Sequential Gradient Descent and Quasi-Newton’s Method for Change-Point Analysis

Xianyang Zhang
Texas A&M University

Trisha Dawn
Texas A&M University

Abstract

One common approach to detecting change-points is minimizing a cost function over possible numbers and locations of change-points. The framework includes several well-established procedures, such as the penalized likelihood and minimum description length. Such an approach requires finding the cost value repeatedly over different segments of the data set, which can be time-consuming when (i) the data sequence is long and (ii) obtaining the cost value involves solving a non-trivial optimization problem. This paper introduces a new sequential updating method (SE) to find the cost value effectively. The core idea is to update the cost value using the information from previous steps without re-optimizing the objective function. The new method is applied to change-point detection in generalized linear models and penalized regression. Numerical studies show that the new approach can be orders of magnitude faster than the Pruned Exact Linear Time (PELT) method without sacrificing estimation accuracy.

Keywords. Change-point analysis, Dynamic programming, Stochastic gradient descent.

1 INTRODUCTION

Change-point analysis is concerned with detecting and locating structure breaks in the underlying model of a data sequence ordered by time (space or other variables). The first work on change point analysis goes back to the 1950s, where the goal was to locate a shift in the mean of an independent and identically distributed Gaussian sequence for industrial quality control purposes (Page, 1954, 1955). Since then, change-point analysis has generated important activity in statistics and various application settings such as signal processing, climate science, economics, financial analysis, medical science, and bioinformatics. We refer the readers to Brodsky and Darkhovsky (1993); Csörgő et al. (1997); Tartakovsky et al. (2014) for book-length treatments and Aue and Horváth (2013); Niu et al. (2016); Aminikhanghahi and Cook (2017); Truong et al. (2020); Liu et al. (2021) for reviews on this subject.

There are two main branches of change-point detection methods: online methods that aim to detect changes as early as they occur in an online setting and offline methods that retrospectively detect changes when all samples have been observed. The focus of this paper is on the offline setting. A typical offline change-point detection method involves three major components: the cost function, the search method, and the penalty/constraint (Truong et al., 2020). The choice of the cost function and search method has a crucial impact on the method’s computational complexity. As increasingly larger data sets are being collected in modern applications, there is an urgent need to develop more efficient algorithms to handle such big data sets. Examples include testing the structure breaks for genetics data and detecting changes in the volatility of big financial data.

One popular way to tackle the change-point detection problem is to cast it into a model-selection problem by solving a penalized optimization problem over possible numbers and locations of change-points. The framework includes several well-established procedures, such as the penalized likelihood and minimum description length. The corresponding optimization can be solved exactly using dynamic programming (Auger and Lawrence, 1989; Jackson et al., 2005) whose computational cost is \( \sum_{t=1}^{T} \sum_{s=1}^{n} q(s) \), where \( T \) is the number of data points and \( q(s) \) denotes the time complexity for calculating the cost function value based on \( s \) data points. Killick et al. (2012) introduced the pruned exact linear time (PELT) algorithm with a pruning step in dy-
namic programming. PELT reduces the computational cost without affecting the exactness of the resulting segmentation. Rigail (2010) proposed an alternative pruned dynamic programming algorithm with the aim of reducing the computational effort. However, in the worst case scenario, the computational cost of dynamic programming coupled with the above pruning strategies remains the order of \( O(\sum_{t=1}^{T} \sum_{s=1}^{l_t} q(s)) \).

Unlike the pruning strategy, this paper aims to improve the computational efficiency of dynamic programming from a different perspective. We focus on the class of problems where computing the cost function involves solving a non-trivial optimization problem without a closed-form solution. Dynamic programming requires repeatedly solving the optimization over different data sequence segments, which can be very time-consuming for big data. We make the following contributions to address the issue.

(i) A new sequential method (SE) that can be coupled with the gradient descent (SeGD) and Newton’s method (SeN) is proposed to update the parameter estimate and the cost value in dynamic programming. The new strategy avoids repeatedly optimizing the objective function based on each data segment. It thus significantly improves the computational efficiency of the vanilla PELT, especially when computing the cost function involves solving a non-trivial optimization problem without a closed-form solution (see Table 1). Though our algorithm is no longer exact, numerical studies suggest that the new method achieves almost the same estimation accuracy as PELT does.

(ii) SeGD is related to the stochastic gradient descent (SGD) without-replacement sampling (Shamir, 2016; Nagaraj et al., 2019; Rajput et al., 2020). The main difference is that our update is along the time order of the data points, and hence no sampling or additional randomness is introduced. Using some techniques from SGD and transductive learning theory, we obtain the convergence rate of the approximate cost value derived from the algorithm to the true cost value.

(iii) The proposed method applies to a broad class of statistical models, such as parametric/nonparametric likelihood models, generalized linear models, and penalized regression.

Finally, we mention two other routes to reduce the computational complexity in change-point analysis. The first one is to relax the \( l_0 \) penalty on the number of parameters to an \( l_1 \) penalty (such as the total variation penalty) on the parameters to encourage a piece-wise constant solution. The resulting convex optimization problem can be solved in nearly linear time (Harchaou and Lévy-Leduc, 2010). In contrast, our method directly tackles the problem with the \( l_0 \) penalty. The second approach includes different approximation schemes, including window-based methods, binary segmentation and its variants (Vostrikova, 1981; Fryzlewicz, 2014), and bottom-up segmentation (Keogh et al., 2001). These methods are usually quite efficient and can be combined with various test statistics though they only provide approximate solutions. Our method can be regarded as a new approximation scheme for the \( l_0 \) penalization problem.

The rest of the paper is organized as follows. In Section 2, we briefly review the dynamic programming and the pruning scheme in change-point analysis. We describe the details of the SE algorithms in Section 3, including the motivation, its application in generalized linear models, and an extension to handle the case where computing the cost value involves solving a penalized optimization. We study the convergence property of the algorithm in Section 4. Sections 5 presents numerical results for synthesized and real data. Section 6 concludes.

<table>
<thead>
<tr>
<th>Dynamic programming</th>
<th>SE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time complexity</td>
<td>( \sum_{t=1}^{T} \sum_{s=1}^{l_t} q(s) )</td>
</tr>
</tbody>
</table>

Table 1: Comparison of the computational complexity. Here \( R_t \) is defined in Section 2.2 and \( q(s) \) denotes the time complexity for calculating the cost value based on \( s \) data points and \( q_0 \) is the time complexity for performing the one-step update described in Section 3.

2 DYNAMIC PROGRAMMING AND PRUNING

2.1 Dynamic programming

Change-point analysis concerns the partition of a data set ordered by time (space or other variables) into piece-wise homogeneous segments such that each piece shares the same behavior. Specifically, we denote the dataset by \( z = (z_1, \ldots, z_T) \). For \( 1 \leq s \leq t \leq T \), let \( z_{s:t} = (z_s, \ldots, z_t) \). If we assume that there are \( k \) change-points in the data, then we can split the data into \( k + 1 \) distinct segments. We let the location of the \( j \)th change-point be \( \tau_j \) for \( j = 1, 2, \ldots, k \), and set \( \tau_0 = 0 \) and \( \tau_{k+1} = T \). The \( (j + 1) \)th segment contains the data \( z_{\tau_j+1}, \ldots, z_{\tau_{j+1}} \) for \( j = 0, 1, \ldots, k \). We let \( \tau = (\tau_1, \ldots, \tau_k) \) be the set of change-point locations. The problem we aim to address is to infer both the number of change points and their locations.

Throughout the discussions, we let \( C(z_{s:t}) \) for \( s \leq t \) denote the cost for a segment consisting of the data points \( z_{s+1}, \ldots, z_t \). Of particular interest is the cost function...
defined as

\[ C(z_{s+1:t}) = \min_{\theta \in \Theta} \sum_{i=s+1}^{t} l(z_i, \theta) \] (1)

where \( l(\cdot, \theta) \) is the individual cost parameterized by \( \theta \) that belongs to a compact parameter space \( \Theta \subset \mathbb{R}^d \). Examples include (i) \( l(\cdot, \theta) \) is the negative log-likelihood of \( z_i \); (2) \( l(z_i, \theta) = L(f(x_i, \theta), y_i) \) with \( z_i = (x_i, y_i) \), where \( L \) is a loss function and \( f(\cdot, \theta) \) is an unknown regression function parameterized by \( \theta \). See more details and discussions in Section 3.2.

In this paper, we consider segmenting data by solving a penalized optimization problem. For \( 0 \leq k \leq T - 1 \), define

\[ C_{k,T} = \min_{\tau} \sum_{j=0}^{k} C(z_{\tau_j+1:\tau_{j+1}}). \]

We estimate the number of change-points by minimizing a linear combination of the cost value and a penalty function \( f \), i.e., \( \min_k \{ C_{k,T} + f(k, T) \} \). If the penalty function is linear in \( k \) with \( f(k, T) = \beta_T (k + 1) \) for some \( \beta_T > 0 \), then we can write the objective function as

\[ \min_{k} \{ C_{k,T} + f(k, T) \} = \min_{k, \tau} \sum_{j=0}^{k} \{ C(z_{\tau_j+1:\tau_{j+1}}) + \beta_T \}. \]

One way to solve the penalized optimization problem is through the dynamic programming approach (Killick et al., 2012; Jackson et al., 2005). Consider segmenting the data \( z_{1:t} \). Denote \( F(t) \) to be the minimum value of the penalized cost \( \min_k \{ C_{k,T} + f(k, T) \} \) for segmenting such data. We derive a recursion for \( F(t) \) by conditioning on the last change-point location,

\[ F(t) := \min_{k \leq t} \sum_{j=0}^{k} \{ C(z_{\tau_j+1:\tau_{j+1}}) + \beta_T \} \]

\[ = \min_{k} \left[ \sum_{j=0}^{k-1} \{ C(z_{\tau_j+1:\tau_{j+1}}) + \beta_T \} + C(z_{\tau_{k+1:t}}) + \beta_T \right] \]

\[ = \min_{\tau} \left[ \sum_{j=0}^{k} \{ C(z_{\tau_j+1:\tau_{j+1}}) + \beta_T \} + C(z_{\tau_{k+1:t}}) + \beta_T \right] \]

\[ = \min_{\tau} \{ F(\tau) + C(z_{\tau_{k+1:t}}) + \beta_T \}, \] (2)

where \( \tau_{k+1} = t \) in the first equation and \( \tau_{k+1} = \tau \) in the third equation. The segmentations can be recovered by taking the argument \( \tau \) which minimizes (2), i.e.,

\[ \tau^* = \arg\min_{0 \leq \tau < t} \{ F(\tau) + C(z_{\tau_{k+1:t}}) + \beta_T \}, \] (3)

which gives the optimal location of the last change-point in the segmentation of \( z_{1:t} \). The procedure is repeated until all the change-point locations are identified.

2.2 Pruning

A popular way to increase the efficiency of dynamic programming is by pruning the candidate set for finding the last change-point in each iteration. For the cost function in (1), we have for any \( \tau < t < T \), \( C(z_{\tau+1:t}) + C(z_{t+1:T}) \leq C(z_{\tau+1:T}) \). Killick et al. (2012) showed that for some \( t > \tau \) if

\[ F(\tau) + C(z_{\tau+1:t}) > F(t), \]

then at any future point \( t' > t, \tau \) can never be the optimal location of the most recent change-point prior to \( t' \). Define a sequence of sets \( \{ R_t \} \) recursively as \( R_t = \{ \tau \in R_{t-1} \cup \{ t - 1 \} : F(\tau) + C(z_{\tau+1:t-1}) \leq F(t - 1) \} \).

Then \( F(t) \) can be computed as

\[ F(t) = \min_{\tau \in R_t} \{ F(\tau) + C(z_{\tau+1:t}) + \beta \} \]

and the minimizer \( \tau^* \) in (3) belongs to \( R_t \). This pruning technique forms the basis of the Pruned Exact Linear Time (PELT) algorithm. Under suitable conditions that allow the expected number of change-points to increase linearly with \( T \), Killick et al. (2012) showed that the expected computational cost for PELT is bounded by \( LT \) for some constant \( L < \infty \). In the worst case where no pruning occurs, the computational cost of PELT is the same as the vanilla dynamic programming.

3 METHODOLOGY

3.1 Sequential algorithms

For large-scale data, the computational cost of PELT can still be prohibitive due to the burden of repeatedly solving the optimization problem (1). For many statistical models, the time complexity for obtaining \( C(z_{s+1:t}) \) is linear in the number of observations \( t - s \). Therefore, in the worst-case scenario, the overall time complexity can be as high as \( O(T^3) \). To alleviate the problem, we propose a fast algorithm by sequentially updating the cost function using a gradient-type method to reduce the computational cost while maintaining similar estimation accuracy. Instead of repeatedly solving the optimization problem to obtain the cost value for each data segment, we propose to update the cost value using the parameter estimates from the previous intervals. As the new method sequentially updates the parameter, we name it the sequential algorithm (SE).

We derive the algorithm here based on a heuristic argu-
ment. A rigorous justification for the convergence of the algorithm is given in Section 4. Suppose we have calculated \( \hat{\theta}_{t+1:t-1} \), the approximation to \( \hat{\theta}_{t+1:t} \) that minimizes the cost function based on the data segment \( z_{t+1:t-1} \). We want to find the cost value for the next data segment \( z_{t+1:t} \),

\[
C(z_{t+1:t}) = \min_{\theta \in \Theta} \sum_{i=t+1}^{t} l(z_i, \theta),
\]

where \( \tau \geq 0 \) and \( t \leq T \). Assume that \( l(z, \theta) \) is twice differentiable in \( \theta \). Let \( \hat{\theta}_{t+1:t} \) be the minimizer of (4), which satisfies the first order condition (FOC)

\[
0 = \sum_{i=t+1}^{t} \nabla l(z_i, \hat{\theta}_{t+1:t})
\]

\[
\approx \sum_{i=t+1}^{t} \nabla l(z_i, \hat{\theta}_{t+1:t-1}) + \sum_{i=t+1}^{t-1} \nabla^2 l(z_i, \hat{\theta}_{t+1:t-1})(\hat{\theta}_{t+1:t} - \hat{\theta}_{t+1:t-1}) \approx \nabla l(z_i, \hat{\theta}_{t+1:t-1}) + \sum_{i=t+1}^{t-1} \nabla^2 l(z_i, \hat{\theta}_{t+1:t-1})(\hat{\theta}_{t+1:t} - \hat{\theta}_{t+1:t-1}),
\]

Rearranging the terms, we get

\[
\hat{\theta}_{t+1:t} \approx \hat{\theta}_{t+1:t-1} - \left( \sum_{i=t+1}^{t-1} \nabla^2 l(z_i, \hat{\theta}_{t+1:t-1}) \right)^{-1} \nabla l(z_i, \hat{\theta}_{t+1:t-1}).
\]

As the RHS of (5) does not necessarily fall into the parameter space \( \Theta \), we suggest a projection step. Specifically, let \( P_\Theta(\theta) \) be the projection of any \( \theta \in \mathbb{R}^d \) onto \( \Theta \). The above observation motivates us to consider the following update

\[
\hat{\theta}_{t+1:t} = P_\Theta(\hat{\theta}_{t+1:t-1} - H^{-1}_{t+1:t-1} \nabla l(z_i, \hat{\theta}_{t+1:t-1})),
\]

where \( H_{t+1:t-1} \) is a preconditioning matrix that serves as a surrogate for the second order information \( \sum_{i=t+1}^{t-1} \nabla^2 l(z_i, \hat{\theta}_{t+1:t-1}) \). When the second order information is available, we suggest update the preconditioning matrix through the iteration

\[
H_{t+1:t} = H_{t+1:t-1} + \nabla^2 l(z_i, \hat{\theta}_{t+1:t}).
\]

Alternatively, by the idea of Fisher scoring, we can also update the preconditioning matrix by

\[
H_{t+1:t} = H_{t+1:t-1} + \nabla^2 l(z_i, \hat{\theta}_{t+1:t}),
\]

where \( \nabla^2 l(z_i, \hat{\theta}_{t+1:t}) \) is a preconditioning matrix that

\[
\mathbb{E} \left[ \nabla^2 l(z_i, \hat{\theta}_{t+1:t}) \right] \text{ is a subvector of } z_t \text{ such as the covariates in the regression setting.}
\]

Finally, we approximate \( \hat{\theta}_{t+1:t} \) by \( (t - \tau)^{-1} \sum_{j=t+1}^{t} \hat{\theta}_{t+1:j} \) and the cost value \( C(z_{t+1:t}) \) by

\[
\tilde{C}(z_{t+1:t}) = \sum_{i=t+1}^{t} l(z_i, (t - \tau)^{-1} \sum_{j=t+1}^{t} \hat{\theta}_{t+1:j}).
\]

Algorithm 1 summarizes the details of the algorithm. Also, see the supplement for a graphical illustration.

**Remark 3.1.** When the second order information is available, we suggest setting \( A_i(\hat{\theta}_{t+1:t}) = \nabla^2 l(z_i, \hat{\theta}_{t+1:t}) \) in Algorithm 1, which leads to a type of sequential Newton’s method. On the other hand, our theory in Section 4 allows \( A_i(\hat{\theta}_{t+1:t}) = \mu I/2 \) for some constant \( \mu > 0 \) defined in Assumption A2.

**Remark 3.2.** To improve the optimization, we can add a relatively large momentum term to the gradient, which leads to the update:

\[
P_\Theta(\hat{\theta}_{t+1:t-1} - H^{-1}_{t+1:t-1} \nabla l(z_i, \hat{\theta}_{t+1:t-1}) - \alpha_{t+1:t-1} (\hat{\theta}_{t+1:t-1} - \hat{\theta}_{t+1:t-2})),
\]

where \( \alpha_{t+1:t-1} \) represents the momentum.

**Remark 3.3.** To initialize the estimate \( \hat{\theta}_{t:t} \), we suggest dividing the data into a pre-determined number of segments and estimating the parameters using the data within each segment. We set \( \hat{\theta}_{t:t} \) to be the preliminary estimate using the data in the segment to which \( t \) belongs.

**Remark 3.4.** In practice, a post-processing step is recommended to remove the change-points in \( C(T) \) that are too close to the boundaries and merge those change-points that are too close to each other.

**Remark 3.5.** As suggested by Theorem 4.1, the approximation error of the cost to the true cost value depends on the length of the data segment, which can be relatively large for a small data segment. To address this issue, one can perform exact minimization for data segments with a length less than a pre-specified number.

### 3.2 Generalized linear models

As an illustration of our algorithm, we consider the change-point detection problem in the generalized linear models (GLM). In this case, \( z_t \) contains a response \( y_t \).
Algorithm 1 Sequential Updating Algorithm

• Input the data \( \{z_i\}_{i=1}^T \), the individual cost function \( l(\cdot, \theta) \) and the penalty constant \( \beta \).
• Set \( F(0) = -\beta, C = \emptyset \) and \( R_1 = \{0\} \).
• Iterate for \( t = 1, 2, \ldots, T \):
  1. Initialize \( S_{t:t} = \hat{\theta}_{t:t} \) and \( H_{t:t} \). For \( \tau \in R_t \setminus \{t - 1\} \) with \( t \geq 2 \), perform the following updates:
     \[
     \hat{\theta}_{\tau+1:t} = \mathcal{P}_\Theta(\hat{\theta}_{\tau+1:t-1} - H_{\tau+1:t-1}^{-1} \nabla l(z_t, \hat{\theta}_{\tau+1:t-1})), \\
     H_{\tau+1:t} = H_{\tau+1:t-1} + A_t(\hat{\theta}_{\tau+1:t}), \\
     S_{\tau+1:t} = S_{\tau+1:t-1} + \hat{\theta}_{\tau+1:t}.
     \]
  2. For each \( \tau \in R_t \), compute
     \[
     \tilde{C}(\mathbf{z}_{\tau+1:t}) = \sum_{i=\tau+1}^{t} l(z_i, (t-\tau)^{-1}S_{\tau+1:t}).
     \]
  3. Calculate
     \[
     F(t) = \min_{\tau \in R_t} \left\{ F(\tau) + \tilde{C}(\mathbf{z}_{\tau+1:t}) + \beta \right\}, \\
     \tau^* = \text{argmin}_{\tau \in R_t} \left\{ F(\tau) + \tilde{C}(\mathbf{z}_{\tau+1:t}) + \beta \right\}.
     \]
  4. Let \( C(t) = \{C(\tau^*), \tau^*\} \).
  5. Set
     \[
     R_{t+1} = \{\tau \in R_t \cup \{t\} : F(\tau) + C(\mathbf{z}_{\tau+1:t}) \leq F(t)\}.
     \]
• Output \( C(T) \).

and a set of predictors/covariates \( x_i \). Suppose \( y_i \) follows a distribution in the canonical exponential family

\[
 f(y_i; \gamma_i, \phi) = \exp \left\{ \frac{y_i\gamma_i - b(\gamma_i)}{w^{-1} \phi} + c(y_i, \phi) \right\},
\]

where \( \gamma_i \) is the canonical parameter, \( \phi \) is the dispersion parameter and \( w \) is some known weight. The mean of \( y_i \) is related to \( x_i^T \theta_i \) via \( g(E[y_i]) = g(\nabla b(\gamma_i)) = x_i^T \theta_i \), where \( g \) is a known link function. Suppose the observations from the time point \( \tau + 1 \) to \( t \) share the same parameter \( \theta \), i.e., \( \theta_{\tau+i} = \theta \) for \( \tau + 1 \leq i \leq t \). When \( \phi \) is known, we let

\[
 l(z_i, \theta) = \frac{y_i\gamma_i - b(\gamma_i)}{w^{-1} \phi} - c(y_i, \phi), \quad \tau + 1 \leq i \leq t.
\]

Some algebra yields that

\[
 \nabla l(z_i, \hat{\theta}_{\tau+1:t}) = -\frac{a}{\phi v(\hat{\mu}_{\tau+1:t})g'(\hat{\mu}_{\tau+1:t})}(y_i - \hat{\mu}_{\tau+1:t})x_i, \\
 I(x_i, \hat{\theta}_{\tau+1:t}) = \frac{a}{\phi v(\hat{\mu}_{\tau+1:t})[g'(\hat{\mu}_{\tau+1:t})]^2}x_i^T x_i.
\]

where \( g(\hat{\mu}_{\tau+1:t}) = x_i^T \hat{\theta}_{\tau+1:t} \) and \( v \) is related to the variance of \( y_i \) through \( \text{var}(y_i) = \phi w^{-1} v(\mu_i) \). In the cases of the logistic and Poisson regressions, we have \( \omega = 1 \) and \( g'(\mu) v(\mu) = 1 \). Hence for the logistic regression,

\[
 \nabla l(z_i, \hat{\theta}_{\tau+1:t}) = -\left( y_i - \frac{e^{x_i^T \hat{\theta}_{\tau+1:t}}}{1 + e^{x_i^T \hat{\theta}_{\tau+1:t}}} \right) x_i, \\
 I(x_i, \hat{\theta}_{\tau+1:t}) = (\hat{\mu}_{\tau+1:t}/(1 - \hat{\mu}_{\tau+1:t}))x_i^T x_i.
\]

While for the Poisson regression,

\[
 \nabla l(z_i, \hat{\theta}_{\tau+1:t}) = -\left( y_i - e^{x_i^T \hat{\theta}_{\tau+1:t}} \right) x_i, \\
 I(x_i, \hat{\theta}_{\tau+1:t}) = \hat{\mu}_{\tau+1:t} x_i^T x_i.
\]

We shall investigate the performance of the corresponding algorithms in Section 5.

3.3 Sequential proximal gradient descent

In this section, we extend our algorithm to handle the case where the cost function value results from solving a penalized optimization problem. More precisely, let us consider

\[
 C(\mathbf{z}_{\tau+1:t}) = \min_{\theta \in \Theta} \sum_{i=\tau+1}^{t} l(z_i, \theta) + \lambda_{\tau+1:t} \text{pen}(\theta), \quad (6)
\]
where the penalty term enforces a constraint on the parameter $\theta$ (e.g., the smoothness or sparsity constraint) and $\lambda_{1:t} > 0$ is allowed to vary over data segments. Let

$$\text{Prox}(a; \lambda) = \arg\min_z \frac{1}{2\lambda} \|z - a\|^2 + \text{pen}(z)$$

be the proximal operator associated with the penalty term. For $a = (a_1, \ldots, a_d)$ with $a_i \neq 0$ and $\lambda = (\lambda_1, \ldots, \lambda_d)$ with $\lambda_i > 0$, we write $a^{-1} = (a_1^{-1}, \ldots, a_d^{-1})$ and $\text{Prox}(a; \lambda) = (\text{Prox}(a_1; \lambda_1), \ldots, \text{Prox}(a_d; \lambda_d))$. We update the parameter estimate by $\hat{\theta}_{1:t} = \text{Prox}_{\theta} (\hat{\theta}_{1:t-1})$ with

$$\hat{\theta}_{1:t} = \text{Prox}_{\lambda_{1:t}} \left( \hat{\theta}_{1:t-1} - H_{1:t-1}^{-1} \nabla l(z_t, \hat{\theta}_{1:t-1} ; \lambda_{1:t}) \right),$$

where $\|H\|_2$ denotes the spectral norm of $H$ (one can also replace $\|H\|_2$ by the Frobenius norm of $H$). An example here is the Lasso regression, where $z_i = (x_i, y_i) \in \mathbb{R} \times \mathbb{R}^d$ and the objective function in (6) can be written as

$$\frac{1}{2} \sum_{i=1}^d \|y_i - x_i^\top \theta\|^2 + \lambda_{1:t} \sum_{i=1}^d |\theta_i|$$

with $\theta = (\theta_1, \ldots, \theta_d)$. In this case, $\text{Prox}(a; \lambda) = \text{sign}(a) \max(|a| - \lambda, 0)$ is the soft thresholding operator.

### 3.4 Choice of the penalty constant $\beta$

Our method aims to solve the following $l_0$ penalized optimization problem approximately

$$\min_k \min_\tau \left\{ \sum_{j=0}^k \min_{\theta \in \Theta} \sum_{i=\tau_{j+1}}^{\tau_{j+1}} l(z_i, \theta) + f(k, T) \right\}$$

where we simultaneously optimize over the number of change-points $k$, the locations of change-points $\tau$, and the parameters within each segment. With $k$ change-points that divide the data sequence into $k + 1$ segments, the total number of parameters is $(k + 1)d + k$, where $(k + 1)d$ counts the number of parameters from the $k + 1$ segments and $k$ corresponds to the number of change-points. We recommend setting $f(k, T) = \{(k + 1)d + k\} \log(T)/2$ or equivalently

$$\beta_T = (d + 1) \log(T)/2,$$

which leads to the BIC criterion.

### 4 CONVERGENCE ANALYSIS

To understand why our method works, it is crucial to investigate how well the sequential gradient method can approximate the cost value for each data segment. To be clear, let us focus on the segment $z_1:n$ with $1 \leq n \leq T$. Let $\hat{\theta}^* = \arg\min_{\theta \in \Theta} F_n(\theta)$, where $F_n(\theta) = n^{-1} \sum_{i=1}^n l(z_i, \theta)$. Recall that given $\hat{\theta}_1$ (which only depends on $z_1$), we have the following updating scheme for finding an approximation to $\hat{\theta}^*$

$$\hat{\theta}_{1:t} = \mathcal{P}_\Theta (\hat{\theta}_{1:t-1} - H_{1:t-1}^{-1} \nabla l(z_t, \hat{\theta}_{1:t-1})), \quad 2 \leq t \leq n,$$

where $H_{1:t-1}$ is a preconditioning matrix that only depends on $z_1, \ldots, z_{t-1}$. Throughout this section, we write $\hat{\theta}_t = \hat{\theta}_t, H_{1:t} = H_t$ and $l(z_t, \theta) = l_t(\theta)$ for the ease of notation. Our analysis here focuses on the SeGD.

**Definition 4.1 (Strong convexity).** A differentiable function $F$ is said to be $\mu$-strongly convex, with $\mu > 0$, if and only if

$$F(\eta) \geq F(\theta) + \nabla F(\theta)^\top (\eta - \theta) + \frac{\mu}{2} \|\eta - \theta\|^2.$$

**Definition 4.2 (Smoothness).** A differential function $F$ is said to be $L$-smooth if

$$|F(\eta) - F(\theta) - \nabla F(\theta)^\top (\eta - \theta)| \leq \frac{L}{2} \|\eta - \theta\|^2,$$

for any $\eta, \theta$ in the domain of $F$.

We aim to quantify the difference $F_n(n^{-1} \sum_{t=1}^n \hat{\theta}_t) - F_n(\hat{\theta}^*)$ and derive the convergence rate. To this end, we make the following assumptions.

A1. There is an unknown change-point $1 \leq \xi < T$. The first $\xi$ observations are assigned with their time locations through a random permutation $(\sigma(1), \ldots, \sigma(\xi))$ while the last $n - \xi$ observations are assigned with the time locations through a random permutation $(\sigma(\xi + 1), \ldots, \sigma(T))$;

A2. $F_n$ is $\mu$-strongly convex;

A3. $H_t = \eta_t I$ with $\eta_t = t\mu/2$;

A4. $\sup_{1 \leq t \leq n, \theta \in \Theta} \|\nabla l_t(\theta)\| \leq C$ for some constant $C > 0$;

A5. $\Theta$ is a compact set;

A6. $l_t(\theta) = f(x_t^\top \theta, y_t) + r(\theta)$, where $f(a, y)$ is $L_1$-Lipschitz and $L_2$-smooth in $a$ for any given $y, \|x_t\|$ is bounded almost surely, and $r$ is some fixed function.
Theorem 4.1. Let $E^*$ be the expectation with respect to the random permutation $\sigma$ conditional on the observed data values $\{z_i\}_{i=1}^n$. Under Assumptions A1-A6, we have

$$E^* \left[ F_n \left( \sum_{i=1}^n \hat{\theta}_i - F_n(\hat{\theta}^*) \right) \right] \leq \frac{c \log(n)}{n},$$

for some constant $c$ that depends on $\mu, C, L_1$ and $L_2$.

Remark 4.1. The proof of Theorem 4.1 is given in the supplement. In Assumption A1, we assume that there is a single change point. Similar arguments can be used to handle the cases of no change point and multiple change-points.

Remark 4.2. Assumptions A2, A4 and A6 are fulfilled for logistic models when $\|x_i\|$ is bounded almost surely and the smallest eigenvalue of $n^{-1} \sum_{i=1}^n x_i x_i^\top$ is bounded away from zero almost surely. We also remark that the same conclusion can be justified when Assumptions A2, A4, and A6 hold with probability converging to one by using the conditioning argument.

Remark 4.3. Under Assumption A1, we can cast our method into a type of SGD without replacement sampling and employ the related techniques (Shamir, 2016) to prove Theorem 4.1.

5 NUMERICAL STUDIES

In this section, we apply the PELT method and the proposed SE method to several simulated data sets and a real data set to compare their estimation accuracy measured by the rand index and the computational time (in seconds).

5.1 Logistic regression

We first consider the GLM with piecewise constant regression coefficients. The details for implementing SE under GLM has been described in Section 3.2. We only present the results for logistic regression in the main paper. Results for the cases of no change-point and the Poisson regression are relegated to the supplement.

Consider the logistic regression model:

$$y_i \sim \text{Bernoulli} \left( \frac{1}{1 + e^{-x_i^\top \theta_i}} \right), \quad x_i \sim N_d(0, \Sigma)$$

with \(\Sigma = (0.9^{|i-j|})_{d \times d}, 1 \leq i \leq T.

Throughout the simulations, we set $T = 1500, d = 50$ and vary the value of $\theta_i$ leading to different magnitudes of change. Let $\delta_d \in \mathbb{R}^d$ be the difference between the coefficients before and after a change-point. We choose $\delta_d$ such that $M(\delta_d) := \delta_d^\top \Sigma \delta_d \in \{0.36, 0.81, 1.96\}$ corresponding to small, medium and large magnitudes of change respectively. We remark that the results are not sensitive to the specific choice of $\delta_d$ as long as $M(\delta_d)$ is held at the same level. For each configuration, we shall consider the number of change-points equal to 1, 3 and 5. The detailed simulation settings for each case are given in the supplement. As seen from Figure 1, SE achieves the same estimation accuracy in terms of the rand index as PELT does. SE could be around 350 times faster than PELT, making SE a highly scalable method in practice. For example, when the magnitude of change is small with three change-points for $d = 5$, SE finished the analysis within 8.77 seconds while it took 3133.58 seconds for PELT to get the same result.

5.2 Penalized linear regression

We consider the linear model

$$y_i = x_i^\top \theta_i + \epsilon_i, \quad x_i \sim N_d(0, \Sigma) \quad \text{with} \quad \Sigma = 0.5I_{d \times d} \quad \text{and} \quad \epsilon_i \sim N(0, 0.5), \quad 1 \leq i \leq T.$$

Set $T = 1500, d = 50$ and $s \in \{1, 3, 5\}$, where $s$ is the number of non-zero components of the $d$ dimensional regression coefficients $\theta_i$. The locations of the nonzero components are randomly selected. The magnitude of change is reflected by the difference between the $\theta_i$ values before and after the change-point(s). The values of the non-zero components of the regression coefficients $\theta_i$ within each odd-numbered segment partitioned by the change-points $\{\tau_i\}$ (i.e. $1 \leq i \leq \tau_1$ and $\tau_j < i \leq \tau_{j+1}$ when $j$ is even) are set to be 1. For the even-numbered segment (i.e. $\tau_j < i \leq \tau_{j+1}$ when $j$ is odd), the non-zero coefficients are generated from $N(1, \delta)$ with $\delta \in \{0.1, 0.4, 1\}$ corresponding to small, medium and large magnitudes of changes, respectively. Like the GLM simulation settings, we shall consider 1, 3, and 5 change-points for different combinations of $s$ and magnitude of change. We consider the cost function

$$C(z_{\tau+1:d}) = \min_{\theta \in \Theta} \frac{1}{2} \sum_{i=\tau+1}^t \|y_i - x_i^\top \theta\|^2 + \lambda_{\tau+1:d} \sum_{i=1}^d |\theta_i|,$$

where $\lambda_{\tau+1:d} = \hat{\sigma} \sqrt{2 \log(d)/(t-\tau)}$ with $\hat{\sigma}$ being a preliminary estimate of the noise level. In particular, we divide the data into ten segments, estimate the noise level within each segment using Lasso, and set $\hat{\sigma}$ to be the average of these estimates. We implement both PELT and SeGD (without the second order information) in this case. As seen from Figures 2, SE achieves competitive accuracy compared to PELT in most cases with lower computational cost. For instance, when $s = 1$ and there is only one medium change-point, SE is about 8 times...
faster than PELT.

5.3 A real data example

We illustrate the method using a dataset from the immune correlates study of Maternal To Child Transmission (MTCT) of HIV-1 (Fong et al., 2015). The data set contains three variables: the 0/1 response $y_i$ indicating whether HIV transits from mother to child (79 HIV-transmitting mothers and 157 non-transmitting mothers, leading to $T = 236$), childbirth delivery type $x_i$ (C-section/Vaginal), and the NAb score $z_i$ measuring the amount and breadth of neutralizing antibodies. We consider the following change point/threshold model: $y_i \sim \text{Bernoulli}(p_i)$ with

$$
\log \left( \frac{p_i}{1 - p_i} \right) = \bar{x}_i^T \beta(z_i), \quad \bar{x}_i = (1, x_i)^T,
$$

where $\beta(z) = \sum_{j=0}^{k} \beta_j 1\{a_j \leq z < a_{j+1}\}$ with $\min_j z_i = a_0 < a_1 < a_2 < \cdots < a_{k+1} = +\infty$. In words, the regression coefficient is a piece-wise constant function of the NAb score.

To implement PELT and SE, we first sort the data in descending order according to the NAb score. Both methods find a single change that corresponds to the NAb score. Both methods find a single change that corresponds to the NAb score. SE finishes the analysis in 0.62 seconds, while PELT takes 22 seconds to get the same result.

6 CONCLUDING REMARKS

We point out three possible strategies namely accelerated SeGD, multiple epochs and backward updating scheme to improve estimation accuracy by speeding up the convergence in SeGD. In Theorem 4.1, we have shown that the difference between $F_n \left( n^{-1} \sum_{t=1}^{n} \hat{\theta}_t \right)$ and the target cost value is of the order $O(\log(n)/n)$ with $n$ being the length of the segment. An interesting future direction is to develop an accelerated sequential gradient method to improve the convergence rate. Motivated by the accelerated SGD, we may consider the following update strategy:

$$
\beta_{r+1:t-1} = \alpha \hat{\beta}_{r+1:t-1} + (1 - \alpha) v_{r+1:t-1},
\hat{\theta}_{r+1:t} = \beta_{r+1:t-1} - H_{r+1:t-1}^{-1} \nabla L(f(x_t, \beta_{r+1:t-1}), y_t),
\gamma_{r+1:t-1} = \beta_{r+1:t-1} + (1 - \beta) v_{r+1:t-1},
$$

$$
v_{r+1:t} = \gamma_{r+1:t-1} - \zeta H_{r+1:t-1}^{-1} \nabla L(f(x_t, \beta_{r+1:t-1}), y_t),
$$

where we set $v_{r+1:t+1} = \hat{\beta}_{r+1:t+1}$ and $\alpha, \beta, \zeta$ are tuning parameters. An in-depth analysis of this algorithm is left for future research. Another way to improve the convergence is by using multiple epochs/passes over the data points in each segment. Algorithm 1 only uses each data point once (one-pass) in updating the parameter estimates for a particular segment. Using multiple epochs has been shown to improve the rate of convergence (Nagaraj et al., 2019). In the supplement, we describe such an extension of our algorithm. Finally, one can introduce a backward updating scheme. Together with the forward scheme, we can update $\hat{\theta}_{r:t}$ using the estimates based on nearby segments $\theta_{r+1:t}$ and $\theta_{r:t-1}$.

Figure 1: Average rand index (upper panel) and computational time (lower panel) for SE and PELT under logistic regression models with different number of change-points (1, 3, 5), magnitude of changes (small, medium and large) and dimension $d$ (1, 3, 5). Error bars represent the 95% CIs ($\pm 2 \times$ standard error).
Figure 2: Average rand index (upper panel) and computational time (lower panel) for SE and PELT under penalized linear regression models with different number of change-points (1, 3, 5), magnitude of changes (small, medium and large) and the number of non-zero coefficients $s$ (1, 3, 5). Error bars represent the 95% CIs ($\pm 2 \times$ standard error).

Acknowledgements

This work was supported in part by NSF DMS-2113359.

References


A APPENDIX

A.1 Technical details

Proof of Theorem 4.1. Write \( \sigma_{a,b} = (\sigma(a), \ldots, \sigma(b)) \) for \( 1 \leq a \leq b \leq n \). By the definition of the algorithm,

\[
E^*[||\hat{\theta}_t - \hat{\theta}^*||^2] = E^*[||\mathcal{P}_a(\hat{\theta}_{t-1} - H_{t-1}^{-1} \nabla l_{\sigma_t}(\hat{\theta}_{t-1})) - \hat{\theta}^*||^2]
\]

\[
\leq E^*[||\hat{\theta}_{t-1} - H_{t-1}^{-1} \nabla l_{\sigma_t}(\hat{\theta}_{t-1}) - \hat{\theta}^*||^2]
\]

\[
\leq E^*[||\hat{\theta}_{t-1} - \hat{\theta}^*||^2 + \eta_{t-1}^{-2}C^2 - 2\eta_{t-1}^{-1} E^*[\nabla l_{\sigma_t}(\hat{\theta}_{t-1})^T (\hat{\theta}_{t-1} - \hat{\theta}^*)]
\]

\[
= E^*[||\hat{\theta}_{t-1} - \hat{\theta}^*||^2 + \eta_{t-1}^{-2}C^2 - 2\eta_{t-1}^{-1} E^*[\{\nabla l_{\sigma_t}(\hat{\theta}_{t-1}) - \nabla F_n(\hat{\theta}_{t-1})\}^T (\hat{\theta}_{t-1} - \hat{\theta}^*)]
\]

\[
- 2\eta_{t-1}^{-1} E^*[\nabla F_n(\hat{\theta}_{t-1})^T (\hat{\theta}_{t-1} - \hat{\theta}^*)].
\]

By Assumption A2, namely the strong convexity, we have

\[
\nabla F_n(\hat{\theta}_{t-1})^T (\hat{\theta}_{t-1} - \hat{\theta}^*) \geq F_n(\hat{\theta}_{t-1}) - F_n(\hat{\theta}^*) + \frac{\mu}{2} ||\hat{\theta}^* - \hat{\theta}_{t-1}||^2,
\]

which implies that

\[
E^*[||\hat{\theta}_t - \hat{\theta}^*||^2] \leq E^*[||\hat{\theta}_{t-1} - \hat{\theta}^*||^2 + \eta_{t-1}^{-2}C^2 - 2\eta_{t-1}^{-1} E^*[\{\nabla l_{\sigma_t}(\hat{\theta}_{t-1}) - \nabla F_n(\hat{\theta}_{t-1})\}^T (\hat{\theta}_{t-1} - \hat{\theta}^*)]
\]

\[
- 2\eta_{t-1}^{-1} E^*[\nabla F_n(\hat{\theta}_{t-1})^T (\hat{\theta}_{t-1} - \hat{\theta}^*)] + \eta_{t-1}^{-1} \mu ||\hat{\theta}^* - \hat{\theta}_{t-1}||^2.
\]

Re-arranging the terms, we get

\[
E^*[F_n(\hat{\theta}_{t-1}) - F_n(\hat{\theta}^*)] \leq \left(\frac{\eta_{t-1}}{2} - \frac{\mu}{2}\right) E^*[||\hat{\theta}_{t-1} - \hat{\theta}^*||^2] - \frac{\eta_{t-1}^{-1}}{2} E^*[||\hat{\theta}_t - \hat{\theta}^*||^2] + \frac{C^2}{2\eta_{t-1}}
\]

\[
+ E^*[\{\nabla F_n(\hat{\theta}_{t-1}) - \nabla l_{\sigma_t}(\hat{\theta}_{t-1})\}^T (\hat{\theta}_{t-1} - \hat{\theta}^*)].
\]

To deal with the last term on the RHS, we consider two cases, namely \( t \leq \xi \) and \( t > \xi \), separately. Let us first consider the case \( t > \xi \). Conditional on \( \sigma_{1:t-1} \) and the data values \( \{z_i\}_{i=1}^n \), \( \hat{\theta}_{t-1} \) is fixed. Note that \( \hat{\theta}^* \) is independent of any permutation of the data. Moreover, \( \sigma_1 \) is uniformly distributed on \( \{1, 2, \ldots, n\} \setminus \sigma_{1:t-1} = \{\sigma_t, \ldots, \sigma_n\} \). We also note that

\[
E^*[\nabla F_n(\hat{\theta}_{t-1})|\sigma_{1:t-1}] = \frac{1}{n} \sum_{j=1}^{t-1} \nabla l_{\sigma_j}(\hat{\theta}_{t-1}) + \frac{1}{n} \sum_{j=t}^{n} E^*[\nabla l_{\sigma_j}(\hat{\theta}_{t-1})|\sigma_{1:t-1}]
\]

\[
= \frac{1}{n} \sum_{j=1}^{t-1} \nabla l_{\sigma_j}(\hat{\theta}_{t-1}) + \frac{1}{n} \sum_{j=t}^{n} \nabla l_{\sigma_j}(\hat{\theta}_{t-1})
\]

\[
= \nabla F_n(\hat{\theta}_{t-1}).
\]

Using these facts, we have

\[
E^*[\{\nabla F_n(\hat{\theta}_{t-1}) - \nabla l_{\sigma_t}(\hat{\theta}_{t-1})\}^T (\hat{\theta}_{t-1} - \hat{\theta}^*)]
\]

\[
= E^*[E^*[\{\nabla F_n(\hat{\theta}_{t-1}) - \nabla l_{\sigma_t}(\hat{\theta}_{t-1})\}^T (\hat{\theta}_{t-1} - \hat{\theta}^*)|\sigma_{1:t-1}]]
\]

\[
= E^*[\left\{\nabla F_n(\hat{\theta}_{t-1}) - (n - t + 1)^{-1} \sum_{j=t}^{n} \nabla l_{\sigma_j}(\hat{\theta}_{t-1})\right\}^T (\hat{\theta}_{t-1} - \hat{\theta}^*)]
\]

\[
= \frac{t-1}{n} E^*[\left\{\nabla l_{1:t-1}(\hat{\theta}_{t-1}) - \nabla l_{1:n}(\hat{\theta}_{t-1})\right\}^T (\hat{\theta}_{t-1} - \hat{\theta}^*)]
\]
where the first inequality follows from the fact that \( w \) with \( n \).

Next we consider the case where \( t \leq \xi \). Conditional on \( \sigma_{1:t-1}, \sigma_t \) is uniformly distributed on \( \{1, 2, \ldots, \xi\} \setminus \{\sigma_{1:t-1} = \{\sigma_, \ldots, \sigma_{\xi}\} \). Similar arguments show that

\[
E^*[\{\nabla F_{n}(\hat{\theta}_{t-1}) - \nabla l_{\sigma_t}(\hat{\theta}_{t-1})\}^\top(\hat{\theta}_{t-1} - \hat{\theta}^*)]
\]

\[
= E^*[\{\nabla F_{n}(\hat{\theta}_{t-1}) - (\xi - t + 1)^{-1}\sum_{j=t}^{\xi} \nabla l_{\sigma_j}(\hat{\theta}_{t-1})\}^\top(\hat{\theta}_{t-1} - \hat{\theta}^*)]
\]

\[
= \frac{n - \xi + t - 1}{n} E^*[\|\hat{\theta}_{t-1} - \hat{\theta}^*\|^2 \left\{\nabla l_{- \xi}(\hat{\theta}_{t-1}) - \nabla l_{\xi}(\hat{\theta}_{t-1})\right\}^\top(\hat{\theta}_{t-1} - \hat{\theta}^*)]
\]

\[
\leq \frac{n - \xi + t - 1}{n} \sqrt{E^*[\|\hat{\theta}_{t-1} - \hat{\theta}^*\|^2]} \sqrt{E^*[\left(\sup_{\theta \in \Theta} \left\{\nabla l_{- \xi}(\theta) - \nabla l_{\xi}(\theta)\right\}^\top(\theta - \hat{\theta}^*)\right)^2]},
\]

where \( \nabla l_{- \xi}(\theta) = (n - \xi + t - 1)^{-1}\sum_{j=1}^{t-1} \nabla l_j(\theta) + \sum_{j=\xi+1}^{n} \nabla l_j(\theta) \). As

\[
\nabla l_{- \xi}(\theta) - \nabla l_{\xi}(\theta) = w(\nabla l_{t-1}(\theta) - \nabla l_{t}(\theta)) + (1 - w)(\nabla l_{\xi+1:n}(\theta) - \nabla l_{\xi}(\theta))
\]

with \( w = (t - 1)/(n - \xi + t - 1) \), we have

\[
E^*[\left(\sup_{\theta \in \Theta} \left\{\nabla l_{\xi}(\theta) - \nabla l_{- \xi}(\theta)\right\}^\top(\theta - \hat{\theta}^*)\right)^2]
\]

\[
\leq 2w^2 E^*[\left(\sup_{\theta \in \Theta} \left\{\nabla l_{\xi}(\theta) - \nabla l_{t-1}(\theta)\right\}^\top(\theta - \hat{\theta}^*)\right)^2]
\]

\[
+ 2(1 - w)^2 E^*[\left(\sup_{\theta \in \Theta} \left\{\nabla l_{\xi}(\theta) - \nabla l_{\xi+1:n}(\theta)\right\}^\top(\theta - \hat{\theta}^*)\right)^2].
\]
Similar argument as before gives
\[
E^* \left[ \left( \nabla F_n(\hat{\theta}_{t-1}) - \nabla l_{\sigma_i}(\hat{\theta}_{t-1}) \right)^\top (\hat{\theta}_{t-1} - \hat{\theta}^*) \right] \\
\leq \frac{\mu}{4} E^* \left[ \|\hat{\theta}_{t-1} - \hat{\theta}^*\|^2 \right] + \frac{4C^2}{\mu n^2} \left\{ \frac{(t-1)^2}{\xi - t + 1} + 1 \right\} + \frac{(n-\xi)^2}{\xi - t + 1} + n - \xi \right\}. 
\]

Using the above bounds and averaging over 2, \ldots, n + 1 of (9), we obtain
\[
E^* \left[ n^{-1} \sum_{i=1}^{n} F_n(\hat{\theta}_i) - F_n(\hat{\theta}^*) \right] \\
\leq \frac{1}{2n} \sum_{i=1}^{n} \left( \eta_t - \mu/2 - \eta_{t-1} \right) E^* \left[ \|\hat{\theta}_t - \hat{\theta}^*\|^2 \right] + \frac{C^2}{2n} \sum_{t=1}^{n} \frac{1}{\eta_t} \\
+ \frac{4C^2}{\mu n^2} \sum_{t=2}^{\xi} \left\{ \frac{(t-1)^2}{\xi - t + 1} + 1 \right\} + \frac{(n-\xi)^2}{\xi - t + 1} + n - \xi \right\} \\
+ \frac{4C^2}{\mu n^2} \sum_{t=\xi+1}^{n+1} \left( t - t + \frac{(t-1)^2}{n - t + 1} \right), 
\]
where \( \eta_0 = 0 \) and we have replaced the dummy variable \( t - 1 \) with \( t \) in the summation. Note that
\[
\sum_{t=2}^{\xi} \left\{ \frac{(t-1)^2}{\xi - t + 1} + 1 \right\} + \frac{(n-\xi)^2}{\xi - t + 1} + n - \xi \right\} \\
= \sum_{t=1}^{n} t + \sum_{t=1}^{\xi-1} \left\{ \frac{t^2}{\xi - t} + \frac{(n-\xi)^2}{\xi - t} + n - \xi \right\} + \sum_{t=\xi}^{n} \frac{t^2}{n - t} \\
\leq C'n^2 \log(n), 
\]
for \( C' > 0 \), where we have used the following facts
\[
\sum_{t=1}^{n} t = \frac{(n + 1)n}{2} \leq C_1 n^2, \\
\sum_{t=1}^{\xi-1} \frac{(n-\xi)^2}{\xi - t} \leq C_2 n^2 \log(n), \\
\sum_{t=1}^{\xi-1} \frac{t^2}{\xi - t} \leq (\xi - 1) \sum_{t=1}^{\xi-1} 1 \leq C_3 n^2 \log(n), \\
\sum_{t=\xi}^{n} \frac{t^2}{n - t} \leq n^2 \sum_{t=\xi}^{n} \frac{1}{n - t} \leq C_4 n^2 \log(n), 
\]
for some positive constants \( C_i \) with \( 1 \leq i \leq 4 \). Finally, using the definition \( \eta_t = \frac{\mu}{2} \) and the convexity of \( F_n \), we have
\[
E^* \left[ F_n \left( n^{-1} \sum_{i=1}^{n} \hat{\theta}_i \right) - F_n(\hat{\theta}^*) \right] \leq E^* \left[ n^{-1} \sum_{i=1}^{n} F_n(\hat{\theta}_i) - F_n(\hat{\theta}^*) \right] \leq \frac{c \log(n)}{n}, 
\]
for some \( c > 0 \). The conclusion thus follows.

The result below follows from Corollary 3 of Shamir (2016), which is proved using the transductive learning theory.
Lemma A.1. Under Assumptions A5-A6, we have

$$E^a \left[ \left( \sup_{\theta \in \Theta} \{ \nabla L_{1:a}(\theta) - \nabla L_{a+1:n}(\theta) \}^\top \frac{\theta - \hat{\theta}^*}{\|\theta - \hat{\theta}^*\|} \right)^2 \right] \leq C_1^2 \left( \frac{1}{\sqrt{a}} + \frac{1}{\sqrt{n-a}} \right)^2,$$

where $C_1$ is some constant that depends on $L_1$, $L_2$ and $D$ (the diameter of $\Theta$).

A.2 Simulation settings

The tables below summarize the values of the regression coefficients $\theta_t$ (for both the logistic and Poisson regressions) within each segment partitioned by the change-point locations $\{\tau_i\}$.

- Single change-point ($k = 1$): $\tau_1 = 750$ and

<table>
<thead>
<tr>
<th>$d = 1$</th>
<th>$d = 3$</th>
<th>$d = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1 \leq i \leq \tau_1$</td>
<td>1.2</td>
<td>(1, 1.2, -1)</td>
</tr>
<tr>
<td>$\tau_1 &lt; i \leq T$</td>
<td>$1.2 + \delta_1$</td>
<td>(1, 1.2, -1) + $\delta_3$</td>
</tr>
</tbody>
</table>

- Three change-points ($k = 3$): $\tau_1 = 375$, $\tau_2 = 750$, $\tau_3 = 1125$ and

<table>
<thead>
<tr>
<th>$d = 1$</th>
<th>$d = 3$</th>
<th>$d = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1 \leq i \leq \tau_1$</td>
<td>1.2</td>
<td>(1, 1.2, -1)</td>
</tr>
<tr>
<td>$\tau_1 &lt; i \leq \tau_2$</td>
<td>$1.2 + \delta_1$</td>
<td>(1, 1.2, -1) + $\delta_3$</td>
</tr>
<tr>
<td>$\tau_2 &lt; i \leq \tau_3$</td>
<td>1.2</td>
<td>(1, 1.2, -1)</td>
</tr>
<tr>
<td>$\tau_3 &lt; i \leq T$</td>
<td>$1.2 - \delta_1$</td>
<td>(1, 1.2, -1) - $\delta_3$</td>
</tr>
</tbody>
</table>

- Five change-points ($k = 5$): $\tau_1 = 250$, $\tau_2 = 500$, $\tau_3 = 750$, $\tau_4 = 1000$, $\tau_5 = 1250$ and

<table>
<thead>
<tr>
<th>$d = 1$</th>
<th>$d = 3$</th>
<th>$d = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1 \leq i \leq \tau_1$</td>
<td>1.2</td>
<td>(1, 1.2, -1)</td>
</tr>
<tr>
<td>$\tau_1 &lt; i \leq \tau_2$</td>
<td>$1.2 + \delta_1$</td>
<td>(1, 1.2, -1) + $\delta_3$</td>
</tr>
<tr>
<td>$\tau_2 &lt; i \leq \tau_3$</td>
<td>1.2</td>
<td>(1, 1.2, -1)</td>
</tr>
<tr>
<td>$\tau_3 &lt; i \leq \tau_4$</td>
<td>$1.2 - \delta_1$</td>
<td>(1, 1.2, -1) - $\delta_3$</td>
</tr>
<tr>
<td>$\tau_4 &lt; i \leq \tau_5$</td>
<td>1.2</td>
<td>(1, 1.2, -1)</td>
</tr>
<tr>
<td>$\tau_5 &lt; i \leq T$</td>
<td>$1.2 + \delta_1$</td>
<td>(1, 1.2, -1) + $\delta_3$</td>
</tr>
</tbody>
</table>

A.3 Multiple epochs

This section describes an extension of Algorithm 1 to allow multiple epochs. Specifically, we will use each data point $K \geq 1$ times in updating the parameter estimates for a particular segment. The details are summarized in Algorithm 2 below. SE with multiple epochs/passes allows more efficient use of the data with an additional computational expense controlled by $K$. We leave a detailed analysis of this trade-off between statistical and computational efficiencies for future investigation.
Algorithm 2 Sequential Updating Algorithm with Multiple Epochs

- Input the data \( \{z_i\}_{i=1}^T \), the individual cost function \( l(\cdot, \theta) \), the penalty constant \( \beta \) and the number of epochs \( K \).
- Set \( F(0) = -\beta, \mathcal{C} = \emptyset \) and \( R_1 = \{0\} \).
- Iterate for \( t = 1, 2, \ldots, T \):
  1. Initialize \( S_{t,t}^{(K)} = \hat{\theta}^{(K)}_{t,t} \) and \( H_{t,t}^{(K)} \). For \( \tau \in R_t \setminus \{t - 1\} \), perform the update
     \[
     \hat{\theta}_{t+1:1:t}^{(1,t)} = P_{\Theta}(\hat{\theta}_{t+1:t-1}^{(K)}, - H_{t+1:t-1}^{(K)} \nabla l(z_t, \hat{\theta}_{t+1:t-1}^{(K)})),
     \]
     \[
     H_{t+1:1:t}^{(1,t)} = H_{t+1:t-1}^{(K)} + A(\hat{\theta}_{t+1:t}^{(1,t)}).
     \]
   Next for \( k = 2, \ldots, K \), perform the update
   \[
   \hat{\theta}_{t+1:1:t}^{(k,t)} = P_{\Theta}(\hat{\theta}_{t+1:1:t-1}^{(k-1,t)}, - H_{t+1:t-1}^{(k-1,t)} \nabla l(z_t, \hat{\theta}_{t+1:1:t-1}^{(k-1,t)})),
   \]
   \[
   H_{t+1:1:t}^{(k,t)} = H_{t+1:1:t-1}^{(k-1,t)} + A(\hat{\theta}_{t+1:1:t}^{(k,t)}),
   \]
   over \( j = \tau + 1, \ldots, t \), where \( (\hat{\theta}_{t+1:1:t}^{(k,t)}, H_{t+1:1:t}^{(k,t)}) = (\hat{\theta}_{t+1:1:t}^{(k-1,t)}, H_{t+1:1:t}^{(k-1,t)}), \) Set \( \hat{\theta}_{t+1:1:t}^{(K,t)} = \hat{\theta}_{t+1:1:t}^{(K-1,t)}, H_{t+1:1:t}^{(K,t)} = H_{t+1:1:t}^{(K-1,t)} \) and
   \[
   S_{t+1:1:t}^{(K)} = S_{t+1:t-1}^{(K)} + \hat{\theta}_{t+1:1:t}^{(K)}.
   \]
  2. For each \( \tau \in R_t \), compute
     \[
     \hat{C}(z_{t+1:t}) = \sum_{i=\tau+1}^t l(z_i, (t - \tau)^{-1} S_{\tau+1:1:t}^{(K)}) .
     \]
  3. Calculate
     \[
     F(t) = \min_{\tau \in R_t} \left\{ F(\tau) + \hat{C}(z_{t+1:t}) + \beta \right\} ,
     \]
     \[
     \tau^* = \arg\min_{\tau \in R_t} \left\{ F(\tau) + \hat{C}(z_{t+1:t}) + \beta \right\} .
     \]
  4. Let \( C(t) = \{C(\tau^*), \tau^*\} \).
  5. Set
     \[
     R_{t+1} = \{ \tau \in R_t \cup \{1\} : F(\tau) + C(z_{t+1:t}) \leq F(t) \} .
     \]
- Output \( C(T) \).