Biometrics *, 1–24 DOI: ***
May 2019

On Computation of Semi-Parametric Maximum Likelihood Estimators with Shape Constraints

Yudong Wang[1,*], Zhi-Sheng Ye[1,], and Hongyuan Cao[2,***]**

[1]Department of Industrial Systems Engineering & Management, National University of Singapore, Singapore

[2]Department of Statistics, Florida State University, Florida, U.S.A.

**email:* yudongw@u.nus.edu

***email:* yez@nus.edu.sg

****email:* hcao@fsu.edu

Summary: Large sample theory of semi-parametric models based on maximum likelihood estimation (MLE) with shape constraint on the non-parametric component is well studied. Relatively less attention has been paid to the computational aspect of semi-parametric MLE. The computation of semi-parametric MLE based on existing approaches such as the EM algorithm can be computationally prohibitive when the missing rate is high. In this paper, we propose a computational framework for semi-parametric MLE based on an inexact block coordinate ascent algorithm. We show theoretically that the proposed algorithm converges. This computational framework can be applied to a wide range of data with different structures, such as panel count data, interval-censored data and degradation data, among others. Simulation studies demonstrate favorable performance compared with existing algorithms in terms of accuracy and speed. Two data sets are used to illustrate the proposed computational method. The R package `BCA1SG` implementing the proposed algorithm is available online.

Key words: generalized Rosen algorithm, semi-parametric MLE, shape constraint

This paper has been submitted for consideration for publication in *Biometrics*

1. Introduction

Semi-parametric models usually impose shape constraints on the non-parametric component. This arises naturally in various applications. For instance, the cumulative function of a point process and the cumulative hazard function should be non-decreasing (Sun and Fang, 2003; Kosorok et al., 2004); In economics, the concavity constraint on utility functions follows from the law of diminishing marginal utility (Wu and Sickles, 2017); In engineering, as degradation is an irreversible process of cumulative damage to the product, the mean degradation function in a semi-parametric degradation model is restricted to be non-decreasing (Wang, 2010).

In the maximum likelihood estimation (MLE) of semi-parametric models, the non-parametric component $\Lambda(\cdot)$ is usually estimated by a step function or piecewise linear function (Pan, 1999; Wellner and Zhang, 2007; Wang, 2010), where the knots depend on the observed data. For example, in panel count and interval-censored data, the knots are the set of distinct observation times. The number of knots can be large compared with the sample sizes as in Sun and Wei (2000), Gomez et al. (2000), Giorgio et al. (2012) and Chiou et al. (2018), among others. Another strategy is to approximate $\Lambda(\cdot)$ using splines (Lu et al., 2009; Zhang et al., 2010; Zhou et al., 2017) and convert the problem to estimation of the spline coefficients. Similarly, the number of spline coefficients increases with the sample size. Therefore, it is imperative to develop stable and computationally efficient algorithms for semi-parametric MLE (Fleming and Lin, 2000; Tsodikov, 2003; Kosorok, 2009).

For interval-censored and panel count data, Huang and Wellner (1997) and Wellner and Zhang (2007) proposed a doubly iterative algorithm which updates the regression coefficient $\boldsymbol{\beta}$ and non-parametric function $\Lambda(\cdot)$ through the Newton-Raphson and the iterative convex minorant (ICM) algorithm, respectively. As mentioned in Section 2 of Wellner and Zhang (2007), this algorithm is extremely time-consuming even with moderate sample size. This alternating strategy was also adopted in Sun and Shen (2009) and Dumbgen et al. (2011) for

computing MLE in semi-parametric regression models. Another approach is to update both the parametric and non-parametric components simultaneously in each iteration. Towards this goal, Pan (1999) reformulated the ICM algorithm as a generalized gradient projection method and extended it to semi-parametric problems for interval-censored data. The algorithm was found to be time-consuming in Cheng et al. (2011), and they further extended it to a projected Newton-Raphson algorithm that makes use of the full Hessian matrix in each iteration. The algorithm takes only a few iterations to converge, yet it takes a long time to compute the full Hessian and its inverse matrix in each iteration, especially when the searching space in the non-parametric function is large.

As some semi-parametric problems can be cast into the missing data framework, the EM algorithm is constantly used for inference. By introducing latent Poisson variables, Wang et al. (2016) developed the EM algorithm for the spline-based semi-parametric proportional hazard model with interval-censored data. Zeng et al. (2016) used the EM algorithm for semi-parametric transformation models with interval-censored data. The algorithm converges slowly when the censoring intervals are wide resulting in high missing rate. For degradation data, Wang (2010), Ye et al. (2014) and Zhai and Ye (2018) developed the EM algorithm for different degradation process models by augmenting the possibly unobserved degradation of a product unit in all distinct observation times. The convergence speed is slow when the observation times are different for the product units, or when the true parameters are close to the boundary of the parameter space.

There are several consequences when an inefficient computational method is used. First, the optimization algorithm can terminate immaturely when common termination rules are used, leading to highly biased parameter estimation. Second, when the bootstrap is used for inference, a large number of replications is needed and the computational time can be infeasible. This is especially relevant in the big data era, where massive data sets, such as

those collected from wearable devices in clinical trials and epidemiological studies, require scalable computational algorithms.

In this paper, we propose a new computational framework for MLE in semi-parametric models. Denote $\ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ as the log-likelihood function of a general semi-parametric model, where $\boldsymbol{\beta}$ is the parametric component, and $\boldsymbol{\Lambda}$ is the vector consisting of all the parameters in estimating $\Lambda(\cdot)$. If $\Lambda(\cdot)$ is estimated by monotone splines, for example, $\boldsymbol{\Lambda}$ is the vector of all spline coefficients. It is common that $\boldsymbol{\beta} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ for fixed $\boldsymbol{\Lambda}$ is nonconcave. In these cases, simultaneously updating all parameters is unstable unless the initial value is close enough to the MLE. A natural remedy is to use the block coordinate ascent (BCA) that iteratively maximizes the log-likelihood function with respect to each of the two parts while keeping the other fixed. From observations in the simulation studies, it is a waste of time to exactly maximize $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ at the first few BCA iterations because $\boldsymbol{\beta}$ is still far from optimal and the exact maximization takes a non-negligible time. Motivated by this observation, we propose to update $\boldsymbol{\Lambda}$ by maximizing $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ inexactly at each BCA iteration.

In the gradient projection (Rosen, 1960), the search direction is obtained by projecting the gradient vector $\boldsymbol{g}$ onto the region defined by the active constraints. Borrowing the idea of Newton's method, Jamshidian (2004) proposed the generalized Rosen (GR) algorithm that uses the generalized gradient $\tilde{\boldsymbol{g}} = W^{-1}\boldsymbol{g}$, where W is a positive definite matrix. Similar to Newton's method versus the steepest descent, GR converges much faster than the original version if W is properly chosen. GR was used by Zhang and Jamshidian (2004) in computing the non-parametric MLE of the survival function with censored data, Lu et al. (2009) and Zhang et al. (2010) to compute spline-based semi-parametric MLE of panel count data and interval-censored data, and Lu (2010) to compute semi-parametric MLE in partial linear models. Based on the GR algorithm, we propose a new algorithm where the non-parametric vector $\boldsymbol{\Lambda}$ is updated by implementing only one GR iteration, and the parametric component

$\boldsymbol{\beta}$ is updated through maximizing $\boldsymbol{\beta} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ exactly. The newly proposed algorithm is called the block coordinate ascent with one-step GR (BCA1SG). The one-step GR update significantly speeds up the algorithm in terms of run time. When $\boldsymbol{\beta} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ can be stably maximized and $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is concave, we show that the BCA1SG is convergent. The two conditions above are satisfied in many semi-parametric models, and thus the proposed computational framework has a wide spectrum of applications.

The rest of the paper is organized as follows. Section 2 provides some definitions in constrained optimization and briefly describes the general GR algorithm. In Section 3, we introduce the block coordinate ascent with one-step GR and show that it converges under mild conditions. We apply the proposed algorithm to panel count data, interval-censored data, and degradation data, respectively, in Section 4. Section 5 and Section 6 are devoted respectively to simulation studies and data analysis. We conclude the paper in Section 7. All the proofs of the main theorems and technical lemmas are collected in the Supplementary Material.

2. Preliminaries

2.1 *Some definitions in optimization*

Consider the following constrained maximization problem that is common in likelihood inference of non-parametric statistical models:

$$\begin{aligned} &\text{maximize} \quad \ell(\boldsymbol{\Lambda}), \\ &\text{subject to} \quad A\boldsymbol{\Lambda} \geqslant \boldsymbol{b}, \end{aligned} \tag{1}$$

where $A = (\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_p)^T \in \mathbb{R}^{p \times m}$ is of full row rank, and $\boldsymbol{b} = (b_1, b_2, \ldots, b_p)^T \in \mathbb{R}^p$. In (1), $\ell(\cdot)$ is the log-likelihood function, $\boldsymbol{\Lambda} \in \mathbb{R}^m$ is the vector consisting of all the parameters in estimating $\Lambda(\cdot)$, and $A\boldsymbol{\Lambda} \geqslant \boldsymbol{b}$ represents shape constraints, such as nonnegativity or convexity, on $\boldsymbol{\Lambda}$.

As a preliminary, some definitions in constrained optimization are first introduced as follows.

DEFINITION 1: A point $\boldsymbol{\Lambda} \in \mathbb{R}^m$ is *feasible* if it satisfies the constraint $A\boldsymbol{\Lambda} \geqslant \boldsymbol{b}$.

DEFINITION 2: An inequality constraint $\boldsymbol{a}_i^T\boldsymbol{\Lambda} \geqslant b_i$ is said to be *active* at a feasible point $\boldsymbol{\Lambda}$ if $\boldsymbol{a}_i^T\boldsymbol{\Lambda} = b_i$, and *inactive* if $\boldsymbol{a}_i^T\boldsymbol{\Lambda} > b_i$.

DEFINITION 3: A point $\boldsymbol{\Lambda}^*$ is called a *stationary point* of problem (1) if $\nabla\ell(\boldsymbol{\Lambda}^*)^T(\boldsymbol{\Lambda} - \boldsymbol{\Lambda}^*) \leqslant 0$ for any feasible $\boldsymbol{\Lambda}$.

If $\tilde{\boldsymbol{\Lambda}}$ is a local maximizer of problem (1), then there exists $\boldsymbol{\lambda} \in \mathbb{R}^p$, which is called the *Lagrange multiplier* vector, such that

- *primal feasibility*: $A\tilde{\boldsymbol{\Lambda}} \geqslant \boldsymbol{b}$;
- *stationarity*: $\nabla\ell(\tilde{\boldsymbol{\Lambda}}) + A^T\boldsymbol{\lambda} = \boldsymbol{0}$;
- *complementary slackness*: $\lambda_i(\boldsymbol{a}_i^T\tilde{\boldsymbol{\Lambda}} - b_i) = 0$, $i = 1, 2, \ldots, p$;
- *dual feasibility*: $\lambda_i \geqslant 0$, $i = 1, 2, \ldots, p$.

The above four conditions are called the *Karush-Kuhn-Tucker (KKT) conditions* (Luenberger and Ye, 2015, Section 11.8). They are necessary conditions for a point $\boldsymbol{\Lambda}$ to be the maximizer of problem (1).

2.2 *The GR algorithm*

GR is an active set algorithm developed to solve the constrained optimization problem displayed in (1) (Zhang and Jamshidian, 2004). Let $\boldsymbol{\Lambda}_0$ be the parameter value at the beginning of a GR iteration. Define $\mathcal{A} = \{i : \boldsymbol{a}_i^T\boldsymbol{\Lambda}_0 = b_i, i \leqslant p\}$ as the index set of all active constraints. GR first computes the gradient $\boldsymbol{g} = \nabla\ell(\boldsymbol{\Lambda}_0)$ and then the generalized gradient $\tilde{\boldsymbol{g}} = W^{-1}\boldsymbol{g}$ (Jamshidian and Jennrich, 1993), where W is a positive definite matrix that can be either fixed or adaptively updated at each iteration, and $\tilde{\boldsymbol{g}}$ is the gradient of $\ell(\boldsymbol{\Lambda})$ evaluated

at $\boldsymbol{\Lambda}_0$ in the metric induced by the norm $\|\boldsymbol{\Lambda}\|_W = \sqrt{\boldsymbol{\Lambda}^T W \boldsymbol{\Lambda}}$. When $\nabla^2 \ell(\boldsymbol{\Lambda}_0)$ is negative definite, W is often specified as $-\nabla^2 \ell(\boldsymbol{\Lambda}_0)$ to accelerate the convergence. To maintain the active constraints, GR projects $\tilde{\boldsymbol{g}}$ onto $\{\boldsymbol{d} : \boldsymbol{a}_i^T \boldsymbol{d} = 0, i \in \mathcal{A}\}$ in the metric of W and uses the projection as search direction. Define $A_{\mathcal{A}}$ as the matrix which only keeps the ith row of A for $i \in \mathcal{A}$. Then the projection is given by $\tilde{\boldsymbol{d}} = (I - W^{-1} A_{\mathcal{A}}^T (A_{\mathcal{A}} W^{-1} A_{\mathcal{A}}^T)^{-1} A_{\mathcal{A}}) \tilde{\boldsymbol{g}}$ (Jamshidian, 2004). After that, compute the largest possible step length $\alpha = \sup_{x>0} \{x; A(\boldsymbol{\Lambda}_0 + x\tilde{\boldsymbol{d}}) \geqslant \boldsymbol{b}\}$. Then the step length is computed as $\rho = \arg\max_{0<x\leqslant\alpha} \ell(\boldsymbol{\Lambda}_0 + x\tilde{\boldsymbol{d}})$, and the new iterate is given by $\boldsymbol{\Lambda}_0 + \rho\tilde{\boldsymbol{d}}$. GR repeats the above procedure until $\tilde{\boldsymbol{d}} = \boldsymbol{0}$, and the solution is denoted as $\tilde{\boldsymbol{\Lambda}}$.

Afterwards the four conditions in the KKT conditions are checked one by one. First, $\tilde{\boldsymbol{\Lambda}}$ obviously satisfies the primal feasibility. Second, because $\tilde{\boldsymbol{d}} = \boldsymbol{0}$, we have

$$\nabla \ell(\tilde{\boldsymbol{\Lambda}}) - A_{\mathcal{A}}^T (A_{\mathcal{A}} W^{-1} A_{\mathcal{A}}^T)^{-1} A_{\mathcal{A}} W^{-1} \nabla \ell(\tilde{\boldsymbol{\Lambda}}) = \boldsymbol{0}. \tag{2}$$

Define $I_{\mathcal{A}}$ as the matrix which only keeps the ith column of the $p \times p$ identity matrix for $i \in \mathcal{A}$, and denote $\tilde{\boldsymbol{\lambda}} = -I_{\mathcal{A}} (A_{\mathcal{A}} W^{-1} A_{\mathcal{A}}^T)^{-1} A_{\mathcal{A}} W^{-1} \nabla \ell(\tilde{\boldsymbol{\Lambda}})$. Then equation (2) is equivalent to

$$\nabla \ell(\tilde{\boldsymbol{\Lambda}}) + A^T \tilde{\boldsymbol{\lambda}} = \boldsymbol{0},$$

which means the stationarity condition holds, and $\tilde{\boldsymbol{\lambda}}$ is the respective Lagrange multiplier vector. Third, it is easy to see that $\tilde{\lambda}_i = 0$ if $i \notin \mathcal{A}$, where $\tilde{\lambda}_i$ is the ith component of $\tilde{\boldsymbol{\lambda}}$. On the other hand, $\boldsymbol{a}_i^T \tilde{\boldsymbol{\Lambda}} - b_i = 0$ holds for $i \in \mathcal{A}$. Therefore,

$$\tilde{\lambda}_i (\boldsymbol{a}_i^T \tilde{\boldsymbol{\Lambda}} - b_i) = 0, \ \ i = 1, 2, \ldots, p,$$

which means the complementary slackness holds. Lastly, if $\min(\tilde{\boldsymbol{\lambda}}) \geqslant 0$, the dual feasibility holds and all the four conditions are satisfied. Otherwise, the index corresponding to the smallest Lagrange multiplier is removed from the index set $\mathcal{A}$ and the next iteration starts. The algorithm terminates when $\tilde{\boldsymbol{d}} = \boldsymbol{0}$ and $\min(\tilde{\boldsymbol{\lambda}}) \geqslant 0$.

For the comprehensive description of the GR algorithm, see Zhang and Jamshidian (2004).

Convergence of the active set algorithm was discussed in Section 12.3 in Luenberger and Ye (2015).

3. Block Coordinate Ascent with One-Step GR

Computing the MLE of a semi-parametric model usually requires solving

$$\begin{aligned} &\text{maximize} && \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda}), \\ &\text{subject to} && \boldsymbol{\beta} \in \Theta, \\ & && A\boldsymbol{\Lambda} \geqslant \boldsymbol{b}, \end{aligned} \tag{3}$$

where Θ is the parameter space for $\boldsymbol{\beta}$, and the linear constraint on $\boldsymbol{\Lambda}$ usually results from the shape constraint on $\Lambda(\cdot)$. The linear constraint in (3) is generic because common shape constraints on $\Lambda(\cdot)$, such as nonnegativity, monotonicity (Zhang and Jamshidian, 2004), and concavity (Dumbgen et al., 2011), can be readily transformed into linear inequalities on $\boldsymbol{\Lambda}$. When $\ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is jointly concave in $\boldsymbol{\beta}$ and $\boldsymbol{\Lambda}$, the GR algorithm can be used to solve this problem by setting W as $-\nabla^2 \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$. However, the concavity does not hold in many cases, such as the semi-parametric inverse Gaussian process model (Ye and Chen, 2014). Under such scenarios, the selection of the matrix W becomes tricky, which limits the application of the GR algorithm.

A close investigation into the log-likelihood reveals that $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is usually concave. As a result, $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ can be readily maximized by the GR algorithm. On the other hand, maximizing $\boldsymbol{\beta} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is almost the same as solving the MLE in the corresponding parametric models, and the maximization is usually well-studied. This motivates the use of BCA algorithm which alternates between the following two procedures until convergence.

- Keeping $\boldsymbol{\Lambda}$ fixed, update the parametric component by maximizing $\boldsymbol{\beta} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$;
- Keeping $\boldsymbol{\beta}$ fixed, update $\boldsymbol{\Lambda}$ by maximizing $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ through the GR algorithm.

According to our simulation experience, the value of $\boldsymbol{\beta}$ is usually far from optimal at first few

BCA iterations. Given a poor estimate of $\boldsymbol{\beta}$, the value of $\boldsymbol{\Lambda}$ which maximizes $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is very likely to be poor as well. Therefore, it is a waste of time to exactly maximize $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ at first few BCA iterations. In order to reduce the excessive runtime of a full GR, we propose the BCA1SG algorithm which runs only one iteration of the GR algorithm when updating $\boldsymbol{\Lambda}$. The key difference between our method and the BCA algorithm is that BCA runs the GR algorithm until convergence when updating $\boldsymbol{\Lambda}$ while we only run one iteration.

A building block of the BCA1SG algorithm is a subroutine which will be introduced in Section 3.1. Before formally introducing the subroutine, we first briefly discuss how it works. The subroutine first partitions the constraints on $\boldsymbol{\Lambda}$ into two groups: those to be treated as active and those to be treated as inactive. As defined in Section 2.1, a constraint is called active if the equality holds. The constraints partitioned as inactive are preserved, while those partitioned as active are replaced by corresponding equality constraints. For example, if the constraint $\boldsymbol{a}_1^T \boldsymbol{\Lambda} \geqslant b_1$ is partitioned as active, then it is replaced by $\boldsymbol{a}_1^T \boldsymbol{\Lambda} = b_1$. Under these newly defined constraints, the subroutine then solves the maximization problem. Because the feasible set defined by these new constraints is a subset of the feasible set of problem (3), the solution of problem (3) may fall outside this subset. Hence the solution given by the subroutine may not be the solution of problem (3). However, a nice result which will be shown later is that the solution given by the subroutine satisfies all the KKT conditions for problem (3) except the dual feasibility. Therefore, to complete the BCA1SG algorithm, we further develop a procedure that ensures the dual feasibility in Section 3.2.

3.1 *The Subroutine*

Given initial values $\boldsymbol{\beta}^{(0)}, \boldsymbol{\Lambda}^{(0)}$ and an initial index set of active constraints $\mathcal{A}^{(0)}$, the subroutine alternates between maximizing ℓ w.r.t. $\boldsymbol{\beta}$ exactly and $\boldsymbol{\Lambda}$ inexactly until the stopping criterion is satisfied. At the kth iteration of the subroutine, denote $\boldsymbol{\beta}^{(k)}, \boldsymbol{\Lambda}^{(k)}$ as the parameter values, $W^{(k)}$ the user-specified positive definite matrix, and $\mathcal{A}^{(k)}$ the index set. The subroutine is

described in Algorithm 1, where the detailed procedures to conduct the backtracking line search in line 6 are given in the Supplementary Material.

Algorithm 1 The Subroutine

Input: $\boldsymbol{\beta}^{(0)}, \boldsymbol{\Lambda}^{(0)}, \mathcal{A}^{(0)}$.

Output: $\tilde{\boldsymbol{\beta}}, \tilde{\boldsymbol{\Lambda}}, \tilde{\mathcal{A}}, \tilde{W}$.

1: **Initialization**: $k \leftarrow 0$.

2: **while** stopping criterion is not satisfied **do**

3: Fix $\boldsymbol{\Lambda}^{(k)}$, and update $\boldsymbol{\beta}^{(k+1)} = \arg\max_{\boldsymbol{\beta}\in\Theta} \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda}^{(k)})$.

4: Evaluate $\boldsymbol{g}^{(k)} = \nabla_{\boldsymbol{\Lambda}} \ell(\boldsymbol{\beta}^{(k+1)}, \boldsymbol{\Lambda}^{(k)})$, $W^{(k)}$ and $\tilde{\boldsymbol{g}}^{(k)} = (W^{(k)})^{-1}\boldsymbol{g}^{(k)}$.

5: Compute the projection

$$\boldsymbol{d}^{(k)} = \left[I - \left(W^{(k)}\right)^{-1} A_{\mathcal{A}^{(k)}}^T \left\{ A_{\mathcal{A}^{(k)}} \left(W^{(k)}\right)^{-1} A_{\mathcal{A}^{(k)}}^T \right\}^{-1} A_{\mathcal{A}^{(k)}} \right] \tilde{\boldsymbol{g}}^{(k)}. \tag{4}$$

6: Determine the step length $\rho^{(k)}$ along $\boldsymbol{d}^{(k)}$ through an inexact backtracking line search.

7: Update $\boldsymbol{\Lambda}^{(k+1)} = \boldsymbol{\Lambda}^{(k)} + \rho^{(k)}\boldsymbol{d}^{(k)}$, and $\mathcal{A}^{(k+1)} = \{i : \boldsymbol{a}_i^T \boldsymbol{\Lambda}^{(k+1)} = b_i\}$.

8: $k \leftarrow k + 1$.

9: **end while**

10: **return** $\boldsymbol{\beta}^{(k)}, \boldsymbol{\Lambda}^{(k)}, \mathcal{A}^{(k)}, W^{(k)}$.

A good property of the subroutine which provides much convenience for discussing the convergence is that the sequence of the index sets $\{\mathcal{A}^{(k)}\}$ will eventually become invariant. The following lemma summarizes this property.

LEMMA 1: *There exists an $N > 0$ such that $\{\mathcal{A}^{(k)}\}_{k>N}$ is a constant sequence.*

Based on this result, we then discuss convergence properties of the sequence $\{(\boldsymbol{\beta}^{(k)}, \boldsymbol{\Lambda}^{(k)})\}$ generated by the subroutine. To establish the convergence result, the following assumptions are needed.

(C1) The superlevel set $\mathcal{S} = \{(\boldsymbol{\beta}, \boldsymbol{\Lambda}) : \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda}) \geqslant \ell(\boldsymbol{\beta}^{(0)}, \boldsymbol{\Lambda}^{(0)})\}$ is compact.

(C2) For any $\boldsymbol{\beta} \in \Theta$, the function $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is concave.

(C3) For any feasible $\boldsymbol{\Lambda}$, the function $\boldsymbol{\beta} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ has a unique maximizer.

(C4) There exist $0 < L < U$ such that the eigenvalues of $W^{(k)}$ are bounded in the interval $[L, U]$ for all k.

(C5) The set of stationary points of the following constrained optimization problem is discrete.

$$\begin{aligned} \text{maximize} \quad & \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda}), \\ \text{subject to} \quad & \boldsymbol{\beta} \in \Theta, \\ & \boldsymbol{a}_i^T \boldsymbol{\Lambda} = b_i, \ i \in \tilde{\mathcal{A}}, \\ & \boldsymbol{a}_i^T \boldsymbol{\Lambda} \geqslant b_i, \ i \notin \tilde{\mathcal{A}}, \end{aligned} \tag{5}$$

where $\tilde{\mathcal{A}}$ is the limit of $\{\mathcal{A}^{(k)}, k \geqslant 1\}$.

Condition (C1) is a commonly adopted assumption in optimization, e.g., the EM algorithm (Wu, 1983) and the coordinate descent (Tseng, 2001; Cassioli et al., 2013). Condition (C2) is satisfied by many semi-parametric models, such as the models for panel count data, interval-censored data and degradation data. A detailed proof of the concavity of $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ in these three semi-parametric models is given in the Supplementary Material. Condition (C3) is required to guarantee the identifiablility of the corresponding parametric models, and it is satisfied in most semi-parametric models. If $W^{(k)}$ is specified as the negative of the Hessian matrix, i.e., $W^{(k)} = -\nabla^2_{\boldsymbol{\Lambda}} \ell(\boldsymbol{\beta}^{(k+1)}, \boldsymbol{\Lambda}^{(k)})$, then condition (C4) can be replaced by the condition that $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is twice continuously differentiable and $\nabla^2_{\boldsymbol{\Lambda}} \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is negative definite for any feasible $\boldsymbol{\beta}$. This condition holds for many semi-parametric models, such as models for panel count data and degradation data. Condition (C5) is needed when proving the sequence of iterates is convergent. A similar discrete assumption is also adopted in Wu (1983).

THEOREM 1: *Suppose that conditions (C1)-(C5) hold. Then* $\lim_{k \to \infty} \boldsymbol{d}^{(k)} = 0$ *and the se-*

quence $\{(\boldsymbol{\beta}^{(k)}, \boldsymbol{\Lambda}^{(k)})\}$ *generated by the subroutine converges to a stationary point of problem (5).*

Theorem 1 reveals that the output of the subroutine $(\tilde{\boldsymbol{\beta}}, \tilde{\boldsymbol{\Lambda}})$ is a stationary point of problem (5). However, whether it is a good solution for problem (3), which is of primary interest, is still unknown. In the next theorem, we show that $(\tilde{\boldsymbol{\beta}}, \tilde{\boldsymbol{\Lambda}})$ satisfies the stationarity in the KKT conditions for problem (3). To establish this result, we need two more assumptions as follows.

(C6) For any feasible $\boldsymbol{\Lambda}$, the maximizer of $\boldsymbol{\beta} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is an interior point of Θ.

(C7) $\nabla_{\boldsymbol{\beta}} \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is a continuous function in $(\boldsymbol{\beta}, \boldsymbol{\Lambda})$.

Condition (C6) holds in most semi-parametric models because the parameter space Θ for the parametric component $\boldsymbol{\beta}$ is often open. Condition (C7) is mild and fulfilled by most semi-parametric models. See the semi-parametric models in Wellner and Zhang (2007), Ye and Chen (2014), and Zeng et al. (2016) for examples.

THEOREM 2: *Suppose conditions (C1)-(C7) hold. Then* $(\tilde{\boldsymbol{\beta}}, \tilde{\boldsymbol{\Lambda}})$ *satisfies the stationarity in the KKT conditions for problem (3). The Lagrange multipliers associated with constraints on* $\boldsymbol{\beta}$ *are all zeros, and the Lagrange multiplier vector associated with constraints on* $\boldsymbol{\Lambda}$ *is*

$$\tilde{\boldsymbol{\lambda}} = -I_{\tilde{\mathcal{A}}} \Big(A_{\tilde{\mathcal{A}}} \tilde{W}^{-1} A_{\tilde{\mathcal{A}}}^T \Big)^{-1} A_{\tilde{\mathcal{A}}} \tilde{W}^{-1} \nabla_{\boldsymbol{\Lambda}} \ell(\tilde{\boldsymbol{\beta}}, \tilde{\boldsymbol{\Lambda}}).$$

Theorem 2 implies that the output of the subroutine satisfies the stationarity in the KKT conditions for problem (3). Besides, the primal feasibility obviously holds. Because the Lagrange multipliers associated with constraints on $\boldsymbol{\beta}$ and inactive constraints on $\boldsymbol{\Lambda}$ are all zeros, the complementary slackness is also satisfied. Therefore, it suffices to check whether $\tilde{\boldsymbol{\lambda}}$ satisfies the dual feasibility in the next section.

3.2 *The BCA1SG Algorithm*

As noted, the Lagrange multipliers $\tilde{\boldsymbol{\lambda}}$ associated with constraints on $\boldsymbol{\Lambda}$ may not be dual feasible. To improve on this, the BCA1SG algorithm alternates between the subroutine and the procedure that checks the dual feasibility until the dual feasibility holds.

Denote the Kth output of the subroutine as $\tilde{\boldsymbol{\beta}}^{(K)}, \tilde{\boldsymbol{\Lambda}}^{(K)}, \tilde{\mathcal{A}}^{(K)}, \tilde{W}^{(K)}$. Then the dual feasibility is checked through computing the Lagrange multiplier vector

$$\tilde{\boldsymbol{\lambda}}^{(K)} = -I_{\tilde{\mathcal{A}}^{(K)}} \Big(A_{\tilde{\mathcal{A}}^{(K)}} \big(\tilde{W}^{(K)}\big)^{-1} A^T_{\tilde{\mathcal{A}}^{(K)}} \Big)^{-1} A_{\tilde{\mathcal{A}}^{(K)}} \big(\tilde{W}^{(K)}\big)^{-1} \nabla_{\boldsymbol{\Lambda}} \ell(\tilde{\boldsymbol{\beta}}^{(K)}, \tilde{\boldsymbol{\Lambda}}^{(K)}). \tag{6}$$

If $\min(\tilde{\boldsymbol{\lambda}}^{(K)}) \geqslant 0$, the algorithm is terminated and $(\tilde{\boldsymbol{\beta}}^{(K)}, \tilde{\boldsymbol{\Lambda}}^{(K)}, \tilde{\boldsymbol{\lambda}}^{(K)})$ satisfies all the KKT conditions. Otherwise, drop the index corresponding to the smallest elements of $\tilde{\boldsymbol{\lambda}}^{(K)}$ from $\tilde{\mathcal{A}}^{(K)}$, and return $\tilde{\boldsymbol{\beta}}^{(K)}, \tilde{\boldsymbol{\Lambda}}^{(K)}, \tilde{\mathcal{A}}^{(K)}$ as the input for the subroutine to conduct the next iteration.

Write the subroutine in a functional form as $(\tilde{\boldsymbol{\beta}}, \tilde{\boldsymbol{\Lambda}}, \tilde{\mathcal{A}}, \tilde{W}) = \text{SUBROUTINE}(\boldsymbol{\beta}, \boldsymbol{\Lambda}, \mathcal{A})$, where $(\boldsymbol{\beta}, \boldsymbol{\Lambda}, \mathcal{A})$ and $(\tilde{\boldsymbol{\beta}}, \tilde{\boldsymbol{\Lambda}}, \tilde{\mathcal{A}}, \tilde{W})$ are the input and output, respectively. The proposed BCA1SG algorithm is described in Algorithm 2.

In the following, we investigate when the BCA1SG algorithm terminates. According to Sard's theorem for real-analytic functions (Souček and Souček, 1972), the sequence $\{\ell(\tilde{\boldsymbol{\beta}}^{(K)}, \tilde{\boldsymbol{\Lambda}}^{(K)})\}$ contains only finite distinct elements if $\ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is real-analytic. On the other hand, the algorithm strictly increases the objective function in each iteration. As a result, the BCA1SG algorithm always terminates in finite iterations. The following theorem summarizes this property.

THEOREM 3: *Suppose conditions (C1)-(C7) hold and the function $\ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is real-analytic on an open set containing $\{(\boldsymbol{\beta}, \boldsymbol{\Lambda}); \boldsymbol{\beta} \in \Theta, A\boldsymbol{\Lambda} \geqslant \boldsymbol{b}\}$. Then the BCA1SG algorithm is able to find the solution satisfying KKT conditions for problem (3) within finite iterations.*

Theorem 3 ensures that there exists an $N > 0$ such that $\min(\tilde{\boldsymbol{\lambda}}^{(N)}) \geqslant 0$, so the BCA1SG

Algorithm 2 BCA1SG Algorithm

Input: $\tilde{\boldsymbol{\beta}}^{(0)}, \tilde{\boldsymbol{\Lambda}}^{(0)}, \tilde{\mathcal{A}}^{(0)}$.

1: **Initialization**: $K \leftarrow 0$.

2: $(\tilde{\boldsymbol{\beta}}^{(K+1)}, \tilde{\boldsymbol{\Lambda}}^{(K+1)}, \tilde{\mathcal{A}}^{(K+1)}, \tilde{W}^{(K+1)}) \leftarrow \text{SUBROUTINE}(\tilde{\boldsymbol{\beta}}^{(K)}, \tilde{\boldsymbol{\Lambda}}^{(K)}, \tilde{\mathcal{A}}^{(K)})$.

3: Compute $\tilde{\boldsymbol{\lambda}}^{(K+1)}$ according to (6).

4: **if** $\min(\tilde{\boldsymbol{\lambda}}^{(K+1)}) \geqslant 0$ **then**

5: Terminate the algorithm and output $(\tilde{\boldsymbol{\beta}}^{(K+1)}, \tilde{\boldsymbol{\Lambda}}^{(K+1)}, \tilde{\boldsymbol{\lambda}}^{(K+1)})$.

6: **else**

7: Drop the index corresponding to the smallest elements of $\tilde{\boldsymbol{\lambda}}^{(K+1)}$ from $\tilde{\mathcal{A}}^{(K+1)}$.

8: Let $K \leftarrow K + 1$ and go back to step 2.

9: **end if**

Output: $(\boldsymbol{\beta}^*, \boldsymbol{\Lambda}^*, \boldsymbol{\lambda}^*)$ that satisfies the KKT conditions.

algorithm always terminates within finite iterations. The importance of this theorem lies in the fact that the algorithm would not be trapped in an endless loop.

In addition to nice theoretical properties, the BCA1SG algorithm has many advantages in the computational perspective. For example, it avoids wasting too much time on a poor estimate of $\boldsymbol{\beta}$, so the efficiency is significantly improved and the algorithm is less sensitive to initial values. Besides, very few numerical solutions are needed in this computational framework because almost all procedures in Algorithms 1 and 2 have closed forms except the maximization of $\boldsymbol{\beta} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$, which depends on the specific form of log-likelihood functions. Therefore, the algorithm is efficient and easy to implement.

4. Applications of the BCA1SG Algorithm

This section demonstrates the applications of the BCA1SG algorithm to semi-parametric models for panel count data, interval-censored survival data and degradation data. For all the three models, the non-parametric component $\Lambda(\cdot)$ is estimated using a nondecreasing step

function which is identifiable at the collection of all distinct observation times $\{T_0 < T_1 < \cdots < T_m\}$, where $T_0 = 0$ and $\Lambda(0) = 0$ by convention. Therefore, it suffices to estimate the m-dimensional vector $\big(\Lambda(T_1), \ldots \Lambda(T_m)\big)^T$ subject to the monotone constraint $\Lambda(T_1) \leqslant \cdots \leqslant \Lambda(T_m)$. For ease of computation, we transform $\big(\Lambda(T_1), \ldots \Lambda(T_m)\big)^T$ to $(\Delta\Lambda_1, \ldots, \Delta\Lambda_m)^T$, where $\Delta\Lambda_l = \Lambda(T_l) - \Lambda(T_{l-1})$ $(l = 1, \ldots, m)$. Then the monotone constraint becomes

$$I\boldsymbol{\Lambda} \geqslant \mathbf{0},$$

where I is an $m \times m$ identity matrix and $\boldsymbol{\Lambda} = (\Delta\Lambda_1, \ldots, \Delta\Lambda_m)^T$.

When implementing BCA1SG, we need to specify $W^{(k)}$ for the subroutine. If $\nabla^2_{\boldsymbol{\Lambda}}\ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is negative definite, a natural choice which usually leads to convergence in very few iterations is $-\nabla^2_{\boldsymbol{\Lambda}}\ell(\boldsymbol{\beta}^{(k+1)}, \boldsymbol{\Lambda}^{(k)})$. When $\boldsymbol{\Lambda}$ is high-dimensional, however, computing the Hessian and its inverse consumes much time. As with the literature (Jongbloed, 1998; Zhang and Jamshidian, 2004), we let $W^{(k)}$ be the diagonal matrix $-D^{(k)}$ that discards the second-order mixed derivatives of the Hessian, i.e., $[D^{(k)}]_{ll} = [\nabla^2_{\boldsymbol{\Lambda}}\ell(\boldsymbol{\beta}^{(k+1)}, \boldsymbol{\Lambda}^{(k)})]_{ll}$ $(l = 1, \ldots, m)$. If $\nabla^2_{\boldsymbol{\Lambda}}\ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is only negative semi-definite, $D^{(k)}$ may not be invertible. In such cases, we can adopt the Levenberg-Marquadt adjustment (Pan, 1999) by setting $W^{(k)} = -D^{(k)} + \epsilon_0 I$, where ϵ_0 is a suitably chosen small number. With such diagonal choices of $W^{(k)}$, the projection in the subroutine, and the Lagrange mulitipliers in the main algorithm have neat expressions, as shown in the following properties.

PROPERTY 1: *When $W^{(k)}$ is diagonal and $\boldsymbol{\Lambda}$ is subject to nonnegativity constraints only, the projection $\boldsymbol{d}^{(k)}$ in the kth iteration of the subroutine can be obtained as follows*

$$d_i^{(k)} = \begin{cases} \tilde{g}_i^{(k)}, & \text{if } i \notin \mathcal{A}^{(k)}, \\ 0, & \text{if } i \in \mathcal{A}^{(k)}, \end{cases}$$

where $d_i^{(k)}$ and $\tilde{g}_i^{(k)}$ are the ith component of $\boldsymbol{d}^{(k)}$ and $\tilde{\boldsymbol{g}}^{(k)}$, respectively.

PROPERTY 2: *When $\tilde{W}^{(K)}$ is diagonal and $\boldsymbol{\Lambda}$ is subject to nonnegativity constraints only,*

the Lagrange multipliers in the Kth iteration of the BCA1SG algorithm can be obtained as follows

$$\tilde{\lambda}_i^{(K)} = \begin{cases} -\frac{\partial \ell(\tilde{\boldsymbol{\beta}}^{(K)}, \tilde{\boldsymbol{\Lambda}}^{(K)})}{\partial \Delta \Lambda_i}, & \text{if } i \in \tilde{\mathcal{A}}^{(K)}, \\ 0, & \text{if } i \notin \tilde{\mathcal{A}}^{(K)}, \end{cases}$$

where $\tilde{\lambda}_i^{(K)}$ is the ith component of $\tilde{\boldsymbol{\lambda}}^{(K)}$.

Based on the above discussion, the BCA1SG algorithm can be readily applied to semi-parametric models for panel count data, interval-censored survival data, and degradation data. Implementation details for the three semi-parametric models are given in the Supplementary Material.

5. Simulation Study

Comprehensive simulation is used to examine the performance of the proposed algorithm on the semi-parametric models for panel count data, interval-censored survival data, and degradation data. The results on the model for panel count data are presented in this section, and the results on the other two models can be found in the Supplementary Material. In the simulation, the BCA1SG subroutine as well as the other algorithms considered in this section are terminated when the infinity norm of the difference between successive iterates is less than the threshold ϵ. In BCA1SG, we set $W^{(k)} = -D^{(k)} + 10^{-8} I$. The simulations are repeated 1,000 times in all the three subsections. All the computations are performed on a server with 32 cores (Intel Xeon CPU 2.60 GHz) using `R`.

5.1 *Nonhomogeneous Poisson Process*

Consider a nonhomogeneous Poisson process (NHPP) $\{\mathbb{N}(t), t \geqslant 0\}$ with mean cumulative function $E(\mathbb{N}(t)|\boldsymbol{Z}) = e^{\boldsymbol{\beta}^T \boldsymbol{Z}} \Lambda(t)$, where $\boldsymbol{Z} \in \mathbb{R}^d$ is the covariate vector, $\boldsymbol{\beta}$ is the regression coefficient, and $\Lambda(t)$ is a nondecreasing baseline mean cumulative function. There are three

covariates, $Z_1 \sim \text{Uniform}(0,1)$, $Z_2 \sim N(0,1)$ and $Z_3 \sim \text{Bernouli}(0.5)$. The true values of the baseline mean cumulative function and the regression coefficients are $\Lambda(t) \equiv t^{3/4}$ and $\boldsymbol{\beta} \equiv (\beta_1, \beta_2, \beta_3)^T = (0.5, 0.5, 1.5)^T$, respectively. The observation process and the counting process $\{\mathbb{N}(t), t \geqslant 0\}$ of interest are independent (Gruger et al., 1991). Specifically, the number of observation times is randomly selected from $\{26, \ldots, 35\}$, and the observation times are randomly chosen from $\{1, \ldots, 100\}$. We examine two sample sizes $n = 50, 100$. Both BCA1SG and the doubly iterative algorithm in Wellner and Zhang (2007) are used to compute the semi-parametric MLE. As suggested by Wellner and Zhang (2007), we first use the maximum pseudo likelihood estimates as initial values. The simulation results based on the 1,000 Monte Carlo replications are summarized in Figure 1 and Table 1. As can be seen, BCA1SG is faster and the estimates are more accurate. It is interesting to see that the runtime of BCA1SG decreases when the sample size increases, because the maximum pseudo likelihood estimates get closer to the MLE. To examine the sensitivity to the initial values, we rerun the simulation by setting $(\beta_1^{(0)}, \beta_2^{(0)}, \beta_3^{(0)}) = (0,0,0)$, $\Delta\Lambda_l^{(0)} = r(T_l - T_{l-1})$ $(l = 1, \ldots, m)$ where $r \sim \text{Uniform}(1,5)$. The results are displayed in Table 2. Again, BCA1SG produces more accurate estimates, and it is insensitive to the initial values. On the other hand, the doubly iterative algorithm becomes very time-consuming, which agrees with the observations in Wellner and Zhang (2007).

[Figure 1 about here.]

[Table 1 about here.]

[Table 2 about here.]

5.2 *NHPP with Gamma Frailties*

Consider the counting process $\{\mathbb{N}(t), t \geqslant 0\}$ with conditional mean $E(\mathbb{N}(t)|\gamma, \boldsymbol{Z}) = \gamma e^{\boldsymbol{\beta}^T \boldsymbol{Z}} \Lambda(t)$, where $\gamma \sim \text{Gamma}(\nu, 1/\nu)$. Without loss of generality, let $\nu = 1$. Other settings including the

covariates, the regression coefficient $\boldsymbol{\beta}$, the baseline mean function $\Lambda(t)$, and the observation schemes are the same as those in the last subsection. Two sample sizes $n = 50, 100$ are considered. Both BCA1SG and the doubly iterative algorithm in Wellner and Zhang (2007) are used to fit the synthetic data. As for initial values, we set $(\beta_1^{(0)}, \beta_2^{(0)}, \beta_3^{(0)}) = (0, 0, 0)$, $\nu^{(0)} \sim \text{Uniform}(0, 10)$, and $\Delta\Lambda_l^{(0)} = r(T_l - T_{l-1})$ $(l = 1, \ldots, m)$ with $r \sim \text{Uniform}(1, 5)$ for both methods. The simulation results based on the 1,000 Monte Carlo replications are presented in Table 3. The conclusions are similar to those in Section 5.1. That is, BCA1SG is much faster, and the associated estimates of $\boldsymbol{\Lambda}$ and $\boldsymbol{\beta}$ are much more accurate.

[Table 3 about here.]

6. Real Data Analysis

6.1 *Panel count data*

Chiou et al. (2018) did a semi-parametric analysis to the skin cancer data collected in the clinical trial conducted at the University of Wisconsin Comprehensive Cancer Center. The study recruited 290 patients experiencing skin cancer and randomly allocated them to two groups, a treatment group with oral difluoromethylornithine (DFMO) at a daily dose of 0.5 gram/m^2 and a placebo group. These patients were followed for 3-5 years, and the follow-up times of different patients differ. In each follow-up visit the newly developed skin tumors since the last examination were counted and removed, and then the treatment continued. Information about the treatment, age, gender and number of tumors at the beginning of the study for each patient are available. The dataset "skiTum" in R package `spef` including 100 patients is used in our analysis. We model the treatment effect of DFMO through the semi-parametric model

$$E\big(\mathbb{N}(t)|Z\big) = \Lambda(t)\exp(\beta_1 Z_1 + \beta_2 Z_2 + \beta_3 Z_3 + \beta_4 Z_4),$$

where $\mathbb{N}(t)$ is the cumulative counts of tumors observed from the beginning to time t. Z_1 is a dummy variable that takes one when the age at enrollment is larger than 65. Z_2, Z_3 are two indicators for the gender and treatment. Z_4 represents the number of tumors at enrollment.

We compute the semi-parametric MLE using both BCA1SG and the doubly iterative algorithm in Wellner and Zhang (2007). Initial values are $\boldsymbol{\beta}^{(0)} = (0, 0, 0, 0)$, and $\Delta\Lambda_l^{(0)} = T_l - T_{l-1}$ $(l = 1, \ldots, m)$. Two convergence thresholds $\epsilon = 10^{-4}$ and 10^{-5} are used. The estimation results are presented in Table 4, and the estimates of the baseline mean function are provided in the Supplementary Material. From Table 4, we can see that the doubly iterative algorithm is much more time-consuming than BCA1SG, especially when the convergence threshold is small. When $\epsilon = 10^{-4}$, the doubly iterative algorithm stops immaturely, making the estimates of regression coefficients far from the true MLE.

[Table 4 about here.]

6.2 *Interval-censored survival data*

Gomez et al. (2000) reported HIV infection data collected in a study conducted by the Germans Trias i Pujol Hospital among 972 intravenous drug users. During the study patients recurrently went to the hospital to take HIV antibody tests. The following information was collected from each patient: the age at enrollment; gender; the date of first usage of intravenous drugs; the date of last negative HIV antibody test and the date of first positive antibody test. As a result, the elapsed time between intravenous drug initiation and HIV infection is interval-censored. In line with Gomez et al. (2000), we focus on the 767 patients who started drug usage after 1980 and consider the following Cox model

$$\Lambda(t|Z) = \Lambda(t) \exp(\beta_1 Z_1 + \beta_2 Z_2),$$

where $\Lambda(t)$ is the baseline cumulative hazard function for the elapsed time and Z_1 and Z_2 represent gender and the age at enrollment, respectively. We compute the semi-parametric MLE using the proposed BCA1SG algorithm, the EM algorithm in Zeng et al. (2016),

and the doubly iterative algorithm in Huang and Wellner (1997) where the non-parametric component is updated through the iterative convex minorant algorithm. Initial values are $\boldsymbol{\beta}^{(0)} = (0, 0)$, $\Delta\Lambda_l^{(0)} = 1/m$ $(l = 1, \ldots, m)$, and two thresholds $\epsilon = 10^{-4}$ and 10^{-5} are used. Table 5 presents the results from the three algorithms, and the estimates of the baseline cumulative hazard function are provided in the Supplementary Material. As can be seen, the estimates given by the three algorithms are similar and the log-likelihood given by BCA1SG is slightly larger than those by the EM and the doubly iterative algorithm under both convergence thresholds. The EM and the doubly iterative algorithm take much longer time when we decrease ϵ from 10^{-4} to 10^{-5} while BCA1SG is much more efficient. Another interesting finding is that the doubly iterative algorithm is much more efficient than the EM algorithm in this example. The reason is twofold. First, based on our simulation experience, the ICM algorithm embedded in the doubly iterative algorithm is quite efficient if the true maximizer of $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ contains many zeros. In our case, the true ML estimate of the 111-dimensional vector $\boldsymbol{\Lambda}$ contains 88 zeros. Second, the observed censoring intervals are wide in this HIV data, leading to a high missing rate and thus a slow convergence for the EM algorithm. Specifically, the average length of the observed intervals (except the right-censored observations) is 67 months, which is quite large considering that the observation times range from 1 month to 178 months. Therefore, the doubly iterative algorithm is much faster than the EM algorithm in this case.

[Table 5 about here.]

7. Discussion

This paper investigated the computation of MLE in semi-parametric models with shape constraints on the non-parametric component. A new computational framework was proposed and its convergence properties were investigated. Unlike the original block coordinate

ascent algorithm (Tseng, 2001; Wellner and Zhang, 2007), the BCA1SG algorithm does not waste time and resources on maximizing $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ exactly for a poor estimate of $\boldsymbol{\beta}$, so it is insensitive to initial values. Similar to Newton's method, the proposed algorithm takes advantage of the information contained in the matrix W, which is often specified as $-\nabla^2_{\boldsymbol{\Lambda}} \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ when $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\beta}, \boldsymbol{\Lambda})$ is concave. As a result, the convergence speed is significantly improved. Without the proposed algorithm, resampling methods such as bootstrap and jackknife for semi-parametric models are prohibitive when $\boldsymbol{\Lambda}$ is high-dimensional. The BCA1SG algorithm makes these resampling methods feasible as it significantly improves the computational efficiency. Therefore, it is of great significance in constructing confidence intervals for semi-parametric models through the bootstrap procedure.

A topic for future research is to further improve the computational efficiency by adaptively changing the number of GR iterations to implement when updating $\boldsymbol{\Lambda}$. For example, the number can be small at the beginning and gradually increases as the algorithm proceeds.

Supplementary Materials

Web appendices, referenced in Sections 3–6, are available with this paper at the Biometrics website on Wiley Online Library. The R codes of the proposed algorithm and the data sets analyzed in the manuscript are integrated into the R package `BCA1SG`, which is now available on Github (yudongstat/BCA1SG). Interested readers can install this package by `install_github("yudongstat/BCA1SG/BCA1SG")`. This package will be available on the Comprehensive R Archive Network (CRAN) in the near future.

References

Cassioli, A., Di Lorenzo, M., and Sciandrone, M. (2013). On the convergence of inexact block coordinate descent methods for constrained optimization. *European Journal of Operational Research* **231,** 274–281.

Cheng, G., Zhang, Y., and Lu, L. (2011). Efficient algorithms for computing the non and semi-parametric maximum likelihood estimates with panel count data. *Journal of Nomparametric Statistics* **23,** 567–579.

Chiou, S. H., Xu, G., Yan, J., and Huang, C. (2018). Semiparametric estimation of the accelerated mean model with panel count data under informative examination times. *Biometrics* **74,** 944–953.

Dumbgen, L., Samworth, R., and Schuhmacher, D. (2011). Approximation by log-concave distributions, with applications to regression. *The Annals of Statistics* **39,** 702–730.

Fleming, T. R. and Lin, D. (2000). Survival analysis in clinical trials: past developments and future directions. *Biometrics* **56,** 971–983.

Giorgio, M., Guida, M., and Pulcini, G. (2012). A state-dependent wear model with an application to marine engine cylinder liners. *Technometrics* **52,** 172–187.

Gomez, G., Calle, M., Egma, J., and Muga, R. (2000). Risk of HIV infection as a function of the duration of intravenous drug use: a non-parametric bayesian approach. *Statistics in Medicine* **19,** 2641–2656.

Gruger, J., Kay, R., and Schumacher, M. (1991). The validity of inferences based on incomplete observations in disease state models. *Biometrics* **47,** 595–605.

Huang, J. and Wellner, J. (1997). Interval-censored survival data: a review of recent progress. In Lin, D. and Fleming, T., editors, *Proceedings of the First Seattle Symposium in Biostatistics*, pages 123–169, New York. Springer.

Jamshidian, M. (2004). On algorithms for restricted maximum likelihood estimation. *Computational Statistics & Data Analysis* **45,** 137–157.

Jamshidian, M. and Jennrich, I., R. (1993). Conjugate gradient acceleration of the EM algorithm. *Journal of the American Statistical Association* **88,** 221–228.

Jongbloed, G. (1998). The iterative convex minorant algorithm for nonparametric estimation.

Journal of Computational and Graphical Statistics **7,** 310–321.

Kosorok, M. R. (2009). What's so special about semiparametric methods? *The Indian Journal of Statistics, Series A* **71,** 331–353.

Kosorok, M. R., Lee, B. L., and Fine, J. P. (2004). Robust inference for univariate proportional hazards frailty regression models. *Annals of Statistics* **32,** 1448–1491.

Lu, M. (2010). Spline-based sieve maximum likelihood estimation in the partly linear model under monotonicity constraints. *Journal of Multivariate Analysis* **101,** 2528–2542.

Lu, M., Zhang, Y., and Huang, J. (2009). Semiparametric estimation methods for panel count data using monotone B-splines. *Journal of the American Statistical Association* **104,** 1060–1070.

Luenberger, D. and Ye, Y. (2015). *Linear and Nonlinear Programming.* Springer.

Pan, W. (1999). Extending the iterative convex minorant algorithm to the Cox model for interval-censored data. *Journal of Computational and Graphical Statistics* **8,** 109–120.

Rosen, J. (1960). The gradient projection method for nonlinear programming. *Journal of the Society for Industrial and Applied Mathematics* **8,** 181–217.

Souček, J. and Souček, V. (1972). Morse-Sard theorem for real-analytic functions. *Commentationes Mathematicae Universitatis Carolinae* **13,** 45–51.

Sun, J. and Fang, H.-B. (2003). A nonparametric test for panel count data. *Biometrika* **90,** 199–208.

Sun, J. and Shen, J. (2009). Efficient estimation for the proportional hazards model with competing risks and current status data. *Canadian Journal of Statistics* **37,** 592–606.

Sun, J. and Wei, L. (2000). Regression analysis of panel count data with covariate-dependent observation and censoring times. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **62,** 293–302.

Tseng, P. (2001). Convergence of a block coordinate descent method for nondifferentiable

minimization. *Journal of Optimization Theory and Applications* **109,** 475–494.

Tsodikov, A. (2003). Semiparametric models: a generalized self-consistency approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **65,** 759–774.

Wang, L., McMahan, C., Hudgens, M., and Qureshi, Z. (2016). A flexible, computationally efficient method for fitting the proportional hazards model to interval-censored data. *Biometrics* **72,** 222–231.

Wang, X. (2010). Wiener processes with random effects for degradation data. *Journal of Multivariate Analysis* **101,** 340–352.

Wellner, J. and Zhang, Y. (2007). Two likelihood-based semiparametric estimation methods for panel count data with covariates. *The Annals of Statistics* **35,** 2106–2142.

Wu, C. (1983). On the convergence properties of the EM algorithm. *The Annals of Statistics* **11,** 95–103.

Wu, X. and Sickles, R. (2017). Semiparametric estimation under shape constraints. *Ecnometrics and Statistics* **56,** 504–513.

Ye, Z.-S. and Chen, N. (2014). The inverse gaussian process as a degradation model. *Technometrics* **56,** 302–311.

Ye, Z.-S., Xie, M., Tang, L.-C., and Chen, N. (2014). Semiparametric estimation of gamma processes for deteriorating products. *Technometrics* **56,** 504–513.

Zeng, D., Mao, L., and Lin, D. (2016). Maximum likelihood estimation for semiparametric transformation models with interval-censored data. *Biometrika* **103,** 253–271.

Zhai, Q. and Ye, Z.-S. (2018). Degradation in common dynamic environments. *Technometrics* **60,** 461–471.

Zhang, Y., Hua, L., and Huang, J. (2010). A spline-based semiparametric maximum likelihood estimation method for the Cox model with interval-censored data. *Scandinavian Journal of Statistics* **37,** 338–354.

Zhang, Y. and Jamshidian, M. (2004). On algorithms for the nonparametric maximum likelihood estimator of the failure function with censored data. *Journal of Computational and Graphical Statistics* **13,** 123–140.

Zhou, Q., Hu, T., and Sun, J. (2017). A sieve semiparametric maximum likelihood approach for regression analysis of bivariate interval-censored failure time data. *Journal of the American Statistical Association* **112,** 664–672.

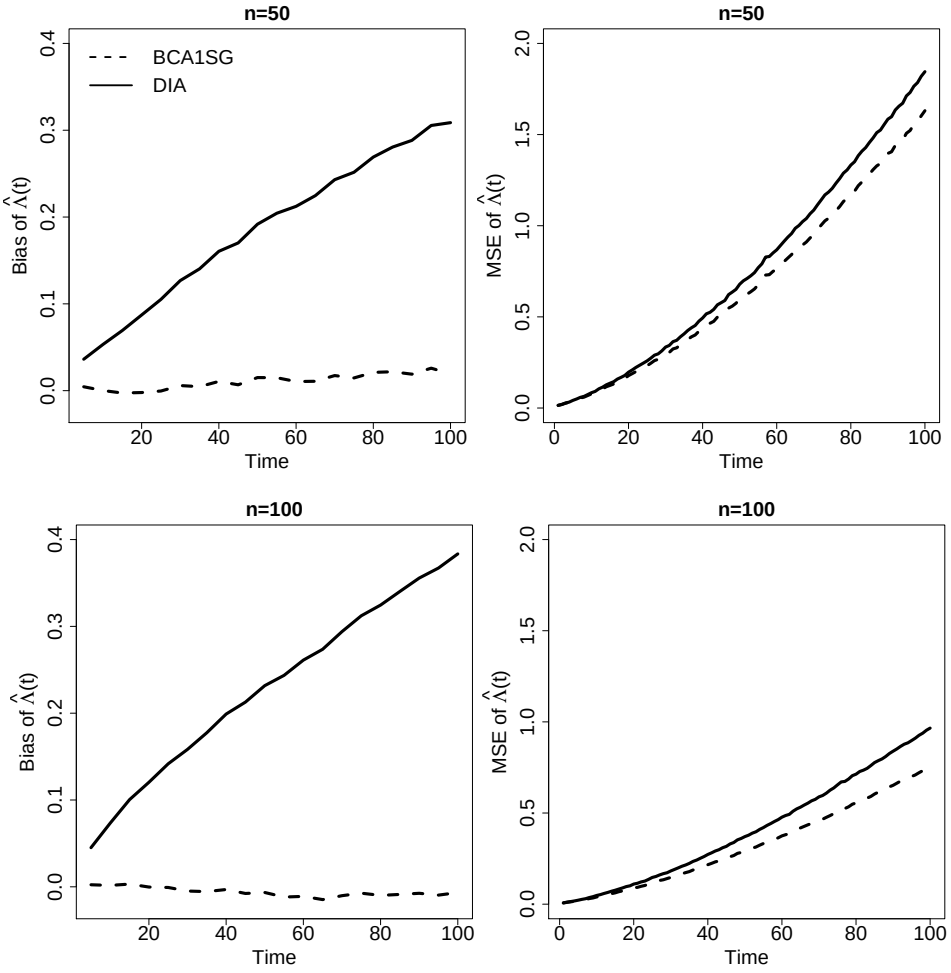

Figure 1. Pointwise biases (left panel) and MSEs (right panel) of the ML estimate of $\Lambda(t)$ under difference sample sizes for the NHPP model with MPLE as initial value and $\epsilon = 10^{-6}$.

Table 1
With maximum pseudo likelihood estimates as initial values, the number of iterations, time cost in seconds, averaged $\|\widehat{\Lambda}(t) - \Lambda(t)\|_\infty$*, and RMSE of* $\widehat{\boldsymbol{\beta}}$ *for the NHPP model when* $\epsilon = 10^{-6}$*,* $(\beta_1, \beta_2, \beta_3) = (0.5, 0.5, 1.5)$ *and* $\Lambda(t) = t^{3/4}$*. The standard errors are shown in parentheses*

	$n = 50$		$n = 100$	
	BCA1SG	DIA	BCA1SG	DIA
Iteration	160.9 (64.0)	225.2 (613.6)	81.6 (31.7)	122.2 (243.2)
Time cost	10.6 (5.0)	87.4 (112.6)	9.7 (4.0)	100.6 (130.7)
$\|\widehat{\Lambda}(t) - \Lambda(t)\|_\infty$	1.16 (0.73)	1.25 (0.78)	0.82 (0.50)	0.90 (0.58)
RMSE of $\widehat{\beta}_1 \times 10^2$	48.09	49.28	32.21	34.46
RMSE of $\widehat{\beta}_2 \times 10^2$	14.44	14.46	9.69	9.69
RMSE of $\widehat{\beta}_3 \times 10^2$	34.24	35.48	24.25	25.05

Table 2

Under random initial values, the number of iterations, time cost in seconds, averaged $\|\widehat{\Lambda}(t) - \Lambda(t)\|_\infty$*, and RMSE of* $\widehat{\boldsymbol{\beta}}$ *for the NHPP model when* $\epsilon = 10^{-6}$*,* $(\beta_1, \beta_2, \beta_3) = (0.5, 0.5, 1.5)$ *and* $\Lambda(t) = t^{3/4}$*. The standard errors are shown in parentheses*

	$n = 50$		$n = 100$	
	BCA1SG	DIA	BCA1SG	DIA
Iteration	254.7 (84.8)	470.9 (800.8)	150.7 (35.1)	296.7 (294.7)
Time cost	19.5 (7.9)	4170.6 (847.3)	20.9 (6.4)	6463.7 (752.0)
$\|\widehat{\Lambda}(t) - \Lambda(t)\|_\infty$	1.16 (0.73)	1.28 (0.79)	0.82 (0.50)	0.96 (0.61)
RMSE of $\widehat{\beta}_1 \times 10^2$	48.10	49.87	32.22	35.45
RMSE of $\widehat{\beta}_2 \times 10^2$	14.44	14.51	9.69	9.71
RMSE of $\widehat{\beta}_3 \times 10^2$	34.22	35.33	24.24	25.97

Table 3
Under random initial values, the number of iterations, time cost in seconds, averaged $\|\widehat{\Lambda}(t) - \Lambda(t)\|_\infty$*, and RMSEs of* $\widehat{\nu}$ *and* $\widehat{\boldsymbol{\beta}}$ *for the NHPP model with gamma frailties when* $\epsilon = 10^{-6}$*,* $\nu = 1$*,* $(\beta_1, \beta_2, \beta_3) = (0.5, 0.5, 1.5)$ *and* $\Lambda(t) = t^{3/4}$*. The standard errors are shown in parentheses*

	$n = 50$		$n = 100$	
	BCA1SG	DIA	BCA1SG	DIA
Iteration	1589.2 (671.1)	17.6 (15.7)	1444.6 (397.5)	13.7 (2.1)
Time cost	67.5 (29.4)	24264.8 (11840.2)	156.7 (51.5)	58666.8 (22071.3)
$\|\widehat{\Lambda}(t) - \Lambda(t)\|_\infty$	8.41 (6.38)	36.73 (17.90)	5.92 (4.37)	36.62 (12.82)
RMSE of $\widehat{\nu} \times 10^2$	24.93	22.00	16.33	14.81
RMSE of $\widehat{\beta}_1 \times 10^2$	49.17	98.13	37.46	95.76
RMSE of $\widehat{\beta}_2 \times 10^2$	15.06	15.83	11.59	12.59
RMSE of $\widehat{\beta}_3 \times 10^2$	31.68	42.70	19.37	33.49

Table 4
Number of iterations, time cost in seconds, log-likelihood and estimates of the regression coefficients when fitting the skin cancer data under different convergence thresholds ϵ

ϵ	Algorithm	Iteration	Time cost	Log-likelihood	$\widehat{\beta}_1$	$\widehat{\beta}_2$	$\widehat{\beta}_3$	$\widehat{\beta}_4$
10^{-4}	BCA1SG	549	20.9	272.945	0.175	0.177	0.141	0.123
	DIA	17	602.9	269.106	0.103	0.018	0.029	0.114
10^{-5}	BCA1SG	803	31.9	272.946	0.175	0.177	0.141	0.123
	DIA	38	2150.8	272.877	0.166	0.155	0.126	0.122

Table 5
Number of iterations, time cost in seconds, log-likelihood and estimates of the regression coefficients when fitting the HIV infection data under different convergence thresholds ϵ

ϵ	Algorithm	Iteration	Time cost	Log-likelihood	$\widehat{\beta}_1 \times 10^2$	$\widehat{\beta}_2 \times 10^2$
10^{-4}	BCA1SG	137	5.8	-604.20976	23.18	-1.37
	DIA	123	31.4	-604.21429	23.35	-1.26
	EM	442	220.1	-604.34055	23.20	-1.36
10^{-5}	BCA1SG	197	9.0	-604.20936	23.20	-1.37
	DIA	153	94.9	-604.20944	23.20	-1.36
	EM	1245	607.7	-604.21550	23.20	-1.37