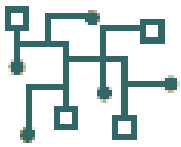

ISE

Industrial and Systems Engineering

A Subspace Acceleration Method for Minimization Involving a Group Sparsity-Inducing Regularizer

Frank E. Curtis, Yutong Dai, and Daniel P. Robinson

Department of Industrial and Systems Engineering, Lehigh University

COR@L Technical Report 20T-015

LEHIGH UNIVERSITY

COR@L

COMPUTATIONAL OPTIMIZATION RESEARCH AT LEHIGH

A Subspace Acceleration Method for Minimization Involving a Group Sparsity-Inducing Regularizer

Frank E. Curtis*[1], Yutong Dai†[1], and Daniel P. Robinson‡[1]

[1]Department of Industrial and Systems Engineering, Lehigh University

July 29, 2020

Abstract

We consider the problem of minimizing an objective function that is the sum of a convex function and a group sparsity-inducing regularizer. Problems that integrate such regularizers arise in modern machine learning applications, often for the purpose of obtaining models that are easier to interpret and that have higher predictive accuracy. We present a new method for solving such problems that utilize subspace acceleration, domain decomposition, and support identification. Our analysis shows, under common assumptions, that the iterate sequence generated by our framework is globally convergent, converges to an ϵ-approximate solution in at most $O(\epsilon^{-(1+p)})$ (respectively, $O(\epsilon^{-(2+p)})$) iterations for all ϵ bounded above and large enough (respectively, all ϵ bounded above) where $p > 0$ is an algorithm parameter, and exhibits superlinear local convergence. Preliminary numerical results for the task of binary classification based on regularized logistic regression show that our approach is efficient and robust, with the ability to outperform a state-of-the-art method.

1 Introduction

We consider the minimization of a function that may be written as the sum of a convex function and a nonoverlapping group sparsity-inducing regularizer. Specifically, given a convex and twice continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, a collection of $n_{\mathcal{G}} > 0$ nonoverlapping groups $\mathcal{G} := \{\mathcal{G}_i\}_{i=1}^{n_{\mathcal{G}}}$ that forms a partition of $\{1, 2, \cdots, n\}$ (i.e., $\mathcal{G}_i \cap \mathcal{G}_j = \emptyset$ for all $i \neq j$ and $\cup_{i=1}^{n_{\mathcal{G}}} \mathcal{G}_i = \{1, 2, \cdots, n\}$), and group-wise weighting parameters $\{\lambda_i\}_{i=1}^{n_{\mathcal{G}}} > 0$, our algorithm solves the problem

$$\min_{x \in \mathbb{R}^n} \{f(x) + r(x)\}, \text{ where } r(x) := \sum_{i=1}^{n_{\mathcal{G}}} \lambda_i \left\| [x]_{\mathcal{G}_i} \right\|_2 \tag{1}$$

and $[x]_{\mathcal{G}_i}$ is the subvector of x corresponding to elements in $\mathcal{G}_i$. The regularizer r generalizes the ℓ_1-norm, which is recovered by choosing $\mathcal{G}_i = \{i\}$ for all $i \in \{1, 2, \ldots, n\}$.

Despite the successes of ℓ_1-norm regularization, its inadequacy in the context of many modern machine learning applications has been noticed by researchers, and is one motivation for the use of group regularization. In some machine learning applications the covariates come in groups (e.g., genes that regulate hormone levels in microarray data [23]), in which case one may wish to select them jointly. Also, integrating group information into the modeling process can improve both the interpretability and accuracy [35] of the resulting model. Yuan and Lin [34] observed that in the multi-factor analysis-of-variance problem, where each factor is expressed through a set of dummy variables, deleting an irrelevant factor is equivalent to deleting a *group* of dummy variables; the ℓ_1-norm regularizer fails to achieve this goal.

*E-mail: frank.e.curtis@gmail.com
†E-mail: yud319@lehigh.edu
‡E-mail: daniel.p.robinson@gmail.com

1.1 State-of-the art methods

There is a long history of algorithms for solving regularized problems of the form (1) (see [1] and the references therein). Here, we review some of the state-of-the-art approaches for solving sparsity-promoting problems that are most closely related to our proposed approach.

First-order methods. Proximal methods are designed to solve problems of the form (1) and have received attention in the machine learning community [3, 7, 31]. A well-known example for ℓ_1-norm regularized problems is the iterative shrinkage-thresholding algorithm (ISTA), which is obtained by applying a proximal gradient (PG) iteration to minimize a smooth function plus the ℓ_1-norm regularizer [10, 12]. Under certain assumptions, one can prove a worst-case complexity bound on the number of iterations required by the PG method before it correctly identifies the support of the optimal solution [28]. Combined with the acceleration technique proposed by Nesterov [27, 26], one obtains the algorithm FISTA [3]. One obtains a related, but distinct approach from ISTA by posing an equivalent smooth reformulation of the problem—separating the positive and negative parts of the variables—and applying a gradient projection method to the resulting formulation [13, 15]. All of these approaches have been shown to work well in practice, at least compared to other first-order methods such as the subgradient algorithm. However, these algorithms are often inferior in practice compared to alternative approaches that employ space decomposition techniques and/or second-order derivatives [5, 6, 18].

As an alternative to PG and gradient projection techniques, researchers have considered (block) coordinate descent for solving ℓ_1-norm regularized problems. Such a strategy is appealing, since when minimizing an ℓ_1-norm regularized objective along coordinate directions, it is common that the objective is minimized with variables being zero. These approaches are also easy to implement to exploit parallel computing; see, e.g., the accelerated randomized proximal coordinate gradient method in [20], the parallel coordinate descent methods in [29], and the asynchronous coordinate descent technique in [22]. A downside of these approaches is that the space decomposition is performed in a prescribed manner, rather than in an adaptive way that can benefit from information acquired during the solution process. Also, these approaches do not effectively exploit second-order derivative information and require exact minimization along coordinate directions. An exception to this latter criticism is the inexact coordinate descent algorithm from [30], although this approach does not effectively exploit second-order derivatives and uses a prescribed space decomposition strategy.

Various other approaches have been proposed for solving problems involving specific regularizers. In [21], the authors discuss various methods for sparse learning that make use of projection techniques. A well-known package is `GLMNET` [16], which is designed for solving problems with the elastic-net regularization. Finally, let us mention the work in [32], which proposes and tests a groupwise-majorization-descent algorithm (called `gglasso`) for solving problems involving the group-ℓ_1-norm regularizer. A potential downside of this approach is that it updates variables by groups in a cycle, rather than by using an adaptive space decomposition technique.

Second-order methods. Relatively few second-order methods have been proposed for minimizing sparsity-promoting objective functions. In s[17], an accelerated regularized Newton scheme is proposed. A similar proximal-Newton method is proposed in [19], which under some assumptions can be shown to converge locally superlinearly. These approaches can be effective in practice, although they appear to lack good worst-case guarantees in terms of identification of the optimal solution support. Other approaches, such as the orthant-based method in [18], can predict the solution support, but in practice are often outperformed by a closely related method called `FaRSA` [5, 6]. As for publicly available solvers based on second-order methods, most have been designed for specific loss functions and regularizers. For example, `newGLMNET` in [33] is designed for ℓ_1-regularized logistic regression and the method in [14] is designed for regularized logistic regression and support vector machines.

1.2 Notation and assumptions

Let $\mathbb{R}$ denote the set of real numbers, $\mathbb{R}^n$ denote the set of n-dimensional real vectors, and $\mathbb{R}^{m\times n}$ denote the set of m-by-n-dimensional real matrices. The set of natural numbers is denoted as $\mathbb{N} := \{0, 1, 2, \dots\}$. For any set $\mathcal{I} \subseteq \{1, 2, \dots, n\}$, we define the projection of $x \in \mathbb{R}^n$ onto the subspace spanned by the coordinate

vectors indexed by the entries of $\mathcal{I}$ as $P_{\mathcal{I}}(x)$, so that

$$[P_{\mathcal{I}}(x)]_i := \begin{cases} x_i & \text{if } i \in \mathcal{I}, \\ 0 & \text{if } i \notin \mathcal{I}. \end{cases} \tag{2}$$

For a function $h : \mathbb{R}^n \to \mathbb{R}$, vector $x \in \mathbb{R}^n$, and direction $d \in \mathbb{R}^n$, the directional derivative of h at x in the direction d is defined as the following limit:

$$D_h(x; d) := \lim_{t \searrow 0} \frac{h(x + td) - h(x)}{t}.$$

The following assumption is assumed to hold throughout the paper.

Assumption 1.1. *The function $f : \mathbb{R}^n \to \mathbb{R}$ used in the definition of the objective function of problem (1) is convex and continuously differentiable. It follows that there exists a constant L_f such that $\|\nabla f(x)\|_2 \leq L_f$ for all $x \in \mathcal{L} := \{x \in \mathbb{R}^n : f(x) + r(x) \leq f(x_0) + r(x_0)\}$ for any initial estimate x_0 of a solution to problem (1). The objective function $f + r$ is bounded below and the gradient function ∇f is Lipschitz continuous on $\mathcal{L}$ with Lipschitz constant L_g.*

1.3 Organization

In Section 2, we present preliminary results related to PG calculations. In Section 3, by using PG-calculations as a starting point, we propose a reduced-space second-order domain decomposition algorithm for solving problem (1). The algorithm is analyzed in Section 4 and numerical results are presented in Section 5. Finally, in Section 6, we provide concluding remarks.

2 Preliminaries

In this section, we discuss preliminary material related to the objective function $f + r$ and its associated PG calculations. (All proofs may be found in Appendix A.) For any $\overline{x} \in \mathbb{R}^n$ and $\overline{\alpha} > 0$, we define the PG *update* as

$$T(\overline{x}, \overline{\alpha}) := \operatorname*{argmin}_{x \in \mathbb{R}^n} \left\{ \tfrac{1}{2\overline{\alpha}} \|x - \big(\overline{x} - \overline{\alpha}\nabla f(\overline{x})\big)\|_2^2 + r(x) \right\} \tag{3}$$

and the associated PG *step* as

$$s(\overline{x}, \overline{\alpha}) := T(\overline{x}, \overline{\alpha}) - \overline{x}. \tag{4}$$

The next result shows that the directional derivative of $f + r$ along the PG step is negative with magnitude proportional to the squared norm of the PG direction.

Lemma 2.1. *For any $\overline{x} \in \mathbb{R}^n$ and $\overline{\alpha} > 0$, the PG step $s(\overline{x}, \overline{\alpha})$ in (4) satisfies*

$$D_{f+r}(\overline{x}; s(\overline{x}, \overline{\alpha})) \leq -\tfrac{1}{\overline{\alpha}} \|s(\overline{x}, \overline{\alpha})\|_2^2.$$

The PG update defined in (3) can be computed group-wise for each $\mathcal{G}_i \in \mathcal{G}$ by

$$\begin{aligned} [T(\overline{x}, \overline{\alpha})]_{\mathcal{G}_i} &= \Big[\operatorname*{argmin}_{x \in \mathbb{R}^n} \Big\{ \tfrac{1}{2\overline{\alpha}} \|x - \big(\overline{x} - \overline{\alpha}\nabla f(\overline{x})\big)\|_2^2 + \sum_{i=1}^{n_{\mathcal{G}}} \lambda_i \|x_{\mathcal{G}_i}\|_2 \Big\} \Big]_{\mathcal{G}_i} \\ &= \max\left\{ 1 - \frac{\overline{\alpha}\lambda_i}{\|[\overline{x}]_{\mathcal{G}_i} - \overline{\alpha}\nabla_{\mathcal{G}_i} f(\overline{x})\|_2}, 0 \right\} \Big([\overline{x}]_{\mathcal{G}_i} - \overline{\alpha}\nabla_{\mathcal{G}_i} f(\overline{x}) \Big). \end{aligned} \tag{5}$$

Combining this observation with Lemma 2.1 leads to the following corollary, which will be relevant to the manner in which we design the algorithm we propose in Section 3.

Lemma 2.2. *For any $\overline{x} \in \mathbb{R}^n$, $\overline{\alpha} > 0$, and set $\mathcal{I}$ equal to the union of a subset of $\{\mathcal{G}_i\}_{i=1}^{n_\mathcal{G}}$, the PG step $s(\overline{x}, \overline{\alpha})$ defined in (4) satisfies*

$$D_{f+r}\big(\overline{x}; P_\mathcal{I}(s(\overline{x}, \overline{\alpha}))\big) \leq -\tfrac{1}{\overline{\alpha}} \|P_\mathcal{I}(s(\overline{x}, \overline{\alpha}))\|_2^2 \tag{6}$$

where the projection operator $P_\mathcal{I}$ is defined through (2).

Our next result quantifies the decrease in $f+r$ that one can expect to obtain by taking a PG step $s(\overline{x}, \overline{\alpha})$, provided the PG parameter $\overline{\alpha}$ is sufficiently small.

Lemma 2.3. *For any $\overline{x} \in \mathbb{R}^n$, $\overline{\alpha} \in (0, 2/L)$, and $\mathcal{I}$ equal to the union of a subset of $\{\mathcal{G}_i\}_{i=1}^{n_\mathcal{G}}$, the objective function decrease satisfies*

$$f(\overline{x} + P_\mathcal{I}(\overline{x}, \bar{s})) + r(\overline{x} + P_\mathcal{I}(\overline{x}, \bar{s})) \leq f(\overline{x}) + r(\overline{x}) - (\tfrac{1}{\overline{\alpha}} - \tfrac{L}{2}) \|P_\mathcal{I}\big(s(\overline{x}, \overline{\alpha})\big)\|_2^2.$$

The next result shows that, when restricted to certain groups, the size of the PG step is bounded above by the gradient of the objective function.

Lemma 2.4. *If the pair $(\overline{x}, \overline{\alpha})$ and group $\mathcal{G}_i$ satisfy $\overline{\alpha} \in (0, 1]$, $[\overline{x}]_{\mathcal{G}_i} \neq 0$, and $[\overline{x} + s(\overline{x}, \overline{\alpha})]_{\mathcal{G}_i} \neq 0$, where $s(\overline{x}, \overline{\alpha})$ is defined in (4), then*

$$\|\nabla_{\mathcal{G}_i}(f + r)(\overline{x})\|_2 \geq \|[s(\overline{x}, \overline{\alpha})]_{\mathcal{G}_i}\|_2.$$

With the preliminaries now completed, we can propose our new algorithm.

3 Proposed Algorithm Framework

We propose Algorithm 1, which we call `FaRSA-Group` (`Fa`st `R`educed-`S`pace `A`lgorithm for `Group` sparsity-inducing regularization) for solving problem (1) that uses ideas related to domain decomposition, subspace acceleration, and support identification. An overview of the algorithm is described in Section 3.1. During each iteration of our method, at least one of three subroutines is called. The three subroutines are described in Sections 3.2–3.4.

3.1 Main algorithm (Algorithm 1)

Our main algorithm is formally stated as Algorithm 1. At the beginning of the kth iteration, x_k and $\alpha_k > 0$ denote the current solution estimate for problem (1) and the PG parameter, respectively. We then compute s_k in Line 5 as the PG step associated with problem (1), namely,

$$s_k := s(x_k, \alpha_k) \quad \text{with} \quad s(x_k, \alpha_k) \text{ defined in (4).} \tag{7}$$

Although the repeated computation of PG steps is the basis for a first-order method, here we primarily use it to *predict* the zero/nonzero structure of a solution and to formulate optimality measures. Specifically, in Line 6 we compute the index set

$$\begin{aligned} \bar{\mathcal{I}}_k^{\text{cg}} := \{j \in \mathcal{G}_i : [x_k]_{\mathcal{G}_i} \neq 0, \;\; [x_k + s_k]_{\mathcal{G}_i} \neq 0, \text{ and} \\ \|[x_k]_{\mathcal{G}_i}\|_2 \geq \kappa_1 \|\nabla_{\mathcal{G}_i}(f + r)(x_k)\|_2\} \end{aligned} \tag{8}$$

for some $\kappa_1 \in (0, \infty)$. The groups of variables that compose $\bar{\mathcal{I}}_k^{\text{cg}}$ are *candidates* for use in a Newton-type calculation aimed to accelerated convergence. Before using them, however, we first check to see if each candidate block is sufficiently far from zero, and those that are not are removed. Specifically, we first define

$$\mathcal{I}_k^{\text{small}} := \{j \in \mathcal{G}_i : \mathcal{G}_i \subseteq \bar{\mathcal{I}}_k^{\text{cg}} \text{ and } \|[x_k]_{\mathcal{G}_i}\|_2 < \kappa_2 \|\nabla_{\bar{\mathcal{I}}_k^{\text{cg}}}(f + r)(x_k)\|_2^p\} \tag{9}$$

for some $\{\kappa_2, p\} \subset (0, \infty)$, and then define in Line 7 the sets and optimality measures

$$\left\{ \begin{array}{l} \mathcal{I}_k^{\rm cg} := \bar{\mathcal{I}}_k^{\rm cg} \setminus \mathcal{I}_k^{\rm small} \\ \mathcal{I}_k^{\rm pg} := \{1, 2, \ldots, n\} \setminus \mathcal{I}_k^{\rm cg} \end{array} \right\} \quad \text{and} \quad \left\{ \begin{array}{l} \chi_k^{\rm cg} := \|[s_k]_{\mathcal{I}_k^{\rm cg}}\|_2 \\ \chi_k^{\rm pg} := \|[s_k]_{\mathcal{I}_k^{\rm pg}}\|_2 \end{array} \right\} \tag{10}$$

where by convention $\|[\ \cdot\ ]_\emptyset\|_2 = 0$. (See Lemma 4.1 for a justification that these sets together represent a measure of optimality.) This construction of sets also ensures that the subvector of x_k that corresponds to $\mathcal{G}_i$ for each $\mathcal{G}_i \subseteq \mathcal{I}_k^{\rm cg}$ is at least a distance

$$\rho_{k,i} := \max\{\kappa_1 \|\nabla_{\mathcal{G}_i}(f + r)(x_k)\|_2, \kappa_2 \|\nabla_{\mathcal{I}_k^{\rm cg}}(f + r)(x_k)\|_2^p\} \tag{11}$$

away from zero (see Lemma 4.5(i)), which is crucial in our analysis.

Armed with $\chi_k^{\rm pg}$ and $\chi_k^{\rm cg}$, Algorithm 1 seeks decrease in the objective function in a subspace that is likely to allow for significant progress. We consider two cases.

Case 1: the condition $\chi_k^{\rm pg} \leq \chi_k^{\rm cg}$ checked in Line 8 holds. In this case, the inequality $\chi_k^{\rm pg} \leq \chi_k^{\rm cg}$ indicates that significant reduction in the objective function can be achieved by focusing on variables in the set $\mathcal{I}_k^{\rm cg}$. Therefore, in Line 9 we choose any index set $\mathcal{I}_k$ that is (i) a subset of $\mathcal{I}_k^{\rm cg}$, (ii) equal to the union of some subset of groups from $\mathcal{G}$, and (iii) the size of the PG step restricted to the index set $\mathcal{I}_k$ is at least a fraction of the size of the PG step when restricted to the index set $\mathcal{I}_k^{\rm cg}$. The easiest choice that satisfies these conditions is $\mathcal{I}_k \equiv \mathcal{I}_k^{\rm cg}$, but for large-scale problems it may be beneficial to restrict $|\mathcal{I}_k|$. The opposite extreme choice is selecting $\mathcal{I}_k$ as the group $\mathcal{G}_i$ contained in $\mathcal{I}_k^{\rm cg}$ with largest associated PG step, in which case one would choose $\varphi = 1/\sqrt{n_{\mathcal{G}}}$ for the user-defined parameter in Line 9. Once $\mathcal{I}_k$ has been selected, a *reduced-space* gradient g_k and *reduced-space* positive-definite matrix H_k is computed in Line 10, where the derivatives are taken with respect to variables in $\mathcal{I}_k$. (In practice, H_k could be selected based on $\nabla^2_{\mathcal{I}_k\mathcal{I}_k}(f + r)(x_k)$ to ensure a fast local convergence rate.) Note that such derivatives exist since by construction $\mathcal{I}_k \subseteq \mathcal{I}_k^{\rm cg} \subseteq \bar{\mathcal{I}}_k^{\rm cg}$, and from (8) the objective function $f + r$ is differentiable with respect to groups of variables in $\bar{\mathcal{I}}_k^{\rm cg}$. Next, g_k and H_k are used to compute a direction $\bar{d}_k$ of sufficient descent for $f + r$ by calling the subroutine CG_DIRECTION (see Section 3.2). Once a full-space vector d_k is obtained by padding $\bar{d}_k$ with zeros in Line 12, a *projected* line search is performed by calling subroutine UPDATE_CG in Line 13 (see Section 3.3).

Case 2: the condition $\chi_k^{\rm pg} \leq \chi_k^{\rm cg}$ checked in Line 8 does not hold. In this case, the inequality $\chi_k^{\rm pg} > \chi_k^{\rm cg}$ indicates that significant reduction in the objective function can be achieved by focusing on variables in the set $\mathcal{I}_k^{\rm pg}$. Therefore, in Line 16, we choose any index set $\mathcal{I}_k$ that is (i) a subset of $\mathcal{I}_k^{\rm pg}$, (ii) equal to the union of some subset of groups from $\mathcal{G}$, and (iii) the size of the PG step restricted to the index set $\mathcal{I}_k$ is at least a fraction of the size of the PG step restricted to the index set $\mathcal{I}_k^{\rm pg}$. The easiest choice that satisfies these conditions is $\mathcal{I}_k \equiv \mathcal{I}_k^{\rm pg}$. Once $\mathcal{I}_k$ has been chosen, the next iterate is obtained by performing a line search along the PG direction in Line 17 by calling the subroutine UPDATE_PG (for details, see Section 3.4). If the subroutine returns $\text{flag}_k^{\rm pg} = \texttt{decrease\_}\alpha$, the PG parameter is decreased for the next iteration.

3.2 Computing a CG direction (Algorithm 2)

This subroutine returns a reduced-space direction $\bar{d}_k$ that satisfies conditions (13)–(15). We call it a reduced-space vector because the inputs g_k and H_k are elements in $\mathbb{R}^{|\mathcal{I}_k|}$ and $\mathbb{R}^{|\mathcal{I}_k| \times |\mathcal{I}_k|}$, respectively, where $\mathcal{I}_k$ is computed in Line 9 of Algorithm 1. Condition (13) ensures that $\bar{d}_k$ is a descent direction for the objective function as a consequence of how the reference direction d_k^R is computed in Line 28. Condition (14) ensures that $\bar{d}_k$ reduces the model m_k at least as much as a zero step. Finally, condition (15) promotes fast *local* convergence of the iterate sequence $\{x_k\}$ (see Section 4.2), but its enforcement (or lack of enforcement) is irrelevant with respect to the complexity result that we prove in Section 4.1. The subroutine name CG_DIRECTION indicates our intent to use the linear CG algorithm in our implementation, although other possible options include a block-wise coordinate descent method applied to the model m_k in (12). In particular, the direction associated with every iteration of the CG algorithm satisfies conditions (13)–(14),

and condition (15) is satisfied by all sufficiently large CG iterations. Thus, the requirements of this subroutine can always be met.

3.3 Reduced-space search using the CG direction (Algorithm 3)

This subroutine performs a search using the direction d_k returned by the subroutine CG_DIRECTION in Line 11 of Algorithm 1. For an illustration of this search, which incorporates projections, see Figure 1. The approach uses the direction d_k, without modification, for each block of variables $\mathcal{G}_i$ such that the ray $\{[x_k + \tau d_k]_{\mathcal{G}_i} : \tau \geq 0\}$ does not intersect the ball centered at zero of radius $\bar{\rho}_{k,i} = \min\{\rho_{k,i}, \sin(\theta)\|[x_k]_{\mathcal{G}_i}\|_2\}$, where $\rho_{k,i}$ is defined in (11) and $\theta \in (0, \pi/2)$ is a user-defined parameter. When they do intersect, we first compute $\tau_{k,i}$ as the smallest step along the Newton direction (restricted to block $\mathcal{G}_i$) that intersects the ball. Then, during the search that follows, anytime the trial step size ξ^j is larger than $\tau_{k,i}$, the trial step for block $\mathcal{G}_i$ is set to zero; otherwise, the Newton direction is used so that the trial step (with respect to block $\mathcal{G}_i$) is $[x_k + \xi^j d_k]_{\mathcal{G}_i}$ (see Line 47). If termination occurs in Line 49, then a new block of variables will become zero, in which case we require the objective function not to increase (see Line 50). On the other hand, if termination occurs in Line 57, then it indicates that the objective function has been sufficiently reduced (see Line 56) and no new groups of zeros have been formed.

3.4 Reduced-space line search along a PG step (Algorithm 4)

This subroutine performs a line search along the PG direction $P_{\mathcal{I}}(s_k)$. The search ensures that the next iterate yields decrease in the objective of size at least $(\eta\xi^j/\alpha_k)\|P_{\mathcal{I}_k}(s_k)\|_2^2$ for some positive integer j computed within the while loop in Line 65. Once the while loop terminates, the update $\text{flag}_k^{\text{pg}} \leftarrow$ `same_`α is made if $j = 0$, and set as $\text{flag}_k^{\text{pg}} \leftarrow$ `decrease_`α otherwise. The motivation for this update is Lemma 2.3, which shows that the while loop in Line 65 will terminate with $j = 0$ if the PG parameter α_k is sufficiently small. Therefore, anytime $j > 0$, Algorithm 4 returns $\text{flag}_k^{\text{pg}} \leftarrow$ `decrease_`α to Algorithm 1 in Line 17 so that the PG parameter value for the next iteration is reduced by a factor of $\xi \in (0, 1)$ in Line 19.

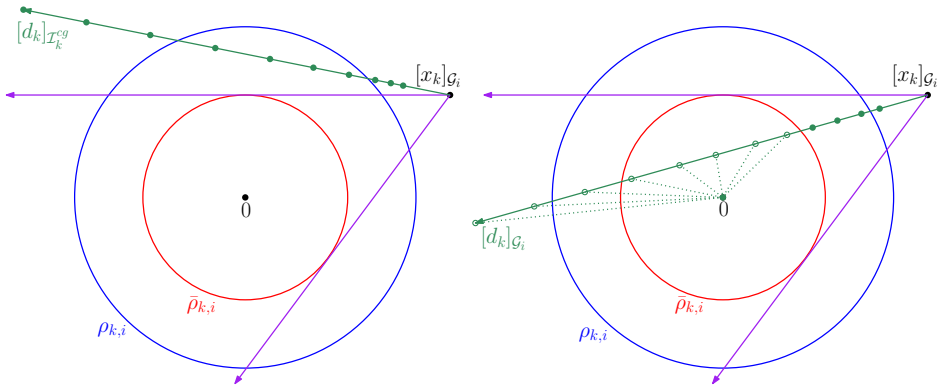

Figure 1: The reduced-space projected search based on the Newton-CG direction d_k described in Section 3.3. In the figure on the left, the direction d_k does not intersect the ball of radius $\bar{\rho}_{k,i}$. In this case, standard backtracking is used, as indicated by the solid green dots. In the figure on the right, the direction d_k does intersect the ball of radius $\bar{\rho}_{k,i}$. In this case, all points after the first point of intersection (indicated by hollow green circles) are projected to zero. Once the backtracking points leave the ball of radius $\bar{\rho}_{k,i}$ (indicated as solid green dots), standard backtracking is resumed.

Algorithm 1 `FaRSA-Group` for solving problem (1).

Input: x_0
Constants: $\{\varphi, \xi, \eta, \zeta\} \subset (0,1)$, $\{\kappa_1, \kappa_2, p\} \subset (0, \infty)$, $\theta \in (0, \pi/2)$, and $q \in [1, 2]$.
Choose any initial PG parameter $\alpha_0 \in (0, 1]$.
for $k = 0, 1, 2, \ldots$ **do**
  Compute the step s_k from (7).
  Compute the set $\bar{\mathcal{I}}_k^{\text{cg}}$ from (8).
  Compute $\mathcal{I}_k^{\text{cg}}$ and $\mathcal{I}_k^{\text{pg}}$ and their optimality measures χ_k^{cg} and χ_k^{pg} from (10).
  if $\chi_k^{\text{pg}} \leq \chi_k^{\text{cg}}$ **then**
    Choose any $\mathcal{I}_k \subseteq \mathcal{I}_k^{\text{cg}}$ such that

$$\|[s_k]_{\mathcal{I}_k}\|_2 \geq \varphi \|[s_k]_{\mathcal{I}_k^{\text{cg}}}\|_2 \equiv \varphi \chi_k^{\text{cg}} \text{ and } \mathcal{I}_k \text{ is the union of some } \{\mathcal{G}_j\}.$$

    Set $g_k \leftarrow \nabla_{\mathcal{I}_k}(f + r)(x_k)$ and pick a positive-definite $H_k \in \mathbb{R}^{|\mathcal{I}_k| \times |\mathcal{I}_k|}$.
    Call Algorithm 2 to obtain $\bar{d}_k \leftarrow \text{CG\_DIRECTION}(g_k, H_k)$.
    Set $[d_k]_{\mathcal{I}_k} \leftarrow \bar{d}_k$ and $[d_k]_{\mathcal{I}_k^c} \leftarrow 0$.
    Call Algorithm 3 to obtain $(x_{k+1}, \text{flag}_k^{\text{cg}}) \leftarrow \text{UPDATE\_CG}(x_k, d_k, \mathcal{I}_k)$.
    Set $\alpha_{k+1} \leftarrow \alpha_k$.
  else
    Choose any $\mathcal{I}_k \subseteq \mathcal{I}_k^{\text{pg}}$ such that

$$\|[s_k]_{\mathcal{I}_k}\|_2 \geq \varphi \|[s_k]_{\mathcal{I}_k^{\text{pg}}}\|_2 \equiv \varphi \chi_k^{\text{pg}} \text{ and } \mathcal{I}_k \text{ is the union of some } \{\mathcal{G}_j\}.$$

    Call Algorithm 4 to obtain $(x_{k+1}, \text{flag}_k^{\text{pg}}) \leftarrow \text{UPDATE\_PG}(x_k, s_k, \alpha_k, \mathcal{I}_k)$.
    if $\text{flag}_k^{\text{pg}} =$ `decrease_`α **then**
      $\alpha_{k+1} \leftarrow \zeta \alpha_k$.
    else
      $\alpha_{k+1} \leftarrow \alpha_k$.
    end if
  end if
end for

Algorithm 2 Computing $\bar{d}_k$ in Line 11 of Algorithm 1.

procedure $\bar{d}_k = \text{CG\_DIRECTION}(g_k, H_k)$
  Constant: q is provided by Algorithm 1.
  Define the model

$$m_k(d) := g_k^T d + \tfrac{1}{2} d^T H_k d. \tag{12}$$

  Compute the reference direction (an approximate minimizer of m_k) as

$$d_k^R \leftarrow -\beta_k g_k, \text{ where } \beta_k \leftarrow \|g_k\|_2^2 / (g_k^T H_k g_k).$$

  Choose $\mu_k \in (0, 1]$ and then compute any $\bar{d}_k \approx \operatorname{argmin}_d m_k(d)$ that satisfies

$$g_k^T \bar{d}_k \leq g_k^T d_k^R, \tag{13}$$

$$m_k(\bar{d}_k) \leq m_k(0), \quad \text{and} \tag{14}$$

$$\|H_k \bar{d}_k + g_k\|_2 \leq \mu_k \|g_k\|_2^q. \tag{15}$$

  return $\bar{d}_k$
end procedure

Algorithm 3 Computing x_{k+1} in Line 13 of Algorithm 1.

procedure $(x_{k+1}, \text{flag}_k^{\text{cg}}) = \text{UPDATE\_CG}(x_k, d_k, \mathcal{I}_k)$
  Constants: η, ξ, and θ provided by Algorithm 1.
  for each i such that $\mathcal{G}_i \subseteq \mathcal{I}_k$ **do**
    Compute $\rho_{k,i}$ as defined in (11).
    Set $\bar{\rho}_{k,i} \leftarrow \min\{\rho_{k,i}, \sin(\theta)\|[x_k]_{\mathcal{G}_i}\|_2\}$.
    if $\{[x_k + \tau d_k]_{\mathcal{G}_i} : \tau \geq 0\} \cap \{x \in \mathbb{R}^{|\mathcal{G}_i|} : \|x\|_2 \leq \bar{\rho}_{k,i}\} = \emptyset$ **then**
      Set $\tau_{k,i} \leftarrow \infty$.
    else
      Set $\tau_{k,i}$ as the smallest positive root of $\|[x_k + \tau d_k]_{\mathcal{G}_i}\|_2 = \bar{\rho}_{k,i}$.
    end if
  end for
  Set $j \leftarrow 0$ and $\tau_k := \min_i\{\tau_{k,i} : \mathcal{G}_i \subseteq \mathcal{I}_k\}$.
  while $\xi^j \geq \tau_k$ **do**
    Set $[y_j]_{\mathcal{I}_k^c} \leftarrow [x_k]_{\mathcal{I}_k^c}$.
    for each i such that $\mathcal{G}_i \in \mathcal{I}_k$ **do**
      Set $[y_j]_{\mathcal{G}_i} \leftarrow \begin{cases} [x_k]_{\mathcal{G}_i} + \xi^j [d_k]_{\mathcal{G}_i} & \text{if } \xi^j < \tau_{k,i}, \\ 0 & \text{if } \xi^j \geq \tau_{k,i}. \end{cases}$
    end for
    if $f(y_j) + r(y_j) \leq f(x_k) + r(x_k)$ **then**
      return $x_{k+1} \leftarrow y_j$ and $\text{flag}_k^{\text{cg}} \leftarrow$ `new_zero`
    end if
    Set $j \leftarrow j + 1$.
  end while
  loop
    Set $y_j \leftarrow x_k + \xi^j d_k$.
    if $f(y_j) + r(y_j) \leq f(x_k) + r(x_k) + \eta \xi^j \nabla_{\mathcal{I}_k}(f + r)(x_k)^T [d_k]_{\mathcal{I}_k}$ **then**
      return $x_{k+1} \leftarrow y_j$ and $\text{flag}_k^{\text{cg}} \leftarrow$ `suff_descent`
    end if
    Set $j \leftarrow j + 1$.
  end loop
end procedure

4 Analysis

Our analysis considers worst-case complexity (Section 4.1) and local convergence (Section 4.2) properties of Algorithm 1. To identify an approximate solution to problem (1), we use the measure $\max\{\chi_k^{\text{pg}}, \chi_k^{\text{cg}}\}$, as we now justify.

Lemma 4.1. *Let $\mathcal{K} \subseteq \mathbb{N}$ be such that $\lim_{k \in \mathcal{K}} x_k = x_*$ and $\lim_{k \in \mathcal{K}} \alpha_k = \alpha_* > 0$. Then, x_* is a solution to problem (1) if and only if $\lim_{k \in \mathcal{K}} \max\{\chi_k^{\text{pg}}, \chi_k^{\text{cg}}\} = 0$.*

Proof. First, we may apply [8, Theorem 3.2.8], with the choice $y = (\overline{x}, \overline{\alpha})$ and the set map $\mathcal{C}(y) = \mathbb{R}^n$, to the objective function appearing in (3) to conclude that $T(\overline{x}, \overline{\alpha})$ is continuous on $\mathbb{R}^n \times (0, \infty)$. Combining this property with the definition of T in (3) and the assumption that $\lim_{k \in \mathcal{K}}(x_k, \alpha_k) = (x_*, \alpha_*)$ with $\alpha_* > 0$ shows that $\lim_{k \in \mathcal{K}} s_k = \lim_{k \in \mathcal{K}} \big(T(x_k, \alpha_k) - x_k\big) = T(x_*, \alpha_*) - x_*$. It follows from this limit and the fact that Assumption 1.1 and [2, Theorem 10.7] together show that x_* is a solution to problem (1) if and only if $T(x_*, \alpha_*) = x_*$. □

If $\max\{\chi_k^{\text{cg}}, \chi_k^{\text{pg}}\} = 0$ for some $k \in \mathbb{N}$, then Lemma 4.1 implies that x_k is a solution to problem (1). Hence, all that remains is to consider the behavior of Algorithm 1 when an infinite number of iterations is

Algorithm 4 Computing x_{k+1} in Line 17 of Algorithm 1.

62: **procedure** $(x_{k+1}, \text{flag}_k^{\text{pg}}) = \text{UPDATE\_PG}(x_k, s_k, \alpha_k, \mathcal{I}_k)$
63: **Constants:** η and ξ provided by Algorithm 1.
64: Set $j \leftarrow 0$ and $y_0 \leftarrow x_k + P_{\mathcal{I}_k}(s_k)$.
65: **while** $f(y_j) + r(y_j) > f(x_k) + r(x_k) - \eta\xi^j \frac{1}{\alpha_k}\|P_{\mathcal{I}_k}(s_k)\|_2^2$ **do**
66: Set $j \leftarrow j + 1$ and then $y_j \leftarrow x_k + \xi^j P_{\mathcal{I}_k}(s_k)$.
67: **end while**
68: **if** $j = 0$ **then**
69: **return** $x_{k+1} \leftarrow y_j$ and $\text{flag}_k^{\text{pg}} \leftarrow$ `same_`α
70: **else**
71: **return** $x_{k+1} \leftarrow y_j$ and $\text{flag}_k^{\text{pg}} \leftarrow$ `decrease_`α
72: **end if**
73: **end procedure**

performed. To focus on this case, we make the following assumption, which is assumed to hold throughout the rest of this section.

Assumption 4.1. *For all iterations $k \in \mathbb{N}$, it holds that* $\max\{\chi_k^{\text{cg}}, \chi_k^{\text{pg}}\} > 0$.

Since our analysis considers the properties of the sequence of iterates, it is convenient to define the following partition of iterations performed by Algorithm 1:

$$\begin{aligned}
\mathcal{K}^{\text{cg}} &:= \{k \in \mathbb{N} : \text{Line 13 is reached during the } k\text{th iteration}\},\\
\mathcal{K}_0^{\text{cg}} &:= \{k \in \mathcal{K}^{\text{cg}} : \text{subroutine UPDATE_CG returns } \text{flag}_k^{\text{cg}} = \texttt{new_zero} \text{ in Line 13}\},\\
\mathcal{K}_{\text{sd}}^{\text{cg}} &:= \{k \in \mathcal{K}^{\text{cg}} : \text{subroutine UPDATE_CG returns } \text{flag}_k^{\text{cg}} = \texttt{suff_descent} \text{ in Line 13}\},\\
\mathcal{K}^{\text{pg}} &:= \{k \in \mathbb{N} : \text{Line 17 is reached during the } k\text{th iteration}\},\\
\mathcal{K}_{\rightarrow}^{\text{pg}} &:= \{k \in \mathcal{K}^{\text{pg}} : \text{subroutine UPDATE_PG returns } \text{flag}_k^{\text{pg}} = \texttt{same_}\alpha \text{ in Line 17}\}, \text{ and}\\
\mathcal{K}_{\downarrow}^{\text{pg}} &:= \{k \in \mathcal{K}^{\text{pg}} : \text{subroutine UPDATE_PG returns } \text{flag}_k^{\text{pg}} = \texttt{decrease_}\alpha \text{ in Line 17}\},
\end{aligned}$$

so that $\mathcal{K}^{\text{cg}} = \mathcal{K}_0^{\text{cg}} \cup \mathcal{K}_{\text{sd}}^{\text{cg}}$, $\mathcal{K}^{\text{pg}} = \mathcal{K}_{\rightarrow}^{\text{pg}} \cup \mathcal{K}_{\downarrow}^{\text{pg}}$, and $\mathbb{N} = \mathcal{K}^{\text{cg}} \cup \mathcal{K}^{\text{pg}}$.

Finally, we assume that the symmetric and positive-definite matrices required in Line 10 are chosen to be bounded and uniformly positive definite.

Assumption 4.2. *The matrix sequence $\{H_k\}_{k \in \mathcal{K}^{\text{cg}}}$ chosen in Line 10 is bounded and uniformly positive definite. That is, there exist constants $0 < \mu_{\min} \leq \mu_{\max} < \infty$ such that $\mu_{\min}\|v\|_2^2 \leq v^T H_k v \leq \mu_{\max}\|v\|_2^2$ for all $k \in \mathcal{K}^{\text{cg}}$ and $v \in \mathbb{R}^{|\mathcal{I}_k|}$.*

4.1 Complexity result

We first focus our attention on iterations in $\mathcal{K}^{\text{pg}}$. The next result shows that Algorithm 4 is well posed and that the new iterate that it produces satisfies a decrease property that will be useful for our complexity analysis.

Lemma 4.2. *For each $k \in \mathcal{K}^{\text{pg}}$, Algorithm 4 is called in Line 17 and successfully returns x_{k+1} and $\text{flag}_k^{\text{pg}}$. Moreover, the value of $\text{flag}_k^{\text{pg}}$ indicates whether $k \in \mathcal{K}_{\downarrow}^{\text{pg}}$ or $k \in \mathcal{K}_{\rightarrow}^{\text{pg}}$, and for these respective cases the following properties hold:*

(i) If $k \in \mathcal{K}_{\rightarrow}^{\text{pg}}$, then $\alpha_{k+1} = \alpha_k$ and

$$f(x_{k+1}) + r(x_{k+1}) \leq f(x_k) + r(x_k) - \tfrac{\eta\varphi^2}{\alpha_k}(\chi_k^{\text{pg}})^2. \tag{16}$$

(ii) If $k \in \mathcal{K}_{\downarrow}^{\text{pg}}$, then $\alpha_{k+1} = \xi\alpha_k$ and $f(x_{k+1}) + r(x_{k+1}) < f(x_k) + r(x_k)$.

Proof. Since $k \in \mathcal{K}^{\text{pg}}$, we know that the condition tested in Line 8 of Algorithm 1 must not hold, meaning that $\chi_k^{\text{pg}} > \chi_k^{\text{cg}}$. Combining this observation with Line 16 of Algorithm 1 shows that the set $\mathcal{I}_k$ defined in Line 16 satisfies

$$\|P_{\mathcal{I}_k}(s_k)\|_2 = \|[s_k]_{\mathcal{I}_k}\|_2 \geq \varphi \chi_k^{\text{pg}} > 0. \tag{17}$$

Combining this result with Lemma 2.2 (using $\mathcal{I} = \mathcal{I}_k$, $\overline{x} = x_k$, and $\overline{\alpha} = \alpha_k$) yields

$$D_{f+r}(x_k; P_{\mathcal{I}_k}(s_k)) \leq -\tfrac{1}{\alpha_k}\|P_{\mathcal{I}_k}(s_k)\|_2^2 < 0. \tag{18}$$

It is possible that Algorithm 4 terminates in Line 69 because the inequality in Line 65 does not hold for $j = 0$. In this case, Algorithm 4 successfully returns $x_{k+1} = y_0 = x_k + P_{\mathcal{I}_k}(s_k)$ and $\text{flag}_k^{\text{pg}} = \texttt{same\_}\alpha$, also indicating that $k \in \mathcal{K}_{\rightarrow}^{\text{pg}}$. Since the while-loop in Line 65 terminates with $j = 0$, we can conclude that

$$f(x_{k+1}) + r(x_{k+1}) \equiv f(y_0) + r(y_0) \leq f(x_k) + r(x_k) - \tfrac{\eta}{\alpha_k}\|P_{\mathcal{I}_k}(s_k)\|_2^2. \tag{19}$$

Combining this inequality with (17) shows that (16) holds. Finally, since $\text{flag}_k^{\text{pg}} = \texttt{same\_}\alpha$, it follows from Line 21 that $\alpha_{k+1} = \alpha_k$, completing the proof in this case.

It remains to consider the case when Algorithm 4 is unable to terminate in Line 69 because the inequality in Line 65 holds for $j = 0$. In this case, it follows from (18) and standard results for a backtracking Armijo line search that, for all sufficiently large j, the vector $y_j \leftarrow x_k + \xi^j P_{\mathcal{I}_k}(s_k)$ defined in Line 66 of Algorithm 4 satisfies

$$\begin{aligned} f(y_j) + r(y_j) &\leq f(x_k) + r(x_k) + \eta\xi^j D_{f+r}(x_k; P_{\mathcal{I}_k}(s_k)) \\ &\leq f(x_k) + r(x_k) - \eta\xi^j \tfrac{1}{\alpha_k}\|P_{\mathcal{I}_k}(s_k)\|_2^2. \end{aligned} \tag{20}$$

This inequality shows that the while loop starting in Line 65 of Algorithm 4 will terminate finitely, and thus Algorithm 4 successfully returns $x_{k+1} = y_j = x_k + \xi^j P_{\mathcal{I}_k}(s_k)$ for some $j > 0$ and $\text{flag}_k^{\text{pg}} = \texttt{decrease\_}\alpha$, also indicating that $k \in \mathcal{K}_{\downarrow}^{\text{pg}}$. Combining (20), $y_j = x_{k+1}$, and (18) proves that $f(x_{k+1}) + r(x_{k+1}) < f(x_k) + r(x_k)$, as claimed. Finally, since $\text{flag}_k^{\text{pg}} = \texttt{decrease\_}\alpha$, we see in Line 19 that $\alpha_{k+1} = \xi\alpha_k$. □

Next, we prove that the PG parameter remains bounded away from zero.

Lemma 4.3. *The PG parameter sequence generated by Algorithm 1 satisfies*

$$1 \geq \alpha_k \geq \alpha_{\min} := \min\left\{\alpha_0, \tfrac{2\xi(1-\eta)}{L}\right\} > 0 \quad \textit{for all } k \in \mathbb{N}. \tag{21}$$

Moreover, a bound on the number of times the PG parameter is decreased is given by

$$|\mathcal{K}_{\downarrow}^{\text{pg}}| \leq c_{\downarrow}^{\alpha} := \max\left\{0, \left\lceil \log\left(\tfrac{\alpha_0 L}{2(1-\eta)}\right) / \log(\xi^{-1}) \right\rceil\right\}. \tag{22}$$

Proof. We first prove (21). Since $\alpha_0 \in (0, 1]$ in Line 3 and $\alpha_{k+1} \leq \alpha_k$ for all $k \in \mathbb{N}$, we need only prove the lower bound on α_k in (21). With that goal in mind, for the purpose of obtaining a contradiction, suppose that there exists an iteration k satisfying $\alpha_k \leq 2(1-\eta)/L < 2/L$, with the latter inequality holding since $\eta \in (0, 1)$.

First suppose that $k \in \mathcal{K}^{\text{pg}}$. With $y_0 = x_k + P_{\mathcal{I}_k}(s_k)$ as defined in Line 64 of Algorithm 4, it follows from Lemma 2.3 with $\overline{x} = x_k$, $\overline{\alpha} = \alpha_k$, and $s(\overline{x}, \overline{\alpha}) = s_k$ that

$$\begin{aligned} f(y_0) + r(y_0) &\leq f(x_k) + r(x_k) - (\tfrac{1}{\alpha_k} - \tfrac{L}{2})\|P_{\mathcal{I}}(s_k)\|_2^2 \\ &\leq f(x_k) + r(x_k) - \left(\tfrac{1}{\alpha_k} - \tfrac{2(1-\eta)}{2\alpha_k}\right)\|P_{\mathcal{I}}(s_k)\|_2^2 \\ &= f(x_k) + r(x_k) - \tfrac{\eta}{\alpha_k}\|P_{\mathcal{I}}(s_k)\|_2^2. \end{aligned}$$

This inequality implies that the condition checked in Line 65 for $j = 0$ will not hold, meaning that $j = 0$ when Line 68 is reached so that $\text{flag}_k^{\text{pg}} \leftarrow \texttt{same\_}\alpha$ in Line 69. Thus, when Line 18 in Algorithm 1 is reached, the update $\alpha_{k+1} \leftarrow \alpha_k$ will take place. Second, if $k \in \mathcal{K}^{\text{cg}}$, then Algorithm 1 sets $\alpha_{k+1} \leftarrow \alpha_k$. To summarize, anytime $\alpha_k \leq 2(1-\eta)/L$, the update $\alpha_{k+1} \leftarrow \alpha_k$ takes place. Combining this property with the fact that

when the PG parameter is decreased the update $\alpha_{k+1} \leftarrow \xi\alpha_k$ is used (see Line 19 in Algorithm 1), shows that (21) holds.

We now prove (22). Let us observe from the first paragraph in this proof that if $\alpha_0 \leq 2(1-\eta)/L$ then $|\mathcal{K}_{\downarrow}^{\text{pg}}| = 0$, which verifies that (22) holds. Therefore, for the remainder of the proof, suppose that $\alpha_0 > 2(1-\eta)/L$. Combining this bound with the fact that when the PG parameter is decreased the update $\alpha_{k+1} \leftarrow \xi\alpha_k$ is used, we can see that an upper bound on $|\mathcal{K}_{\downarrow}^{\text{pg}}|$ is the smallest integer ℓ such that $\alpha_0\xi^\ell \leq 2(1-\eta)/L$. Solving this inequality for ℓ shows that the result in (22) holds. □

We now switch our attention to iterations in $\mathcal{K}^{\text{cg}}$. The next result establishes that Algorithm 2 is well posed, and that the direction d_k that results from it when called by Algorithm 1 satisfies a certain descent property.

Lemma 4.4. *For each $k \in \mathcal{K}^{\text{cg}}$, Algorithm 2 is well posed. Moreover, the resulting direction $\bar{d}_k$, which is used to compute d_k in Line 12, guarantees that d_k satisfies*

(i) $\nabla_{\mathcal{I}_k}(f+r)(x_k)^T[d_k]_{\mathcal{I}_k} \leq -\frac{1}{\mu_{\max}}\|\nabla_{\mathcal{I}_k}(f+r)(x_k)\|_2^2 < 0$, *and*

(ii) $\|d_k\|_2 \leq (2/\mu_{\min})\|\nabla_{\mathcal{I}_k}(f+r)(x_k)\|_2$

where $\mathcal{I}_k \subseteq \mathcal{I}_k^{\text{cg}}$ is the set in Line 9 used as an input to Algorithm 2 in Line 13.

Proof. Since $k \in \mathcal{K}^{\text{cg}}$, Algorithm 2 is called in Line 11 with input $\mathcal{I}_k$ defined in Line 9. We first prove that $g_k = \nabla_{\mathcal{I}_k}(f+r)(x_k)$, as defined in Line 10, is nonzero. For a proof by contradiction, suppose that $g_k = 0$ so that $\nabla_{\mathcal{G}_i}(f+r)(x_k) = 0$ for all i such that $\mathcal{G}_i \subseteq \mathcal{I}_k$. Consider arbitrary such i. Note that $[x_k]_{\mathcal{G}_i} \neq 0$ and $[x_k+s_k]_{\mathcal{G}_i} \neq 0$ since $\mathcal{G}_i \subseteq \mathcal{I}_k \subseteq \mathcal{I}_k^{\text{cg}}$ (see Line 9) and by how $\mathcal{I}_k^{\text{cg}}$ is defined. This allows us to conclude from Lemma 2.4 that $[s_k]_{\mathcal{G}_i} = 0$, i.e., that $[s_k]_{\mathcal{I}_k} = 0$ since i with $\mathcal{G}_i \subseteq \mathcal{I}_k$ was arbitrary. This fact and Line 9 yields $\chi_k^{\text{cg}} = 0$, but since the inequality in Line 8 must hold, we also have $\chi_k^{\text{pg}} = 0$. This contradicts Assumption 4.1, thus establishing that $g_k \neq 0$. Now, it follows from Lines 10, 12, 29, and 28, $g_k \neq 0$, and Assumption 4.2 that

$$\begin{aligned}\nabla_{\mathcal{I}_k}(f+r)(x_k)^T[d_k]_{\mathcal{I}_k} \equiv g_k^T\bar{d}_k &\leq g_k^T d_k^R = -\beta_k\|g_k\|_2^2 \\ &= -\|g_k\|_2^4/(g_k^T H_k g_k) \leq -\tfrac{1}{\mu_{\max}}\|g_k\|_2^2.\end{aligned}$$

The result in (i) follows from this inequality and $g_k = \nabla_{\mathcal{I}_k}(f+r)(x_k) \neq 0$.

Part (ii) is precisely [5, Lemma 3.8] under our Assumption 4.2 since our conditions placed upon the step d_k are exactly the same as those used in [5]. □

The next lemma shows that, for $k \in \mathcal{K}^{\text{cg}}$, a local Lipschitz property holds along a certain portion of the search path defined by the reduced-space Newton-CG direction.

Lemma 4.5. *Let $k \in \mathcal{K}^{\text{cg}}$ so that $\mathcal{I}_k$ is computed in Line 9. The following hold:*

(i) The constant $\theta \in (0, \pi/2)$ and index set $\mathcal{I}_k$ passed into Algorithm 3 satisfy, for each i such that $\mathcal{G}_i \subseteq \mathcal{I}_k$ with $\rho_{k,i}$ computed in (11) and $\bar{\rho}_{k,i}$ computed in Line 36, the following conditions:

- *(a)* $\|[x_k+s_k]_{\mathcal{G}_i}\|_2 \neq 0$,
- *(b)* $\|[x_k]_{\mathcal{G}_i}\|_2 \geq \rho_{k,i} \geq \bar{\rho}_{k,i} \geq \sin(\theta)\rho_{k,i} > 0$, *and*
- *(c)* $\|[x_k]_{\mathcal{G}_i}\|_2 - \bar{\rho}_{k,i} \geq \kappa_2(1-\sin(\theta))\|\nabla_{\mathcal{I}_k}(f+r)(x_k)\|_2^p$.

(ii) For all step sizes $\beta \in [0, \tau_k)$ with τ_k computed in Line 43, it holds, with

$$\lambda_{\max} := \max\{\lambda_1, \lambda_2, \ldots, \lambda_{n_{\mathcal{G}}}\} \quad \text{and} \quad \rho_{k,\min} := \min_i\{\rho_{k,i} : \mathcal{G}_i \subseteq \mathcal{I}_k\} \tag{23}$$

that $\|\nabla_{\mathcal{I}_k}(f+r)(x_k) - \nabla_{\mathcal{I}_k}(f+r)(x_k+\beta d_k)\|_2 \leq \beta\big(L + \frac{\lambda_{\max}}{\rho_{k,\min}}\big)\|[d_k]_{\mathcal{I}_k}\|_2$.

Proof. We first prove part (i). Consider arbitrary i with $\mathcal{G}_i \subseteq \mathcal{I}_k$, where $\mathcal{I}_k \subseteq \mathcal{I}_k^{\rm cg}$ is passed into Algorithm 3 and constructed to satisfy the condition in Line 9. Part (a) follows from $\mathcal{I}_k^{\rm cg} \subseteq \bar{\mathcal{I}}_k^{\rm cg}$ and the definition of $\bar{\mathcal{I}}_k^{\rm cg}$ in (8). The first inequality in part (b) follows from $\mathcal{I}_k^{\rm cg} \subseteq \bar{\mathcal{I}}_k^{\rm cg}$, and how $\mathcal{I}_k^{\rm cg}$, $\mathcal{I}_k^{\rm small}$, and $\bar{\mathcal{I}}_k^{\rm cg}$ are defined. The second inequality in (b) follows from how $\bar{\rho}_{k,i}$ is defined in Line 36. The third inequality in (b) follows from Line 36 and the first inequality in (b). To complete the proof for part (b), we must prove that $\rho_{k,i} > 0$. For a proof by contradiction, assume that $\rho_{k,i} = 0$, which by (11) means that $\|\nabla_{\mathcal{I}_k^{\rm cg}}(f+r)(x_k)\|_2 = 0$. It follows from this fact that each i with $\mathcal{G}_i \subseteq \mathcal{I}_k \subseteq \mathcal{I}_k^{\rm cg}$ satisfies $\|\nabla_{\mathcal{G}_i}(f+r)(x_k)\|_2 = 0$, which in light of Lemma 2.4 (using $\overline{x} = x_k$, $\overline{\alpha} = \alpha_k$, and $s(\overline{x}, \overline{\alpha}) = s_k$) and the definition of $\mathcal{I}_k^{\rm cg}$ implies that $\|[s_k]_{\mathcal{G}_i}\|_2 = 0$ for each $\mathcal{G}_i \subseteq \mathcal{I}_k$, i.e., that $\|[s_k]_{\mathcal{I}_k}\|_2 = 0$. It now follows from Line 9 that $\chi_k^{\rm cg} = 0$, which combined with the inequality in Line 8 shows that $\chi_k^{\rm pg} = 0$. Since we have reached a contradiction to Assumption 4.1, we must conclude that $\rho_{k,i} > 0$, as claimed. Finally, we aim to prove part (c). It follows from Line 36, $\theta \in (0, \pi/2)$, part (b), (11), and the fact that $\mathcal{I}_k \subseteq \mathcal{I}_k^{\rm cg}$ that

$$\begin{aligned}\|[x_k]_{\mathcal{G}_i}\|_2 - \bar{\rho}_{k,i} &\geq \|[x_k]_{\mathcal{G}_i}\|_2 - \sin(\theta)\|[x_k]_{\mathcal{G}_i}\|_2 = (1-\sin(\theta))\|[x_k]_{\mathcal{G}_i}\|_2 \\ &\geq (1-\sin(\theta))\rho_{k,i} \geq \kappa_2(1-\sin(\theta))\|\nabla_{\mathcal{I}_k^{\rm cg}}(f+r)(x_k)\|_2^p \\ &\geq \kappa_2(1-\sin(\theta))\|\nabla_{\mathcal{I}_k}(f+r)(x_k)\|_2^p,\end{aligned}$$

which completes the proof of part (c).

To prove part (ii), let $\beta \in [0, \tau_k)$. It follows from part (i) and the definition of τ_k in Line 43 that every point on the segment that connects $[x_k]_{\mathcal{G}_i}$ to $[x_k + \beta d_k]_{\mathcal{G}_i}$ is outside of the ball in $\mathbb{R}^{|\mathcal{G}_i|}$ centered at zero of radius $\bar{\rho}_{k,i} > 0$. This means that both $\|[x_k]_{\mathcal{G}_i}\| \geq \bar{\rho}_{k,i}$ and $\|[x_k + \beta d_k]_{\mathcal{G}_i}\| \geq \bar{\rho}_{k,i}$. It now follows that

$$\begin{aligned}&\|\nabla_{\mathcal{G}_i} r(x_k) - \nabla_{\mathcal{G}_i} r(x_k + \beta d_k)\|_2 \\ &= \lambda_i \left\| \frac{[x_k]_{\mathcal{G}_i}}{\|[x_k]_{\mathcal{G}_i}\|_2} - \frac{[x_k + \beta d_k]_{\mathcal{G}_i}}{\|[x_k + \beta d_k]_{\mathcal{G}_i}\|_2} \right\|_2 = \frac{\lambda_i}{\bar{\rho}_{k,i}} \left\| \frac{\bar{\rho}_{k,i}[x_k]_{\mathcal{G}_i}}{\|[x_k]_{\mathcal{G}_i}\|_2} - \frac{\bar{\rho}_{k,i}[x_k + \beta d_k]_{\mathcal{G}_i}}{\|[x_k + \beta d_k]_{\mathcal{G}_i}\|_2} \right\|_2 \\ &\leq \frac{\lambda_i}{\bar{\rho}_{k,i}} \|[x_k]_{\mathcal{G}_i} - [x_k + \beta d_k]_{\mathcal{G}_i}\|_2 = \frac{\lambda_i \beta}{\bar{\rho}_{k,i}} \|[d_k]_{\mathcal{G}_i}\|_2,\end{aligned} \tag{24}$$

where the (only) inequality follows from the nonexpansive property of the projection (of $[x_k]_{\mathcal{G}_i}$ and $[x_k + \beta d_k]_{\mathcal{G}_i}$) onto the ball of radius $\bar{\rho}_{k,i}$. From (24) we have

$$\begin{aligned}&\|\nabla_{\mathcal{I}_k} r(x_k) - \nabla_{\mathcal{I}_k} r(x_k + \beta d_k)\|_2^2 \\ &= \sum_{i:\mathcal{G}_i \subseteq \mathcal{I}_k} \|\nabla_{\mathcal{G}_i} r(x_k) - \nabla_{\mathcal{G}_i} r(x_k + \beta d_k)\|_2^2 \leq \beta^2 \sum_{i:\mathcal{G}_i \subseteq \mathcal{I}_k} \frac{\lambda_i^2}{\bar{\rho}_{k,i}^2} \|[d_k]_{\mathcal{G}_i}\|_2^2 \\ &\leq \frac{\beta^2 \lambda_{\max}^2}{\rho_{k,\min}^2} \sum_{i:\mathcal{G}_i \subseteq \mathcal{I}_k} \|[d_k]_{\mathcal{G}_i}\|_2^2 = \frac{\beta^2 \lambda_{\max}^2}{\rho_{k,\min}^2} \|[d_k]_{\mathcal{I}_k}\|_2^2.\end{aligned} \tag{25}$$

It follows from Assumption 1.1, $[d_k]_{\mathcal{I}_k^c} = 0$, the triangle inequality, and (25) that

$$\begin{aligned}&\|\nabla_{\mathcal{I}_k}(f+r)(x_k) - \nabla_{\mathcal{I}_k}(f+r)(x_k + \beta d_k)\|_2 \\ &\leq \|\nabla_{\mathcal{I}_k} f(x_k) - \nabla_{\mathcal{I}_k} f(x_k + \beta d_k)\|_2 + \|\nabla_{\mathcal{I}_k} r(x_k) - \nabla_{\mathcal{I}_k} r(x_k + \beta d_k)\|_2 \\ &\leq L\beta\|d_k\|_2 + \left(\beta \tfrac{\lambda_{\max}}{\rho_{k,\min}}\right) \|[d_k]_{\mathcal{I}_k}\|_2 = \beta \left(L + \tfrac{\lambda_{\max}}{\rho_{k,\min}}\right) \|[d_k]_{\mathcal{I}_k}\|_2,\end{aligned}$$

which completes the proof. □

We now show that Algorithm 4 is well posed and that the new iterate it produces satisfies a decrease property that will be used in the final complexity result.

Lemma 4.6. *For each $k \in \mathcal{K}^{\rm cg}$, Algorithm 3 is called in Line 13 and successfully returns x_{k+1} and $\text{flag}_k^{\rm cg}$. Moreover, the value of $\text{flag}_k^{\rm cg}$ indicates whether $k \in \mathcal{K}_0^{\rm cg}$ or $k \in \mathcal{K}_{\rm sd}^{\rm cg}$, and for these respective cases the following properties hold:*

(i) If $k \in \mathcal{K}_0^{\mathrm{cg}}$, then $f(x_{k+1}) + r(x_{k+1}) \leq f(x_k) + r(x_k)$ and x_{k+1} has at least one additional block of zeros compared to x_k.

(ii) If $k \in \mathcal{K}_{\mathrm{sd}}^{\mathrm{cg}}$, then

$$f(x_{k+1}) + r(x_{k+1}) \leq f(x_k) + r(x_k) - \min\{c_1(\chi_k^{\mathrm{cg}})^{1+p}, c_2(\chi_k^{\mathrm{cg}})^{2+p}\} \tag{26}$$

where

$$\begin{aligned} c_1 &:= \frac{\eta\xi\mu_{\min}\kappa_2\big(1-\sin(\theta)\big)\varphi^{1+p}}{2\mu_{\max}} > 0 \quad \textit{and} \\ c_2 &:= \frac{\kappa_2\mu_{\min}^2\xi\eta(1-\eta)\varphi^{2+p}}{2\mu_{\max}^2\big(L\kappa_2(L_f+\lambda_{\max}\sqrt{n_{\mathcal{G}}})^p + \lambda_{\max}\big)} > 0. \end{aligned} \tag{27}$$

Proof. Throughout, we use $F := f + r$. It is possible that Algorithm 3 successfully terminates in Line 50, in which case it follows from Line 50 and Line 49 that the returned x_{k+1} and $\mathrm{flag}_k^{\mathrm{cg}}$ satisfy $F(x_{k+1}) \leq F(x_k)$ and $\mathrm{flag}_k^{\mathrm{cg}} = \texttt{new\_zero}$, indicating that $k \in \mathcal{K}_0^{\mathrm{cg}}$. Moreover, upon termination, the value j satisfies $\xi^j \geq \tau_k$ (see Line 44), which combined with Line 47 shows that at least one additional group of variables has become zero at x_{k+1}. This proves that part (i) holds.

Next, suppose that Algorithm 3 does not terminate in Line 50. Observe from the definition of τ_k in Line 43 that $\tau_k > 0$ (this follows from Lemma 4.5(i) and the definition of $\bar{\rho}_{k,i}$). Therefore, it follows that the **while** loop starting in Line 44 will terminate with the smallest nonnegative integer $\bar{j}$ such that $\xi^{\bar{j}} < \tau_k$, and the **loop** in Line 54 will begin with $j = \bar{j}$. We now claim that the condition in Line 56 used to determine termination of the **loop** is satisfied for all $j \geq \bar{j}$ such that

$$\xi^j \in \left[0, \frac{2(\eta-1)\nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k}}{(L+\lambda_{\max}/\rho_{k,\min})\|[d_k]_{\mathcal{I}_k}\|_2^2}\right] \subset [0, \tau_k). \tag{28}$$

To see that this claim holds, we can use the integral form of Taylor's Theorem and Lemma 4.5(ii) (using the fact that $\gamma\xi^j \in [0, \tau_k)$ for all $\gamma \in [0,1]$) to obtain

$$\begin{aligned} &|F(x_k + \xi^j d_k) - F(x_k) - \xi^j \nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k}| \\ &\quad \leq \left| \int_0^1 \xi^j [d_k]_{\mathcal{I}_k}^T \big(\nabla_{\mathcal{I}_k}F(x_k + \gamma\xi^j d_k) - \nabla_{\mathcal{I}_k}F(x_k)\big)\mathrm{d}\gamma \right| \\ &\quad \leq \xi^j \int_0^1 \|[d_k]_{\mathcal{I}_k}\|_2 \|\nabla_{\mathcal{I}_k}F(x_k + \gamma\xi^j d_k) - \nabla_{\mathcal{I}_k}F(x_k)\big)\|_2 \mathrm{d}\gamma \\ &\quad \leq \xi^{2j}(L + \lambda_{\max}/\rho_{k,\min})\|[d_k]_{\mathcal{I}_k}\|_2^2 \int_0^1 \gamma \mathrm{d}\gamma = \tfrac{1}{2}\xi^{2j}(L+\lambda_{\max}/\rho_{k,\min})\|[d_k]_{\mathcal{I}_k}\|_2^2. \end{aligned}$$

Combining this inequality with (28) yields

$$\begin{aligned} F(x_k + \xi^j d_k) &\leq F(x_k) + \xi^j \nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k} + \tfrac{1}{2}\xi^{2j}(L+\lambda_{\max}/\rho_{k,\min})\|[d_k]_{\mathcal{I}_k}\|_2^2 \\ &= F(x_k) + \xi^j \nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k} + \xi^j(\eta-1)\nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k} \\ &= F(x_k) + \eta\xi^j \nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k}, \end{aligned}$$

which establishes our claim that the inequality in Line 56 holds for all $j \geq \bar{j}$ such that ξ^j satisfies (28). This shows that the **loop** will successfully terminate with $\mathrm{flag}_k^{\mathrm{cg}} = \texttt{suff\_descent}$ (thus indicating that $k \in \mathcal{K}_{\mathrm{sd}}^{\mathrm{cg}}$) and x_{k+1} satisfying

$$F(x_{k+1}) \leq F(x_k) + \eta\xi^{\hat{j}}\nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k} \tag{29}$$

for some $\hat{j}$ satisfying

$$\xi^{\hat{j}} \geq \min\left\{\xi^{\bar{j}}, \frac{2\xi(\eta-1)\nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k}}{(L+\lambda_{\max}/\rho_{k,\min})\|[d_k]_{\mathcal{I}_k}\|_2^2}\right\}$$

$$\geq \min\left\{\xi\tau_k, \frac{2\xi(\eta-1)\nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k}}{(L+\lambda_{\max}/\rho_{k,\min})\|[d_k]_{\mathcal{I}_k}\|_2^2}\right\} \tag{30}$$

where the second inequality follows from the fact that $\bar{j}$ is the *smallest* nonnegative integer such that $\xi^{\bar{j}} < \tau_k$. We now consider two cases.

Case 1: the minimum in (30) is $\xi\tau_k$, from which we may conclude that $\tau_k < \infty$. Using (29) and Lemma 4.4(i) we have that

$$F(x_{k+1}) \leq F(x_k) + \eta\xi^{\hat{j}}\nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k} \leq F(x_k) - \tfrac{\eta\xi}{\mu_{\max}}\tau_k\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^2. \tag{31}$$

We now seek a lower bound on τ_k. Consider i such that $\tau_{k,i} < \infty$ when computed in Algorithm 3. The triangle inequality gives $\bar{\rho}_{k,i} = \|[x_k + \tau_{k,i}d_k]_{\mathcal{G}_i}\|_2 \geq \|[x_k]_{\mathcal{G}_i}\|_2 - \tau_{k,i}\|[d_k]_{\mathcal{G}_i}\|_2$, which together with Lemma 4.5(i)(c) and Lemma 4.4(ii) shows that

$$\begin{aligned}\tau_{k,i} &\geq \frac{\|[x_k]_{\mathcal{G}_i}\|_2 - \bar{\rho}_{k,i}}{\|[d_k]_{\mathcal{G}_i}\|_2} \\ &\geq \frac{\mu_{\min}\kappa_2(1-\sin(\theta))\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^p}{2\|\nabla_{\mathcal{I}_k}F(x_k)\|_2} = \tfrac{1}{2}\mu_{\min}\kappa_2(1-\sin(\theta))\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^{p-1}.\end{aligned}$$

From this, it follows that $\tau_k \geq \frac{1}{2}\mu_{\min}\kappa_2(1-\sin(\theta))\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^{p-1}$. Using this inequality with (31), Lemma 2.4, and the set $\mathcal{I}_k$ from Line 9 shows that

$$\begin{aligned}F(x_{k+1}) &\leq F(x_k) - \frac{\eta\xi\mu_{\min}\kappa_2(1-\sin(\theta))}{2\mu_{\max}}\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^{1+p} \\ &\leq F(x_k) - \frac{\eta\xi\mu_{\min}\kappa_2(1-\sin(\theta))}{2\mu_{\max}}\|[s_k]_{\mathcal{I}_k}\|_2^{1+p} \\ &\leq F(x_k) - \frac{\eta\xi\mu_{\min}\kappa_2\big(1-\sin(\theta)\big)\varphi^{1+p}}{2\mu_{\max}}(\chi_k^{\mathrm{cg}})^{1+p},\end{aligned}$$

thus completing the proof for this case.

Case 2: the minimum in (30) is $\frac{2\xi(\eta-1)\nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k}}{(L+\lambda_{\max}/\rho_{k,\min})\|[d_k]_{\mathcal{I}_k}\|_2^2}$. Combining this fact with (29), (30), Lemma 4.4(i), and Lemma 4.4(ii) shows that

$$\begin{aligned}F(x_{k+1}) &\leq F(x_k) + \eta\xi^{\hat{j}}\nabla_{\mathcal{I}_k}F(x_k)^T[d_k]_{\mathcal{I}_k} \\ &\leq F(x_k) - \frac{2\xi\eta(1-\eta)\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^4}{\mu_{\max}^2(L+\lambda_{\max}/\rho_{k,\min})\|[d_k]_{\mathcal{I}_k}\|_2^2} \\ &\leq F(x_k) - \frac{2\mu_{\min}^2\xi\eta(1-\eta)\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^4}{4\mu_{\max}^2(L+\lambda_{\max}/\rho_{k,\min})\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^2} \\ &= F(x_k) - \frac{\mu_{\min}^2\xi\eta(1-\eta)\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^2}{2\mu_{\max}^2(L+\lambda_{\max}/\rho_{k,\min})}.\end{aligned} \tag{32}$$

It follows from (23), (11), and $\mathcal{I}_k \subseteq \mathcal{I}_k^{\mathrm{cg}}$ that $\rho_{k,\min} \geq \kappa_2\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^p$. Combining this bound with (32) shows that

$$\begin{aligned}F(x_{k+1}) &\leq F(x_k) - \frac{\mu_{\min}^2\xi\eta(1-\eta)\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^2}{2\mu_{\max}^2(L+\lambda_{\max}/\rho_{k,\min})} \\ &\leq F(x_k) - \frac{\mu_{\min}^2\xi\eta(1-\eta)\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^2}{2\mu_{\max}^2\big(L+\lambda_{\max}/(\kappa_2\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^p)\big)} \\ &= F(x_k) - \frac{\kappa_2\mu_{\min}^2\xi\eta(1-\eta)\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^{2+p}}{2\mu_{\max}^2(L\kappa_2\|\nabla_{\mathcal{I}_k}F(x_k)\|_2^p+\lambda_{\max})}.\end{aligned} \tag{33}$$

Next, we know from Lemma 4.2, Lemma 4.6(i), and equations (31) and (33) that $F(x_k) \leq F(x_0)$ for all $k \in \mathbb{N}$, i.e., $x_k \in \mathcal{L}$ for all $k \in \mathbb{N}$. Combining this fact with the triangle inequality, Assumption 1.1, the definition of r, and (23) gives

$$\begin{aligned}
\|\nabla_{\mathcal{I}_k} F(x_k)\|_2 &\leq \|\nabla_{\mathcal{I}_k} f(x_k)\|_2 + \|\nabla_{\mathcal{I}_k} r(x_k)\|_2 \\
&= \|\nabla_{\mathcal{I}_k} f(x_k)\|_2 + \Big(\sum_{i:\mathcal{G}_i \subseteq \mathcal{I}_k} \|\nabla_{\mathcal{G}_i} r(x_k)\|_2^2 \Big)^{1/2} \\
&\leq L_f + \Big(\sum_{i:\mathcal{G}_i \subseteq \mathcal{I}_k} \|\lambda_i [x_k]_{\mathcal{G}_i} / \|[x_k]_{\mathcal{G}_i}\|_2 \|_2^2 \Big)^{1/2} \\
&= L_f + \Big(\sum_{i:\mathcal{G}_i \subseteq \mathcal{I}_k} \lambda_i^2 \Big)^{1/2} \leq L_f + \Big(\sum_{i:\mathcal{G}_i \subseteq \mathcal{I}_k} \lambda_{\max}^2 \Big)^{1/2} \leq L_f + \lambda_{\max} \sqrt{n_{\mathcal{G}}}.
\end{aligned}$$

Combining this with (33) gives

$$F(x_{k+1}) \leq F(x_k) - \left(\frac{\kappa_2 \mu_{\min}^2 \xi \eta (1-\eta)}{2\mu_{\max}^2 \big(L \kappa_2 (L_f + \lambda_{\max} \sqrt{n_{\mathcal{G}}})^p + \lambda_{\max} \big)} \right) \|\nabla_{\mathcal{I}_k} F(x_k)\|_2^{2+p},$$

which combined with Lemma 2.4 and how the index set $\mathcal{I}_k$ in Line 9 is defined gives

$$\begin{aligned}
F(x_{k+1}) &\leq F(x_k) - \left(\frac{\kappa_2 \mu_{\min}^2 \xi \eta (1-\eta)}{2\mu_{\max}^2 \big(L \kappa_2 (L_f + \lambda_{\max} \sqrt{n_{\mathcal{G}}})^p + \lambda_{\max} \big)} \right) \|[s_k]_{\mathcal{I}_k}\|_2^{2+p} \\
&\leq F(x_k) - \left(\frac{\kappa_2 \mu_{\min}^2 \xi \eta (1-\eta) \varphi^{2+p}}{2\mu_{\max}^2 \big(L \kappa_2 (L_f + \lambda_{\max} \sqrt{n_{\mathcal{G}}})^p + \lambda_{\max} \big)} \right) (\chi_k^{\mathrm{cg}})^{2+p},
\end{aligned}$$

thus completing the proof. □

The result in (26) motivates us to define the following subsets of $\mathcal{K}_{\mathrm{sd}}^{\mathrm{cg}}$:

$$\mathcal{K}_{\mathrm{sd,big}}^{\mathrm{cg}} := \{k \in \mathcal{K}_{\mathrm{sd}}^{\mathrm{cg}} : \chi_k^{\mathrm{cg}} \geq c_1/c_2\} \quad \text{and} \quad \mathcal{K}_{\mathrm{sd,small}}^{\mathrm{cg}} := \mathcal{K}_{\mathrm{sd}}^{\mathrm{cg}} \setminus \mathcal{K}_{\mathrm{sd,big}}^{\mathrm{cg}}. \tag{34}$$

This distinction plays a role in our complexity result. First, we require a lemma.

Lemma 4.7. *The objective function $f + r$ is monotonically decreasing over the sequence of iterates $\{x_k\}$ and $\lim_{k\to\infty} \big(f(x_k) + r(x_k)\big) =: F_{\min} > -\infty$.*

Proof. It follows from Lemma 4.2 and Lemma 4.6 that the objective function is monotonically decreasing over the iterate sequence. The remaining conclusion of the lemma follows from the monotonicity property and Assumption 1.1. □

The main theorem can now be stated. It gives an upper bound on the number of iterations performed by Algorithm 1 before an approximate solution is obtained.

Theorem 4.1. *Let c_1 and c_2 be the constants defined in (27) and let us define $c_3 := \eta\varphi^2/\alpha_0 > 0$. For any $\epsilon > 0$, define $\mathcal{K}_\epsilon := \{k \in \mathbb{N} : \max\{\chi_k^{\mathrm{cg}}, \chi_k^{\mathrm{pg}}\} > \epsilon\}$. Then,*

$$\begin{aligned}
|\mathcal{K}_{\rightarrow}^{\mathrm{pg}} \cap \mathcal{K}_\epsilon| &\leq c_{\mathrm{pg}} \epsilon^{-2} + 1, \\
|\mathcal{K}_{\mathrm{sd,big}}^{\mathrm{cg}} \cap \mathcal{K}_\epsilon| &\leq c_{\mathrm{big}} \epsilon^{-(1+p)} + 1, \quad \textit{and} \\
|\mathcal{K}_{\mathrm{sd,small}}^{\mathrm{cg}} \cap \mathcal{K}_\epsilon| &\leq c_{\mathrm{small}} \epsilon^{-(2+p)} + 1
\end{aligned} \tag{35}$$

where the constants $c_{\rm pg}$, $c_{\rm big}$, *and* $c_{\rm small}$ *are given, respectively, by*

$$\begin{aligned} c_{\rm pg} &:= \big(f(x_0) + r(x_0) - F_{\min}\big)/c_3, \\ c_{\rm big} &:= \big(f(x_0) + r(x_0) - F_{\min}\big)/c_1, \quad \textit{and} \\ c_{\rm small} &:= \big(f(x_0) + r(x_0) - F_{\min}\big)/c_2. \end{aligned} \tag{36}$$

Therefore, if $\epsilon \geq c_1/c_2$, *then*

$$|\mathcal{K}_\epsilon| \leq \big(c_\downarrow^\alpha + c_{\rm pg}\epsilon^{-2} + c_{\rm big}\epsilon^{-(1+p)} + 2\big)(1 + n_{\mathcal{G}}) + n_{\mathcal{G}} \tag{37}$$

where $c_\downarrow^\alpha$ *is defined in* (22)*; otherwise, i.e., if* $\epsilon < c_1/c_2$, *then*

$$|\mathcal{K}_\epsilon| \leq \big(c_\downarrow^\alpha + c_{\rm pg}\epsilon^{-2} + c_{\rm big}\epsilon^{-(1+p)} + c_{\rm small}\epsilon^{-(2+p)} + 3\big)(1 + n_{\mathcal{G}}) + n_{\mathcal{G}}. \tag{38}$$

Proof. Note that the definitions of $\mathcal{K}^{\rm cg}$ and $\mathcal{K}^{\rm pg}$ together with Line 8 show that

$$\chi_k^{\rm cg} \geq \chi_k^{\rm pg} \text{ for } k \in \mathcal{K}^{\rm cg} \quad \text{and} \quad \chi_k^{\rm pg} > \chi_k^{\rm cg} \text{ for } k \in \mathcal{K}^{\rm pg}. \tag{39}$$

Define $\Delta_k := f(x_k)+r(x_k)-\big(f(x_{k+1})+r(x_{k+1})\big)$ and $m_k := \max\{\chi_k^{\rm pg}, \chi_k^{\rm cg}\}$. Using Lemma 4.2(i), Lemma 4.3, Lemma 4.6(ii), the definitions of c_3 and $\mathcal{K}_\epsilon$ in the statement of the theorem, and (39) shows for arbitrary $\bar{k} \in \mathbb{N}$ that

$$\begin{aligned} f(x_0) + r(x_0) - \big(f(x_{\bar{k}+1}) + r(x_{\bar{k}+1})\big) &= \sum_{0\leq k\leq \bar{k}} \Delta_k \\ &\geq \sum_{\substack{k\in\mathcal{K}_{\rightarrow}^{\rm pg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} \Delta_k + \sum_{\substack{k\in\mathcal{K}_{\rm sd,big}^{\rm cg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} \Delta_k + \sum_{\substack{k\in\mathcal{K}_{\rm sd,small}^{\rm cg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} \Delta_k \\ &\geq \sum_{\substack{k\in\mathcal{K}_{\rightarrow}^{\rm pg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} c_3(\chi_k^{\rm pg})^2 + \sum_{\substack{k\in\mathcal{K}_{\rm sd,big}^{\rm cg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} c_1(\chi_k^{\rm cg})^{1+p} + \sum_{\substack{k\in\mathcal{K}_{\rm sd,small}^{\rm cg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} c_2(\chi_k^{\rm cg})^{2+p} \\ &= \sum_{\substack{k\in\mathcal{K}_{\rightarrow}^{\rm pg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} c_3 m_k^2 + \sum_{\substack{k\in\mathcal{K}_{\rm sd,big}^{\rm cg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} c_1 m_k^{1+p} + \sum_{\substack{k\in\mathcal{K}_{\rm sd,small}^{\rm cg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} c_2 m_k^{2+p} \\ &\geq \sum_{\substack{k\in\mathcal{K}_{\rightarrow}^{\rm pg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} c_3\epsilon^2 + \sum_{\substack{k\in\mathcal{K}_{\rm sd,big}^{\rm cg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} c_1\epsilon^{1+p} + \sum_{\substack{k\in\mathcal{K}_{\rm sd,small}^{\rm cg}\cap\mathcal{K}_\epsilon \\ 0\leq k\leq\bar{k}}} c_2\epsilon^{2+p}. \end{aligned}$$

From this inequality, Lemma 4.7, and (36) one finds that (35) follows.

Next, suppose that $\epsilon \geq c_1/c_2$. It then follows from (34) and (39) that $\chi_k^{\rm cg} = \max\{\chi_k^{\rm pg}, \chi_k^{\rm cg}\} > \epsilon \geq c_1/c_2$ for all $k \in \mathcal{K}^{\rm cg}$, which implies that $\mathcal{K}_{\rm sd,small}^{\rm cg} \cap \mathcal{K}_\epsilon = \emptyset$. The result in (37) follows from this observation, (35), (22), and since (by Lemma 4.6(i)) at most $n_{\mathcal{G}}$ iterations in $\mathcal{K}_0^{\rm cg}$ can occur before the first, after the last, or between any two iterations in $\mathcal{K}_\downarrow^{\rm pg} \cup \mathcal{K}_{\rightarrow}^{\rm pg} \cup \mathcal{K}_{\rm sd}^{\rm cg}$.

The final result (38) follows using the same argument as in the previous paragraph, except now $\mathcal{K}_{\rm sd,small}^{\rm cg} \cap \mathcal{K}_\epsilon$ is no longer necessarily empty. □

We see from (38) that, for all sufficiently small ϵ, the worst case complexity result for Algorithm 1 is $\epsilon^{-(2+p)}$, which is worse than the ϵ^{-2} result that holds for the PG method. If one is concerned with such a result, the difference can be made arbitrarily small (for a range of ϵ values typically used in practice) by choosing p sufficiently small. However, as is typical with well-designed second-derivative methods, although the complexity bound is worse, it typically performs better (see Section 5).

4.2 Local convergence

We now consider the local convergence rate of the iterates generated by Algorithm 1. Our analysis is performed under the following additional assumption that will be assumed to hold throughout this section.

Assumption 4.3. *The function f is twice continuously differentiable and strongly convex. It follows that that there exists a unique solution x_* to the optimization problem (1) with optimal support $\mathcal{S}_* := \{i : [x_*]_{\mathcal{G}_i} \neq 0\}$. Moreover, we assume that $\nabla^2 f : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ is Lipschitz continuous in a neighborhood of the solution x_*, and that $f + r$ is nondegenerate at x_* in the sense that $\|[\nabla f(x_*)]_{\mathcal{G}_i}\|_2 < \lambda_i$ for all $i \notin \mathcal{S}_*$.*

Optimality conditions for problem (1) imply that $\|[\nabla f(x_*)]_{\mathcal{G}_i}\|_2 \leq \lambda_i$ for all $i \notin \mathcal{S}_*$. Thus, the final condition in Assumption 4.3 is a strengthening of this fact.

Assumption 4.4. *The following algorithmic choices are made in Algorithm 1:*

(i) The backtracking parameter is chosen to satisfy $\eta \in (0, 1/2)$.

(ii) For all sufficiently large $k \in \mathbb{N}$, $\mathcal{I}_k$ in Line 9/16 is chosen as

$$\mathcal{I}_k = \begin{cases} \mathcal{I}_k^{\mathrm{cg}} & \text{if } k \in \mathcal{K}^{\mathrm{cg}}, \\ \mathcal{I}_k^{\mathrm{pg}} & \text{if } k \in \mathcal{K}^{\mathrm{pg}}. \end{cases} \tag{40}$$

(iii) For all sufficiently large $k \in \mathcal{K}^{\mathrm{cg}}$, $H_k = \nabla^2_{\mathcal{I}_k\mathcal{I}_k}(f + r)(x_k)$ is chosen in Line 10.

The next result establishes that the iterate sequence converges to x_*.

Theorem 4.2. *The iterate sequence $\{x_k\}$ generated by Algorithm 1 satisfies*

$$\lim_{k\to\infty} x_k = x_* \quad \text{and} \quad \lim_{k\to\infty} \max\{\chi_k^{\mathrm{pg}}, \chi_k^{\mathrm{cg}}\} = 0.$$

Proof. Theorem 4.1 gives $\lim_{k\to\infty} \max\{\chi_k^{\mathrm{pg}}, \chi_k^{\mathrm{cg}}\} = 0$. Since $\{x_k\}$ is bounded due to monotonicity of $\{f(x_k) + r(x_k)\}$ (see Lemma 4.7) and Assumption 4.3, there exists an infinite $\mathcal{K} \subseteq \mathbb{N}$ and $\hat{x}$ so that $\lim_{k\in\mathcal{K},k\to\infty} x_k = \hat{x}$. It follows from Lemma 4.1 and Lemma 4.3 that $\hat{x}$ is a solution to problem (1), but with Assumption 4.3 this means that $\hat{x} = x_*$, so $\lim_{k\in\mathcal{K},k\to\infty} x_k = x_*$. The fact that the *entire* sequence $\{x_k\}$ converges to x_* follows from this fact, Assumption 4.3, and monotonicity of $\{f(x_k) + r(x_k)\}$. □

We now show for groups whose variables are all equal to zero at the solution that the PG step will eventually predict them to be zero.

Lemma 4.8. *For all $i \notin \mathcal{S}_*$ and sufficiently large k, it holds that $[x_k + s_k]_{\mathcal{G}_i} = 0$.*

Proof. First note that Lemma 4.3 and the update strategy for $\{\alpha_k\}$ in Algorithm 1 ensure that there exists $\bar{k}_1$ such that $\alpha_k = \alpha_* > 0$ for all $k \geq \bar{k}_1$. Now, let $i \notin \mathcal{S}_*$ so that $[x_*]_{\mathcal{G}_i} = 0$. It follows from Assumption 4.3 that

$$\frac{\alpha_*\lambda_i}{\|[x_* - \alpha_*\nabla f(x_*)]_{\mathcal{G}_i}\|_2} = \frac{\lambda_i}{\|[\nabla f(x_*)]_{\mathcal{G}_i}\|_2} > 1.$$

Combining this with Theorem 4.2, $\alpha_k = \alpha_* > 0$ for all $k \geq \bar{k}_1$, and Assumption 1.1 shows that there exists a $\bar{k}_2 \geq \bar{k}_1$ such that $1 - \alpha_k\lambda_i/\|[x_k - \alpha_k\nabla f(x_k)]_{\mathcal{G}_i}\|_2 < 0$ for all $k \geq \bar{k}_2$. Using this fact with (4) and (5) shows that $[x_k + s_k]_{\mathcal{G}_i} = 0$ for all $k \geq \bar{k}_2$. This completes the proof since the choice $i \notin \mathcal{S}_*$ was arbitrary and $n_{\mathcal{G}}$ is finite. □

We now show that, eventually, the set $\mathcal{S}_*$ determines the sets $\mathcal{I}_k^{\mathrm{pg}}$ and $\mathcal{I}_k^{\mathrm{cg}}$.

Lemma 4.9. *For all sufficiently large k, it holds that*

$$\mathcal{I}_k^{\mathrm{pg}} \equiv \{j \in \mathcal{G}_i : i \notin \mathcal{S}_*\} \quad \text{and} \quad \mathcal{I}_k^{\mathrm{cg}} \equiv \{j \in \mathcal{G}_i : i \in \mathcal{S}_*\}$$

where the sets $\mathcal{I}_k^{\mathrm{pg}}$ and $\mathcal{I}_k^{\mathrm{cg}}$ are defined in (10).

Proof. Let $\bar{k}_1$ be large enough so that the conclusion of Lemma 4.8 holds, i.e., if $k \geq \bar{k}_1$ and $i \notin \mathcal{S}_*$, then $[x_k + s_k]_{\mathcal{G}_i} = 0$. Together with (8), this shows that $\mathcal{G}_i \cap \bar{\mathcal{I}}_k^{\text{cg}} = \emptyset$ for all $k \geq \bar{k}_1$ and $i \notin \mathcal{S}_*$, and thus $\mathcal{G}_i \subseteq \mathcal{I}_k^{\text{pg}}$ (see (10)) for all $k \geq \bar{k}_1$ and $i \notin \mathcal{S}_*$. In other words, it holds that $\{j \in \mathcal{G}_i : i \notin \mathcal{S}_*\} \subseteq \mathcal{I}_k^{\text{pg}}$ for all $k \geq \bar{k}_1$.

Next, we prove that there exists $\bar{k}_2$ such that $\mathcal{I}_k^{\text{pg}} \subseteq \{j \in \mathcal{G}_i : i \notin \mathcal{S}_*\}$ for all $k \geq \bar{k}_2$. For a proof by contradiction, suppose that there exists an infinite subsequence $\mathcal{K} \subseteq \mathbb{N}$ and group index $\bar{i}$ such that $\mathcal{G}_{\bar{i}} \subseteq \mathcal{I}_k^{\text{pg}}$ and $\bar{i} \in \mathcal{S}_*$ for all $k \in \mathcal{K}$. Since $\mathcal{G}_{\bar{i}} \subseteq \mathcal{I}_k^{\text{pg}}$ for all $k \in \mathcal{K}$, it follows from (8), (9), and (10) that at least one of

$$[x_k]_{\mathcal{G}_{\bar{i}}} = 0, \quad [x_k + s_k]_{\mathcal{G}_{\bar{i}}} = 0, \quad \|[x_k]_{\mathcal{G}_{\bar{i}}}\|_2 < \kappa_1 \|\nabla_{\mathcal{G}_{\bar{i}}}(f + r)(x_k)\|_2 \quad \text{or} \tag{41}$$

$$\|[x_k]_{\mathcal{G}_{\bar{i}}}\|_2 < \kappa_2 \|\nabla_{\bar{\mathcal{I}}_k^{\text{cg}}}(f + r)(x_k)\|_2^p \tag{42}$$

holds for all $k \in \mathcal{K}$. However, since $\bar{i} \in \mathcal{S}_*$, it follows from Theorem 4.2 that the first condition in (41) does not hold for all sufficiently large $k \in \mathcal{K}$. Also, it follows from Theorem 4.2, the facts that $\chi_k^{\text{pg}} \equiv \|[s_k]_{\mathcal{I}_k^{\text{pg}}}\|_2$ and $\chi_k^{\text{cg}} \equiv \|[s_k]_{\mathcal{I}_k^{\text{cg}}}\|_2$, and the fact that $\mathcal{I}_k^{\text{cg}} \cup \mathcal{I}_k^{\text{pg}} = \{1, \ldots, n\}$ that $\lim_{k \to \infty} \|s_k\|_2 = 0$, which combined with $\bar{i} \in \mathcal{S}_*$ proves that $[x_k + s_k]_{\mathcal{G}_{\bar{i}}} \neq 0$ for all sufficiently large k. Hence, the second condition in (41) does not hold for all sufficiently large $k \in \mathcal{K}$. Next, from the optimality conditions for problem (1), the fact that $\bar{i} \in \mathcal{S}_*$, Theorem 4.2, Assumption 1.1, and the fact that $f + r$ is differentiable over the variables in $\mathcal{G}_{\bar{i}}$ for sufficiently large k that we have $\lim_{k \to \infty} \|\nabla_{\mathcal{G}_{\bar{i}}}(f + r)(x_k)\|_2 = 0$. This limit, $[x_*]_{\mathcal{G}_{\bar{i}}} \neq 0$, and Theorem 4.2 show that $\|[x_k]_{\mathcal{G}_{\bar{i}}}\|_2 \geq \kappa_1 \|\nabla_{\mathcal{G}_{\bar{i}}}(f + r)(x_k)\|_2$ for all sufficiently large k, meaning that the third condition in (41) does not hold for all sufficiently large $k \in \mathcal{K}$. Therefore, we must conclude that the inequality in (42) holds for all sufficiently large $k \in \mathcal{K}$. Combining this with $\bar{i} \in \mathcal{S}_*$ shows that there exists $\epsilon > 0$ such that

$$\|\nabla_{\bar{\mathcal{I}}_k^{\text{cg}}}(f + r)(x_k)\|_2 \geq \epsilon > 0 \quad \text{for all sufficiently large } k \in \mathcal{K}, \tag{43}$$

which in particular shows that $\bar{\mathcal{I}}_k^{\text{cg}} \neq \emptyset$ for all sufficiently large $k \in \mathcal{K}$. Since the optimality conditions for problem (1) together with Theorem 4.2, Assumption 1.1, and the fact that $f + r$ is differentiable over the variables in $\mathcal{G}_i$ for sufficiently large k imply that $\lim_{k \to \infty} \|\nabla_{\mathcal{G}_i}(f + r)(x_k)\|_2 = 0$ for all $i \in \mathcal{S}_*$, we must conclude from (43) that, for all sufficiently large $k \in \mathcal{K}$, there exists an $i_k \notin \mathcal{S}_*$ such that $\mathcal{G}_{i_k} \subseteq \bar{\mathcal{I}}_k^{\text{cg}}$. However, Lemma 4.8 yields $[x_k + s_k]_{\mathcal{G}_{i_k}} = 0$ for all sufficiently large $k \in \mathcal{K}$, which together with (8) shows that $\mathcal{G}_{i_k} \not\subseteq \bar{\mathcal{I}}_k^{\text{cg}}$, which is a contradiction. Therefore, there exists $\bar{k}_2$ such that $\mathcal{I}_k^{\text{pg}} \subseteq \{j \in \mathcal{G}_i : i \notin \mathcal{S}_*\}$ for all $k \geq \bar{k}_2$.

The conclusions of the two previous paragraphs yields $\mathcal{I}_k^{\text{pg}} \equiv \{j \in \mathcal{G}_i : i \notin \mathcal{S}_*\}$ for all sufficiently large k. The final assertion, namely that $\mathcal{I}_k^{\text{cg}} \equiv \{j \in \mathcal{G}_i : i \in \mathcal{S}_*\}$, follows from the fact that $\mathcal{I}_k^{\text{pg}}$ and $\mathcal{I}_k^{\text{cg}}$ partition $\{1, 2, \ldots, n\}$ for every iteration k. □

The next result shows that, for iterations k sufficiently large, the support of x_k agrees with the support of the solution x_*.

Lemma 4.10. *For all sufficiently large k, it holds that*

$$[x_k]_{\mathcal{G}_i} \neq 0 \text{ for all } i \in \mathcal{S}_* \quad \text{and} \quad [x_k]_{\mathcal{G}_i} = 0 \text{ for all } i \notin \mathcal{S}_*.$$

Proof. Theorem 4.2 shows that $[x_k]_{\mathcal{G}_i} \neq 0$ for all sufficiently large k and all $i \in \mathcal{S}_*$, which is the first desired result. Hence, let us proceed by considering arbitrary $i \notin \mathcal{S}_*$. Assumption 4.4(ii), Lemma 4.8, Lemma 4.9, and Lemma 4.3 ensure the existence of an iteration $\bar{k}$ such that, for all $k \geq \bar{k}$, the following hold:

$$\mathcal{G}_i \subseteq \mathcal{I}_k^{\text{pg}}, \quad [x_k + s_k]_{\mathcal{G}_i} = 0, \quad \text{and } \alpha_k = \alpha_{\bar{k}}. \tag{44}$$

We claim that the second desired result follows from (44) if there exists some sufficiently large $\hat{k} \geq \bar{k}$ such that $\hat{k} \in \mathcal{K}^{\text{pg}}$ and $[x_{\hat{k}+1}]_{\mathcal{G}_i} = [x_{\hat{k}} + s_{\hat{k}}]_{\mathcal{G}_i} = 0$. Indeed, since i is an arbitrary element from $\{1, \ldots, n_{\mathcal{G}}\} \setminus \mathcal{S}_*$, $n_{\mathcal{G}}$ is finite, and the second condition in (44) shows that values of the variables in $\mathcal{G}_i$ can only be modified if $k \in \mathcal{K}^{\text{pg}}$, the existence of such $\hat{k}$ along with (44) shows that iteration $\hat{k} \in \mathcal{K}^{\text{pg}}$ sets $[x_{\hat{k}+1}]_{\mathcal{G}_i}$ to zero, and these variables will remain zero for all future iterations.

Let us now show the existence of such $\hat{k} \geq \bar{k}$. We claim that there exists $k \geq \bar{k}$ such that $[x_k]_{\mathcal{G}_i} = 0$. For a proof by contradiction, suppose that $[x_k]_{\mathcal{G}_i} \neq 0$ for all $k \geq \bar{k}$. Combining this with Theorem 4.2, $i \notin \mathcal{S}_*$, and the fact that the variables in $\mathcal{G}_i$ can have their values changed only if $k \in \mathcal{K}^{\text{pg}}$ implies that there exists $\hat{k} \geq \bar{k}$ such that $\hat{k} \in \mathcal{K}^{\text{pg}}$. Now, since $\hat{k} \in \mathcal{K}^{\text{pg}}$ and $\alpha_k = \alpha_{\bar{k}}$ for all $k \geq \bar{k}$, it follows from Algorithm 1 that $\text{flag}^{\text{pg}}_{\hat{k}} = \texttt{same\_}\alpha$ is returned in Line 17. Using this fact, the update used in Line 69, and (44) shows that $[x_{\hat{k}+1}]_{\mathcal{G}_i} = [x_{\hat{k}} + s_{\hat{k}}]_{\mathcal{G}_i} = 0$. □

We require one more lemma that shows that eventually all iterations are in $\mathcal{K}^{\text{cg}}_{\text{sd}}$.

Lemma 4.11. *For all k sufficiently large, it holds that $k \in \mathcal{K}^{\text{cg}}_{\text{sd}}$.*

Proof. We first show that all sufficiently large k are in $\mathcal{K}^{\text{cg}}$. It follows from Lemma 4.9 that $\mathcal{I}^{\text{pg}}_k \equiv \{j \in \mathcal{G}_i : i \notin \mathcal{S}_*\}$ for all sufficiently large k. Combining this with Lemma 4.10 and Lemma 4.8 shows that there exists an iteration $\bar{k}$ such that $[x_k]_{\mathcal{I}^{\text{pg}}_k} = 0$ and $[x_k + s_k]_{\mathcal{I}^{\text{pg}}_k} = 0$ for all $k \geq \bar{k}$, which means that $\chi^{\text{pg}}_k = \|[s_k]_{\mathcal{I}^{\text{pg}}_k}\|_2 = 0$ for all $k \geq \bar{k}$. It follows from this fact, Line 8, and Assumption 4.1 that $k \in \mathcal{K}^{\text{cg}}$ for all $k \geq \bar{k}$. Now, notice that at most $n_{\mathcal{G}} - 1$ iterations from $\bar{k}$ onward can be in $\mathcal{K}^{\text{cg}}_0$ because of Lemma 4.6(i). (Every iteration $k \in \mathcal{K}^{\text{cg}}_0$ fixes at least one new group of variables to zero and if they ever all become zero so that $\mathcal{I}^{\text{cg}}_k = \emptyset$, then the contradiction $k \in \mathcal{K}^{\text{pg}}$ is reached.) Therefore, it follows that all sufficiently large k must be in $\mathcal{K}^{\text{cg}}_{\text{sd}}$. □

We can now state our main local convergence result.

Theorem 4.3. *If in Algorithm 2 we choose either $q \in (1, 2]$, or $q = 1$ and $\{\mu_k\} \to 0$, then $\{x_k\} \to x_*$ at a superlinear rate. In particular, if we choose $q = 2$, then the rate of convergence is quadratic.*

Proof. It follows from Lemma 4.9, Lemma 4.10, and Lemma 4.11 that, for all sufficiently large k, the iterates generated by Algorithm 1 satisfy the recurrence $x_{k+1} = x_k + \xi^{j_k} d_k$, where j_k is the result of the backtracking Armijo line search in Line 56, $\|[x_k]_{\mathcal{I}^{\text{pg}}_k}\|_2 = \|[d_k]_{\mathcal{I}^{\text{pg}}_k}\|_2 = 0$, and $[d_k]_{\mathcal{I}^{\text{cg}}_k} = \bar{d}_k$ with $\bar{d}_k$ computed by Algorithm 2 to satisfy (15). In other words, for all sufficiently large k, we have $[x_k]_{\mathcal{I}^{\text{pg}}_k} = [x_*]_{\mathcal{I}^{\text{pg}}_k} = 0$ and the values of the variables in $\mathcal{I}^{\text{cg}}_k \equiv \{j \in \mathcal{G}_i : i \in \mathcal{S}_*\}$ are updated exactly as those of an inexact Newton method for computing a root of $\nabla_{\mathcal{I}^{\text{cg}}_k}(f + r)$. Since, by Theorem 4.2, we have $\lim_{k\to\infty} x_k = x_*$, the desired conclusions follow under the stated conditions from [11, Theorem 3.3] and noting the well-known result that the unit step size $\xi^{j_k} = 1$ is accepted (asymptotically) by a backtracking Armijo line search when $\eta \in (0, 1/2)$ (see Assumption 4.4) under our assumptions. □

Theorem 4.3 states conditions under which Algorithm 1 yields a superlinear, or even quadratic, rate of local convergence. The neighborhood about x_* in which such a rate will be achieved, and the explicit constants in the convergence rate that will be achieved, depend as usual on magnitudes of a Lipschitz constant for $\nabla_{\mathcal{I}_*}(f + r)$ and an upper bound on a norm of the inverse of $\nabla^2_{\mathcal{I}_*}(f + r)$, where $\mathcal{I}_* := \{j \in \mathcal{G}_i : i \in \mathcal{S}_*\}$. Due to the properties of the regularizer r, the latter of these values may be inversely proportional to the norms of the groups of variables in the support at the solution.

5 Numerical Results

In this section, we present the results of numerical experiments with an implementation of `FaRSA-Group` (Algorithm 1) applied to solve a collection of group sparse regularized logistic regression problems of the form

$$\min_{x \in \mathbb{R}^n} \frac{1}{N} \sum_{i=1}^{N} \log\left(1 + e^{-y_i x^T d_i}\right) + \sum_{i=1}^{n_{\mathcal{G}}} \lambda_i \|[x]_{\mathcal{G}_i}\|_2 , \tag{45}$$

where $d_i \in \mathbb{R}^n$ is the ith data point, N is the number of data points in the data set, $y_i \in \{-1, 1\}$ is the class label for the ith data point, and λ_i is the weight parameter for the ith group. We first describe details of our implementation, then describe the data sets considered in our experiments, and finally present our experimental results.

5.1 Implementation details

We have developed a Python implementation of `FaRSA-Group` that is available upon request. The values of the input parameters for Algorithm 1 and Algorithm 2 that we used are given in Figure 2 (with some caveats that are mentioned in the following paragraph).

We initialized x_0 as the zero vector and α_0 as an estimate of the inverse of the Lipschitz constant of f at x_0. To be precise, our software randomly generated a vector $y_0 \in \mathbb{R}^n$ such that $\|x_0 - y_0\|_2 = 10^{-8}$, and then set $\alpha_0 = \min\{1, \|x_0 - y_0\|_2 / \|\nabla f(x_0) - \nabla f(y_0)\|_2\}$. Since $\varphi = 1$, it follows from Algorithm 1 that (40) holds for all $k \in \mathbb{N}$. (However, for data sets with $N < n$, we initially chose $\varphi = 0.8$ and switched to $\varphi = 1$ when an iteration in $\mathcal{K}^{\text{cg}}$ satisfied $f(x_k) - f(x_{k+1}) \leq 10^{-3}$. When $N < n$, the matrix $\nabla^2 f(x_k)$ is singular, which in practice often led to large CG directions and multiple backtracks in the line search. These ill effects were partly remedied by this scheme for updating φ.)

param.	value	param.	value
φ	1	κ_1	0.1
ξ	0.5	κ_2	10^{-2}
η	10^{-3}	θ	$\pi/4$
ζ	0.8	q	1
p	2	μ_k	1

Figure 2: Parameter values used in our tests for Algorithm 1 and Algorithm 2.

When defining the set $\mathcal{I}_k^{\text{small}}$ in (9), we used $\tilde{\kappa}_{2,i} = \kappa_2 |\mathcal{G}_i| / \|\bar{\mathcal{I}}_k^{\text{cg}}\|$ in place of κ_2 for all i such that $\mathcal{G}_i \subseteq \bar{\mathcal{I}}_k^{\text{cg}}$. This choice accounted for the fact that the two different norms in (9) are associated with vectors of different dimension. Note that since $(1/n)\kappa_2 \leq \tilde{\kappa}_{2,i} \leq n\kappa_2$, this choice is easily incorporated into the analysis in Section 4. The choice of H_k in Line 10 was based on a regularization of the exact second-derivatives of f. In particular, for any scalar $\delta \geq 0$, consider

$$\tfrac{1}{N} D^T \Sigma_\delta(x) D \approx \nabla^2 f(x)$$

where $D^T := [d_1, d_2, \cdots, d_N]$ and $\Sigma_\delta(x)$ is the diagonal matrix with ith diagonal entry

$$[\Sigma_\delta(x)]_{ii} := \max\{\sigma_i(x)(1 - \sigma_i(x)), \delta\} \quad \text{with} \quad \sigma_i(x) := \exp(y_i d_i^T x) / \big(1 + \exp(y_i d_i^T x)\big)$$

for all $i \in \{1, 2, \cdots, N\}$. Notice that if $\delta = 0$, then $(1/N) D^T \Sigma_0(x) D \equiv \nabla^2 f(x)$. In order to use a small amount of regularization in our tests, we chose $\delta = 10^{-8}$. With this choice of δ, our choice of H_k in Line 10 can now be written as

$$H_k \leftarrow [\tfrac{1}{N} D^T \Sigma_\delta(x_k) D]_{\mathcal{I}_k \mathcal{I}_k} + \nabla^2_{\mathcal{I}_k \mathcal{I}_k} r(x_k),$$

where we remind the reader that $\nabla^2_{\mathcal{I}_k \mathcal{I}_k} r(x_k)$ is well defined because the construction of $\mathcal{I}_k \subseteq \mathcal{I}_k^{\text{cg}}$ ensures that $[x_k]_{\mathcal{G}_i} \neq 0$ for all $\mathcal{G}_i \subseteq \mathcal{I}_k$.

In Algorithm 2, we applied the CG method to the system $H_k d = -g_k$ to approximately solve the optimization problem defined in Line 29. As pointed out in Section 3.2, the direction associated with every iteration of the CG algorithm satisfies condition (13) and condition (14), which were required to establish the complexity result in Theorem 4.1. To reduce the cost of the CG computation and limit the number of backtracking steps required by Algorithm 3, we terminated Algorithm 2 when at least one of three conditions was satisfied. To describe these conditions checked during the kth iteration, let $d_{j,k}$ denote the jth CG iterate and let $t_{j,k} := \|H_k d_{j,k} + g_k\|_2$ denote the jth CG residual. The three conditions are given by

$$t_{j,k} \leq \max\{\min\left\{0.1 t_{0,k}, t_{0,k}^{1.5}\right\}, 10^{-10}\}, \tag{46a}$$

$$\|d_{j,k}\| \geq 10^3 \min\{1, \|\nabla_{\mathcal{I}_k}(f + r)(x_k)\|_2\}, \text{ and} \tag{46b}$$

$$j = |\mathcal{I}_k|. \tag{46c}$$

Outcome (46a) is the ideal termination condition since it indicates that the residual of the linear system has been sufficiently reduced (see (15)). Outcome (46b) serves as a trust-region constraint on the norm of the trial step d_k; in particular, when the inequality in (46b) holds, the size of the CG iterate $d_{j,k}$ is relatively large, indicating that x_k is not close to an optimal solution. Therefore, we restrict its size with the intent of needing fewer backtracking steps during the subsequent line search. Outcome (46c) caps the number of CG iterations to $|\mathcal{I}_k|$ (the size of the reduced space) since, in exact arithmetic, CG converges to an exact solution in at most $|\mathcal{I}_k|$ iterations.

Algorithm 1 decreases the value of the PG parameter (see Line 19) for the next iteration using a simple multiplicative factor when $\text{flag}_k^{\text{pg}} = \texttt{decrease\_}\alpha$. However, in practice, we found an adaptation of the approach in [9] to be more efficient. To describe this approach, let d_k and ξ^{j_k} be the search direction and step size used to obtain $x_{k+1} = x_k + \xi^{j_k} d_k$. It is well known [2, Lemma 5.7] that if $\alpha \in (0, 1/L_f]$, then $f(x_{k+1}) \leq f(x_k) + \xi^{j_k} \nabla f(x_k)^T d_k + \frac{1}{2\alpha} \|\xi^{j_k} d_k\|_2^2$. Setting this inequality to be an equality and then solving for α, one obtains

$$\hat{\alpha}_k := \frac{\|\xi^{j_k} d_k\|_2^2}{2\left(f(x_{k+1}) - f(x_k) - \xi^{j_k} \nabla f(x_k)^T d_k\right)},$$

which can be viewed as a local Lipschitz constant estimate for f at x_k. In our tests, we updated the PG parameter at the end of each iteration of Algorithm 1 as

$$\alpha_{k+1} \leftarrow \min\left\{1, \hat{\alpha}_k/2\right\}. \tag{47}$$

Although this PG parameter update strategy worked better than the basic strategy in Algorithm 1 (see Line 19 and Line 21), it is not covered by our analysis in Section 4. However, a simple modification of our analysis would be to allow the update in (47) to increase the PG parameter at most a finite number of times, say 100 times, at which point the update $\alpha_{k+1} \leftarrow \min\left\{\alpha_k, \hat{\alpha}_k/2\right\} \leq \alpha_k$ would be used. This strategy is covered by our earlier analysis (with a larger constant in the complexity result).

We terminate our algorithm when $\max\{\chi_k^{\text{cg}}, \chi_k^{\text{pg}}\} \leq 10^{-6} \max\{\chi_0^{\text{cg}}, \chi_0^{\text{pg}}, 1\}$.

5.2 Data sets

We tested `FaRSA-Group` on problem (45) using data sets from the LIBSVM repository.[1] From this repository, we excluded all regression instances and multiple-class (greater than two) classification instances. We compared the performance of our algorithm to the well-cited package `gglasso` [32], which is a state-of-the-art group-wise majorization descent method.[2] Since `gglasso` does not support sparse data matrix inputs, we excluded all data sets that were too large to be stored in memory (6GB). Finally, for the adult data (a1a–a9a) and webpage data (w1a–w8a), we used only the largest instances, namely a9a and w8a. This left us with our final subset of 25 data sets that can be found in Table 1.

Scaling of the data sets can be important. If the LIBSVM website indicated that a data set was already scaled, then we used the data set without modification. However, when the website did not indicate that scaling for a data set was used, we scaled each column of the feature data (i.e., feature-wise scaling) into the range $[-1, 1]$ by dividing each of its entries by the largest entry in absolute value. Labels for some data sets (e.g., breast-cancer, covtype, liver-disorders, mushrooms, phishing, skin-nonskin and svmguide1) do not take values in $\{-1, 1\}$, but rather in $\{0, 1\}$ or $\{1, 2\}$. For these data sets, we mapped the smaller label to -1 and the larger label to 1.

5.3 Experimental setup and test results

We tested `FaRSA-Group` and `gglasso` for solving problem (45) using the data sets in Table 1. All default settings for `gglasso` were used, including the same starting point $x_0 = 0$ used by `FaRSA-Group`. We considered four group structures and two different solution sparsity levels. Specifically, we considered the four different numbers of groups

$$\text{number of groups} \in \{\lfloor 0.25n \rfloor, \lfloor 0.50n \rfloor, \lfloor 0.75n \rfloor, n\},$$

where n is the problem dimension; notice that the last setting recovers ℓ_1-norm regularization. Then, for a given number of groups, the variables were sequentially distributed (as evenly as possible) to the

[1] `https://www.csie.ntu.edu.tw/cjlin/libsvmtools/datasets`

[2] `https://cran.r-project.org/web/packages/gglasso`

Table 1: The first column (data set) gives the name of the data set. The second column (N) and third column (n) indicate the number of data points and problem dimension, respectively. The fourth column (scale) provides the feature-wise scaling used: each feature is either scaled into the given interval or scaled to have mean zero ($\mu = 0$) and variance one ($\sigma^2 = 1$). The fifth column (who) indicates whether the data set came pre-scaled from the LIBSVM website (website), or it did not come pre-scaled and we scaled it (us) as described in Section 5.2. Finally, the sixth column (used) indicates the number of problem instances used in the numerical results presented in Figure 3.

data set	N	n	scale	who	used
a9a	32561	123	[0,1]	website	8
australian	690	140	[-1,1]	website	2
breast-cancer	683	10	[-1,1]	website	0
cod-rna	59535	8	[-1,1]	us	8
colon-cancer	62	2000	$(\mu, \sigma^2) = (0, 1)$	website	8
covtype.binary	581012	54	[0,1]	website	8
diabetes	768	8	[-1,1]	website	0
duke breast-cancer	44	7192	$(\mu, \sigma^2) = (0, 1)$	website	8
fourclass	862	2	[-1,1]	website	0
german-numer	1000	24	[-1,1]	website	0
gisette	6000	5000	[-1,1]	website	8
heart	270	13	[-1,1]	website	2
ijcnn1	49990	22	[-1.5, 1.5]	website	8
ionosphere	351	34	[-1,1]	website	0
leukemia	38	7129	$(\mu, \sigma^2) = (0, 1)$	website	8
liver-disorders	145	5	[-1,1]	website	0
madelon	2000	500	[-1,1]	us	8
mushrooms	8124	112	[0,1]	website	6
phishing	11055	68	[0,1]	website	7
skin-nonskin	245057	3	[-1,1]	us	8
splice	1000	60	[-1,1]	website	0
sonar	208	60	[-1,1]	website	4
svmguide1	3089	4	[-1,1]	us	0
svmguide3	1243	21	[-1,1]	website	0
w8a	49749	300	[0,1]	website	8

groups; e.g., 10 variables among 3 groups would have been distributed as $\mathcal{G}_1 = \{1, 2, 3\}$, $\mathcal{G}_2 = \{4, 5, 6\}$, and $\mathcal{G}_3 = \{7, 8, 9, 10\}$. For the two different solution sparsity levels, we considered groups weights

$$\lambda_i = 0.1\lambda_{\min}\sqrt{|\mathcal{G}_i|} \quad \text{and} \quad \lambda_i = 0.01\lambda_{\min}\sqrt{|\mathcal{G}_i|}$$

where $\lambda_{\min} = \min\left\{\lambda \geq 0 : \text{the solution to (45) with } \lambda_i = \lambda\sqrt{|\mathcal{G}_i|} \text{ is } x = 0\right\}$ (see [32, equation (23)]). Since there were 25 data sets, a total of 200 problem instances were tested (each data set has 8 instances). The experiments were conducted using the cluster in the Computational Optimization Research Laboratory (COR@L) at Lehigh University with an AMD Opteron Processor 6128 2.0 GHz CPU. In the following paragraphs, we compared the performance of `FaRSA-Group` with that of `gglasso` with respect to CPU time (seconds), final objective value, and solution sparsity.

First consider the CPU time. For each problem instance, we allowed a maximum of 1000 seconds. If the CPU time in a run went above this limit, we terminated that run and considered the algorithm to have failed. Out of the 200 problem instances, `FaRSA-Group` failed 2 times and `gglasso` failed 7 times. Figure 3 illustrates a performance profile based on [24] for comparing the computing times on problem instances that `FaRSA-Group` and/or `gglasso` took at least 1 second to terminate; this resulted in 109 problem instances. The last column of Table 1 gives the number of instances for each data set used in this profile. Each bar in

the plot corresponds to a problem instance, with the height of the bar given by

$$-\log_2\left(\frac{\text{time required by } \texttt{FaRSA-Group}}{\text{time required by } \texttt{gglasso}}\right). \tag{48}$$

Therefore, an upward pointing bar indicates that `FaRSA-Group` took less time to find the optimal solution for that problem instance and a downward pointing bar means that `gglasso` took less time, and in either case the size of the bar indicates the magnitude of the outperformance factor. A bar that reaches the y-axis limit of ± 10 is used when indicating that an algorithm was successful when solving a problem instance while the competing algorithm was unsuccessful.

To compare final objective function values, let $F_{\texttt{FaRSA-Group}}$ and $F_{\texttt{gglasso}}$ denote (for a given problem instance) the objective values returned by `FaRSA-Group` and `gglasso`, respectively. If $F_{\texttt{gglasso}} - F_{\texttt{FaRSA-Group}} > 10^{-8}$, then we considered `FaRSA-Group` to have obtained a lower objective function value; if $F_{\texttt{FaRSA-Group}} - F_{\texttt{gglasso}} > 10^{-8}$, then we considered `gglasso` to have obtained a lower objective function value; and if $|F_{\texttt{FaRSA-Group}} - F_{\texttt{gglasso}}| \leq 10^{-8}$, then we considered them to have performed equally. From the 109 problem instances that at least one algorithm took at least one second to terminate, `FaRSA-Group` outperformed `gglasso` 95 times and `gglasso` outperformed `FaRSA-Group` 7 times. From the entire 200 instances, `FaRSA-Group` outperformed `gglasso` 153 times and `gglasso` outperformed `FaRSA-Group` 35 times.

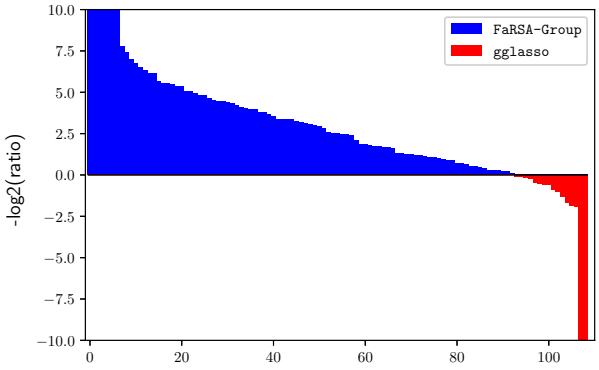

Figure 3: Performance profile for CPU time (seconds). `FaRSA-Group` outperforms `gglasso` on 93 of the 109 problem instances. For each problem instance, the height of the bar is given by (48).

In terms of solution sparsity, we considered `FaRSA-Group` to have outperformed `gglasso` if the following two conditions held: (i) all zero groups in the `gglasso` solution were also zero groups in the `FaRSA-Group` solution, and (ii) the solution returned by `FaRSA-Group` had at least one zero group that was not a zero group in the `gglasso` solution. A similar criteria was used to define when `gglasso` was considered to have outperformed `FaRSA-Group`. From the 109 test instances, `FaRSA-Group` outperformed `gglasso` in 30 cases and `gglasso` outperformed `FaRSA-Group` in 7 cases. From the entire collection of 200 problem instances, `FaRSA-Group` outperformed `gglasso` in 33 cases and `gglasso` outperformed `FaRSA-Group` in 8 cases.

6 Conclusion

We presented a new framework for solving optimization problems that incorporate group sparsity-inducing regularization by using subspace acceleration, domain decomposition, and support identification. In terms of theory, we proved a complexity result on the maximum number of iterations before an ϵ-approximate solution is computed (Theorem 4.1), and a local superlinear convergence rate (Theorem 4.3). The strong convergence theory was supported by experimental results for minimizing a group sparsity-regularized logistic function for the task of classification. In terms of robustness, computational time, final objective value obtained, and solution sparsity, the numerical results showed that our proposed `FaRSA-Group` framework outperformed a state-of-the-art method.

A Proofs

In this appendix, for completeness, we provide detailed proofs of the results from Section 2 related to the PG computations.

Proof of Lemma 2.1. Let $x_+ = T(\overline{x}, \overline{\alpha})$ denote the PG update in (3) so that $x_+ = \overline{x} + s(\overline{x}, \overline{\alpha})$ with $s(\overline{x}, \overline{\alpha})$ defined in (4). It follows from the optimality conditions for the problem in (3) that there exists $g_+ \in \partial r(x_+)$ such that

$$x_+ - \overline{x} + \overline{\alpha}\nabla f(\overline{x}) + \overline{\alpha} g_+ = 0. \tag{49}$$

Next, for an arbitrary $g_{f+r} \in \partial(f+r)(\overline{x})$, it follows from Assumption 1.1 and [4, Proposition 5.4.6] that there exits $g_r \in \partial r(\overline{x})$ satisfying $g_{f+r} = \nabla f(\overline{x}) + g_r$. From the definitions of g_r and g_+ and convexity of r, it follows that

$$r(x_+) \geq r(\overline{x}) + g_r^T(x_+ - \overline{x}) \;\text{ and }\; r(\overline{x}) \geq r(x_+) + g_+^T(\overline{x} - x_+). \tag{50}$$

Adding the two equations in (50) together yields $(g_r - g_+)^T(x_+ - \overline{x}) \leq 0$. Combining this with the definition of g_{f+r}, (49), and the definition of x_+ that

$$\begin{aligned} s(\overline{x}, \overline{\alpha})^T g_{f+r} &= (x_+ - \overline{x})^T(\nabla f(\overline{x}) + g_r) \\ &= \tfrac{1}{\overline{\alpha}}(x_+ - \overline{x})^T(\overline{x} - x_+ - \overline{\alpha} g_+ + \overline{\alpha} g_r) \\ &= -\tfrac{1}{\overline{\alpha}}\|x_+ - \overline{x}\|_2^2 + (x_+ - \overline{x})^T(g_r - g_+) \leq -\tfrac{1}{\overline{\alpha}}\|s(\overline{x}, \overline{\alpha})\|_2^2. \end{aligned} \tag{51}$$

Since $g_{f+r} \in \partial(f+r)(\overline{x})$ was arbitrary, the result [25, Theorem 2.87] and (51) yield

$$D_{f+r}(\overline{x}; s(\overline{x}, \overline{\alpha})) = \sup_{g \in \partial(f+r)(\overline{x})} s(\overline{x}, \overline{\alpha})^T g \leq -\tfrac{1}{\overline{\alpha}}\|s(\overline{x}, \overline{\alpha})\|_2^2,$$

which is the desired result and completes the proof.

Proof of Lemma 2.2. The proof follows exactly as in the proof of Lemma 2.1 above, but where all calculations are restricted to groups in the set $\mathcal{I}$ (also see (5)).

Proof of Lemma 2.3. The result, for the case $\mathcal{I} = \{1, 2, \ldots, n\}$, can be found in [2, Lemma 10.4]. For the general case, i.e., when $\mathcal{I}$ is equal to the union of a subset of $\{\mathcal{G}_i\}_{i=1}^{n_\mathcal{G}}$, the result follows by using the same proof as for [2, Lemma 11.9].

Proof of Lemma 2.4. Denote $g_i := \nabla_{\mathcal{G}_i} f(\overline{x})$, $x_i = [\overline{x}]_{\mathcal{G}_i}$, and $s_i = [s(\overline{x}, \overline{\alpha})]_{\mathcal{G}_i}$. Since $f + r$ is differentiable with respect to the variables in $\mathcal{G}_i$ at $\overline{x}$ since $[\overline{x}]_{\mathcal{G}_i} \neq 0$, we have

$$\|\nabla_{\mathcal{G}_i}(f+r)(\overline{x})\|_2^2 = \|g_i + \lambda_i x_i / \|x_i\|_2\|_2^2 = \|g_i\|_2^2 + 2\lambda_i \frac{g_i^T x_i}{\|x_i\|_2} + \lambda_i^2,$$

which means that it is sufficient to prove that

$$\|g_i\|_2^2 + 2\lambda_i \frac{g_i^T x_i}{\|x_i\|_2} + \lambda_i^2 \geq \|s_i\|_2^2.$$

Since $x_i + s_i \neq 0$ by assumption, we know that s_i (see (5)) satisfies

$$\begin{aligned} s_i &= \left(1 - \frac{\overline{\alpha}\lambda_i}{\|x_i - \overline{\alpha} g_i\|_2}\right)(x_i - \overline{\alpha} g_i) - x_i \\ &= x_i - \overline{\alpha} g_i - \frac{\overline{\alpha}\lambda_i(x_i - \overline{\alpha} g_i)}{\|x_i - \overline{\alpha} g_i\|_2} - x_i = -\overline{\alpha}\left(g_i + \frac{\overline{\alpha}\lambda_i(x_i - \overline{\alpha} g_i)}{\|x_i - \overline{\alpha} g_i\|_2}\right) \end{aligned}$$

so that

$$\|s_i\|_2^2 = \overline{\alpha}^2\left(\|g_i\|_2^2 + 2\overline{\alpha}\lambda_i \frac{g_i^T(x_i - \overline{\alpha} g_i)}{\|x_i - \overline{\alpha} g_i\|_2} + \overline{\alpha}^2\lambda_i^2\right).$$

Thus, it is sufficient to prove that

$$\|g_i\|_2^2 + 2\lambda_i \frac{g_i^T x_i}{\|x_i\|_2} + \lambda_i^2 \geq \overline{\alpha}^2\left(\|g_i\|_2^2 + 2\overline{\alpha}\lambda_i \frac{g_i^T(x_i - \overline{\alpha} g_i)}{\|x_i - \overline{\alpha} g_i\|_2} + \overline{\alpha}^2\lambda_i^2\right).$$

We consider two cases, and note that $x_i \neq 0$ by assumption and that $x_i - \overline{\alpha} g_i \neq 0$ as a consequence of (5) and the assumption that $x_i + s_i \neq 0$.

Case 1: $\overline{\alpha} = 1$. In this case, the desired inequality simplifies to

$$\frac{g_i^T x_i}{\|x_i\|_2} \geq \frac{g_i^T (x_i - g_i)}{\|x_i - g_i\|_2}. \tag{52}$$

We now consider the following two subcases.

Case 1a: $g_i^T x_i \geq 0$. The desired inequality clearly holds if $g_i^T (x_i - g_i) \leq 0$. Thus, for the remainder of this subcase, we assume that $g_i^T (x_i - g_i) > 0$, which equivalently means that $g_i^T x_i > \|g_i\|_2^2$, which implies that $-2x_i^T g_i + \|g_i\|_2^2 < 0$. It follows from this inequality and the fact that $(g_i^T x_i)^2 \leq \|g_i\|_2^2 \|x_i\|_2^2$ (by Cauchy-Schwarz) that

$$(g_i^T x_i)^2(-2x_i^T g_i + \|g_i\|_2^2) \geq (-2x_i^T g_i + \|g_i\|_2^2)\|g_i\|_2^2\|x_i\|_2^2 = \left(\|g_i\|_2^4 - 2g_i^T x_i \|g_i\|_2^2\right)\|x_i\|_2^2.$$

We can now add the term $(g_i^T x_i)^2 \|x_i\|_2^2$ to both sides to obtain

$$(g_i^T x_i)^2(\|x_i\|_2^2 - 2x_i^T g_i + \|g_i\|_2^2) \geq \left((g_i^T x_i)^2 + \|g_i\|_2^4 - 2g_i^T x_i \|g_i\|_2^2\right)\|x_i\|_2^2,$$

which can be written equivalently as

$$(g_i^T x_i)^2 \|x_i - g_i\|_2^2 \geq (g_i^T x_i - \|g_i\|_2^2)^2 \|x_i\|_2^2 = (g_i^T (x_i - g_i))^2 \|x_i\|_2^2.$$

After taking the square root of both sides, we obtain (52).

Case 1b: $g_i^T x_i < 0$. Using $g_i^T x_i < 0$ and $(g_i^T x_i)^2 \leq \|g_i\|_2^2 \|x_i\|_2^2$ (by Cauchy-Schwarz), we have

$$(g_i^T x_i)^2(-2x_i^T g_i + \|g_i\|_2^2) \leq (-2x_i^T g_i + \|g_i\|_2^2)\|g_i\|_2^2\|x_i\|_2^2 = \left(\|g_i\|_2^4 - 2g_i^T x_i \|g_i\|_2^2\right)\|x_i\|_2^2.$$

We can now add the term $(g_i^T x_i)^2 \|x_i\|_2^2$ to both sides to obtain

$$(g_i^T x_i)^2(\|x_i\|_2^2 - 2x_i^T g_i + \|g_i\|_2^2) \leq \left((g_i^T x_i)^2 + \|g_i\|_2^4 - 2g_i^T x_i \|g_i\|_2^2\right)\|x_i\|_2^2,$$

which can be written equivalently as

$$(g_i^T x_i)^2 \|x_i - g_i\|_2^2 \leq (g_i^T x_i - \|g_i\|_2^2)^2 \|x_i\|_2^2 = (g_i^T (x_i - g_i))^2 \|x_i\|_2^2.$$

After taking the square root of both sides and rearranging, we obtain

$$\frac{|g_i^T x_i|}{\|x_i\|_2} \leq \frac{|g_i^T (x_i - g_i)|}{\|x_i - g_i\|_2}.$$

Combining this result with $0 > g_i^T x_i \geq g_i^T (x_i - g_i)$ gives (52), as claimed.

Case 2: $\overline{\alpha} \in (0, 1)$. The proof of follows from Case 1 and [2, Theorem 10.9], which in our notation from (4) proves that $\|s(\overline{x}, \overline{\alpha})\|_2 \leq \|s(\overline{x}, 1)\|_2$ when $\overline{\alpha} \in (0, 1)$.

References

[1] Francis Bach, Rodolphe Jenatton, Julien Mairal, and Guillaume Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends® in Machine Learning*, 4(1):1–106, 2012.

[2] Amir Beck. *First-order methods in optimization*, volume 25. SIAM, 2017.

[3] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[4] Dimitri P Bertsekas. *Convex optimization theory*. Athena Scientific, Belmont, Ma., 2009.

[5] Tianyi Chen, Frank E. Curtis, and Daniel P. Robinson. A reduced-space algorithm for minimizing ℓ_1-regularized convex functions. *SIAM Journal on Optimization*, 27(3):1583–1610, 2017.

[6] Tianyi Chen, Frank E. Curtis, and Daniel P. Robinson. FaRSA for ℓ_1-regularized convex optimization: local convergence and numerical experience. 33(2):396–415, 2018.

[7] Patrick Combettes and Jean-Christophe Pesquet. Proximal splitting methods in signal processing. In *Fixed-point Algorithms for Inverse Problems in Science and Eng.*, pages 185–212. Springer, 2011.

[8] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. *Trust-Region Methods*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

[9] Frank E Curtis and Daniel P Robinson. Exploiting negative curvature in deterministic and stochastic optimization. *Mathematical Programming*, 176(1-2):69–94, 2019.

[10] I. Daubechies, M. Defrise, and C. Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Comm. Pure Appl. Math.*, 58:1413–1457, 2004.

[11] Ron S Dembo, Stanley C Eisenstat, and Trond Steihaug. Inexact newton methods. *SIAM Journal on Numerical analysis*, 19(2):400–408, 1982.

[12] D. Donoho. Denoising by soft-thresholding. *Trans. Inform. Theory*, 41:613–627, 1995.

[13] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, 2008.

[14] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *JMLR*, 9(Aug):1871–1874, 2008.

[15] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE J. Selected Topics Signal Process.*, 1:586–597, 2007.

[16] J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1, 2010.

[17] Geovani N. Grapiglia and Yurii Nesterov. Accelerated regularized newton methods for minimizing composite convex functions. *SIAM Journal on Optimization*, 29(1):77–99, 2019.

[18] N. Keskar, J. Nocedal, F. Oztoprak, and A. Wächter. A second-order method for convex ℓ_1-regularized optimization with active-set prediction. *Optimization Methods and Software*, 31(3):605–621, 2016.

[19] Jason D. Lee, Yuekai Sun, and Michael A. Saunders. Proximal newton-type methods for minimizing composite functions. *SIAM Journal on Optimization*, 24(3):1420–1443, 2014.

[20] Qihang. Lin, Zhaosong. Lu, and Lin. Xiao. An accelerated randomized proximal coordinate gradient method and its application to regularized empirical risk minimization. *SIAM Journal on Optimization*, 25(4):2244–2273, 2015.

[21] J. Liu, S. Ji, and J. Ye. *SLEP: Sparse Learning with Efficient Projections*. Arizona State University, 2009.

[22] Ji Liu and Stephen J. Wright. Asynchronous stochastic coordinate descent: Parallelism and convergence properties. *SIAM Journal on Optimization*, 25(1):351–376, 2015.

[23] Shuangge Ma, Xiao Song, and Jian Huang. Supervised group Lasso with applications to microarray data analysis. *BMC bioinformatics*, 8(1):60, 2007.

[24] José Luis Morales. A numerical study of limited memory BFGS methods. *Applied Mathematics Letters*, 15(4):481–487, 2002.

[25] Boris S Mordukhovich and Nguyen Mau Nam. *An easy path to convex analysis and applications*, volume 6. Morgan & Claypool Publishers, 2013.

[26] Yu Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

[27] Yurii Nesterov. A method of solving a convex programming problem with convergence rate $\mathcal{O}(1/k^2)$. *Soviet Mathemtics Doklady*, 27(2):372–376, 1983.

[28] Julie Nutini, Mark Schmidt, and Warren Hare. Active-set complexity of proximal gradient: How long does it take to find the sparsity pattern? *Optimization Letters*, 13(4):645–655, 2019.

[29] Peter Richtárik and Martin Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 156(1):433–484, Mar 2016.

[30] Rachael Tappenden, Peter Richtárik, and Jacek Gondzio. Inexact coordinate descent: complexity and preconditioning. *J. of Optimization Theory and Applications*, 170(1):144–176, 2016.

[31] Stephen J. Wright, Robert D. Nowak, and Mário A.T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.

[32] Yi Yang and Hui Zou. A fast unified algorithm for solving group-lasso penalize learning problems. *Statistics and Computing*, 25(6):1129–1141, Nov 2015.

[33] Guo-Xun Yuan, Chia-Hua Ho, and Chih-Jen Lin. An improved GLMNET for ℓ_1-regularized logistic regression. *Journal of Machine Learning Research*, 13(Jun):1999–2030, 2012.

[34] Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *J. of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

[35] Yaohui Zeng and Patrick Breheny. Overlapping group logistic regression with applications to genetic pathway selection. *Cancer Informatics*, 15:CIN–S40043, 2016.