Proximal Splitting Meets Variance Reduction

Fabian Pedregosa
ETH Zürich and UC Berkeley[1]
USA and Switzerland

Kilian Fatras
Univ. Bretagne-Sud, CNRS, IRISA
INRIA Rennes and OBELIX

Mattia Casotto
Akur8
France

Abstract

Despite the raise to fame of stochastic variance reduced methods like SAGA and ProxSVRG, their use in non-smooth optimization is still limited to a few simple cases. Existing methods require to compute the proximal operator of the non-smooth term at each iteration, which, for complex penalties like the total variation, overlapping group lasso or trend filtering, is an iterative process that becomes unfeasible for moderately large problems. In this work we propose and analyze VR-TOS, a variance-reduced method to solve problems with an arbitrary number of non-smooth terms. Like other variance reduced methods, it only requires to evaluate one gradient per iteration and converges with a constant step size, and so is ideally suited for large scale applications. Unlike existing variance reduced methods, it admits multiple non-smooth terms whose proximal operator only needs to be evaluated once per iteration. We provide a convergence rate analysis for the proposed methods that achieves the same asymptotic rate as their full gradient variants and illustrate its computational advantage on 4 different large scale datasets.

1 Introduction

Stochastic variance reduced methods (Le Roux et al., 2012; Johnson and Zhang, 2013; Shalev-Shwartz and Zhang, 2013) have been recently proposed as an improved alternative to the venerable stochastic gradient descent (SGD) method (Robbins and Monro, 1951). As SGD, these methods only require to visit a small batch of random examples per iteration. This makes them ideally suited for large scale machine learning problems. Unlike SGD, the variance of the updates decreases to zero –hence the name– and converge with non-decreasing step sizes.

While initial stochastic variance reduced methods only considered smooth objectives, variants with support for a non-smooth term like ProxSVRG (Xiao and Zhang, 2014) and SAGA (Defazio et al., 2014) were soon developed. These methods are highly efficient whenever the nonsmooth part is *proximal*, that is, its proximal operator is available in closed form or at least fast to compute. This includes penalties such as the ℓ_1 or group lasso norm, but not more complex ones like the overlapping group lasso (Jacob et al., 2009), multidimensional total variation (Barbero and Sra, 2014) or trend filtering (Kim et al., 2009), to name a few.

A key observation is that many of these complex penalties can be decomposed as a sum of proximal terms. Proximal splitting methods like the three operator splitting (Davis and Yin, 2017) or the Condat-Vũ algorithm (Condat, 2013b; Vũ, 2013) then provide a principled approach to incorporate these penalties into the optimization. However, these methods require to compute the full gradient of the smooth term at each iteration, which can become costly in the context of large scale machine learning problems as it involves a full pass over the dataset. A question of key practical interest is whether these proximal splitting methods can be accelerated through the use of stochastic variance reduction techniques.

Our **main contribution** is the development and analysis of VR-TOS, a stochastic variance reduced method that can solve problems with a sum of proximal terms.

The proposed method bridges two previously distant families of algorithms and inherit the best of both: like the three operator splitting of Davis and Yin (2017), it can solve problems with multiple proximal terms, and like variance reduced stochastic methods its cost is independent on the number of smooth terms, converges with a fixed step size, and reaches the same asymptotic

[1]Currently at Google AI, Canada

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

convergence rate than full gradient methods. Furthermore, we also develop a sparse variant of the proposed algorithm which can take advantage of the sparsity in the input data. The paper is organized as follows:

- *Method.* §2 describes the VR-TOS algorithm, and extends it in §2.1 to leverage sparsity in the input data. §2.2 extends these methods to the case of an arbitrary number of proximal terms.
- *Analysis.* In §4 we provide a non-asymptotic convergence analysis of the proposed method. We show that, like other variance reduced methods, it converges with a fixed step size and can achieve the same asymptotic rate as the full gradient variants.
- *Experiments.* In §5 we compare the proposed method and related algorithms on a logistic regression problem with overlapping group lasso penalty on 4 datasets.

1.1 Definitions and notation

By convention, we denote vectors and vector-valued functions in lowercase boldface (e.g. $\boldsymbol{x}$) and matrices in uppercase boldface letters (e.g. $\boldsymbol{D}$). The proximal operator of a convex lower semicontinuous function h is defined as $\mathbf{prox}_{\gamma h}(\boldsymbol{x}) \stackrel{\text{def}}{=} \arg\min_{\boldsymbol{z}\in\mathbb{R}^p}\{h(\boldsymbol{z}) + \frac{1}{2\gamma}\|\boldsymbol{x}-\boldsymbol{z}\|^2\}$. We say a function f is L-smooth if it is differentiable and its gradient is L-Lipschitz, while it is μ-strongly convex if $f - \frac{\mu}{2}\|\cdot\|^2$ is convex.

We denote by $[\,\boldsymbol{x}\,]_b$ the b-th coordinate in $\boldsymbol{x}$. This notation is overloaded so that for a collection of blocks $T = \{B_1, B_2, \ldots\}$, $[\boldsymbol{x}]_T$ denotes the vector $\boldsymbol{x}$ restricted to the coordinates in the blocks of T. For convenience, when T consists of a single block B we use $[\boldsymbol{x}]_B$ as a shortcut of $[\boldsymbol{x}]_{\{B\}}$. Finally, we distinguish $\mathbb{E}$, the full expectation taken with respect to all the randomness in the system, from $\mathbf{E}$, the conditional expectation with respect to the random index sampled at iteration t, conditioned on all randomness up to iteration t.

2 Methods

In this section we present our main contribution, the variance reduced three operator splitting method. We will first consider problems with only two non-smooth terms, and generalize this formulation to an arbitrary number in §2.2.

We consider the following optimization problem

$$\begin{aligned}&\underset{\boldsymbol{x}\in\mathbb{R}^p}{\text{minimize}}\ f(\boldsymbol{x}) + g(\boldsymbol{x}) + h(\boldsymbol{x})\,,\\ &\text{with } f(\boldsymbol{x}) = \tfrac{1}{n}\textstyle\sum_{i=1}^n \psi_i(\boldsymbol{x}) + \omega(\boldsymbol{x})\end{aligned} \tag{OPT}$$

Algorithm 1: Variance Reduced TOS (VR-TOS)

Input: $\boldsymbol{y}_0 \in \mathbb{R}^p$, $\boldsymbol{\alpha}_0 \in \mathbb{R}^{n\times p}$, $\gamma > 0$
Temporary storage: $\boldsymbol{z}_t$, $\boldsymbol{v}_t$ and $\boldsymbol{x}_t$, all in $\mathbb{R}^p$
Result: approximate solution to (OPT)
for $t = 0, 1, \ldots$ **do**
  $\boldsymbol{z}_t = \mathbf{prox}_{\gamma h}(\boldsymbol{y}_t)$
  Sample $i \in \{1, \ldots, n\}$ uniformly at random
  $\boldsymbol{v}_t = \nabla\psi_i(\boldsymbol{z}_t) - \boldsymbol{\alpha}_{i,t} + \overline{\boldsymbol{\alpha}}_t + \nabla\omega(\boldsymbol{z}_t)$
  $\boldsymbol{x}_t = \mathbf{prox}_{\gamma g}(2\boldsymbol{z}_t - \boldsymbol{y}_t - \gamma\boldsymbol{v}_t)$
  $\boldsymbol{y}_{t+1} = \boldsymbol{y}_t + \boldsymbol{x}_t - \boldsymbol{z}_t$
  Update $\boldsymbol{\alpha}_{t+1}$ according to (1)
return $\boldsymbol{z}_t$

where each ψ_i is convex and L_ψ-smooth, ω is convex and L_ω-smooth and g, h are *proximal*, i.e., convex and we have access to their proximal operator.

This formulation allows to express a broad range of problems arising in machine learning and signal processing: the finite-sum includes common loss functions such as least squares or logistic loss; the two proximal terms g, h can be extended to an arbitrary number and include penalties such as the group lasso with overlap, total variation, ℓ_1 trend filtering, etc. Furthermore, the proximal terms can be extended-valued, thus allowing for convex constraints through the use of the indicator function. With respect to previous work, this significantly enlarges the class of functions stochastic variance reduced methods can solve efficiently.

We allow the terms inside the finite sum to be an addition of two terms: ψ_i and ω. This might seem superfluous since it is not more general than the standard formulation with a single term. However, in practice ψ_i (e.g., a least squares or logistic loss, see Appendix F.1) can be highly structured and allow for reduced storage schemes and/or have sparse gradients (see §2.1), properties which might not be shared by ω, (e.g., an ℓ_2 regularization term).

Central to our algorithm is the concept of q-memorization (Hofmann et al., 2015), which we recall below. It provides a convenient abstraction over common gradient memorization techniques like the ones in SAGA and SVRG.

Definition 1. *A* uniform q-memorization *algorithm selects at each iteration t a random index set J_t of memory terms to update according to*

$$\boldsymbol{\alpha}_{j,t+1} = \begin{cases}\nabla f_j(\boldsymbol{z}_t) & \text{if } j \in J_t\\ \boldsymbol{\alpha}_{j,t} & \text{otherwise}\,,\end{cases} \tag{1}$$

such that any j has the same probability q/n of being updated.

We now introduce the variance-reduced three operator

splitting (Vr-Tos), a method to solve problems of the form (OPT). It is specified in Algorithm 1 and takes as input a vector of coefficients $\boldsymbol{y}_0 \in \mathbb{R}^p$, a table $\boldsymbol{\alpha}_0 \in \mathbb{R}^{n\times p}$ to store previous gradients and a step size $\gamma > 0$. Although in the general case this table is required to be of size $n \times p$, for linearly-parametrized loss functions like the logistic or least squares loss this can be reduced to size n (Appendix F.1). Furthermore, the Svrg-like update detailed below avoids the need for this storage at the expense of a lightly increased per iteration cost.

The proposed method performs one evaluation of each of the proximal terms and builds the gradient estimator $\boldsymbol{v}_t$ from the table of previous gradients $\boldsymbol{\alpha}_t$ and the index i sampled uniformly at random. It is easy to see that $\boldsymbol{v}_t$ is an unbiased estimate of the gradient, that is, $\mathbf{E}\, \boldsymbol{v}_t = \nabla f(\boldsymbol{z}_t)$.

This method allows the memory terms to be updated using any scheme that verifies the q-memorization framework (line 8). Some common schemes are:

- Saga-*like update*. At each iteration, the algorithm updates the same coefficient that has been sampled, i.e. $J_t = \{i\}$. In this scheme each memory term has probability $1/n$ of being updated, and so $q = 1$.

- Svrg-*like update*. Fix parameter $q > 0$ and draw at each iteration r from a uniform distribution in the $[0, 1]$ interval. If $r < q/n$, the algorithm performs a complete update $\boldsymbol{\alpha}_{j,t+1} = \nabla\psi_j(\boldsymbol{z}_t)$ for all j, otherwise they are left unchanged.

  Like in the Svrg algorithm (Johnson and Zhang, 2013), it is possible to avoid storing the memory terms since the $\overline{\boldsymbol{\alpha}}_t$ is constant unless a full refresh is triggered. In this setting, only the p-dimensional vectors $\overline{\boldsymbol{\alpha}}_t$ and $\widetilde{\boldsymbol{z}}_t$ needs to be stored, where $\widetilde{\boldsymbol{z}}_t$ is the value of $\boldsymbol{z}_t$ last time a full refresh was triggered. This variant avoids the need to store $\boldsymbol{\alpha}_t$, at the cost of a slight per iteration cost, as $\boldsymbol{\alpha}_i = \nabla f_i(\widetilde{\boldsymbol{z}}_t)$ needs to be computed at each iteration.

  This memory update scheme was proposed by Hofmann et al. (2015), and unlike the original Svrg algorithm the number of iterates between two full regresh is a random variable instead of a fixed number of iterations.

2.1 Sparse Vr-Tos

Need for a sparse variant. Modern web-scale optimization problems that arise in machine learning are not only large, they are also often *sparse*. For example, in the `LibSVM` datasets suite[2], 8 out of the 11 datasets with more than a million samples have a density below 0.01%, and the largest one in number of samples has a density below 1 per million. Linearly-parametrized loss functions of the form $\psi_i(\boldsymbol{x}) = l_i(\boldsymbol{a}_i^T\boldsymbol{x})$ have a gradient of the form $\nabla\psi_i(\boldsymbol{x}) = \boldsymbol{a}_i l_i'(\boldsymbol{a}_i^T\boldsymbol{x})$, which inherits the same sparsity pattern as the data $\boldsymbol{a}_i$. Since the data might be extremely sparse, it is hence of great practical interest to leverage sparsity in the partial gradients. This is the case in generalized linear models such as least squares or logistic regression, where $\boldsymbol{a}_i$ are the rows of a data matrix.

In this subsection we assume that g and ω are block separable, i.e., can be decomposed block coordinate-wise as $g(\boldsymbol{x}) = \sum_{B\in\mathcal{B}} g_B([\boldsymbol{x}]_B)$ and $\omega(\boldsymbol{x}) = \sum_{B\in\mathcal{B}}\omega_B([\boldsymbol{x}]_B)$, where $\mathcal{B}$ is a partition of the coefficients into subsets which will call *blocks* and g_B, ω_B only depends on coordinates in block B. Furthermore, we will make use of the following notation:

- *Extended support.* We define the extended support of $\nabla\psi_i$, denoted T_i as the set of blocks of $\mathcal{B}$ that intersect with its support, formally defined as $T_i \overset{\text{def}}{=} \{B : \operatorname{supp}(\nabla f_i) \cap B \neq \varnothing,\, B \in \mathcal{B}\}$. For totally separable penalties such as the ℓ_1 norm, the blocks are individual coordinates and so the extended support covers the same coordinates as the support.

- *Reweighting constants.* Let $\boldsymbol{P}_i$ be the projection onto the extended support, i.e., the diagonal matrix where $[\boldsymbol{P}_i]_{B,B}$ is the identity if $B \in T_i$ and zero otherwise. For simplicity we assume that each block appears in at least one T_i, as otherwise the problem can be reformulated without it. For each block $B \in \mathcal{B}$ we define d_B as the inverse frequency of that block in the extended support, i.e. $d_B = 1/(\frac{1}{n}\sum_{i=1}^n \mathbb{1}\{B \in T_i\})^{-1}$. For notational convenience we define the block-diagonal matrix $\boldsymbol{D}$ as $[\boldsymbol{D}]_{B,B} = d_B\boldsymbol{I}$ for each block $B \in \mathcal{B}$. Note that by definition $\frac{1}{n}\sum_{i=1}^n \boldsymbol{P}_i = \boldsymbol{D}^{-1}$. Computation of this diagonal matrix should be done as a preprocessing step of the algorithm.

- The *scaled proximal operator* is defined for a function φ, step size $\gamma > 0$, positive definite matrix $\boldsymbol{H}$ and norm $\|\cdot\|_{\boldsymbol{H}}^2 \overset{\text{def}}{=} \langle\cdot, \boldsymbol{H}\cdot\rangle$ as

$$\mathbf{prox}_{\gamma\varphi}^{\boldsymbol{H}}(\boldsymbol{x}) \overset{\text{def}}{=} \operatorname*{arg\,min}_{\boldsymbol{z}\in\mathbb{R}^p} \left\{ \varphi(\boldsymbol{z}) + \frac{1}{2\gamma}\|\boldsymbol{x} - \boldsymbol{z}\|_{\boldsymbol{H}}^2 \right\} \quad (2)$$

We now have all necessary ingredients to present the sparse variant of Vr-Tos. This is specified in Algorithm 2. In this variant, all operations are restricted to the extended support.

The algorithm requires to compute the scaled proximal operators of g and h. By block separability of g its scaled proximal operator can be computed in

[2] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

Algorithm 2: Sparse VR-TOS

Input: $\boldsymbol{y}_0 \in \mathbb{R}^p$, $\boldsymbol{\alpha}_0 \in \mathbb{R}^{n\times p}$, $\gamma > 0$

Temporary storage: $\boldsymbol{z}_t$, $\boldsymbol{v}_t$ and $\boldsymbol{x}_t$, all in $\mathbb{R}^p$

Result: approximate solution to (OPT)

for $t = 0, 1, \ldots$ **do**

- Sample $i \in \{1, \ldots, n\}$ uniformly at random
- T_i = extended support of $\nabla\psi_i$
- $[\boldsymbol{z}_t]_{T_i} = [\mathbf{prox}_{\gamma h}^{\boldsymbol{D}^{-1}}(\boldsymbol{y}_t)]_{T_i}$
- $[\boldsymbol{v}_t]_{T_i} = [\nabla\psi_i(\boldsymbol{z}_t) - \boldsymbol{\alpha}_{i,t} + \boldsymbol{D}(\overline{\boldsymbol{\alpha}}_t + \nabla\omega(\boldsymbol{z}_t))]_{T_i}$
- $[\boldsymbol{x}_t]_{T_i} = [\mathbf{prox}_{\gamma\varphi_i}(2\boldsymbol{z}_t - \boldsymbol{y}_t - \gamma\boldsymbol{v}_t)]_{T_i}$
- $[\boldsymbol{y}_{t+1}]_{T_i} = [\boldsymbol{y}_t + \boldsymbol{x}_t - \boldsymbol{z}_t]_{T_i}$
- Update $\boldsymbol{\alpha}_{t+1}$ according to (1)

return $\mathbf{prox}_{\gamma h}^{\boldsymbol{D}^{-1}}(\boldsymbol{y}_t)$

block-wise as $[\mathbf{prox}_{\gamma g}^{\boldsymbol{D}^{-1}}(\boldsymbol{x})]_B = [\mathbf{prox}_{(d_B\gamma)h}(\boldsymbol{x})]_B$ for all $B \in \mathcal{B}$. Hence the cost of computing $[\boldsymbol{x}_t]_{T_i}$ will depend on the extended support size and not on the dimensionality.

We can unfortunately not guarantee the same complexity for $[\boldsymbol{z}_t]_{T_i}$ since we do not have a closed form for the scaled proximal operator of h in general. We review some specific cases in which it is possible to compute this scaled proximal operator in Appendix D. Alternatively, in the next subsection we propose a reformulation that avoids the need to compute this scaled proximal operator at the expense of higher memory usage.

In the case that one proximal term is zero, the proposed algorithm with SAGA-like update of the memory terms defaults to the Sparse SAGA variant of Pedregosa et al. (2017). With SVRG-like update of the memory terms it instead yields a novel sparse variant of ProxSVRG (Xiao and Zhang, 2014). For both of the proposed algorithms, when input is dense, $\boldsymbol{P}_i = \boldsymbol{D} = \boldsymbol{I}$ and we recover Algorithm 1.

2.2 Extension to an arbitrary number of proximal terms

The proposed method can be easily extended to the more general setting of an objective function with an arbitrary number of proximal terms of the form

$$\begin{aligned} &\underset{\boldsymbol{x}\in\mathbb{R}^p}{\text{minimize}}\ f(\boldsymbol{x}) + \textstyle\sum_{j=1}^k g_j(\boldsymbol{x})\,, \\ &\text{with } f(\boldsymbol{x}) = \tfrac{1}{n}\textstyle\sum_{i=1}^n \psi_i(\boldsymbol{x}) + \omega(\boldsymbol{x})\ , \end{aligned} \tag{OPT-k}$$

where ψ_i and ω are as in (OPT) and $g_1, \ldots, g_k$ are proximal. This is done by expressing the above as a problem of the form (OPT) in an enlarged space and then applying the proposed algorithm to this reformulation. For this, we will introduce k new variables which we will constrain to be equal via an indicator function. The above problem can be written equivalently as follows,

$$\min_{\boldsymbol{X}\in\mathbb{R}^{k\times p}} f(\overline{\boldsymbol{X}}) + \underbrace{\textstyle\sum_{j=1}^k g_j(\boldsymbol{X}_j)}_{\stackrel{\text{def}}{=} g(\boldsymbol{X})} + \underbrace{\imath\{\boldsymbol{X}_1 = \cdots = \boldsymbol{X}_k\}}_{\stackrel{\text{def}}{=} h(\boldsymbol{X})}\,,$$

where we have split the original variable into k variables $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k$ and constrained them to be equal using an indicator function in the last term. In this formulation the first term is smooth, and the other two terms are proximal. The second term is proximal since the variables in g_i are decoupled, each g_i is proximal by assumption and the last term is an indicator function over a linear subspace, and hence its scaled proximal operator can be computed in closed form as follows (Lemma 15):

$$\begin{aligned} &[\mathbf{prox}_{\gamma h}^{\boldsymbol{D}^{-1}}(\boldsymbol{X})]_{i,j} = \left(\textstyle\sum_{i=1}^n a_{i,j}\boldsymbol{X}_{i,j}\right) / \left(\textstyle\sum_{i=1}^n a_{i,j}\right) \\ &\text{with } a_{i,j} = \boldsymbol{D}^{-1}_{ip+j,ip+j}\ , \end{aligned} \tag{3}$$

Hence, the problem with multiple proximal terms (OPT-k) can be formulated as a problem with two proximal terms (OPT) and so it is possible to apply the proposed method defined in the previous subsections. This gives a variance reduced method for problems with an arbitrary number of proximal term. It is worth noting that for the sparse variants this formulation avoids the potentially difficult computation of the scaled proximal operator of h.

3 Related work

We comment on the most closely related ideas, summarized in Table 1.

Methods that support objective functions of the form (OPT) with two or more proximal terms and a smooth term accessed via its gradient have recently been proposed. Examples are the the primal-dual hybrid gradient method (also known as the Condat-Vũ) (Condat, 2013a; Vũ, 2013),[3] the generalized forward-backward splitting (Raguet et al., 2013) or the three operator splitting (Davis and Yin, 2017). Due to its excellent empirical performance and amenability to sparse updates we have chosen this last method as the basis for the proposed method. The proposed VR-TOS method can be seen as a generalization of this last method, as both method are identical when $n = 1$.

A different stochastic variant of the three operator splitting was proposed by Yurtsever et al. (2016) for the slightly more general case in which f is given by

[3] We note that this method can optimize the more general objective function $f(\boldsymbol{x}) + g(\boldsymbol{x}) + h(\boldsymbol{L}\boldsymbol{x})$, for an arbitrary linear operator $\boldsymbol{L}$ that is fixed to the identity in our setting.

Methods	incremental updates	non-decreasing step size	multiple non-smooth terms	sparse updates
Vr-Tos (*this work*)	✓	✓	✓	✓
Saga (Defazio et al., 2014)	✓	✓	✗	✓ (Pedregosa et al., 2017)
ProxSvrg (Xiao and Zhang, 2014)	✓	✓	✗	✗ †
Tos (Davis and Yin, 2017)	✗	✓	✓	N/A
Stochastic Tos (Yurtsever et al., 2016)	✓	✗	✓	✗

Table 1: **Comparison with related work.** The proposed method is unique in that it combines the advantages of variance-reduced methods (incremental updates, non-decreasing step sizes and sparse updates) with the advantages of proximal splitting (support for multiple non-smooth terms). †: a sparse variant of ProxSvrg follows as a special case of Algorithm 2 with $h = 0$ and the SVRG-like update of the memory terms.

an expectation. Like the proposed algorithms, this method only needs to evaluate the gradient of one element in the finite sum per iteration. Unlike the proposed methods, the variance of the updates does not decrease to zero and requires –as other non-variance reduced method– a decreasing step size. Furthermore, all updates are dense even in the presence of sparse gradients so the method performs poorly on large sparse problems.

(Balamurugan and Bach, 2016) proposed a variance-reduced method to solve problems a general class of saddle point problems including $\min_{\boldsymbol{x}} \max_{\boldsymbol{u}} \frac{1}{n}\sum_{i=1}^n f_i(\boldsymbol{x}) + M(\boldsymbol{x}, \boldsymbol{u})$, where $M(\cdot)$ is proximal. With $M(\boldsymbol{x}, \boldsymbol{u}) = g(\boldsymbol{x}) + \langle \boldsymbol{x}, \boldsymbol{u}\rangle - h^*(\boldsymbol{u})$, this is equivalent to the problem in (OPT). However, the method requires M to be strongly concave in $\boldsymbol{u}$, which is equivalent to h being smooth, and so is not applicable to the same class of problems as the proposed method. We note that this requirement is not merely an artifact of the theory, as the algorithm requires knowledge of this smoothness parameter.

Stochastic variance-reduced variants of ADMM have also been recently proposed, see e.g. (Zheng and Kwok, 2016; Yu and Huang, 2017). Compared to the proposed methods, none of the existing variants support sparse updates and require tuning more than one step-size parameter.

4 Analysis

In this section we provide a non-asymptotic convergence rate analysis for the proposed method:

- All the proposed variants converge with a step size $1/(3L_f)$, with $L_f \stackrel{\text{def}}{=} L_\psi + d_{\max} L_\omega$, where $d_{\max}$ is the maximum element in the diagonal matrix $\boldsymbol{D}$ ($d_{\max} = 1$ for non-sparse variants).
- For Vr-Tos (Algorithm 1) we obtain convergence rates that asymptotically match those of the full-gradient variant, i.e., $\mathcal{O}(1/t)$ convergence rate for convex problems (Theorem 1) and a linear convergence rate under strong convexity of f and smoothness of h (Theorem 3).
- For the sparse variant, Sparse Vr-Tos (Algorithm 2), we obtain a linear convergence rate under the same assumptions (Theorem 3). However, for general convex objectives we could only obtain a worse $\mathcal{O}(1/\sqrt{t})$ convergence rate (Theorem 2).

In this section we will use the following **extra notation**. We define the following primal ($\mathcal{P}$), and dual function ($\mathcal{D}$) as:

$$\mathcal{P}(\boldsymbol{x}) \stackrel{\text{def}}{=} f(\boldsymbol{x}) + g(\boldsymbol{x}) + h(\boldsymbol{x}) \,,$$
$$\mathcal{D}(\boldsymbol{u}) \stackrel{\text{def}}{=} (f + g)^*(-\boldsymbol{u}) + h^*(\boldsymbol{u}) \,, \tag{4}$$

where * denotes the Fenchel conjugate. We denote by $\boldsymbol{x}^\star$ an arbitrary minimizer of the primal objective and define the "dual iterate" $\boldsymbol{u}_t \stackrel{\text{def}}{=} \boldsymbol{D}^{-1}(\boldsymbol{y}_t - \boldsymbol{z}_t)/\gamma$ ($\boldsymbol{D} = \boldsymbol{I}$ for the dense variants). We also define the following generalized three operator splitting operator:

$$\boldsymbol{G}_\gamma(\boldsymbol{y}) \stackrel{\text{def}}{=} \boldsymbol{y} - \boldsymbol{z}_{\boldsymbol{y}} + \mathbf{prox}_{\gamma g}^{\boldsymbol{D}^{-1}}(2\boldsymbol{z}_{\boldsymbol{y}} - \boldsymbol{y} - \gamma \boldsymbol{D}\nabla f(\boldsymbol{z}_{\boldsymbol{y}})) \,,$$
$$\text{with } \boldsymbol{z}_{\boldsymbol{y}} = \mathbf{prox}_{\gamma h}^{\boldsymbol{D}^{-1}}(\boldsymbol{y}) \,, \tag{5}$$

and its set of fixed points, which we denote $\mathrm{Fix}(\boldsymbol{G}_\gamma)$. Another quantity that will appear often in the analysis is $H_0 \stackrel{\text{def}}{=} 1/(2nL_f)\sum_{i=1}^n \|\boldsymbol{\alpha}_{i,0} - \psi_i(\boldsymbol{x}^\star)\|^2$.

Throughout this section we make the following two technical assumptions:

Assumption 1: Regularity. We assume each ψ_i is L_ψ-smooth, ω is L_ω-smooth, g and h are proper (i.e., have nonempty domain), lower semicontinuous (i.e., its sublevel sets are closed) convex functions. We recall

that lower semicontinuity is a weak form of continuity that allows extended-valued functions with domain over a closed set.

Assumption 2: Qualification conditions. We assume the relative interior of $\operatorname{dom} g$ and $\operatorname{dom} h$ have a non-empty intersection. This is a very weak and standard assumption, which allows to rule out pathological cases such as disjoint domains and allows to relate the primal and dual optimal objective (see e.g.(Bauschke and Combettes, 2017, Proposition 15.13) or (Bertsekas, 2015, Proposition 5.3.8)), a property sometimes referred to as strong or total duality.

Sublinear convergence. The following theorem shows a $\mathcal{O}(1/t)$ convergence rate for VR-TOS on arbitrary convex objectives.

One of the issues when analyzing the convergence of the three operator splitting is that the objective function might be $+\infty$, for example when both proximal terms are an indicator function. Following Chambolle and Pock (2015); Pedregosa and Gidel (2018), we will state the convergence rate for general functions in terms of the *saddle point suboptimality*, defined as

$$\mathcal{L}(\widetilde{\boldsymbol{x}}, \boldsymbol{u}) - \mathcal{L}(\boldsymbol{x}, \widetilde{\boldsymbol{u}}) \,, \quad \text{with}$$
$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{u}) \stackrel{\text{def}}{=} f(\boldsymbol{x}) + g(\boldsymbol{x}) + \langle \boldsymbol{x}, \boldsymbol{u} \rangle - h^*(\boldsymbol{u}) \,, \tag{6}$$

where $\mathcal{L}$ is the Lagrangian associated with $\mathcal{P}$ and $\mathcal{D}$. As Davis and Yin (2017), we will also state convergence rates in terms of the objective suboptimality under a Lipschitz assumption on h in (8).

Theorem 1. *Let $\overline{\boldsymbol{x}}_t$ denote the averaged (also known as ergodic) iterate, i.e., $\overline{\boldsymbol{x}}_t = (\sum_{k=0}^t \boldsymbol{x}_k)/(t+1)$ and $\overline{\boldsymbol{u}}_t = (\sum_{k=0}^t \boldsymbol{u}_k)/(t+1)$. Then the* VR-TOS *method (Algorithm 1) converges for any step size $\gamma \leq 1/(3L_f)$, and for $\gamma = 1/(3L_f)$ we have the following bound for all $(\boldsymbol{x}, \boldsymbol{u}) \in \operatorname{dom} g \times \operatorname{dom} h^*$:*

$$\mathbb{E}\left[\mathcal{L}(\overline{\boldsymbol{x}}_t, \boldsymbol{u}) - \mathcal{L}(\boldsymbol{x}, \overline{\boldsymbol{u}}_t)\right] \leq \frac{10n}{q(t+1)} C_0, \tag{7}$$

with $\boldsymbol{y} = \boldsymbol{x} + \gamma \boldsymbol{u}$, $\boldsymbol{y}^\star \in \operatorname{Fix}(\boldsymbol{G}_\gamma)$, and $C_0 = \left[\frac{3L_f q}{20n}\|\boldsymbol{y}_0 - \boldsymbol{y}\|^2 + \frac{3L_f q}{2n}\|\boldsymbol{y}_0 - \boldsymbol{y}^\star\|^2 + H_0\right]$, where we recall $H_0 = 1/(2nL_f)\sum_{i=1}^n \|\boldsymbol{\alpha}_{i,0} - \psi_i(\boldsymbol{x}^\star)\|^2$.

Furthermore, if h is β_h-Lipschitz we have the following rate in terms of the primal objective:

$$\mathcal{P}(\overline{\boldsymbol{x}}_t) - \mathcal{P}(\boldsymbol{x}^\star) \leq \frac{10n}{q(t+1)} \widetilde{C}_0 \,, \tag{8}$$

with $\widetilde{C}_0 = \frac{6L_f q}{20n}\|\boldsymbol{z}_0 - \boldsymbol{x}^\star\|^2 + \frac{3L_f q}{2n}\|\boldsymbol{y}_0 - \boldsymbol{y}^\star\|^2 + \frac{q}{15nL_f}\beta_h^2 + H_0$.

The previous theorem gives a $\mathcal{O}(1/t)$ convergence rate in terms of the saddle point suboptimality for arbitrary convex functions and $\mathcal{O}(1/t)$ rate in function suboptimality under a Lipschitz assumption on h, matching the strongest bounds of SAGA (Defazio et al., 2014).

For their sparse variants, however, we have only been able to prove a slower $\mathcal{O}(1/\sqrt{t})$ rate on the operator residual, despite the fact that in practice the algorithm exhibits a much faster empirical convergence (see §5). Appendix B contains a characterization of the fixed points of this operator that justifies why this is a meaningful suboptimality criterion for (OPT). Although there is no direct correspondence between rates on the gradient and on objective values, lower bounds are asymptotically equivalent (Nesterov, 2012).

Theorem 2. *Sparse* VR-TOS *(Algorithm 2) converges for every step size $\gamma \leq 1/(3L_f)$. In particular, for $\gamma = 1/(3L_f)$ and $\boldsymbol{y}_t$ obtained after $t \geq 1$ updates we have the bound*

$$\min_{k=0,\dots,t} \{\mathbb{E}\|\boldsymbol{y}_k - \boldsymbol{G}_\gamma(\boldsymbol{y}_k)\|\} \leq \sqrt{\frac{C_0}{Lq(t+1)}} = \mathcal{O}\left(\frac{1}{\sqrt{t}}\right), \tag{9}$$

with $C_0 = \frac{5d_{\max}n}{Lq(t+1)}\left[(2Lq/n)\|\boldsymbol{y}_0 - \boldsymbol{y}^\star\|^2 + H_0\right]$.

Linear convergence. The three operator splitting has been shown to have a linear convergence rate under the assumption of strong convexity of the smooth term and smoothness of one of the proximal terms (Davis and Yin, 2015, §4.4). Although this last condition is rarely verified in practice since its main application is on non-smooth proximal terms, it is instructive to see that the proposed method –despite the reduced cost per iteration– also enjoys a linear convergence rate under the same assumptions.

Theorem 3 (Linear convergence)**.** *Let ψ_i be μ_ψ-strongly convex and ω be μ_ω-strongly convex, where $\mu_\psi + \mu_\omega > 0$. Furthermore, let h be L_h-smooth. Then for any step size $\gamma \leq 1/(3L_f)$, all the proposed methods converge geometrically in expectation. For $\gamma = 1/(3L_f)$, we have the following bound for Algorithm 1 ($d_{\max} = 1$ in this case) and Algorithm 2:*

$$\mathbb{E}\|\boldsymbol{z}_{t+1} - \boldsymbol{x}^\star\|^2 \leq \left(1 - \min\left\{\frac{q}{4n}, \frac{1}{3d_{\max}^3\delta^2\kappa}\right\}\right)^t D_0 \quad , \tag{10}$$

with $D_0 \stackrel{def}{=} d_{\max}\left[\frac{q}{2\gamma(1-\gamma\mu)n}\|\boldsymbol{y}_0 - \boldsymbol{y}^\star\|^2 + H_0\right]$, $\delta = (1 + L_h/(3L_f))$, $\kappa = L_f/\mu$ and $\boldsymbol{y}^\star \in \operatorname{Fix}(\boldsymbol{G}_\gamma)$.

4.1 Discussion

Comparison of convergence rates. We summarize the obtained convergence rates for the proposed methods and compare them against the best known rates for related stochastic methods in Table 2. In the linearly-convergent regime, we obtain rates that are

	Method	step size	Proximal oracle	Convergence rate	Extra assumptions
Geometric	SAGA (Defazio et al., 2014)	$1/3L_f$	$\mathbf{prox}_{\gamma(g+h)}$	$\left(1 - \min\left\{\frac{1}{4n}, \frac{1}{3\kappa}\right\}\right)^t C_0$	Each ψ_i is μ-cvx
	ProxSVRG (Xiao and Zhang, 2014)	$1/10L_f$	$\mathbf{prox}_{\gamma(g+h)}$	$\left(\frac{1}{\kappa 0.6m} + \frac{2}{3}\right)^t C_0$	f is μ-cvx
	VR-TOS (this work)	$1/3L_f$	$\mathbf{prox}_{\gamma g}$, $\mathbf{prox}_{\gamma h}$	$\left(1 - \min\left\{\frac{q}{4n}, \frac{1}{3d_{\max}^2 \delta^2 \kappa}\right\}\right)^t C_0$	Each ψ_i is μ-cvx and h is L_h-smooth
Sublinear	SAGA (Defazio et al., 2014)	$1/3L_f$	$\mathbf{prox}_{\gamma(g+h)}$	$\mathcal{O}(1/t)$	None
	Stochastic TOS (Yurtsever et al., 2016)	$\mathcal{O}(1/t)$	$\mathbf{prox}_{\gamma g}$, $\mathbf{prox}_{\gamma h}$	$\mathcal{O}(1/t)$	f is μ-cvx + bound on gradients
	VR-TOS (this work, dense/sparse variant)	$1/3L_f$	$\mathbf{prox}_{\gamma g}$, $\mathbf{prox}_{\gamma h}$	$\mathcal{O}(1/t)$ / $\mathcal{O}(1/\sqrt{t})$	None

Table 2: **Assumptions and properties of related incremental methods.** In every case, we take the step size recommended by the theory, where we assume $\omega = 0$ to make them comparable. Proximal oracle is the proximal operators that are needed by the algorithm. Extra assumptions refer to those other than Assumptions 1 and 2. The linear rates use the quantities $\delta = (1 + \gamma L_h)$, $\kappa = L_f/\mu$. For ProxSVRG, m denotes the epoch size and the convergence rate is relative to the number of epochs and not iterations like the rest.

similar to SAGA but with the rate factor multiplied by $1/(\delta^2 d_{\max}^3)$, quantity that depends on the smoothness of g and the sparsity of the gradients.

An improved ProxSVRG variant. The analysis of ProxSVRG (Xiao and Zhang, 2014) requires that the step size verifies an implicit equation that depends among other things on the strong convexity parameter. For typical choices of the parameters this is $1/(10L_f)$ (Xiao and Zhang, 2014, Theorem 1). In contrast, Sparse VR-TOS with SVRG-like sampling with $h = 0$ yields a variant of ProxSVRG with more favorable properties. First, none of its parameters depend on the strong convexity constant (while still obtaining a linear convergence rate since $L_h = 0$ in this case), which is most often unknown. Second, it admits the much larger step size $1/(3L_f)$, which is, to the best of our knowledge, the largest step size of any SVRG variant. Third, it can leverage sparsity in the input data through sparse updates.

Linear convergence without smoothness of the proximal term. Theorem 3 requires smoothness of one of the proximal terms to guarantee linear convergence. Despite this, linear convergence is observed in practice without this assumption (Figure 1). This has also been observed in the case of the original (non-variance reduced) three operator splitting (Davis and Yin, 2017; Pedregosa and Gidel, 2018), although an explanation for this is still an open problem. Furthermore, the lack of linear convergence when both proximal terms are non-smooth does not seem to be a limitation of the proof, as a counterexample was provided in (Davis and Yin, 2015, Appendix D.6). In this work, the authors constructed a strongly monotone operator with a sublinear convergence.

Step size adaptivity to linear convergence. A practical consequence of the above theorems is that using the same step size $\gamma = 1/(3L_f)$ we obtain a sublinear convergence by Theorem 1 and a linear rate (under additional assumptions) by Theorem 3. That is, one can use the "universal" step size $1/(3L_f)$ and automatically obtain linear convergence whenever the assumptions of Theorem 3 are verified.

Limitations. The following are some scenarios under which the proposed method is expected to perform poorly. The cost in computation and storage scales linearly with the number of proximal terms, hence it cannot cope with other scenarios with many non-smooth terms such as empirical risk minimization with the hinge loss or group lasso with overlap with a large number of overlaps (for instance > 100). Also, there are still penalties that cannot be reduced to a sum of proximal terms, such as the nuclear norm. Algorithms based on Frank-Wolfe (Jaggi, 2013) or with approximate proximal operators (Schmidt et al., 2011) might be better suited in such regimes.

5 Experiments

Although the proposed methods can be applied more broadly, we consider for the experiments a logistic regression problem with squared ℓ_2 regulrization and an overlapping group lasso penalty (Jacob et al., 2009). Following Jacob et al. (2009) we choose groups of 10 variables with 2 variables of overlap between two successive groups: $\{\{1, \dots, 10\}, \{8, \dots, 18\}, \{16, \dots, 26\}, \dots\}$. The amount of group regularization was chosen such that the solution has roughly 10% of non-zero coefficients and the of ℓ_2 regularization was fixed to $1/n$. We consider the following methods:

Dataset	#samples	#dimensions	density	L_f/μ
RCV1 (full) (Lewis et al., 2004)	697,641	47,236	1.5×10^{-3}	2.50×10^{4}
URL (Ma et al., 2009)	2,396,130	3,231,961	3.5×10^{-5}	1.28×10^{7}
KDD10 (Yu et al., 2010)	19,264,097	29,890,095	9.8×10^{-7}	5.2×10^{8}
Criteo (Juan et al., 2016)	45,840,617	1,000,000	3.8×10^{-5}	1.1×10^{7}

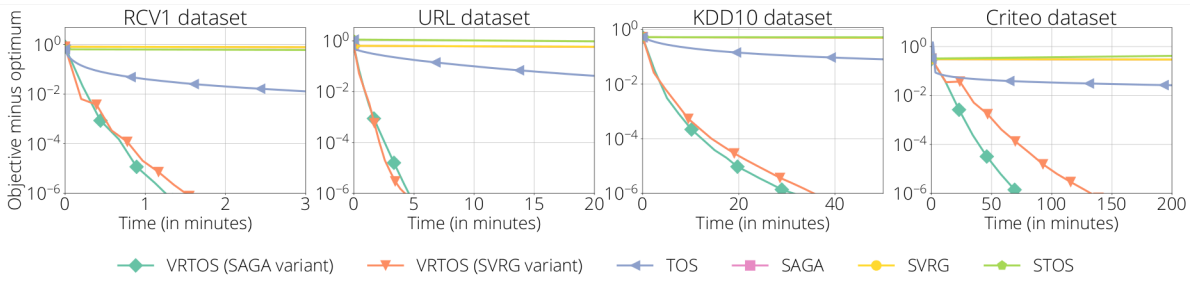

Figure 1: **Top**: Description of considered datasets. **Bottom:** Suboptimality vs time of different algorithms on a logistic regression with overlapping group lasso penalty problem.

- The proposed method Sparse Vr-Tos (Algorithm 2), where the overlapping group lasso penalty is split as a sum of two non-overlapping group lasso penalties, for which the proximal operator is available in closed form. We used the formulation with 3 proximal terms of §2.2 to better leverage sparsity in the dataset and consider Saga and Svrg-like updates, denoted Vr-Tos (Saga variant) and Vr-Tos (Svrg variant) respectively. This implementation is publicly available in the C-OPT package.[4]

  It is worth noting that while original penalty is *not* block separable, each of the terms in the splitting as two group lasso penalties *is* block separable. This will allow us to make a much more efficient use of sparsity than what is possible on on methods like Saga and ProxSvrg.

- The three operator splitting (denoted Tos), in its recently proposed variant with adaptive step size (Pedregosa and Gidel, 2018).

- The stochastic three operator spitting of (Yurtsever et al., 2016) with the same splitting as Vr-Tos, denoted Stos.

- Saga and ProxSvrg, where the proximal operator is evaluated approximately using 10 iterations of the Douglas-Rachford method.

The above methods were compared on 4 large-scale datasets described in the table of Figure 1. Further details and implementation aspects are discussed in Appendix F.1.

The best performing algorithms overall are the proposed Vr-Tos variants, which are over an order of magnitude faster than the second best method, the adaptive three operator splitting. The stochastic three operator splitting, not being able to take advantage of the sparsity in the gradients, performs poorly in this benchmark, appearing as a straight line. Saga and ProxSvrg were the slowest since they require to compute a costly proximal operator at each iteration and are unable to leverage the sparsity of the dataset due to the non-block-separability of the non-smooth term.

It is worth noting from Figure 1 that the two variants of Sparse Vr-Tos exhibit an empirical linear convergence, despite the fact that the theory only predicts in this regime a much slower $\mathcal{O}(1/\sqrt{t})$ convergence rate (Theorem 1).

We provide extra experiments in Appendix F.2.

6 Future work

This work can be extended in several ways. As highlighted in §4.1, a theoretical explanation for the empirical linear convergence without smoothess of any proximal term, even for the full gradient algorithm, is lacking. We conjecture *partly smooth* is a sufficient condition on the penalties to ensure local linear convergence, as recently proven for related methods (Liang et al., 2018). Second, we conjecture that the convergence rate of the sparse variant can be improved up to to $\mathcal{O}(1/t)$. A third direction for future work would be the development an extension that allow for a linear operator inside one of the proximal terms, as in (Condat, 2013b; Zhao and Cevher, 2018; Yan, 2018).

[4] http://openopt.github.io/copt/

Acknowledgements

The authors warmly thank Vincent Roulet, Vlad Niculae, Rémi Leblond and Federico Vaggi for their feedback on the manuscript, as well as Adrien Taylor, Alexandre D'Aspremont, Gabriel Peyré, Guillaume Obozinski, P. Balamurugan, Francis Bach and Marwa El Halabi for fruitful discussions.

This work has been done while FP was under funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodorowska-Curie grant agreement 748900. KF is funded through the project OATMIL ANR-17-CE23-0012 of the French National Research Agency (ANR). Computing time on was donated by Amazon through the program "AWS Cloud Credits for Research".

References

Balamurugan, P. and Bach, F. (2016). Stochastic Variance Reduction Methods for Saddle-Point Problems. *Advances in Neural Information Processing Systems*.

Barbero, Á. and Sra, S. (2014). Modular proximal optimization for multidimensional total-variation regularization. *arXiv preprint arXiv:1411.0589*.

Bauschke, H. H., Boţ, R. I., Hare, W. L., and Moursi, W. M. (2012). Attouch–Théra duality revisited: paramonotonicity and operator splitting. *Journal of Approximation Theory*.

Bauschke, H. H. and Combettes, P. L. (2017). *Convex analysis and monotone operator theory in Hilbert spaces*. Springer Science & Business Media.

Bertsekas, D. P. (2015). *Convex optimization algorithms*. Athena Scientific Belmont.

Chambolle, A. and Pock, T. (2015). On the ergodic convergence rates of a first-order primal–dual algorithm. *Mathematical Programming*.

Condat, L. (2013a). A primal–dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms. *Journal of Optimization Theory and Applications*.

Condat, L. (2013b). A direct algorithm for 1D total variation denoising. *IEEE Signal Processing Letters*.

Davis, D. and Yin, W. (2015). A three-operator splitting scheme and its optimization applications. *preprint arXiv:1504.01032v1*.

Davis, D. and Yin, W. (2017). A three-operator splitting scheme and its optimization applications. *Set-Valued and Variational Analysis*.

Defazio, A., Bach, F., and Lacoste-Julien, S. (2014). SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*.

Giselsson, P. and Boyd, S. (2016). Linear Convergence and Metric Selection in Douglas-Rachford Splitting and ADMM. *IEEE Transactions on Automatic Control*.

Hofmann, T., Lucchi, A., Lacoste-Julien, S., and McWilliams, B. (2015). Variance Reduced Stochastic Gradient Descent with Neighbors. In *Advances in Neural Information Processing Systems*.

Iusem, A. N. (1998). On Some Properties of Generalized Proximal Point Methods for Variational Inequalities. *Journal of Optimization Theory and Applications*.

Jacob, L., Obozinski, G., and Vert, J.-P. (2009). Group lasso with overlap and graph lasso. In *Proceedings of the 26th annual international conference on machine learning*. ACM.

Jaggi, M. (2013). Revisiting Frank-Wolfe: projection-free sparse convex optimization. In *International Conference on Machine Learning*.

Johnson, N. (2013). A dynamic programming algorithm for the fused lasso and L_0-segmentation. *Journal of Computational and Graphical Statistics*.

Johnson, R. and Zhang, T. (2013). Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*.

Juan, Y., Zhuang, Y., Chin, W.-S., and Lin, C.-J. (2016). Field-aware factorization machines for CTR prediction. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM.

Kim, S.-J., Koh, K., Boyd, S., et al. (2009). ℓ_1 trend filtering. *SIAM review*.

Le Roux, N., Schmidt, M., and Bach, F. (2012). A stochastic gradient method with an exponential convergence rate for finite training sets.

Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. (2004). RCV1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397.

Liang, J., Fadili, J., and Peyré, G. (2018). Local linear convergence analysis of primal–dual splitting methods. *Optimization*.

Ma, J., Saul, L. K., et al. (2009). Identifying suspicious URLs: an application of large-scale online learning. In *Proceedings 26th ACM international conference on machine learning*.

Nesterov, Y. (2004). *Introductory lectures on convex optimization*. Springer.

Nesterov, Y. (2012). How to make the gradients small. *Optima*.

Pedregosa, F. and Gidel, G. (2018). Adaptive Three Operator Splitting. *Proceedings of the 35th International Conference on Machine Learning*.

Pedregosa, F., Leblond, R., and Lacoste-Julien, S. (2017). Breaking the Nonsmooth Barrier: A Scalable Parallel Method for Composite Optimization. *Advances in Neural Information Processing System 30 (NIPS)*.

Raguet, H., Fadili, J., and Peyré, G. (2013). A generalized forward-backward splitting. *SIAM Journal on Imaging Sciences*.

Robbins, H. and Monro, S. (1951). A Stochastic Approximation Method. *Ann. Math. Statist.*

Rockafellar, R. T. (1997). Convex analysis.

Rockafellar, R. T. and Wets, R. J.-B. (1998). *Variational analysis*. Springer.

Schmidt, M., Le Roux, N., and Bach, F. (2011). Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in neural information processing systems 24*.

Shalev-Shwartz, S. and Zhang, T. (2013). Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*.

Vũ, B. C. (2013). A splitting algorithm for dual monotone inclusions involving cocoercive operators. *Advances in Computational Mathematics*.

Xiao, L. and Zhang, T. (2014). A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*.

Yan, M. (2018). A New Primal–Dual Algorithm for Minimizing the Sum of Three Functions with a Linear Operator. *Journal of Scientific Computing*.

Yu, H.-F., Lo, H.-Y., Hsieh, H.-P., Lou, J.-K., McKenzie, T. G., Chou, J.-W., Chung, P.-H., Ho, C.-H., Chang, C.-F., Wei, Y.-H., et al. (2010). Feature engineering and classifier ensemble for KDD cup 2010. In *KDD Cup*.

Yu, Y. and Huang, L. (2017). Fast stochastic variance reduced admm for stochastic composition optimization. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)*.

Yurtsever, A., Vu, C. B., and Cevher, V. (2016). Stochastic Three-Composite Convex Minimization. In *Advances in Neural Information Processing Systems*.

Zhao, R. and Cevher, V. (2018). Stochastic Three-Composite Convex Minimization with a Linear Operator. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*.

Zheng, S. and Kwok, J. T. (2016). Stochastic variance-reduced ADMM. *arXiv preprint arXiv:1604.07070*.