algorithms

MDPI

Article

Coordinate Descent for Variance-Component Models

Anant Mathur [1,*], Sarat Moka [2] and Zdravko Botev [1,*]

[1] School of Mathematics and Statistics, University of New South Wales, Sydney, NSW 2052, Australia
[2] School of Mathematical and Physical Sciences, Macquarie University, Sydney, NSW 2109, Australia
[*] Correspondence: anant.mathur@unsw.edu.au (A.M.); botev@unsw.edu.au (Z.B.)

Abstract: Variance-component models are an indispensable tool for statisticians wanting to capture both random and fixed model effects. They have applications in a wide range of scientific disciplines. While maximum likelihood estimation (MLE) is the most popular method for estimating the variance-component model parameters, it is numerically challenging for large data sets. In this article, we consider the class of coordinate descent (CD) algorithms for computing the MLE. We show that a basic implementation of coordinate descent is numerically costly to implement and does not easily satisfy the standard theoretical conditions for convergence. We instead propose two parameter-expanded versions of CD, called PX-CD and PXI-CD. These novel algorithms not only converge faster than existing competitors (MM and EM algorithms) but are also more amenable to convergence analysis. PX-CD and PXI-CD are particularly well-suited for large data sets—namely, as the scale of the model increases, the performance gap between the parameter-expanded CD algorithms and the current competitor methods increases.

Keywords: linear-mixed models; maximum likelihood estimation; numerical optimization

check for updates

Citation: Mathur, A.; Moka, S.; Botev, Z. Coordinate Descent for Variance-Component Models. *Algorithms* **2022**, *15*, 354. https://doi.org/10.3390/a15100354

Academic Editors: Georgy Sofronov and Andras Farago

Received: 30 July 2022
Accepted: 23 September 2022
Published: 28 September 2022

1. Introduction

Linear models that contain both fixed and random effects are referred to as variance components or linear-mixed models (LMMs). They arise in numerous applications, such as genetics [1], biology, economics, epidemiology and medicine. A broad coverage of existing methodologies and applications of these models can be found in the textbooks [2,3].

In the simplest variance component setup, we observe a response vector $\boldsymbol{y} \in \mathbb{R}^n$ and a predictor matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ and assume that $\boldsymbol{y}$ is an outcome of a normal random variable $\boldsymbol{Y} \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \boldsymbol{\Omega})$, where the covariance is of the form,

$$\boldsymbol{\Omega} = \sum_{i=0}^{m} \gamma_i \mathbf{V}_i \in \mathbb{R}^{n \times n}, \quad \gamma_i \geq 0.$$

The matrices $\mathbf{V}_0, \ldots, \mathbf{V}_m$ are fixed positive semi-definite matrices, and $\mathbf{V}_0$ is non-singular. The unknown mean effects $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)$ and variance component parameters $\boldsymbol{\gamma} = (\gamma_0, \ldots, \gamma_m)$ can be estimated by maximizing the log-likelihood function,

$$L(\boldsymbol{\beta}, \boldsymbol{\gamma}) = -\frac{1}{2} \ln \det \boldsymbol{\Omega} - \frac{1}{2} (\boldsymbol{y} - \mathbf{X}\beta)^{\top} \boldsymbol{\Omega}^{-1} (\boldsymbol{y} - \mathbf{X}\beta). \tag{1}$$

If $\boldsymbol{\Omega}$ is known, the maximum likelihood estimator (MLE) for $\boldsymbol{\beta}$ is given by

$$\hat{\beta} = \left(\mathbf{X}^{\top} \boldsymbol{\Omega}^{-1} \mathbf{X}\right)^{-1} \mathbf{X}^{\top} \boldsymbol{\Omega}^{-1} \boldsymbol{y}.$$

To simplify the MLE estimate for γ, one can adopt the restricted MLE (REML) method [4] to remove the mean effect in the likelihood expression by projecting $\boldsymbol{y}$ onto the null space of $\mathbf{X}$. Let $\nu = n - p$ and suppose we have the QR decomposition,

$$\mathbf{X} = \begin{bmatrix} \mathbf{Q}_{[p]} & \mathbf{Q}_{[\nu]} \end{bmatrix} \begin{bmatrix} \mathbf{R} \\ \mathbf{0}_{\nu \times p} \end{bmatrix},$$

where $\mathbf{R}$ is an $p \times p$ upper triangular matrix, $\mathbf{0}_{\nu\times p}$ is an $\nu \times p$ zero matrix, $\mathbf{Q}_{[p]}$ is an $n \times p$ matrix, $\mathbf{Q}_{[\nu]}$ is $n \times \nu$, and $\mathbf{Q}_{[p]}$ and $\mathbf{Q}_{[\nu]}$ both have orthogonal columns. If we take the Cholesky decomposition $\mathbf{L}\mathbf{L}^\top$ of the matrix $\mathbf{Q}_{[\nu]}^\top \mathbf{V}_0 \mathbf{Q}_{[\nu]}$, then the transformation, $\mathbf{L}^{-1}\mathbf{Q}_{[\nu]}^\top : \mathbb{R}^n \mapsto \mathbb{R}^\nu$ removes the mean from the response, and we obtain $\boldsymbol{y}' := \mathbf{L}^{-1}\mathbf{Q}_{[\nu]}^\top \boldsymbol{y} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Omega}')$, where $\boldsymbol{\Omega}' := \gamma_0 \mathbf{I} + \sum_{i=1}^m \gamma_i \mathbf{V}_i' \in \mathbb{R}^{\nu\times\nu}$ and $\mathbf{V}_i' := \mathbf{L}^{-1}\mathbf{Q}_{[\nu]}^\top \mathbf{V}_i \mathbf{Q}_{[\nu]} \mathbf{L}^{-\top}$. After this transformation, the restricted likelihood $-\frac{1}{2}\ln\det\boldsymbol{\Omega}' - \frac{1}{2}(\boldsymbol{y}')^\top(\boldsymbol{\Omega}')^{-1}\boldsymbol{y}'$ will not depend on $\boldsymbol{\beta}$.

Henceforth, and without loss of generality, we assume that such a transformation has been performed so that we can focus on minimizing an objective function of the form:

$$-2L(\boldsymbol{\gamma}) = \ln\det\boldsymbol{\Omega} + \boldsymbol{y}^\top\boldsymbol{\Omega}^{-1}\boldsymbol{y}. \tag{2}$$

There exists extensive literature on optimization methods for the log-likelihood expression (2), including Newton's Method [5], Fisher Scoring Method [6], the EM and MM Algorithms [7–9]. Newton's method is known to scale poorly as n, or the number of variance components $m+1$, increase due the cost of $O(mn^3) + O(m^3)$ flops required to invert a Hessian matrix at each update. Both the EM and MM algorithms have simple updating steps; however, numerical experience shows that they are slow to identify the active set $\{i : \hat{\gamma}_i = 0\}$, where $\hat{\boldsymbol{\gamma}}$ is the MLE.

One class of algorithms yet to be applied to this problem are coordinate-descent (CD) algorithms. These algorithms successively minimize the objective function along coordinate directions and can be effective when the optimization for each sub-problem can be made sufficiently simple. Furthermore, only few assumptions are needed to prove that accumulation points of the iterative sequence are stationary points of the objective function. CD algorithms have been used to solve optimization problems for many years, and their popularity has grown considerably in the past few decades because of their usefulness in data science and machine learning tasks. Further in-depth discussions of CD algorithms can be found in the articles [10–12].

In this paper, we show that a basic implementation of CD is costly for large-scale problems and does not easily succumb to standard convergence analysis. In contrast, our novel coordinate-descent algorithm called parameter-expanded coordinate descent (PX-CD) is computationally faster and more amenable to theoretical guarantees.

The PX-CD is computationally cheaper to run than the basic CD implementation because the first and second derivatives for each sub-problem can be evaluated efficiently with the conjugate gradient (CG) algorithm [13], whereas the basic CD implementation requires repeat Cholesky factorizations for each coordinate update, each with a complexity of $O(n^3)$. Further to this, it is often the case that the $\mathbf{V}_i$ are low-rank, and we can take advantage of this by employing the well-known Woodbury matrix identity or QR transformation within PX-CD to reduce the computational cost of each univariate minimization.

In PX-CD, the extended parameters are treated as a block of coordinates, which is updated at each iteration by searching through a coordinate hyper-plane rather than single-coordinate directions. We provide an alternate version of PX-CD, which we call parameter expanded-immediate coordinate descent (PXI-CD), where the extended coordinate block is updated multiple times within each cycle of the original parameters. We observe numerically that, for large-scale models, the number of iterations needed to converge greatly offsets the additional computational cost for each coordinate cycle. As a result, the overall convergence time is better than that of the PX-CD.

From a theoretical point of view, we show that the accumulation points of the iterative sequence generated by both the PX-CD and PXI-CD are coordinate-wise minimum points of (2).

We remark that the improved efficiency of the PX-CD algorithm is similar to the well-known superior performance of the PX-DA (parameter-expanded data-augmentation) algorithm [14,15] in the Markov-chain Monte Carlo (MCMC) context—namely, the PX-DA algorithm is often much faster to converge than a basic data-augmentation Gibbs algorithm. This similarity is also the reason for using the same prefix "PX" in our nomenclature.

The remainder of the paper is structured as follows. In Section 2, we describe the basic implementation of CD and provide examples for which it performs unsatisfactorily. In Section 3, we introduce the PX-CD and PXI-CD and show that accumulation points of the iterations are coordinate-wise minima for the optimization. We then discuss their practical implementation and detail how to reduce the computational cost when the $\mathbf{V}_i$ are low-rank. We also extend the PX-CD algorithm for penalized estimation to perform variable selection. Then, in Section 4, we provide numerical results when $\mathbf{V}_i$ are computer simulated and when $\mathbf{V}_i$ are constructed from a real-world genetic data set. We have made our code for these simulations available on GitHub (https://github.com/anantmathur44/CD-for-variance-components) (accessed on 1 July 2022).

2. Basic Coordinate Descent

Recall that, after the REML procedure to remove the mean effect, we have $\boldsymbol{Y} \sim \mathcal{N}(\mathbf{0}, \Omega)$ and $\boldsymbol{\Omega} = \sum_{i=0}^{m} \gamma_i \mathbf{V}_i$, where $\mathbf{V}_0 = \mathbf{I}_n$. We thus seek to compute:

$$\hat{\gamma} = \underset{\gamma}{\arg\min}\ G(\boldsymbol{\Omega}), \quad \gamma_i \geq 0, \quad i = 0, 1 \ldots, m \tag{3}$$

$$G(\boldsymbol{\Omega}) := \boldsymbol{y}^\top \boldsymbol{\Omega}^{-1} \boldsymbol{y} + \ln \det \boldsymbol{\Omega}. \tag{4}$$

CD can be applied to solve this problem by successively minimizing G along coordinate directions. There is significant scope for variation in the way components are selected to be updated. In the most conventional way the algorithm cycles through the parameters in the order $\gamma_0 \to \gamma_1 \to \cdots \to \gamma_m$ and updates each in turn. This version, known as cyclic coordinate descent, is shown in Algorithm 1.

Algorithm 1 Cyclic CD for $G(\boldsymbol{\Omega})$.

```
input: $\boldsymbol{y}$ and $[\mathbf{V}_0, \mathbf{V}_1, \ldots, \mathbf{V}_m]$ where $\mathbf{V}_0 = \mathbf{I}_n$
Choose $\boldsymbol{\gamma} \in \{\mathbf{u} \in \mathbb{R}^{m+1} \mid u_i > 0\}$
$\boldsymbol{\Omega} \leftarrow \gamma_0 \mathbf{I} + \sum_{i=1}^{m} \gamma_i \mathbf{V}_i$
repeat
    for $k = 0, 1, \ldots, m$ do
        $\boldsymbol{\Omega}_{-k} \leftarrow \boldsymbol{\Omega} - \gamma_k \mathbf{V}_k$
        $\gamma_k \leftarrow \underset{x \geq 0}{\arg\min}\ G(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)$
        $\boldsymbol{\Omega} \leftarrow \boldsymbol{\Omega}_{-k} + \gamma_k \mathbf{V}_k$
until termination test satisfied
return $\gamma, \boldsymbol{\Omega}$
```

If we choose to minimize along one coordinate at a time, then the update of a parameter consists of a line search along the selected coordinate direction—that is, if the selected parameter is component k, then the new parameter value is updated as

$$\gamma_k^{(t+1)} \leftarrow \underset{x \geq 0}{\arg\min}\ G\left(\sum_{i=0}^{k-1} \gamma_i^{(t+1)} \mathbf{V}_i + \sum_{i=k+1}^{m} \gamma_i^{(t)} \mathbf{V}_i + x\mathbf{V}_k\right).$$

2.1. Implementation

The non-trivial component of Algorithm 1 is to compute the line search,

$$\arg\min_{x \geq 0} G(\mathbf{\Omega}_{-k} + x\mathbf{V}_k),$$

where $\mathbf{\Omega}_{-k} := \mathbf{\Omega} - \gamma_k \mathbf{V}_k$. When $\mathbf{\Omega}_{-k}$ is invertible, this step can be simplified to a one-dimensional algebraic expression that can be numerically solved without repeated evaluations of the terms $\boldsymbol{y}^\top(\mathbf{\Omega}_{-k} + x\mathbf{V}_k)\boldsymbol{y}$ and $\ln\det(\mathbf{\Omega}_{-k} + x\mathbf{V}_k)$. The simplification is achieved using the Generalized Eigenvalue Decomposition (GEV) [13] to decompose $(\mathbf{V}_k, \mathbf{\Omega}_{-k}) \to (\mathbf{D}_k, \mathbf{U}_k)$ such that,

$$\mathbf{V}_k\mathbf{U}_k = \mathbf{\Omega}_{-k}\mathbf{U}_k\mathbf{D}_k \quad \text{and} \quad \mathbf{U}_k^\top\mathbf{\Omega}_{-k}\mathbf{U}_k = \mathbf{I}_n,$$

where $\mathbf{D}_k \in \mathbb{R}^{n\times n}$ is a diagonal matrix with non-negative entries and $\mathbf{U}_k \in \mathbb{R}^{n\times n}$ is invertible. Using the above expressions we obtain the factorized expression, $\mathbf{\Omega}_{-k} + x\mathbf{V}_k = \mathbf{U}_k^{-\top}(\mathbf{I} + x\mathbf{D})\mathbf{U}_k^{-1}$. Therefore, we can express the inverse and log-determinant terms in $G(\mathbf{\Omega}_{-k} + x\mathbf{V}_k)$ as,

$$(\mathbf{\Omega}_{-k} + x\mathbf{V}_k)^{-1} = \mathbf{U}_k(\mathbf{I}_n + x\mathbf{D}_k)^{-1}\mathbf{U}_k^\top, \tag{5}$$

$$\ln\det(\mathbf{\Omega}_{-k} + x\mathbf{V}_k) = \ln\det(\mathbf{I}_n + x\mathbf{D}_k) - 2\ln\det\mathbf{U}_k. \tag{6}$$

From (4), we have that,

$$G(\mathbf{\Omega}_{-k} + x\mathbf{V}_k) = \boldsymbol{y}^\top\mathbf{U}_k(\mathbf{I}_n + x\mathbf{D}_k)^{-1}\mathbf{U}_k^\top\boldsymbol{y} + \ln\det(\mathbf{I}_n + x\mathbf{D}_k) + \text{const.}$$

Let $\boldsymbol{\alpha}_k = \mathbf{U}_k^\top\boldsymbol{y}$ and $\boldsymbol{d}_k = \operatorname{diag}(\mathbf{D}_k)$. Then, the function to be minimized at the k-th component is of the form,

$$g_k(x) := G(\mathbf{\Omega}_{-k} + x\mathbf{V}_k) = \sum_{j=1}^{n} \frac{\alpha_{k,j}^2}{d_{k,j}x + 1} + \ln(1 + d_{k,j}x) + \text{const.} \tag{7}$$

where $d_j \geq 0$ for all j. Unless $n = 1$, in general, there is no closed-form expression for the minimum. We thus resort to a numerical method, such as Newton's method or the Golden-section search method. With the above simplification the majority of the cost is attributed to the GEV, which has a time complexity of $O(14n^3)$. Alternatively, one could employ iterative methods without prior simplification of $g_k(x)$ via the GEV. In that case, however, evaluation of $g_k(x)$ and its derivatives at each step requires one full Cholesky factorization, costing $O(n^3)$. Either way, for problems where n is large, basic CD is too costly per one update.

2.2. Convergence

An interesting question is whether the sequence generated in Algorithm 1 converges to a local minimum of the objective function (3) (which is assumed to have a global minimum). To make a general point about the convergence theory, take $\mathcal{C}$ to be the set of limit points of a coordinate-descent sequence $\{\gamma^{(t)}\}$. It is well-known [16,17] that the following existence and uniqueness assumption:

$$g_k(x) = G(\mathbf{\Omega}_{-k} + x\mathbf{V}_k) \text{ has a unique (global) minimizer for } x \geq 0 \tag{8}$$

is one of the simplest sufficient conditions that ensures the set $\mathcal{C}$ is not empty and contains only singletons. Assuming that the existence and uniqueness assumption holds for a given coordinate-descent algorithm and $\gamma^* \in \mathcal{C}$ then, by construction, γ^* is a global *coordinate-wise minimum* of (3)—that is, one cannot reduce the value of the objective function by moving along each of the coordinate directions (that is, $G(\mathbf{\Omega}_{-k} + x\mathbf{V}_k) \geq G(\mathbf{\Omega}_{-k} + \gamma_k^*\mathbf{V}_k)$ for each k and all $x \geq 0$).

Even if $\gamma^* \in \mathcal{C}$ is a coordinate-wise minimum, it may not be a local minimum of (3). It may well be a saddle point of (3). For example, the function $f(\gamma_1, \gamma_2) = \gamma_1^2 + \gamma_2^2 - 5\gamma_1\gamma_2$ does not have a local minimum at zero ($f(\gamma, \gamma) = -3\gamma^2$ indicating that a minimum does not even exist); however, it has a coordinate-wise minimum at $(\gamma_1, \gamma_2) = (0, 0)$.

Thus, it is important to remember that the set of local minima of the optimization problem (3) is a subset of $\mathcal{C}$ because $\mathcal{C}$ may contain coordinate-wise minima, which are still saddle points of (3). One positive aspect is that the saddle points found by any coordinate-descent algorithm (under the existence and uniqueness assumption) are a subset of all the saddle points of (3) because they are constrained to be coordinate-wise minima. Stated another way, the set $\mathcal{C}$ consists of either local minimizers or saddle points that look like coordinate-wise minimizers.

Either way, there is simply no guarantee that any of our coordinate-descent procedures in this paper will converge to a strict local minimum of (3), see [16,18] for more in-depth discussions. Of course, this is an issue for any of the existing optimization algorithms for (3) (MM and EM algorithms simply being a special case of coordinate descent, and Newton's method known for convergence only when initialized near a local minimum), and thus it should not be viewed as a particular disadvantage of our proposals.

Unfortunately, the existence and uniqueness assumption (8) cannot be used to deal with the convergence of the basic CD Algorithm 1. This is because $g_k(x)$ can exhibit multiple local minima, as illustrated next.

Suppose $n = 2$, then from Equation (7), we have

$$g_k(x) = \frac{\alpha_{k,1}^2}{d_{k,1}x + 1} + \ln(1 + d_{k,1}x) + \frac{\alpha_{k,2}^2}{d_{k,2}x + 1} + \ln(1 + d_{k,2}x) + \text{const.}$$

In Figure 1, we observe two minimizers for $g_k(x)$ when $\boldsymbol{\alpha}_k = (1.2, 3)$ and $\mathbf{d}_k = (10, 0.2)$, respectively. This implies that the existence and uniqueness assumption does not hold for the basic implementation of CD, and we cannot ensure that accumulation points of the sequence $\{\gamma^{(t)}\}$ are coordinate-wise minima of G.

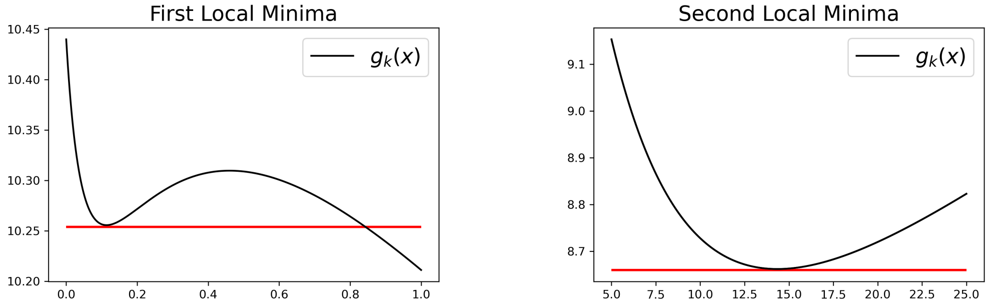

Figure 1. Two local minima: (0.11, 10.26) and (14.35, 8.66) obtained for $g_k(x)$ when $\alpha_{k,1} = 1.2$, $\alpha_{k,2} = 3$, $d_{k,1} = 10$ and $d_{k,2} = 0.2$.

Example 1 (Sufficient Conditions for a Unique Minimum). *In this example, we show that strong conditions are needed for the existence and uniqueness assumption to hold, making the basic CD Algorithm 1 less than attractive. Suppose* $\delta := \frac{1}{n}\sum_{j=1}^{n} \alpha_{k,j}^2 > 1$ *and there is a constant* $d > 0$ *such that* $d_{k,j} = d$ *for all* $j = 1, \ldots, n$. *In that case, we have*

$$g_k(x) = \frac{n\,\delta}{dx + 1} + n\ln(1 + dx) + \text{const.}$$

Therefore, the first and second derivatives of $g_k(x)$ *are, respectively, given by*

$$g_k'(x) = \frac{n\,d}{(1 + dx)}\left(1 - \frac{\delta}{(1 + dx)}\right) \quad \textit{and} \quad g_k''(x) = -\frac{n\,d^2}{(1 + dx)^2}\left(1 - \frac{2\,\delta}{(1 + dx)}\right).$$

Since $d > 0$, it easy to see that the equation $g_k'(x) = 0$ has a unique (positive) solution, which is $x_1^* = \frac{\delta-1}{d}$. Similarly, the solution of $g_k''(x) = 0$ is $x_2^* = \frac{2\delta-1}{d}$, which is greater than x_1^*.

Since $g_k''(x) > 0$ for every $x \in [0, x_2^*)$, $g_k(x)$ is (strictly) convex over $[0, x_2^*)$. As a result, since $0 < x_1^* < x_2^*$, $g_k(x)$ exhibits a global minimum at x_1^*.

3. Parameter-Expanded CD

Since the basic CD Algorithm 1 is both expensive per coordinate update and is not amenable to standard convergence analysis [16,17], we consider an alternative called the parameter-expanded CD or PX-CD. We argue that our novel coordinate-descent algorithm is both faster to converge and also amenable to simple convergence analysis because the existence and uniqueness assumption holds. This constitutes our main contribution.

In the PX-CD, we use the supporting hyper-plane (first-order Taylor approximation) to the concave matrix function $f(\mathbf{A}) = \ln\det\mathbf{A}$, where $\mathbf{A} \succ \mathbf{0}$. The supporting hyper-plane gives the bound [9]:

$$\ln\det\mathbf{A} \le \ln\det\mathbf{C} + \operatorname{tr}\Big(\mathbf{C}^{-1}(\mathbf{A}-\mathbf{C})\Big),$$

where $\mathbf{C} \in \mathbb{R}^{n\times n}$ is an arbitrary PSD matrix, and equality is achieved if and only if $\mathbf{C} = \mathbf{A}$. Replacing the log-determinant term in G with the above upper bound, we obtain the surrogate function,

$$H(\mathbf{\Omega}, \mathbf{C}) := \boldsymbol{y}^\top \mathbf{\Omega}^{-1}\boldsymbol{y} + \sum_{i=0}^{m} \gamma_i \operatorname{tr}(\mathbf{C}^{-1}\mathbf{V}_i) + \ln\det(\mathbf{C}) - n \ge G(\mathbf{\Omega}). \tag{9}$$

The surrogate function H has $\mathbf{C}$ as an extra variable in our optimization, which we set to be of the form:

$$\mathbf{C} = \sum_{i=0}^{m} \widetilde{\gamma}_i \mathbf{V}_i,$$

where $\widetilde{\gamma} = (\widetilde{\gamma}_0, \widetilde{\gamma}_1, \dots, \widetilde{\gamma}_m)$ are latent parameters. Similar to the MM algorithmic recipe [9], we then jointly minimize the surrogate function H with respect to both γ and $\widetilde{\gamma}$ using CD.

The most apparent way of selecting our coordinates is to cyclically update in the order:

$$\gamma_0 \to \gamma_1 \to \dots \to \gamma_m \to \widetilde{\gamma},$$

where the last update is a block update of the entire block $\widetilde{\gamma}$. In other words, the expanded parameters $\widetilde{\gamma}$ are treated as a block of coordinates that is updated in each cycle by searching through the coordinate hyper-plane rather than the single-coordinate directions. We refer to a full completion of updates in a single ordering as a "cycle" of updates. Suppose the initial guess for the parameters are $(\boldsymbol{\gamma}^{(0)}, \widetilde{\boldsymbol{\gamma}}^{(0)})$, then, at the end of cycle t, we denote the updated parameters as $(\boldsymbol{\gamma}^{(t)}, \widetilde{\boldsymbol{\gamma}}^{(t)})$. In Theorem 1, we state that under certain conditions, the sequence $\{(\boldsymbol{\gamma}^{(t)}, \widetilde{\boldsymbol{\gamma}}^{(t)})\}_{t\ge 0}$ generated by PX-CD has limit-points, which are coordinate-wise minima for G.

Let $\mathbf{\Omega}^{(t)} = \sum_{i=0}^{m} \gamma_i^{(t)} \mathbf{V}_i$ be the updated covariance matrix after the $m+1$ original parameters have been updated in cycle t. Then, as the inequality in (9) achieves equality if and only if $\mathbf{C} = \mathbf{\Omega}$ the update for the expanded block parameters $\widetilde{\gamma}$ in cycle t is,

$$\widetilde{\gamma}^{(t)} = \arg\min_{\widetilde{\gamma}} H(\mathbf{\Omega}^{(t)}, \sum_{i=0}^{m} \widetilde{\gamma}_i \mathbf{V}_i) = \gamma^{(t)}.$$

In practice, we simply store $\mathbf{C}^{(t)} = \mathbf{\Omega}^{(t)}$ at the end of each cycle.

Minimizing H with respect to the k-th component of the original parameter γ yields a function of the form:

$$h_k(x) := H(\mathbf{\Omega}_{-k} + x\mathbf{V}_k, \mathbf{C}) = \boldsymbol{y}^\top(\mathbf{\Omega}_{-k} + x\mathbf{V}_k)^{-1}\boldsymbol{y} + x\operatorname{tr}\Big(\mathbf{C}^{-1}\mathbf{V}_k\Big) + \text{const.}, \quad x \ge 0. \tag{10}$$

One of the main advantages of the PX-CD procedure over the basic coordinate descent in Algorithm 1 is that the optimization along each coordinate has a unique minimum.

Lemma 1. *$h_k(x)$ has a unique minimizer for $x \geq 0$.*

Proof. We now show that on $[0, \infty)$, the function $h_k(x)$ is either strictly convex or a linear function with a strictly positive gradient.

We first consider the case where $\boldsymbol{\Omega}_{-k}$ is invertible. From [13], we have the GEV decomposition: $(\mathbf{V}_k, \boldsymbol{\Omega}_{-k}) \to (\mathbf{D}_k, \mathbf{U}_k)$ where $\mathbf{D}_k \in \mathbb{R}^{n \times n}$ is a diagonal matrix with non-negative entries and $\mathbf{U}_k \in \mathbb{R}^{n \times n}$ is invertible. In a similar fashion to the simplified basic CD expression (7), let $\boldsymbol{\alpha}_k = \mathbf{U}_k^\top \boldsymbol{y}$ and $\boldsymbol{d}_k = \operatorname{diag}(\mathbf{D}_k)$. Then, $h_k(x)$ can be simplified to,

$$h_k(x) = \sum_{j=1}^{n} \frac{\alpha_{k,j}^2}{d_{k,j} x + 1} + x \operatorname{tr}\left(\mathbf{C}^{-1} \mathbf{V}_k\right) + \text{const}. \tag{11}$$

We then obtain the first and second derivatives,

$$h_k'(x) = -\sum_{j=1}^{n} \frac{\alpha_{k,j}^2 d_{k,j}}{(d_{k,j} x + 1)^2} + \operatorname{tr}\left(\mathbf{C}^{-1} \mathbf{V}_k\right), \qquad h_k''(x) = \sum_{j=1}^{n} \frac{2\alpha_{k,j}^2 d_{k,j}^2}{(d_{k,j} x + 1)^3}. \tag{12}$$

where $d_j \geq 0$. If there exists j such that $\alpha_{k,j} d_{k,j} \neq 0$, then $h''(x) > 0$ for $x \geq 0$. Then, h is strictly convex and attains a unique global minimizer $x^* \in [0, \infty)$. Suppose that $\alpha_{k,j} d_{k,j} = 0$ for $j = 1, \ldots, n$ then $h_k'(x) = \operatorname{tr}(\mathbf{C}^{-1} \mathbf{V}_k)$. If we can show that $\operatorname{tr}(\mathbf{C}^{-1} \mathbf{V}_k) > 0$, then $h_k(x)$ is strictly increasing on $[0, \infty)$, and $x^* = 0$ is the unique global minimizer for $x \in [0, \infty)$.

We note that the matrix $\mathbf{C}^{-1}$ is positive-definite since $\mathbf{C} = \sum_{i=0}^{m} \tilde{\gamma}_i \mathbf{V}_i$ is invertible and positive semi-definite. Therefore, the symmetric square root factorization, $\mathbf{C}^{-1} = \mathbf{C}^{-1/2} \mathbf{C}^{-1/2}$ exists, and

$$\operatorname{tr}\left(\mathbf{C}^{-1} \mathbf{V}_k\right) = \operatorname{tr}\left(\mathbf{C}^{-1/2} \mathbf{V}_k \mathbf{C}^{-1/2}\right),$$

due to the invariant cyclic nature of the trace. The matrix $\mathbf{C}^{-\frac{1}{2}} \mathbf{V}_k \mathbf{C}^{-\frac{1}{2}}$ is positive semi-definite as $\boldsymbol{z}^\top \mathbf{C}^{-1/2} \mathbf{V}_k \mathbf{C}^{-1/2} \boldsymbol{z} = \|\mathbf{V}_k^{1/2} \mathbf{C}^{-1/2} \boldsymbol{z}\|_2^2 \geq 0$ for all $\boldsymbol{z} \in \mathbb{R}^n$. Since $\mathbf{C}^{-\frac{1}{2}} \mathbf{V}_k \mathbf{C}^{-\frac{1}{2}}$ is a non-zero matrix and positive-semi-definite, $\operatorname{tr}(\mathbf{C}^{-1} \mathbf{V}_k) > 0$ and $h_k(x)$ has a strictly positive slope, which implies that $x^* = 0$ is the unique global minimizer for $x \in [0, \infty)$.

Consider the case $\dim(\operatorname{span}\{\mathbf{V}_1, \ldots, \mathbf{V}_m\}) = r < n$. Assuming $\gamma_0 > 0$, then $\boldsymbol{\Omega}_{-k}$ will be invertible except when $k = 0$. When $\boldsymbol{\Omega}_{-0}$ is singular, a simplified expression in the form of (11) may be difficult to find. Instead, we take the singular value decomposition (SVD) of the symmetric matrix $\boldsymbol{\Omega}_{-0}$,

$$\boldsymbol{\Omega}_{-0} = \mathbf{Q}^\top \boldsymbol{\Lambda} \mathbf{Q}, \quad \boldsymbol{\Lambda} = \begin{bmatrix} \operatorname{diag}(\lambda_1, \ldots, \lambda_r) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix},$$

where $\lambda_1, \ldots, \lambda_r > 0$ are the real-positive eigenvalues of $\boldsymbol{\Omega}_{-k}$, and the matrix $\mathbf{Q} \in \mathbb{R}^{n \times n}$ is orthogonal. Then, we can express the inverse as

$$(\boldsymbol{\Omega}_{-0} + x\mathbf{I})^{-1} = \mathbf{Q}(\boldsymbol{\Lambda} + x\mathbf{I}_n)^{-1} \mathbf{Q}^\top.$$

If we assume $\boldsymbol{y} \notin \operatorname{span}\{\mathbf{V}_1, \ldots, \mathbf{V}_m\}$, then $\boldsymbol{\alpha} = \mathbf{Q}^\top \boldsymbol{y} \neq \mathbf{0}$. Then,

$$h_0(x) = \sum_{j=1}^{r} \frac{\alpha_j^2}{\lambda_j + x} + \sum_{j=r+1}^{n} \alpha_j^2 x^{-1} + x \operatorname{tr}\left(\mathbf{C}^{-1} \mathbf{V}_k\right) + \text{const}. \tag{13}$$

and

$$h_0''(x) = \sum_{j=1}^{r} \frac{2\alpha_j^2}{(\lambda_j + x)^3} + \sum_{j=r+1}^{n} \frac{2\alpha_j^2}{x^3} > 0,$$

when $x > 0$. Therefore, $h_0(x)$ still attains a unique minimizer as the function is strongly convex for $x > 0$. □

The result of this lemma ensures the existence and uniqueness condition (8), and thus we can ensure that accumulation or limit points of the CD iteration are also coordinate-wise minimal points. The details of the optimization follow.

3.1. Univariate Minimization via Newton's Method

Unlike the basic CD Algorithm 1, for which each coordinate update costs $O(n^3)$, here we show that a coordinate update for the PX-CD algorithm costs only $O(jn^2)$ for some constant j where typically $j \ll n$.

The function $h_k(x)$ can be minimized via the second-order Newton's method, which numerically finds the root of $h_k'(x)$. The basic algorithm starts with an initial guess x_0 of the root, and then

$$x_{n+1} = x_n - h_k'(x_n)[h_k''(x_n)]^{-1}, \tag{14}$$

are successive better approximations. The algorithm can be terminated once successive iterates are sufficiently close together. The first and second derivatives of h are given as

$$h_k'(x) = -\boldsymbol{y}^\top(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)^{-1}\mathbf{V}_k(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)^{-1}\boldsymbol{y} + \operatorname{tr}\left(\mathbf{C}^{-1}\mathbf{V}_k\right), \tag{15}$$

$$h_k''(x) = 2\boldsymbol{y}^\top(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)^{-1}\mathbf{V}_k(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)^{-1}\mathbf{V}_k(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)^{-1}\boldsymbol{y}, \tag{16}$$

where we used differentiation of a matrix inverse, which implies that

$$\frac{\partial(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)^{-1}}{\partial x} = -(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)^{-1}\frac{\partial(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)}{\partial x}(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)^{-1}.$$

Similar to the basic CD implementation, computing the algebraic expression in (11) via GEV is expensive. Evaluating (15) and (16) by explicitly calculating $(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)^{-1}$ is also expensive for large n and is of time complexity $O(n^3)$. Instead, we utilize the conjugate gradient (CG) algorithm [13] to efficiently solve linear systems. At each iteration of Newton's method, we approximately solve,

$$(\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)\boldsymbol{b} = \boldsymbol{y}, \qquad (\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)\boldsymbol{c} = \mathbf{V}_k\boldsymbol{b}, \tag{17}$$

and store the solution in $\boldsymbol{b}$ and $\boldsymbol{c}$, respectively, via CG algorithm. Generally, $\|\boldsymbol{b} - (\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)^{-1}\boldsymbol{y}\|$ and $\|\mathbf{c} - (\boldsymbol{\Omega}_{-k} + x\mathbf{V}_k)^{-1}\mathbf{V}_k\boldsymbol{b}\|$ can be made small with $l \ll n$ iterations, where each iteration requires a matrix-vector-multiplication operation with a $n \times n$ matrix. The CG algorithm has complexity $O(ln^2)$ and can be easily implemented with standard Linear Algebra packages. With the stored approximate solutions, we evaluate the first and second derivatives as

$$h_k'(x) = -\boldsymbol{b}^\top\mathbf{V}_k\boldsymbol{b} + \operatorname{tr}\left(\mathbf{C}^{-1}\mathbf{V}_k\right), \qquad h_k''(x) = 2\boldsymbol{b}^\top\mathbf{V}_k\boldsymbol{c}. \tag{18}$$

Before initiating Newton's method, we can check if k is in the active constraint set $\{k : \hat{\gamma}_k = 0\}$. Following from Lemma 1, if $h_k'(0) \geq 0$, then $h_k(x)$ is non-decreasing on $[0, \infty)$. Then, $h_k(0)$ is the global minimum for $x \in [0, \infty)$, and we let $\gamma_i^{(t+1)} = 0$ if we are in cycle $t+1$ of PX-CD. If $h_k'(0) < 0$, we initiate Newton's method at the current value of the variance component, $x_0 = \gamma_k^{(t)}$. If $\dim(\operatorname{span}\{\mathbf{V}_1, \ldots, \mathbf{V}_m\}) < n$, we require $\gamma_0 > 0$ so that $\boldsymbol{\Omega}$ is invertible.

In this case, $k = 0$ cannot be in the active constraint set and we immediately initiate Newton's method at the starting point $x_0 = \gamma_0^{(t)}$. In rare cases $h_k'(x)$ is sufficiently flat at x_n and (14) may significantly overstep the location of the minimizer and return an approximation $x_{n+1} < 0$. In this case, we dampen the step size until $x_{n+1} > 0$.

3.2. Updating Regime

We now consider an alternative to the cyclic ordering of updates. Suppose we update the block $\tilde{\gamma}$ after every co-ordinate update—that is, the updating order of one complete cycle is

$$\gamma_0 \to \tilde{\gamma} \to \gamma_1 \to \tilde{\gamma} \to \ldots \to \gamma_m \to \tilde{\gamma}.$$

This ordering regime satisfies the "essentially cyclic" condition whereby, in every stretch of $2(m+1)$ updates, each component is updated at least once. We refer to CD with this ordering as parameter expanded-immediate coordinate descent (PXI-CD).

In practice, this ordering implies updating the matrix $\mathbf{C}$ after every update made to each γ_k. Since the expression for $h'_k(x)$ requires one to evaluate $\operatorname{tr}(\mathbf{C}^{-1}\mathbf{V}_k)$ and $\mathbf{C}$ is updated after every coordinate, we must re-compute $\mathbf{C}^{-1}$ for each k. This implies that each cycle in PXI-CD will be more expensive than PX-CD. However, we observe that in situations where $\mathbf{V}_i$ is full rank and n is sufficiently large, the number of cycles needed to converge is significantly less than that required for PX-CD and basic CD.

This results in PXI-CD being the most time-efficient algorithm when the scale of the problem is large. In Section 3.3, we show that, when $\mathbf{V}_i$ is low-rank, re-computing $\operatorname{tr}(\mathbf{C}^{-1}\mathbf{V}_k)$ comes at no additional-cost through the use of the Woodbury matrix identity. However, in this particular scenario, where $\mathbf{V}_k$ are low-rank, the performance gain from PXI-CD is not as significant as when $\mathbf{V}_i$ are full rank, and both PXI-CD and PX-CD show similar performance. Algorithm 2 summarizes both PX-CD and PXI-CD methods to obtain $\hat{\gamma}$.

Algorithm 2 PX-CD and PXI-CD for $G(\mathbf{\Omega})$

```
input: $y$ and $[\mathbf{V}_0, \mathbf{V}_1, \ldots, \mathbf{V}_m]$ where $\mathbf{V}_0 = \mathbf{I}_n$
Choose $\gamma \in \{\mathbf{u} \in \mathbb{R}^{m+1} \mid u_i > 0\}$
$\mathbf{\Omega} \leftarrow \gamma_0 \mathbf{I} + \sum_{i=1}^m \gamma_i \mathbf{V}_i$
repeat
    $\mathbf{C} \leftarrow \mathbf{\Omega}$
    for $k = 0, 1, \ldots, m$ do
        $\mathrm{T} \leftarrow \operatorname{tr}(\mathbf{C}^{-1}\mathbf{V}_k)$
        $\mathbf{\Omega}_{-k} \leftarrow \mathbf{\Omega} - \gamma_k \mathbf{V}_k$
        $\gamma_k \leftarrow \arg\min_{x \geq 0} \left[ y^\top (\mathbf{\Omega}_{-k} + x\mathbf{V}_k)^{-1} y + x\mathrm{T} \right]$ via Newton's Method
        $\mathbf{\Omega} \leftarrow \mathbf{\Omega}_{-k} + \gamma_k \mathbf{V}_k$
        if PXI-CD then
            $\mathbf{C} \leftarrow \mathbf{\Omega}$
until termination test satisfied
return $\gamma, \mathbf{\Omega}$
```

As mentioned previously, the novel parameter-expanded coordinate-descent algorithms, PX-CD and PXI-CD, are both amenable to standard convergence analysis.

Theorem 1 (PX-CD and PXI-CD Limit Points)**.** *For both PX-CD and PXI-CD in Algorithm 2, let $\{\gamma^{(t)}, \tilde{\gamma}^{(t)}\}_{t \geq 0}$ be the coordinate-descent sequence. Then, either $G(\sum_k \gamma_k^{(t)} \mathbf{V}_k) \to -\infty$, or every limit-point of $\{\gamma^{(t)}\}_{t \geq 0}$ is a coordinate-wise minimum of (3). If we further assume that $y \notin \operatorname{span}\{\mathbf{V}_1, \ldots, \mathbf{V}_m\} < n$, then the sequence $\{\gamma^{(t)}\}_{t \geq 0}$ is bounded and $G(\sum_k \gamma_k^{(t)} \mathbf{V}_k) \to -\infty$ is ruled out.*

Proof. Recall that $\mathbf{\Omega} = \sum_{i=0}^m \gamma_i \mathbf{V}_i$ and $\mathbf{C} = \sum_{i=0}^m \tilde{\gamma}_i \mathbf{V}_i$. Denote $(x_1, \ldots, x_{m+1}) := \gamma$ and $x_{m+2} := \tilde{\gamma} \in \mathbb{R}^{m+1}$ as well as $x := (x_1, \ldots, x_{m+1}, x_{m+2}) \in \mathbb{R}^{2(m+1)}$. We can rewrite the optimization problem (3) in the penalized form:

$$f(\boldsymbol{x}) := f_0(\boldsymbol{x}) + \sum_{k=1}^{m+1} f_k(x_k) + f_{m+2}(\boldsymbol{x}_{m+2}),$$

where $f_0(\boldsymbol{x}) := H(\boldsymbol{\Omega}, \mathbf{C})$ and

$$f_k(x) := \begin{cases} 0, & x \geq 0 \\ \infty, & x < 0 \end{cases}, \quad k = 1, \ldots, m+1, \quad f_{m+2}(\boldsymbol{x}) := \begin{cases} 0, & \boldsymbol{x} \in [\mathbf{0}, \boldsymbol{\infty}] \\ \infty, & \boldsymbol{x} \notin [\mathbf{0}, \boldsymbol{\infty}) \end{cases}.$$

We can then apply the results in [18], which state that every limit point of $\{\gamma^{(t)}\}_{t \geq 0}$ is a coordinate-wise minimum provided that:

1. Each function $x_k \mapsto f(\boldsymbol{x})$, $k = 1, \ldots, m+1$ and $\boldsymbol{x}_{m+2} \mapsto f(\boldsymbol{x})$ has a unique minimum. For $k = 1, \ldots, m+1$ this has already been verified in Lemma 1. For $\boldsymbol{x}_{m+2} \mapsto f(\boldsymbol{x})$, we simply recall that $H(\boldsymbol{\Omega}, \mathbf{C}) \geq G(\boldsymbol{\Omega})$ with equality achieved if and only if $\mathbf{C} = \boldsymbol{\Omega}$, or equivalently $\boldsymbol{x}_{m+2} = (x_1, \ldots, x_{m+1})$.
2. Each $f_k, k = 1, \ldots, m+2$ is lower semi-continuous. This is clearly true for $k = 1, \ldots, m+1$ because, at the point of discontinuity, we have $\liminf_{x \to 0} f_k(x) \geq f_k(0) = 0$. For f_{m+2}, we simply check that $\{\boldsymbol{x} : f_{m+2}(\boldsymbol{x}) \leq c\}$ for a given $c \in \mathbb{R}$ is a closed set.
3. The domain of f_0 is a Cartesian product and f_0 is continuous on its domain. Clearly, the domain is the $2(m+1)$ Cartesian product $\mathcal{D} := [0, \infty) \times \cdots \times [0, \infty)$ and f_0 is continuous on its effective domain $\{\boldsymbol{x} \in \mathcal{D} : f(\boldsymbol{x}) < \infty\}$.
4. The updating rule is *essentially cyclic*—that is, there exists a constant $T \geq m+2$ such that every block in $(x_1, \ldots, x_{m+1}, \boldsymbol{x}_{m+2})$ is updated at least once between the r-th iteration and the $(r+T-1)$-th iteration for all r. In our case, each block is updated at least once in one iteration of Algorithm 2 so that we satisfy the essentially cyclic condition. In the PXI-CD, we actually update $(m+1)$ times the block $\boldsymbol{x}_{m+2}$.

Thus, we can conclude by Proposition 5.1 in [18] that either $f_0(\gamma^{(t)}) \to -\infty$ or the limit points of $\{\gamma^{(t)}\}_{t \geq 0}$ are coordinate-wise minima of $H(\boldsymbol{\Omega}, \mathbf{C})$.

If we further assume that $\boldsymbol{y} \notin \text{span}\{\mathbf{V}_1, \ldots, \mathbf{V}_m\} < n$, then we can show that set $\{\gamma \geq \mathbf{0} : f_0(\gamma) \leq f_0(\gamma^{(0)})\}$ is compact, which ensures that the sequence $\{\gamma^{(t)}\}_{t \geq 0}$ is bounded and rules out the possibility that $\lim_t f_0(\gamma^{(t)}) = -\infty$. To see that this is the case, note that $G(\boldsymbol{\Omega})$ provides a lower bound to $H(\boldsymbol{\Omega}, \mathbf{C})$, and this is sufficient to show that $\{\gamma \geq \mathbf{0} : G(\sum_k \gamma_k \mathbf{V}_k) \leq c\}$ is compact for any $c \in \mathbb{R}$ under the assumption that $\mathbf{V}_0 := \mathbf{I}$ and $\boldsymbol{y} \notin \text{span}\{\mathbf{V}_1, \ldots, \mathbf{V}_m\} < n$. However, these are precisely the conditions of Lemma 3 in [9], which ensure that $\{\gamma \geq \mathbf{0} : L(\gamma) \geq c\}$ is compact for a likelihood $L(\gamma) := -G(\sum_k \gamma_k \mathbf{V}_k)$.

Finally, note that since we update the entire block $\tilde{\gamma}$ simultaneously (in both PX-CD and PXI-CD), this means that a coordinate-wise minimum of $H(\boldsymbol{\Omega}, \mathbf{C})$ is also a coordinate-wise minimum for $G(\boldsymbol{\Omega})$. □

We again emphasize that, as with all the competitor methods, the theorem does not guarantee convergence of the coordinate-descent sequence $\{\gamma^{(t)}\}_{t \geq 0}$ or that the convergence will be to a local minimum. The only thing we can say for sure is that, when the sequence converges, then the limit will be a coordinate-wise minimum (which could be a saddle point in some special cases). Nevertheless, our numerical experience in Section 4 is that the sequence always converges to a coordinate-wise minimum and that the coordinate-wise minimum is in fact a (local) minimum.

3.3. Linear Mixed Model Implementation

We now show that, for Linear Mixed Models (LMM), we can reduce the computational complexity of each sub-problem to $O(jd^2)$, for some constants $j \ll n$ and $d < n$. As shown in Section 3.1, solving each univariate sub-problem can be simplified to implementing Newton's method, where at each Newton's update, we solve two n-dimensional linear systems.

In settings where $\text{rank}(\mathbf{V}i) < n$, for $i = 1, \ldots, m$, we are able to reduce the dimensions of the linear system that is required to be solved. To see this, let us first specify the general variance-component model (also known as the general mixed ANOVA model) [19]. Suppose,

$$\boldsymbol{y} = \mathbf{X}\beta + \mathbf{Z}_1\mathbf{b}_1 + \cdots + \mathbf{Z}_m\mathbf{b}_m + \boldsymbol{\epsilon}, \tag{19}$$

where $\mathbf{X}$ is an $n \times p$ matrix of known fixed numbers, $p \leq n$; $\boldsymbol{\beta}$ is an $p \times 1$ vector of unknown constants; $\mathbf{Z}_i$ is a full-rank $n \times c_i$ matrix of known fixed numbers, $c_i \leq n$; $\mathbf{b}_i$ is an $c_i \times 1$ vector of independent variables from $\mathcal{N}(0, \gamma_i)$, which are unknown and $\boldsymbol{\epsilon}$ is an $n \times 1$ vector of independent errors from $\mathcal{N}(0, \gamma_0)$ that are unknown. In this setup, $\gamma_0, \ldots, \gamma_m$, are the variance component parameters that are to be estimated, $\mathbf{V}_i = \mathbf{Z}_i\mathbf{Z}_i^\top$ and $\boldsymbol{\Omega} = \gamma_0\mathbf{I} + \mathbf{Z}\boldsymbol{\Sigma}\mathbf{Z}^\top$, where

$$\mathbf{Z} = \begin{bmatrix} | & | & & | \\ \mathbf{Z}_1 & \mathbf{Z}_2 & \cdots & \mathbf{Z}_m \\ | & | & & | \end{bmatrix}, \quad \boldsymbol{\Sigma} = \textbf{block diag}(\gamma_1\mathbf{I}_{c_1}, \ldots, \gamma_m\mathbf{I}_{c_m}).$$

We now provide two methods that take advantage of $\mathbf{V}_i$ being low-rank. Let,

$$c := \sum_{i=1}^{m} \text{rank}(\mathbf{V}_i) = \sum_{i=1}^{m} c_i.$$

In the first method, we utilize a QR factorization that reduces the computational complexity when $c < n$, i.e., the column rank of $\mathbf{Z}$ is less than n. In the second method, we use the Woodbury matrix identity to reduce the complexity when $c_i < n$ for $i = 1, \ldots, m$, i.e., the column rank of each of the matrices $\mathbf{Z}_i$ is less than n.

3.3.1. QR Method

The following QR factorization can be viewed as a data pre-processing step that allows all PX-CD and PXI-CD computations to be c-dimensional instead of n-dimensional. The QR factorization only needs to be computed once initially with a cost of $O(cn^2)$ operations. Let the QR factorization of $\mathbf{Z} \in \mathbb{R}^{n \times c}$ be

$$\mathbf{Z} = \overbrace{\begin{bmatrix}\mathbf{Q}_{[c]} & \mathbf{Q}_{[n-c]}\end{bmatrix}}^{\mathbf{Q}} \begin{bmatrix}\mathbf{R} \\ \mathbf{0}\end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} | & | & & | \\ \mathbf{R}_1 & \mathbf{R}_2 & \cdots & \mathbf{R}_m \\ | & | & & | \end{bmatrix},$$

where $\mathbf{R}$ is an $c \times c$ upper triangular matrix, $\mathbf{0}$ is an $(n-c) \times c$ zero matrix, $\mathbf{Q}_{[c]}$ is an $n \times c$ matrix, $\mathbf{Q}_{[n-c]}$ is $n \times (n-c)$, and $\mathbf{Q}_{[c]}$ and $\mathbf{Q}_{[n-c]}$ both have orthogonal columns. The matrix $\mathbf{R}$ is partitioned such that the number of columns in $\mathbf{R}_i$ is equal to the number of columns in $\mathbf{Z}_i$. Let $\widetilde{\boldsymbol{y}} = [\widetilde{\boldsymbol{y}}_{[c]}, \widetilde{\boldsymbol{y}}_{[n-c]}]^\top = \mathbf{Q}^\top\boldsymbol{y}$, where $\widetilde{\boldsymbol{y}}_{[c]}$ are the first c elements of $\widetilde{\boldsymbol{y}}$ and $\widetilde{\boldsymbol{y}}_{[n-c]}$ are the last $n-c$ elements of $\widetilde{\boldsymbol{y}}$. Then,

$$H(\boldsymbol{\Omega}, \mathbf{C}) = \widetilde{\boldsymbol{y}}_{[c]}^\top \widetilde{\boldsymbol{\Omega}}^{-1} \widetilde{\boldsymbol{y}}_{[c]} + \sum_{i=0}^{m} \gamma_i \text{tr}(\widetilde{\mathbf{C}}^{-1}\widetilde{\mathbf{V}}_i) + \ln\det(\mathbf{C}) - n + \alpha,$$

where we define: $\widetilde{\mathbf{V}}_i := \mathbf{R}_i\mathbf{R}_i^\top$, $\widetilde{\mathbf{V}}_0 := \mathbf{I}_c$, $\widetilde{\boldsymbol{\Omega}} := \sum_{i=0}^m \gamma_i\widetilde{\mathbf{V}}_i$, $\widetilde{\mathbf{C}} := \sum_{i=0}^m \widetilde{\gamma}_i\widetilde{\mathbf{V}}_i$ and $\alpha := \gamma_0\widetilde{\gamma_0}^{-1}(n-c) + \gamma_0^{-1}\widetilde{\boldsymbol{y}}_{[n-c]}^\top\widetilde{\boldsymbol{y}}_{[n-c]}$. The details of this derivation are provided in the Appendix A. To implement PX-CD or PXI-CD after this transformation we run Algorithm 2 with inputs $\widetilde{\mathbf{V}}_0, \ldots, \widetilde{\mathbf{V}}_m \in \mathbb{R}^{c \times c}$ and $\widetilde{\boldsymbol{y}}_{[c]} \in \mathbb{R}^{c \times 1}$. In this simplification of H, we have the additional term α, which is dependent on γ_0. Therefore, when we update the parameter γ_0 and implement Newton's method we must also add

$$\frac{\partial\alpha}{\partial\gamma_0} = \widetilde{\gamma_0}^{-1}(n-c) - \gamma_0^{-2}\widetilde{\boldsymbol{y}}_{[n-c]}^\top\widetilde{\boldsymbol{y}}_{[n-c]} \quad \text{and} \quad \frac{\partial^2\alpha}{\partial\gamma_0^2} = 2\gamma_0^{-3}\widetilde{\boldsymbol{y}}_{[n-c]}^\top\widetilde{\boldsymbol{y}}_{[n-c]}$$

to the corresponding derivatives derived for Newton's method in Section 3.1.

3.3.2. Woodbury Matrix Identity

Alternatively, if c is large (say $c > n$) but individually $c_i < n$, for $i = 1, \dots, m$, we can use the Woodbury identity to reduce each linear system to c_k dimensions (instead of n dimensions) when updating the component γ_k. Suppose we are in cycle $t+1$ of either PX-CD or PXI-CD and we wish to update the parameter γ_k, where $k \neq 0$, then we can simplify the optimization,

$$\gamma_k^{(t+1)} = \arg\min_{x \geq 0} h_k(x), \qquad h_k(x) = H(\mathbf{\Omega}_{-k} + x\mathbf{V}_k, \mathbf{C}),$$

by viewing $\mathbf{\Omega}_{-k} + x\mathbf{V}_k = \mathbf{\Omega} + (x - \gamma_k^{(t)})\mathbf{Z}_k\mathbf{Z}_k^\top$ as a low-rank perturbation to the matrix $\mathbf{\Omega}$. The Woodbury identity gives the expression for the inverse,

$$\left[\mathbf{\Omega} + (x-\gamma_k^{(t)})\mathbf{Z}_k\mathbf{Z}_k^\top\right]^{-1} = \mathbf{\Omega}^{-1} - (x-\gamma_k^{(t)})\mathbf{\Omega}^{-1}\mathbf{Z}_k\left[\mathbf{I}_{c_k} + (x-\gamma_k^{(t)})\mathbf{Z}_k^\top\mathbf{\Omega}^{-1}\mathbf{Z}_k\right]^{-1}\mathbf{Z}_k^\top\mathbf{\Omega}^{-1}, \quad (20)$$

which contains the unperturbed inverse matrix $\mathbf{\Omega}^{-1}$ and the inverse of a smaller $c_k \times c_k$ matrix. In this implementation of PXI-CD and PX-CD we re-compute and store the matrix $\mathbf{\Omega}^{-1}$ after each coordinate update. Let $\boldsymbol{w} := \mathbf{Z}_k^\top\mathbf{\Omega}^{-1}\boldsymbol{y}$, then the line search along the k-th component ($k \neq 0$) of the function H simplifies to

$$h_k(x) = -(x-\gamma_k^{(t)})\boldsymbol{w}^\top\left[\mathbf{I}_{c_k} + (x-\gamma_k^{(t)})\mathbf{Z}_k^\top\mathbf{\Omega}^{-1}\mathbf{Z}_k\right]^{-1}\boldsymbol{w} + x\,\mathrm{tr}\left(\mathbf{C}^{-1}\mathbf{Z}_k\mathbf{Z}_k^\top\right) + \text{const.}$$

When implementing PXI-CD, there is no additional cost when using the Woodbury identity, as the update $\mathbf{C} \leftarrow \mathbf{\Omega}$ is made after every coordinate update and the trace term in the line search can be evaluated cheaply because $\mathbf{\Omega}^{-1}$ is known. We can now implement Newton's method to find the minimum of $h_k(x)$. Let

$$\mathbf{B} := \mathbf{Z}_k^\top\mathbf{\Omega}^{-1}\mathbf{Z}_k, \qquad \mathbf{M} := \mathbf{I}_{c_k} + (x-\gamma_k^{(t)})\mathbf{B}.$$

If we then solve the c_k-dimensional linear systems $\mathbf{M}\boldsymbol{d} = \boldsymbol{w}$ and $\mathbf{M}\boldsymbol{f} = \mathbf{B}\boldsymbol{d}$ with CG and store the solution in the vectors $\boldsymbol{d}$ and $\boldsymbol{f}$, respectively, we can evaluate the first and second derivatives of $h_k(x)$ as

$$h_k'(x) = -\boldsymbol{w}^\top\boldsymbol{d} + (x-\gamma_k^{(t)})\boldsymbol{d}^\top\mathbf{B}\boldsymbol{d} + \mathrm{tr}(\mathbf{B}), \qquad h_k''(x) = 2\boldsymbol{d}^\top\mathbf{B}\boldsymbol{d} + 2(x-\gamma_k^{(t)})\boldsymbol{d}^\top\mathbf{B}\boldsymbol{f}, \quad (21)$$

and implement the Newton steps (14). The derivation of these derivative expressions are provided in Appendix B. After each coordinate k is updated we evaluate and store the updated inverse covariance matrix,

$$\mathbf{\Omega}^{-1} \leftarrow \left[\mathbf{\Omega} + (\gamma_k^{(t+1)} - \gamma_k^{(t)})\mathbf{Z}_k\mathbf{Z}_k^\top\right]^{-1},$$

using (20), where we invert a smaller $c_k \times c_k$ matrix only. When $k = 0$, no reduction in complexity can be made as the perturbation to $\mathbf{\Omega}$ is full-rank and we update γ_0 as we did in Section 3.1. If $c = \sum_{i=1}^m c_i < n$, then we can use an alternate form of the Woodbury identity,

$$\mathbf{\Omega}^{-1} = \gamma_0^{-1}\left[\mathbf{I} - \mathbf{W}(\gamma_0\mathbf{I} + \mathbf{W}^\top\mathbf{W})^{-1}\mathbf{W}^\top\right],$$

where $\mathbf{W} := \mathbf{Z}\mathbf{\Sigma}^{1/2}$ to update $\mathbf{\Omega}^{-1}$ after γ_0 has been updated. If $c > n$, we invert the full $n \times n$ updated matrix covariance to obtain $\mathbf{\Omega}^{-1}$ using a Cholesky factorization at cost $O(n^3)$. This $O(n^3)$ cost for updating γ_0 is a disadvantage for this implementation if $c > n$; however, numerical simulations suggest that, when $c_i \ll n$ for $i = 1, \dots, m$, the Woodbury implementation is the fastest implementation.

3.4. Variable Selection

When the number of variance components is large, performing variable selection can enhance the model interpretation and provide more stable parameter estimation. To impose sparsity when estimating γ, a lasso or ridge penalty can be added to the negative log-likelihood [20]. The MM implementation [9] provides modifications to the MM algorithm such that both lasso and ridge penalized expressions can be minimized. We now show that, with PX-CD, we can minimize the penalized negative log-likelihood when using the $\|\cdot\|_1$ penalty. Consider the penalized negative log-likelihood expression,

$$G(\mathbf{\Omega}) + \lambda \sum_{i=0}^{m} \gamma_i, \quad \gamma_i \geq 0,\, \lambda > 0.$$

We then have the surrogate function,

$$J(\mathbf{\Omega}, \mathbf{C}) := H(\mathbf{\Omega}, \mathbf{C}) + \lambda \sum_{i=0}^{m} \gamma_i.$$

If we use PX-CD to minimize J, we need to repeatedly minimize the one-dimensional function along each co-ordinate, $j_k(x) := h_k(x) + \lambda x + \lambda \sum_{i \neq k}^{m} \gamma_i$. Here, we implement Newton's method as before with the only difference now being that the derivative is increased by the constant λ,

$$j_k'(x) := h_k'(x) + \lambda.$$

It follows from Lemma 1 that $j(x)$ is either strictly convex or linear with strict positive gradient for $x \in [0, \infty)$. We check if $h_k'(0) + \lambda \geq 0$ to determine if $j_k(0)$ is the global minimizer for $x \in [0, \infty)$. If it is, we let $\gamma_i^{(t+1)} = 0$ if we are in cycle $t+1$. If $j_k'(0) < 0$, we initiate Newton's method at the current value of the variance component $x_0 = \gamma_k^{(t)}$. The larger the parameter λ is, the greater number of times this active constraint condition will be met, and therefore more variance components will be set to zero.

4. Numerical Results

In this section, we assess the efficiency of PX-CD and PXI-CD via simulation and compare them against the best current alternative method, the MM algorithm [9]. In [9] the MM algorithm is found to be superior to both the EM and Fisher Scoring Method in terms of performance. This superior performance was also described in [21].

In our experiments, we additionally include the Expectation–Maximization (EM) and Fisher-Scoring (FS) method in the small-scale problem only, where $\gamma_0 = 0.1$. We exclude the EM and FS method for more difficult problems, as they are too slow and unsuitable. The MM, EM and FS are executed with the Julia implementation in [9]. We provide results in three settings. First, we simulate data from the model (Section 3.3), where $c_i < n$, i.e., when the matrices $\mathbf{V}_i$ are low-rank. Secondly, we simulate when $c_i = n$, i.e., the matrices $\mathbf{V}_i$ are full-rank, and finally, we simulate data from model (Section 3.3), where the matrices $\mathbf{V}_i$ are constructed from a real data-set containing genetic variants of mice.

4.1. Simulations

For the following simulations, we simulate data from the model (Section 3.3). Since the fixed effects $\boldsymbol{\beta}$ can always be eliminated from the model using the REML, we focus solely on the estimation of the variance component parameters. In other words, the value of $\boldsymbol{\beta}$ in our simulations is irrelevant. In each simulation, we generate the fixed matrices $\mathbf{V}_i$ as

$$\mathbf{V}_i = \frac{\sum_{j=1}^{r} \boldsymbol{Z}_{i,j} \boldsymbol{Z}_{i,j}^{\top}}{\left\| \sum_{j=1}^{r} \boldsymbol{Z}_{i,j} \boldsymbol{Z}_{i,j}^{\top} \right\|_F}, \tag{22}$$

where $\boldsymbol{Z}_{i,j} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_n)$ and $\|\cdot\|_F$ is the Frobenius matrix norm. The rank of each $\mathbf{V}_i$ is equal to the parameter r, which we vary.

In each simulation, for $k \neq 0$, we draw the m true variance components as $\gamma_k = (1+\rho)^2$ where $\rho \sim \mathcal{N}(0,1)$. Then, we simulate the response from $\boldsymbol{y} \sim \mathcal{N}(\mathbf{0}, \sum_{i=0}^{m} \gamma_i \mathbf{V}_i)$ and estimate the vector $\hat{\gamma}$. We vary the value of γ_0 from the set $\{0.1, 1, 10\}$ and keep $n = 1000$.

4.1.1. Low-Rank

We now present the results for where $\mathbf{V}_i$ are generated as stated above and $r < n$. As $\mathbf{V}_i$ are low-rank, we run the Woodbury implementation of PXI-CD and exclude PX-CD as it has the same computational cost as PXI-CD for each update and exhibits almost identical performance. First, PXI-CD is run until the relative change $\left[L(\gamma^{(t+1)}) - L(\gamma^{(t)})\right] / \left[\mid L(\gamma^{(t)}) \mid +1\right]$ is less than 10^{-10} and we store the final objective value as L^*. We then run the algorithms MM, EM and FS and terminate once $L(\gamma^{(t)}) > L^*$. We initialize all algorithms to start from the point $\gamma^{(0)} = \mathbf{1}$. Each simulation scenario is replicated 10 times. The mean running time is reported along with the standard error of the mean running time provided in parentheses.

The results of the low-rank simulations are given in Tables 1–3. The results indicate that, apart from the smallest scale problem when $r = m = 20$, our PXI-CD algorithm outperforms the MM algorithm and significantly so that the scale of the model (both m and r) increases.

Table 1. The running times (s) when $\mathbf{V}_i$ are low-rank, $n = 1000$, and $\gamma_0 = 0.1$.

		m			
Method	***r***	**20**	**50**	**75**	**100**
PXI-CD	20	4.20 (0.07)	17.00 (1.18)	25.81 (1.57)	51.8 (5.09)
MM		1.89 (0.22)	42.54 (7.69)	220.06 (61.19)	845.2 (136.89)
EM		23.30 (0.52)	-	-	-
FS		91.80 (0.99)	-	-	-
PXI-CD	50	5.42 (0.16)	21.36 (0.85)	37.46 (2.72)	40.21 (2.97)
MM		95.63 (41.20)	203.24 (53.20)	584.76 (53.20)	1167.22 (114.38)
PXI-CD	100	8.96 (0.25)	30.17 (2.24)	28.26 (0.73)	45.65 (2.56)
MM		112.58 (21.48)	555.38 (82.01)	699.89 (90.55)	1327.54 (53.80)
PXI-CD	150	14.13 (0.63)	33.18 (4.67)	38.24 (1.54)	48.97 (2.08)
MM		122.44 (38.48)	628.24 (55.61)	929.67 (75.87)	1243.97 (84.30)

Table 2. The running times (s) when $\mathbf{V}_i$ are low-rank, $n = 1000$, and $\gamma_0 = 1$.

		m			
Method	***r***	**20**	**50**	**75**	**100**
PXI-CD	20	3.10 (0.08)	9.52 (0.28)	16.30 (0.43)	26.05 (0.77)
MM		3.11 (0.84)	117.40 (24.42)	280.31 (51.43)	719.32 (158.51)
PXI-CD	50	4.08 (0.11)	14.35 (0.76)	29.94 (1.33)	53.10 (3.7)
MM		157.98 (34.53)	501.60 (90.15)	546.56 (91.48)	1133.83 (129.96)
PXI-CD	100	6.00 (0.36)	25.34 (0.95)	61.69 (3.38)	73.23 (10.6)
MM		103.08 (31.4)	544.06 (61.35)	743.79 (104.67)	1254.97 (87.32)
PXI-CD	150	9.18 (0.45)	42.30 (2.08)	66.13 (8.3)	66.67 (10.2)
MM		176.80 (31.64)	498.12 (63.39)	986.87 (61.46)	1110.90 (105.59)

Table 3. The running times (s) when $\mathbf{V}_i$ are low-rank, $n = 1000$, and $\gamma_0 = 10$.

		m			
Method	***r***	**20**	**50**	**75**	**100**
PXI-CD	20	3.62 (0.10)	8.79 (0.20)	15.60 (0.41)	23.13 (0.91)
MM		10.17 (0.84)	318.53 (90.49)	648.64 (132.62)	839.24 (158.52)
PXI-CD	50	4.36 (0.09)	14.39 (0.4)	24.65 (0.97)	42.07 (1.13)
MM		184.03 (38.8)	473.58 (80.82)	648.33 (114.98)	1230.56 (128.79)
PXI-CD	100	6.53 (0.22)	26.07(1.51)	48.48 (2.48)	60.72 (5.33)
MM		124.68 (19.93)	511.66 (65.85)	880.53 (69.53)	1279.40 (81.93)
PXI-CD	150	10.15 (0.37)	35.34 (1.72)	66.94 (2.28)	103.89 (10.36)
MM		199.80 (33.17)	512.35 (49.97)	943.84 (84.24)	1244.32 (72.13)

4.1.2. Full-Rank

We now present the results when $\text{rank}(\mathbf{V}_i) = n = 1000$. We implement the standard PX-CD, PXI-CD and MM algorithms, where either the Woodbury, or the QR method cannot be used. Initially, PX-CD is run until the relative change $\left[L(\gamma^{(t+1)}) - L(\gamma^{(t)})\right] / \left[\mid L(\gamma^{(t)}) \mid +1\right]$ is less than 10^{-10}, and we store the final objective value as L^*. The other algorithms terminate once $L(\gamma^{(t+1)}) > L^*$. For the following simulations, we consider one iteration of a CD algorithm as a single cycle of updates. Each simulation scenario is replicated 10 times. The mean running time and mean iteration number is reported with the standard error of the mean running time and mean iteration provided in parentheses.

The results of the full-rank simulations are given in Tables 4–6. PX-CD and PXI-CD both significantly outperform the MM and the basic CD algorithms in these examples. We observe that as the number of components m increases the problem becomes increasingly difficult for the MM algorithm. An intuitive explanation for this performance gap is that the CD algorithms are able to identify the active constraint set $\{k : \hat{\gamma}_k = 0\}$ in only a few cycles.

When $\gamma_0 = 0.1$ and $\gamma_0 = 1$ and m is large ($m = 50$, $m = 100$), PXI-CD is the fastest algorithm, even though it is computationally the most expensive per cycle. When $\gamma_0 = 10$ and $m = 100$, PXI-CD is the fastest algorithm. In fact, as the problem size grows, the number of iterations that PXI-CD requires to converge is less than that of the basic CD. This simulation indicates that PXI-CD is well-suited to problems with large m, n and when $\mathbf{V}_i$ are full-rank. The basic CD algorithm, while numerically the inferior compared to the PX-CD and PXI-CD algorithms still outperforms the MM algorithm in these simulations.

Table 4. The convergenceresults when $\mathbf{V}_i$ are full-rank, $n = 1000$, and $\gamma_0 = 0.1$.

Method	***m***	**Iterations**	**Time (s)**	**Objective**
PX-CD	25	89.00 (2.90)	37.47 (1.71)	−1383.44
PXI-CD		146.90 (21.17)	97.49 (13.58)	−1383.44
CD		109.40 (17.52)	286.96 (19.83)	−1383.44
MM		3957.90 (300.77)	281.53 (21.03)	−1383.44
PX-CD	50	182.40 (11.34)	147.21 (10.64)	−1831.40
PXI-CD		73.10 (2.97)	104.42 (4.64)	−1831.40
CD		103.40 (6.18)	852.41 (70.22)	−1831.40
MM		10,240.40 (2140.25)	1557.79 (343.47)	−1831.40
PX-CD	100	279.10 (15.09)	376.18 (24.14)	−2143.93
PXI-CD		80.70 (2.97)	211.19 (8.58)	−2143.93
CD		164.00 (8.84)	2060.69 (155.37)	−2143.93
MM		12,171.90 (1526.30)	3482.30 (465.51)	−2143.93

Table 5. The convergence results when $\mathbf{V}_i$ are full-rank, $n = 1000$, and $\gamma_0 = 1$.

Method	*m*	Iterations	Time (s)	Objective
PX-CD	25	82.60 (4.05)	32.00 (1.93)	−1707.53
PXI-CD		172.70 (7.08)	112.31 (5.15)	−1707.53
CD		116.90 (6.31)	279.61 (18.03)	−1707.53
MM		4313.90 (347.85)	303.13(24.3)	−1707.53
PX-CD	50	192.50 (8.89)	155.03 (8.11)	−1957.26
PXI-CD		103.00 (18.85)	147.79 (26.11)	−1957.26
CD		110.20 (4.53)	940.96 (58.0)	−1957.26
MM		15,860.80 (2872.82)	2488.70 (484.24)	−1957.26
PX-CD	100	313.60 (13.88)	423.78 (20.48)	−2203.61
PXI-CD		86.80 (3.06)	226.10 (8.51)	−2203.61
CD		185.60 (8.33)	2422.93 (143.76)	−2203.61
MM		12,820.60 (2102.69)	3792.45 (725.63)	−2203.61

Table 6. The convergence results when $\mathbf{V}_i$ are full-rank, $n = 1000$, and $\gamma_0 = 10$.

Method	*m*	Iterations	Time (s)	Objective
PX-CD	25	75.20 (2.88)	25.74 (1.46)	−2603.51
PXI-CD		152.00 (6.98)	95.59 (5.12)	−2603.51
CD		38.50 (1.60)	172.03 (11.70)	−2603.51
MM		3616.90 (513.57)	254.87 (36.27)	−2603.51
PX-CD	50	143.70 (6.93)	108.55 (5.89)	−2668.99
PXI-CD		177.50 (28.72)	249.73 (39.38)	−2668.99
CD		79.80 (4.59)	668.65 (40.71)	−2668.99
MM		11,306.30 (1824.79)	1697.89 (275.36)	−2668.99
PX-CD	100	304.00 (12.39)	412.54 (19.01)	−2731.09
PXI-CD		87.40 (2.54)	230.54 (7.27)	−2731.09
CD		181.80 (7.63)	2697.20 (150.64)	−2731.09
MM		11,796.10 (1732.0)	3358.14 (521.92)	−2731.09

4.2. Genetic Data

We now present simulation results when $\mathbf{Z}_i$ are constructed from the https://openmendel.github.io/SnpArrays.jl/latest/#Example-data (accessed on 1 July 2022) mouse single nucleotide polymorphism (SNP) array data set available from the Open Mendel project [21]. The dataset consists of $\mathbf{Z}$, an $n \times c$ matrix consisting of c genetic variants for n individual mice. For this experiment, c = 10,200 and $n = 500$. We artificially generate m different genetic regions by partitioning the columns of $\mathbf{Z}$ into $\mathbf{Z}_{i=1,\dots,m}$ gene matrices, where $\mathbf{Z}_i \in \mathbb{R}^{n \times r}$. Then, we can compose our fixed matrices $\mathbf{V}_1, \dots, \mathbf{V}_m$ as,

$$\mathbf{V}_i = \frac{\mathbf{Z}_i \mathbf{Z}_i^\top}{\left\| \sum_{j=1}^m \mathbf{Z}_j \mathbf{Z}_j^\top \right\|_F}.$$

We simulate γ and $\boldsymbol{y}$ as we did in Section 4.1. In this case, $\boldsymbol{y}$ mimics a vector of quantitative trait measurements of the n mice. This data set is well-suited for testing our method when m is large ($m > n$). In these cases, we observe that, when initialized at the same point, the MM and PXI-CD method converge to different solutions, i.e., they may converge to different stationary points. Therefore, we run all algorithms until the relative change $\left[L(\gamma^{(t+1)}) - L(\gamma^{(t+1)})\right] / \left[\mid L(\gamma^{(t)}) \mid +1\right]$ is less than 10^{-10}. Since $r < n$, we implement PXI-CD utilizing the Woodbury identity. Each simulation scenario is replicated 10 times. The mean running time and mean iteration number are reported with the standard error of the mean running time and mean iteration provided in parentheses.

The results of the genetic study simulation are provided in Table 7. We observe that PXI-CD outperforms the MM algorithm for all values of m and r for this data set in both the number of iterations and running time until convergence. When $m > n$, we observe that the MM and PXI-CD method converge to noticeably different objective values. We suspect that this is because when $m > n$ the likelihood in (2) exhibits many more local minima. On average, PXI-CD converges to a more optimal stationary point when m is large and $m > n$.

Table 7. The running times (s)—mouse data.

Method	m	r	Iterations	Time (s)	Objective
PXI-CD	100	102	95.1 (9.9)	23.1 (2.66)	−43.6
MM			1580.2 (194.76)	108.7 (15.37)	−43.6
PXI-CD	200	51	180.0 (13.41)	58.8 (4.4)	−82.5
MM			2797.8 (401.1)	466.0 (88.95)	−82.6
PXI-CD	500	20	397.8 (53.73)	258.1 (34.56)	−100.3
MM			2700.8 (366.57)	1055.4 (179.97)	−106.6
PXI-CD	1000	10	434.7 (35.7)	507.5 (39.91)	−82.1
MM			3004.8 (343.61)	2329.4 (296.37)	−91.2

5. Conclusions

The MLE solution for variance-component models requires the optimization of a non-convex log-likelihood function. In this paper, we showed that a basic implementation of the cyclic CD algorithm is computationally expensive to run and is not amenable to traditional convergence analysis.

To remedy this, we proposed a novel parameter-expanded CD (PX-CD) algorithm, which is both computationally faster and also subject to theoretical guarantees. PX-CD optimizes a higher-dimensional surrogate function that attains a coordinate-wise minimum with respect to each of the variance component parameters. The extra speed is derived from the fact that required quantities (such as first and second-order derivatives) are evaluated via the conjugate-gradient algorithm.

Additionally, we propose an alternative updating regime called PXI-CD, where the expanded block parameters are updated immediately after each coordinate update. This new updating regime requires more computation for each iteration as compared to PX-CD. However, numerically, we observed that, for large-scale models, where the number of variance components $m + 1$ is large and $\mathbf{V}_i$ are full-rank, the number of iterations needed to converge greatly offsets the additional computational cost per coordinate update cycle.

Our numerical experiments suggest that PX-CD and PXI-CD outperform the best current alternative—the MM algorithm. When the number of variance components m is large, we observed that PXI-CD was significantly faster than the MM algorithm and tended to converge to more optimal stationary points.

A potential extension of this work is to apply parameter-expanded CD algorithms to the multivariate-response variance-component model. Instead of the univariate response, one considers the multivariate response model with a $n \times d$ response matrix $\mathbf{Y}$. In this setup, $\mathbb{E}[\mathbf{Y}] = \mathbf{XB}$ where $\mathbf{B}$ is a $p \times d$ matrix. The $nd \times nd$ covariance matrix is of the form

$$\boldsymbol{\Omega} = \mathrm{cov}(\mathrm{vec}(\mathbf{Y})) = \sum_{i=0}^{m} \boldsymbol{\Gamma}_i \otimes \mathbf{V}_i,$$

where $\boldsymbol{\Gamma}_i$ are unknown $d \times d$ variance components and $\mathbf{V}_i$ are the known $n \times n$ covariance matrices. The challenging aspect of this problem is that each optimization with respect to a parameter $\boldsymbol{\Gamma}_i$ is not univariate but is rather a search over positive semi-definite matrices—itself, a difficult optimization problem.

Author Contributions: Conceptualization, A.M., S.M. and Z.B.; methodology, A.M., S.M. and Z.B.; software, A.M.; formal analysis, A.M., S.M. and Z.B. data curation, A.M.; writing—original draft preparation, A.M.; writing—review and editing, A.M., S.M. and Z.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: https://openmendel.github.io/SnpArrays.jl/latest (accessed on 1 July 2022) mouse single nucleotide polymorphism (SNP) array data set.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CD	Coordinate descent
PX-CD	Parameter expanded coordinate descent
PXI-CD	Parameter expanded-immediate coordinate descent

Appendix A. QR Method

In this section, we provide further details on the QR factorization used in Section 3.3.1. Recall that we have the decomposition,

$$\mathbf{Z} = \overbrace{\begin{bmatrix} \mathbf{Q}_{[c]} & \mathbf{Q}_{[n-c]} \end{bmatrix}}^{\mathbf{Q}} \begin{bmatrix} \mathbf{R} \\ \mathbf{O} \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} | & | & & | \\ \mathbf{R}_1 & \mathbf{R}_2 & \cdots & \mathbf{R}_m \\ | & | & & | \end{bmatrix},$$

where $\mathbf{Q}$ is an orthogonal matrix and $\mathbf{R}$ is partitioned such that the number of columns in $\mathbf{R}_i$ is equal to number of columns in $\mathbf{Z}_i$. Recall that $\widetilde{\boldsymbol{y}} = [\widetilde{\boldsymbol{y}}_{[c]}, \widetilde{\boldsymbol{y}}_{[n-c]}]^\top = \mathbf{Q}^\top \boldsymbol{y}$ where $\widetilde{\boldsymbol{y}}_{[c]}$ are the first c elements of $\widetilde{\boldsymbol{y}}$ and $\widetilde{\boldsymbol{y}}_{[n-c]}$ are the last $n-c$ elements of $\widetilde{\boldsymbol{y}}$. Then,

$$\begin{aligned} \boldsymbol{\Omega} &= \gamma_0 \mathbf{I} + \mathbf{Z}\boldsymbol{\Sigma}\mathbf{Z}^\top \\ &= \mathbf{Q}\left(\gamma_0 \mathbf{I} + \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix} \boldsymbol{\Sigma} \begin{bmatrix} \mathbf{R}^\top & 0 \end{bmatrix}\right)\mathbf{Q}^\top \\ &= \mathbf{Q}\begin{bmatrix} \mathbf{R}\boldsymbol{\Sigma}\mathbf{R}^\top + \gamma_0 \mathbf{I}_c & \mathbf{0} \\ \mathbf{0} & \gamma_0 \mathbf{I}_{n-c} \end{bmatrix}\mathbf{Q}^\top. \end{aligned}$$

Taking the inverse of this matrix yields

$$\boldsymbol{\Omega}^{-1} = \mathbf{Q}\begin{bmatrix} \left(\mathbf{R}\boldsymbol{\Sigma}\mathbf{R}^\top + \gamma_0 \mathbf{I}_c\right)^{-1} & \mathbf{0} \\ \mathbf{0} & \gamma_0^{-1} \mathbf{I}_{n-c} \end{bmatrix}\mathbf{Q}^\top.$$

and

$$\boldsymbol{y}^\top \boldsymbol{\Omega}^{-1} \boldsymbol{y} = \widetilde{\boldsymbol{y}}_{[c]}^\top \widetilde{\boldsymbol{\Omega}}^{-1} \widetilde{\boldsymbol{y}}_{[c]} + \gamma_0^{-1} \widetilde{\boldsymbol{y}}_{[n-c]}^\top \widetilde{\boldsymbol{y}}_{[n-c]}. \tag{A1}$$

We now consider the simplifications for the trace terms in H. Let

$$\widetilde{\boldsymbol{\Sigma}} = \textbf{block diag}(\widetilde{\gamma}_1 \mathbf{I}_{c_1}, \ldots, \widetilde{\gamma}_m \mathbf{I}_{c_m}).$$

Then, $\mathbf{C} = \widetilde{\gamma}_0 \mathbf{I} + \mathbf{Z}\widetilde{\boldsymbol{\Sigma}}\mathbf{Z}^\top$ and

$$\mathbf{C}^{-1} = \mathbf{Q}\begin{bmatrix} \left(\mathbf{R}\widetilde{\boldsymbol{\Sigma}}\mathbf{R}^\top + \widetilde{\gamma}_0 \mathbf{I}_c\right)^{-1} & \mathbf{0} \\ \mathbf{0} & \widetilde{\gamma}_0^{-1} \mathbf{I}_{n-c} \end{bmatrix}\mathbf{Q}^\top.$$

If we substitute this expression into the trace term when $k \neq 0$, we obtain

$$\begin{aligned}
\operatorname{tr}\left(\mathbf{C}^{-1}\mathbf{V}_k\right) &= \operatorname{tr}\left(\mathbf{Z}_k^\top \mathbf{C}^{-1}\mathbf{Z}_k\right) \\
&= \operatorname{tr}\left(\mathbf{Z}_k^\top \mathbf{Q}\begin{bmatrix}\left(\mathbf{R}\widetilde{\boldsymbol{\Sigma}}\mathbf{R}^\top + \widetilde{\gamma}_0\mathbf{I}_c\right)^{-1} & \mathbf{0} \\ \mathbf{0} & \widetilde{\gamma}_0^{-1}\mathbf{I}_{n-c}\end{bmatrix}\mathbf{Q}^\top\mathbf{Z}_k\right) \\
&= \operatorname{tr}\left(\begin{bmatrix}\mathbf{R}_k^\top & \mathbf{0}\end{bmatrix}\begin{bmatrix}\left(\mathbf{R}\widetilde{\boldsymbol{\Sigma}}\mathbf{R}^\top + \widetilde{\gamma}_0\mathbf{I}_c\right)^{-1} & \mathbf{0} \\ \mathbf{0} & \widetilde{\gamma}_0^{-1}\mathbf{I}_{n-c}\end{bmatrix}\begin{bmatrix}\mathbf{R}_k \\ \mathbf{0}\end{bmatrix}\right) \\
&= \operatorname{tr}\left(\left(\mathbf{R}\widetilde{\boldsymbol{\Sigma}}\mathbf{R}^\top + \widetilde{\gamma}_0\mathbf{I}_c\right)^{-1}\mathbf{R}_k\mathbf{R}_k^\top\right). \qquad \text{(A2)}
\end{aligned}$$

When $k = 0$, we have

$$\begin{aligned}
\operatorname{tr}\left(\mathbf{C}^{-1}\mathbf{V}_0\right) &= \operatorname{tr}\left(\mathbf{Q}\begin{bmatrix}\left(\mathbf{R}\widetilde{\boldsymbol{\Sigma}}\mathbf{R}^\top + \widetilde{\gamma}_0\mathbf{I}_c\right)^{-1} & \mathbf{0} \\ \mathbf{0} & \widetilde{\gamma}_0^{-1}\mathbf{I}_{n-c}\end{bmatrix}\mathbf{Q}^\top\right) \\
&= \operatorname{tr}\left(\begin{bmatrix}\left(\mathbf{R}\widetilde{\boldsymbol{\Sigma}}\mathbf{R}^\top + \widetilde{\gamma}_0\mathbf{I}_c\right)^{-1} & \mathbf{0} \\ \mathbf{0} & \widetilde{\gamma}_0^{-1}\mathbf{I}_{n-c}\end{bmatrix}\right) \\
&= \operatorname{tr}\left(\left(\mathbf{R}\widetilde{\boldsymbol{\Sigma}}\mathbf{R}^\top + \widetilde{\gamma}_0\mathbf{I}_c\right)^{-1}\right) + \widetilde{\gamma}_0^{-1}(n-c). \qquad \text{(A3)}
\end{aligned}$$

If we recall the definitions $\widetilde{\mathbf{V}}_i := \mathbf{R}_i\mathbf{R}_i^\top$, $\widetilde{\mathbf{V}}_0 := \mathbf{I}_c$, $\widetilde{\boldsymbol{\Omega}} := \sum_{i=0}^m \gamma_i\widetilde{\mathbf{V}}_i$ and $\widetilde{\mathbf{C}} := \sum_{i=0}^m \widetilde{\gamma}_i\widetilde{\mathbf{V}}_i$ and combine Equations (A1)–(A3), we obtain

$$\begin{aligned}
h(\boldsymbol{\Omega}, \mathbf{C}) &= \boldsymbol{y}^\top\boldsymbol{\Omega}^{-1}\boldsymbol{y} + \sum_{i=0}^m \gamma_i \operatorname{tr}(\mathbf{C}^{-1}\mathbf{V}_i) + \ln\det(\mathbf{C}) - n \\
&= \widetilde{\boldsymbol{y}}_{[c]}^\top\widetilde{\boldsymbol{\Omega}}^{-1}\widetilde{\boldsymbol{y}}_{[c]} + \sum_{i=0}^m \gamma_i \operatorname{tr}(\widetilde{\mathbf{C}}^{-1}\widetilde{\mathbf{V}}_i) + \ln\det(\mathbf{C}) - n + \gamma_0\widetilde{\gamma_0}^{-1}(n-c) + \gamma_0^{-1}\widetilde{\boldsymbol{y}}_{[n-c]}^\top\widetilde{\boldsymbol{y}}_{[n-c]}.
\end{aligned}$$

Appendix B. Woodbury Identity

Recall from Section 3.3.2 that we have the definitions $\boldsymbol{w} := \mathbf{Z}_k^\top\boldsymbol{\Omega}^{-1}\boldsymbol{y}$; $\mathbf{B} := \mathbf{Z}_k^\top\boldsymbol{\Omega}^{-1}\mathbf{Z}_k$ and $\mathbf{M} := \mathbf{I}_{c_k} + (x - \gamma_k^{(t)})\mathbf{B}$ and the simplified univariate function,

$$h_k(x) = -(x - \gamma_k^{(t)})\boldsymbol{w}^\top\mathbf{M}^{-1}\boldsymbol{w} + x\operatorname{tr}(\mathbf{B}) + \text{const.}$$

We now derive the first and second derivatives of this function. The differentiation of an invertible symmetric matrix implies that

$$\frac{\partial\mathbf{M}^{-1}}{\partial x} = -\mathbf{M}^{-1}\mathbf{B}\mathbf{M}^{-1}. \qquad \text{(A4)}$$

Then, from the product rule of differentiation,

$$h_k'(x) = (x - \gamma_k^{(t)})\boldsymbol{w}^\top\mathbf{M}^{-1}\mathbf{B}\mathbf{M}^{-1}\boldsymbol{w} - \boldsymbol{w}^\top\mathbf{M}^{-1}\boldsymbol{w} + \operatorname{tr}(\mathbf{B}).$$

If we then approximately solve the linear system $\mathbf{M}\boldsymbol{d} = \boldsymbol{w}$ with CG, then

$$h_k'(x) = -\boldsymbol{w}^\top\boldsymbol{d} + (x - \gamma_k^{(t)})\boldsymbol{d}^\top\mathbf{B}\boldsymbol{d} + \operatorname{tr}(\mathbf{B}).$$

Using the matrix product rule of differentiation, we have that

$$\frac{\partial \mathbf{M}^{-1}\mathbf{B}\mathbf{M}^{-1}}{\partial x} = 2\mathbf{M}^{-1}\mathbf{B}\mathbf{M}^{-1}\mathbf{B}\mathbf{M}^{-1}. \tag{A5}$$

Then, using (A4) and (A5) to differentiate $h'_k(x)$, we obtain

$$h''_k(x) = 2\boldsymbol{w}^\top \mathbf{M}^{-1}\mathbf{B}\mathbf{M}^{-1}\boldsymbol{w} + 2(x - \gamma_k^{(t)})\boldsymbol{w}^\top \mathbf{M}^{-1}\mathbf{B}\mathbf{M}^{-1}\mathbf{B}\mathbf{M}^{-1}\boldsymbol{w}.$$

If we solve $\mathbf{M}\boldsymbol{d} = \boldsymbol{w}$ and $\mathbf{M}\boldsymbol{j} = \mathbf{B}\boldsymbol{d}$ with CG then j will approximate the matrix-vector product $\mathbf{M}^{-1}\mathbf{B}\mathbf{M}^{-1}\boldsymbol{w}$ and the second derivative can be evaluated as

$$h''_k(x) = 2\boldsymbol{d}^\top \mathbf{B}\boldsymbol{d} + 2(x - \gamma_k^{(t)})\boldsymbol{d}^\top \mathbf{B}\boldsymbol{j}.$$

References

1. Kang, H.M.; Sul, J.H.; Service, S.K.; Zaitlen, N.A.; Kong, S.y.; Freimer, N.B.; Sabatti, C.; Eskin, E. Variance component model to account for sample structure in genome-wide association studies. *Nat. Genet.* **2010**, *42*, 348–354. [CrossRef] [PubMed]
2. Searle, S.; Casella, G.; McCulloch, C. *Variance Components*; Wiley Series in Probability and Statistics; Wiley: Hoboken, NJ, USA, 2009.
3. Jiang, J.; Nguyen, T. *Linear and Generalized Linear Mixed Models and Their Applications*; Springer: New York, NY, USA, 2007; Volume 1.
4. Harville, D.A. Maximum likelihood approaches to variance component estimation and to related problems. *J. Am. Stat. Assoc.* **1977**, *72*, 320–338. [CrossRef]
5. Jennrich, R.I.; Sampson, P. Newton–Raphson and related algorithms for maximum likelihood variance component estimation. *Technometrics* **1976**, *18*, 11–17. [CrossRef]
6. Longford, N.T. A fast scoring algorithm for maximum likelihood estimation in unbalanced mixed models with nested random effects. *Biometrika* **1987**, *74*, 817–827. [CrossRef]
7. Dempster, A.P.; Laird, N.M.; Rubin, D.B. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. Ser. B (Methodol.)* **1977**, *39*, 1–22.
8. Lindstrom, M.J.; Bates, D.M. Newton–Raphson and EM algorithms for linear mixed-effects models for repeated-measures data. *J. Am. Stat. Assoc.* **1988**, *83*, 1014–1022.
9. Zhou, H.; Hu, L.; Zhou, J.; Lange, K. MM algorithms for variance components models. *J. Comput. Graph. Stat.* **2019**, *28*, 350–361. [CrossRef] [PubMed]
10. Wright, S.J. Coordinate-descent algorithms. *Math. Program.* **2015**, *151*, 3–34. [CrossRef]
11. Nesterov, Y. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM J. Optim.* **2012**, *22*, 341–362. [CrossRef]
12. Luo, Z.Q.; Tseng, P. On the convergence of the coordinate descent method for convex differentiable minimization. *J. Optim. Theory Appl.* **1992**, *72*, 7–35. [CrossRef]
13. Golub, G.H.; Van Loan, C.F. *Matrix Computations*; JHU Press: Baltimore, MD, USA, 2013.
14. Liu, J.S.; Wu, Y.N. Parameter expansion for data augmentation. *J. Am. Stat. Assoc.* **1999**, *94*, 1264–1274. [CrossRef]
15. Meng, X.L.; Van Dyk, D.A. Seeking efficient data augmentation schemes via conditional and marginal augmentation. *Biometrika* **1999**, *86*, 301–320. [CrossRef]
16. Bezdek, J.C.; Hathaway, R.J. Convergence of alternating optimization. *Neural Parallel Sci. Comput.* **2003**, *11*, 351–368.
17. Bezdek, J.; Hathaway, R. Some Notes on Alternating Optimization. In *AFSS International Conference on Fuzzy Systems, Proceedings of the Advances in Soft Computing—AFSS 2002, Calcutta, India, 3–6 February 2002*; Springer: Berlin/Heidelberg, Germany, 2002; Volume 2275, pp. 288–300. [CrossRef]
18. Tseng, P. Convergence of a block coordinate descent method for nondifferentiable minimization. *J. Optim. Theory Appl.* **2001**, *109*, 475–494. [CrossRef]
19. Hartley, H.O.; Rao, J.N. Maximum-likelihood estimation for the mixed analysis of variance model. *Biometrika* **1967**, *54*, 93–108. [CrossRef] [PubMed]
20. Schelldorfer, J.; Bühlmann, P.; DE GEER, S.V. Estimation for high-dimensional linear mixed-effects models using l1-penalization. *Scand. J. Stat.* **2011**, *38*, 197–214. [CrossRef]
21. Zhou, H.; Sinsheimer, J.S.; Bates, D.M.; Chu, B.B.; German, C.A.; Ji, S.S.; Keys, K.L.; Kim, J.; Ko, S.; Mosher, G.D.; et al. OpenMendel: A cooperative programming project for statistical genetics. *Hum. Genet.* **2020**, *139*, 61–71. [CrossRef] [PubMed]