There is tremendous effort in determining a proper value of $\mu$. For the standard 2-norm problem, i.e. $\boldsymbol{M} = \boldsymbol{N} = \boldsymbol{I}$, the classical parameter-selection methods include the L-curve (LC) criterion[23], generalized cross-validation (GCV) [22], unbiased predictive risk estimation (UPRE) [47] and discrepancy principle (DP) [40]. There are also some iterative methods based on solving a nonlinear equation of $\mu$; see e.g. [38, 2, 46, 18, 20]. However, the aforementioned methods can not be directly applied to (1.2). A common procedure needs to first transform (1.2) into the standard 2-norm form

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \{\|\boldsymbol{L}_M(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})\|^2_2 + \mu\|\boldsymbol{L}_N\boldsymbol{x}\|^2_2\}, \tag{1.3}$$

where $\boldsymbol{M}^{-1} = \boldsymbol{L}_M^\top \boldsymbol{L}_M$ and $\boldsymbol{N}^{-1} = \boldsymbol{L}_N^\top \boldsymbol{L}_N$ are the Cholesky factorizations, and then apply the parameter-selection methods. This procedure needs the matrix inversions of $\boldsymbol{M}$ and $\boldsymbol{N}$

as well as the Cholesky factorizations of $\boldsymbol{M}^{-1}$ and $\boldsymbol{N}^{-1}$. For large-scale matrices, the above two types of computations are almost impossible or extremely expensive.

For large-scale problems, there exist some iterative regularization methods that can avoid choosing $\mu$ in advance. A class of commonly used iterative methods is based on Krylov subspaces [36], where the original linear system is projected onto lower-dimensional subspaces to become a series of small-scale problems [41, 44, 19, 27, 35]. For dealing with the general-form Tikhonov regularization term $\|\boldsymbol{L}_N\boldsymbol{x}\|_2^2$, some recent Krylov iterative methods include [39, 30, 45, 26, 33] and so on. When the Cholesky factor $L_{\boldsymbol{N}}$ is not accessible, a key difficulty is dealing with the prior covariance $\boldsymbol{N}$, which means that the subspaces should be constructed elaborately such that the prior information of $\boldsymbol{x}$ can be effectively incorporated into these subspaces [8, 34]. Such methods have been proposed in [9, 7, 8], where the statistically inspired priorconditioning technique is used to whiten the noise and the desired solution. However, these methods still require large-scale matrix inversions and Cholesky factorizations, which prohibits their applications to large-scale problems.

Recently, there are several Krylov methods for solving (1.1) without choosing $\mu$ in advance and can avoid the matrix inversions and Cholesky factorizations [12, 34]. These methods use the generalized Golub-Kahan bidiagonalization (gen-GKB), which can iteratively reduce the original large-scale problem to small-scale ones and generate Krylov subspaces that effectively incorporate the prior information of $\boldsymbol{x}$. In [34], the regularization effect of the proposed method comes from early stopping the iteration, where the iteration number plays the role of the regularization parameter, while in [12], the authors proposed a hybrid regularization method that simultaneously computes the regularized parameter and solution step by step. Although these two methods are very efficient for large-scale problems, there may be some issues in certain situations. The method in [34] only computes a good regularized solution but not a good $\mu$. However, in some applications, we need an accurate estimate of $\mu$ to get the posterior distribution of $\boldsymbol{x}$ for sampling and uncertainty quantification [51, 50, 16]. For the hybrid method in [12], the convergence property does not have a solid theoretical foundation, and it has been numerically found that the method sometimes does not converge to a good solution, which is a common potential flaw for hybrid methods [11, 48].

Many optimization methods have been proposed for solving inverse problems, especially those arising from image processing that leads to total variation regularization and $\ell_p$ regularization. These methods include the Bregman iteration [42, 54, 21], iterative shrinkage thresholding [15, 3], and some others [1, 52, 37]. However, these methods either need a good parameter $\mu$ in advance or can not well deal with $\boldsymbol{M}^{-1}$ and $\boldsymbol{N}^{-1}$. In [32] the author proposed a modification of the Newton method that can iteratively compute a good $\mu$ and regularized solution simultaneously. However, this method needs to solve a large-scale linear system at each iteration, which is very costly for large-scale problems. This method was improved in [13, 14], where the Newton method is successfully combined with a Krylov subspace method to get a so-called projected Newton method. Compared with the original method, the projected Newton method only needs to solve a small-scale linear system at each iteration, thereby very efficient for large-scale Tikhonov regularization (1.3). However, for solving (1.2), this method needs to compute $\nabla(\frac{1}{2}\|x\|_{\boldsymbol{N}^{-1}}^2) = \boldsymbol{N}^{-1}x$ to construct subspaces, which is also very costly. Besides, their methods lack rigorous proof of convergence.

In this paper, we develop a new and efficient iterative method for (1.2) that simultaneously updates the regularization parameter and solution step by step, where the matrix inversions and Cholesky factorizations are not required any longer. This method is based on a Newton-type method for a constrained minimization problem, and the gen-GKB process is used to compute a projected Newton direction by solving a small-scale linear system at each iteration, thereby we also name it the *projected Newton method* (PNT). The main contributions of this paper are listed as follows:

- Based on the discrepancy principle, the regularization of the original Bayesian linear inverse problems is reformulated as a noise constrained minimization problem. We

show the correspondence between the constrained minimization problem and Tikhonov regularization (1.2), where the Lagrangian multiplier $\lambda$ is just the reciprocal of $\mu$.

- For the constrained minimization problem, we compute the solution by optimizing its Lagrangian function while getting the corresponding Lagrangian multiplier, which is now an unconstrained optimization with a nonlinear nonconvex function. A Newton-type method for this problem is combined with the gen-GKB process to get the new projected Newton method, where only a small-scale linear system needs to be solved to compute a descent direction at each iteration.
- A rigorous convergence proof of the proposed method is provided. With a very practical initialization $(\boldsymbol{x}_0, \lambda_0)$, we prove that PNT always converges to the unique solution of the constrained minimization problem and its corresponding Lagrangian multiplier.

We use both small-scale and large-scale inverse problems to test the proposed method and compare it with other state-of-the-art methods. The experimental results demonstrate excellent convergence properties of PNT, and it is very robust and efficient for large-scale Bayesian linear inverse problems. Since the most computationally intensive operations in PNT primarily involve matrix-vector products, it is especially appropriate for large-scale problems.

This paper is organized as follows. In Section 2, we formulate the noise constrained minimization problem for regularizing (1.1) and study its properties. In Section 3, we propose the new projected Newton method. In Section 4, we prove the convergence of the propsoed method. Numerical results are presented in Section 5, and conclusions are provided in Section 6.

Throughout the paper, we denote by $\boldsymbol{I}$ and $\boldsymbol{0}$ the identity matrix and zero matrix/vector, respectively, with orders clear from the context, and denote by $\text{span}\{\cdot\}$ the subspace spanned by a group of vectors or columns of a matrix.

# 2 Noise constrained minimization for Bayesian inverse problems

In order to get a good estimate of $\mu$ in (1.2), the discrepancy principle (DP) criterion is commonly used, which depends on the variance of the noise. Based on DP, we can rewrite (1.2) as an equivalent form of noise constrained minimization problem. The Lagrangian of this problem can be solved by the Newton method.

## 2.1 Noise constrained minimization

For the case that $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$ is a white Gaussian noise, the DP criterion states that the 2-norm discrepancy between the data and predicted output $\|\boldsymbol{A}\boldsymbol{x}(\mu) - \boldsymbol{b}\|_2$ should be of the order of $\|\boldsymbol{\epsilon}\|_2$, where $\boldsymbol{x}(\mu)$ is the solution to (1.2); note that $\|\boldsymbol{\epsilon}\|_2 \approx \sqrt{m}\sigma$; see [29, §5.6]. If $\boldsymbol{\epsilon}$ is a general Gaussian noise, notice that (1.1) leads to

$$\boldsymbol{L}_M \boldsymbol{b} = \boldsymbol{L}_M \boldsymbol{A}\boldsymbol{x} + \boldsymbol{L}_M \boldsymbol{\epsilon}, \tag{2.1}$$

and $\boldsymbol{L}_M \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, thereby this transformation whitens the noise. Since $\bar{\boldsymbol{\epsilon}} := \boldsymbol{L}_M \boldsymbol{\epsilon}$ is a white Gaussian noise with zero mean and covariance $\boldsymbol{I}$, it follows that

$$\mathbb{E}\left[\|\bar{\boldsymbol{\epsilon}}\|_2^2\right] = \mathbb{E}\left[\text{trace}\left(\bar{\boldsymbol{\epsilon}}^\top \bar{\boldsymbol{\epsilon}}\right)\right] = \mathbb{E}\left[\text{trace}\left(\bar{\boldsymbol{\epsilon}}\bar{\boldsymbol{\epsilon}}^\top\right)\right] = \text{trace}\left(\mathbb{E}\left[\bar{\boldsymbol{\epsilon}}\bar{\boldsymbol{\epsilon}}^\top\right]\right) = \text{trace}\left(\boldsymbol{I}\right) = m.$$

Therefore, the DP for (2.1) can be written as

$$\|\boldsymbol{A}\boldsymbol{x}(\mu) - \boldsymbol{b}\|_{\boldsymbol{M}^{-1}}^2 = \|\boldsymbol{L}_M \boldsymbol{A}\boldsymbol{x}(\mu) - \boldsymbol{L}_M \boldsymbol{b}\|_2^2 = \tau m \tag{2.2}$$

where $\tau$ is chosen to be marginally greater than 1, such as $\tau = 1.01$.

Using this expression of DP, we rewrite the regularization of (1.1) as the noise constrained minimization problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{x}\|^2_{\boldsymbol{N}^{-1}} \quad \text{s.t.} \quad \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} \le \frac{\tau m}{2} \tag{2.3}$$

and its Lagrangian

$$\mathcal{L}(\boldsymbol{x},\lambda) = \frac{1}{2}\|\boldsymbol{x}\|^2_{\boldsymbol{N}^{-1}} + \frac{\lambda}{2}\left(\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} - \tau m\right), \tag{2.4}$$

where $\lambda \ge 0$ is the Lagrangian multiplier. Note that $\lambda$ plays the role of $\mu^{-1}$ in (1.2), meaning that the solution to (1.2) with $\mu = \lambda^{-1}$ is just the solution to (2.4). In fact, there is a one-to-one correspondence between (1.2) and (2.3). We first state the following basic assumption, which is used throughout the paper.

**Assumption 2.1** *For all* $\boldsymbol{x} \in \{\boldsymbol{x}\in\mathbb{R}^n : \|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_{\boldsymbol{M}^{-1}} = \min\}$*, it holds*

$$\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} < \tau m < \|\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}}. \tag{2.5}$$

The first inequality means that the naive solutions to (1.1) fit the observation very well, and it ensures the feasible set of (2.3) is nonempty. The second inequality comes from the condition $\|L_{\boldsymbol{M}}\boldsymbol{\epsilon}\|_2 < \|L_{\boldsymbol{M}}\boldsymbol{b}\|_2$ for (2.1), meaning that the noise does not dominate the observation, which ensures the fruitfulness of the regularization. Under this assumption, the following result describes the solution to (2.3).

**Theorem 2.1** *The noise constrained minimization* (2.3) *has a unique solution* $\boldsymbol{x}^*$ *satisfying* $\|\boldsymbol{A}\boldsymbol{x}^*-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} = \tau m$*. Furthermore, there is a unique* $\lambda^* > 0$*, which is the Lagrangian multiplier corresponding to* $\boldsymbol{x}^*$ *in* (2.4)*.*

**Proof.** Let $\varphi(\boldsymbol{x}) := \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}}$, which is a convex function. In (2.3) we seek solutions to $\min \frac{1}{2}\|\boldsymbol{x}\|^2_{\boldsymbol{N}^{-1}}$ in the feasible set $S := \{\boldsymbol{x}\in\mathbb{R}^n : \varphi(\boldsymbol{x}) \le \frac{\tau m}{2}\}$, which is the $\frac{\tau m}{2}$-lower level set of $\varphi(\boldsymbol{x})$. Note that $S$ is a compact and convex set, and $\frac{1}{2}\|\boldsymbol{x}\|^2_{\boldsymbol{N}^{-1}}$ is continous and strictly convex since $\boldsymbol{N}^{-1}$ is positive definite. Thus, there is a unique solution $\boldsymbol{x}^*$ to (2.3).

Suppose $\lambda^*$ is a Lagrangian multiplier corresponding to $\boldsymbol{x}^*$. By the Karush-Kuhn-Tucker (KKT) condition [28, §12.3], the solution $(\boldsymbol{x}^*, \lambda^*)$ satisfies

$$\begin{cases} \boldsymbol{N}^{-1}\boldsymbol{x}^* + \lambda^*\nabla\varphi(\boldsymbol{x}^*) = \boldsymbol{0}, \\ \lambda^*\varphi(\boldsymbol{x}^*) = 0, \\ \lambda^* \ge 0. \end{cases}$$

If $\lambda^* = 0$, then $\boldsymbol{N}^{-1}\boldsymbol{x}^*$, leading to $\boldsymbol{x}^* = \boldsymbol{0}$. This means $\boldsymbol{0} \in S$, i.e. $\varphi(\boldsymbol{0}) \le \frac{\tau m}{2}$, thereby $\|\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} \le \tau m$, a contradiction. Consequently, it must hold $\lambda^* > 0$. From the relation $\lambda^*\varphi(\boldsymbol{x}^*) = 0$ we have $\varphi(\boldsymbol{x}^*) = 0$, i.e. $\|\boldsymbol{A}\boldsymbol{x}^*-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} = \tau m$.

To prove the uniqueness of $\lambda^*$, first note that for any $\lambda \ge 0$, there is a unique $\boldsymbol{x}_\lambda$ that solves the first equality of the KKT condition:

$$\boldsymbol{N}^{-1}\boldsymbol{x} + \lambda\nabla\varphi(\boldsymbol{x}) = \boldsymbol{0} \quad \Leftrightarrow \quad (\boldsymbol{N}^{-1} + \lambda\boldsymbol{A}^\top\boldsymbol{M}^{-1}\boldsymbol{A})x = \lambda\boldsymbol{A}^\top\boldsymbol{M}^{-1}\boldsymbol{b}, \tag{2.6}$$

since $\boldsymbol{N}^{-1} + \lambda\boldsymbol{A}^\top\boldsymbol{M}^{-1}\boldsymbol{A}$ is positive definite. Here we prove a stronger property: *there exist a unique* $\lambda \ge 0$ *such that* $\|\boldsymbol{A}\boldsymbol{x}_\lambda-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} = \tau m$. The existence of such a $\lambda$ has been proved, since $\boldsymbol{x}^* = \boldsymbol{x}_{\lambda^*}$. For the uniqueness, define two functions

$$K(\lambda) := \frac{1}{2}\|\boldsymbol{x}_\lambda\|^2_{\boldsymbol{N}^{-1}}, \quad H(\lambda) := \frac{1}{2}\left(\|\boldsymbol{A}\boldsymbol{x}_\lambda-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} - \tau m\right).$$

Note that $\mathcal{L}(\boldsymbol{x},\lambda)$ is strictly convex for a fixed $\lambda>0$, which has the unique minimizer $\boldsymbol{x}_\lambda$. Thus, for any two positive $\lambda_1\neq\lambda_2$, we have

$$\mathcal{L}(\boldsymbol{x}_{\lambda_1},\lambda_1)<\mathcal{L}(\boldsymbol{x}_{\lambda_2},\lambda_1)\quad\Leftrightarrow\quad K(\lambda_1)+\lambda_1 H(\lambda_1)<K(\lambda_2)+\lambda_1 H(\lambda_2),$$

since $\boldsymbol{x}_{\lambda_1}\neq\boldsymbol{x}_{\lambda_2}$; see the following Lemma 2.1. Similarly, we have

$$K(\lambda_2)+\lambda_2 H(\lambda_2)<K(\lambda_1)+\lambda_2 H(\lambda_1).$$

Adding the above two inequalities leads to

$$(\lambda_1-\lambda_2)(H(\lambda_1)-H(\lambda_2))<0,$$

meaning that $H(\lambda)$ is a strictly monotonic decreasing function. Therefore, there is a unique $\lambda$ such that $H(\lambda)=0$. □

**Lemma 2.1** *For each $\lambda\geq 0$, the regularization problem*

$$\min_{\boldsymbol{x}\in\mathbb{R}^n}\{\lambda\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}}+\|\boldsymbol{x}\|^2_{\boldsymbol{N}^{-1}}\}\tag{2.7}$$

*has the unique solution $\boldsymbol{x}_\lambda$. If $\lambda_1\neq\lambda_2$, then $\boldsymbol{x}_{\lambda_1}\neq\boldsymbol{x}_{\lambda_2}$.*

**Proof.** Note that the normal equation of (2.7) is equivalent to (2.6). Thus, $\boldsymbol{x}_\lambda$ is the unique solution to (2.7). Using the Cholesky factors of $\boldsymbol{M}^{-1}$ and $\boldsymbol{N}^{-1}$, and noticing that $\boldsymbol{L}_N$ is invertible, we can write the generalized singular value decomposition (GSVD) [53] of $\{\boldsymbol{L}_M\boldsymbol{A},\boldsymbol{L}_N\}$ as

$$\boldsymbol{L}_M\boldsymbol{A}=\boldsymbol{U}_A\boldsymbol{\Sigma}_A\boldsymbol{Z}^{-1},\quad \boldsymbol{L}_N=\boldsymbol{U}_N\boldsymbol{\Sigma}_N\boldsymbol{Z}^{-1},$$

with

$$\boldsymbol{\Sigma}_A=\begin{pmatrix}\boldsymbol{D}_A & \\ & \boldsymbol{0}\end{pmatrix}\begin{matrix}r\\ m-r\end{matrix},\quad \boldsymbol{\Sigma}_N=\begin{pmatrix}\boldsymbol{D}_N & \\ & \boldsymbol{I}\end{pmatrix}\begin{matrix}r\\ n-r\end{matrix},$$
$$\begin{matrix}r & n-r\end{matrix}\qquad\qquad\begin{matrix}r & n-r\end{matrix}$$

where $r=\operatorname{rank}(\boldsymbol{A})$, the two matrices $\boldsymbol{U}_A\in\mathbb{R}^{m\times m}$ and $\boldsymbol{U}_N\in\mathbb{R}^{n\times n}$ are orthogonal, and $\boldsymbol{D}_A=\operatorname{diag}(\sigma_1,\ldots,\sigma_r)$ with $1>\sigma_1\geq\cdots\geq\sigma_r>0$ and $\boldsymbol{D}_N=\operatorname{diag}(\rho_1,\ldots,\rho_r)$ with $0<\rho_1\leq\cdots\leq\rho_r<1$, such that $\sigma_i^2+\rho_i^2=1$. Then $\boldsymbol{x}_\lambda$ can be expressed as

$$\boldsymbol{x}_\lambda=[\lambda(\boldsymbol{L}_M\boldsymbol{A})^\top\boldsymbol{L}_M\boldsymbol{A}+\boldsymbol{L}_N^\top\boldsymbol{L}_N]^{-1}\lambda(\boldsymbol{L}_M\boldsymbol{A})^\top\boldsymbol{L}_M\boldsymbol{b}=\sum_{i=1}^{r}\frac{\lambda\sigma_i}{\lambda\sigma_i^2+\rho_i^2}(\boldsymbol{u}_{A,i}^\top\boldsymbol{L}_M\boldsymbol{b})\boldsymbol{z}_i,$$

where $\boldsymbol{u}_{A,i}$ is the $i$-th column of $\boldsymbol{U}_A$. Since $\{\boldsymbol{z}_i\}_{i=1}^r$ are linear independent, if $\boldsymbol{x}_{\lambda_1}=\boldsymbol{x}_{\lambda_2}$, then it must hold

$$\frac{\lambda_1\sigma_i}{\lambda_1\sigma_i^2+\rho_i^2}=\frac{\lambda_2\sigma_i}{\lambda_2\sigma_i^2+\rho_i^2}\quad\Leftrightarrow\quad(\lambda_1-\lambda_2)\sigma_i\rho_i^2=0,\quad i=1,\ldots,r.$$

Since $\sigma_i\rho_i>0$ for $i=1,\ldots,r$, we obtain $\lambda_1=\lambda_2$. □

Note that $\boldsymbol{x}^*=\boldsymbol{x}_{\lambda^*}$. Comparing (2.7) with (1.2), we can use $(\lambda^*)^{-1}$ as a good estimate of the optimal regularization parameter. From Theorem 2.1 and its proof, we have the following result.

**Corollary 2.1** *Let $\mathbb{R}^+=[0,\infty)$. Write the gradient of $\mathcal{L}(\boldsymbol{x},\lambda)$ as*

$$F(\boldsymbol{x},\lambda)=\begin{pmatrix}\lambda\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b})+\boldsymbol{N}^{-1}\boldsymbol{x}\\ \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}}-\frac{\tau m}{2}\end{pmatrix}.\tag{2.8}$$

*Then $F(\boldsymbol{x},\lambda)=\boldsymbol{0}$ has a unique solution $(\boldsymbol{x}^*,\lambda^*)$ in $\mathbb{R}^n\times\mathbb{R}^+$, which is the unique minimizer and corresponding Lagrangian multiplier of (2.3).*

The Newton method can be used to solve the nonlinear equation $F(\boldsymbol{x},\lambda)=\boldsymbol{0}$ to obtain simultaneously the solution pair $(\boldsymbol{x}^*,\lambda^*)$.

### 2.2 Newton method

A modification of the Newton method proposed in [32] can be used to compute the nonlinear equation $F(\boldsymbol{x}, \lambda) = \boldsymbol{0}$, which is referred to as the Lagrange method since it is based on the Lagrangian of (2.3). In this method, the Jacobian matrix of $F(\boldsymbol{x}, \lambda)$ is first computed as

$$J(\boldsymbol{x}, \lambda) = \begin{pmatrix} \lambda \boldsymbol{A}^\top \boldsymbol{M}^{-1} \boldsymbol{A} + \boldsymbol{N}^{-1} & \boldsymbol{A}^\top \boldsymbol{M}^{-1} (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) \\ (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})^\top \boldsymbol{M}^{-1} \boldsymbol{A} & 0 \end{pmatrix} \tag{2.9}$$

at the current iterate $(\boldsymbol{x}, \lambda)$, and then it computes the Newton direction $(\Delta \boldsymbol{x}^\top, \Delta\lambda)^\top$ by solving inexactly the linear system

$$J(\boldsymbol{x}, \lambda) \begin{pmatrix} \Delta \boldsymbol{x} \\ \Delta \lambda \end{pmatrix} = -F(\boldsymbol{x}, \lambda) \tag{2.10}$$

using the MINRES solver [43]. We remark that this method is essentially a Newton-Krylov method [5] for optimizing the nonlinear and nonconvex Lagrangian function (2.4). It was shown that the computed $(\Delta \boldsymbol{x}, \Delta\lambda)$ in the Lagrange method is a descent direction for the merit function $h_w : \mathbb{R}^n \times \mathbb{R}^+ \to \mathbb{R}^+$ defined as

$$h_w(\boldsymbol{x}, \lambda) = \frac{1}{2} \left( \|\nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}, \lambda)\|_2^2 + w |\nabla_\lambda \mathcal{L}(\boldsymbol{x}, \lambda)|^2 \right)$$

with a positive $w$, meaning that $\nabla h(\boldsymbol{x}, \lambda)^\top \begin{pmatrix} \Delta \boldsymbol{x} \\ \Delta \lambda \end{pmatrix} \leq 0$. By a backtracking line search strategy to determine a step length $\gamma > 0$, the iterate is updated as $(\boldsymbol{x}, \lambda) \leftarrow (\boldsymbol{x}, \lambda) + \gamma(\Delta \boldsymbol{x}, \Delta\lambda)$. It was shown that the iterate eventually converges to the global minimizer of $h(\boldsymbol{x}, \lambda)$. Note that $h(\boldsymbol{x}, \lambda)$ achieves its global mininum at the unique point $(\boldsymbol{x}^*, \lambda^*)$, which is the zero point of $F(\boldsymbol{x}, \lambda)$.

A big advantage of this method is that it can compute a good regularized solution and its regularization parameter simultaneously. However, for large-scale problems, we need to compute $\boldsymbol{M}^{-1}$ and $\boldsymbol{N}^{-1}$ to form $F(\boldsymbol{x}, \lambda)$ and $J(\boldsymbol{x}, \lambda)$, which is almost impossible. Moreover, at each iteration, an $(n+1) \times (n+1)$ linear system (2.10) needs to be solved, which is very computationally expensive even if we only compute a less accurate solution by an iterative algorithm.

In [13], the authors proposed a projected Newton method, where at each iteration, the large-scale linear system (2.10) is projected to be a small-scale linear system that can be solved cheaply. However, this method can only deal with the standard $\ell_2 - \ell_2$ regularization, which means we can only apply this method to (1.3) by the substitution $\bar{\boldsymbol{x}} = \boldsymbol{L}_N \boldsymbol{x}$, requiring the expensive Cholesky factorization of $\boldsymbol{N}^{-1}$. A generalization of this method [14] can deal with a general-form regularization term. However, for (2.3), it needs to compute $\nabla(\frac{1}{2}\|x\|_{\boldsymbol{N}^{-1}}^2) = \boldsymbol{N}^{-1}x$ to construct subspace for projecting (2.10), also very costly.

## 3 Projected Newton method based on generalized Golub-Kahan bidiagonalization

To reduce expensive computations of the Newton method for large-scale problems, we design a new projected Newton method to solve (2.3). This method uses the generalized Golub-Kahan bidiagonalization (gen-GKB) to construct Krylov subspaces to compute projected Newton directions by only solving small-scale problems, and it does not need any matrix factorizations and inversions.

## 3.1 Overview

Algorithm 1 outlines the basic framework of the new projected Newton method (PNT), which is composed by the following three main steps:

Step 1: *Construct Kylov subspaces.* We adopt the gen-GKB process to iteratively construct a series of low-dimensional Krylov subspaces. The procedure is presented in Algorithm 2.

Step 2: *Compute the projected Newton direction.* At each iteration, based on gen-GKB we compute the projected Newton direction by only solving a small-scale problem; see (3.14).

Step 3: *Determine the step-length to update solution.* With the projected Newton direction, we use the Armijo backtracking line search strategy to determine a step-length, and then update the solution.

**Algorithm 1** PNT: Projected Newton method

**Input:** $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{b} \in \mathbb{R}^m$, $\boldsymbol{M} \in \mathbb{R}^{m\times m}$, $\boldsymbol{N} \in \mathbb{R}^{n\times n}$, $\tau \gtrsim 1$

1: Initialize $\boldsymbol{x}_0, \lambda_0$
2: **for** $k = 1, 2, \ldots,$ until convergence **do**
3: Step 1: Perform gen-GKB to construct $k$-dimensional Kylov subspace $\boldsymbol{V}_k$
4: Step 2: Compute the projected Newton direction $(\Delta\boldsymbol{y}_k, \Delta\lambda_k)$ as (3.14)
5: Step 3: Determine a step-length $\gamma_k$ by backtracking line search, and update the solution $(\boldsymbol{x}_k, \lambda_k) = (\boldsymbol{x}_{k-1}, \lambda_{k-1}) + \gamma_k(\boldsymbol{V}_k\Delta\boldsymbol{y}_k, \Delta\lambda_k)$
6: Check the convergence condition to stop iteration
7: **end for**

**Output:** Final solution for approximating $(\boldsymbol{x}^*, \lambda^*)$

In the next subsection, we present detailed derivations of the whole algorithm. All the proofs can be found in Section 3.3.

## 3.2 Derivation of projected Newton method

This subsection presents detailed derivations for the three steps in Algorithm 1.

**Step 1: Construct Kylov subspaces by gen-GKB.** The gen-GKB process has been proposed for solving Bayesian linear inverse problems in [12, 34]. The basic idea is to treat $\boldsymbol{A}$ as the compact linear operator under the canonical bases of $\mathbb{R}^n$ and $\mathbb{R}^m$

$$\boldsymbol{A} : (\mathbb{R}^n, \langle\cdot,\cdot\rangle_{\boldsymbol{N}^{-1}}) \to (\mathbb{R}^m, \langle\cdot,\cdot\rangle_{\boldsymbol{M}^{-1}}), \quad \boldsymbol{x} \mapsto \boldsymbol{A}\boldsymbol{x}$$

between the two finite dimensional Hilbert spaces $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_{\boldsymbol{N}^{-1}})$ and $(\mathbb{R}^m, \langle\cdot,\cdot\rangle_{\boldsymbol{M}^{-1}})$, where the inner products are defined as

$$\langle\boldsymbol{x}, \boldsymbol{x}'\rangle_{\boldsymbol{N}^{-1}} := \boldsymbol{x}^\top\boldsymbol{N}^{-1}\boldsymbol{x}', \quad \langle\boldsymbol{y}, \boldsymbol{y}'\rangle_{\boldsymbol{M}^{-1}} := \boldsymbol{y}^\top\boldsymbol{M}^{-1}\boldsymbol{y}'. \tag{3.1}$$

Then the adjoint of $\boldsymbol{A}$

$$\boldsymbol{A}^* : (\mathbb{R}^m, \langle\cdot,\cdot\rangle_{\boldsymbol{M}^{-1}}) \to (\mathbb{R}^n, \langle\cdot,\cdot\rangle_{\boldsymbol{N}^{-1}}), \quad \boldsymbol{y} \mapsto \boldsymbol{A}^*\boldsymbol{y}$$

defined by the relation $\langle\boldsymbol{A}\boldsymbol{x}, \boldsymbol{y}\rangle_{\boldsymbol{M}^{-1}} = \langle\boldsymbol{x}, \boldsymbol{A}^*\boldsymbol{y}\rangle_{\boldsymbol{N}^{-1}}$ has the matrix-form expression

$$\boldsymbol{A}^* = \boldsymbol{N}\boldsymbol{A}^\top\boldsymbol{M}^{-1}, \tag{3.2}$$

since $(\boldsymbol{A}\boldsymbol{x})^\top\boldsymbol{M}^{-1}\boldsymbol{y} = \boldsymbol{x}^\top\boldsymbol{N}^{-1}\boldsymbol{A}^*\boldsymbol{y}$ for any $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{y} \in \mathbb{R}^m$.

Now we can generate Krylov subspaces while reducing $\boldsymbol{A}$ to a bidiagonal form. This can be achieved by applying the standard GKB process to the compact operator $\boldsymbol{A}$ with starting

vector $\boldsymbol{b}$ between the two Hilbert spaces $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_{\boldsymbol{N}^{-1}})$ and $(\mathbb{R}^m, \langle\cdot,\cdot\rangle_{\boldsymbol{M}^{-1}})$; see [10]. The basic recursive relations are as follows:

$$\beta_1 \boldsymbol{u}_1 = \boldsymbol{b}, \tag{3.3a}$$
$$\alpha_i \boldsymbol{v}_i = \boldsymbol{A}^* \boldsymbol{u}_i - \beta_i \boldsymbol{v}_{i-1}, \tag{3.3b}$$
$$\beta_{i+1} \boldsymbol{u}_{i+1} = \boldsymbol{A}\boldsymbol{v}_i - \alpha_i \boldsymbol{u}_i, \tag{3.3c}$$

where $\alpha_i$ and $\beta_i$ are computed such that $\|\boldsymbol{u}_i\|_{\boldsymbol{M}^{-1}} = \|\boldsymbol{v}_i\|_{\boldsymbol{N}^{-1}} = 1$, and $\boldsymbol{v}_0 := \boldsymbol{0}$. In order to avoid explicitly computing $\boldsymbol{N}^{-1}$-norm to obtain $\alpha_i$, let $\bar{\boldsymbol{u}}_i = \boldsymbol{M}^{-1}\boldsymbol{u}_i$ and $\bar{\boldsymbol{v}}_i = \boldsymbol{N}^{-1}\boldsymbol{v}_i$. By (3.3b) and (3.2), we have $\alpha_i \bar{\boldsymbol{v}}_i = \boldsymbol{A}^\top \bar{\boldsymbol{u}}_i - \beta_i \bar{\boldsymbol{v}}_{i-1}$. Let $\bar{\boldsymbol{r}}_i = A^\top \bar{\boldsymbol{u}}_i - \beta_i \bar{\boldsymbol{v}}_{i-1}$ and $\boldsymbol{s}_i = \boldsymbol{A}\boldsymbol{v}_i - \alpha_i \boldsymbol{u}_i$. Then $\alpha_i = \|\boldsymbol{N}\bar{\boldsymbol{r}}_i\|_{\boldsymbol{N}^{-1}} = (\bar{\boldsymbol{r}}_i^\top \boldsymbol{N}\bar{\boldsymbol{r}}_i)^{1/2}$ and $\beta_{i+1} = \|\boldsymbol{s}_i\|_{\boldsymbol{M}^{-1}} = (\boldsymbol{s}_i^\top \boldsymbol{M}^{-1}\boldsymbol{s}_i)^{1/2}$. We remark that computing with $\boldsymbol{M}^{-1}$ can not be avoided to get $\boldsymbol{u}_i$, but for the most commonly encountered cases that $\boldsymbol{\epsilon}$ is a Gaussian noise with uncorrelated components, $\boldsymbol{M}$ is diagonal and thereby $\boldsymbol{M}^{-1}$ can be directly obtained. The whole iterative process is summarized in Algorithm 2.

**Algorithm 2** Generalized Golub-Kahan bidiagonalization (gen-GKB)

**Input:** $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{b} \in \mathbb{R}^m$, $\boldsymbol{M} \in \mathbb{R}^{m\times m}$, $\boldsymbol{N} \in \mathbb{R}^{n\times n}$

$\bar{\boldsymbol{s}} = \boldsymbol{M}^{-1}\boldsymbol{b}$, $\beta_1 = \bar{\boldsymbol{s}}^\top \boldsymbol{b}$, $\boldsymbol{u}_1 = \boldsymbol{b}/\beta_1$, $\bar{\boldsymbol{u}}_1 = \bar{\boldsymbol{s}}/\beta_1$

$\bar{\boldsymbol{r}} = \boldsymbol{A}^\top \bar{\boldsymbol{u}}_1$, $\boldsymbol{r} = \boldsymbol{N}\bar{\boldsymbol{r}}$

$\alpha_1 = (\boldsymbol{r}^\top \bar{\boldsymbol{r}})^{1/2}$, $\bar{\boldsymbol{v}}_1 = \bar{\boldsymbol{r}}/\alpha_1$, $\boldsymbol{v}_1 = \boldsymbol{r}/\alpha_1$

**for** $i = 1, 2, \ldots, k$ **do**

  $\boldsymbol{s} = \boldsymbol{A}\boldsymbol{v}_i - \alpha_i \boldsymbol{u}_i$, $\bar{\boldsymbol{s}} = \boldsymbol{M}^{-1}\boldsymbol{s}$

  $\beta_{i+1} = (\boldsymbol{s}^\top \bar{\boldsymbol{s}})^{1/2}$, $\boldsymbol{u}_{i+1} = \boldsymbol{s}/\beta_{i+1}$, $\bar{\boldsymbol{u}}_{i+1} = \bar{\boldsymbol{s}}/\beta_{i+1}$

  $\bar{\boldsymbol{r}} = \boldsymbol{A}^\top \bar{\boldsymbol{u}}_{i+1} - \beta_{i+1}\bar{\boldsymbol{v}}_i$, $\boldsymbol{r} = \boldsymbol{N}\bar{\boldsymbol{r}}$

  $\alpha_{i+1} = (\boldsymbol{r}^\top \bar{\boldsymbol{r}})^{1/2}$, $\bar{\boldsymbol{v}}_{i+1} = \bar{\boldsymbol{r}}/\alpha_{i+1}$, $\boldsymbol{v}_{i+1} = \boldsymbol{r}/\alpha_{i+1}$

**end for**

**Output:** $\{\alpha_i, \beta_i\}_{i=1}^{k+1}$, $\{\boldsymbol{u}_i, \boldsymbol{v}_i\}_{i=1}^{k+1}$

After $k$ steps, the gen-GKB generates two orthonormal bases $\{\boldsymbol{u}_i\}_{i=1}^{k+1}$ and $\{\boldsymbol{v}_i\}_{i=1}^{k+1}$ for $(\mathbb{R}^m, \langle\cdot,\cdot\rangle_{\boldsymbol{M}^{-1}})$ and $(\mathbb{R}^n, \langle\cdot,\cdot\rangle_{\boldsymbol{N}^{-1}})$, respectively. Define $\boldsymbol{U}_{k+1} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{k+1})$ and $\boldsymbol{V}_{k+1} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{k+1})$. Then $\boldsymbol{U}_{k+1}$ and $\boldsymbol{V}_{k+1}$ are two $\boldsymbol{M}^{-1}$ and $\boldsymbol{N}^{-1}$ orthonormal matrices, respectively, meaning $\boldsymbol{U}_{k+1}^\top \boldsymbol{M}^{-1}\boldsymbol{U}_{k+1} = \boldsymbol{I}$ and $\boldsymbol{V}_{k+1}^\top \boldsymbol{N}^{-1}\boldsymbol{V}_{k+1} = \boldsymbol{I}$. This property is presented in the following result; see [34] for the proof.

**Proposition 3.1** *The group of vectors $\{\boldsymbol{u}_i\}_{i=1}^k$ is an $\boldsymbol{M}^{-1}$-orthonormal basis of the Krylov subspace*

$$\mathcal{K}_k(\boldsymbol{A}\boldsymbol{N}\boldsymbol{A}^\top \boldsymbol{M}^{-1}, b) = \operatorname{span}\{(\boldsymbol{A}\boldsymbol{N}\boldsymbol{A}^\top \boldsymbol{M}^{-1})^i b\}_{i=0}^{k-1}, \tag{3.4}$$

*and $\{\boldsymbol{v}_i\}_{i=1}^k$ is an $\boldsymbol{N}^{-1}$-orthonormal basis of the Krylov subspace*

$$\mathcal{K}_k(\boldsymbol{N}\boldsymbol{A}^\top \boldsymbol{M}^{-1}\boldsymbol{A}, \boldsymbol{N}\boldsymbol{A}^\top \boldsymbol{M}^{-1}\boldsymbol{b}) = \operatorname{span}\{(\boldsymbol{N}\boldsymbol{A}^\top \boldsymbol{M}^{-1}\boldsymbol{A})^i \boldsymbol{N}\boldsymbol{A}^\top \boldsymbol{M}^{-1}\boldsymbol{b}\}_{i=0}^{k-1}. \tag{3.5}$$

We remark that the gen-GKB will eventually terminate at most $\min\{m, n\}$ steps, since the column rank of $\boldsymbol{U}_k$ or $\boldsymbol{V}_k$ can not exceed $\min\{m, n\}$. If we define the *termination step* as

$$k_t := \max\{k : \alpha_k \beta_k > 0\}, \tag{3.6}$$

which means that $k_t$ is the first iteration such that $\alpha_{k_t+1} = 0$ or $\beta_{k_t+1} = 0$, then $\boldsymbol{V}_k$ will eventually expand to be $\boldsymbol{V}_{k_t}$ with $k_t \le \min\{m, n\}$.

By (3.3a)–(3.3c), we can write the $k$-step ($k \le k_t$) gen-GKB in the matrix-form:

$$\beta_1 \boldsymbol{U}_{k+1} e_1 = \boldsymbol{b}, \tag{3.7a}$$
$$\boldsymbol{A}\boldsymbol{V}_k = \boldsymbol{U}_{k+1}\boldsymbol{B}_k, \tag{3.7b}$$
$$\boldsymbol{N}\boldsymbol{A}^\top \boldsymbol{M}^{-1}\boldsymbol{U}_{k+1} = \boldsymbol{V}_k \boldsymbol{B}_k^\top + \alpha_{k+1}\boldsymbol{v}_{k+1} e_{k+1}^\top, \tag{3.7c}$$

where $\boldsymbol{e}_1$ and $\boldsymbol{e}_{k+1}$ are the first and $(k+1)$-th columns of the identity matrix of order $k+1$, respectively, and

$$\boldsymbol{B}_k = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \beta_3 & \ddots & \\ & & \ddots & \alpha_k \\ & & & \beta_{k+1} \end{pmatrix} \in \mathbb{R}^{(k+1)\times k}. \tag{3.8}$$

If $k < k_t$, i.e. the gen-GKB does not terminate before the $k$-th iteration, then $\boldsymbol{B}_k$ has full column rank. At the $k_t$-th iteration, maybe $\beta_{k_t+1} = 0$ happens first or $\alpha_{k_t+1} = 0$ happens first. For the former case, the relations (3.7) are replaced by

$$\beta_1 \boldsymbol{U}_{k_t} e_1 = \boldsymbol{b}, \tag{3.9a}$$
$$\boldsymbol{A}\boldsymbol{V}_{k_t} = \boldsymbol{U}_{k_t} \underline{\boldsymbol{B}}_{k_t}, \tag{3.9b}$$
$$\boldsymbol{N}\boldsymbol{A}^\top \boldsymbol{M}^{-1} \boldsymbol{U}_{k_t} = \boldsymbol{V}_{k_t} \underline{\boldsymbol{B}}_{k_t}^\top, \tag{3.9c}$$

where $\underline{\boldsymbol{B}}_{k_t}$ is the first $k \times k$ part of $\boldsymbol{B}_{k_t}$ by discarding $\beta_{k_t+1}$.

**Step 2: Compute the projected Newton direction.** At the $k$-th iteration, we update $\boldsymbol{x}_k \in \text{span}\{\boldsymbol{V}_k\}$ and $\lambda_k$ from the previous one. For any $\boldsymbol{x} \in \text{span}\{\boldsymbol{V}_k\}$ of the form $\boldsymbol{x} = \boldsymbol{V}_k \boldsymbol{y}$ with $\boldsymbol{y} \in \mathbb{R}^k$, define the projected gradient of $\mathcal{L}(\boldsymbol{x}, \lambda)$ as

$$F^{(k)}(\boldsymbol{y}, \lambda) = \begin{pmatrix} \boldsymbol{V}_k^\top & \\ & 1 \end{pmatrix} F(\boldsymbol{x}, \lambda) \tag{3.10}$$

and the projected Jacobian of $F(\boldsymbol{x}, \lambda)$ as

$$J^{(k)}(\boldsymbol{y}, \lambda) = \begin{pmatrix} \boldsymbol{V}_k^\top & \\ & 1 \end{pmatrix} J(\boldsymbol{x}, \lambda) \begin{pmatrix} \boldsymbol{V}_k & \\ & 1 \end{pmatrix}. \tag{3.11}$$

**Remark 3.1** *Since gen-GKB must terminate at the $k_t$-th iteration and $\boldsymbol{V}_k$ eventually expands to be $\boldsymbol{V}_{k_t}$, we need to discuss the two different cases that $k \le k_t$ and $k > k_t$. For notational simplicity, in the rest part of the paper, we use $\boldsymbol{V}_k$ and $\boldsymbol{B}_k$ by default unless otherwise specified to denote*

$$\boldsymbol{V}_k = \begin{cases} \boldsymbol{V}_k, & k \le k_t \\ \boldsymbol{V}_{k_t}, & k > k_t \end{cases}, \quad \boldsymbol{B}_k = \begin{cases} \boldsymbol{B}_k, & k \le k_t \\ \boldsymbol{B}_{k_t}, & k > k_t \end{cases}, \quad \boldsymbol{x}_k = \begin{cases} \boldsymbol{V}_k \boldsymbol{y}_k, & k \le k_t \\ \boldsymbol{V}_{k_t} \boldsymbol{y}_k, & k > k_t \end{cases},$$

*where $\boldsymbol{y} \in \mathbb{R}^k$ for $k \le k_t$ and $\boldsymbol{y} \in \mathbb{R}^{k_t}$ for $k > k_t$. Moreover, for the case $\beta_{k_t+1} = 0$, the relations (3.7) are replaced by (3.9) and $\boldsymbol{B}_{k_t}$ is replaced by $\underline{\boldsymbol{B}}_{k_t}$. In the following discussions, we use the unified notations as in (3.7), but the readers can easily distinguish between the two cases.*

Notice that $\boldsymbol{y}$ is uniquely determined from $\boldsymbol{x} = \boldsymbol{V}_k \boldsymbol{y}$ since $\boldsymbol{V}_k$ has full-column rank. Thus, $F^{(k)}(\boldsymbol{y}, \lambda)$ and $J^{(k)}(\boldsymbol{y}, \lambda)$ are well-defined. The next result shows how we can obtain $F^{(k)}(\boldsymbol{y}, \lambda)$ and $J^{(k)}(\boldsymbol{y}, \lambda)$ from $\boldsymbol{B}_k$ without any extra computations.

**Lemma 3.1** *For any $\boldsymbol{x} \in \text{span}\{\boldsymbol{V}_k\}$ with the form $\boldsymbol{x} = \boldsymbol{V}_k \boldsymbol{y}$, the projected gradient of $\mathcal{L}(\boldsymbol{x}, \lambda)$ has the expression*

$$F^{(k)}(\boldsymbol{y}, \lambda) = \begin{pmatrix} \lambda \boldsymbol{B}_k^\top (\boldsymbol{B}_k \boldsymbol{y} - \beta_1 e_1) + \boldsymbol{y} \\ \frac{1}{2} \|\boldsymbol{B}_k \boldsymbol{y} - \beta_1 e_1\|_2^2 - \frac{\tau m}{2} \end{pmatrix}, \tag{3.12}$$

*and the projected Jacobian of $F(\boldsymbol{x}, \lambda)$ has the expression*

$$J^{(k)}(\boldsymbol{y}, \lambda) = \begin{pmatrix} \lambda \boldsymbol{B}_k^\top \boldsymbol{B}_k + \boldsymbol{I} & \boldsymbol{B}_k^\top (\boldsymbol{B}_k \boldsymbol{y} - \beta_1 e_1) \\ (\boldsymbol{B}_k \boldsymbol{y} - \beta_1 e_1)^\top \boldsymbol{B}_k & 0 \end{pmatrix} \tag{3.13}$$

Now we can compute the projected Newton direction used for updating the solution. Starting from an initial solution $(\boldsymbol{x}_0, \lambda_0)$, consider the following two cases:

- Update $(\boldsymbol{x}_k, \lambda_k)$ from $(\boldsymbol{x}_{k-1}, \lambda_{k-1})$ for $k \le k_t$. Suppose at the $(k-1)$-th iteration, we have computed a solution $\boldsymbol{x}_{k-1} = \boldsymbol{V}_{k-1} \boldsymbol{y}_{k-1}$, where $\boldsymbol{x}_0 := \boldsymbol{0}$ and $\boldsymbol{y}_0 := ()$ is an empty vector. Let $\bar{\boldsymbol{y}}_{k-1} = (\boldsymbol{y}_{k-1}^\top, 0)^\top \in \mathbb{R}^k$. If $J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})$ is nonsingular, we can calculate the Newton direction for the projected function $F^{(k)}(\boldsymbol{y}, \lambda)$ at $(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})$:

$$\begin{pmatrix} \Delta \boldsymbol{y}_k \\ \Delta \lambda_k \end{pmatrix} = -J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})^{-1} F^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1}). \tag{3.14a}$$

  Then we update $(\bar{\boldsymbol{y}}_k, \lambda_k)$ by

$$\begin{cases} \boldsymbol{y}_k = \bar{\boldsymbol{y}}_{k-1} + \gamma_k \Delta \boldsymbol{y}_k \\ \lambda_k = \lambda_{k-1} + \gamma_k \Delta \lambda_k \end{cases} \tag{3.14b}$$

  with a suitably chosen step-length $\gamma_k > 0$, and let $\boldsymbol{x}_k = \boldsymbol{V}_k \boldsymbol{y}_k$.

- Update $(\boldsymbol{x}_k, \lambda_k)$ from $(\boldsymbol{x}_{k-1}, \lambda_{k-1})$ for $k > k_t$. At each iteration, we seek a solution of the form $\boldsymbol{x}_k = \boldsymbol{V}_{k_t} \boldsymbol{y}_k$ with $\boldsymbol{y}_k \in \mathbb{R}^{k_t}$. We calculate the Newton direction

$$\begin{pmatrix} \Delta \boldsymbol{y}_k \\ \Delta \lambda_k \end{pmatrix} = -J^{(k)}(\boldsymbol{y}_{k-1}, \lambda_{k-1})^{-1} F^{(k)}(\boldsymbol{y}_k, \lambda_{k-1}), \tag{3.14c}$$

  and then compute

$$\begin{cases} \boldsymbol{y}_k = \boldsymbol{y}_{k-1} + \gamma_k \Delta \boldsymbol{y}_k \\ \lambda_k = \lambda_{k-1} + \gamma_k \Delta \lambda_k \end{cases} \tag{3.14d}$$

  to get $\boldsymbol{x}_k = \boldsymbol{V}_{k_t} \boldsymbol{y}_k$.

For both the two cases, we call $(\Delta \boldsymbol{y}_k, \Delta \lambda_k)$ the *projected Newton direction*, since it is the Newton direction of the projected problem. The corresponding update formula for $\boldsymbol{x}_k$ is

$$\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \gamma_k \Delta \boldsymbol{x}_k, \quad \Delta \boldsymbol{x}_k := \boldsymbol{V}_k \Delta \boldsymbol{y}_k. \tag{3.15}$$

Remember that $\boldsymbol{V}_k = \boldsymbol{V}_{k_t}$ for $k > k_t$. The above formula is easy to be verified:

$$\boldsymbol{x}_k = \boldsymbol{V}_k \boldsymbol{y}_k = \boldsymbol{V}_k \bar{\boldsymbol{y}}_{k-1} + \gamma_k \boldsymbol{V}_k \Delta \boldsymbol{y}_k = \boldsymbol{V}_{k-1} \boldsymbol{y}_{k-1} + \Delta \boldsymbol{x}_k = \boldsymbol{x}_{k-1} + \gamma_k \Delta \boldsymbol{x}_k$$

for $k \le k_t$, and

$$\boldsymbol{x}_k = \boldsymbol{V}_{k_t} \boldsymbol{y}_k = \boldsymbol{V}_{k_t} \boldsymbol{y}_{k-1} + \gamma_k \boldsymbol{V}_{k_t} \Delta \boldsymbol{y}_k = \boldsymbol{x}_{k-1} + \gamma_k \Delta \boldsymbol{x}_k$$

for $k > k_t$. For notational simplicity, in the following part of the paper, we always use the unified notation

$$\bar{\boldsymbol{y}}_{k-1} = \begin{cases} (\boldsymbol{y}_{k-1}^\top, \ 0)^\top, & k \le k_t \\ \boldsymbol{y}_{k-1}, & k > k_t \end{cases} \tag{3.16}$$

for $\bar{\boldsymbol{y}}_{k-1}$. Following the notations in (3.1) and (3.16), we can use (3.14a), (3.14b) and (3.15) to describe the update procedure for both the two cases.

It is vital to make sure that the updating procedure does not breakdown, i.e. the projected Jacobian matrix $J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})$ is nonsingular. This desired property is given in the following result. The proof appears as a part of the proof of Lemma 4.5.

**Proposition 3.2** *If we choose* $\boldsymbol{x}_0 = \mathbf{0}$ *and* $\bar{\boldsymbol{y}}_0 = 0$*, then at each iteration* $J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})$ *is nonsingular as long as* $\lambda_{k-1} \geq 0$*.*

In order to investigate the convergence behavior of the algorithm, define the following merit function:

$$h(\boldsymbol{x}, \lambda) = \frac{1}{2}\left( \|\lambda \boldsymbol{A}^\top \boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) + \boldsymbol{N}^{-1}\boldsymbol{x}\|_{\boldsymbol{N}}^2 + \left( \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_{\boldsymbol{M}^{-1}}^2 - \frac{\tau m}{2} \right)^2 \right). \tag{3.17}$$

Note by Corollary 2.1 that $(\boldsymbol{x}^*, \lambda^*)$ is the unique minimizer of $h(\boldsymbol{x}, \lambda)$ and $h(\boldsymbol{x}^*, \lambda^*) = 0$. The following result shows that $(\Delta \boldsymbol{x}_k^\top, \Delta\lambda_k)^\top$ is indeed a descent direction for $h(\boldsymbol{x}, \lambda)$.

**Theorem 3.1** *Let* $(\Delta \boldsymbol{y}_k^\top, \Delta\lambda_k)^\top$ *be defined as* (3.14a) *and let* $\Delta \boldsymbol{x}_k = \boldsymbol{V}_k \Delta \boldsymbol{y}_k$*. Then it holds*

$$\nabla h(\boldsymbol{x}_{k-1}, \lambda_{k-1})^\top \begin{pmatrix} \Delta \boldsymbol{x}_k \\ \Delta\lambda_k \end{pmatrix} = -2h(\boldsymbol{x}_{k-1}, \lambda_{k-1}) \leq 0. \tag{3.18}$$

Theorem 3.1 is a desired property for a gradient descent type algorithm. At the $(k-1)$-th iteration, if $h(\boldsymbol{x}_{k-1}, \lambda_{k-1}) = 0$, then we have $(\boldsymbol{x}_{k-1}, \lambda_{k-1}) = (\boldsymbol{x}^*, \lambda^*)$, meaning we have obtained the unique solution to (2.3). Otherwise, $(\Delta \boldsymbol{x}_k^\top, \Delta\lambda_k)^\top$ is a descent direction of $h(\boldsymbol{x}, \lambda)$ at the point $(\boldsymbol{x}_{k-1}, \lambda_{k-1})$, thereby we can continue updating the solution by a backtracking line search strategy.

**Step 3: Determine step-length by backtracking line search.** For the case that $h(\boldsymbol{x}_{k-1}, \lambda_{k-1}) \neq 0$, we need to determine a step-length $\gamma_k$ such that $h(\boldsymbol{x}_k, \lambda_k)$ decreases strictly. To this end, we use the backtracking line search procedure to ensure that the *Armijo condition* [28, §3.1] is satisfied:

$$h(\boldsymbol{x}_k, \lambda_k) \leq h(\boldsymbol{x}_{k-1}, \lambda_{k-1}) + c\gamma_k (\Delta \boldsymbol{x}_{k-1}^\top, \Delta\lambda_k) \nabla h(\boldsymbol{x}_{k-1}, \lambda_{k-1}), \tag{3.19}$$

where $(\boldsymbol{x}_k, \lambda_k) = (\boldsymbol{x}_{k-1}, \lambda_{k-1}) + \gamma_k(\Delta \boldsymbol{x}_k, \Delta\lambda_k)$, and $c \in (0, 1)$ is a fixed constant. At each iteration, we can quickly compute $h(\boldsymbol{x}_k, \lambda_k)$ based on the following result.

**Lemma 3.2** *Let*

$$\bar{F}^{(k)}(\boldsymbol{y}, \lambda) = \begin{pmatrix} \lambda \bar{\boldsymbol{B}}_k^\top (\bar{\boldsymbol{B}}_k \boldsymbol{y} - \beta_1 e_1) + y \\ \frac{1}{2}\|\bar{\boldsymbol{B}}_k \boldsymbol{y} - \beta_1 e_1\|_2^2 - \frac{\tau m}{2} \end{pmatrix}, \quad \bar{\boldsymbol{B}}_k = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_{k+1} & \alpha_{k+1} \end{pmatrix}.$$

*Then we have*

$$h(\boldsymbol{x}_{k-1}, \lambda_{k-1}) = \frac{1}{2}\|F^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})\|_2^2, \tag{3.20}$$

$$h(\boldsymbol{x}_k, \lambda_k) = \frac{1}{2}\|\bar{F}^{(k)}(\bar{\boldsymbol{y}}_k, \lambda_k)\|_2^2. \tag{3.21}$$

We remark that in the above expression we have $\bar{\boldsymbol{B}}_k = \boldsymbol{B}_{k_t}$ for $k \geq k_t$, and specifically, we have $\bar{\boldsymbol{B}}_k = \underline{\boldsymbol{B}}_{k_t}^\top$ if $\beta_{k_t+1} = 0$.

The following theorem shows the existence of a suitable step-length; see e.g. [4, pp. 121, Theorem 2.1] for details.

**Theorem 3.2 (Armijo-backtracking)** *For any continuously differentiable function* $f(\boldsymbol{s}) : \mathbb{R}^l \to \mathbb{R}$*, suppose* $\nabla f$ *is Lipschitz continuous with constant* $\zeta(\boldsymbol{s})$ *at* $\boldsymbol{s}$*. If* $\boldsymbol{p}$ *is a descent direction at* $\boldsymbol{s}$*, i.e* $\nabla f(\boldsymbol{s})^\top \boldsymbol{p} < 0$*, then for a fixed* $c \in (0, 1)$ *the Armijo condition*

$$f(\boldsymbol{s} + \gamma \boldsymbol{p}) \leq f(\boldsymbol{s}) + c\gamma \nabla f(\boldsymbol{s})^T \boldsymbol{p}$$

*is satisfied for all* $\gamma \in [0, \gamma_{\max}]$ *where*

$$\gamma_{\max} = \frac{2(c-1)\nabla f(\boldsymbol{s})^\top \boldsymbol{p}}{\zeta(\boldsymbol{s})\|\boldsymbol{p}\|^2}.$$

Therefore, with the aid of Lemma 3.2, a suitable step-length $\gamma_k$ can be found using the following backtracking line search strategy.

**Routine 1** ***Armijo backtracking line search:***

1. *Given* $\gamma_{init} > 0$, *let* $\gamma^{(0)} = \gamma_{init}$ *and* $l = 0$.
2. *Until* $\frac{1}{2}\|\bar{F}^{(k)}(\bar{\boldsymbol{y}}_k, \lambda_k)\|_2^2 < \left(\frac{1}{2} - c\gamma^{(l)}\right)\|F^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})\|_2^2$,
   *(i) set* $\gamma^{(l+1)} = \eta\gamma^{(l)}$, *where* $\eta \in (0,1)$ *is a fixed constant;*
   *(ii)* $l \leftarrow l + 1$.
3. *Set* $\gamma_k = \gamma^{(l)}$.

We set $c = 10^{-4}$, $\gamma_{\text{init}} = 1.0$ and $\eta = 0.9$ by default. Note that at each iteration we need to make sure that $\lambda_k > 0$. Suppose at the $(k-1)$-th iteration we already have $\lambda_{k-1} > 0$. Then the at $k$-th iteration if $\Delta\lambda_k < 0$, we only need to enforce $\gamma_{\text{init}} < -\lambda_{k-1}/\Delta\lambda_k$.

Overall, the whole procedure of the projected Newton method is presented in Algorithm 3. In the PNT algorithm, at each $k$-th iteration, computing the projected Newton direction only needs to solve the $(k+1)$-order linear system (3.14a), which can be done very quickly when $k \ll n$. From the termination step $k_t$, at each iteration, a $(k_t+1)$-order linear system (3.14c) needs to be solved. Furthermore, we numerically find that the algorithm almost always obtains a satisfied solution before gen-GKB terminates.

**Algorithm 3** Projected Newton method (PNT) for (2.3) and (2.4)

**Input:** $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{b} \in \mathbb{R}^m$, $\boldsymbol{M} \in \mathbb{R}^{m\times m}$, $\boldsymbol{N} \in \mathbb{R}^{n\times n}$, $\tau \gtrsim 1$

Initialization: $\lambda_0 > 0$, $\bar{\boldsymbol{y}}_0 = \boldsymbol{0}$; $c = 10^{-4}$, $\eta = 0.9$; tol$>0$
Compute $\beta_1$, $\alpha_1$, $\boldsymbol{u}_1$, $\boldsymbol{v}_1$ by Algorithm 2
**for** $k = 1, 2, \ldots$ **do**
    Compute $\beta_{k+1}$, $\alpha_{k+1}$, $\boldsymbol{u}_{k+1}$, $\boldsymbol{v}_{k+1}$ by Algorithm 2; Form $\boldsymbol{B}_{k+1}$ and $\boldsymbol{V}_k$
    (Terminate gen-GKB if $\beta_{k+1}$ or $\alpha_{k+1}$ is extremely small)
    Compute $F^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})$ and $J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})$ by (3.12) and (3.13)
    Compute $(\Delta\boldsymbol{y}_k, \Delta\lambda_k)$ by solving (3.14a)
    **if** $\Delta\lambda_k > 0$ **then**
        $\gamma_{\text{init}} = 1$
    **else**
        $\gamma_{\text{init}} = \min\{1, -\eta\lambda_{k-1}/\Delta\lambda_k\}$ ▷ Ensure the positivity of $\lambda_k$
    **end if**
    Determine the step-length $\gamma_k$ by Routine 1
    Update $(\boldsymbol{y}_k, \lambda_k)$ by (3.14b)
    **if** $\frac{1}{2}\|\bar{F}^{(k)}(\bar{\boldsymbol{y}}_k, \lambda_k)\|_2 \le$ tol **then**
        Compute $\boldsymbol{x}_k = \boldsymbol{V}_k\boldsymbol{y}_k$; Stop iteration
    **end if**
**end for**

**Output:** Final solution $(\boldsymbol{x}_k, \lambda_k)$

## 3.3 Proofs

Here we give the proofs of all the results in Section 3.2. Remember that we always follow the notations as stated in Remark 3.1 and (3.16).

**Proof of Lemma 3.1.** By (3.10) and (3.11) we have

$$F^{(k)}(\boldsymbol{y},\lambda)=\begin{pmatrix}\lambda(\boldsymbol{A}\boldsymbol{V}_k)^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}-\boldsymbol{b})+\boldsymbol{V}_k^\top\boldsymbol{N}^{-1}\boldsymbol{V}_k\boldsymbol{y}\\ \frac{1}{2}\|\boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}}-\frac{\tau m}{2}\end{pmatrix},$$

and

$$J^{(k)}(\boldsymbol{y},\lambda)=\begin{pmatrix}\lambda(\boldsymbol{A}\boldsymbol{V}_k)^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{V}_k)+\boldsymbol{V}_k^\top\boldsymbol{N}^{-1}\boldsymbol{V}_k & (\boldsymbol{A}\boldsymbol{V}_k)^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}-\boldsymbol{b})\\ (\boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}-\boldsymbol{b})^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{V}_k) & 0\end{pmatrix}.$$

Using relations (3.7) and Proposition 3.1, we have $\boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}-\boldsymbol{b}=\boldsymbol{U}_{k+1}(\boldsymbol{B}_k-\beta_1e_1)$, $\boldsymbol{V}_k^\top\boldsymbol{N}^{-1}\boldsymbol{V}_k=\boldsymbol{I}$ and $\boldsymbol{U}_{k+1}^\top\boldsymbol{M}^{-1}\boldsymbol{U}_{k+1}=\boldsymbol{I}$, leading to

$$(\boldsymbol{A}\boldsymbol{V}_k)^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}-\boldsymbol{b})=(\boldsymbol{U}_{k+1}\boldsymbol{B}_k)^\top\boldsymbol{M}^{-1}\boldsymbol{U}_{k+1}(\boldsymbol{B}_k-\beta_1e_1)=\boldsymbol{B}_k^\top(\boldsymbol{B}_k-\beta_1e_1),$$

and

$$\|\boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}}=\|\boldsymbol{U}_{k+1}(\boldsymbol{B}_k-\beta_1e_1)\|^2_{\boldsymbol{M}^{-1}}=\|\boldsymbol{B}_k-\beta_1e_1\|_2^2.$$

We remark that if $\beta_{k_t+1}=0$, then for $k\geq k_t$, the relation $\boldsymbol{A}\boldsymbol{V}_k\boldsymbol{y}-\boldsymbol{b}=\boldsymbol{U}_{k+1}(\boldsymbol{B}_k-\beta_1e_1)$ is replaced by $\boldsymbol{A}\boldsymbol{V}_{k_t}\boldsymbol{y}-\boldsymbol{b}=\boldsymbol{U}_{k_t}(\underline{\boldsymbol{B}}_{k_t}-\beta_1e_1)$. Therefore, the above identity is also applied to the case $k\geq k_t$. Now we have proved (3.12). The expression (3.13) can be proved similarly. □

In order to prove Lemma 3.2, we first give the following result.

**Lemma 3.3** *Let* $\widehat{\boldsymbol{N}}=\begin{pmatrix}\boldsymbol{N} & \\ & 1\end{pmatrix}$. *Then we have the following identity:*

$$\|F(\boldsymbol{x}_{k-1},\lambda_{k-1})\|_{\widehat{\boldsymbol{N}}}=\|F^{(k)}(\bar{\boldsymbol{y}}_{k-1},\lambda_{k-1})\|_2. \tag{3.22}$$

**Proof.** First notice that

$$\|F(\boldsymbol{x}_{k-1},\lambda_{k-1})\|^2_{\widehat{\boldsymbol{N}}}=\left\|\lambda_{k-1}\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})+\boldsymbol{N}^{-1}\boldsymbol{x}\right\|^2_{\boldsymbol{N}}+\left(\frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}}-\frac{\tau m}{2}\right)^2.$$

For the first term of the above summation, we have

$$\begin{aligned}&\left\|\lambda_{k-1}\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})+\boldsymbol{N}^{-1}\boldsymbol{x}_{k-1}\right\|^2_{\boldsymbol{N}}\\ =&\left(\lambda_{k-1}\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})+\boldsymbol{N}^{-1}\boldsymbol{x}_{k-1}\right)^\top\boldsymbol{N}\left(\lambda_{k-1}\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})+\boldsymbol{N}^{-1}\boldsymbol{x}_{k-1}\right),\end{aligned}$$

and

$$\begin{aligned}&\boldsymbol{N}\left(\lambda_{k-1}\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})+\boldsymbol{N}^{-1}\boldsymbol{x}_{k-1}\right)\\ =&\lambda_{k-1}\boldsymbol{N}\boldsymbol{A}^\top\boldsymbol{M}^{-1}\boldsymbol{U}_{k+1}(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1e_1)+\boldsymbol{V}_k\bar{\boldsymbol{y}}_{k-1}\\ =&\lambda_{k-1}(\boldsymbol{V}_k\boldsymbol{B}_k^\top+\alpha_{k+1}\boldsymbol{v}_{k+1}e_{k+1}^\top)(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1e_1)+\boldsymbol{V}_k\bar{\boldsymbol{y}}_{k-1}\\ =&\boldsymbol{V}_k\left(\lambda_{k-1}\boldsymbol{B}_k^\top(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1e_1)+\bar{\boldsymbol{y}}_{k-1}\right),\end{aligned}$$

where we have used

$$\alpha_{k+1}\boldsymbol{v}_{k+1}e_{k+1}^\top(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1e_1)=\alpha_{k+1}\beta_{k+1}\boldsymbol{v}_{k+1}e_k^\top\bar{\boldsymbol{y}}_{k-1}=\begin{cases}0 & (e_k^\top\bar{\boldsymbol{y}}_{k-1}=0\text{ for }k\leq k_t)\\ 0 & (\alpha_{k+1}\beta_{k+1}=0\text{ for }k>k_t)\end{cases}.$$

Similarly, we have

$$\lambda_{k-1}\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})+\boldsymbol{N}^{-1}\boldsymbol{x}_{k-1}=\boldsymbol{N}^{-1}\boldsymbol{V}_k\left(\lambda_{k-1}\boldsymbol{B}_k^\top(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)+\bar{\boldsymbol{y}}_{k-1}\right).$$

Using $\boldsymbol{V}_k^\top\boldsymbol{N}^{-1}\boldsymbol{V}_k=\boldsymbol{I}$, we get

$$\left\|\lambda_{k-1}\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})+\boldsymbol{N}^{-1}\boldsymbol{x}_{k-1}\right\|_{\boldsymbol{N}}=\left\|\lambda_{k-1}\boldsymbol{B}_k^\top(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)+\bar{\boldsymbol{y}}_{k-1}\right\|_2.$$

From the above derivation, we also have $\frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}}-\frac{\tau m}{2}=\frac{1}{2}\|\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\|_2^2-\frac{\tau m}{2}$. The desired result immediately follows by using (3.12). □

**Proof of Lemma 3.2.** First notice that

$$h(\boldsymbol{x},\lambda)=\frac{1}{2}\|F(\boldsymbol{x},\lambda)\|^2_{\widehat{N}}. \tag{3.23}$$

Combining the above relation with Lemma 3.3 we obtain (3.20). Also, for $k<k_t$ we have

$$h(\boldsymbol{x}_k,\lambda_k)=\frac{1}{2}\|F^{(k+1)}(\bar{\boldsymbol{y}}_k,\lambda_k)\|_2^2$$

with

$$F^{(k+1)}(\bar{\boldsymbol{y}}_k,\lambda_k)=\begin{pmatrix}\lambda\boldsymbol{B}_{k+1}^\top(\boldsymbol{B}_{k+1}\bar{\boldsymbol{y}}_k-\beta_1 e_1)+\bar{\boldsymbol{y}}_k\\ \frac{1}{2}\|\boldsymbol{B}_{k+1}\bar{\boldsymbol{y}}_k-\beta_1 e_1\|_2^2-\frac{\tau m}{2}\end{pmatrix}.$$

Since the last element of $\beta_1 e_1$ and $\bar{\boldsymbol{y}}_k$ is zero, it is easy to verify that

$$\boldsymbol{B}_{k+1}\bar{\boldsymbol{y}}_k-\beta_1 e_1=\begin{pmatrix}\bar{\boldsymbol{B}}_k\bar{\boldsymbol{y}}_k-\beta_1 e_1\\ 0\end{pmatrix},\quad \boldsymbol{B}_{k+1}^\top(\boldsymbol{B}_{k+1}\bar{\boldsymbol{y}}_k-\beta_1 e_1)=\bar{\boldsymbol{B}}_k^\top(\bar{\boldsymbol{B}}_k\bar{\boldsymbol{y}}_k-\beta_1 e_1).$$

For $k\geq k_t$, we have $F^{(k)}(\boldsymbol{y},\lambda)=F^{(k+1)}(\boldsymbol{y},\lambda)=\bar{F}^{(k)}(\boldsymbol{y},\lambda)$, and (3.21) is always true. Therefore, we finally prove (3.21). □

In order to prove Theorem 3.1, we need Lemma 3.3 and the following result.

**Lemma 3.4** *For any $k\geq 1$ we have the following identity:*

$$\begin{pmatrix}\boldsymbol{V}_k^\top & \\ & 1\end{pmatrix}J(\boldsymbol{x}_{k-1},\lambda_{k-1})\widehat{\boldsymbol{N}}F(\boldsymbol{x}_{k-1},\lambda_{k-1})=J^{(k)}(\bar{\boldsymbol{y}}_{k-1},\lambda_{k-1})F^{(k)}(\bar{\boldsymbol{y}}_{k-1},\lambda_{k-1}).$$

**Proof.** First notice from (2.9) that

$$\begin{pmatrix}\boldsymbol{V}_k^\top & \\ & 1\end{pmatrix}J(\boldsymbol{x}_{k-1},\lambda_{k-1})\widehat{\boldsymbol{N}}=\begin{pmatrix}\lambda_{k-1}(\boldsymbol{A}\boldsymbol{V}_k)^\top\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{N}+\boldsymbol{V}_k^\top & (\boldsymbol{A}\boldsymbol{V}_k)^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})\\ (\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})^\top\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{N} & 0\end{pmatrix}.$$

Using (3.7c) and similar derivations to the proof of Lemma 3.3, we get

$$\begin{aligned}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})^\top\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{N}&=\left(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\right)^\top\boldsymbol{U}_{k+1}^\top\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{N}\\&=\left(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\right)^\top\left(\boldsymbol{B}_k\boldsymbol{V}_k^\top+\alpha_{k+1}e_{k+1}\boldsymbol{v}_{k+1}^\top\right)\\&=\left(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\right)^\top\boldsymbol{B}_k\boldsymbol{V}_k^\top.\end{aligned} \tag{3.24}$$

Also, we can get

$$\begin{aligned}(\boldsymbol{A}\boldsymbol{V}_k)^\top\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{N}=(\boldsymbol{U}_{k+1}\boldsymbol{B}_k)^\top\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{N}&=\boldsymbol{B}_k^\top\left(\boldsymbol{B}_k\boldsymbol{V}_k^\top+\alpha_{k+1}e_{k+1}\boldsymbol{v}_{k+1}^\top\right)\\&=\boldsymbol{B}_k^\top\boldsymbol{B}_k\boldsymbol{V}_k^\top+\alpha_{k+1}\beta_{k+1}e_k\boldsymbol{v}_{k+1}^\top,\end{aligned}$$

and

$$(\boldsymbol{A}\boldsymbol{V}_k)^\top \boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b}) = (\boldsymbol{B}_k\boldsymbol{U}_{k+1})^\top \boldsymbol{M}^{-1}\boldsymbol{U}_{k+1}(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)$$
$$= \boldsymbol{B}_k^\top(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1).$$

Using (3.13), we get

$$\begin{aligned}
&\begin{pmatrix}\boldsymbol{V}_k^\top & \\ & 1\end{pmatrix} J(\boldsymbol{x}_{k-1},\lambda_{k-1})\widehat{\boldsymbol{N}} \\
&= \begin{pmatrix}(\lambda_{k-1}\boldsymbol{B}_k^\top\boldsymbol{B}_k+\boldsymbol{I})\boldsymbol{V}_k^\top & \boldsymbol{B}_k^\top(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)\\ (\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)^\top\boldsymbol{B}_k\boldsymbol{V}_k^\top & 0\end{pmatrix} + \begin{pmatrix}\alpha_{k+1}\beta_{k+1}e_{k+1}\boldsymbol{v}_{k+1}^\top & \\ & 0\end{pmatrix} \\
&= J^{(k)}(\bar{\boldsymbol{y}}_{k-1},\lambda_{k-1})\begin{pmatrix}\boldsymbol{V}_k^\top & \\ & 1\end{pmatrix} + \begin{pmatrix}\alpha_{k+1}\beta_{k+1}e_k\boldsymbol{v}_{k+1}^\top & \\ & 0\end{pmatrix}
\end{aligned}$$

Using similar derivations to the proof of Lemma 3.3, we get

$$\begin{aligned}
F(\boldsymbol{x}_{k-1},\lambda_{k-1}) &= \begin{pmatrix}\boldsymbol{N}^{-1} & \\ & 1\end{pmatrix}\begin{pmatrix}\lambda_{k-1}\boldsymbol{N}\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})+\boldsymbol{x}_{k-1}\\ \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}}-\frac{\tau m}{2}\end{pmatrix} \\
&= \begin{pmatrix}\boldsymbol{N}^{-1} & \\ & 1\end{pmatrix}\begin{pmatrix}\boldsymbol{V}_k & \\ & 1\end{pmatrix}\begin{pmatrix}\lambda_{k-1}\boldsymbol{B}_k^\top(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)+\bar{\boldsymbol{y}}_{k-1}\\ \frac{1}{2}\|\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\|_2^2-\frac{\tau m}{2}\end{pmatrix} \\
&= \begin{pmatrix}\boldsymbol{N}^{-1} & \\ & 1\end{pmatrix}\begin{pmatrix}\boldsymbol{V}_k & \\ & 1\end{pmatrix} F^{(k)}(\bar{\boldsymbol{y}}_{k-1},\lambda_{k-1}).
\end{aligned}$$

Using the relations

$$\begin{pmatrix}\boldsymbol{V}_k^\top & \\ & 1\end{pmatrix}\begin{pmatrix}\boldsymbol{N}^{-1} & \\ & 1\end{pmatrix}\begin{pmatrix}\boldsymbol{V}_k & \\ & 1\end{pmatrix} = \boldsymbol{I},\quad \begin{pmatrix}\alpha_{k+1}\beta_{k+1}e_k\boldsymbol{v}_{k+1}^\top & \\ & 0\end{pmatrix}\begin{pmatrix}\boldsymbol{N}^{-1} & \\ & 1\end{pmatrix}\begin{pmatrix}\boldsymbol{V}_k & \\ & 1\end{pmatrix} = \boldsymbol{0},$$

we finally obtain the desired result. □

Now we can now give the proof of Theorem 3.1.

**Proof of Theorem 3.1.** Noticing (3.23) and that $J(\boldsymbol{x},\lambda)$ is the Jacobian of $F(\boldsymbol{x},\lambda)$, we have

$$\nabla h(\boldsymbol{x},\lambda) = J(\boldsymbol{x},\lambda)\widehat{\boldsymbol{N}}F(\boldsymbol{x},\lambda).$$

Using Lemma 3.3 and Lemma 3.4, we obtain

$$\begin{aligned}
\nabla h(\boldsymbol{x}_{k-1},\lambda_{k-1})^\top\begin{pmatrix}\Delta\boldsymbol{x}_k\\ \Delta\lambda_k\end{pmatrix} &= \begin{pmatrix}\Delta\boldsymbol{y}_k\\ \Delta\lambda_k\end{pmatrix}^\top\begin{pmatrix}\boldsymbol{V}_k^\top & \\ & 1\end{pmatrix} J(\boldsymbol{x}_{k-1},\lambda_{k-1})\widehat{\boldsymbol{N}}F(\boldsymbol{x}_{k-1},\lambda_{k-1}) \\
&= \begin{pmatrix}\Delta\boldsymbol{y}_k\\ \Delta\lambda_k\end{pmatrix}^\top J^{(k)}(\bar{\boldsymbol{y}}_{k-1},\lambda_{k-1})F^{(k)}(\bar{\boldsymbol{y}}_{k-1},\lambda_{k-1}) \\
&= -\|F^{(k)}(\bar{\boldsymbol{y}}_{k-1},\lambda_{k-1})\|_2^2 = -\|F(\boldsymbol{x}_{k-1},\lambda_{k-1})\|^2_{\widehat{\boldsymbol{N}}} \\
&= -2h(\boldsymbol{x}_{k-1},\lambda_{k-1}) \le 0.
\end{aligned}$$

The proof is completed. □

# 4 Convergence analysis

This section aims to prove the convergence of the iterated solution, which is stated in the following result and Corollary 4.1.

**Theorem 4.1** *Suppose the PNT algorithm is initialized with* $\bar{\boldsymbol{y}}_0 = \boldsymbol{0}$, $\boldsymbol{x}_0 = \boldsymbol{0}$ *and* $\lambda_0 > 0$. *Then we either have*

$$h(\boldsymbol{x}_k, \lambda_k) = 0 \tag{4.1}$$

*for some* $k < \infty$, *or have*

$$\lim_{k\to\infty} h(\boldsymbol{x}_k, \lambda_k) = 0. \tag{4.2}$$

Notice that $(\boldsymbol{x}^*, \lambda^*)$ is the unique minimizer of $h(\boldsymbol{x}, \lambda)$ and $h(\boldsymbol{x}^*, \lambda^*) = 0$. Therefore, (4.1) implies that the algorithm finds the exact solution to (2.3) and (2.4) at the $k$-th iteration. In the following part, we prove (4.2) under the assumption that $h(\boldsymbol{x}_k, \lambda_k) > 0$ for any $k \geq 1$.

We need a series of lemmas, which are Lemma 4.1–Lemma 4.6. All these lemmas follow the assumption of Theorem 4.1.

**Lemma 4.1** *For any matrix* $\boldsymbol{C} \in \mathbb{R}^{m\times n}$ *with full column rank and* $\boldsymbol{d} \in \mathbb{R}^m$, *if the vector sequence* $\{\boldsymbol{w}_k\} \in \mathbb{R}^n$ *satisfies*

$$\lim_{k\to\infty} \|\boldsymbol{C}^\top(\boldsymbol{C}\boldsymbol{w}_k - \boldsymbol{d})\|_2 = 0,$$

*then*

$$\lim_{k\to\infty} \boldsymbol{w}_k = \boldsymbol{w}^* := \operatorname*{argmin}_{\boldsymbol{w}\in\mathbb{R}^n} \|\boldsymbol{C}\boldsymbol{w} - \boldsymbol{d}\|_2.$$

**Proof.** First note that $\boldsymbol{w}^*$ is well-defined, since $\operatorname{argmin}_{\boldsymbol{w}\in\mathbb{R}^n} \|\boldsymbol{C}\boldsymbol{w}-\boldsymbol{d}\|_2$ has a unique solution for the full column rank matrix $\boldsymbol{C}$. For any $\boldsymbol{w}_k$, let $\boldsymbol{w}_k = \boldsymbol{w}^* + \bar{\boldsymbol{w}}_k$. Then we have

$$\lim_{k\to\infty} \|\boldsymbol{C}^\top(\boldsymbol{C}\boldsymbol{w}_k - \boldsymbol{d})\|_2 = \lim_{k\to\infty} \|\boldsymbol{C}^\top(\boldsymbol{C}\boldsymbol{w}^* - \boldsymbol{d}) + \boldsymbol{C}^\top\boldsymbol{C}\bar{\boldsymbol{w}}_k\|_2 = \lim_{k\to\infty} \|\boldsymbol{C}^\top\boldsymbol{C}\bar{\boldsymbol{w}}_k\|_2 = 0,$$

since $\boldsymbol{C}^\top(\boldsymbol{C}\boldsymbol{w}^* - \boldsymbol{d}) = \boldsymbol{0}$. Now we have $\|\bar{\boldsymbol{w}}_k\|_2 \to 0$ since $\boldsymbol{C}^\top\boldsymbol{C}$ is positive definite and all norms of $\mathbb{R}^n$ are equivalent. Therefore, we have $\|\boldsymbol{w}_k - \boldsymbol{w}^*\|_2 \to 0$ or the equivalent form $\lim_{k\to\infty} \boldsymbol{w}_k = \boldsymbol{w}^*$. □

**Lemma 4.2** *If the unique solution to* $\min_{\boldsymbol{y}\in\mathbb{R}^{k_t}} \|\boldsymbol{B}_{k_t}y - \beta_1 e_1\|_2$ *is* $\boldsymbol{y}_{\min}$, *then* $\boldsymbol{x}_{\min} := \boldsymbol{V}_{k_t}\boldsymbol{y}_{\min}$ *is the unique solution to*

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{x}\|_{\boldsymbol{N}^{-1}} \quad \text{s.t.} \quad \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_{\boldsymbol{M}^{-1}} = \min. \tag{4.3}$$

**Proof.** It is easy to verify that both $\min_{\boldsymbol{y}\in\mathbb{R}^{k_t}} \|\boldsymbol{B}_{k_t}y - \beta_1 e_1\|_2$ and (4.3) have a unique solution. A vector $\boldsymbol{x}$ is the unique solution to (4.3) if and only if

$$\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) = \boldsymbol{0}, \quad \boldsymbol{x} \perp_{\boldsymbol{N}^{-1}} \mathcal{N}(\boldsymbol{A}),$$

where $\perp_{\boldsymbol{N}^{-1}}$ means the orthogonality relation under the $\boldsymbol{N}^{-1}$-inner product. Now we verify the above two conditions for $\boldsymbol{x}_{\min}$. For the first condition, using the relations $\boldsymbol{A}\boldsymbol{x}_{\min} = \boldsymbol{A}\boldsymbol{V}_{k_t}\boldsymbol{y}_{\min} = \boldsymbol{U}_{k_t+1}\boldsymbol{B}_{k_t}\boldsymbol{y}_{\min}$ and (3.7c), we have

$$\begin{aligned}
\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{\min} - \boldsymbol{b}) &= \boldsymbol{A}^\top\boldsymbol{M}^{-1}\boldsymbol{U}_{k_t+1}(\boldsymbol{B}_{k_t}\boldsymbol{y}_{\min} - \beta_1 e_1) \\
&= \boldsymbol{N}^{-1}(\boldsymbol{V}_{k_t}\boldsymbol{B}_{k_t}^\top + \alpha_{k_t+1}\boldsymbol{v}_{k_t+1}e_{k_t+1}^\top)\,(\boldsymbol{B}_{k_t}\boldsymbol{y}_{\min} - \beta_1 e_1) \\
&= \boldsymbol{N}^{-1}\left[\boldsymbol{V}_{k_t}\boldsymbol{B}_{k_t}^\top(\boldsymbol{B}_{k_t}\boldsymbol{y}_{\min} - \beta_1 e_1) + \alpha_{k_t+1}\beta_{k_t+1}\boldsymbol{v}_{k_t+1}e_{k_t}^\top\boldsymbol{y}_{\min}\right] \\
&= 0,
\end{aligned}$$

since $\boldsymbol{B}_{k_t}^\top(\boldsymbol{B}_{k_t}\boldsymbol{y}_{\min}-\beta_1 e_1)=\boldsymbol{0}$ and $\alpha_{k_t+1}\beta_{k_t+1}=0$. For the second condition, by Proposition 3.1 we have

$$\boldsymbol{x}_{\min}\in \mathrm{span}\{\boldsymbol{V}_{k_t}\}=\mathrm{span}\{(\boldsymbol{N}\boldsymbol{A}^\top\boldsymbol{M}^{-1}\boldsymbol{A})^i\boldsymbol{N}\boldsymbol{A}^\top\boldsymbol{M}^{-1}\boldsymbol{b}\}_{i=0}^{k_t-1}\subseteq\mathcal{R}(\boldsymbol{N}\boldsymbol{A}^\top)=\boldsymbol{N}\mathcal{N}(\boldsymbol{A})^\perp.$$

Write $\boldsymbol{x}_{\min}=\boldsymbol{N}\bar{\boldsymbol{x}}_{\min}$ with $\bar{\boldsymbol{x}}_{\min}\in\mathcal{N}(\boldsymbol{A})^\perp$. For any $\boldsymbol{w}\in\mathcal{N}(\boldsymbol{A})$, we have

$$\langle\boldsymbol{x}_{\min},\boldsymbol{w}\rangle_{\boldsymbol{N}^{-1}}=\langle\boldsymbol{N}\bar{\boldsymbol{x}}_{\min},\boldsymbol{w}\rangle_{\boldsymbol{N}^{-1}}=\langle\bar{\boldsymbol{x}}_{\min},\boldsymbol{w}\rangle_2=0.$$

Therefore, it holds that $\boldsymbol{x}_{\min}\perp_{\boldsymbol{N}^{-1}}\mathcal{N}(\boldsymbol{A})$. □

**Lemma 4.3** *There exist a positive constant $C_1$ such that for any $k\geq 1$, it holds*

$$\|\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})\|_{\boldsymbol{N}}=\|\boldsymbol{B}_k^\top(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)\|_2\geq C_1>0 \tag{4.4}$$

**Proof.** First, we get from (3.24) the first indentity:

$$\begin{aligned}
&\|\boldsymbol{A}^\top\boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})\|_N^2\\
&=(\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})^\top\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{N}\boldsymbol{N}^{-1}\left((\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b})^\top\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{N}\right)^\top\\
&=\left(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\right)^\top\boldsymbol{B}_k\boldsymbol{V}_k^\top\boldsymbol{N}^{-1}\boldsymbol{V}_k^\top\boldsymbol{B}_k^\top\left(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\right)\\
&=\|\boldsymbol{B}_k^\top\left(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\right)\|_2^2.
\end{aligned}$$

Then, we prove

$$\|\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b}\|_{\boldsymbol{M}^{-1}}=\|\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\|_2\geq\sqrt{\tau m} \tag{4.5}$$

by mathematical induction. For $k=1$, we have $\|\boldsymbol{A}\boldsymbol{x}_0-\boldsymbol{b}\|_{\boldsymbol{M}^{-1}}=\|\boldsymbol{b}\|_{\boldsymbol{M}^{-1}}>\sqrt{\tau m}$ since $\boldsymbol{x}_0=\boldsymbol{0}$; see Assumption 2.1. Suppose $\|\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b}\|_{\boldsymbol{M}^{-1}}\geq\sqrt{\tau m}$ for $k\geq 1$. We have

$$\begin{aligned}
\|\boldsymbol{A}\boldsymbol{x}_k-\boldsymbol{b}\|_{\boldsymbol{M}^{-1}}^2&=\|\boldsymbol{A}\boldsymbol{V}_k(\bar{\boldsymbol{y}}_{k-1}+\gamma_k\Delta\boldsymbol{y}_k)-\boldsymbol{b}\|_{\boldsymbol{M}^{-1}}^2\\
&=\|\boldsymbol{A}\boldsymbol{x}_{k-1}-\boldsymbol{b}\|_{\boldsymbol{M}^{-1}}^2+\gamma_k^2\|\boldsymbol{A}\boldsymbol{V}_k\Delta\boldsymbol{y}_k\|_{\boldsymbol{M}^{-1}}^2+2\gamma_k(\boldsymbol{A}\boldsymbol{x}_{k-1}-b)^\top\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{V}_k\Delta\boldsymbol{y}_k\\
&=\|\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\|_2^2+\gamma_k^2\|\boldsymbol{A}\boldsymbol{V}_k\Delta\boldsymbol{y}_k\|_{\boldsymbol{M}^{-1}}^2+2\gamma_k(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)^\top\boldsymbol{B}_k\Delta\boldsymbol{y},
\end{aligned}$$

since

$$(\boldsymbol{A}\boldsymbol{x}_{k-1}-b)^\top\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{V}_k=(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)^\top\boldsymbol{U}_{k+1}^\top\boldsymbol{M}^{-1}\boldsymbol{U}_{k+1}\boldsymbol{B}_k=(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)^\top\boldsymbol{B}_k.$$

Writing the equation $J^{(k)}(\bar{\boldsymbol{y}}_{k-1},\lambda_{k-1})\begin{pmatrix}\Delta\boldsymbol{y}_k\\ \Delta\lambda_k\end{pmatrix}=-F^{(k)}(\bar{\boldsymbol{y}}_{k-1},\lambda_{k-1})$ in the matrix form

$$\begin{pmatrix}\lambda_{k-1}\boldsymbol{B}_k^\top\boldsymbol{B}_k+\boldsymbol{I} & \boldsymbol{B}_k^\top(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)\\ (\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)^\top\boldsymbol{B}_k & 0\end{pmatrix}\begin{pmatrix}\Delta\boldsymbol{y}_k\\ \Delta\lambda_k\end{pmatrix}=-\begin{pmatrix}\lambda_{k-1}\boldsymbol{B}_k^\top(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)+\bar{\boldsymbol{y}}_{k-1}\\ \frac{1}{2}\|\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\|_2^2-\frac{\tau m}{2}\end{pmatrix}$$

and using $\|\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\|_2\geq\sqrt{\tau m}$, we get from the second equility of the above equation that

$$(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1)^\top\boldsymbol{B}_k\Delta\boldsymbol{y}=-\frac{1}{2}\left(\|\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\|_2^2-\tau m\right)\leq 0.$$

Since $\gamma_k\leq 1$, we get

$$\begin{aligned}
\|\boldsymbol{A}\boldsymbol{x}_k-\boldsymbol{b}\|_{\boldsymbol{M}^{-1}}^2&\geq\|\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\|_2^2+\gamma_k^2\|\boldsymbol{A}\boldsymbol{V}_k\Delta\boldsymbol{y}_k\|_{\boldsymbol{M}^{-1}}^2-\left(\|\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\|_2^2-\tau m\right)\\
&=\tau m+\gamma_k^2\|\boldsymbol{A}\boldsymbol{V}_k\Delta\boldsymbol{y}_k\|_{\boldsymbol{M}^{-1}}^2\geq\tau m.
\end{aligned}$$

Therefore, we prove (4.5).

To obtain the lower bound in (4.4), we investigate two cases: $k<k_t$ and $k\geq k_t$. **Case 1:** $k<k_t$**.** For this case, we have

$$\|\boldsymbol{B}_k^\top\left(\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\right)\|_2\geq\sigma_{\min}(\boldsymbol{B}_k)\|\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1}-\beta_1 e_1\|_2\geq\sigma_{\min}(\underline{\boldsymbol{B}}_{k_t})\sqrt{\tau m}>0,$$

where $\sigma_{\min}(\cdot)$ is the smallest singular value of a matrix, and

$$\underline{\boldsymbol{B}}_{k_t} = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_{k_t} & \alpha_{k_t} \end{pmatrix}.$$

**Case 2:** $k \geq k_t$**.** For this case, we can write $\boldsymbol{x}_{k-1}$ as $\boldsymbol{x}_{k-1} = \boldsymbol{V}_{k_t}\bar{\boldsymbol{y}}_{k-1}$. Remember that $\bar{\boldsymbol{y}}_{k-1} = \boldsymbol{y}_{k-1}$ if $k > k_t$. We first prove $\|\boldsymbol{B}_{k_t}^\top(\boldsymbol{B}_{k_t}\bar{\boldsymbol{y}}_{k-1} - \beta_1 e_1)\|_2 \neq 0$. If it is not true, then $\bar{\boldsymbol{y}}_{k-1} = \operatorname{argmin}_{\boldsymbol{y}} \|\boldsymbol{B}_{k_t}\boldsymbol{y} - \beta_1 e_1\|_2$. By Lemma 4.2, $\boldsymbol{x}_{k-1}$ is the solution to (4.3). Thus, it must hold that $\|\boldsymbol{A}\boldsymbol{x}_{k-1} - \boldsymbol{b}\|_{\boldsymbol{M}^{-1}} < \sqrt{\tau m}$ by Assumption 2.1, in contradiction with (4.5). Now suppose the lower bound in (4.4) is not true. Then there exists a subsequence $\{\bar{\boldsymbol{y}}_{k_j-1}\}$ with $k_j \geq k_t$ such that

$$\lim_{j\to\infty} \|\boldsymbol{B}_{k_t}^\top(\boldsymbol{B}_{k_t}\bar{\boldsymbol{y}}_{k_j-1} - \beta_1 e_1)\|_2 = 0.$$

By Lemma 4.1, we have

$$\lim_{j\to\infty} \bar{\boldsymbol{y}}_{k_j-1} = \boldsymbol{y}_{\min} := \operatorname*{argmin}_{\boldsymbol{y}} \|\boldsymbol{B}_{k_t}\boldsymbol{y} - \beta_1 e_1\|_2,$$

leading to

$$\lim_{j\to\infty} \boldsymbol{x}_{k_j-1} = \lim_{k_j\to\infty} \boldsymbol{V}_{k_t}\bar{\boldsymbol{y}}_{k_j-1} = \boldsymbol{V}_{k_t}\boldsymbol{y}_{\min}.$$

It follows from Lemma 4.2 that $\boldsymbol{V}_{k_t}\boldsymbol{y}_{\min} = \boldsymbol{x}_{\min}$, which is the solution to (4.3). Therefore, it must hold

$$\lim_{j\to\infty} \|\boldsymbol{A}\boldsymbol{x}_{k_j-1} - \boldsymbol{b}\|_{\boldsymbol{M}^{-1}} = \|\boldsymbol{A}\boldsymbol{x}_{\min} - \boldsymbol{b}\|_{\boldsymbol{M}^{-1}} < \sqrt{\tau m}$$

by Assumption 2.1, which is in contradiction with (4.5). Summarizing both the two cases, the desired result is proved. □

**Lemma 4.4** *The points* $\{(\boldsymbol{x}_k, \lambda_k)\}_{i=0}^{\infty}$ *generated by the* PNT *algorithm lie in a bounded set of* $\mathbb{R}^n \times \mathbb{R}^+$.

**Proof.** We only need to prove $\{(\boldsymbol{x}_k, \lambda_k)\}_{k\geq k_t}$ is bounded above. In this case, recall that $\boldsymbol{x}_k = \boldsymbol{V}_{k_t}\boldsymbol{y}_k$ and

$$\begin{aligned} h(\boldsymbol{x}_k, \lambda_k) &= \frac{1}{2}\left(\|\lambda_k \boldsymbol{A}^\top \boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}) + \boldsymbol{N}^{-1}\boldsymbol{x}_k\|_{\boldsymbol{N}}^2 + \left(\frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}\|_{\boldsymbol{M}^{-1}}^2 - \frac{\tau m}{2}\right)^2\right) \\ &= \frac{1}{2}\left(\|\lambda_k \boldsymbol{A}^\top \boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}) + \boldsymbol{N}^{-1}\boldsymbol{x}_k\|_{\boldsymbol{N}}^2 + \left(\frac{1}{2}\|\boldsymbol{B}_{k_t}\boldsymbol{y}_k - \beta_1 e_1\|_2^2 - \frac{\tau m}{2}\right)^2\right). \end{aligned}$$

Notice that $h(\boldsymbol{x}_0, \lambda_0) \geq h(\boldsymbol{x}_1, \lambda_1) \geq \cdots$. If the points do not lie in a bounded set, there exists a subsequence $\{(\boldsymbol{x}_{k_j}, \lambda_{k_j})\}$ with $k_j \geq k_t$ such that $(\boldsymbol{x}_{k_j}, \lambda_{k_j}) \to \infty$. If $\|\boldsymbol{x}_{k_j}\|_2 \to \infty$, then $\|\boldsymbol{y}_{k_j}\|_2 \to \infty$, since $\boldsymbol{x}_{k_j} = \boldsymbol{V}_{k_t}\boldsymbol{y}_{k_j}$ and $\boldsymbol{V}_{k_t}$ has full column rank. This leads to $\|\boldsymbol{B}_{k_t}\boldsymbol{y}_{k_j}\|_2 \to \infty$ since $\boldsymbol{B}_{k_t}$ has full column rank. If follows that the second term of $h(\boldsymbol{x}_{k_j}, \lambda_{k_j})$ tends to infinity and $h(\boldsymbol{x}_{k_j}, \lambda_{k_j}) \to \infty$, a contradiction. Therefore, it must hold that $\|\boldsymbol{x}_{k_j}\|_2$ is bounded above and $\lambda_{k_j} \to \infty$. By Lemma 4.3, we have $\|\lambda_{k_j}\boldsymbol{A}^\top \boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k_j} - \boldsymbol{b})\|_{\boldsymbol{N}} \geq \lambda_{k_j} C_1$. Notice that $\{\boldsymbol{N}^{-1}\boldsymbol{x}_{k_j}\}$ lie in a bounded set. It follows that $\|\lambda_{k_j}\boldsymbol{A}^\top \boldsymbol{M}^{-1}(\boldsymbol{A}\boldsymbol{x}_{k_j} - \boldsymbol{b}) + \boldsymbol{N}^{-1}\boldsymbol{x}_{k_j}\|_{\boldsymbol{N}} \to \infty$ and $h(\boldsymbol{x}_{k_j}, \lambda_{k_j}) \to \infty$, also a contradiction. □

**Lemma 4.5** *There exist a positive constant* $C_2 < +\infty$ *such that for any* $k \geq 1$*, it holds*

$$\|J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})^{-1}\|_2 \leq C_2. \tag{4.6}$$

**Proof.** First, we prove that $J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})$ is always nonsingular. Write it as

$$J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1}) = \begin{pmatrix} \lambda_{k-1}\boldsymbol{B}_{k_t}^\top \boldsymbol{B}_k + \boldsymbol{I} & \boldsymbol{B}_k^\top(\boldsymbol{B}_{k_t}\bar{\boldsymbol{y}}_{k-1} - \beta_1 e_1) \\ (\boldsymbol{B}_k\bar{\boldsymbol{y}}_{k-1} - \beta_1 e_1)^\top \boldsymbol{B}_k & 0 \end{pmatrix} =: \begin{pmatrix} \boldsymbol{C}_k & \boldsymbol{d}_k \\ \boldsymbol{d}_k^\top & 0 \end{pmatrix}$$

and notice that

$$\begin{pmatrix} \boldsymbol{C}_k & \boldsymbol{d}_k \\ \boldsymbol{d}_k^\top & 0 \end{pmatrix} = \begin{pmatrix} \boldsymbol{I} & \boldsymbol{0} \\ -\boldsymbol{d}_k^\top \boldsymbol{C}_k^{-1} & 1 \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{C}_k & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{d}_k^\top \boldsymbol{C}_k^{-1}\boldsymbol{d}_k \end{pmatrix} \begin{pmatrix} \boldsymbol{I} & -\boldsymbol{C}_k^{-1}\boldsymbol{d}_k \\ \boldsymbol{0} & 1 \end{pmatrix}^{-1}.$$

It follows that

$$\begin{pmatrix} \boldsymbol{C}_k & \boldsymbol{d}_k \\ \boldsymbol{d}_k^\top & 0 \end{pmatrix}^{-1} = \begin{pmatrix} \boldsymbol{I} & -\boldsymbol{C}_k^{-1}\boldsymbol{d}_k \\ \boldsymbol{0} & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{C}_k^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & -(\boldsymbol{d}_k^\top \boldsymbol{C}_k^{-1}\boldsymbol{d}_k)^{-1} \end{pmatrix} \begin{pmatrix} \boldsymbol{I} & \boldsymbol{0} \\ -\boldsymbol{d}_k^\top \boldsymbol{C}_k^{-1} & 1 \end{pmatrix}, \tag{4.7}$$

Since $\boldsymbol{C}_k$ is positive definite and $\|\boldsymbol{d}_k\|_2 \geq C_1 > 0$.

To give an upper bound on $\|J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})^{-1}\|_2$, we only need to consider $k \geq k_t$, where $\boldsymbol{B}_k = \boldsymbol{B}_{k_t}$ in $J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})$. Since $\sigma_{\min}(\boldsymbol{C}_k) = \sigma_{\min}(\lambda_{k-1}\boldsymbol{B}_{k_t}^\top \boldsymbol{B}_{k_t} + \boldsymbol{I}) \geq 1$, we have $\|\boldsymbol{C}_k^{-1}\|_2 \leq 1$. By Lemma 4.4, there exist a positive constant $C_3 < +\infty$ such that $\lambda_k \leq C_3$, thereby

$$\sigma_{\max}(\boldsymbol{C}_k) \leq \sigma_{\max}(\lambda_{k-1}\boldsymbol{B}_{k_t}^\top \boldsymbol{B}_{k_t}) + \sigma_{\max}(\boldsymbol{I}) \leq C_3\sigma_{\max}(\boldsymbol{B}_{k_t}^\top \boldsymbol{B}_{k_t}) + 1 =: \bar{C}_3.$$

By Lemma 4.3 we have $\|\boldsymbol{d}_k\|_2 = \|\boldsymbol{B}_{k_t}^\top(\boldsymbol{B}_{k_t}\bar{\boldsymbol{y}}_{k-1} - \beta_1 e_1)\|_2 \geq C_1$. On the other hand, by Lemma 4.4 we know that $\|\boldsymbol{x}_{k-1}\|_2 = \|\boldsymbol{V}_{k_t}\bar{\boldsymbol{y}}_{k-1}\|_2$ is bounded above, thereby $\|\bar{\boldsymbol{y}}_{k-1}\|_2$ is bounded above since $\boldsymbol{V}_{k_t}$ has full column rank. Thus, there exists a positive constant $\bar{C}_1$ such that $\|\boldsymbol{d}_k\|_2 \leq \bar{C}_1$, leading to

$$\|\boldsymbol{C}_k^{-1}\boldsymbol{d}_k\|_2 \leq \|\boldsymbol{C}_k^{-1}\|_2\|\boldsymbol{d}_k\|_2 \leq \bar{C}_1\sigma_{\min}(\boldsymbol{C}_k)^{-1} \leq \bar{C}_1,$$

and

$$\boldsymbol{d}_k^\top \boldsymbol{C}_k^{-1}\boldsymbol{d}_k \geq \sigma_{\min}(\boldsymbol{C}_k^{-1})\|\boldsymbol{d}_k\|_2^2 = \sigma_{\max}(\boldsymbol{C}_k)^{-1}\|\boldsymbol{d}_k\|_2^2 \geq C_1^2/\bar{C}_3 > 0.$$

Therefore, we have

$$\left\| \begin{pmatrix} \boldsymbol{0} & -\boldsymbol{C}_k^{-1}\boldsymbol{d}_k \\ \boldsymbol{0} & 0 \end{pmatrix} \right\|_2^2 = \left\| \begin{pmatrix} \boldsymbol{0} & -\boldsymbol{C}_k^{-1}\boldsymbol{d}_k \\ \boldsymbol{0} & 0 \end{pmatrix}^\top \begin{pmatrix} \boldsymbol{0} & -\boldsymbol{C}_k^{-1}\boldsymbol{d}_k \\ \boldsymbol{0} & 0 \end{pmatrix} \right\|_2 = \left\| \begin{pmatrix} \boldsymbol{0} & \\ & \|\boldsymbol{C}_k^{-1}\boldsymbol{d}_k\|_2^2 \end{pmatrix} \right\|_2 \leq \bar{C}_1^2$$

and

$$\left\| \begin{pmatrix} \boldsymbol{I} & -\boldsymbol{C}_k^{-1}\boldsymbol{d}_k \\ \boldsymbol{0} & 1 \end{pmatrix} \right\|_2 \leq \left\| \begin{pmatrix} \boldsymbol{I} & \\ & 1 \end{pmatrix} \right\|_2 + \left\| \begin{pmatrix} \boldsymbol{0} & -\boldsymbol{C}_k^{-1}\boldsymbol{d}_k \\ \boldsymbol{0} & 0 \end{pmatrix} \right\|_2 \leq 1 + \bar{C}_1.$$

Similarly, we have

$$\left\| \begin{pmatrix} \boldsymbol{C}_k^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & -(\boldsymbol{d}_k^\top \boldsymbol{C}_k^{-1}\boldsymbol{d}_k)^{-1} \end{pmatrix} \right\|_2 \leq \max\{1, \bar{C}_3/C_1^2\}.$$

Using the expression of $J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})$ in (4.7), we finally obtain the desired result. □

**Lemma 4.6** *There exists a positive constant $C_4$ such that for any $k \geq 1$, the step-length satisfies*

$$\gamma_k \geq C_4 > 0. \tag{4.8}$$

**Proof.** By Theorem 3.2 and Theorem 3.1, at each iteration the Armijo backtracking line search must terminate in finite steps with a $\gamma_k$ satisfying

$$\gamma_k \geq \min\left\{1, \frac{4(1-c)\eta h(\boldsymbol{x}_{k-1}, \lambda_{k-1})}{\zeta(\boldsymbol{x}_{k-1}, \lambda_{k-1})\|(\Delta \boldsymbol{x}_k^\top, \Delta\lambda_k)\|_2^2}\right\}, \tag{4.9}$$

where $\zeta(\boldsymbol{x}_{k-1}, \lambda_{k-1})$ is the Lipschitz constant of $\nabla h$ at $(\boldsymbol{x}_{k-1}, \lambda_{k-1})$; see also [4, pp. 122, Corollary 2.1]. Now we prove $\zeta(\boldsymbol{x}_{k-1}, \lambda_{k-1})$ are bounded above. Notice that $\nabla h(\boldsymbol{x}, \lambda) = J(\boldsymbol{x}, \lambda)\widehat{N}F(\boldsymbol{x}, \lambda)$. Thus, all the elements in the Jacobian of $\nabla h(\boldsymbol{x}, \lambda)$ are polynomials of $(\boldsymbol{x}, \lambda)$ with degrees not bigger than 4. Since $\{(\boldsymbol{x}_{k-1}, \lambda_{k-1})\}$ lie in a bounded set, the norms of the Jacobians of $\nabla h(\boldsymbol{x}, \lambda)$ at the points $\{(\boldsymbol{x}_{k-1}, \lambda_{k-1})\}$ are bounded above. Therefore, the Lipschitz constants $\zeta(\boldsymbol{x}_{k-1}, \lambda_{k-1})$ are bounded above.

Let $\zeta(\boldsymbol{x}_{k-1}, \lambda_{k-1}) \leq \zeta_0$ with $0 < \zeta_0 < +\infty$ for any $k \geq 1$. Then by Lemma 4.5 and Lemma 3.3, we have

$$\begin{aligned}
\left\|\begin{pmatrix}\Delta \boldsymbol{x}_k \\ \Delta\lambda_k\end{pmatrix}\right\|_2 &\leq \left\|\begin{pmatrix}\boldsymbol{V}_k & \\ & 1\end{pmatrix}\right\|_2 \left\|\begin{pmatrix}\Delta \boldsymbol{y}_k \\ \Delta\lambda_k\end{pmatrix}\right\|_2 \\
&\leq (\|\boldsymbol{V}_{k_t}\|_2 + 1)\|J^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})^{-1}\|_2 \|F^{(k)}(\bar{\boldsymbol{y}}_{k-1}, \lambda_{k-1})\|_2 \\
&\leq C_2(\|\boldsymbol{V}_{k_t}\|_2 + 1)\|F(\boldsymbol{x}_{k-1}, \lambda_{k-1})\|_{\widehat{\boldsymbol{N}}} \\
&= C_2(\|\boldsymbol{V}_{k_t}\|_2 + 1)(2h(\boldsymbol{x}_{k-1}, \lambda_{k-1}))^{1/2}.
\end{aligned}$$

Then we obtain

$$\gamma_k \geq \min\left\{1, \frac{4(1-c)\eta h(\boldsymbol{x}_{k-1}, \lambda_{k-1})}{\zeta_0\|(\Delta \boldsymbol{x}_k^\top, \Delta\lambda_k)^\top\|_2^2}\right\} \geq \min\left\{1, \frac{2(1-c)\eta}{\zeta_0 C_2^2(\|\boldsymbol{V}_{k_t}\|_2 + 1)^2}\right\} =: C_4.$$

The desired result is obtained. □

Now we can now give the proof of Theorem 4.1.

**Proof of Theorem 4.1.** By Lemma 4.4, the sequence $\{(\boldsymbol{x}_k, \lambda_k)\}_{k=1}^\infty$ is contained in a bounded set, thereby there exists a convergent subsequence $\{(\boldsymbol{x}_{k_j}, \lambda_{k_j})\}_{j=1}^\infty$. Suppose $(\boldsymbol{x}_{k_j}, \lambda_{k_j}) \to (\widehat{\boldsymbol{x}}, \widehat{\lambda})$. Then it follows that $h(\boldsymbol{x}_{k_j}, \lambda_{k_j}) \to h(\widehat{\boldsymbol{x}}, \widehat{\lambda})$ since $h(\boldsymbol{x}, \lambda)$ is continuous. Note that $h(\boldsymbol{x}_{k_j}, \lambda_{k_j})$ is nonincreasing, thereby $h(\widehat{\boldsymbol{x}}, \widehat{\lambda}) \leq h(\boldsymbol{x}_{k_j}, \lambda_{k_j})$ for any $k_j$. Thus, for any $\varepsilon > 0$, there exist a $k_\star \in \mathbb{N}$ such that

$$h(\boldsymbol{x}_{k_j}, \lambda_{k_j}) < h(\widehat{\boldsymbol{x}}, \widehat{\lambda}) + \varepsilon, \quad k_j > k_\star.$$

Select one $k_j$ that satisfies $k_j > k_\star$. Then for any $k \geq k_j$, it holds that

$$h(\boldsymbol{x}_k, \lambda_k) \leq h(\boldsymbol{x}_{k_j}, \lambda_{k_j}) < h(\widehat{\boldsymbol{x}}, \widehat{\lambda}) + \varepsilon,$$

which means that

$$\lim_{k\to\infty} h(\boldsymbol{x}_k, \lambda_k) = h(\widehat{\boldsymbol{x}}, \widehat{\lambda}).$$

By the Armijo condition and Theorem 3.1, we have

$$h(\boldsymbol{x}_{k+1}, \lambda_{k+1}) - h(\boldsymbol{x}_k, \lambda_k) \leq c\gamma_k \left(\Delta \boldsymbol{x}_k^\top, \Delta\lambda_k\right) \nabla h(\boldsymbol{x}_k, \lambda_k) \leq 0.$$

Taking the limit on both sides leads to

$$\lim_{k\to\infty} c\gamma_k \left(\Delta \boldsymbol{x}_k^\top, \Delta\lambda_k\right) \nabla h(\boldsymbol{x}_k, \lambda_k) = 0$$

By Lemma 4.6 we get $\lim_{k\to\infty} \left(\Delta \boldsymbol{x}_k^\top, \Delta\lambda_k\right) \nabla h(\boldsymbol{x}_k, \lambda_k) = 0$. Noticing by Theorem 3.1 that $-2h(\boldsymbol{x}_k, \lambda_k) = \left(\Delta \boldsymbol{x}_k^\top, \Delta\lambda_k\right) \nabla h(\boldsymbol{x}_k, \lambda_k)$, we obtain

$$h(\widehat{\boldsymbol{x}}, \widehat{\lambda}) = \lim_{k\to\infty} h(\boldsymbol{x}_k, \lambda_k) = 0. \tag{4.10}$$

This proves the desired result. □

Now we can give the convergence result of $(\boldsymbol{x}_k, \lambda_k)$ with respect to the true solution $(\boldsymbol{x}^*, \lambda^*)$.

**Corollary 4.1** *The sequence $\{(\boldsymbol{x}_k, \lambda_k)\}_{k=0}^{\infty}$ generated by the PNT algorithm eventually converges to $(\boldsymbol{x}^*, \lambda^*)$, i.e. the solution of (2.3) and the corresponding Lagrange multiplier.*

**Proof.** Using the same notations as the proof of Theorem 4.1, we obtain from (4.10) that $(\widehat{\boldsymbol{x}}, \widehat{\lambda}) = (\boldsymbol{x}^*, \lambda^*)$, since $h(\boldsymbol{x}, \lambda)$ has the unique zero point $(\boldsymbol{x}^*, \lambda^*)$. Therefore, the subsequence $\{(\boldsymbol{x}_{k_j}, \lambda_{k_j})\}_{j=1}^{\infty}$ defined in the proof of Theorem 4.1 converges to $(\boldsymbol{x}^*, \lambda^*)$. Now we need to prove the whole sequence $\{(\boldsymbol{x}_k, \lambda_k)\}_{k=1}^{\infty}$ converges to $(\boldsymbol{x}^*, \lambda^*)$. Assume that there is a subsequence $\{(\boldsymbol{x}_{l_j}, \lambda_{l_j})\}_{j=1}^{\infty}$ that does not converge to $(\boldsymbol{x}^*, \lambda^*)$. We can select a subsequence from $\{(\boldsymbol{x}_{l_j}, \lambda_{l_j})\}_{j=1}^{\infty}$ that converges to a point $(\bar{\boldsymbol{x}}, \bar{\lambda}) \neq (\boldsymbol{x}^*, \lambda^*)$. Since $h(\boldsymbol{x}_{l_j}, \lambda_{l_j})$ is nonincreasing with respect to $j$, using the same procedure as the proof of Theorem 4.1, we can obtain again that $h(\bar{\boldsymbol{x}}, \bar{\lambda}) = 0$, leading to $(\bar{\boldsymbol{x}}, \bar{\lambda}) = (\boldsymbol{x}^*, \lambda^*)$, a contradiction. Therefore, any subsequence of $\{(\boldsymbol{x}_k, \lambda_k)\}_{k=0}^{\infty}$ converges to $(\boldsymbol{x}^*, \lambda^*)$, thereby $\{(\boldsymbol{x}_k, \lambda_k)\}_{k=0}^{\infty}$ converges to $(\boldsymbol{x}^*, \lambda^*)$. □

# 5 Experimental results

We test the PNT method and compare it with the standard Newton method. These two methods use the same initialization and backtracking line search strategy. The setting of hyperparameters follows Algorithm 3, and we set $\tau = 1.001$ and $\lambda_0 = 0.1$ in all the experiments. We also implement the generalized hybrid iterative method proposed in [12] (denoted by genHyb), which is also based on gen-GKB. The genHyb iteratively computes approximations to $\mu_{opt}$ and $\boldsymbol{x}_{opt} = \boldsymbol{x}(\mu_{opt})$, where $\mu_{opt}$ is the optimal Tikhonov regularization parameter, that is $\mu_{opt} = \min_{\mu>0} \|\boldsymbol{x}(\mu) - \boldsymbol{x}_{\text{true}}\|_2$; the $k$-th corresponding Lagrangian multiplier is $\lambda_k = 1/\mu_k$. All the experiments are performed on MATLAB R2023b. The codes are available at https://github.com/Machealb/InverProb_IterSolver,

## 5.1 Small-scale problems

We choose two small-scale 1D inverse problems from [24]. The first problem is heat, an inverse heat equation described by the Volterra integral equation of the first kind on $[0, 1]$ with the kernel

$$k(s,t) = \phi(s-t), \quad \phi(t) = \frac{t^{-3/2}}{2\sqrt{\pi}} \exp\left(-\frac{1}{4t}\right).$$

The second problem is shaw, a one-dimensional image restoration model described by the Fredholm integral equation of the first kind on $[-\pi/2, \pi/2]$ with the kernel

$$k(s,t) = (\cos s + \cos t)^2 \left(\frac{\sin u}{u}\right)^2, \quad u = \pi(\sin s + \sin t).$$

We use the code in [24] to discretize the two problems to generate $\boldsymbol{A}$, $\boldsymbol{x}_{\text{true}}$ and $\boldsymbol{b}_{\text{true}} = \boldsymbol{A}\boldsymbol{x}_{\text{true}}$, where $m = n = 2000$ and $m = n = 3000$ for heat and shaw, respectively. We set the noisy observation vector $\boldsymbol{b}$ as $\boldsymbol{b} = \boldsymbol{b}_{\text{true}} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is a Gaussian noise. For heat, we set $\boldsymbol{\epsilon}$ as a white noise (i.e. $\boldsymbol{M}$ is a scalar matrix) with noise level $\varepsilon := \|\boldsymbol{\epsilon}\|_2/\|\boldsymbol{b}_{\text{true}}\|_2 = 5 \times 10^{-2}$; for shaw, we set $\boldsymbol{\epsilon}$ as a uncorrelated non-white noise (i.e. $\boldsymbol{M}$ is a diagonal matrix) with noise level $\varepsilon = 10^{-2}$. The true solutions and noisy observed data for these two problems are shown in Figure 5.1.

For heat, we assume a Gaussian prior $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{0}, \lambda^{-1}\boldsymbol{N})$ with $\boldsymbol{N}$ coming from the Gaussian kernel $\kappa_G$, i.e. the $ij$ element of $\boldsymbol{N}$ is

$$[\boldsymbol{N}]_{ij} = K_G(r_{ij}), \quad K_G(r) := \exp\left(-r^2/(2l^2)\right),$$

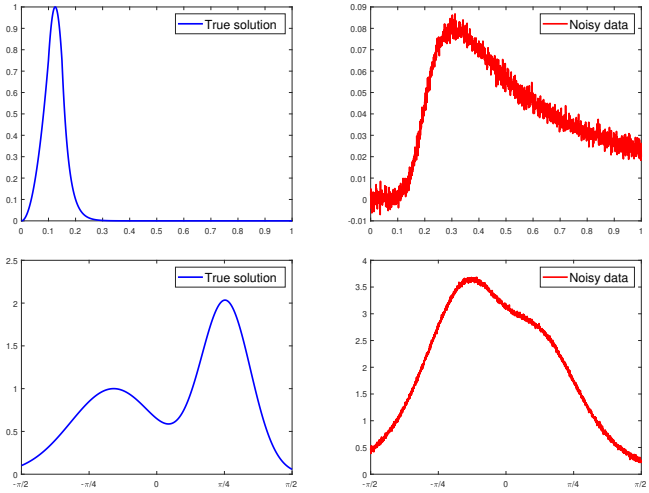


Figure 5.1: True solution and noisy observed data. Top: heat. Bottom: shaw.

where $r_{ij} = \|\boldsymbol{p}_i - \boldsymbol{p}_j\|_2$ and $\{\boldsymbol{p}_i\}_{i=1}^n$ are discretized points of the domain of the true solution; the parameter $l$ is set as $l = 0.1$. For shaw, we construct $\boldsymbol{N}$ using the exponential kernel

$$K_{exp}(r) := \exp\left(-(r/l)^\nu\right),$$

where the parameters $l$ and $\nu$ are set as $l = 0.1$ and $\nu = 1$. We set $\tau = 1.001$ for both the two problems. We factorize $\boldsymbol{M}^{-1}$ and $\boldsymbol{N}^{-1}$ to form (1.3) and solve it directly to find $\mu_{opt}$ and $\boldsymbol{x}_{opt}$; the corresponding Lagrangian multiplier is $\lambda_{opt} = 1/\mu_{opt}$. We also compute the $\mu$ of (2.2) and the corresponding regularized solution, which is denoted by $\mu_{DP}$ and $\boldsymbol{x}_{DP}$, respectively. Therefore, the solution to (2.3) and (2.4) is $(\boldsymbol{x}^*, \lambda^*) = (\boldsymbol{x}_{DP}, 1/\mu_{DP})$. We use the optimal Tikhonov solution and the DP solution as the baseline for the subsequent tests.

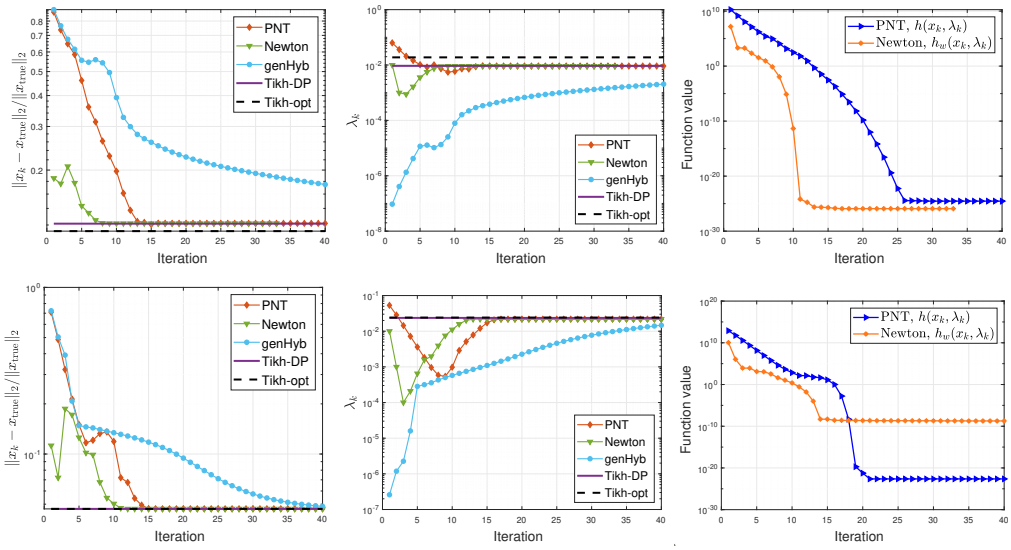


Figure 5.2: Relative errors of iterative solutions, convergence of $\lambda_k$, and convergence of merit functions. Top: heat. Bottom: shaw.

We compare the convergence behavior of PNT, Newton and genHyb methods by plotting the relative error curve of $\boldsymbol{x}_k$ with respect to $\boldsymbol{x}_{\text{true}}$ and the convergence curves of $\lambda_k$ and

merit functions. The relative errors and the $\lambda$ of $(\boldsymbol{x}_{DP}, 1/\mu_{DP})$ and $(\boldsymbol{x}_{opt}, 1/\mu_{opt})$ are used as baselines. From Figure 5.2 we find that both PNT and Newton methods converge very fast to $\boldsymbol{x}_{DP}$ and $\lambda_{DP} := 1/\mu_{DP}$ with very few iterations, and PNT converges only slightly slower than Newton. For heat, the error of the DP solution is slightly higher than the optimal Tikhonov solution, because DP slightly under-estimates $\lambda$. The merit functions of both PNT and Newton decrease monotonically, and $h(\boldsymbol{x}_k, \lambda_k)$ of PNT eventually decreases to an extremely small value for the two problems. We remark that we set $w = 1$ for $h_w(\boldsymbol{x}, \lambda)$ in all the tests. For Newton method for heat, we stop the iterate at $k = 34$ because the step-length $\gamma_k$ is too small. In comparison, the genHyb method converges much slower than the previous two methods, especially for heat.

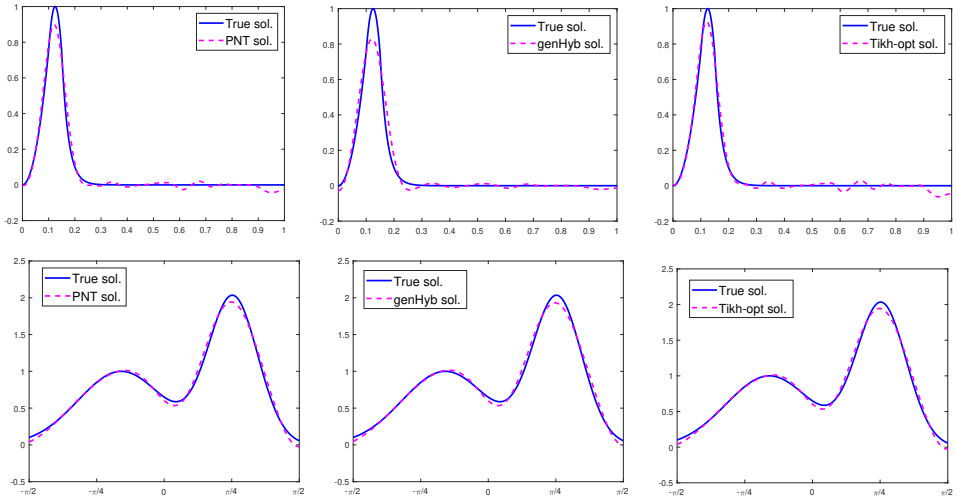


Figure 5.3: Comparision of reconstructed solutions at the final iterations with the optimal Tikhonov regularized solution. Top: heat. Bottom: shaw.

Figure 5.3 plots the recovered solutions computed by PNT and genHyb methods at the final iterations; the solution by Newton is almost the same as that by PNT, thereby we omit it. We also plot the optimal Tikhonov regularized solution as a comparison, where the DP solution is very similar and omitted. We find that both PNT and genHyb can recover good regularized solutions, and PNT is slightly better for heat.

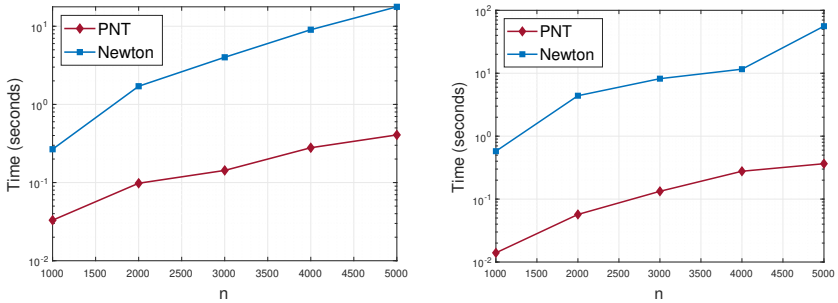


Figure 5.4: Comparision of scalability of PNT and Newton methods as the scale of the problems increasing from $n = 1000$ to $n = 5000$. Left: heat. Right: shaw.

To show the advantage of the computational efficiency of PNT over Newton, we gradually increase the scale of the test problems and measure the running time of the two methods, where both the algorithms stop at the first iteration such that $\left| \|\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} - \tau m \right| \leq$

Table 5.1: Running time (measured in seconds) of PNT and Newton methods as the scale of the problems increasing from $n = 1000$ to $n = 5000$. Both the two methods stop at the first $k$ (in parentheses) such that $\left|\|\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}\|^2_{\boldsymbol{M}^{-1}} - \tau m\right| \leq 10^{-8}$.

| $n$ | 1000 | 2000 | 3000 | 4000 | 5000 |
|---|---|---|---|---|---|
| | | | heat | | |
| PNT | 0.033 (18) | 0.098 (21) | 0.143 (19) | 0.279 (19) | 0.407 (19) |
| Newton | 0.267 (10) | 1.708 (11) | 4.007 (10) | 9.027 (10) | 17.715 (11) |
| **ratio** | **8.1** | **17.4** | **28.0** | **32.4** | **43.5** |
| | | | shaw | | |
| PNT | 0.014 (17) | 0.057 (16) | 0.133 (19) | 0.277 (18) | 0.356 (16) |
| Newton | 0.577 (17) | 4.349 (17) | 8.196 (16) | 11.600 (13) | 55.919 (16) |
| **ratio** | **41.2** | **77.9** | **61.6** | **41.9** | **157.1** |

$10^{-8}$. The time data are listed in Table 5.1. We also compute the ratio of the running time, i.e. the value of Newton-time/PNT-time. For shaw, we find that both PNT and Newton stop with similar iteration numbers, and the computational speed of PNT is much faster than Newton, with the speedup ratio varying from 41 to 157. For heat, we find that Newton stops with only about half iteration numbers of PNT's. However, the total running time of PNT is still much smaller than Newton's, with the speedup ratio varying from 8 to 43. To compare the scalability of PNT and Newton more clearly, we use the data in Table 5.1 to plot the curve of time growth with respect to $n$. Clearly, PNT saves much more time compared to Newton while obtaining solutions with the same accuracy.

## 5.2 Large-scale problems

We choose three 2D image deblurring and computed tomography inverse problems from [17]. The first problem is PRblurshake, which simulates a spatially invariant motion blur caused by the shaking of a camera. The second problem is PRblurspeckle, which simulates a spatially invariant blur caused by atmospheric turbulence. The third problem is PRspherical that models spherical means tomography. The true images and noisy observed data are shown in Figure 5.5, where all the images have $128 \times 128$ pixels, and $\boldsymbol{\epsilon}$ are uncorrelated non-white Gaussian noises with $\varepsilon = 10^{-3}$, $5 \times 10^{-3}$ and $10^{-2}$, respectively. Therefore, we have $m = n = 128^2$ for all the three problems.

For PRblurshake and PRblurspeckle, we construct $\boldsymbol{N}$ using the Gaussian kernel with $l = 5.0$ and $l = 1.0$, respectively. For PRspherical, we construct $\boldsymbol{N}$ using the Matérn kernel

$$K_M(r) := \frac{2^{1-\nu}}{\Gamma(\nu)} \left( \frac{\sqrt{2\nu} r}{l} \right)^{\nu} B_\nu \left( \frac{\sqrt{2\nu} r}{l} \right),$$

where $\Gamma(\cdot)$ is the gamma function, $B_\nu(\cdot)$ is the modified Bessel function of the second kind, and $l$ and $\nu$ are two positive parameters of the covariance; we set $l = 100$ and $\nu = 1.5$.

For the three large-scale problems, it is almost impossible to get $(\mu_{opt}, \boldsymbol{x}(\mu_{opt}))$ and $(\mu_{DP}, \boldsymbol{x}(\mu_{DP}))$ by solving (1.3). The standard Newton method and the methods in [13, 14] can not be applied because these methods have to deal with $\boldsymbol{N}^{-1}$. To test the performance of PNT, here we only compare it with genHyb for convergence behavior and accuracy of the regularized solutions.

The relative error curves of the two methods, the convergence curves of $\lambda_k$ and $h(\boldsymbol{x}_k, \lambda_k)$ are plotted in Figure 5.6. It can be observed that PNT for both the three problems converge

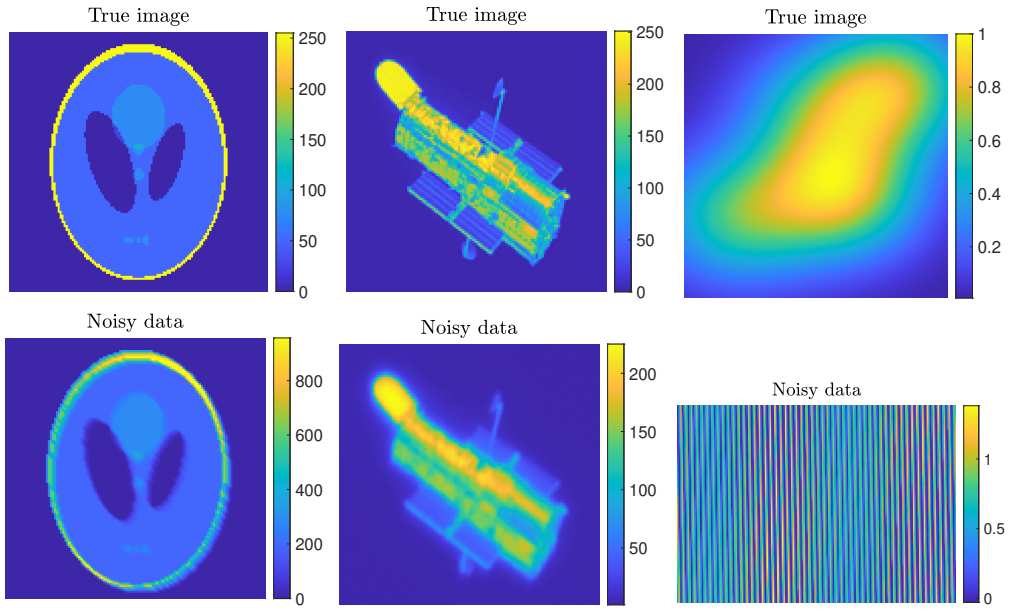


Figure 5.5: True solution and noisy observed data for deblurring and tomography problems. From the leftmost column to the rightmost column are PRblurshake, PRblurspeckle and PRspherical.

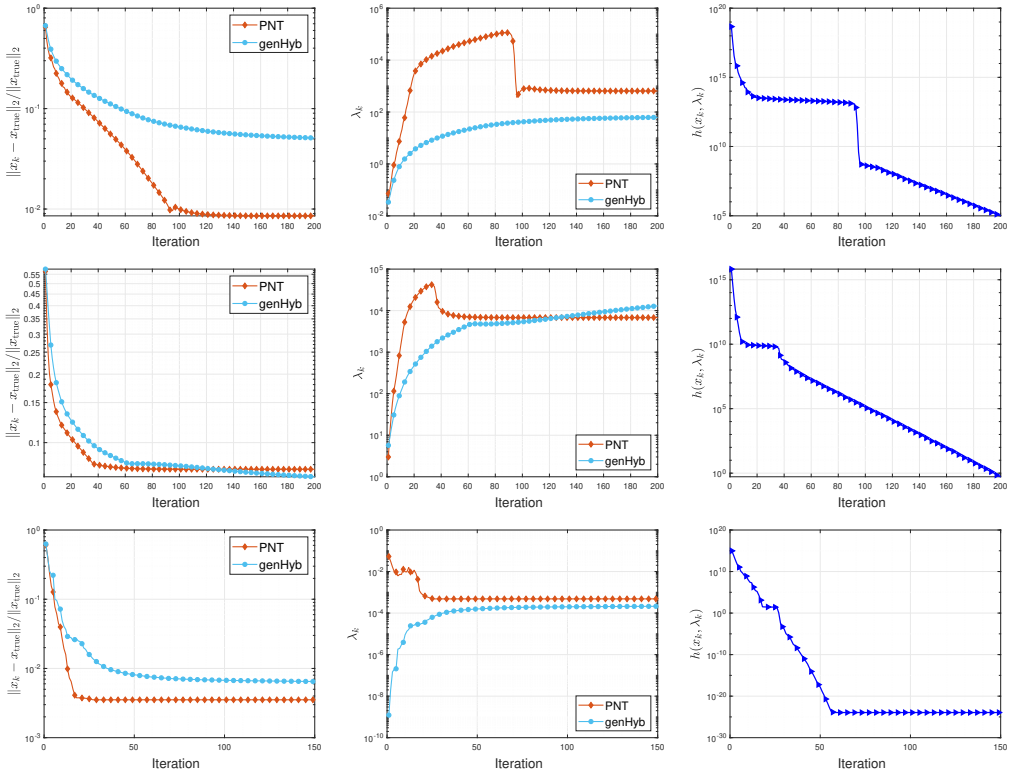


Figure 5.6: Relative errors of iterative solutions, convergence of $\lambda_k$, and convergence of merit functions. Top: PRblurshake. Middle: PRblurspeckle. Bottom: PRspherical.

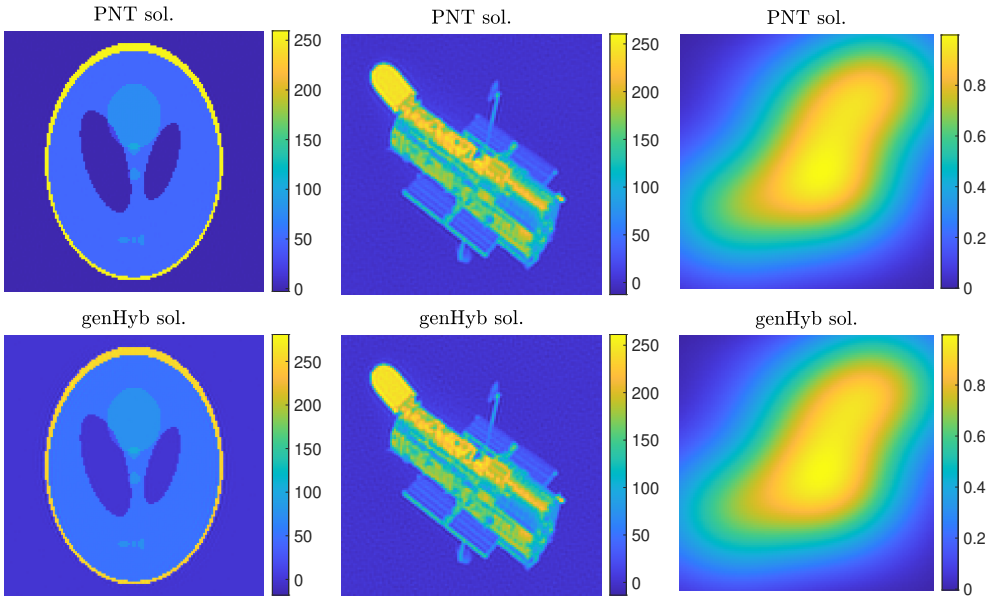


Figure 5.7: Reconstructed solutions at the final iterations by PNT and genHyb. From the leftmost column to the rightmost column are PRblurshake, PRblurspeckle and PRspherical.

very fast: the variations of relative error and $\lambda_k$ become quickly stabilized after 50 to 150 iterations, although for the first two problems $h(\boldsymbol{x}_k, \lambda_k)$ are still decreasing significantly after 200 iterations. The genHyb method for PRblurshake converges much slower, and after 200 iterations it only obtains a solution with a much larger relative error than that of PNT, this is because genHyb significantly under-estimates $\lambda$ than PNT. For PRblurspeckle, genHyb computes a regularized solution with accuracy slightly better than PNT, while for PRspherical, the accuracy of PNT is slightly better. The reconstructed images are shown in Figure 5.7, which reveals the effectiveness of both the two methods.

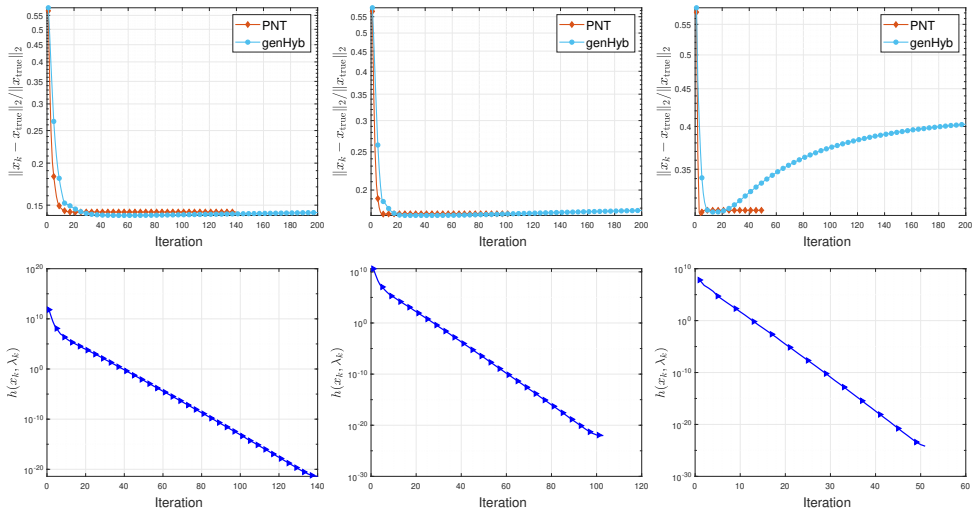


Figure 5.8: Relative errors of iterative solutions by PNT and genHyb, and the decrease of $h(\boldsymbol{x}_k, \lambda_k)$. The test problem is PRblurspeckle. From the leftmost column to the rightmost column, the noise levels are $\varepsilon = 5 \times 10^{-2},\ 10^{-1},\ 5 \times 10^{-1}$.

To further test the robustness of PNT and genHyb as the noise level gradually increases, we set the noise level of PRblurspeckle to be $\varepsilon = 5 \times 10^{-2},\ 10^{-1},\ 5 \times 10^{-1}$. Figure 5.8 shows the corresponding relative error curves and the curves of $h(\boldsymbol{x}_k, \lambda_k)$. We can find that, when the noise is not very big, both PNT and genHyb converge stably with almost the same accuracy. However, when the noise gradually increases, the situations are very different. First, we find that as the noise increases, PNT still converges stably, and faster. Second, $h(\boldsymbol{x}_k, \lambda_k)$ can always decrease to an extremely small value, which is promised by Theorem 4.1. The iterate of PNT stops when the step-length $\gamma_k$ becomes too small (less than $10^{-16}$), which happens more early if the noise is bigger. In comparison, the convergence of genHyb becomes unstable as the noise increases. For $\varepsilon = 10^{-1}$, it can be observed that the relative error for genHyb slightly increases after a certain iteration, while for $\varepsilon = 5\times10^{-1}$, the increase of relative error happens more early and more clearly. This is a common potential flaw for hybrid regularization methods, which is overcome by the PNT method.

# 6 Conclusion

We have proposed the projected Newton method (PNT) as a novel iterative approach for simultaneously updating both the regularization parameter and solution without any computationally expensive matrix inversions or decompositions. By reformulating the Tikhonov regularization as a corresponding constrained minimization problem and leveraging its Lagrangian function, the regularized solution and the corresponding Lagrangian multiplier can be obtained from the unconstrained Lagrangian function using a Newton-type method. To reduce the computational overhead of the Newton method, the generalized Golub-Kahan bidiagonalization is applied to project the original large-scale problem to become small-scale ones, where the projected Newton direction is obtained by solving the small-scale linear system at each iteration. We have proved that the projected Newton direction is a descent direction of a merit function, and the points generated by PNT eventually converge to the unique minimizer of this merit function, which is just the regularized solution and the corresponding Lagrangian multiplier.

Experimental tests on both small and large-scale Bayesian inverse problems have demonstrated the excellent convergence property, robustness and efficiency of PNT. The most demanding computational tasks in PNT are primarily matrix-vector products, making it particularly well-suited for large-scale problems.

# Acknowledgments

The author thanks Dr. Felipe Atenas for helpful discussions.

# References

[1] M. V. Afonso, J. M. Bioucas-Dias, and M. A. Figueiredo. An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems. *IEEE Trans. Image Process.*, 20(3):681–695, 2010.

[2] F. S. V. Bazán. Fixed-point iterations in determining the Tikhonov regularization parameter. *Inverse Probl.*, 24(3):035001, 2008.

[3] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2(1):183–202, 2009.

[4] J. Blowey, A. Craig, and T. Shardlow. *Frontiers in numerical analysis: Durham 2002*. Springer Science & Business Media, 2003.

[5] P. N. Brown and Y. Saad. Convergence theory of nonlinear newton–krylov algorithms. *SIAM J. Optim.*, 4(2):297–330, 1994.

[6] T. Buzug. *Computed Tomography*. Springer, 2008.

[7] D. Calvetti, F. Pitolli, J. Prezioso, E. Somersalo, and B. Vantaggi. Priorconditioned CGLS-based quasi-MAP estimate, statistical stopping rule, and ranking of priors. *SIAM J. Sci. Comput.*, 39(5):S477–S500, 2017.

[8] D. Calvetti, F. Pitolli, E. Somersalo, and B. Vantaggi. Bayes meets Krylov: Statistically inspired preconditioners for CGLS. *SIAM Rev.*, 60(2):429–461, 2018.

[9] D. Calvetti and E. Somersalo. Priorconditioners for linear systems. *Inverse Probl.*, 21(4):1397, 2005.

[10] N. A. Caruso and P. Novati. Convergence analysis of LSQR for compact operator equations. *Linear Algebra Appl.*, 583:146–164, 2019.

[11] J. Chung, J. G. Nagy, and D. P. O'Leary. A weighted-GCV method for Lanczos-hybrid regularization. *Electr. Trans. Numer. Anal.*, 28(29):149–167, 2008.

[12] J. Chung and A. K. Saibaba. Generalized hybrid iterative methods for large-scale Bayesian inverse problems. *SIAM J. Sci. Comput.*, 39(5):S24–S46, 2017.

[13] J. Cornelis, N. Schenkels, and W. Vanroose. Projected Newton method for noise constrained Tikhonov regularization. *Inverse Probl.*, 36(5):055002, 2020.

[14] J. Cornelis and W. Vanroose. Projected Newton method for noise constrained ell_p regularization. *Inverse Probl.*, 36(12):125004, 2020.

[15] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Commun. Pure Appl. Math.*, 57(11):1413–1457, 2004.

[16] A. Garbuno-Inigo, F. Hoffmann, W. Li, and A. M. Stuart. Interacting Langevin diffusions: Gradient structure and ensemble Kalman sampler. *SIAM J. Appl. Dyn. Syst.*, 19(1):412–441, 2020.

[17] S. Gazzola, P. C. Hansen, and J. G. Nagy. IR Tools: A MATLAB package of iterative regularization methods and large-scale test problems. *Numer. Algor.*, 81(3):773–811, 2019.

[18] S. Gazzola and P. Novati. Automatic parameter setting for Arnoldi–Tikhonov methods. *J. Comput. Appl. Math.*, 256:180–195, 2014.

[19] S. Gazzola, P. Novati, and M. R. Russo. On Krylov projection methods and Tikhonov regularization. *Electr. Trans. Numer. Anal.*, 44:83–123, 2015.

[20] S. Gazzola and M. Sabaté Landman. Krylov methods for inverse problems: Surveying classical, and introducing new, algorithmic approaches. *GAMM-Mitteilungen*, 43(4):e202000017, 2020.

[21] T. Goldstein and S. Osher. The split Bregman method for l1-regularized problems. *SIAM J. Imaging Sci.*, 2(2):323–343, 2009.

[22] G. H. Golub and H. G. Wahba. Generalized Cross-Validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

[23] P. C. Hansen. Analysis of discrete ill-posed problems by means of the L-curve. *SIAM Rev.*, 34(4):561–580, 1992.

[24] P. C. Hansen. Regularization Tools version 4.0 for Matlab 7.3. *Numer. Algor.*, 46(2):189–194, 2007.

[25] P. C. Hansen, J. G. Nagy, and D. P. O'Leary. *Deblurring Images: Matrices, Spectra and Filtering*. SIAM, Philadelphia, 2006.

[26] G. Huang, L. Reichel, and F. Yin. On the choice of subspace for large-scale Tikhonov regularization problems in general form. *Numer. Algor.*, 81:33–55, 2019.

[27] Z. Jia and H. Li. The joint bidiagonalization method for large GSVD computations in finite precision. *SIAM J. Matrix Anal. Appl.*, 44(1):382–407, 2023.

[28] N. Jorge and J. W. Stephen. *Numerical Optimization*. Spinger, 2006.

[29] J. Kaipio and E. Somersalo. *Statistical and Computational Inverse Problems*. Springer, 2006.

[30] M. E. Kilmer, P. C. Hansen, and M. I. Espanol. A projection–based approach to general-form Tikhonov regularization. *SIAM J. Sci. Comput.*, 29(1):315–330, 2007.

[31] K. Z. Kody Law, Andrew Stuart. *Data Assimilation: A Mathematical Introduction*. Springer, 2015.

[32] G. Landi. The Lagrange method for the regularization of discrete ill-posed problems. *Comput. Optim. Appl.*, 39(3):347–368, 2008.

[33] H. Li. A preconditioned Krylov subspace method for linear inverse problems with general-form Tikhonov regularization. *arXiv:2308.06577v1*, 2023.

[34] H. Li. Subspace projection regularization for large-scale Bayesian linear inverse problems. *arXiv preprint, arXiv:2310.18618*, 2023.

[35] H. Li. The joint bidiagonalization of a matrix pair with inaccurate inner iterations. *SIAM J. Matrix Anal. Appl.*, 45(1):232–259, 2024.

[36] J. Liesen and Z. Strakos. *Krylov subspace methods: principles and analysis*. Numerical Mathematics and Scie, 2013.

[37] N. Luiken and T. Van Leeuwen. Relaxed regularization for linear inverse problems. *SIAM J. Sci. Comput.*, 43(5):S269–S292, 2021.

[38] J. L. Mead and R. A. Renaut. A Newton root-finding algorithm for estimating the regularization parameter for solving ill-conditioned least squares problems. *Inverse Probl.*, 25(2):025002, 2008.

[39] S. Morigi, L. Reichel, and F. Sgallari. Orthogonal projection regularization operators. *Numer. Algor.*, 44:99–114, 2007.

[40] V. A. Morozov. Regularization of incorrectly posed problems and the choice of regularization parameter. *USSR Computational Mathematics and Mathematical Physics*, 6(1):242–251, 1966.

[41] D. P. O'Leary and J. A. Simmons. A bidiagonalization-regularization procedure for large scale discretizations of ill-posed problems. *SIAM J. Sci. Statist. Comput.*, 2(4):474–489, 1981.

[42] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin. An iterative regularization method for total variation-based image restoration. *Multiscale Model. Simul.*, 4(2):460–489, 2005.

[43] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12(4):617–629, 1975.

[44] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Software*, 8:43–71, 1982.

[45] L. Reichel, F. Sgallari, and Q. Ye. Tikhonov regularization based on generalized Krylov subspace methods. *Appl. Numer. Math.*, 62(9):1215–1228, 2012.

[46] L. Reichel and A. Shyshkov. A new zero-finder for tikhonov regularization. *BIT Numer. Math.*, 48:627–643, 2008.

[47] R. A. Renaut, A. W. Helmstetter, and S. Vatankhah. Unbiased predictive risk estimation of the tikhonov regularization parameter: convergence with increasing rank approximations of the singular value decomposition. *BIT Numer. Math.*, 59:1031–1061, 2019.

[48] R. A. Renaut, S. Vatankhah, and V. E. Ardesta. Hybrid and iteratively reweighted regularization by unbiased predictive risk and weighted GCV for projected systems. *SIAM J. Sci. Comput.*, 39(2):B221–B243, 2017.

[49] M. Richter. *Inverse Problems: Basics, Theory and Applications in Geophysics.* Springer, 2016.

[50] A. K. Saibaba, J. Chung, and K. Petroske. Efficient Krylov subspace methods for uncertainty quantification in large Bayesian linear inverse problems. *Numer. Linear Algebra Appl.*, 27(5):e2325, 2020.

[51] A. M. Stuart. Inverse problems: a Bayesian perspective. *Acta Numer.*, 19:451–559, 2010.

[52] W. Tian and X. Yuan. Linearized primal-dual methods for linear inverse problems with total variation regularization and finite element discretization. *Inverse Probl.*, 32(11):115011, 2016.

[53] C. F. Van Loan. Generalizing the singular value decomposition. *SIAM J. Numer. Anal.*, 13(1):76–83, 1976.

[54] W. Yin, S. Osher, D. Goldfarb, and J. Darbon. Bregman iterative algorithms for \ell_1-minimization with applications to compressed sensing. *SIAM J. Imaging Sci.*, 1(1):143–168, 2008.