Quasi-Newton Acceleration of EM and MM Algorithms via Broyden’s Method

Medha Agarwala\textsuperscript{a} and Jason Xu\textsuperscript{b}

\textsuperscript{a}Department of Statistics, University of Washington, Seattle, WA; \textsuperscript{b}Department of Statistical Science, Duke University, Durham, NC

\textbf{ABSTRACT}

The principle of majorization-minimization (MM) provides a general framework for eliciting effective algorithms to solve optimization problems. However, the resulting methods often suffer from slow convergence, especially in large-scale and high-dimensional data settings. This has motivated several acceleration schemes tailored for MM algorithms, but many existing approaches are either problem-specific, or rely on approximations and heuristics loosely inspired by the optimization literature. We propose a novel quasi-Newton method for accelerating any valid MM algorithm, cast as seeking a fixed point of the MM algorithm map. The method does not require specific information or computation from the objective function or its gradient, and enjoys a limited-memory variant amenable to efficient computation in high-dimensional settings. By rigorously connecting our approach to Broyden’s classical root-finding methods, we establish convergence guarantees and identify conditions for linear and super-linear convergence. These results are validated numerically and compared to peer methods in a thorough empirical study, showing that it achieves state-of-the-art performance across a diverse range of problems. Supplementary materials for this article are available online.

\textbf{1. Introduction}

Iterative procedures are becoming increasingly prevalent for statistical tasks that are cast as optimization of an objective function (Everitt 2012). The canonical setting of minimizing a measure of fit together with a penalty term sits at the heart of statistics, yet challenges still arise from high dimensionality, missing data, constraints, and other aspects of contemporary data. The principle of majorization-minimization (MM) provides a framework for designing effective algorithms well-suited for such problems. Perhaps the most well-known special case is the expectation-maximization (EM) algorithm, a workhorse for maximum likelihood estimation under missing data. Besides EM, instances of MM abound in statistics, ranging from matrix factorization (Lee and Seung 1999) to nonconcave penalized likelihood estimation (Zou and Li 2008). The MM principle is attractive because it admits algorithms that (a) are simple to implement and (b) provide stable performance by obeying monotonicity in the objective (Dempster, Laird, and Rubin 1977; Laird 1978).

Consider $\mathbf{x} \in \mathbb{R}^p$ and the goal of minimizing a “difficult” objective function $f : \mathbb{R}^p \rightarrow \mathbb{R}$, that is, finding $\mathbf{x}^* = \text{argmin}_x f(x)$ which is not available in closed form. An MM algorithm transfers this task onto an iterative scheme, successively minimizing a sequence of \textit{surrogate} functions which dominate the objective function $f$ and are tangent to it at the current iterate $\mathbf{x}_k$. This defines an MM algorithm map $F$, updating $\mathbf{x}_k$ to $\mathbf{x}_{k+1} := F(\mathbf{x}_k)$.

Unfortunately, MM algorithms typically converge at a locally linear rate, which can translate to impractically slow progress in many statistical problems, especially in high dimensions (Wu 1983; Boyles 1983; Meng and Rubin 1994). To address this issue, a body of work designs generic acceleration schemes for numerical optimization methods, including Nesterov’s schemes (Nesterov 1983), SAG (Schmidt, Le Roux, and Bach 2017) and SAGA (Defazio, Bach, and Lacoste-Julien 2014), catalyst acceleration (Lin, Mairal, and Harchaoui 2017), and SDCA (Shalev-Shwartz and Zhang 2014). Special attention has also been given to designing acceleration methods specific to MM algorithms (Jamshidian and Jennrich 1993, 1997; Lange 1995; Zhou, Alexander, and Lange 2011). Broadly, these methods seek additional information to better inform the search direction and/or step lengths of the unadorned algorithm. Improvements may come from high-order differentials of the objective or MM algorithm map and typically incur additional computational cost. As a result, it becomes necessary to balance these tradeoffs.

One approach employs \textit{hybrid} accelerators (Jamshidian and Jennrich 1997) that work directly on the original objective function $f$. These methods typically require evaluating the likelihood $f$ and/or its gradient $df$ at each iteration; an approximation to the Hessian $d^2f$ is usually constructed iteratively using information from the MM map. A hybrid approach in Lange (1995) to accelerate EM algorithms requires evaluating the Hessian of the surrogate at each iterate, which can be computationally prohibitive. A related hybrid accelerator by Jamshidian and Jennrich (1993, 1997) relies on the EM step \textit{approximating} the generalized
gradient of $f$ near the fixed point, and the approximation quality deteriorates as one moves away from the fixed point. Outside of the missing data context, classical tools such as quasi-Newton and conjugate gradient methods can be applied to a similar effect. In contrast to these tools, an advantage of MM algorithms lies in sidestepping unwieldy objectives in favor of operating on simpler surrogates—for instance, EM works well because it bypasses the need to consider the observed data log-likelihood. In view of this, hybrid methods fail to fully capitalize on this key advantage of MM algorithms.

An alternative is to instead consider directly accelerating the MM algorithm map $F$ in a way that is largely agnostic to the optimization objective. These have been classified as pure accelerators and are related to a fixed-point iteration view; see Jamshidian and Jennrich (1997). One class of first-order pure accelerators comprises quasi-Newton (QN) algorithms, which build an approximate Jacobian of the map $F$ to find a fixed point $x^* = F(x^*)$. This goal can be equivalently stated as seeking the root of the MM residual $G(x) := F(x) - x$. The STEM and SQUAREM methods of Varadhan and Roland (2008) approximate this Jacobian $dG$ as a scalar multiple of the identity matrix. The QN method of Zhou, Alexander, and Lange (2011) proposes an elegant approximation derived by assuming proximity to the stationary point. While these pure accelerators tend to preserve the simplicity, convergence properties, and low computational cost of the original algorithm, they rely largely on heuristic approximations of $dG$ (or $dG^{-1}$). While inspired by the theory behind quasi-Newton methods, it can be argued that these methods do not fully and formally take advantage of the prior optimization literature. We will see that further improvement can be gained by modifying general Broyden methods (Broyden, Dennis, and Moré 1973) to leverage extra information in the MM map.

This article seeks to fill this methodological gap by proposing a generic accelerator for any MM algorithm via a quasi-Newton root-finding method. We build off of the wisdom in Zhou, Alexander, and Lange (2011), referring to their method as ZAL in this article, and a few related methods which seek to find the root of MM residuals using QN. Various such methods approximate $dG(x)^{-1}$ as the solution to a constrained optimization problem subject to linear constraints via secant approximations of $G$. Our contribution extrapolates additional information from the MM algorithm map to better inform secant approximations. Instead of only considering the norm of the Jacobian near the fixed point as in ZAL, we optimize a richer objective that connects to classical approaches that minimally perturb the Jacobian across iterations, furnishing a rank-two update formula for $dG^{-1}$. This simple yet effective approach guarantees MM acceleration in a more general setting and allows us to establish theoretical guarantees.

This article is organized as follows: in Section 2, we present general background on MM algorithms and existing acceleration techniques. We then formally derive our MM acceleration algorithm and study its convergence in Section 3. Our standard quasi-Newton recipe demands storing the approximate Jacobian matrices in each iteration, which can be computationally ineffective for high dimensions. To address this issue, we further propose a limited-memory variant of our method amenable to high-dimensional settings. We then assess the performance of our algorithm in Section 4.

## 2. Background: EM, MM, and Acceleration

Majorization-minimization is becoming increasingly popular in solving large-scale and high-dimensional optimization problems in statistics and machine learning (Lange and Wu 2008; Zhou et al. 2015; Xu and Lange 2019). An MM algorithm minimizes the objective function $f$ by successively minimizing a sequence of surrogate functions $g(x | x_k)$ that dominate the objective function $f(x)$ and are tangential to it at the current iterate $x_k$. That is, they require that $g(x_k | x_k) = f(x_k)$ and $g(x | x_k) \geq f(x)$ for all $x$ at each iteration $k$. Decreasing $g(x | x_k)$ typically produces a decrease in $f(x)$. The resulting update $x_{k+1} = \text{argmin}_x g(x | x_k)$ implies the string of inequalities $f(x_{k+1}) \leq g(x_{k+1} | x_k) \leq g(x_k | x_k) = f(x_k)$ validating the descent property. Therefore, the MM principle converts a hard optimization problem into a sequence of manageable subproblems, expressed as $x_{k+1} = F(x_k)$.

From the perspective of the algorithm map $F : x_k \mapsto x_{k+1}$, convergence of an MM algorithm amounts to finding the root of $G(x) := F(x) - x$. This fixed-point view has paved the way for quasi-Newton (QN) acceleration regimes that attempt to well approximate the inverse of the Jacobian of $G$ at $x_k$; see Luenberger and Ye (1984) and Dennis Jr and Schnabel (1996) for a more detailed discussion. Let $dG(x)$ be the differential of $G$ evaluated at $x$: then $dG(x) = (dF(x) - I_p)$ where $I_p$ is the $p \times p$ identity matrix. Denoting some approximation to $dG(x_k)^{-1}$ by $H_k$, QN updates of $x_k$ take the form

$$x_{k+1} = x_k - H_k G(x_k).$$

A given instance of QN is defined by the way it approximates $dG(x_k)^{-1}$. A common thread behind deriving various QN methods begins with the secant condition, which sets $H_k$ as the inverse Jacobian of a linear function joining $(x_k, G(x_k))$ and some other point to be chosen $(y, G(y))$. For instance, Broyden’s classical method takes $y$ to be $x_{k-1}$. That is, the secant constraint mandates that $H_k$ satisfies

$$x_k - y = H_k (G(x_k) - G(y)).$$

For $x \in \mathbb{R}^p$, $H_k$ is a $p \times p$ matrix and the secant constraint fixes $p$ degrees of freedom. The remaining $p^2 - p$ degrees entail that (2) is underdetermined, satisfied by infinitely many $H_k$. To admit a well-defined procedure, one proceeds by specifying an additional criterion; we survey various popular approaches in doing so.

### 2.1. Existing MM Acceleration Schemes

Perhaps the most transparent and well-studied QN acceleration scheme was proposed by Jamshidian and Jennrich (1997). Their method, which we refer to as QN1, directly applies the QN method for root finding of Broyden (1965) to update $H_k$ at any step $k$. Contributions since have noted that this dense matrix update becomes computationally prohibitive in high dimensions typical of contemporary data analysis. The STEM method by
Varadhan and Roland (2008) instead provides a simple and effective approximation of $H_k$ as a scalar multiple of the identity matrix. Assuming $H_k = \alpha_k I_p$, three variants of STEM entail slightly different inverse Jacobian approximations under the choices

$$
\begin{align*}
\alpha_k^{(1)} &= \frac{u_k^T v_k}{v_k^T v_k}, & \alpha_k^{(2)} &= \frac{u_k^T u_k}{u_k^T v_k}, & \alpha_k^{(3)} &= -\frac{\|u_k\|}{\|v_k\|},
\end{align*}
$$

where

$$
u_k = F(x_k) - x_k, \quad \text{and} \quad v_k = G(F(x_k)) - G(x_k)
= F^2(x_k) - 2F(x_k) + x_k.
$$

The scalars $\alpha_k$ in (3) can be understood as various steplengths for each update rule. An extension to these ideas known as SQUAREM was later proposed (Varadhan and Roland 2008), using the idea of a "squared" Cauchy method which may outperform traditional Cauchy methods. While SQUAREM performs well on a wide range of problems and is chiefly regarded for its simplicity, the loss of information due to the identity matrix approximation can remain severe, especially in high-dimensional cases.

More recently, Zhou, Alexander, and Lange (2011) propose an effective acceleration scheme which we will abbreviate as ZAL. It enjoys the same computational complexity as SQUAREM by avoiding a matrix approximation of $dG(x_k)^{-1}$ in (1). It is worth mentioning the secant constraint used in ZAL here, as we will motivate a similar condition to define our effective approximation of $F(x_k) - x_k$. Recall the QN secant condition given in (2). Motivated by Zhou, Alexander, and Lange (2011), using the MM update $F(x_k)$ as our choice of $y$ gives the following MM-modified secant condition

$$
F(x_k) - x_k = H_k[G(F(x_k)) - G(x_k)].
$$

This serves as a point of departure for deriving our proposed method. Before presenting the details, we begin with a toy example that highlights the difference between (5) and Broyden’s standard method.

**Illustrative Example.** We begin by accelerating a classic MM example of minimizing the cosine function $f(x) = \cos(x)$. To derive a surrogate, consider the following quadratic expansion about $y \in \mathbb{R}$: for $z$ lying between $x, y$

$$
\cos(x) = \cos(y) - \sin(y)(x - y) - \frac{1}{2} \cos(z)(x - y)^2
\leq \cos(y) - \sin(y)(x - y) + \frac{1}{2}(x - y)^2 := g(x \mid y);
$$

the inequality follows since $|\cos(z)| \leq 1$ (Lange 2016). It is straightforward to minimize $g$ and obtain the nonlinear formula for the MM update $x_{k+1} = \bar{F}(x_k) = x_k + \sin(x_k)$ where the interest is in finding the root of the MM residual $G(x) = F(x) - x = \sin(x)$.

Figure 1 presents two consecutive steps of the QN method for finding the root of $G$ in one scenario after two initial iterations labeled $A$ and $B$. Each plot shows the updates resulting from both secant approximations. We use capital letters to denote...
a point on the Cartesian plane, and lowercase to denote its corresponding x-coordinate. In the left figure, note a and b lie on opposite sides of the root \( x^* = \pi \), marked by the vertical dashed line.

Now, let \( c^* = F(b) \) be the unaccelerated MM update from \( b \). Rather than moving to \( C^* \), the classical Broyden iteration approximates the search direction using the slope of the line joining A and B (dotted red line). Our proposed update, detailed in the next section, will employ the line joining B and \( c^* \) as the search direction (dashed green line). As a result, Broyden’s method produces \( C_1 \) as the next iterate, and our modified update instead leads to \( C_2 \). Specifically, the updates are given by

\[
c_1 = b - G(b) \frac{b - a}{G(b) - G(a)} \quad \text{and} \quad c_2 = b - G(b) \frac{c^* - b}{G(c^*) - G(b)}.
\]

The difference in quality of the approximations is revealed upon considering the next step as displayed in the right panel. Let \( c^{**} = F(c_2) \), and denote the updates given by Broyden’s method and our method as \( C_3 \) and \( C_4 \), respectively. It is visually evident that the standard Broyden iteration is far from the optimum, while our modified Broyden method with extrapolation converges to \( C_4 \).

Even in a univariate setting, where \( H_k \) can be completely determined from the secant condition, the advantage provided by our approximation is stark when the current state does not render a good search direction. Because the secants drawn in Broyden’s method can depend only linearly on the current and previous state, a bad quasi-Newton step propagates to a bad secant approximation, straying far from the fixed point. Our proposed method avoids this by drawing a secant that incorporates information at the current state together with extrapolation from the next MM iterate. This additional information can act as a correction when the original algorithm produces a poor update such as \( b \) in this example.

We repeat this comparison over 1000 trials starting from randomly generated points between 0 and \( 2\pi \). Table 1 gives the summary of the number of iterations until convergence for (a) Broyden’s method with standard secants, and (b) Broyden’s method with MM-modified secants. This toy example requires some suspension of disbelief as the objective would have been trivial to optimize directly, but serves to illustrate an advantage that tends to become more pronounced in higher dimensions where the added directional information we harness from the MM extrapolation is richer. The same intuition is used to now derive our QN acceleration method using the general form of this secant approximation for \( p \geq 2 \).

**Table 1.** Summary of number of iterations until convergence for both methods.

| Method               | Minimum | I Quantile | II Quantile | III Quantile | Maximum |
|----------------------|---------|------------|-------------|--------------|---------|
| Standard Broyden’s   | 2       | 4          | 5           | 6            | 1389    |
| MM-modified Broyden’s| 1       | 2          | 3           | 3            | 10      |

**Deriving the proposed accelerator.** Recall that we are interested in finding the root of \( G(x) \) numerically via QN. Using the differences \( u_k, v_k \) as introduced in the previous section, the secant condition in (5) can be expressed as \( H_k v_k = u_k \); one may impose several secant approximations \( H_k(v^i_k) = u^i_k \) for \( i \in \{1, \ldots, q\} \) for any \( q < p \) as desired. These can be generated at the current iterate \( x_k \) and previous \((q-1)\) iterates, and may yield better performance at the cost of extra computation. Collecting them into two \( p \times q \) matrices \( U_k = (u_1 \ldots u_q) \) and \( V_k = (v_1 v_2 \ldots v_q) \), the corresponding linear constraint for \( H_k \) in the multiple secant case can be written \( H_k V_k = U_k \).

The \( p \times p \) inverse Jacobian matrix \( H_k \) has \( p^2 \) degrees of freedom, of which \( pq \) degrees of freedom are fixed by the secant approximation. To derive a well-defined update, one must choose how to fix the remaining \( p^2 - pq \) degrees of freedom. We follow classical intuitions that yield a connection to Broyden’s root-finding method. We seek the smallest perturbation to \( H_{k-1} \) when updating to \( H_k \), which can also be viewed as imposing a degree of smoothness in the sequence of iterates. The resulting optimization problem can be formulated as

\[
\text{Minimize: } \| H_k - H_{k-1} \|_F \\
\text{subject to: } H_k V_k = U_k,
\]

where \( \| \cdot \|_F \) denotes the Frobenius norm. We now take partial derivatives of the Lagrangian

\[
\mathcal{L} = \frac{1}{2} \| H_k - H_{k-1} \|_F^2 + \Lambda^T (H_k V_k - U_k)
\]

with respect to \( h_k^j \) and set to 0. Here \( h_k^j \) denotes the \( j \)th element of the matrix \( H_k \). As a consequence, we obtain the Lagrange multiplier equation \( h_k^j - h_{k-1}^j + \sum_{q=1}^p \lambda_i v_{iq} = 0 \), which can be expressed in matrix form as

\[
H_k - H_{k-1} + \Lambda V_k^T = 0.
\]

Right-multiplying (7) by \( V_k \) and imposing the constraint from (5) gives the solution for \( \Lambda = (H_{k-1} V_k - U_k)(V_k^T V_k)^{-1} \). This yields

\[
H_k = H_{k-1} - H_{k-1} v_k v_k^T + u_k v_k^T v_k^T - u_k v_k^T.
\]

At this juncture, it is worth comparing (9) to Broyden’s original update formula

\[
H_k = H_{k-1} - H_{k-1} \Delta G(x_k) \Delta G(x_k)^T + \frac{(x_k - x_{k-1}) \Delta G(x_k)^T}{\Delta G(x_k)^T \Delta G(x_k)}
\]

where \( \Delta G(x_k) = G(x_k) - G(x_{k-1}) \). The novelty of our method originates from the new secant condition that extrapolates information from the original MM update. We see (9) can be written as \( H_k = H_k - A_k + B_k \) where both \( A_k \) and \( B_k \) are rank-1 matrices, revealing a rank-2 update as expected.

The search direction \( p_k \) at iteration \( k \) is given by \( p_k = -H_k G(x_k) \), with a corrected update formula \( x_{k+1} = x_k + \gamma_k p_k \), where \( \gamma_k = \omega_k / \| p_k \| \) is an appropriate scaling factor in the search direction. Here \( \omega_k \) is the steplength and \( \| \cdot \| \)
denotes the $L_2$ vector norm. The corresponding step length for the unaccelerated MM algorithm is $\|F(x_k) - x_k\| = \|u_k\|$, and for a SQUAREM algorithm is $\|a_k\|\|u_k\|$ for $i \in \{1, 2, 3\}$. We choose the step length $|a_k^{(3)}|\|u_k\| = \|u_k\|^2/\|v_k\|$ from (3) for our experiments in this paper due to its intuitive explanation (Varadhan and Roland 2008).

Even after fixing a suitable step length and overlaying the acceleration method on a monotonic algorithm, the accelerated update can violate the descent property, which can potentially lead to failure to converge. Most acceleration schemes operating on monotonic algorithms rely on some kind of fallback mechanism to avoid this issue. Zhou, Alexander, and Lange (2011) revert back to the original MM update whenever the accelerated updates defy the descent property. DAAREM and SQUAREM methods rely on a more flexible partial-monotonicity approach which allows users to set the threshold for the permitted degree of nonmonotonicity. Setting this threshold to zero imposes strict monotonicity, but may slow the algorithm when it reverts back to the original MM updates often. Usually, a tradeoff is reached by setting the threshold to some suitable finite value; these strategies apply analogously to our approach.

**Intuition and relation to existing methods.** We have seen how the proposed approach is grounded in the wisdom behind Broyden's root-finding method, but makes use of an improved secant approximation (5). As illustrated in the demonstrative example, this benefit of the approximation is 2-fold: $F(x_k) - x_k$ gives a more reliable direction to move along using the descent property of the MM map, especially when $x_k$ was a poor update from $x_{k-1}$. Second, instead of only one constraint, the MM map enables us to impose multiple linear constraints that should become increasingly accurate as iterates $x_k$ approach $x^{*}$.

The STEM and SQUAREM methods employ **scalar** multiples of the identity as approximations to the Jacobian matrix, potentially ignoring valuable curvature information compared to a denser approximation such as ours. Compared to traditional root finders for nonlinear functionals (Broyden 1965; Pearson 1969), their convergence properties are not as well-established. The ZAL method is also clearly related to our accelerator but the update formula from (9) for BQN is not reasonable. It is our understanding that perturbing $H_k$ across iterates. It is unclear how convergence is affected when initiated far from a stationary point when this linear approximation is not reasonable. It is our understanding that this approach may fail or converge slowly in such cases, since penalizing $M_k$ discourages large steps even when the current estimate is far from stationarity.

To understand the relationship to DAAREM, we establish some notation: for the $r$th iterate, let $f_i := F(x_i) - x_i$, $\Delta x_i = x_{i+1} - x_i$, $\Delta f_i = f_{i+1} - f_i$. Further, let $X_i$ and $F_i$ denote the $m \times p$ matrices $X_i = [\Delta x_{i-1} \ldots \Delta x_1]$, $F_i = [\Delta f_{i-1} \ldots \Delta f_1]$. A connection between QN and AA becomes evident from examining their respective update formulas:

$$x_{k+1} = x_k - H_{k-1}f_k = (X_k - H_{k-1}F_k)(F_k^T F_k)^{-1} F_k^T f_k;$$

$$x_{k+1} = x_k + f_k - (X_k + F_k)(F_k^T F_k)^{-1} F_k^T f_k.$$  

In particular, AA is equivalent to the QN method in the special case that $H_{k-1} = -I$. When $H_{k-1}$ is very close to $-I$, then the QN update is close to the original update $x_{k+1} = F(x_k)$. From this view, AA can therefore be intuitively understood as a QN method that promotes updates close to the original fixed point method while obeying $m$ secant conditions. This can be desirable, but can also limit potential acceleration especially in settings when more complete curvature information from $H_{k-1}$ can guide a successful update. Replacing these secant conditions with our modified secant condition yields the BQN update.

### 3.1. A Limited Memory Variant for High-Dimensional Settings

A chief advantage of the prior work we survey lies in computational simplicity—they are quite scalable to high-dimensional problems as their space complexity only grows linearly in the number of variables. Though our method brings us closer to established optimization theory, it requires Jacobians that may become computationally unwieldy as the number of variables grows. To ameliorate this, we now describe a low-memory variant. Examining (9) reveals that BQN requires $H_k$ to perform a rank-two update at each step, which can be computationally prohibitive in high dimensions. Additionally, storing this full $p \times p$ matrix at each step can incur high computational cost. Fortunately, limited memory variants of quasi-Newton algorithms have been proposed (Shanno 1978; Nocedal 1980), and rooting our method in Broyden’s framework allows us to import some of these ideas.

We will construct the limited memory version of our algorithm denoted by L-BQN by analogy to the way BFGS algorithm is made scalable using L-BFGS (Liu and Nocedal 1989). BFGS (Fletcher 2013) is a quasi-Newton optimization method that stores an approximation of the inverse Hessian matrix of the objective function at each iteration. In large dimensional problems, L-BFGS surpasses this problem by instead storing only a few vectors that represent the inverse Hessian approximation implicitly. Likewise, we will also store pre-defined $m$ vectors that will approximate the inverse Jacobian at each step. Recall that our update is given by $x_{k+1} = x_k - H_k G(x_k)$, where $H_k$ follows the update formula from (9):

$$H_k = H_{k-1} \left( I_p - \frac{v_k v_k^T}{v_k^T v_k} \right) + \frac{u_k v_k^T}{v_k^T v_k} =: H_{k-1} W_{k} + \frac{u_k v_k^T}{v_k^T v_k}.$$  

Akin to the L-BFGS method, we may store $m$ previous pairs of $[u_i, v_i], i = k - 1, \ldots, k - m$, where $m$ typically is chosen between $3$ and $20$. The matrix product required at each step $H_k G(x_k)$ can be obtained by performing a sequence of inner products and vector summations involving only $G(x_k)$ and the pairs $[u_i, v_i], i = k, \ldots, k - m$. After the new iterate is computed, the oldest pair $[u_i, v_i]$ is dropped and replaced by the pair $[u_{k+i}, v_{k+i}]$ obtained from the current step.
The limited memory variant proceeds by recursion. At each $k$th step, an estimate of the inverse Jacobian is initialized as a scalar multiple of identity matrix $H_k^{(0)} = v_k^I v_k$. The scale factor $v_k$ attempts to capture the size of the true inverse Jacobian matrix along the most recent search direction. Next, $H_k^{(0)}$ is updated $(m + 1)$ times according to (9) in a nested manner to obtain the relation

$$
H_k = H_k^{(0)}(W_k \cdots W_k) + \frac{u_{k-m} v_{k-m}^T}{v_{k-m} v_{k-m}}(W_{k-m+1} \cdots W_k) + \frac{u_{k-m+1} v_{k-m+1}^T}{v_{k-m+1} v_{k-m+1}}(W_{k-m+2} \cdots W_k) + \cdots + \frac{u_k v_k^T}{v_k v_k}
$$

This is equivalent to updating the initial estimate $H_k^{(0)}(m + 1)$ times, imposing the secant condition defined by the pair $\{u_{k-m+i}, v_{k-m+i}\}$ for $i = 0, \ldots, m$, moving minimally from the previous approximation with respect to Frobenius norm each time. Details on obtaining the nested formula above can be found in Chapter 6 of Nocedal and Wright (2006). There the authors suggest that an effective choice for the scaling factor is given by $v_k = u_k^T v_k / u_k^T v_k$. Through this choice, our L-BQN algorithm can be understood as a generalization of the STEM method (Varadhan and Roland 2008). STEM coincides with the special case where $m = 0$. However, the approximate inverse Jacobian $v_k^I v_k$ for STEM is derived by minimizing the distance between the zeros of two linear secant-like approximations for $G(x)$—one centered around $x_k$, and another at $F(x_k)$. While the approach leading to this approximation is quite different, one may accord more confidence in L-BQN as for $m > 0$, the inverse Jacobian approximation can be made more robust by leveraging curvature information from the last $m$ iterates. As discussed by Nocedal (1980), a higher choice of $m$ tends to make the algorithm more robust but can lead to poor estimates of $H_k$ when older secants provide incorrect information about the current gradient. Furthermore, smaller $m$ makes the algorithm faster and is suggested for smooth objective functions. Consequently, the optimal choice of $m$ is problem dependent, and the users are advised to choose $m$ based on prior knowledge about the optimization objective or simply by tuning.

**Remark 1.** Note that L-BQN can be slower than BQN with an informed choice of $q$ if the problem dimension is low. This is unsurprising because the cost of computing the recursive calculations in (10) may outweigh the matrix operations in (9) for small $p$. The advantage of L-BQN begins to become apparent in high-dimensional problems, as we will illustrate through the examples in Section 4.

### 3.2. Convergence

We now analyze the convergence properties of the proposed method. The two essential components are (a) convergence of the base MM algorithm to the stationary point $x^*$, and (b) convergence of Broyden’s root finding quasi-Newton method to the stationary point $x^*$ of the map $G$. Our study bridges careful analyses of these two facets.

Naturally, establishing convergence guarantees for our proposed acceleration scheme rests on the convergence of the underlying MM map, which typically exhibits a locally linear rate of convergence (Lange 2016). We assume the base algorithm to be locally convergent in a neighborhood $S$ of $x^*$ with the rate of convergence denoted by $r > 0$. In this section, we prove that BQN is also locally convergent to $x^*$ in a subset of this neighborhood, and further identify conditions that establish its convergence rate. Recall $\{x_k\}$ converges to $x^*$ at a linear rate if, for some chosen vector norm $\| \cdot \|_k$, $\frac{\|x_k - x^*\|_k}{\|x_k - x^*\|} \leq r$ for some rate of convergence $r \in (0, 1)$. The convergence rate is superlinear if $\frac{\|x_k + 1 - x^*\|_k}{\|x_k - x^*\|} \rightarrow 0$ as $k \rightarrow \infty$.

A seminal work of Broyden, Dennis, and Moré (1973) derives local linear and $Q$-superlinear convergence results for several single and double-rank quasi-Newton methods. Our approach stands close to Broyden’s second method, while the improved secant approximation through MM extrapolation will be incorporated into the analysis. We assume that $G$ is differentiable in a neighborhood of $x^*$, in that the Jacobian matrix $dG(x^*)$ exists and is nonsingular. In many instances, we will treat $(x, dG(x)^{-1})$ as a tuple whose individual components are updated via (1) and (9). It is crucial to prove that the update function in (1) is well defined in some neighborhood of the limit point $(x^*, dG(x^*)^{-1})$. To this end, we first prove by induction the local convergence of our algorithm under certain conditions. We then carefully construct a neighborhood of $(x^*, dG(x^*)^{-1})$ to satisfy these conditions explicitly.

To simplify notation, our current iterate is denoted by $(x, H)$ in a neighborhood of $(x^*, dG(x^*)^{-1})$. We use $\bar{x}$ to denote the update on $x$ given by (1), $\bar{H}$ to denote the update on $H$ from (9), and introduce further notations:

$$
s = \bar{x} - x, \quad y = G(\bar{x}) - G(x), \quad u = F(x) - x, \quad v = G(F(x)) - G(x).
$$

In the subsequent discussion, suppose $\| \cdot \|$ denotes a chosen vector norm on $\mathbb{R}^p$, then for a $p \times p$ matrix $A$, $\| A \|$ denotes the corresponding induced operator norm. The following lemma provides useful inequalities to be applied in proving the main theorem.

**Lemma 1.** Assume $G : \mathbb{R}^p \rightarrow \mathbb{R}^p$ is differentiable in the open convex set $D$, and suppose that for some $\bar{x}$ in $D$ and $d > 0$,

$$
\| dG(x) - dG(\bar{x}) \| \leq K \| x - \bar{x} \|^d,
$$

where $K \in \mathbb{R}$ is a constant. Assuming $dG(\bar{x})$ is invertible, for each $y, z$ in $D$,

$$
\| G(y) - G(z) - dG(\bar{x})(y - z) \|
\leq K \max \{\| y - \bar{x} \|^d, \| z - \bar{x} \|^d \} \| y - z \|,
\| dG(\bar{x})^{-1}(G(y) - G(z)) - (y - z) \|
\leq K \| dG(\bar{x})^{-1} \| \max \{\| y - \bar{x} \|^d, \| z - \bar{x} \|^d \} \| y - z \|.
$$

\[ (12) \]
Moreover, there exists $\epsilon > 0$ and $\rho > 0$ such that $\text{max}(\|y - \hat{x}\|^d, \|z - \hat{x}\|^d) < \epsilon$ implies that $y$ and $z$ belong to $D$, and
\[
(1/\rho)\|y - z\| \leq \|G(y) - G(z)\| \leq \rho\|y - z\|. \tag{13}
\]

Inequalities (12) follow from standard arguments using Taylor’s expansion (Ortega and Rheinboldt 2000), while inequality (13) is an immediate consequence of continuity and nonsingularity of $dG$ at $\hat{x}$. In the subsequent analysis, we will also use the matrix norm $\| \cdot \|_M$ where $\|A\|_M := \|MAM\|_F$ and $M$ is any matrix. Recall that there is a constant $\eta > 0$ such that $\|A\| \leq \eta\|A\|_M$ by the equivalence of norms in finite-dimensional vector spaces.

We now derive general sufficient conditions for local convergence in the spirit of a classic result by Broyden, Dennis, and Moré (1973). Since we require the inverse of $dG$, we posit the following conditions before proving convergence, with $S$ and $D$ as defined earlier.

**Assumption 1.** (A1) Let the function $G : \mathbb{R}^p \to \mathbb{R}^p$ be differentiable in the open convex set $D$ containing $x^*$ such that $G(x^*) = 0$ and $dG(x^*)$ is nonsingular. Assume that for some $d > 0$, $G$ satisfies Inequality (11) inside $D$.

**Assumption 2.** (A2) Let the update function in (1) be well-defined in a neighborhood $N_1$ of $x^*$ where $N_1 \subset D \cap S$, and inverse Jacobian update from (9) be well-defined in a neighborhood $N_2$ of $dG(x^*)^{-1}$ containing nonsingular matrices. Assume that there are nonnegative constants $\alpha_1$ and $\alpha_2$ such that for each tuple $(x, H)$ in $N_1 \times N_2$, the following is satisfied,
\[
\|H - dG(x^*)^{-1}\|_M \leq \left[ 1 + \alpha_1 \max \left\{ \|F(x) - x^*\|^d, \|x - x^*\|^d \right\} \right]\|H - dG(x^*)^{-1}\|_M + \alpha_2 \max \left\{ \|F(x) - x^*\|^d, \|x - x^*\|^d \right\}.
\]
\[
\tag{14}
\]

The first assumption warrants the application of Lemma 1 on $G$, and the second assumption lends a key error bound on the inverse Jacobian estimation. The notion of well-defined used in Assumption 2 will be qualified for BQN later in Theorem 2.

**Theorem 1.** Let A1 hold true for the function $G$ and A2 be satisfied for some neighborhoods $N_1$ and $N_2$ and nonnegative constants $\alpha_1$ and $\alpha_2$. Then for each $r \in (0, 1)$ there exist positive constants $\epsilon(r)$ and $\delta(r)$ such that the sequence with $x_{k+1} = x_k - H_kG(x_k)$ is well defined and converges to $x^*$ whenever $\|x_0 - x^*\| < \epsilon(r)$ and $\|H_0 - dG(x^*)^{-1}\|_M < \delta(r)$. Furthermore,
\[
\|x_{k+1} - x^*\| \leq r\|x_k - x^*\| \quad \text{for each} \quad k \geq 0,
\]
and the sequences $\{\|H_k\|\}$ and $\{\|H^{-1}_k\|\}$ are uniformly bounded.

The detailed proof appears in the Appendix. Under Theorem 1, we inherit the following property by an identical argument of Broyden, Dennis, and Moré (1973), with proof omitted here.

**Corollary 1.** Assume that the conditions of Theorem 1 hold. If some subsequence of $\{\|H_k - dG(x^*)^{-1}\|_M\}$ converges to zero, then $\{x_k\}$ converges $Q$-superlinearly to $x^*$.

It remains to show that our acceleration algorithm satisfies the assumptions of Theorem 1 and Corollary 1. The following result and subsequent corollary identify concrete conditions on the update functions $F$ that ensure this.

**Theorem 2.** Let A1 hold true for the function $G$. If
\[
\frac{\|My - M^{-1}v\|}{\|M^{-1}v\|} \leq \mu_2 \|v\|^p, \quad v \neq 0, \tag{15}
\]
for a constant $\mu_2 \geq 0$, nonsingular and symmetric matrix $M \in \mathbb{R}^{p \times p}$, and all $(x, H)$ in a neighborhood $N'$ of $(x^*, dG(x^*)^{-1})$, then the update functions (9) is well-defined in a neighborhood $N'$ of $(x^*, dG(x^*)^{-1})$ and the corresponding iteration $x_{k+1} = x_k - H_kG(x_k)$ is locally convergent to the limit point $x^*$.

We emphasize that this result does not require stronger conditions than those imposed in the classical results pertaining to Broyden acceleration, which have endured as reasonable assumptions in the optimization literature.

**Corollary 2.** If further $\lim_{k \to \infty} \frac{\|x_{k+1} - F(x_k)\|}{\|x_k - x^*\|} = 0$ holds, then the convergence rate of $\{x_k\}$ to $x^*$ is $Q$-superlinear.

The above condition does not explicitly specify the class of EM algorithms that may benefit from superlinear convergence under BQN. In particular, if the baseline algorithm is superlinear, then BQN piggybacks on this good behavior and retains superlinear convergence. In theory, even if EM does not converge superlinearly, the above condition may still hold, as long as the combined decay of $(I + H_k)$ and $G(x_k)$ to 0 is faster than the decay of $x_k - x^*$. Although this condition is not always easy to verify for a general EM algorithm, it establishes some intuition behind the advantages we observe empirically. The complete technical proofs of these results are detailed in the Appendix.

### 4. Results and Empirical Performance

We now turn to a performance assessment on a variety of real and simulated data examples, including (a) quadratic minimization using Landweber’s method, (b) maximum likelihood estimation in a truncated beta-binomial model, (c) the largest (and smallest) eigenvalue problem for symmetric matrices, (d) location-scale estimation of a multivariate $t$-distribution, and (e) high-dimensional sparse logistic regression, with extended comparisons following suggestions by an anonymous referee in the Appendix. These problems were used in prior studies that introduced the competing methods we benchmark against, offering a conservative comparison. To assess our proposed BQN method and its limited memory variant L-BQN, we compare to unaccelerated MM, the ZAL accelerator, DAAREM, SQUAREM, and the classical Broyden method by Jamshidian and Jennrich (1997) which we call QN1.

All methods are implemented using R; we use the implementation of ZAL and SQUAREM from the turboEM package and DAAREM from the daarem package. Throughout our examples, we present the results for all three versions of SQUAREM. The proposed accelerators BQN and L-BQN and
all data examples are available in an open-source R package implementation quasiNewtonMM. The number of secants \( q \) for BQN and ZAL are chosen according to the problem at hand and specified for each example.

Two main stopping criteria considered in the literature are 1) objective-based rules that declare convergence at \( x_i \) if \( \|f(x_i) - f(x_{i-1})\| \leq \epsilon \), and 2) parameter-based rule stopping when \( x_i \) when \( \|x_i - x_{i-1}\| \leq \epsilon \) for a specified norm \( \|\cdot\| \) and tolerance \( \epsilon \). Following the convention set by other methods, we default to the parameter-based stopping rule, but also examine the objective-based stopping rule for the second example due to flatness of the objective near minima. Following Zhou, Alexander, and Lange (2011), we revert to the original MM step whenever updates are not well-behaved and at least on par with its competitors. An overall theme is that no one accelerator dominates the others across the board. Existing methods may outpace our approach on some examples but then falter on a case-by-case basis. BQN succeeds consistently, but grows slowly as dimension increases; the L-BQN version however, is able to remain competitive in terms of runtime in such cases.

4.1. Landweber’s Method for Quadratic Minimization

We begin with the “well-behaved” problem of minimizing a quadratic function \( f : \mathbb{R}^p \rightarrow \mathbb{R} \) using an MM iterative scheme. For \( \theta \in \mathbb{R}^p \), consider a quadratic objective function

\[
f(\theta) = \frac{1}{2} \theta^T A \theta + b^T \theta,
\]

where \( A \) is a \( p \times p \) positive definite matrix and \( b \in \mathbb{R}^p \). The exact solution is available by solving the linear equation \( A \theta = -b \), but incurs a complexity of \( O(p^3) \). To avoid this computational cost, Landweber’s method instead adopts an iterative scheme, making use of the Lipschitz property of gradient \( \nabla f(\theta) \). The method can be viewed from the lens of majorization-minimization (Lange 2016): since \( \nabla f(\theta) = A \theta + b \), we can write the inequality

\[
\|\nabla f(\theta) - \nabla f(\Phi)\| = \|A(\theta - \Phi)\| \leq \|A\|\|\theta - \Phi\|.
\]

As a consequence, the spectral norm of \( A \) is the Lipschitz constant for \( \nabla f(\theta) \). Let the constant \( L > \|A\| \). Landweber’s method gives the following majorization for \( f(\theta) \):

\[
f(\theta) \leq f(\Phi) + \nabla f(\Phi)^T (\theta - \Phi) + \frac{L}{2} \|\theta - \Phi\|^2.
\]

Minimizing the above surrogate function then yields the MM update formula

\[
\theta_{n+1} = \theta_n - \frac{1}{L} \nabla f(\theta_n) = \theta_n - \frac{1}{L} (A \theta_n + b).
\]

We consider a problem setting with dimension \( p = 100 \) and tolerance \( \epsilon = 10^{-7} \). We use a randomly generated \( A \) and \( b \) such that at true minima, the value of the objective function is \(-24.1085\). Due to the simple structure of the optimization problem, we might expect all algorithms to converge to the global minimum, and see the unaccelerated MM algorithm converges very slowly. Table 2 reports performance in terms of the median and interquartile range, comparing the number of \( F \) function evaluations (\( F \) evals), wall-clock time, and objective values at convergence over 100 spread-out random initializations centered at the true mean, perturbing each component by Gaussian noise with standard deviation 1000. Initial values are matched across methods for each trial. All methods successfully reach the global minimum in each replication, which may not be surprising given the strongly convex objective. On this nice objective surface, we see that DAAREM performs exceptionally well, taking roughly a third of the time to converge as compared to the next best methods, L-BQN and BQN. At the same time, L-BQN with \( m = 2 \) outperforms all versions of SQUAREM and ZAL. Intuitively, choosing a small \( m \) performs well here due to the smooth and regular objective function, which does not demand a rich approximation to \( H_\theta \). This example also exhibits how increasing the number of secants \( q \) can lead to better performance of BQN. Finally, it is worth noting that the variations of SQUAREM already perform quite differently from one another, suggesting significant sensitivity to the choice of step length.

4.2. Truncated Beta Binomial Incidence Data

We next consider a more difficult statistical problem, turning to the cold incidence data studied by Lidwell and Sommerville (1951). These data have been modeled as a zero-truncated betabinomial model as the reported households have at least one cold

---

**Table 2.** Quadratic minimization for 100 random starting points where the true minima is \(-24.1085\).

| Algorithm | \( F \) evals | Time (in sec) | Objective |
|-----------|--------------|---------------|-----------|
| MM        | 30726 (29186, 319467) | 6.625 (6.284, 6.871) | \(-24.1085, -24.1085, -24.1085\) |
| BQN, \( q = 1 \) | 8935.0 (7374.0, 10204.5) | 0.547 (0.463, 0.634) | \(-24.1085, -24.1085, -24.1085\) |
| BQN, \( q = 2 \) | 4474.0 (3908.5, 5217.0) | 0.310 (0.267, 0.367) | \(-24.1085, -24.1085, -24.1085\) |
| BQN, \( q = 10 \) | 1369.0 (1178.5, 1544.5) | 0.177 (0.153, 0.205) | \(-24.1085, -24.1085, -24.1085\) |
| L-BQN \( m = 2 \) | 5722.0 (4408.0, 7210.0) | 0.158 (0.122, 0.197) | \(-24.1085, -24.1085, -24.1085\) |
| SQUAREM-1 | 2869.0 (2448.5, 3139.0) | 0.190 (0.166, 0.210) | \(-24.1085, -24.1085, -24.1085\) |
| SQUAREM-2 | 14267.0 (12844.0, 15261.0) | 0.964 (0.851, 1.036) | \(-24.1085, -24.1085, -24.1085\) |
| SQUAREM-3 | 8843.0 (7858.5, 9480.0) | 0.604 (0.545, 0.664) | \(-24.1085, -24.1085, -24.1085\) |
| ZAL, \( q = 2 \) | 33227.5 (31847.0, 34337.5) | 2.489 (2.383, 2.553) | \(-24.1085, -24.1085, -24.1085\) |
| ZAL, \( q = 10 \) | 6770.5 (6554.75, 6972.5) | 0.658 (0.635, 0.685) | \(-24.1085, -24.1085, -24.1085\) |
| QN1 | 3029.5 (2829.25, 3613.75) | 0.275 (0.254, 0.319) | \(-24.1085, -24.1085, -24.1085\) |
| DAAREM | 896.0 (848.5, 936.0) | 0.062 (0.055, 0.067) | \(-24.1085, -24.1085, -24.1085\) |

NOTE: Each cell is in the format median (IQR) for the 100 randomly initialized samples.
incidence. The data includes four different household types. We analyze the subset of data corresponding to all adult households here; further details on the data and results for other subsets of the data appear in the Appendix. Among adults, the number of households with 1, 2, 3, and 4 cases are 15, 5, 2, and 2, respectively.

Suppose \( n \) is the total number of independent observations (households) and \( x_i \) denotes the number of cold cases in the \( i \)th household. This can be modeled as a discrete probability model (Zhou, Alexander, and Lange 2011) with likelihood given by

\[
L(\theta|X) = \prod_{i=1}^{n} \frac{d(x_i|\theta)}{1 - d(0|\theta)}.
\]

Here \( d(x|\theta) \) is the probability density function for a beta-binomial distribution with parameter vector \( \theta \) and maximum count \( m = 4 \), with \( \theta = (\alpha, \pi) \) such that \( \pi \in (0, 1) \) and \( \alpha > 0 \). We use MM to numerically maximize the likelihood, whose updates are given by

\[
\alpha_{t+1} = \frac{\sum_{j=0}^{m-1} s_{1j} \alpha_t}{\pi_t + \frac{x_t}{j} \alpha_t + \frac{s_{2j} \alpha_t}{1 - \pi_t + \frac{x_t}{j} \alpha_t}} + \frac{\sum_{j=0}^{m-1} r_j}{1 + \frac{x_t}{j} \alpha_t} \]

\[
\pi_{t+1} = \frac{\sum_{j=0}^{m-1} s_{1j} \pi_t}{\pi_t + \frac{x_t}{j} \alpha_t} + \frac{\sum_{j=0}^{m-1} s_{2j} (1 - \pi_t)}{1 - \pi_t + \frac{x_t}{j} \alpha_t},
\]

where \( s_{1j}, s_{2j}, r_j \) can be interpreted as pseudocounts, given by

\[
s_{1j} = \sum_{i=1}^{n} 1_{x_i \geq j+1}
\]

\[
s_{2j} = \sum_{i=1}^{n} 1_{x_i \leq m-j-1} \left[ \frac{g(0|\pi_t, \alpha_t)}{1 - g(0|\pi_t, \alpha_t)} \right]
\]

\[
r_j = \sum_{i=1}^{n} \left[ 1 + \frac{g(0|\pi_t, \alpha_t)}{1 - g(0|\pi_t, \alpha_t)} \right] 1_{x_i \geq j+1}.
\]

In this constrained parameter space, quasi-Newton based methods can propose updates that fall outside the parameter space. An inexpensive fix for this situation is to fall back on the MM update \( F(x_t) \), similarly to monotonicity control. This solution comes without additional cost since \( F(x_t) \) and \( F^2(x_t) \) are already stored in memory at each step to calculate the proposed \( x_{k+1} \). Following Zhou, Alexander, and Lange (2011), each algorithm is initialized at \( (\pi, \alpha) = (0.5, 1) \). The tolerance is \( \epsilon = 10^{-7} \) and a stricter partial monotonicity parameter \( \delta = 10^{-5} \) is enforced due to the rough objective function. Table 3 lists the negative log-likelihood values, the number of MM evaluations (F evals), the number of algorithm iterations, and runtime until convergence for each algorithm. Figure 2 provides a closer look, showing the progress path of the best performing version of each algorithm on a contour plot of the objective. SQUAREM methods, though achieving significant acceleration, tend to exhibit slow tail behavior near the optimal value. While SQUAREM-1 outstripped its other variants in the first simple example, we found it to be orders of magnitude slower than the others on this example, so we plot the more successful progress of SQUAREM-2 here. In all cases, our method converges in less time than its competitors despite a more naive implementation. From Figure 2, we can visualize the advantage of our extrapolation-based steps making steady progress, in contrast to the more congested updates near the optimum under existing methods. While the small problem dimension does not call for a limited-memory method, we report the results of L-BQN as well for the sake of completeness. We see that it still provides acceleration over the base MM algorithm, but does not compare well with other methods. This is unsurprising because in such a low-dimensional problem, manipulating the inverse Jacobian matrix in a nested manner will be more computationally expensive than simply performing the matrix calculations directly. The advantage of L-BQN becomes more apparent in high-dimensional problems, as we will see below.

### 4.3. Generalized Eigenvalues

In this example, we consider a more complicated objective function that exhibits a zig-zag descent path under the naïve MM algorithm, rendering progress excruciatingly slow. For two \( p \times p \) matrices \( A \) and \( B \), the generalized eigenvalue problem refers to finding a scalar \( \lambda \) and a nontrivial vector \( x \) such that \( Ax = \lambda Bx \). We consider the case where \( A \) is symmetric and \( B \) is symmetric and positive definite so that the generalized eigenvalues and eigenvectors are real (Zhou, Alexander, and Lange 2011). A simple alternative for finding the generalized eigenvalues iteratively is by optimizing the Rayleigh quotient

\[
R(x) = \frac{x^T Ax}{x^T Bx}, \quad x \neq 0.
\]

The gradient of \( R(x) \) is given by

\[
\nabla R(x) = \frac{2}{x^T Bx} [Ax - R(x)Bx].
\]

Therefore, a solution of \( \nabla R(x) = 0 \) corresponds to a generalized eigenpair, wherein the maximum of \( R(x) \) gives the maximum generalized eigenvalue and the minimum gives the minimum generalized eigenvalue. To optimize \( R(x) \), we consider the line search method for the steepest ascent proposed by Hestenes and Karush (1951) as the base algorithm.

In summary, let the current iterate be \( x_k \) and define \( y = x_k \) and \( z = Ax_k + R(x_k)Bx_k \). Starting from \( y \), we search in the direction of \( z \) to maximize and minimize \( R(x) \). That is we search...
along the line $l(c) : c \rightarrow y + cz$. The Rayleigh quotient along this line is written as a function of $c$

$$R(y + cz) = \frac{(y + cz)^T A(y + cz)}{(y + cz)^T B(y + cz)}.$$  

The values of $c$ that maximize and minimize $R(y + cz)$ are found by setting the derivative with respect to $c$, $\nabla_c R(y + cz)$ to zero which involves solving the quadratic equation

\[
\begin{aligned}
&c^2[(z^T Az)(y^T Bz) - (y^T Ay)(z^T Bz)] \\
&+ c[(z^T Az)(y^T By) - (y^T Ay)(z^T Bz)] \\
&+ [(z^T Ay)(y^T By) - (y^T Ay)(z^T By)] = 0.
\end{aligned}
\]

Let $c_1$ and $c_2$ be the maxima and minima of $R(y_k + cz_k)$, respectively: then the EM updates for finding the maximum and minimum eigenvalues are $y_k + c_1 z_k$ and $y_k + c_2 z_k$, respectively.

Due to the zigzag nature of the steepest ascent on this problem, Zhou, Alexander, and Lange (2011) found naïve acceleration to perform poorly. To improve performance, they considered instead the $s$-fold functional composition of the base algorithm for even-valued $s$ as the underlying MM map.

We refrain from using the same heuristic in order to illustrate the off-the-shelf applicability of our method without side information. We consider a simulation study with symmetric matrices $A$ and $B$ randomly generated with $p = 100$ dimensions.

---

**Figure 2.** Truncated beta-binomial: ascent paths of peer methods on the Lidwell and Somerville household incidence data in a truncated beta-binomial model, with optimum marked in red.

(a) MM  
(b) BQN, $q = 2$  
(c) L-BQN  
(d) SQUAREM-2  
(e) ZAL, $q = 2$  
(f) DAAREM
and run 10 random initializations of each method from matched initial points with tolerance $\epsilon = 10^{-5}$.

Table 4 details the median results for all methods.

It can be seen that without the $s$-fold functional composition, both SQUAREM and ZAL fail to accelerate meaningfully here. On this example, QN1 fails to converge on both the minimum and maximum eigenvalue problem, so it is omitted from Table 4. This can likely be attributed to uninformed secants drawn by QN1 in a more complex problem, leading to invalid updates outside the parameter space. Indeed, the curvature information is more crucial to inform a good search direction in such settings.

Table 4 displays the median and IQR of runtime, number of $F$ evaluations while converging to the right objective value. In fact, when we considered a high-dimensional case with $p = 1000$, we observed that all methods except for L-BQN failed to converge. This illustrates the potential of L-BQN in nonsmooth high dimensional settings. Owing to the zig-zag nature of the objective, it appears that stale information from older iterates may mislead the derivative approximations. As a consequence, higher $q$ leads to poorer performance in BQN.

### 4.4. Multivariate $t$-Distribution

This example turns to estimation under a multivariate $t$-distribution, a robust alternative to multivariate normal modeling when the errors involve heavy tails (Lange, Little, and Taylor 1989). Varadhan and Roland (2008) considered this example to compare SQUAREM to standard EM as well as PX-EM, an efficient data augmentation method (Meng and Van Dyk 1997).

Suppose we have $p$-dimensional data $Y = (y_1, \ldots, y_N)$ that we wish to fit to a multivariate $t$-distribution with unknown degrees of freedom $\nu$. The density is given by

$$f(y|\mu, \Sigma) \propto |\Sigma|^{-1/2} (v + (y - \mu)^T \Sigma^{-1} (y - \mu))^{(v+p)/2},$$

and so the data likelihood is given by $\Pi_{i=1}^N f(y_i|\mu, \Sigma)$. There is no closed-form solution to find $(\mu, \Sigma)$ which maximizes the likelihood, but we can make progress by augmenting the missing data with latent variables. That is, we obtain the complete data $(\{y_i, q_i\}; i = 1, \ldots, N)$ where $q_i$ are IID from $\chi^2_\nu/v$; the maximum likelihood estimator (MLE) now follows from weighted least squares. In an EM algorithm, the E-step finds the expected complete data log-likelihood conditional on parameters from the previous iteration $k$. Conditional on $Y$ and $(\mu_k, \Sigma_k)$, the latent variables are distributed as $q_i \sim \chi^2_{v+p}\left(\frac{v}{v + d_i^{(k)}}\right)$, where $d_i^{(k)} = (y_i - \mu_k)^T \Sigma_k^{-1} (y_i - \mu_k); i = 1, \ldots, N$. As the complete-data log-likelihood is linear in $q_i$, the E-step amounts to defining

$$w_i = \ell(q_i|y_i, \mu_k, \Sigma_k) = (v + p)/(v + d_i^{(k)}); \quad i = 1, \ldots, N.$$

The M-step then yields:

$$\mu_{k+1} = \frac{\sum_i w_i y_i / \sum_i w_i}{\Sigma_{k+1} = \frac{1}{N} \sum_i w_i (y_i - \mu)(y_i - \mu)^T}.$$

The PX-EM method of Meng and Van Dyk (1997) differs only in the $\Sigma$ update, replacing the denominator $N$ by $\sum_i w_i$. We randomly generate $N = 10$ synthetic datasets with $v = 1$ (a multivariate Cauchy distribution) and parameters $\mu = 0, \Sigma = V$, where $V$ is a symmetric randomly generated matrix with dimension $p = 25$, which corresponds to 350 parameters (25 for $\mu$ and 325 for $\Sigma$). We report results obtained from following the initial values suggested by Meng and Van Dyk (1997):

$$\mu_0 = \frac{1}{N} \sum_{i=1}^N y_i, \quad \Sigma_0 = \frac{1}{N} \sum_{i=1}^N (y_i - \bar{y})(y_i - \bar{y})^T.$$

Table 5 displays the median and IQR of runtime, number of $F$ evaluations (F evals), and negative log-likelihood of all acceleration schemes at convergence for 10 random starting points matched across methods using tolerance $\epsilon = 10^{-7}$ and partial monotonicity parameter $\delta = 0.1$. Our method achieves significant acceleration compared to the standard EM algorithm, but is outpaced by SQUAREM here. Note that L-BQN performs on par with SQUAREM. Here ZAL fails to provide meaningful acceleration under its implementation in turboEM—we observe it frequently proposes an update such that $\Sigma_k$ is not positive-definite. In these cases, the algorithm reverts to the default MM update, adding additional computational effort, though the implementation in Zhou, Alexander, and Lange (2011) achieves more success. Though performance is
always quite dependent on implementations, we echo the overall theme in the findings of Varadhan and Roland (2008) and Zhou, Alexander, and Lange (2011) that model-specific augmentation under PX-EM performs remarkably well, outpacing all of the more general methods. Interestingly, QNI also performs quite well on this problem. This example illustrates that despite the robust performance of our proposed method across settings, it is worthwhile to exploit problem-specific structure as does PX-EM when possible.

4.5. Sparse Logistic Regression

As suggested by an anonymous reviewer, we include a high-dimensional example to highlight the utility of the low memory variant L-BQN. This example serves to emphasize that when the full proposed method becomes prohibitively slow in high dimensions, its limited memory variant continues to be competitive with peer methods. We consider acceleration of an MM algorithm for distance-penalized objectives in the context of sparse logistic regression, following the method and implementation in Xu et al. (2018). Briefly, we penalize the likelihood with a distance-to-set penalty \( \text{dist}(\theta, C)^2 = \|\theta - P_C(\theta)\|^2 \), where \( C \) is a desired constraint set. To promote sparse solutions, here it is given by the set of vectors with at most \( k \) nonzero entries \( C = \{ z \in \mathbb{R}^p : \|z\|_0 \leq k \} \); note the set is nonconvex, so the algorithm reaches local minimizers. A useful majorization \( \text{dist}(\theta, C)^2 \leq \|\theta - P_C(\theta_k)\|^2 \) follows immediately from the definition of the projection \( P_C(\theta) \) being the closest point in \( C \), allowing us to substitute the current iterate \( \theta_k \) under the projection in the inequality. Because the penalties are smooth with \( \nabla^2 \text{dist}(\theta, C)^2 = I - P_C'(x) \), MM algorithms under these penalties entail only slight modifications to Fisher scoring for fitting logistic regression or other generalized linear models subject to \( C \). For brevity, we refer the reader to Xu, Chi, and Lange (2017) and Xu et al. (2018) for details of the algorithm, which are also included in the supplemental material.

We compare the peer acceleration methods on this problem in a problem setting with \( n = 1500 \) data points and \( p = 2000 \) dimensions, where the true coefficient vector has \( k = 10 \) nonzero entries. Table 6 presents the number of \( F \)-evaluations, wall clock time, and objective value at convergence, under a tolerance of \( \epsilon = 10^{-6} \) and randomly generated starting points. Compared to the baseline MM algorithm, all methods succeed in converging to a similar objective value, though we note that ZAL failed to converge from one out of the 10 random initializations. While L-BQN is consistently faster, DAAREM and SQUAREM-1 also provide good acceleration in this setting. We notice that the gap widens as we increase the problem dimension. Omitting a full simulation study due to runtime, we found that BQN runs for almost two hours when increasing the problem size to \( p = 3000, k = 100 \), while DAAREM and L-BQN (\( m = 5 \)) remain reasonable and terminate in roughly twenty minutes. The key takeaway here is that when BQN becomes prohibitively slow in high dimensions, L-BQN enables the advantages of the proposed methods to continue to apply.

5. Conclusion

This article presents a novel quasi-Newton acceleration of MM algorithms that extends recent ideas but lends them new

---

**Table 5.** Multivariate t-distribution: number of \( F(x) \) evaluations, runtime, and negative log-likelihood at convergence.

| Algorithm      | F Evals | Time (in sec) | Objective value |
|----------------|---------|---------------|-----------------|
| EM             | 500.0 (499.0, 502.5) | 5.813 (5.777, 5.845) | -30146.6 (−30937.4, −29819.1) |
| PX-EM          | 13.5 (13.0, 14.0)   | 0.157 (0.156, 0.166) | -30146.6 (−30937.4, −29819.1) |
| BQN, \( q = 1 \) | 60 (53, 72)        | 0.857 (0.733, 0.987) | -30146.6 (−30937.4, −29819.1) |
| BQN, \( q = 2 \) | 89 (72, 103)       | 1.229 (1.034, 1.392) | -30146.6 (−30937.4, −29819.1) |
| BQN, \( q = 10 \) | 109.0 (101.5, 120.0) | 1.504 (1.386, 1.702) | -30146.6 (−30937.4, −29819.1) |
| L-BQN, \( m = 10 \) | 47.00 (45.50, 49.75) | 0.644 (0.613, 0.668) | -30146.6 (−30937.4, −29819.1) |
| SQUAREM-1      | 31.5 (29.0, 33.0)  | 0.621 (0.580, 0.663) | -30146.6 (−30937.4, −29819.1) |
| SQUAREM-2      | 83.0 (33.0, 209.5) | 1.699 (0.637, 3.439) | -30146.6 (−30937.4, −29819.1) |
| SQUAREM-3      | 33 (31.35)         | 0.646 (0.614, 0.681) | -30146.6 (−30937.4, −29819.1) |
| ZAL, \( q = 1 \) | 15.0 (12.00, 16.00) | 4.832 (4.700, 6.200) | -30146.6 (−30937.4, −29819.1) |
| ZAL, \( q = 2 \) | 262.5 (262.00, 263.75) | 7.120 (7.052, 7.145) | -30146.6 (−30937.4, −29819.1) |
| ZAL, \( q = 10 \) | 267.5 (266.25, 267.85) | 7.165 (7.100, 7.269) | -30146.6 (−30937.4, −29819.1) |
| QNI            | 18 (18, 18)        | 0.297 (0.295, 0.302) | -30146.6 (−30937.4, −29819.1) |
| DAAREM         | 342.5 (327.25, 392.50) | 5.963 (5.507, 6.934) | -30146.6 (−30937.4, −29819.1) |

**Table 6.** Sparse logistic regression.

| Algorithm      | F Evals | Time (in mins) | Objective value |
|----------------|---------|---------------|-----------------|
| MM             | 159.5 (119.0, 185.7) | 31.20 (21.44, 37.64) | 593.069 (560.538, 642.334) |
| BQN, \( q = 2 \) | 86.0 (107.00, 163.00) | 9.376 (11.64, 17.56) | 591.950 (552.950, 630.475) |
| L-BQN, \( m = 5 \) | 33.0 (27.00, 43.25) | 3.539 (2.895, 4.637) | 614.250 (530.875, 644.375) |
| L-BQN, \( m = 10 \) | 38.0 (35.00, 52.75) | 4.071 (3.751, 5.652) | 617.400 (562.398, 674.307) |
| SQUAREM-1      | 49.0 (42.00, 56.75) | 5.245 (4.490, 6.075) | 593.760 (567.797, 643.010) |
| SQUAREM-2      | 59.5 (46.75, 64.25) | 6.369 (5.005, 6.876) | 614.010 (561.271, 674.166) |
| SQUAREM-3      | 57.5 (54.25, 76.00) | 6.155 (5.808, 8.135) | 592.422 (550.678, 626.350) |
| ZAL            | 15.0 (12.00, 16.00) | 4.832 (4.700, 6.200) | 633.832 (590.333, 640.552) |
| DAAREM         | 35.0 (29.25, 42.25) | 7.802 (6.847, 8.620) | 592.307 (524.001, 645.889) |

NOTE: Each cell is in the format median (IQR) for 10 replications with random starting points.
intuition as well as theoretical guarantees. The method retains gradient information across all components of $F$, which is often ignored in other pure MM accelerators. A key advantage of MM algorithms is their transfer of difficulty away from the original objective function, obtained by the construction of surrogates. While the hybrid quasi-Newton accelerators ensure monotonicity, their appeal is compromised in part by requiring gradient information from the original objective. Our approach seeks to embody the best of both worlds, retaining the simplicity of pure accelerators without restrictive assumptions, maintaining computational tractability so that it is amenable to large and high-dimensional problems, and taking advantage of richer curvature information that yields classical convergence guarantees.

In our representative but inexhaustive empirical study, we find that the limited memory version shows promising performance especially in high-dimensional settings where quasi-Newton methods are prohibitively slow. A fruitful line of research may seek to study the convergence properties of L-BQN explicitly, based on previous analyses on the convergence of the limited memory BFGS method (Liu and Nocedal 1989). Exploring optimal step size selection is another open direction (Nocedal and Wright 2006). Despite being derived from different perspectives, it is satisfying that the steplength for our inverse Jacobian update (9) agrees with the steplength derived in the first version of the STEM method by Varadhan and Roland (2008). Continuing to study the practical and theoretical merits of alternative methods for steplength selection may yield further advantages.

Supplementary Materials

Proofs and Examples: Contains proofs of the theoretical results presented in the article. Contains simulation for the Poisson mixture example, additional results from Example 4.2, and additional details from Example 4.5.

R Code: Contains R code that reproduces the simulation results, figures, and tables.

Disclosure Statement

The authors report that there are no competing interests to declare.

Funding

Jason Xu is kindly supported by NSF grant DMS-2230074.

References

Boyles, R. A. (1983), “On the Convergence of the EM Algorithm,” Journal of the Royal Statistical Society, Series B, 45, 47–50. [393]
Brotherson, C. G. (1965), “A Class of Methods for Solving Nonlinear Simultaneous Equations,” Mathematics of Computation, 19, 577–593. [394,397]
Brodyrden, C. G., Dennis Jr, J., and Moré, J. J. (1973), “On the Local and Superlinear Convergence of Quasi-Newton Methods,” IMA Journal of Applied Mathematics, 12, 223–245. [394,398,399]
Defazio, A., Bach, F., and Lacoste-Julien, S. (2014), “SAGA: A Fast Incremental Gradient Method with Support for Non-strongly Convex Composite Objectives,” in Proceedings of the 27th International Conference on Neural Information Processing Systems (Vol. 1), pp. 1646–1654. [393]
Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977), “Maximum Likelihood from Incomplete Data via the EM Algorithm,” Journal of the Royal Statistical Society, Series B, 39, 1–22. [393]
Dennis Jr, J. E., and Schnabel, R. B. (1996), Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Philadelphia, PA: SIAM. [394]
Everitt, B. (2012), Introduction to Optimization Methods and their Application in Statistics, Dordrecht: Springer. [393]
Fletcher, R. (2013), Practical Methods of Optimization, New York: Wiley. [397]
Henderson, N. C., and Varadhan, R. (2019), “Damped Anderson Acceleration with Restarts and Monotonicity Control for Accelerating EM and EM-like Algorithms,” Journal of Computational and Graphical Statistics, 28, 834–846. [395]
Hestenes, M., and Karush, W. (1951), “Solutions of A$x$=$\lambda$B$x$,” Journal of Research of the National Bureau of Standards, 47, 471–478. [401]
Jamiandidian, M., and Jennrich, R. I. (1993), “Conjugate Gradient Acceleration of the EM Algorithm,” Journal of the American Statistical Association, 88, 221–228. [393]
——— (1997), “Acceleration of the EM Algorithm by Using Quasi-Newton Methods,” Journal of the Royal Statistical Society, Series B, 59, 569–587. [393,394,399]
Laird, N. (1978), “Nonparametric Maximum Likelihood Estimation of a Mixing Distribution,” Journal of the American Statistical Association, 73, 805–811. [393]
Lange, K. (1995), “A Quasi-Newton Acceleration of the EM Algorithm,” Statistica Sinica, 5, 1–18. [393]
——— (2016). MM Optimization Algorithms, Philadelphia, PA: SIAM. [395,398,400]
Lange, K., and Wu, T. (2008), “An MM Algorithm for Multicategory Vertex Discriminant Analysis,” Journal of Computational and Graphical Statistics, 17, 527–544. [394]
Lange, K. L., Little, R. J., and Taylor, J. M. (1989), “Robust Statistical Modeling Using the t Distribution,” Journal of the American Statistical Association, 84, 881–896. [403]
Lee, D. D., and Seung, H. S. (1999), “Learning the Parts of Objects by Non-negative Matrix Factorization,” Nature, 401, 788–791. [393]
Lidwell, O., and Sommerville, T. (1951), “Observations on the Incidence and Distribution of the Common Cold in a Rural Community During 1948 and 1949,” Epidemiology & Infection, 49, 365–381. [400]
Lin, H., Mairal, J., and Harchaoui, Z. (2017), “A Generic Quasi-Newton Algorithm for Faster Gradient-based Optimization,” hal-01376079v2. [393]
Liu, D. C., and Nocedal, J. (1989), “On the Limited Memory BFGS Method for Large Scale Optimization,” Mathematical Programming, 45, 503–528. [397,405]
Luengerer, D. G., and Ye, Y. (1984), Linear and Nonlinear Programming (Vol. 2), New York: Springer. [394]
Meng, X.-L., and Rubin, D. B. (1994), “On the Global and Componentwise Rates of Convergence of the EM Algorithm,” Linear Algebra and its Applications, 199, 413–425. [393]
Meng, X.-L., and Van Dyk, D. (1997), “The EM Algorithm–An Old Folk-Song Sung to a Fast New Tune,” Journal of the Royal Statistical Society, Series B, 59, 511–567. [403]
Nesterov, Y. E. (1983), “A Method for Solving the Convex Programming Problem with Convergence Rate $o(1/k^2)$,” Doklady Akademii Nauk SSSR, 269, 543–547. [393]
Nocedal, J. (1980), “Updating Quasi-Newton Matrices with Limited Storage,” Mathematics of Computation, 35, 773–782. [397,398]
Nocedal, J., and Wright, S. (2006), Numerical Optimization, New York: Springer. [398,405]
Olega, J. M., and Rheinboldt, W. C. (2000), Iterative Solution of Nonlinear Equations in Several Variables, Philadelphia, PA: SIAM. [399]
Pearson, J. D. (1969), “Variable Metric Methods of Minimisation,” The Computer Journal, 12, 171–178. [397]
Schmidt, M., Le Roux, N., and Bach, F. (2017), “Minimizing Finite Sums with the Stochastic Average Gradient,” Mathematical Programming, 162, 83–112. [393]
Shalev-Shwartz, S., and Zhang, T. (2014), "Accelerated Proximal Stochastic Dual Coordinate Ascent for Regularized Loss Minimization," in International Conference on Machine Learning, pp. 64–72, PMLR. [393]

Shanno, D. F. (1978), "Conjugate Gradient Methods with Inexact Searches," Mathematics of Operations Research, 3, 244–256. [397]

Varadhan, R., and Roland, C. (2008), "Simple and Globally Convergent Methods for Accelerating the Convergence of any EM Algorithm," Scandinavian Journal of Statistics, 35, 335–353. [394,395,397,398,403,404,405]

Wu, C. J. (1983), "On the Convergence Properties of the EM Algorithm," The Annals of Statistics, 11, 95–103. [393]

Xu, J., Chi, E., and Lange, K. (2017), "Generalized Linear Model Regression Under Distance-to-Set Penalties," in Advances in Neural Information Processing Systems (Vol. 30). [404]

Xu, J., Chi, E. C., Yang, M., and Lange, K. (2018), "A Majorization-Minimization Algorithm for Split Feasibility Problems," Computational Optimization and Applications, 71, 795–828. [404]

Xu, J., and Lange, K. (2019), "Power k-means Clustering," in International Conference on Machine Learning, pp. 6921–6931, PMLR. [394]

Zhou, H., Alexander, D., and Lange, K. (2011), "A Quasi-Newton Acceleration for High-Dimensional Optimization Algorithms," Statistics and Computing, 21, 261–273. [393,394,395,397,400,401,402,403,404]

Zhou, Z., Hu, Z.-g., Song, T., and Yu, J.-y. (2015), "A Novel Virtual Machine Deployment Algorithm with Energy Efficiency in Cloud Computing," Journal of Central South University, 22, 974–983. [394]

Zou, H., and Li, R. (2008), "One-Step Sparse Estimates in Nonconcave Penalized Likelihood Models," Annals of Statistics, 36, 1509–1533. [393]