A scalable quasi-Newton estimation algorithm for dynamic generalised linear models

Guangbao Guo a, Guoqi Qian b and Lixing Zhu c

a Department of Statistics, Shandong University of Technology, Zibo, People’s Republic of China; b School of Mathematics and Statistics, University of Melbourne, Melbourne, Australia; c Center for Statistics and Data Science, Beijing Normal University, Zhuhai, People’s Republic of China

ABSTRACT
This research develops a scalable computing method based on quasi-Newton algorithm for several estimation problems in dynamic generalised linear models (DGLMs). The new method is developed by applying the principle of maximising a pointwise penalised quasi-likelihood (PPQ) for a DGLM for observed data often of massive size. Statistical and computational challenges involved in this development have been effectively tackled by exploiting the specific block structure and sparsity involved in the underlying projection matrix. The obtained maximum PPQ estimator of the state vector in the DGLM has been shown to be consistent and asymptotically normal under regularity conditions. Numerical studies and real data applications are conducted to assess the performance of the developed method.

ARTICLE HISTORY
Received 28 January 2021
Accepted 16 January 2022

KEYWORDS
Dynamic generalised linear models; penalised quasi-likelihood; scalable statistical computing; quasi-Newton method; high dimension

2020 MATHEMATICS SUBJECT CLASSIFICATIONS
62G99; 62-08

1. Introduction
In this paper, we focus on dynamic generalised linear models (DGLMs). Early works on DGLMs include West, Harrison, and Migon (1985), Gamerman (1998), Fahrmeir (1992), Lindsey and Lambert (1995), among others. In recent years, DGLMs have found applications for relevant data analysis and statistical inference, owing to their flexible framework; see e.g. Chiogna and Gaetan (2002), Triantafyllopoulos (2009), and Das and Dey (2013).

Developing a scalable statistical computing (SSC) procedure is currently in demand for applying a DGLM for data analysis and statistical inference, where the underlying observed data may be massive with high dimension. An effective approach for this development lies on deriving an SSC algorithm capable of being run in a parallel computing environment, and on deriving a statistical inference procedure capable of efficiently collecting information from the multiple parallel computing outcomes simultaneously. Before embarking on a detailed development we give a brief review of statistical computing for the DGLMs here. Two computational methodologies are commonly used for estimation in the DGLMs. The first one is based on Bayesian approach, which often entails using various Markov chain
Monte Carlo (MCMC) algorithms for computing the parameter and state vector estimates. Since MCMC essentially involves sequential computing, it is difficult to implement in a parallel computing environment. The second methodology is based on the maximum quasi-likelihood approach, where Kalman filter, extended Kalman filter, particle filter and Kalman smoother (KS), among others, are often used in the context of the DGLM estimation. In this paper we propose a quasi-Newton estimation algorithm for the DGLMs based on the maximum quasi-likelihood methodology. The proposed algorithm can be computationally scalable once the high-dimensional Hessian-alike matrix in the algorithm is properly transformed to be block-diagonal.

Suppose an \( n \)-dimensional response vector sequence \( \{y_t\}_{t=1}^T \) with \( y_t \in \mathbb{R}^{n \times 1} \), and a state sequence \( \{\alpha_t\}_{t=1}^T \) with \( \alpha_t \in \mathbb{R}^{p \times 1} \) are related through the model

\[
\begin{align*}
  y_t &\sim f(y_t | \alpha_t) = \exp\{\theta_t^\top y_t - b(\theta_t) + c(y_t)\}, \quad \text{(observation equation)} \\
  \alpha_t &= F_t \alpha_{t-1} + \xi_t; \quad t = 1, \ldots, T, \quad \text{(transition equation)}
\end{align*}
\]

where natural parameter \( \theta_t = X_t^\top \alpha_t \in \mathbb{R}^{n \times 1} \) is a vector function with \( p \times n \) design matrix \( X_t \), and functions \( c(\cdot) \) and \( b(\cdot) \) have known forms. \( F_t(\in \mathbb{R}^{p \times p}) \) is a known design matrix, and the random error \( \xi_t \sim N(0, \Sigma_t) \) with \( N(\cdot, \cdot) \) denoting a \( p \)-variate normal distribution, the initial state is \( \xi_0 \sim N(0, \Sigma_0) \) and both \( \Sigma_t \) and \( \Sigma_0 \) are \( p \times p \) variance matrices.

Using the properties of exponential families, the mean and variance functions are:

\[
\begin{align*}
  E(y_t | \alpha_t) &= \mu(\alpha_t) = \partial b(\theta_t) / \partial \theta_t, \quad \text{denoted as } \mu(\alpha_t) = h(X_t^\top \alpha_t), \\
  \text{var}(y_t | \alpha_t) &= \partial^2 b(\theta_t) / \partial \theta_t \partial \theta_t^\top,
\end{align*}
\]

where \( h(\cdot) \) is a two-times continuously differentiable function. Note that we assume the design matrix \( X_t \) involves \( y_s \) with \( s < t \).

Define a DGLM as being specified by (1) with the following conditions:

(C1) Conditional on \( \alpha_t \), the current response \( y_t \) is independent of \( \alpha_{t-1}, \ldots, \alpha_0 \). Namely,

\[
  f(y_t | \alpha_t, \alpha_{t-1}, \ldots, \alpha_0) = f(y_t | \alpha_t); \quad t = 1, \ldots, T.
\]

(C2) The sequence \( \{\alpha_t\}_{t=1}^T \) is Markovian, namely, \( f(\alpha_t | \alpha_{t-1}, \ldots, \alpha_0) = f(\alpha_t | \alpha_{t-1}) \).

(C1) is implied when \( \{\xi_t\}_{t=1}^T \) in the error sequence are mutually independent. The conditional density functions in (C1) are understood to be conditional on \( y_1 \) with \( s < t \) as well if the design matrix \( X_t \) contains past responses. The conditional distribution of \( \alpha_t \) when \( \alpha_{t-1} \) is given, is \( \alpha_t | \alpha_{t-1} \sim N(F_t \alpha_{t-1}, \Sigma_t) \) for given matrix \( \Sigma_t \).

Computation for estimation in the above-defined DGLM is challenging, particularly when some or all of \( (n, p, T) \) are large. This will be tackled as following in this paper. First, we develop a pointwise penalised quasi-likelihood (PPQ) function to replace the classic penalised quasi-likelihood (PQ) function for point and interval estimation of the states and parameters. Second, we explore ways to optimise the PPQ in order for effective and efficient parameter estimation and the associated error analysis. Third, through exploiting a block sparse projection matrix (BSPM), we further develop a scalable quasi-Newton (SQN) algorithm to compute the relevant estimates.
We will also investigate relevant convergence properties of the estimators computed from the SQN algorithm. These properties make it possible to assess the improvement of performance of the new method over that of the existing methods. We will further demonstrate how the SQN algorithm can be implemented in a parallel computing environment through simulation studies. Finally, we apply our proposed method to analyse a real data set on the CSI Internet financial index for illustration.

This paper is organised as follows. Section 2 describes the PPQ estimation procedure for the DGLMs. Section 3 presents the SQN method for computing the PPQ estimators, as well as the convergence properties of the algorithm. Section 4 discusses the statistical errors and asymptotic properties of the SQN method. Section 5 introduces dimension reduction techniques used in high-dimensional DGLMs. Section 6 displays several simulation results. Section 7 discusses the findings and conclusions.

2. Pointwise penalised quasi-likelihood

In this section, we present the PPQ function of DGLMs. Define $y^*_t = (y^*_1, \ldots, y^*_T)^T, \alpha^*_t = (\alpha_1^T, \ldots, \alpha_T^T)^T$. Write $\alpha = (\alpha_0^T, \alpha^*_T)^T$ that is then of $p(T + 1)$-dimensional. Further define $\hat{\alpha}$ as

$$\hat{\alpha} = (a_0^T, \ldots, a_T^T)^T = \arg\max_{\alpha} \{f(\alpha | y^*_T)\},$$

where $a_0$ is the first component of $\hat{\alpha}$ assumed to be known beforehand. The distribution of $\alpha$, given by $y^*_T$, is

$$f(\alpha | y^*_T) = \frac{1}{f(y^*_T)} \left\{ \prod_{t=1}^T f(y_t | \alpha_t, y^*_t) \right\} \prod_{t=1}^T f(\alpha_t | \alpha^*_t, y^*_t-1) \cdot f(\alpha_0).$$

See Fahrmeir and Wagenpfeil (1997, 9) for detail. For any $p(T + 1)$ dimensional vector $\alpha$, by taking logarithm of $f(\alpha | y^*_T)$, we obtain PPQ as an objective function based on (1):

$$\text{PPQ}(\alpha) = \sum_{t=1}^T \left[ \theta_t^T y_t - b(\theta_t) + c(y_t) \right] - \frac{1}{2} (\alpha_0 - a_0)^T \Sigma_0^{-1} (\alpha_0 - a_0)$$

$$- \frac{1}{2} \sum_{t=1}^T (\alpha_t - F_t \alpha_{t-1})^T \Sigma_t^{-1} (\alpha_t - F_t \alpha_{t-1}).$$

For the sufficiently well-behaved model (1), it is easy to see the following holds

$$\hat{\alpha} = \arg\max_{\alpha} \left\{ \text{PPQ}(\alpha) \right\}.$$
We will develop a scalable quasi-Newton algorithm, to implement the calculation for (4). To facilitate this development, we rewrite $PPQ(\alpha)$ as

$$PPQ(\alpha) = I(\alpha) - \frac{1}{2} \alpha^\top K \alpha$$

where

$$I(\alpha) = \sum_{t=0}^{T} I_t(\alpha_t), \quad I_t(\alpha_t) = \theta^\top y_t - b(\theta_t) + c(y_t) \quad (t = 1, 2, \ldots, T),$$

$$I_0(\alpha_0) = -(\alpha_0 - a_0)^\top \Sigma_0^{-1}(\alpha_0 - a_0)/2, \quad \text{and}$$

$$K = \begin{pmatrix}
K_{00} & K_{01} & 0 \\
K_{10} & K_{11} & K_{12} \\
0 & 0 & K_{T-1,T}
\end{pmatrix}$$

being a $p(T+1) \times p(T+1)$ matrix with

$$K_{t-1,t} = K_{t,t-1}^\top = -F_t^\top \Sigma_t^{-1}, \quad K_{tt} = \Sigma_t^{-1} + F_{t+1}^\top \Sigma_{t+1}^{-1} F_{t+1}; \quad t = 1, \ldots, T.$$

$$K_{00} = F_1^\top \Sigma_1^{-1} F_1, \quad F_{T+1} = 0.$$

Denote $y^\top = (a_0^\top, y_1^\top, \ldots, y_T^\top)$, $\mu(\alpha)^\top = \{a_0^\top, h^\top(X_1^\top \alpha_1), \ldots, h^\top(X_T^\top \alpha_T)\}$, and the $p+nT$ by $p+nT$ block-diagonal covariance matrix

$$V(\alpha) = \text{block diag}(\Sigma_0, V_1, \ldots, V_T)$$

with $V_t = \text{Var}(y_t | \alpha_t)$. We also pool all design matrices into a $p(T+1)$ by $p+nT$ block-diagonal matrix $X$ is presented as $X = \text{block diag}\{I_{p \times p}, X_1, \ldots, X_T\}$ where $I_{p \times p}$ is a $p \times p$ identity matrix; and denote another $p+nT$ by $p+nT$ block-diagonal matrix $D(\alpha)$ as

$$D(\alpha) = \text{block diag}\{I_{p \times p}, \partial h(\theta)/\partial \theta_1, \ldots, \partial h(\theta)/\partial \theta_T\}.$$

The quasi-score function equals

$$S_Q(\alpha) = \{S_0^\top(\alpha_0), S_1^\top(\alpha_1), \ldots, S_T^\top(\alpha_T)\}^\top = XD(\alpha) V^{-1}(\alpha) [y - \mu(\alpha)],$$

which is the derivative of the first term in (5) for $PPQ(\alpha)$ with respect to $\alpha$. Accordingly, $S_0(\alpha_0) = \Sigma_0^{-1}(a_0 - \alpha_0), S_t(\alpha_t) = X_tD_t(\alpha_t) V_t^{-1} [y_t - \mu(\alpha_t)] \quad (t = 1, \ldots, T)$, with $D_t(\alpha_t)$
being the \(r\)th diagonal block of \(D(\alpha)\). Denote the following \(p + nT\) by \(p + nT\) block-diagonal weight matrix

\[
W(\alpha) = \text{block diag} \{W_0, W_1(\alpha_1), \ldots, W_T(\alpha_T)\} = D(\alpha) V^{-1}(\alpha) D^\top(\alpha).
\]

We have \(W_0 = \Sigma_0^{-1}, W_t(\alpha_t) = D_t(\alpha_t) V_t^{-1} D_t^\top(\alpha_t)\). The quasi-Fisher information matrix is then provided as

\[
J_Q(\alpha) = \text{block diag}\{ J_0(\alpha_0), J_1(\alpha_1), \ldots, J_T(\alpha_T) \} = X W(\alpha) X^\top
\]

(7)

where \(J_0(\alpha_0) = \Sigma_0^{-1}, J_t(\alpha_t) = X_t W_t(\alpha_t) X_t^\top \ (t = 1, \ldots, T)\). Thus, the quasi-score function and quasi-information matrix of PPQ(\(\alpha\)) are, respectively,

\[
S_{\text{PPQ}}(\alpha) = \partial \text{PPQ}(\alpha) / \partial \alpha = S_Q(\alpha) - K\alpha,
\]

\[
F_{\text{PPQ}}(\alpha) = -E\{ \partial^2 \text{PPQ}(\alpha) / \partial \alpha \partial \alpha^\top \} + K = J_Q(\alpha) + K
\]

(8)

Note that the form of \(\alpha_t\) is symmetric in the second term and the third term of (3). We then have \(E(K\alpha) = 0\) and \(E(S_{\text{PPQ}}(\alpha)) = 0\). The quasi-likelihood equation of PPQ(\(\alpha\)) is

\[
S_{\text{PPQ}}(\hat{\alpha}) = 0.
\]

(9)

The maximum PPQ estimate of \(\alpha\) is then the solution of the above equation which requires a numerical method. Regarding the numerical approach for estimation in DGLMs, Fahrmeir and Wagenpfeil (1997) proposed some numerical smoothing methods illustrated with some real applications; Triantafyllopoulos (2009) reviewed developed numerical strategies; Das and Dey (2013) presented some applications for the DGLMs with numerical approximation; Migon, Schmidt, Ravines, and Pereira (2013) introduced a combination sampling algorithm to deal with the problem.

### 3. A scalable quasi-Newton method for pointwise PQ estimation

As a closed-form solution of \(\hat{\alpha}\) from (9) does not exist, iteration algorithms are often used to obtain approximations. Commonly used iterative methods for solving nonlinear systems are Newton–Raphson (NR) and Gauss–Newton (GN). Given an initial vector \(\hat{\alpha}^{(0)}\), these methods produce the following sequence:

\[
\hat{\alpha}^{(k+1)} = \hat{\alpha}^{(k)} + \delta^{(k)}; \quad k = 0, 1, \ldots
\]

(10)

where \(\delta^{(k)}\) is the solution of \(\delta \in \{-\partial^2 \text{PPQ}(\alpha) / \partial \alpha \partial \alpha^\top\} \delta = S_{\text{PPQ}}(\alpha)\) or \(F_{\text{PPQ}}(\alpha) \delta = S_{\text{PPQ}}(\alpha)\) in \(\alpha = \hat{\alpha}^{(k)}\), depending on whether NR or GN is the underlying method, respectively. Without affecting the convergence of (10), it is possible to replace \(F_{\text{PPQ}}(\hat{\alpha}^{(k)})\) there by \(\ell_k F_{\text{PPQ}}(\hat{\alpha}^{(k)})\) with the learning rate matrix \(\ell_k\) being positive definite, which results in a gradient descent method. Computing \(F_{\text{PPQ}}(\hat{\alpha}^{(k)})^{-1}\) in the aforementioned methods involves inverting a \(p(T + 1)\) by \(p(T + 1)\) matrix, which is challenging when \(p\) or/and \(T\) are large.

We now propose a new SQN method to compute the maximum PPQ estimates which does not involve inverting a high-dimensional matrix directly. The proposed method is
obtained by adding the following secant iteration for finding $\hat{\delta}_\alpha^{(k)}$ to (10): for a given iteration matrix $T_S$ which will be constructed through (14),

$$\hat{\delta}_\alpha^{(k+1)} = T_S(\hat{\alpha}^{(k)})\hat{\delta}_\alpha^{(k)} + d_k; \quad k = 0, 1, \ldots$$

Here we define $T_S(\hat{\alpha}) = M(\hat{\alpha})^{-1}N(\hat{\alpha}) = I_{p(T+1) \times p(T+1)} - M(\hat{\alpha})^{-1}F_{PPQ}(\hat{\alpha})$ based on the decomposition $F_{PPQ}(\hat{\alpha}) = M(\hat{\alpha}) - N(\hat{\alpha})$ which will be explained in the discussion leading to (14). Also $d_k$ is a function of $\hat{\alpha}^{(k)}$, and will be defined in (14) below. Combining the two iterative equations, we can iteratively express the solution from the proposed SQN method as ($k \geq 0$):

$$\hat{\alpha}^{(k+1)} = \hat{\alpha}^{(k)} + \left[ T_S(\hat{\alpha}^{(k)})m_{k-1} + T_S(\hat{\alpha}^{(k)})m_{k-2} + \ldots + I_{p(T+1) \times p(T+1)} \right] M(\hat{\alpha})^{-1} S_{PPQ}(\hat{\alpha}^{(k)}).$$

(11)

Here $m_k$ equals $r$ specified in (13). We can also rewrite the SQN-based solution as

$$\hat{\alpha}^{(k+1)} = \hat{\alpha}^{(k)} + A_{m_k}(\hat{\alpha}^{(k)})S_{PPQ}(\hat{\alpha}^{(k)}) = G_{m_k}(\hat{\alpha}^{(k)}); \quad k = 0, 1, \ldots$$

(12)

in which $A_m(\hat{\alpha})$ and $G_m(\hat{\alpha})$ are given by

$$A_m(\hat{\alpha}) = (I_{p(T+1) \times p(T+1)} - T_S(\hat{\alpha})m)F_{PPQ}(\hat{\alpha})^{-1} \quad \text{and}$$

$$G_m(\hat{\alpha}) = \hat{\alpha} + A_m(\hat{\alpha})S_{PPQ}(\hat{\alpha}); \quad m \in \mathbb{N}^+. $$

The number of elements in $\alpha$ increases with $T$ and $p$, the dimension of $\alpha_t$. To fit DGLMs, we would repeatedly solve a system of $p(T+1)$ equations. This requires a scalable procedure when $p$ and/or $T$ is large. Note that in the above iterations, a properly constructed iteration matrix $T_S(\hat{\alpha})$ (in short, $T_S$) plays a pivotal role ensuring the proposed SQN procedure to be scalable. To see this, it is reasonable to assume the existence of some $r$ ($< p(T+1)$) computationally feasible decompositions for $F_{PPQ}$:

$$\pi_i F_{PPQ} \pi_i^\top = \begin{pmatrix} I_i & K_i \\ L_i & J_{-i} \end{pmatrix}; \quad i = 1, \ldots, r,$$

(13)

where $\pi_i$ is a related projection matrix, $J_i$ is an $n_i \times n_i$ principal submatrix of $F_{PPQ}$, $J_{-i}$ is the complementary principal submatrix, and $L_i$ equals the transpose of $K_i$. We then construct matrix $M_i = \pi_i^\top$ block diag $[J_i, J_{-i}]\pi_i$. It can be shown that $E_iM_i^{-1} = R_i^T J_i^{-1} R_i$ with $E_i = R_i^T R_i$. Here, projection matrix $R_i$ is a BSPM $R_{ib}$ introduced in supplementary material S1.

The iteration matrix $T_S$ can be shown to have the following additive and multiplicative procedures, defined, respectively, as:

$$T_S = T_a = I_{p(T+1) \times p(T+1)} - \frac{1}{4r} \sum_{i=1}^r E_iM_i^{-1} F_{PPQ}, \quad d_k = \sum_{i=1}^r E_iM_i^{-1} S_{PPQ};$$

$$T_S = T_\mu = (I_{p(T+1) \times p(T+1)} - P_r) \ldots (I_{p(T+1) \times p(T+1)} - P_1)$$

$$= I_{p(T+1) \times p(T+1)} - BF_{PPQ}, \quad d_k = BS_{PPQ},$$

(14)
where \( P_i = R_i^T J_i^{-1} R_i F_{PPQ} \), and \( B \equiv B_r \) is given by

\[
B_1 = R_1^T J_1^{-1} R_1, B_i = (I_p(T+1) \times p(T+1) - P_i)B_{i-1} + R_i^T J_i^{-1} R_i, \quad i = 2, \ldots, r.
\]

For given \( T_a \) and \( T_\mu \), we have \( M(\cdot)^{-1} = \frac{1}{4r} \sum_{i=1}^r E_i M_i^{-1} \) and \( M(\cdot)^{-1} = B \), respectively.

---

**Algorithm 1** Scalable quasi-Newton method for the PPQ in the DGLMs

**Require:** \( \hat{\alpha}^{(0)} \) starting vector, \( r \) subblock number, \( k \) iteration number, \( F_{PPQ}(\alpha^{(0)}), S_{PPQ}(\alpha^{(0)}) \).

**repeat**

for \( i = 1 \) to \( r \) do → Parallel

\[
M_i = \pi_i^T \text{ block diag } \{I_i, J_i \} \pi_i, i = 1, \ldots, r \text{ in (3.4). } E_i = R_i^T R_i \text{ where } R_i \text{ is a BSPM } R_b \text{ in S2. } T_S = T_a = I_p(T+1) \times p(T+1) - \frac{1}{4r} \sum_{i=1}^r E_i M_i^{-1} F_{PPQ}, \quad a_k = \sum_{i=1}^r E_i M_i^{-1} S_{PPQ} \text{ or } T_S = T_\mu = \prod_{i=1}^r (I_p(T+1) \times p(T+1) - P_i) = I_p(T+1) \times p(T+1) - BF_{PPQ}, \quad a_k = BS_{PPQ},
\]

where \( P_i = R_i^T J_i^{-1} R_i F_{PPQ} \), and \( B \equiv B_r \) is given by

\[
B_1 = R_1^T J_1^{-1} R_1, B_i = (I_p(T+1) \times p(T+1) - P_i)B_{i-1} + R_i^T J_i^{-1} R_i, \quad i = 1, \ldots, r.
\]

for \( k = 0 \) to \( K \) do → End parallel

\[
\hat{\delta}_\alpha^{(k+1)} = T_S(\hat{\alpha}^{(k)}) \hat{\delta}_\alpha^{(k)} + a_k,
\]

\[
\hat{\alpha}^{(k+1)} = \hat{\alpha}^{(k)} + \hat{\delta}_\alpha^{(k)}.
\]

end for

end for

until \( \varepsilon = 10^{-1}, 10^{-2}, 10^{-3}, \) or \( 10^{-4} \) in (6.1).

---

The SQN procedure using the first \( T_S \) in (14) can be seen as an overlap block Jacobi method, see Gander and Hajian (2015). We shall call it the additive SQN method. The SQN method using the second \( T_S \) in (14) can be seen as a generalisation of block Gauss–Seidel method, see Liu and Keyes (2015), which is later called the multiplicative SQN method. Both procedures have been shown to possess higher convergence rate than the other methods in the literature. The pseudo code for implementing these two methods is presented in Algorithm ???. Denoting the computing complexity of a standard QN method as \( \varphi(p(T+1)) \), an increasing function of \( p(T+1) \), the computing complexity of the proposed SQN method is \( \varphi(p(T+1)/r) \) for given function \( \varphi(\cdot) \). Next we present a key condition and several lemmas for establishing the convergence results for the proposed SQN methods.

**A1:** Let \( \| \cdot \|_2 \) be the Euclidean matrix norm. Assume \( F_{PPQ}(\alpha) \) is a symmetric positive definite (SPD) matrix in a radius-\( r_0 \) neighbourhood of the vector \( \hat{\alpha} \) such that \( S_{PPQ}(\hat{\alpha}) = 0 \). Then there exists an \( L(> 0) \) such that

\[
\| F_{PPQ}(\alpha) - F_{PPQ}(\hat{\alpha}) \|_2 \leq L \| \alpha - \hat{\alpha} \|_2, \quad \alpha \in S_0 = \{ \alpha : \| \alpha - \hat{\alpha} \|_2 < r_0 \}. \quad (15)
\]

The following result provides the convergence of \( \hat{\alpha}^{(k)} \) using the \( T_a \) procedure.

**Theorem 3.1:** Assume that (A1) holds, \( m_k \in \mathbb{N} \) and \( m < \infty \). Also let \( S \) be a neighbourhood of \( \hat{\alpha} \) satisfying \( \hat{\alpha}^{(0)} \in S \). Then the sequence of iterates \( \hat{\alpha}^{(k)} \) converges to \( \hat{\alpha} \) and satisfies

\[
\| \hat{\alpha}^{(k+1)} - \hat{\alpha} \|_2 \leq c^{m_k} \| \hat{\alpha}^{(k)} - \hat{\alpha} \|_2.
\]
Theorem 3.2: Under the same conditions assumed in Theorem 3.1, the sequence \(\{\hat{\alpha}^{(k)}\}_{k \geq 0}\) obtained from using the \(T_\mu\) procedure converges to \(\hat{\alpha}\), and satisfies 
\[\|\hat{\alpha}^{(k+1)} - \hat{\alpha}\|_2 \leq c^m_k \|\hat{\alpha}^{(k)} - \hat{\alpha}\|_2.\]

Theorems 3.1 and 3.2 suggest that the iterates \(\hat{\alpha}^{(k)}\) computed from the proposed SQN methods converge to the maximum PPQ estimator \(\hat{\alpha}\) as \(k \to \infty\), provided that \(\hat{\alpha}\) exists.

4. Approximation properties of the SQN algorithm

In this section, we study the statistical approximation properties of the SQN-based estimator iterate \(\hat{\alpha}^{(k)}\), including consistency, asymptotic unbiasedness and asymptotic normality as \(k \to \infty\). Assume that \(\mathbb{E}(\cdot)\) is the expectation, and particularly \(\mathbb{P}_{\hat{\alpha}}(\cdot)\) and \(\mathbb{E}_{\hat{\alpha}}(\cdot)\) are the probability and expectation, evaluated with respect to the probability distribution of \(\hat{\alpha}\) that is a solution of \(\alpha\) such that \(S_{PPQ}(\hat{\alpha}) = 0\), respectively. We need the following relevant definitions before giving the approximation properties:

Definition 4.1 (Kar, Moura, and Ramanan 2012, strong consistency): A sequence of estimators \(\{\hat{\alpha}^{(k)}\}_{k \geq 0}\) is strongly consistent if
\[\mathbb{P}_{\hat{\alpha}} \left( \lim_{k \to \infty} \hat{\alpha}^{(k)} = \hat{\alpha} \right) = 1. \tag{16}\]

Definition 4.2 (Kar et al. 2012, asymptotic unbiasedness): \(\{\hat{\alpha}^{(k)}\}_{k \geq 0}\) is asymptotically unbiased if
\[\lim_{k \to \infty} \mathbb{E}_{\hat{\alpha}} (\hat{\alpha}^{(k)}) = \hat{\alpha}. \tag{17}\]

Definition 4.3 (Kar et al. 2012, asymptotic normality): \(\{\hat{\alpha}^{(k)}\}_{k \geq 0}\) is asymptotically normal, denoted as
\[\hat{\alpha}^{(k)} - \hat{\alpha} \sim AN(0, \Sigma_\alpha),\]
if there exists a positive definite matrix \(\Sigma_\alpha\) such that the distribution of \(\Sigma_\alpha^{-1/2}(\hat{\alpha}^{(k)} - \hat{\alpha})\) converges to standard normal \(N(0, I_{p(T+1) \times p(T+1)})\) as the iteration time \(k\) goes to infinity. Here \(\Sigma_\alpha\) is called the asymptotic variance of the estimator \(\hat{\alpha}\).

Definition 4.4 (Sieve ratio): For quasi-score \(S_Q(\cdot)\) and \(S_{PPQ}(\cdot)\), the sieve quasi-score ratio is defined as
\[S_s(\hat{\alpha}_P, \hat{\alpha}_Q) = \sup_{\hat{\alpha}_P, \hat{\alpha}_Q} \frac{\|S_{PPQ}(\hat{\alpha}_P) - S_{PPQ}(\hat{\alpha})\|_2}{\|S_Q(\hat{\alpha}_Q) - S_Q(\hat{\alpha})\|_2};\]
for \(\hat{\alpha}_P, \hat{\alpha}_Q \in S_6\) in (S3.2) of supplementary material where \(S_6\) refers to the neighbourhood in Theorems 3.1 and 3.2. Here \(S_{PPQ}(\cdot)\) is the quasi-score function of (8), and \(S_Q(\cdot)\) is the quasi-score function in (6), or in other PQ estimators. Here other PQ estimators refer to the PPQ estimators determined through the kernel smoothing (KS) and MCMC methods.
We now investigate the asymptotic behaviours of the SQN-based estimation iterates under the following assumptions.

(A2): Let $F_{PPQ}(\cdot)$ be the information matrix in (8) and $F_Q(\cdot)$ be the information matrices in (7) or in other PQ estimators. $F_{PPQ}(\hat{\alpha}_P)$ and $F_Q(\hat{\alpha}_Q)$ are invertible for $\hat{\alpha}_P, \hat{\alpha}_Q \in S_6$. Here other PQ estimators refer to the PPQ estimators with the KS and MCMC methods.

(A3): There exists a sieve quasi-score ratio $S_s(\cdot, \cdot)$ satisfying, for $\hat{\alpha}_P, \hat{\alpha}_Q \in S_6$ in (S3.2),

$$S_s(\hat{\alpha}_P, \hat{\alpha}_Q) < C_S,$$

where $C_S$ is a sieve constant with $C_S \in (0, 1]$.

(A4): There exist a sequence of eigenfunctions $\phi_{Pi}$ and the corresponding eigenvalues $\lambda_{Pi}$ over $i$ such that,

$$F_Q(\cdot) = \sum_{i=1}^{p(T+1)} \lambda_{Pi} \phi_{Pi} \phi_{Pi}^\top$$

with $F_Q(\cdot)\phi_{Pi} = \lambda_{Pi}\phi_{Pi}$.

For the sum of the eigenvalues $\sum_{i=1}^{p(T+1)} \lambda_{Pi}$, we have the inequality: $\sum_{i=1}^{p(T+1)} \lambda_{Pi} \leq \rho(F_{PPQ}(\cdot))/C_S$ where $\rho(\cdot)$ is the spectral radius function, and $C_S$ is the sieve constant.

The condition (A4) gives the relationship between sieve constant $C_S$ and trace($F_Q(\cdot)$). Note that the relationship between $\|\hat{\alpha}_P - \hat{\alpha}\|_2$ and $\|\hat{\alpha}_Q - \hat{\alpha}\|_2$ is a very important factor of our proposed method. We use the following two Taylor expansions for $S_{PPQ}(\cdot)$ and $S_Q(\cdot)$:

$$S_{PPQ}(\hat{\alpha}_P) = S_{PPQ}(\hat{\alpha}) + F_{PPQ}(\hat{\alpha}_P)(\hat{\alpha}_P - \hat{\alpha}),$$

$$S_Q(\hat{\alpha}_Q) = S_Q(\hat{\alpha}) + F_Q(\hat{\alpha}_Q)(\hat{\alpha}_Q - \hat{\alpha}).$$

In order to control, the two remainder terms $F_{PPQ}(\hat{\alpha}_P)(\hat{\alpha}_P - \hat{\alpha})$ and $F_Q(\hat{\alpha}_Q)(\hat{\alpha}_Q - \hat{\alpha})$, and make the objective function $\|\hat{\alpha}_P - \hat{\alpha}\|_2/\|\hat{\alpha}_Q - \hat{\alpha}\|_2$ less than 1. We start with these two aspects: first, sieve quasi-score ratio $S_s(\cdot, \cdot)$ is introduced in Definition 4, to process the remainder terms in (S3.2), second, eigenvalue decomposition is performed on $F_Q(\cdot)$ to find the relationship between $S_s(\cdot, \cdot)$ and the eigenvalues of $F_Q(\cdot)$ and $F_{PPQ}(\cdot)$. According to this relationship, we can obtain the following theorem.

**Theorem 4.1:** Under (A2), (A3) and (A4), for a given $\hat{\alpha}_P$ and $\hat{\alpha}_Q \in S_6$,

$$\|\hat{\alpha}_P - \hat{\alpha}\|_2/\|\hat{\alpha}_Q - \hat{\alpha}\|_2 < 1$$

Recall that $\hat{\alpha}^{(k)}$ be an estimated value of $\hat{\alpha}$ in the $k$th iteration. We then get the following result:

**Theorem 4.2 (Asymptotic unbiasedness):** Consider the quasi-Newton method based on the procedures in (14). Under the conditions in Theorems 3.1 and 3.2, $\{\hat{\alpha}^{(k)}\}_{k \geq 0}$ is asymptotically unbiased in the sense that

$$\lim_{k \to \infty} \mathbb{E}_{\hat{\alpha}}(\hat{\alpha}^{(k)}) = \hat{\alpha}.$$

By Lemma 1 of supplementary material and Theorem 3.1 for $T_{\hat{\alpha}}$; and Lemma 1 and Theorem 3.2 for $T_{\hat{\mu}}$, we can prove the above theorem. Some more details can be found in Appendix.
Theorem 4.3 (Strong consistency): Consider the quasi-Newton method based on the procedures in (14). Under the conditions in Theorems 3.1 and 3.2, \( \hat{\alpha}(k) \) is consistent in the sense that

\[
P_{\hat{\alpha}} \left( \lim_{k \to \infty} \hat{\alpha}(k) = \hat{\alpha} \right) = 1.
\]

Using Lemma 1 of supplementary material and Theorems 3.1 and 4.2 for \( T_a \); Lemma 1 of supplementary material and Theorems 3.2 and 4.2 for \( T_\mu \), the above theorem can be proved.

Theorem 4.4 (Asymptotic normality): Suppose that the sequence \( \{\hat{\alpha}(k)\}_{k \geq 0} \) satisfies

\[
\hat{\alpha}(k+1) = \hat{\alpha}(k) + A_{mk}(\hat{\alpha}(k))S_{PPQ}(\hat{\alpha}(k)).
\]

For the given matrix \( \Gamma = F_{PPQ}(\hat{\alpha}) \), the following decomposition holds

\[
A_{mk}(\hat{\alpha}(k))S_{PPQ}(\hat{\alpha}(k)) = -\frac{\Gamma}{k}\hat{\alpha}(k) + \frac{\epsilon_{k+1}}{\sqrt{k}},
\]

where \( E_{\hat{\alpha}}\epsilon_{k+1}^2 \to \sigma^2 \leq C_\epsilon \) (here \( C_\epsilon \) is a given constant), \( E_{\hat{\alpha}}\epsilon_{k+1,t} = E_{\hat{\alpha}}(\epsilon_{k+1,t} | \epsilon_{k,t}) = o(1/k) \), and for given \( \epsilon > 0 \),

\[
\lim_{k \to \infty} E_{\hat{\alpha}}\left[ \epsilon_{k+1,t}^2I_{[\epsilon_{k+1,t} \geq \epsilon k]} \right] = 0; \quad t = 0, 1, \ldots, T.
\]

Then we have

\[
\sqrt{k} \cdot \Gamma^{-1/2} \left( \hat{\alpha}(k) - \hat{\alpha} \right) \sim \mathcal{N} \left( 0, \frac{\sigma^2}{2} I_{p(T+1) \times p(T+1)} \right).
\]

The above theorems can be proved through checking the characteristic function of \( \hat{\alpha}(k) \). See Supplementary material.

From these theorems, we can construct an approximate confidence region of \( \hat{\alpha} \) based on \( \hat{\alpha}(k) \) with \( k \) being sufficiently large. Specifically, an approximate \( (1 - \beta) \)-confidence region of \( \hat{\alpha} \) is \( (\hat{\alpha}(k) \pm Z_{\beta/2} \sqrt{\frac{1}{k}}} \Gamma^{-1/2} \sqrt{k}) \) with \( I_{\epsilon} = (1, 1, \ldots, 1) \) for large \( k \) we set in practical use. For each component of \( \hat{\alpha} \), \( \hat{\alpha}_j \) \( (j = 1, 2, \ldots, p(T+1)) \), its approximate 100\((1 - \beta)\)% confidence interval is

\[
\left( \hat{\alpha}_j(k) \pm \frac{Z_{\beta/2}}{2} \sqrt{\text{var}(\hat{\alpha}_j)} \right)
\]

where \( Z_{\beta/2} \) is the level \( \beta/2 \) standard normal critical value, \( \hat{\alpha}_j(k) \) is the \( j \)th element of \( \hat{\alpha}(k) \), \( \text{var}(\hat{\alpha}_j) \) can be estimated by \( j \)th diagonal element of \( F_{PPQ}(\hat{\alpha}(k))^{-1} \).
5. A brief discussion on dimension reduction

Consider high-dimensional DGLMs where \( p \gg n \), i.e., the number of parameters is larger than the sample size \( nT \) in (19). It is important to note that

\[
\theta = \begin{pmatrix} \theta_1 \\ \vdots \\ \theta_T \end{pmatrix}_{nT \times 1} = \text{block diag}(X_1^T, \ldots, X_T^T)_{nT \times pT} \begin{pmatrix} \alpha_{11} \\ \vdots \\ \alpha_{pT} \end{pmatrix}_{pT \times 1}. \tag{19}
\]

In this situation, it entails a sparse and shrinking estimation of \( \alpha \) in fitting the DGLM, which usually require \( O(npT^2) \) operations. A sparse and shrinkage estimate of \( \alpha \) can be obtained by maximising \( \text{PPQ}(\alpha) + \text{negative penalty term} \), which can still be computed using the proposed SQN algorithm with little modification. Namely, the new estimator \( \hat{\alpha} \) is the following maximiser:

\[
\hat{\alpha} = \arg \max_{\alpha} \left\{ \text{PPQ}(\alpha) + \sum_{t=0}^{T} \sum_{j=1}^{p} p_\lambda(|\alpha_{tj}|) \right\},
\]

where \( \text{PPQ}(\cdot) \) is from (5), and \( p_\lambda(\cdot) \) is a penalty function with \( \lambda(< 0) \). Many different \( p_\lambda(\cdot) \) can be used, e.g., the following commonly used ones:

- **Lasso** (Tibshirani 1996) contains a L1-penalty, namely, \( p_\lambda(|\alpha_{tj}|) = \lambda|\alpha_{tj}| \). Adaptive lasso (Alasso) includes \( p_\lambda(|\alpha_{tj}|) = \lambda \omega_{tj}|\alpha_{tj}| \) where \( \omega_{tj} \) is the weighted function. Group lasso (Glasso) is another alternative that can be expressed as Yuan and Lin (2006):

\[
\sum_{t=0}^{T} \sum_{j=1}^{p} p_\lambda(|\alpha_{tj}|) = \lambda \sum_{i=1}^{G} \|\alpha_i\|_{K_i}
\]

where \( \lambda \) is the tuning parameter, \( \alpha = (\alpha_1^T, \alpha_2^T, \ldots, \alpha_{pG}^T)^T \), \( K = \text{diag}\{K_1, K_2, \ldots, K_G\} \) is a \( p(T+1) \times p(T+1) \) SPD matrix. \( G \) is the number of groups, \( K_i \) is \( p_i \times p_i \) sparse SPD matrix and \( \|v\|_K = (v^T K v)^{1/2} \) for \( v \in \mathbb{R}^{p(T+1) \times 1} \). Glasso is more robust with small-sized group structures. Package \texttt{grplasso} in R software can be used to deal with Glasso in R software. Some other methods include MCP (Zhang 2010) that has the derivative

\[
p_\lambda'(\theta) = \frac{(b\lambda - \theta)_+}{b}; \quad \theta \geq 0,
\]

where \( b (> 1) \) is a shape parameter. SCAD (Fan and Li 2001) uses a penalty function with the derivative

\[
p_\lambda'(\theta) = \lambda \left( I(\theta \leq \lambda) + \frac{(a\lambda - \theta)_+}{\lambda(a-1)} I(\theta > \lambda) \right); \quad a > 2, \theta \geq 0;
\]

SICA (Lv and Fan 2009) uses a penalty function has the following derivative:

\[
p_\lambda'(\theta) = \lambda \frac{(a+1)\theta}{a + \theta}; \quad \theta \geq 0, \ a > 0.
\]
6. Simulation studies

The performance of the SQN method is examined via several measurement indicators. They include the mean squared error (MSE) of $\hat{\alpha}$, bias of $\hat{\alpha}$, coverage rate of confidence intervals of $\hat{\alpha}$, Akaike information criterion (AIC) and Bayesian information criterion (BIC). We also consider the time cost and apply empirical dimension reduction technique in cases of high-dimensional data.

6.1. Simulation settings

We use the natural link function, i.e. $\theta_t$ in each DGLM used in our simulation studies. The matrices $X_t$, $F_t$, $\Sigma_0$ and $\Sigma_t$ in DGLM (1) are specified. The vector response $\{y_t\}^T_{t=1}$ in (1) in our simulation studies are generated from six common probability distributions in exponential family for given values of $T$ and $n$. That is, for $t = 1, \ldots, T$ and $i = 1, \ldots, n$,

1. Binomial (in short, Bi) distribution with $\theta_{ti} = \log[\pi_{ti}/(1 - \pi_{ti})]$ and
   
   $$f(y_{ti}, \pi_{ti}) = \binom{n_{ti}}{y_{ti}} \pi_{ti}^{y_{ti}} (1 - \pi_{ti})^{n_{ti} - y_{ti}}; \quad \pi_{ti} = \frac{3}{4};$$

2. Gamma (Ga) distribution with $\theta_{ti} = \log \beta_{ti}$ and
   
   $$f(y_{ti}, \beta_{ti}) = \frac{\beta_{ti}^{y_{ti}}}{\Gamma(y_{ti})} y_{ti}^{y_{ti} - 1} \exp(-\beta_{ti}y_{ti}); \quad \beta_{ti} = 2, \gamma_{ti} = 5;$$

3. Pareto (Pa) distribution with $\theta_{ti} = \log[\lambda_{ti}/(\lambda_{ti} - 1)]$ and $f(y_{ti}, \lambda_{ti}) = \lambda_{ti}^{y_{ti} - \lambda_{ti} - 1}$ with $\lambda_{ti} = 2$;

4. Poisson (Po) distribution with $\theta_{ti} = \log \lambda_{ti}$ and
   
   $$f(y_{ti}, \lambda_{ti}) = \exp(-\lambda_{ti}) \frac{\lambda_{ti}^{y_{ti}}}{y_{ti}!}; \quad \lambda_{ti} = 2;$$

5. Log-normal (Ln) distribution with $\theta_{ti} = \lambda_{ti}$ and
   
   $$f(y_{ti}, \lambda_{ti}) = \frac{1}{\sqrt{2\pi V}} \exp\left(-\frac{(\log y_{ti} - \lambda_{ti})^2}{2V}\right); \quad \lambda_{ti} = 2, V = 1;$$

6. Weibull (Wei) distribution with $\theta_{ti} = \log \lambda_{ti}$ and
   
   $$f(y_{ti}, \lambda_{ti}) = \frac{v_{ti}}{\lambda_{ti} y_{ti}^{v_{ti} - 1}} \exp\left(-\frac{y_{ti}^{v_{ti}}}{\lambda_{ti}}\right); \quad \lambda_{ti} = 2, v_{ti} = 2.$$

From Theorem 4.1, for any given $\varepsilon > 0$, there exists $k$ sufficiently large such that

$$\|\hat{\alpha}^{(k)} - \hat{\alpha}\|_2 < \varepsilon < 1,$$

We set $\varepsilon = 10^{-1}, 10^{-2}, 10^{-3}$ and $10^{-4}$, respectively, in the simulations. In all experiments reported here, for convenience, we set the computing to stop when

$$\|\hat{\alpha}^{(k)} - \hat{\alpha}\|_2 \leq \varepsilon \||\hat{\alpha}^{(0)} - \hat{\alpha}||_2; \quad \hat{\alpha}^{(0)} \in S_5. \quad (20)$$

Here $\hat{\alpha}^{(0)}$ and $\hat{\alpha}^{(k)}$ are the initial and the $k$th iterative update of $\hat{\alpha}$, respectively.
It facilitates the simulation study by properly choosing the tuning parameter $C_S$ such that
\[
\left(\|S_{PPQ}(\hat{\alpha}^{(k)}) - S_{PPQ}(\hat{\alpha})\|_2 / \|S_Q(\hat{\alpha}^{(k)}) - S_Q(\hat{\alpha})\|_2\right) < C_S.
\]
Observe that $S_Q(\hat{\alpha})$ and $S_{PPQ}(\hat{\alpha})$ are zero or close to zero. For simplicity, in both the numerical and Monte Carlo methods used here we set
\[
\|S_{PPQ}(\hat{\alpha}^{(k)})\|_2 < C_S \cdot \|S(\hat{\alpha}^{(k)})\|_2.
\]

We make use of the available software packages (Rmpi, snowfall, MPICH2 and PETSc) in simulation. The computation is implemented on a cluster, which includes three computers each having quad core processors of specifications 2.67 GHz, 8 GB RAM and 1 TB HDD. Eight threads are used to examine the performance of the SQN method, namely, $r = 8$.

We will compare the proposed SQN method with several existing ones, the details of which are briefly described here. The methods of NR and GN in DGLMs are presented in (10), while the method of Levenberg–Marquardt (LM) numerically solves
\[
(F_{PPQ}(\alpha) + \lambda I_{p(T+1) \times p(T+1)}) \delta = S_{PPQ}(\alpha),
\]
where $\lambda$ is a penalty parameter, cf. Guo (2012). The SQN method with procedure $T_a$ becomes the Newton block–Jacobi (NJ) when it has no overlap sample block in Algorithm 1 of the supplementary. This can also be regarded as a Newton-type block coordinate descent method, cf. Frommer and Renaut (1999). The KS method in DGLMs is as following:
\[
\hat{\alpha}^{(k+1)} = \hat{\alpha}^{(k)} + K^{(k)}[y_T^* - \mu(\hat{\alpha}^{(k)}) - \mu'(\hat{\alpha}^{(k)})(\hat{\alpha} - \hat{\alpha}^{(k)})],
\]
(21)
where
\[
K^{(k)} = [\mu'(\hat{\alpha}^{(k)})]T V^{-1}(\hat{\alpha}^{(k)}) + \Sigma_0^{-1}]^{-1}.\mu(\hat{\alpha}^{(k)}) + \Sigma_0^{-1}]^{-1} \mu(\hat{\alpha}^{(k)}) = V^{-1}(\hat{\alpha}^{(k)}).\]
See Migon et al. (2013) for details.

Furthermore, the estimation of $\alpha$ may also be completed by a Bayesian method where MCMC is expected to be used for the involved computing. Here we will not get into details of the Bayesian method but briefly describe two MCMC methods – the Gibbs sampling and the Metropolis–Hastings (MH) algorithms – in the context of estimating $\alpha$ by posterior random samples. The Gibbs sampling method is as follows. Start from $\alpha^{(0)}$. Step $\alpha^{(k)} \rightarrow \alpha^{(k+1)}$ is generating
\[
\alpha_1^{(k+1)} \sim f(\alpha_1^{(k+1)} | \alpha_2^{(k)}, \ldots, \alpha_{p(T+1)}^{(k)}, y_T^*),
\]
\[
\alpha_2^{(k+1)} \sim f(\alpha_2^{(k+1)} | \alpha_1^{(k+1)}, \alpha_3^{(k)}, \ldots, \alpha_{p(T+1)}^{(k)}, y_T^*),
\]
\[
\vdots
\]
\[
\alpha_{p(T+1)}^{(k+1)} \sim f(\alpha_{p(T+1)}^{(k+1)} | \alpha_1^{(k+1)}, \ldots, \alpha_{p(T+1)-1}^{(k+1)}, y_T^*).
\]

See Fahrmeir (1992) for more details. The Metropolis–Hastings (MH) algorithm is described here. Step $\alpha^{(k)} \rightarrow \alpha^{(k+1)}$ is first generating $\alpha_q \sim f(\alpha_q | y_T^*)$, then accepting $\alpha^{(k+1)} = \alpha_q$ with the following acceptance probability (rate), or accepting $\alpha^{(k+1)} = \alpha^{(k)}$. 

Here \( f(\alpha_q | y^*_{T}) \) is the proposal transition density simply chosen as a normal one. The related acceptance rate is

\[
AR(\alpha^{(k)}, \alpha^{(k+1)}) = \min \left\{ 1, \frac{f(\alpha^{(k+1)} | y^*_{T})q(\alpha^{(k)} | \alpha^{(k+1)})}{f(\alpha^{(k)} | y^*_{T})q(\alpha^{(k+1)} | \alpha^{(k)})} \right\}.
\]

See Gamerman (1998) & Geweke and Tanizaki (2001), for details. Finally, the parallel tempering (PT) method with MH implementation sets

\[
f_i(\cdot) = f(\cdot)^{\frac{1}{T_i}}
\]

with the temperature sequence \( \{T_i\}_{i=1}^N \) satisfying \( T_i < T_{i+1} \) (\( N \) is the number of temperatures), see also Brooks, Gelman, Jones, and Meng (2011).

### 6.2. Simulation results

#### 6.2.1. Low-dimensional simulations

With \( \{n = 5000, p = 10, \hat{\alpha}^{(0)} = (1, \ldots, 1)_{p(T+1)}\} \), Table 1 displays the comparison results of the SQN method with four existing methods: NR, GN, Levenberg–Marquardt (LM) and Newton block–Jacobi (NJ) as described above. The penalty parameter of the LM method is provided by cross-validation. With \( \{T = 50, 100, 200\} \), we present MSEs, biases and coverage rates of 95% pointwise confidence intervals. From the results, the basic message is that when \( T \) is small, say, \( T = 50 \), there is no single winner in each case and the SQN does not perform satisfactorily overall. However, the performance changes with larger \( T \). The SQN is clearly the winner with significant improvement for MSE and bias when \( T = 200 \). Also, for all six distributions, the SQN has robust performance. Interestingly, the coverage rates of confidence intervals are similar for all competitors.

With initial \( \hat{\alpha}^{(0)} = (1, \ldots, 1)_{p(T+1)} \), and \( n = 5000 \), Table 2 lists the comparison results of the SQN method with the methods of KS, Metropolis–Hastings (MH), Gibbs sampling and parallel tempering (PT). With \( \{T = 300, 500, 800\} \), the performance of the SQN is also robust across all distributions and when \( T = 300, 500 \), it works well resulting in small MSE. When \( T = 800 \), the PT has smaller MSE. Regarding the bias, the SQN is always a winner to have smaller bias and regarding the coverage rate all methods perform similarly. In terms of MSEs and biases, the Gibbs, the PT and the SQN obtain smaller values than the LM and the NJ. In terms of coverage rate, the LM, the Gibbs and the SQN obtain more stable values than the NJ and the PT. Therefore, the SQN is overall a recommendable method.

Figure 1 shows the pointwise confidence intervals of \( E(y_{ti}) \) for the DGLMs with all six data generating distributions. The pointwise confidence intervals perform consistently well. These confidence intervals are not symmetric.

Figure 2 presents the confidence intervals of \( E(y_{ti}) \) for the DGLMs with Poisson and Weibull where \( T = 200 \). The confidence intervals accurately reflect the variability. Their ranges span from \(-3\) to \(7\) and from \(-1\) to \(3\), respectively.

We record the computing cost for each of our simulations. As an example, Figure 3 shows the time costs of the DGLMs with Poisson with \( (n, p, T, r) = (200, 5, 5000, 8) \). The computational cost of the SQN is lower than those of the others. Moreover, the SQN is shown to be nearly as efficient as the PT.
Table 1. Comparison results for the SQN method for dynamic generalised linear models.

| T   | MSE ($10^{-1}$) | bias ($10^{-1}$) | CR$_{95}$ (%) |
|-----|-----------------|------------------|---------------|
|     | Bi   | Ga   | Pa   | Po   | Ln   | Wei  | Bi   | Ga   | Pa   | Po   | Ln   | Wei  | Bi   | Ga   | Pa   | Po   | Ln   | Wei  |
| 50  | NR   | 4.12 | 3.51 | 4.45 | 5.63 | 4.78 | 3.94 | 2.17 | 1.89 | 2.22 | 2.83 | 2.31 | 1.95 | 94.7 | 95.1 | 95.2 | 95.3 | 95.4 | 94.8 |
|     | GN   | 3.85 | 3.67 | 4.25 | 5.32 | 5.23 | 4.14 | 1.35 | 2.12 | 1.83 | 2.75 | 1.95 | 2.01 | 94.5 | 95.0 | 95.1 | 95.0 | 94.7 | 94.9 |
|     | LM   | 3.65 | 3.52 | 3.97 | 4.82 | 4.92 | 4.23 | 2.03 | 1.95 | 1.87 | 2.65 | 2.12 | 2.07 | 94.8 | 94.9 | 95.1 | 95.0 | 95.2 | 95.2 |
|     | NJ   | 4.12 | 3.65 | 4.65 | 5.64 | 5.33 | 4.24 | 2.03 | 1.88 | 2.27 | 3.02 | 2.72 | 2.11 | 94.5 | 95.3 | 94.7 | 95.3 | 95.1 | 95.1 |
|     | SQN  | 4.15 | 3.56 | 4.32 | 5.39 | 5.22 | 4.13 | 1.98 | 2.11 | 2.17 | 2.74 | 2.61 | 2.07 | 94.4 | 94.8 | 94.9 | 94.8 | 95.2 | 95.1 |
| 100 | NR   | 3.42 | 3.12 | 3.95 | 4.75 | 4.65 | 3.95 | 2.27 | 1.97 | 2.19 | 2.61 | 2.41 | 1.89 | 94.6 | 94.7 | 95.3 | 95.2 | 94.8 | 94.7 |
|     | GN   | 3.86 | 3.59 | 4.27 | 5.42 | 5.41 | 4.23 | 1.45 | 1.79 | 1.95 | 2.72 | 2.63 | 2.03 | 94.7 | 94.8 | 95.1 | 95.2 | 95.0 | 95.3 |
|     | LM   | 3.75 | 3.48 | 3.85 | 4.95 | 4.72 | 3.92 | 1.78 | 1.65 | 1.73 | 1.94 | 2.79 | 2.32 | 94.6 | 94.5 | 95.4 | 95.3 | 95.2 | 95.1 |
|     | NJ   | 3.52 | 3.17 | 3.65 | 4.32 | 4.55 | 3.72 | 1.52 | 1.03 | 1.19 | 1.72 | 1.32 | 1.42 | 95.0 | 94.8 | 94.9 | 94.9 | 95.1 | 95.2 |
|     | SQN  | 3.57 | 3.12 | 3.53 | 3.75 | 4.17 | 3.27 | 1.32 | 1.21 | 1.33 | 1.65 | 1.27 | 1.29 | 94.8 | 94.9 | 94.9 | 95.1 | 95.2 | 95.1 |
| 200 | NR   | 4.55 | 4.21 | 4.87 | 5.65 | 5.95 | 4.32 | 2.17 | 1.85 | 2.13 | 2.22 | 2.37 | 1.98 | 94.9 | 94.7 | 95.3 | 95.0 | 95.0 | 94.8 |
|     | GN   | 4.67 | 4.37 | 4.92 | 5.37 | 5.85 | 4.47 | 2.03 | 1.94 | 1.87 | 2.13 | 1.94 | 2.17 | 94.5 | 94.6 | 94.7 | 95.2 | 95.2 | 95.1 |
|     | LM   | 4.07 | 3.95 | 4.12 | 4.73 | 4.85 | 4.17 | 1.97 | 1.73 | 1.85 | 1.94 | 1.97 | 1.56 | 94.7 | 94.8 | 95.2 | 94.7 | 95.0 | 95.1 |
|     | NJ   | 3.32 | 2.78 | 3.12 | 3.07 | 3.21 | 3.05 | 1.63 | 1.27 | 1.37 | 1.42 | 1.35 | 1.07 | 95.2 | 94.8 | 95.1 | 94.8 | 95.0 | 95.1 |
|     | SQN  | 2.77 | 2.96 | 2.84 | 3.04 | 2.85 | 1.03 | 1.35 | 1.25 | 1.17 | 1.28 | 1.09 | 94.8 | 94.9 | 95.0 | 95.1 | 95.0 | 95.1 |

Note: Here $(\varepsilon, C_5, b, l_b, p, n) = (10^{-3}, 2, 1, [\sqrt{n}], 10, 5000)$. 


| T  | Bi | Ga | Pa | Po | Ln | Wei | Bi | Ga | Pa | Po | Ln | Wei | Bi | Ga | Pa | Po | Ln | Wei |
|----|----|----|----|----|----|-----|----|----|----|----|----|-----|----|----|----|----|----|-----|
| 300| KS | 8.32| 9.43| 7.52| 8.32| 8.84| 8.95| 4.31| 5.21| 3.78| 4.12| 4.19| 5.01| 94.7| 94.6| 94.9| 95.1| 95.2| 95.0|
|    | MH | 7.52| 9.41| 9.07| 8.75| 8.93| 9.01| 4.61| 4.78| 4.27| 4.12| 4.31| 4.75| 94.8| 94.7| 95.0| 95.1| 95.3| 94.6|
|    | Gibbs | 8.83| 9.12| 8.57| 9.12| 9.13| 8.75| 3.21| 4.65| 3.75| 4.63| 4.57| 4.01| 94.7| 94.6| 95.2| 94.7| 94.9| 94.8|
|    | PT | 9.13| 8.93| 8.61| 8.82| 8.89| 8.64| 4.07| 3.51| 3.67| 4.12| 3.59| 3.77| 94.8| 94.9| 94.8| 94.7| 95.1| 95.2|
|    | SQN | 7.83| 8.42| 8.22| 8.41| 8.73| 8.93| 3.46| 3.52| 3.49| 3.75| 3.67| 3.63| 94.8| 95.1| 95.0| 95.1| 94.9| 95.1|

| 500| KS | 7.15| 8.32| 6.75| 7.12| 7.33| 7.56| 3.41| 4.17| 3.74| 3.42| 3.41| 3.39| 94.7| 94.9| 95.0| 95.1| 94.9| 95.2|
|    | MH | 7.36| 7.95| 6.71| 6.94| 6.89| 7.28| 3.31| 4.12| 2.79| 2.86| 2.82| 3.17| 94.9| 95.0| 95.1| 94.9| 95.1| 95.1|
|    | Gibbs | 6.23| 7.54| 6.49| 6.83| 6.54| 6.94| 3.29| 4.23| 3.31| 3.83| 2.92| 3.15| 94.8| 95.2| 95.1| 94.9| 94.9| 94.8|
|    | PT | 6.84| 7.61| 6.23| 6.53| 6.34| 6.62| 2.53| 3.14| 2.39| 2.37| 2.89| 2.81| 94.9| 95.1| 94.8| 94.9| 95.1| 95.0|
|    | SQN | 6.53| 7.42| 6.83| 6.75| 6.29| 6.17| 2.37| 2.94| 2.67| 2.15| 2.67| 2.35| 95.1| 95.0| 94.8| 94.9| 94.9| 95.1|

| 800| KS | 4.19| 4.57| 4.42| 4.17| 4.23| 4.07| 2.13| 2.05| 1.98| 1.83| 2.17| 1.73| 94.8| 94.9| 95.1| 95.2| 95.0| 94.7|
|    | MH | 4.35| 4.33| 4.17| 4.43| 4.19| 4.23| 2.24| 2.17| 2.07| 1.94| 2.04| 1.84| 95.0| 95.1| 94.8| 94.9| 94.8| 95.1|
|    | Gibbs | 4.27| 4.17| 4.35| 4.29| 4.11| 4.09| 2.11| 2.09| 1.87| 1.79| 1.97| 1.69| 94.8| 95.2| 95.1| 94.8| 95.0| 95.1|
|    | PT | 4.08| 4.21| 4.13| 4.09| 4.07| 4.09| 2.09| 1.94| 1.71| 1.84| 1.96| 1.73| 94.8| 95.1| 95.0| 94.9| 95.1| 95.0|
|    | SQN | 4.17| 4.15| 4.21| 4.27| 4.13| 4.11| 2.06| 1.87| 1.69| 1.74| 1.83| 1.72| 94.9| 95.1| 94.9| 94.9| 95.1| 95.0|

Note: Here \((\epsilon, C_5, b, l_b, p, n) = (10^{-4}, 1, 1, [2\sqrt{nT}], 200, 5000)\).
Figure 1. 95% pointwise confidence regions of $E(y_{it})$ for the DGLMs with Binomial (top left), Gamma (top middle), Pareto (top right), Poisson (bottom right), Log-normal (bottom middle), and Weibull (bottom left). Here $(C_S, b, n, T, \varepsilon) = (0.5, 1, 30, 50, [\sqrt{nT}], 50, 10^{-2})$.

Figure 2. 95% pointwise confidence regions of $E(y_{it})$ for the DGLMs with Poisson (left) and Weibull (right) ($T = 200$). Here $(C_S, b, n, T, \varepsilon) = (2, 1, \sqrt{nT}, 100, 150, 10^{-2})$.

We also study the effects on simulation performance of the sieve constant $C_S$ and the algorithm computing tolerance error $\varepsilon$ with $(s, l_b) = (1, 12)$ in the relevant projection matrix and $(p, n, T) = (150, 200, 2000)$ in the models. Figure 4 displays the performances of the SQN with different $C_S$ and $\varepsilon$ for the Pareto-DGLM as an example. Across different
Figure 3. Execution time comparisons for solving Poisson-DGLMs with different settings. Here \((CS, b, lb, p, n, T) = (2, 1, 27, 5, 200, 5000)\).

Figure 4. Performance comparisons of the SQN method solving Pareto-DGLMs for different sieve constants \(CS\). Here \((b, lb, p, n, T) = (1, 12, 150, 200, 2000)\).

\(\varepsilon\), the indicators (MSE, bias, AIC and BIC) achieve the smallest at \(CS = 1\). Across different \(CS\), the four indicators achieve the largest with \(\varepsilon = 10^{-1}\), and achieve the smallest with \(\varepsilon = 10^{-3}\). In summary, the SQN returns a desirable outcome throughout this designed simulation study.

6.2.2. High-dimensional simulations

As discussed in Section 5, the SQN method needs to be coupled with a dimension reduction criterion in cases of high-dimensional data analysis. The dimension reduction criteria we consider here include Lasso, Alasso, Glasso, MCP, SCAD and SICA. We choose the penalised parameter \(\lambda\) through an eight-fold cross-validation, and evaluate
Table 3. Comparison results for high-dimensional DGLMs with different distributions ($T = 1000$).

| $p$   | MSE ($10^{-1}$) | bias ($10^{-1}$) | CR95 (%) |
|-------|-----------------|------------------|----------|
|       | Bi   | Ga   | Pa   | Po   | Ln   | Wei  | Bi   | Ga   | Pa   | Po   | Ln   | Wei  |
| 1000  | Lasso | 8.32 | 7.59 | 8.53 | 9.44 | 7.94 | 8.79 | 4.29 | 3.74 | 4.53 | 5.04 | 3.94 | 4.06 | 94.7 | 94.8 | 94.9 | 95.0 | 95.1 | 95.1 |
|       | ALasso| 6.43 | 6.94 | 6.79 | 7.32 | 6.21 | 6.74 | 2.93 | 3.12 | 3.04 | 3.52 | 2.79 | 3.02 | 95.1 | 94.9 | 94.8 | 95.0 | 94.8 | 94.9 |
|       | MCP   | 6.34 | 6.87 | 6.84 | 6.94 | 6.57 | 6.82 | 3.45 | 3.42 | 3.51 | 3.67 | 3.28 | 3.52 | 95.2 | 94.8 | 94.9 | 94.9 | 95.1 | 95.0 |
|       | SCAD  | 7.23 | 7.07 | 7.56 | 8.21 | 7.83 | 8.62 | 3.21 | 3.41 | 3.62 | 3.71 | 3.65 | 3.72 | 95.1 | 95.0 | 95.1 | 94.8 | 94.9 | 94.9 |
|       | SICA  | 6.74 | 6.91 | 6.89 | 7.45 | 6.74 | 7.34 | 3.34 | 3.57 | 3.42 | 3.53 | 3.17 | 3.42 | 95.2 | 95.1 | 95.1 | 95.0 | 94.9 | 94.8 |
|       | GLasso| 6.83 | 6.89 | 7.32 | 7.21 | 6.64 | 7.24 | 3.43 | 3.47 | 3.52 | 3.42 | 3.25 | 3.35 | 95.1 | 95.1 | 95.0 | 95.0 | 94.8 | 94.9 |
| 1200  | Lasso | 8.54 | 7.64 | 7.93 | 8.59 | 8.13 | 8.05 | 4.12 | 3.24 | 3.82 | 3.95 | 3.87 | 3.76 | 95.1 | 94.8 | 94.9 | 95.0 | 94.9 | 95.1 |
|       | ALasso| 7.32 | 7.69 | 8.21 | 8.43 | 8.09 | 8.22 | 3.44 | 3.35 | 3.92 | 3.98 | 3.87 | 3.91 | 95.3 | 94.9 | 95.0 | 95.1 | 94.9 | 95.1 |
|       | MCP   | 7.14 | 7.36 | 7.95 | 8.46 | 8.05 | 8.17 | 3.13 | 3.26 | 3.56 | 4.13 | 3.57 | 3.65 | 95.2 | 95.1 | 95.0 | 94.8 | 94.9 | 95.1 |
|       | SCAD  | 7.56 | 7.48 | 7.88 | 8.56 | 8.11 | 8.13 | 3.47 | 3.29 | 3.54 | 4.07 | 4.01 | 3.85 | 95.1 | 94.8 | 94.9 | 95.0 | 94.9 | 94.9 |
|       | SICA  | 7.34 | 7.52 | 7.85 | 8.35 | 7.89 | 7.74 | 3.27 | 3.35 | 3.47 | 4.09 | 3.95 | 3.74 | 95.1 | 95.1 | 94.8 | 95.0 | 85.0 | 85.0 |
|       | GLasso| 7.09 | 7.48 | 7.83 | 8.39 | 7.94 | 7.86 | 2.85 | 3.17 | 3.74 | 3.65 | 3.34 | 3.21 | 95.2 | 95.1 | 95.0 | 94.9 | 94.9 | 94.9 |
| 1500  | Lasso | 8.47 | 8.12 | 8.27 | 8.83 | 8.29 | 8.17 | 4.05 | 3.94 | 3.96 | 4.23 | 3.98 | 3.81 | 95.1 | 95.0 | 95.0 | 94.9 | 94.8 | 94.9 |
|       | ALasso| 7.45 | 7.84 | 7.89 | 8.49 | 8.07 | 7.64 | 3.87 | 3.95 | 3.88 | 4.01 | 4.31 | 3.20 | 95.0 | 95.1 | 94.9 | 95.2 | 94.7 | 94.8 |
|       | MCP   | 7.65 | 8.23 | 7.95 | 8.64 | 8.12 | 7.83 | 3.72 | 4.11 | 3.46 | 4.27 | 4.13 | 3.94 | 95.2 | 95.0 | 94.9 | 95.1 | 94.8 | 94.9 |
|       | SCAD  | 7.53 | 7.86 | 8.34 | 8.78 | 8.42 | 7.79 | 3.61 | 3.82 | 4.21 | 4.34 | 4.16 | 3.79 | 95.1 | 95.2 | 94.9 | 94.9 | 95.0 | 94.8 |
|       | SICA  | 7.38 | 7.64 | 7.74 | 8.23 | 7.78 | 7.68 | 3.42 | 3.65 | 3.68 | 4.22 | 3.79 | 3.65 | 95.0 | 95.1 | 94.9 | 94.8 | 94.9 | 94.8 |
|       | GLasso| 7.29 | 7.78 | 7.83 | 8.58 | 8.27 | 7.75 | 3.42 | 3.57 | 3.74 | 4.24 | 4.12 | 3.48 | 95.1 | 95.1 | 95.0 | 94.9 | 94.8 | 94.8 |

Note: Here $(\epsilon, C_S, b, l_b, n) = (10^{-2}, 1, 3, 84, 600)$. 
Figure 5. Dimension reduction performance for high-dimensional DGLMs with different distributions. Here \((C_S, b, l_b, \varepsilon) = (1, 2, [2 \sqrt{nT}], 10^{-2})\). Top (left, middle): Panels (a) and (b) show the selective variable numbers for binomial-DGLMs with \(n = 300, 400\). Top (right) and bottom (left): Panels (c) and (d) show the selective variable numbers for gamma-DGLMs. Bottom (middle, right): Panels (e) and (f) show the variable numbers for Pareto-DGLMs.

We discuss the number of selected \(\alpha\) states when the dimension \(p\) is given. Figure 5 displays the results of various methods in high-dimensional DGLMs. Panels (a) and (b) are the plots of the DGLMs with binomial distribution, Panels (c) and (d) are the plots for the DGLMs with gamma distribution, Panels (e) and (f) are the plots for the DGLMs with Pareto distribution.

The DGLMs with Poisson, Pareto and Weibull show similar trends. We generate them with different \(n\). With \((b, l_b) = (3, 46)\) in the relevant projection matrix and \(T = 100\), we employ Glasso as a dimension reduction technique to discuss the performance of the SQN with various \(C_S\) and \(p\). We also use this simulation to check the performances of the SQN with different \(C_S\) and \(\varepsilon\). In the previous theoretical results, we need to restrict the value of \(C_S\) in the range \((0, 1)\). In this simulation, we also try other values larger than 1 to see which value of \(C_S\) performs well. Table 4 summarises the estimation results. We can see that the MSEs and biases of \(\hat{\alpha}\) are basically stable in different dimension \(p\) cases. The interesting
Table 4. Comparison results for high-dimensional dynamic generalised linear models on Glasso.

| CS  | \(p\) | Po | Ga | Wei | Po | Ga | Wei | Po | Ga | Wei | Po | Ga | Wei |
|-----|------|----|----|-----|----|----|-----|----|----|-----|----|----|-----|
| 0.5 | 50   | 9.73 | 9.95 | 9.83 | 2.53 | 2.69 | 2.67 | 8.83 | 8.96 | 8.94 | 9.56 | 9.82 | 9.72 |
|     | 80   | 9.64 | 9.79 | 9.75 | 2.11 | 2.24 | 2.21 | 8.52 | 8.64 | 8.64 | 9.23 | 9.48 | 9.45 |
|     | 100  | 9.56 | 9.82 | 9.77 | 2.21 | 2.35 | 2.33 | 8.92 | 8.98 | 8.96 | 9.67 | 9.87 | 9.79 |
|     | 120  | 9.48 | 9.55 | 9.52 | 2.45 | 2.58 | 2.58 | 8.78 | 8.85 | 8.79 | 9.42 | 9.58 | 9.53 |
|     | 150  | 9.52 | 9.68 | 9.58 | 2.38 | 2.56 | 2.53 | 8.84 | 8.97 | 8.82 | 9.59 | 9.71 | 9.68 |
| 1   | 50   | 9.36 | 9.48 | 9.32 | 2.46 | 2.58 | 2.59 | 7.93 | 8.27 | 8.15 | 8.74 | 8.94 | 8.86 |
|     | 80   | 8.83 | 8.95 | 8.92 | 2.07 | 2.13 | 2.08 | 7.64 | 7.96 | 7.89 | 8.46 | 8.69 | 8.65 |
|     | 100  | 9.12 | 9.22 | 9.18 | 2.28 | 2.37 | 2.34 | 7.89 | 8.19 | 8.11 | 8.69 | 8.93 | 8.85 |
|     | 120  | 9.24 | 9.35 | 9.31 | 2.31 | 2.42 | 2.42 | 7.95 | 8.13 | 8.05 | 8.65 | 8.83 | 8.78 |
|     | 150  | 9.36 | 9.48 | 9.32 | 2.42 | 2.56 | 2.53 | 8.21 | 8.45 | 8.33 | 8.95 | 9.23 | 9.13 |
| 1.5 | 50   | 9.66 | 9.74 | 9.71 | 2.66 | 2.78 | 2.72 | 8.46 | 8.58 | 8.55 | 8.95 | 9.12 | 9.07 |
|     | 80   | 9.42 | 9.53 | 9.52 | 2.25 | 2.33 | 2.32 | 8.15 | 8.27 | 8.17 | 8.75 | 9.03 | 8.89 |
|     | 100  | 9.28 | 9.34 | 9.32 | 2.39 | 2.52 | 2.48 | 8.76 | 8.98 | 8.83 | 9.25 | 9.38 | 9.35 |
|     | 120  | 9.56 | 9.68 | 9.66 | 2.47 | 2.66 | 2.65 | 8.64 | 8.62 | 8.63 | 9.13 | 9.32 | 9.26 |
|     | 150  | 9.52 | 9.67 | 9.64 | 2.46 | 2.71 | 2.68 | 8.76 | 8.85 | 8.79 | 9.36 | 9.48 | 9.44 |

Note: Here \((\varepsilon, T, l_b, n) = (10^{-3}, 100, 3, 46, 1000)\).

observation is about the choice of sieve constant \(C_S\). We can see that, when sieve constant \(C_S = 1\) that is on the boundary of the range in our theoretical result, the MSEs and biases are smaller than those with \(C_S = 0.5\) and 1.5, and for AIC and BIC, the results are similar for various \(p\) and \(C_S\).

Further, we generate samples from the DGLM with Weibull where \(\{p = 50, 60, 70, 80, 90\}\), and choose a suitable block sparse projection matrix (BSPM) with the related \(b\) and \(l_b\). Applying Glasso with \(l_b = 16\) and \(\{b = 1, 2, 3, (n T) / \ln(n T)\}\). Figure 6 displays comparison results of the SQN method with different \(b\) and \(p\). With fixed \(l_b\), the MSEs of the SQN when \(b = 1\) are slightly smaller than the others. The bias with \(b = 2\) is smaller than the others. For fixed \(b\), the MSEs and bias with \(p = 50\) and 60 are smaller than the others. Moreover, AIC and BIC have smaller values with \((b, p) = (2, 60)\). Thus, the performance of the proposed method is better in the case.

7. Discussion

The SQN-based estimation proposed in this paper has good asymptotic properties in regard to the parallel iterative computing procedure. The key element in this success is the replacement of the quasi-Fisher information matrix by a BSPM without which simplifies the computing complexity, facilitates the computing speed, but still maintains the estimation accuracy. However, some problems still remain unresolved, for example, the block length choice in the BSPM, the optimisation of the sparse blocks, and the acceleration problem in ultra high-dimension scenarios. The approximation properties in Section 4 would still hold when some sparsity assumptions are imposed. Since the proofs of these properties would be technically much more complicated but do not involve new statistical ideas, we will dispatch them to a future study.

We may opt to use distributed MCMC and distributed resampling method instead in big data analysis. We note that the distributed MCMC methods include PT, Pop-MCMC and particle MCMC, etc. We also note that the PT method has some obvious drawbacks with
Figure 6. Performance comparison of the SQN method for solving Weibull-DGLMs for different BSPMs. We fix $(C_5, n, T, l_b) = (1, 75, 50, 16)$.

respect to information exchange, while Pop-MCMC and particle MCMC might be good choices. For distributed resampling methods, a subblock method to resample the whole data set would be useful. All these deserve further studies.

**Acknowledgements**

The authors would like to thank the editors and the reviewers for their careful work and thoughtful suggestions that have helped improve this paper substantially.

**Disclosure statement**

No potential conflict of interest was reported by the author(s).

**Funding**

The research described herewith was supported by the NSFC grant (Grant Number NSFC11671042).

**ORCID**

Guangbao Guo  [http://orcid.org/0000-0002-4115-6218](http://orcid.org/0000-0002-4115-6218)
References

Brooks, S., Gelman, A., Jones, G., and Meng, X.L. (eds.) (2011), Handbook of Markov Chain Monte Carlo, London: CRC Press.

Chiogna, M., and Gaetan, C. (2002), ‘Dynamic Generalized Linear Models with Application to Environmental Epidemiology’, Applied Statistics, 51, 453–468.

Das, S., and Dey, D. (2013), ‘On Dynamic Generalized Linear Models with Applications’, Methodology and Computing in Applied Probability, 15, 407–421.

Fahrmeir, L. (1992), ‘Posterior Mode Estimation by Extended Kalman Filtering for Multivariate Dynamic Generalized Linear Models’, Journal of the American Statistical Association, 87, 501–509.

Fahrmeir, L., and Wagenpfeil, S. (1997), ‘Penalized Likelihood Estimation and Iterative Kalman Smoothing for Non-Gaussian Dynamic Regression Models’, Computational Statistics and Data Analysis, 24, 295–320.

Fan, J., and Li, R. (2001), ‘Variable Selection Via Nonconcave Penalized Likelihood and Its Oracle Properties’, Journal of the American Statistical Association, 96(456), 1348–1360.

Frommer, A., and Renaut, R.A. (1999), ‘A Unified Approach to Parallel Space Decomposition Methods’, Journal of Computational and Applied Mathematics, 110(1), 205–223.

Gamerman, D. (1998), ‘Markov Chain Monte Carlo for Dynamic Generalised Linear Models’, Biometrika, 85, 215–227.

Gander, M.J., and Hajian, S. (2015), ‘Analysis of Schwarz Methods for a Hybridizable Discontinuous Galerkin Discretization’, SIAM Journal on Numerical Analysis, 53(1), 573–597.

Geweke, J., and Tanizaki, H. (2001), ‘Bayesian Estimation of State-space Models Using the Metropolis-Hastings Algorithm Within Gibbs Sampling’, Computational Statistics and Data Analysis, 37(2), 151–170.

Guo, G. (2012), ‘Parallel Statistical Computing for Statistical Inference’, Journal of Statistical Theory and Practice, 6(3), 536–565.

Kar, S., Moura, J.M., and Ramanan, K. (2012), ‘Distributed Parameter Estimation in Sensor Networks: Nonlinear Observation Models and Imperfect Communication’, IEEE Transactions on Information Theory, 58(6), 3575–3605.

Lindsey, J., and Lambert, P. (1995), ‘Dynamic Generalized Linear Models and Repeated Measurements’, Journal of Statistical Planning and Inference, 47, 129–139.

Liu, L., and Keyes, D.E. (2015), ‘Field-split Preconditioned Inexact Newton Algorithms’, SIAM Journal on Scientific Computing, 37(3), A1388–A1409.

Lv, J., and Fan, Y. (2009), ‘A Unified Approach to Model Selection and Sparse Recovery Using Regularized Least Squares’, Annals of Statistics, 37, 3498–3528.

Migon, H., Schmidt, A., Ravines, R., and Pereira, J. (2013), ‘An Efficient Sampling Scheme for Dynamic Generalized Models’, Computational Statistics, 28(5), 2267–2293.

Tibshirani, R. (1996), ‘Regression Shrinkage and Selection Via the Lasso’, Journal of Royal Statistical Society B, 58, 267–288.

Triantafyllopoulos, K. (2009), ‘Inference of Dynamic Generalized Linear Models: Online Computation and Appraisal’, International Statistical Review, 77, 430–450.

West, M., Harrison, J., and Migon, H. (1985), ‘Dynamic Generalized Linear Models and Bayesian Forecasting’, Journal of the American Statistical Association, 80, 73–83.

Yuan, M., and Lin, Y. (2006), ‘Model Selection and Estimation in Regression with Grouped Variables’, Journal of the Royal Statistical Society: Series B (Statistical Methodology), 68, 49–67.

Zhang, C. (2010), ‘Nearly Unbiased Variable Selection Under Minimax Concave Penalty’, Annals of Statistics, 38, 894–942.

Zhang, K., Zhang, L., and Yang, M.H. (2012), ‘Real-Time Compressive Tracking’, in European Conference on Computer Vision, Berlin Heidelberg: Springer, pp. 864–877.