

Accelerated Block Coordinate Proximal Gradients with Applications in High Dimensional Statistics

Tsz Kit Lau

Department of Mathematics, Hong Kong University of Science and Technology, Hong Kong

tklauag@connect.ust.hk

Yuan Yao

Department of Mathematics, Hong Kong University of Science and Technology, Hong Kong

yuany@ust.hk

Abstract

Nonconvex optimization problems arise in different research fields and arouse lots of attention in signal processing, statistics and machine learning. In this work, we explore the accelerated proximal gradient method and some of its variants which have been shown to converge under nonconvex context recently. We show that a novel variant proposed here, which exploits adaptive momentum and block coordinate update with specific update rules, further improves the performance of a broad class of nonconvex problems. In applications to sparse linear regression with regularizations like Lasso, grouped Lasso, capped ℓ_1 and SCAD, the proposed scheme enjoys provable local linear convergence, with experimental justification.

1 Introduction

Many problems in machine learning are targeted to solve the following minimization problem

$$\min_{\mathbf{x} \in \mathbb{R}^n} F(\mathbf{x}) \equiv f(\mathbf{x}) + g(\mathbf{x}), \quad (1)$$

where f is differentiable, g can possibly be nonsmooth. Convexity is not assumed for f and g . If F is convex, it is shown that (1) can be solved efficiently by the accelerated proximal gradient (APG) method (sometimes referred to as FISTA [4], as in Algorithm 1. APG has a convergence rate of $O(1/k^2)$ which meets the theoretical lower bound of first-order gradient methods for minimizing smooth convex functions.

Algorithm 1: APG

Input: $\mathbf{y}_1 = \mathbf{x}_1 = \mathbf{x}_0, t_1 = 1, t_0 = 0, \eta \leq 1/L$.

for $k = 1, 2, \dots$ **do**

$$\left[\begin{array}{l} \mathbf{y}_k = \mathbf{x}_k + \frac{t_{k-1}-1}{t_k}(\mathbf{x}_k - \mathbf{x}_{k-1}), \\ \mathbf{x}_{k+1} = \text{prox}_{\eta g}(\mathbf{y}_k - \eta \nabla f(\mathbf{y}_k)), \\ t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2}. \end{array} \right.$$

For nonconvex version of (1), APG was first introduced and analyzed by Li and Lin [12], in which they propose monotone APG (mAPG) and nonmonotone APG (nmAPG) by exploiting the Kurdyka-Łojasiewicz (KL) property. The main deficiency of these two algorithms is that they require two proximal steps in each iteration. In view of this, Yao et al. [19] propose nonconvex inexact APG (niAPG) which is equivalent to APG for nonconvex problems (APGnc) in [13] if the proximal step is exact. This is an algorithm with comparable numerical perfor-

mance to other state-of-the-art algorithms.

Li et al. [13] analyze the convergence rates of mAPG and APGnc by exploiting the KL property. They further propose an APGnc⁺ algorithm, which improves APGnc by introducing an adaptive momentum (Algorithm 3, see Appendix). APGnc⁺ has the same theoretical convergence rate as APGnc but has better numerical performance. However, the aforementioned APG-like algorithms do not leverage the special structure of the objective function F (particularly the structure of g), which is what we want to explore in this paper.

In practice, many machine learning and statistics problems in the form of (1) have separable or block-separable regularizer g , so we can rewrite (1) as

$$\min_{\mathbf{x} \in \mathbb{R}^n} F(\mathbf{x}) \equiv f(\mathbf{x}) + \sum_{i=1}^s g_i(\mathbf{x}_i), \quad (2)$$

where the variable $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_s) \in \mathbb{R}^n$ has s blocks, $s \geq 1$, and again the functions $g_i, i = 1, \dots, s$, can be nonconvex and nonsmooth. Block coordinate update is widely applied to solve convex and nonconvex

problems in the form of (2). Since each iteration has low computational cost and small required memory, it is easy for parallel and distributed implementations and thus regarded as a more feasible method for large-scale problems. Xu and Yin [18] propose a block prox-linear (BPL) method (Algorithm 4, see Appendix) which can be viewed as a block coordinate version of APG. At iteration k , only one block $b_k \in \{1, \dots, s\}$ is selected and updated. They establish the whole sequence convergence of BPL to a critical point, first by obtaining subsequence convergence followed by exploiting the KL property again. In their numerical tests, they mainly resort to randomly shuffling of the blocks and show that it leads to better numerical performance, as opposed to cyclic (Gauss-Seidel iteration scheme) updates [1, 2] and randomized block selection [14].

Contribution. Our main contribution is to bring adaptive momentum and block prox-linear method together in a new algorithm, for a further numerical speed-up sharing the same theoretical convergence guarantee under the KL property. Moreover, a new block update rule based on Gauss-Southwell Rule, is shown to beat randomized or cyclic updates as it selects the “best” block maximizing the magnitude of the step at each iteration. In the applications to high dimensional statistics, we particularly show that for sparse linear regressions with regularizations including (grouped) Lasso, capped ℓ_1 , and SCAD, the proposed algorithm has *provable local linear convergence*. §2 presents the algorithm and §3 discusses its applications in those sparse linear regressions together with empirical justification.

2 The BCoAPGnc⁺ Algorithm

We present our proposed algorithm in Algorithm 2, Block-Coordinate APGnc with adaptive momentum (BCoAPGnc⁺), which takes advantage of several acceleration tools in Algorithms 3 and 4.

Algorithm 2: Block-Coordinate APGnc with adaptive momentum (BCoAPGnc⁺)

Input: no. of blocks s , $\mathbf{y}^{(1)} = \mathbf{x}^{(1)} = \mathbf{x}^{(0)} = \mathbf{x}^{(-1)} = \mathbf{v}^{(0)}$, $\beta, t \in (0, 1)$, $\beta^{(0)} = \beta \mathbf{1}_s$, $\mathbf{t} = t \mathbf{1}_s$.

for $k = 1, 2, \dots$ **do**

Pick $b_k \in \{1, \dots, s\}$ in a deterministic (e.g., GS- r rule) or random manner and set $\alpha_i^{(k)}$,

$$\left\{ \begin{array}{ll} \mathbf{x}_i^{(k)} = \mathbf{x}_i^{(k-1)} & \text{if } i \neq b_k, \\ \mathbf{x}_i^{(k)} \in \text{prox}_{\alpha_i^{(k)} g_i} \left(\widehat{\mathbf{x}}_i^{(k)} - \alpha_i^{(k)} \nabla_{\mathbf{x}_i} f \left(\mathbf{x}_{\neq i}^{(k-1)}, \widehat{\mathbf{x}}_i^{(k)} \right) \right) & \text{if } i = b_k, \end{array} \right.$$

$$\left\{ \begin{array}{ll} \text{where } \widehat{\mathbf{x}}_i^{(k)} := \mathbf{x}_i^{(k-1)} + \beta_i^{(k-1)} (\mathbf{x}_i^{(k-1)} - \mathbf{x}_i^{(k-2)}), \\ \mathbf{v}_i^{(k)} = \mathbf{v}_i^{(k-1)} & \text{if } i \neq b_k, \\ \mathbf{v}_i^{(k)} = \mathbf{x}_i^{(k)} + \beta_i^{(k-1)} (\mathbf{x}_i^{(k)} - \mathbf{x}_i^{(k-1)}) & \text{if } i = b_k, \end{array} \right.$$

$$\left\{ \begin{array}{ll} \beta_i^{(k)} = \beta_i^{(k-1)} & \text{if } i \neq b_k, \\ \beta_i^{(k)} = \beta_i^{(k-1)} t_i & \text{if } i = b_k, \end{array} \right.$$

if $F(\mathbf{x}^{(k)}) \leq F(\mathbf{v}^{(k)})$ **then**

$$\left\{ \begin{array}{ll} \beta_i^{(k)} = \beta_i^{(k-1)} & \text{if } i \neq b_k, \\ \beta_i^{(k)} = \beta_i^{(k-1)} t_i & \text{if } i = b_k, \end{array} \right.$$

else if $F(\mathbf{x}^{(k)}) \geq F(\mathbf{v}^{(k)})$ **then**

$$\left\{ \begin{array}{ll} \beta_i^{(k)} = \beta_i^{(k-1)} & \text{if } i \neq b_k, \\ \beta_i^{(k)} = \min \left\{ \beta_i^{(k-1)} / t_i, 1 \right\} & \text{if } i = b_k. \end{array} \right.$$

In each iteration of this algorithm, the updates of $\widehat{\mathbf{x}}_i^{(k)}$ and $\mathbf{x}_i^{(k)}$ follow that of Algorithm 4. The extrapolation step $\mathbf{v}_i^{(k)}$ aims to further exploit the opportunity of acceleration by magnifying the momentum β_i of the block i when $\mathbf{v}^{(k)}$ achieves an even lower objective value. If this does not hold, the momentum is diminished. This intuitive yet efficient step follows the main idea of Algorithm 3.

Gauss-Southwell rules. In [15], three proximal-gradient Gauss-Southwell rules are presented, namely the GS- s , GS- r and GS- q rules (see Appendix for details). In particular, in §3, we test with the GS- r which would possibly speed up the convergence since it selects the block which maximizes the magnitude of the step at each iteration.

We need certain assumptions of Problem (2) in order to establish the convergence of our Algorithm 2.

Assumption 1 We have several assumptions on the functions F , f and g_i : (i) F is proper and bounded below in $\text{dom } F$, f is continuously differentiable, and g_i is proper lower semicontinuous for every i . Problem (4) has a critical point \mathbf{x}^* , i.e., $\mathbf{0} \in \partial F(\mathbf{x}^*)$; (ii) Let $i = b_k$, then $\nabla_{\mathbf{x}_i} f(\mathbf{x}_{\neq i}^{(k-1)}, \mathbf{x}_i)$ has Lipschitz constant L_k with respect to \mathbf{x}_i , which is bounded, finite and positive for all k ; (iii) In Algorithm 2 every block is updated at least once within any T iterations (see Appendix for definitions and notations).

Theorem 1 (Whole sequence convergence) Suppose that Assumption 1 holds. Let $\{\mathbf{x}^{(k)}\}_{k \geq 1}$ be generated from Algorithm 2. Assume (i) $\{\mathbf{x}^{(k)}\}_{k \geq 1}$ has a finite limit point $\bar{\mathbf{x}}$; (ii) F satisfies the KL property (Definition 6, see Appendix) around $\bar{\mathbf{x}}$ with parameters ρ , η and θ ; (iii) For each i , $\nabla_{\mathbf{x}_i} f(\mathbf{x})$ is Lipschitz continuous within $B_{4\rho}(\bar{\mathbf{x}})$ with respect to \mathbf{x} . Then, we have $\lim_{k \rightarrow \infty} \mathbf{x}^{(k)} = \bar{\mathbf{x}}$.

PROOF This theorem mainly follows from Theorem 2 of [18], since in its proof, there are no strict requirements on the momentum as long as it is between 0 and 1, and on the block update rule for each iteration. The step size α_k can be set to fulfill the assumption of this theorem which depends on the choice of Lipschitz constant L_i of $\nabla_{\mathbf{x}_i} f(\mathbf{x}_{\neq i}^{(k-1)}, \mathbf{x}_i)$.

3 Applications in High Dimensional Statistics

Many optimization problems in machine learning and statistics can be formulated in the form of (1), for instance, sparse learning [3], regressions with nonconvex regularizers [9, 20], capped ℓ_1 -norm [21] and the log-sum-penalty [6]. We consider the general class of regularized least squares or regression problems having the form

$$\min_{\mathbf{x} \in \mathbb{R}^n} \frac{1}{2n} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 + R(\mathbf{x}), \quad (3)$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^m$ and $R(\mathbf{x})$ is a separable or block separable regularizer. We test our algorithm with two convex problems and two nonconvex problems, mainly with sparse instances since GS rules can be efficiently calculated in such cases. These strategies can be applied to nonconvex problems since the calculation does not depend on convexity [15]. All objective functions in this section are KL functions since all of them are semialgebraic [1, 17].

ℓ_1 -regularized Underdetermined Sparse Least Squares. In this case, $R(\mathbf{x}) = \lambda \|\mathbf{x}\|_1$ which is a nonsmooth regularizer promoting sparsity. We generate \mathbf{A} and \mathbf{b} in the same way as that of §9 in [15], with $\lambda = 1$, $m = 1000$ and $n = 5000$. In BPL and BCoAPGnc⁺, for illustrative purpose, we separate \mathbf{x} into s blocks (set $s = 5$) of equal size $N := n/s$, and the step size at each iteration for a selected block b_k is chosen to be $\alpha_i^{(k)} := 1/\|\mathbf{A}_{:,b_k}^\top \mathbf{A}_{:,b_k}\|_2$, where $\mathbf{A}_{:,b_k}$ represents the columns of \mathbf{A} corresponds to the block b_k . We also take $\beta = t = 0.9$ for APGnc⁺ and BCoAPGnc⁺.

ℓ_1/ℓ_2 -regularized Sparse Least Squares. In this case, $R(\mathbf{x}) = \lambda \sum_{g \in \mathcal{G}} \|\mathbf{x}_g\|_2$, where \mathcal{G} is a partition of $\{1, \dots, n\}$ containing $\{1 + kn/s, \dots, (k+1)n/s\}$, where $k = 0, \dots, s-1$, for $s = 5$ and $n = 5000$. We use the same data set and parameters except $\beta = 0.8$ and $t = 0.2$. Only block coordinate methods are used since $R(\mathbf{x})$ is not completely separable.

Capped ℓ_1 -regularized Sparse Least Squares. We consider the nonconvex capped ℓ_1 penalty [21], $R(\mathbf{x}) = \lambda \sum_{i=1}^n \min\{|x_i|, \theta\}$, $\theta > 0$. In this case we specify $s = 10$, $\lambda = 0.0001$, $\theta = 0.1\lambda$, $\beta = 0.8$ and $t = 0.2$.

Least Squares with SCAD penalty. We consider another nonconvex penalty term, the smoothly clipped absolute deviation (SCAD) penalty [9], $R(\mathbf{x}) = \sum_{i=1}^n r_{\lambda, \gamma}(x_i)$, where $r_{\lambda, \gamma}(u)$ is defined in Appendix. Both \mathbf{A} and \mathbf{b} are sampled from $\mathcal{N}(0, 1)$ but they are standardized such that \mathbf{b} and each column of \mathbf{A} have zero mean, and each column of \mathbf{A} has unit variance. We take $m = 1000$, $n = 5000$, $s = 10$, $\lambda = 0.0001$ and $\gamma = 3$.

Theorem 2 (Convergence rate) Suppose F is in the form of (3), where $R(\mathbf{x})$ is chosen as the above four examples. Under the assumptions of Theorem 1, we have $\|\mathbf{x}^{(k)} - \bar{\mathbf{x}}\| \leq C\alpha^k$, $\forall k$, for a certain $C > 0$, $\alpha \in [0, 1)$. Thus, $\{\mathbf{x}^{(k)}\}_{k \geq 1}$ converges locally linearly to a stationary point of F .

PROOF According to Propositions 4.1, 4.2 and 5.2 of [11], all four above examples of F are KL functions with an exponent $\theta = 1/2$ (for ℓ_1/ℓ_2 -regularized least squares, we also need a mild assumption that $\inf_{\mathbf{x} \in \mathbb{R}^n} F(\mathbf{x}) > \inf_{\mathbf{x} \in \mathbb{R}^n} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2/(2n)$). Then, the desired result follows immediately from Theorem 3 of [18]. The convergence rate theorem for general F is the same as this Theorem 3 for the same reason as in the proof of Theorem 1.

We plot the value $F(\mathbf{x}) - F(\mathbf{x}^*)$ in each experiment, for the proposed algorithm and some existing APG-like algorithms mentioned in §1. For fair comparison, since in each iteration block coordinate methods only update one block, we consider s block updates as one iteration in the plots. We observe in Figure 1 that our proposed

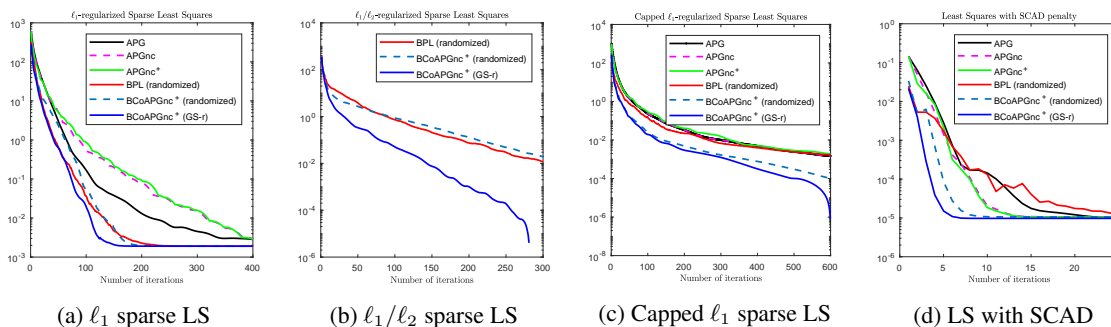


Figure 1: Comparison of APG-like algorithms

algorithm BCoAPGnc⁺ (both randomized and GS- r versions) provides the greatest initial acceleration, in both convex and nonconvex examples. It dominates most existing methods, especially during the first 20 iterations. We also see that in general the BCoAPGnc⁺ with GS- r updates outperforms that with randomized updates, justifying the use of GS- r rule for further acceleration. We see that BCoAPGnc⁺ has superior performance in nonconvex problems revealed in Figures 1c and 1d, where its original counterpart BPL does not give monotone objective value decline (in which its momentum is chosen according to that of APG). Overall, both versions of our proposed BCoAPGnc⁺ speed up the convergence, compared with the current state-of-the-art APGnc⁺ and BPL.

4 Conclusion and Future Work

In this paper, we suggest a new algorithm which considers three main acceleration techniques, namely adaptive momentum, block coordinate update and GS- r update rule. We also implement with adaptive step sizes in our experiments. We show that it shares the same convergence guarantee and convergence rate as BPL. It is noteworthy that all objective functions in sparse linear regression considered here, including (grouped) Lasso, capped ℓ_1 and SCAD regularizations, are KL functions with an exponent $1/2$, and thus have local linear convergence to their stationary points. Experiments show impressive results that our proposed method outperforms the current state-of-the-art.

We focus on experiments with convex losses and (block) separable regularizers. Other applications with block separable regularizers but nonconvex losses such as matrix factorization and completion (e.g., in [17]) deserve further treatment by applying the proposed algorithm. Further acceleration would be the use of variable metrics [7], which makes use of specific preconditioning matrices. For more general nonconvex optimization problems, it is interesting to find the KL exponents of such objective functions in order to find their local convergence rates. Extra theoretical and empirical work in these directions is expected in the future.

Acknowledgments

The authors would like to thank Jian-Feng Cai and Jinshan Zeng for useful discussions and comments.

References

- [1] H. Attouch, J. Bolte, P. Redont, and A. Soubeyran. Proximal alternating minimization and projection methods for nonconvex problems: an approach based on the Kurdyka-Łojasiewicz inequality. *Mathematics of Operations Research*, 35(2):438–457, 2010.
- [2] H. Attouch, J. Bolte, and B. F. Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. *Mathematical Programming*, 137(1):91–129, 2013.
- [3] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends® in Machine Learning*, 4(1):1–106, 2012.

- [4] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.
- [5] J. Bolte, S. Sabach, and M. Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1):459–494, 2014.
- [6] E. J. Candès, M. B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008.
- [7] E. Chouzenoux, J.-C. Pesquet, and A. Repetti. A block coordinate variable metric forward–backward algorithm. *Journal of Global Optimization*, 66(3):457–485, 2016.
- [8] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In H. H. Bauschke, R. S. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, editors, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer New York, 2011.
- [9] J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.
- [10] P. Gong, C. Zhang, Z. Lu, J. Huang, and J. Ye. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 37–45, 2013.
- [11] G. Li and T. K. Pong. Calculus of the exponent of Kurdyka–Łojasiewicz inequality and its applications to linear convergence of first-order methods. *Foundations of Computational Mathematics*, 2017.
- [12] H. Li and Z. Lin. Accelerated proximal gradient methods for nonconvex programming. In *Advances in Neural Information Processing Systems 28*, pages 379–387. 2015.
- [13] Q. Li, Y. Zhou, Y. Liang, and P. K. Varshney. Convergence analysis of proximal gradient with momentum for nonconvex optimization. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 2111–2119, 2017.
- [14] Q. Lin, Z. Lu, and L. 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.
- [15] J. Nutini, M. Schmidt, I. Laradji, M. Friedlander, and H. Koepke. Coordinate descent converges faster with the gauss-southwell rule than random selection. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, volume 37, pages 1632–1641, 2015.
- [16] R. Rockafellar and R. J.-B. Wets. *Variational Analysis*. Springer Verlag, 1998.
- [17] Y. Xu and W. Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on Imaging Sciences*, 6(3):1758–1789, 2013.
- [18] Y. Xu and W. Yin. A globally convergent algorithm for nonconvex optimization based on block coordinate update. *Journal of Scientific Computing*, 72(2):700–734, 2017.
- [19] Q. Yao, J. T. Kwok, F. Gao, W. Chen, and T.-Y. Liu. Efficient inexact proximal gradient algorithm for nonconvex problems. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 3308–3314, 2017.
- [20] C.-H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *Ann. Statist.*, 38(2): 894–942, 2010.
- [21] T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *Journal of Machine Learning Research*, 11:1081–1107, 2010.

Appendix

In this Appendix, we provide more details not mentioned in the main text due to space constraint.

Algorithm 3: APGnc with adaptive momentum (APGnc⁺)

Input: $\mathbf{y}_1 = \mathbf{x}_1 = \mathbf{x}_0, \beta, t \in (0, 1), \eta \leq 1/L$.

for $k = 1, 2, \dots$ **do**

$\mathbf{x}_k = \text{prox}_{\eta g}(\mathbf{y}_k - \eta \nabla f(\mathbf{y}_k)),$
 $\mathbf{v}_k = \mathbf{x}_k + \beta(\mathbf{x}_k - \mathbf{x}_{k-1}),$
if $F(\mathbf{x}_k) \leq F(\mathbf{v}_k)$ **then**
 $\mathbf{y}_{k+1} = \mathbf{x}_k, \beta \leftarrow t\beta,$
else if $F(\mathbf{x}_k) \geq F(\mathbf{v}_k)$ **then**
 $\mathbf{y}_{k+1} = \mathbf{v}_k, \beta \leftarrow \min\{\beta/t, 1\}.$

Algorithm 4: Block prox-linear (BPL)

Input: $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} = \mathbf{x}^{(-1)}$.

for $k = 1, 2, \dots$ **do**

Pick $b_k \in \{1, \dots, s\}$ in a deterministic or random manner,
 Set $\alpha_k > 0$ and $\omega_k \geq 0$,

$$\begin{cases} \mathbf{x}_i^{(k)} = \mathbf{x}_i^{(k-1)} & \text{if } i \neq b_k, \\ \mathbf{x}_i^{(k)} \in \text{prox}_{\alpha_k g_i}(\widehat{\mathbf{x}}_i^{(k)} - \alpha_k \nabla_{\mathbf{x}_i} f(\mathbf{x}_{\neq i}^{(k-1)}, \widehat{\mathbf{x}}_i^{(k)})) & \text{if } i = b_k, \end{cases}$$

 where $\widehat{\mathbf{x}}_i^{(k)} = \mathbf{x}_i^{(k-1)} + \omega_k(\mathbf{x}_i^{(k-1)} - \mathbf{x}_i^{(k-2)}).$

Notation 1 $\mathbf{x}_{<i}$ is the shorthand notation for $(\mathbf{x}_1, \dots, \mathbf{x}_{i-1})$, $\mathbf{x}_{>i}$ is the short-hand notation for $(\mathbf{x}_{i+1}, \dots, \mathbf{x}_s)$, $\mathbf{x}_{\neq i}$ means $(\mathbf{x}_{<i}, \mathbf{x}_{>i})$. So $f(\mathbf{x}_{\neq i}, \widehat{\mathbf{x}}_i)$ means $f(\mathbf{x}_{<i}, \widehat{\mathbf{x}}_i, \mathbf{x}_{>i})$.

Definition 1 (Proximity operator [8]) Let λ be a positive parameter. The *proximity operator* $\text{prox}_{\lambda g} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is defined through

$$\text{prox}_{\lambda g}(\mathbf{x}) := \underset{\mathbf{y} \in \mathbb{R}^n}{\text{argmin}} g(\mathbf{y}) + \frac{1}{2\lambda} \|\mathbf{y} - \mathbf{x}\|^2.$$

If g is convex, proper and lower semicontinuous, prox_g admits a unique solution. If g is nonconvex, then it is generally set-valued.

Definition 2 (Domain) The *domain* of $F : \mathbb{R}^n \rightarrow \mathbb{R}$ is defined by

$$\text{dom } F := \{\mathbf{x} \in \mathbb{R}^n : F(\mathbf{x}) < +\infty\}.$$

Definition 3 (Subdifferential [16])

- For a given $\mathbf{x} \in \text{dom } F$, the *Fréchet subdifferential* of F at \mathbf{x} , written $\widehat{\partial}F(\mathbf{x})$, is the set of all vectors $\mathbf{u} \in \mathbb{R}^n$ which satisfy

$$\liminf_{\mathbf{y} \neq \mathbf{x}, \mathbf{y} \rightarrow \mathbf{x}} \frac{F(\mathbf{y}) - F(\mathbf{x}) - \langle \mathbf{u}, \mathbf{y} - \mathbf{x} \rangle}{\|\mathbf{y} - \mathbf{x}\|} \geq 0.$$

When $\mathbf{x} \notin \text{dom } F$, we set $\widehat{\partial}F(\mathbf{x}) = \emptyset$.

- The *limiting-subdifferential*, or simply the *subdifferential*, of F at $\mathbf{x} \in \text{dom } F$, written $\partial F(\mathbf{x})$, is defined through the following closure process

$$\partial F(\mathbf{x}) := \{\mathbf{v} \in \mathbb{R}^n : \exists \mathbf{x}^k \rightarrow \mathbf{x}, F(\mathbf{x}^k) \rightarrow F(\mathbf{x}), \mathbf{v}^k \in \widehat{\partial}F(\mathbf{x}^k) \rightarrow \mathbf{v}\}.$$

Definition 4 (Sublevel set) Being given real numbers α and β we set

$$[\alpha \leq F \leq \beta] := \{\mathbf{x} \in \mathbb{R}^n : \alpha \leq F(\mathbf{x}) \leq \beta\}.$$

$[\alpha < F < \beta]$ is defined similarly.

Definition 5 (Distance) The distance of a point $\mathbf{x} \in \mathbb{R}^n$ to a closed set $\Omega \subseteq \mathbb{R}^n$ is defined as

$$\text{dist}(\mathbf{x}, \Omega) := \inf_{\mathbf{y} \in \Omega} \|\mathbf{y} - \mathbf{x}\|.$$

If $\Omega = \emptyset$, we have that $\text{dist}(\mathbf{x}, \Omega) = \infty$ for all \mathbf{x} .

Definition 6 (Kurdyka-Łojasiewicz property and KL function [5])

1. The function $F : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ is said to have the *Kurdyka-Łojasiewicz property* at $\bar{\mathbf{x}} \in \text{dom } \partial F$ if there exist $\eta \in (0, +\infty]$, a neighbourhood $\mathcal{B}_\rho(\bar{\mathbf{x}}) := \{\mathbf{x} : \|\mathbf{x} - \bar{\mathbf{x}}\| < \rho\}$ of $\bar{\mathbf{x}}$ and a continuous concave function $\varphi(t) := ct^{1-\theta}$ for some $c > 0$ and $\theta \in [0, 1)$ such that for all $\mathbf{x} \in \mathcal{B}_\rho(\bar{\mathbf{x}}) \cap [F(\bar{\mathbf{x}}) < F < F(\bar{\mathbf{x}}) + \eta]$, the Kurdyka-Łojasiewicz inequality holds

$$\varphi'(F(\mathbf{x}) - F(\bar{\mathbf{x}})) \text{dist}(\mathbf{0}, \partial F(\mathbf{x})) \geq 1.$$

2. Proper lower semicontinuous functions which satisfy the Kurdyka-Łojasiewicz inequality at each point of $\text{dom } \partial F$ are called *KL functions*.

Definition 7 (KL exponent [11]) For a proper closed function F satisfying the KL property at $\bar{\mathbf{x}} \in \text{dom } \partial F$, if the corresponding function φ can be chosen as in Definition 6, the KL inequality can be written as

$$\text{dist}(\mathbf{0}, \partial F(\mathbf{x})) \geq \bar{c}(F(\mathbf{x}) - F(\bar{\mathbf{x}}))^\theta$$

for some $\bar{c} > 0$. We say F has the KL property at $\bar{\mathbf{x}}$ with an exponent θ . If F is a KL function and has the same exponent θ at any $\bar{\mathbf{x}} \in \text{dom } \partial F$, then we can say that F is a KL function with an exponent of θ .

Definition 8 (SCAD penalty [9]) The SCAD penalty $r_{\lambda, \gamma}(u)$ is defined as

$$r_{\lambda, \gamma}(u) = \begin{cases} \lambda|u|, & \text{if } |u| < \lambda, \\ \frac{2\gamma\lambda|u| - (u^2 + \lambda^2)}{2(\gamma-1)}, & \text{if } \lambda < |u| \leq \gamma\lambda, \\ \frac{\lambda^2(\gamma^2-1)}{2(\gamma-1)}, & \text{if } |u| > \gamma\lambda. \end{cases}$$

Proposition 1 (Proximal-Gradient Gauss-Southwell rules [10])

1. In coordinate descent methods, the GS- s rule chooses the coordinate with the most negative directional derivative, given by

$$i_k = \underset{i}{\text{argmax}} \left\{ \min_{s \in \partial g_i} |\nabla_i f(\mathbf{x}^{(k)}) + s| \right\}.$$

We generalize it to the block coordinate scenario which has the form

$$i_k = \underset{i}{\text{argmax}} \left\{ \min_{s \in \partial g_i} \|\nabla_{\mathbf{x}_i} f(\mathbf{x}^{(k)}) + s\| \right\}.$$

2. The GS- r rule selects the coordinate which maximizes the length of the step

$$i_k = \underset{i}{\text{argmax}} \left\{ \left\| x_i^{(k)} - \text{prox}_{g_i/L} \left(x_i^{(k)} - \frac{1}{L} \nabla_i f(\mathbf{x}^{(k)}) \right) \right\| \right\},$$

which is generalized to the block coordinate version

$$i_k = \underset{i}{\text{argmax}} \left\{ \left\| \mathbf{x}_i^{(k)} - \text{prox}_{g_i/L} \left(\mathbf{x}_i^{(k)} - \frac{1}{L} \nabla_{\mathbf{x}_i} f(\mathbf{x}^{(k)}) \right) \right\| \right\}.$$

3. The GS- q rule maximizes the progress assuming a quadratic upper bound on f

$$i_k = \underset{i}{\text{argmax}} \left\{ \min_d \left[f(\mathbf{x}^{(k)}) + \nabla_i f(\mathbf{x}^{(k)})d + \frac{L}{2}d^2 + g_i(\mathbf{x}_i^{(k)} + d) - g_i(\mathbf{x}_i^{(k)}) \right] \right\}.$$

We do not apply this to a block coordinate scenario.

Proposition 2 (Proximity operators of regularizers in §3) If $\mathbf{x}, \mathbf{u} \in \mathbb{R}^n$, then

1. for $R(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$ [3],

$$\text{prox}_{\lambda\|\cdot\|_1}(\mathbf{u}) = \left(\text{prox}_{\lambda|\cdot|}(\mathbf{u}_i) \right)_{1 \leq i \leq n} = (\text{sign}(\mathbf{u}_i)(|\mathbf{u}_i| - \lambda)_+)_{1 \leq i \leq n},$$

where $(u)_+ = \max\{0, u\}$ and the above proximity operator is often referred to as *soft-thresholding*.

2. for the ℓ_1/ℓ_2 -norm $R(\mathbf{x}) = \lambda\|\mathbf{x}\|_2$ [3],

$$\text{prox}_{\lambda\|\cdot\|_2}(\mathbf{u}) = \left(1 - \frac{\lambda}{\|\mathbf{u}\|_2}\right)_+ \mathbf{u}.$$

Further, if \mathcal{G} is a partition of $\{1, \dots, n\}$, for the ℓ_1/ℓ_2 -norm $R : \mathbf{x} \mapsto \lambda \sum_{g \in \mathcal{G}} \|\mathbf{x}_g\|_2$ we have

$$(\text{prox}_R(\mathbf{u}))_g = \left(1 - \frac{\lambda}{\|\mathbf{u}_g\|_2}\right)_+ \mathbf{u}_g,$$

which is often referred to as *group-soft-thresholding*. The problem being solved is called *group Lasso* if it is used with a least square loss in Equation (3).

3. for $r_{\lambda,\gamma}(x)$ defined in Definition 8 [10], let $h_1(x) = \frac{1}{2}(x - u)^2 + r_{\lambda,\gamma}(x)$ and

$$\begin{aligned} u_1 &:= \text{sign}(u) \min\{\lambda, (|u| - \lambda)_+\}, \\ u_2 &:= \text{sign}(u) \min\left\{\gamma\lambda, \max\left\{\lambda, \frac{|u|(\gamma - 1) - \gamma\lambda}{\gamma - 2}\right\}\right\}, \\ u_3 &:= \text{sign}(u) \max\{\gamma\lambda, |u|\}. \end{aligned}$$

Then, we have

$$\text{prox}_{r_{\lambda,\gamma}}(u) = \begin{cases} u_1 & \text{if } u_1 = \text{argmin}_y h_1(y), \\ u_2 & \text{if } u_2 = \text{argmin}_y h_1(y), \\ u_3 & \text{if } u_3 = \text{argmin}_y h_1(y). \end{cases}$$

Thus, we further have

$$\text{prox}_{\sum_{i=1}^n r_{\lambda,\gamma}}(\mathbf{u}) = \left(\text{prox}_{r_{\lambda,\gamma}}(\mathbf{u}_i)\right)_{1 \leq i \leq n}.$$

4. for $R(\mathbf{x}) = \lambda \sum_{i=1}^n \min\{|x_i|, \theta\}$ [10], let $h_2(x) = \frac{1}{2}(x - v)^2 + \lambda \min\{|x|, \theta\}$ and

$$\begin{aligned} v_1 &:= \text{sign}(v) \max\{\theta, |v|\}, \\ v_2 &:= \text{sign}(v) \min\{\theta, (|v| - \lambda)_+\}. \end{aligned}$$

Then, we have

$$\text{prox}_{\lambda \min\{|\cdot|, \theta\}}(v) = \begin{cases} v_1 & \text{if } h_2(v_1) \leq h_2(v_2), \\ v_2 & \text{otherwise.} \end{cases}$$

Thus, we further have

$$\text{prox}_R(\mathbf{v}) = \left(\text{prox}_{\lambda \min\{|\cdot|, \theta\}}(\mathbf{v}_i)\right)_{1 \leq i \leq n}.$$