Target-based Surrogates for Stochastic Optimization

author names withheld

Under Review for OPT 2022

Abstract

We consider minimizing functions for which it is computationally expensive to query the (stochastic) gradient. Such functions are prevalent in applications like reinforcement learning, online imitation learning and bilevel optimization. We exploit the composite structure in these functions and propose a *target optimization* framework. Our framework leverages the smoothness of the loss with respect to an intermediate target space (e.g. the output of a neural network model), and uses gradient information to construct surrogate functions. In the full-batch setting, we prove that the surrogate function is a global upper-bound on the overall loss, and can be (locally) minimized using any black-box optimization algorithm. We prove that the resulting majorization-minimization algorithm ensures convergence to a stationary point at an $O\left(1/T\right)$ rate, thus matching gradient descent. In the stochastic setting, we propose a stochastic surrogate optimization (`SSO`) algorithm that can be viewed as projected stochastic gradient descent in the target space. We leverage this connection in order to prove that `SSO` can match the SGD rate for strongly-convex functions. Experimentally, we evaluate the `SSO` algorithm on convex supervised learning losses and show competitive performance compared to SGD and its variants.

1. Introduction

Stochastic gradient descent (SGD) and its variants [6, 9, 16] are ubiquitous optimization methods in machine learning (ML). In supervised learning, these iterative first-order methods only require computing the gradient over individual mini-batches of examples, resulting in computationally inexpensive iterations. In general, the computational cost of such algorithms is dictated by the cost of computing the gradient. In deep learning, this cost can be due to transferring large amounts of data to/from the disk. On the other hand, for policy optimization in reinforcement learning (RL) or online imitation learning (OIL) [17] gathering data via interaction with the environment constitutes the main computational cost. Stochastic policy gradient methods [21, 25], first-order methods inspired by SGD, are routinely used for these tasks. Common policy gradient approaches such as TRPO [19] and PPO [20] use a single stochastic gradient to construct a surrogate function that is minimized to make multiple policy updates. Similarly, in OIL, methods based on the follow-the-leader algorithm [10, 17] from online convex optimization [15] allow multiple policy updates without

© .

additional interactions with the simulator. Such applications, where access to the (stochastic) gradient oracle is expensive, are the main motivation for our work.

Specifically, we consider algorithms that access the (stochastic) gradient oracle to construct a sequence of surrogate functions that can then be optimized efficiently. Incrementally forming a surrogate and minimizing it is a prevalent technique in optimization. It is commonly referred to as *majorization-minimization* and includes the popular Expectation-Maximization (EM) algorithm [5] as an example. Surrogate functions are often formed by using the convexity and/or smoothness properties of the underlying function, and have been used in the general stochastic setting [11]. Moreover, methods such as MISO [12] construct surrogate functions by effectively exploiting the finite-sum structure in typical loss functions. Similarly, we leverage the composite structure of loss functions prevalent in ML. For example, in supervised learning, common loss functions are of the form $h(\theta) = \ell(f(\theta))$ where ℓ is (usually) a convex loss (for example, the squared loss for regression or the logistic loss for classification), while the mapping f corresponds to some transformation of the features. Such transformations might be linear, or complex, high-dimensional, non-convex (as in the case of neural networks). On the other hand, in OIL, ℓ measures the divergence between the policy being learned and the expert policy, and f is a potentially complicated representation of the policy in terms of the parameters to be learned. We exploit this composite structure in order to form surrogate functions. In particular, we make the following contributions.

Target smoothness surrogate: For supervised learning using a dataset of n input-label pairs, we consider minimizing the loss function $h(\theta) = \ell(f(\theta))$. Here, $f : \Theta \to \mathcal{Z}$ is a potentially non-convex mapping from the feasible set of parameters ($\Theta \in \mathbb{R}^d$) to the *target space* ($\mathcal{Z} \in \mathbb{R}^n$) and $\ell : \mathcal{Z} \to \mathbb{R}$ is a convex loss function. In order to exploit this composite structure, we disentangle the parametric space Θ from the target space $\mathcal{Z}$ and use the smoothness of $\ell(z)$ in order to define a *target smoothness surrogate* (Section 3.1).

Target optimization in the deterministic setting: In Section 3.1, we prove the target smoothness surrogate is a global upper-bound on h. Using this property, we devise a majorization-minimization algorithm: alternatively form a target smoothness surrogate and (locally) minimize it using any black-box algorithm. Importantly, though forming the surrogate requires access to the gradient oracle, it can be minimized without additional oracle calls resulting in computationally efficient updates. We refer to this framework as *target optimization*. In Algorithm 1, we instantiate target optimization by using m steps of gradient descent to approximately minimize the surrogate. Assuming that the surrogate is smooth, we prove that Algorithm 1 converges to the stationary point of h for *any* value of $m \geq 1$ and matches the $O\left(\frac{1}{T}\right)$ rate of gradient descent (Lemma 1). Importantly, this result only requires both ℓ and f to be smooth and does not require any convexity assumptions.

Stochastic target smoothness surrogate: In Section 3.2, we define the *stochastic target smoothness surrogate*. For this, we sample function ℓ_t from the dataset uniformly at random, and form the surrogate similar to the deterministic setting. Importantly, although the surrogate is formed by sampling ℓ_t and is therefore random, once the surrogate is formed, it can be minimized using any black-box deterministic optimization algorithm. In this case,

target optimization has the favourable property of disentangling the stochasticity in ℓ (in the target space) from the possible non-convexity in f (in the parametric space).

Target optimization in the stochastic setting: Similar to the deterministic setting, the stochastic target smoothness surrogate can be minimized using m steps of GD and denote the resulting algorithm as stochastic surrogate optimization `SSO`. In order to prove theoretical guarantees for `SSO`, we interpret (approximately) minimizing the surrogate as (inexact) projected SGD in $\mathbb{R}^n$, where the projection is onto the target space $\mathcal{Z}$. In Section 4, we prove that for smooth, strongly-convex ℓ, the iterate converges linearly to the neighbourhood of the minimizer. The size of this neighbourhood depends on the noise in the stochastic gradients and the projection error. Moreover, we prove that $O \log(1/\epsilon)$ iterations of surrogate minimization are sufficient to ensure an $O(\epsilon)$ projection error.

Experimental evaluation: We empirically compared the stochastic surrogate optimization algorithm (`SSO`) to both SGD with a theoretically chosen step-size, as well as a stochastic line-search (SLS) [22]. Our experiments across a variety of batch sizes, datasets and losses demonstrate that `SSO` is competitive compared to SGD and its variants.

2. Problem Formulation

We consider minimizing a finite-sum of composite functions typically used in machine learning. For example, in supervised learning, linear regression uses the loss function: $h(\theta) = \frac{1}{2}\|X\theta - y\|_2^2 = \sum_{i=1}^n \frac{1}{2}\|X_i\theta - y\|_2^2$ where $X \in \mathbb{R}^{n\times d}$ is the matrix of d-dimensional features for n data-points, X_i is its i-th row, $y \in R^n$ is the vector of the corresponding measurements and $\theta \in R^d$ is the set of parameters to be learned from the data. Similarly, for logistic regression, the loss function is given as: $h(\theta) = \sum_{i=1}^n \log\left(1 + \exp(-y_i\langle X_i, \theta\rangle)\right)$. Both losses are examples of a finite-sum of individual loss-functions (one for each data-point i), and each individual loss function is a composition of $\ell_i(f_i(\theta))$. For example, for linear regression, $f_i(\theta) = X_i\theta$ and $\ell_i(z_i) = \frac{1}{2}(z_i - y_i)^2$. Similarly, when using models such as neural networks, the function f mapping the features X to the labels y is nonlinear, high-dimensional and non-convex. These models are used in conjunction with convex losses such as the squared or logistic loss defined above and also result in minimizing a finite-sum of composite functions. Throughout this paper, we will restrict ourselves to minimizing functions of the form: $\min_{\theta\in\Theta} h(\theta) := \sum_{i=1}^n \ell_i(f_i(\theta))$, where Θ is the feasible set for the parameters. There exists a long line of work in optimization that exploits the finite-sum structure in $h(\theta)$ [4, 7, 18] and result in faster convergence rates. However, there are relatively few works [14] that make use of the additional composite structure in $h(\theta)$. In order to exploit this structure, we disentangle the *parametric space* $\Theta \subseteq \mathbb{R}^d$ from the *target space* $\mathcal{Z} \subseteq R^n$. The function $f : \Theta \to \mathcal{Z}$ is therefore a mapping between the parametric and target spaces and for all $z \in \mathcal{Z}$, $\exists\theta \in \Theta$ such that $z = f(\theta)$. Function ℓ and h are defined as: $\ell : \mathcal{Z} \to \mathbb{R}$ and $h : \Theta \to \mathbb{R}$.

Throughout this paper, we will assume that h is L_θ-smooth and ℓ is an L-smooth function of z. We will also assume that ℓ is a (strongly-)convex function but f (and therefore h) can

be non-convex. Though our algorithms can handle non-convex f (and h), for our theoretical results, we will assume that f is a linear map in θ and hence h is (strongly-)convex in θ. For example, for all generalized linear models including linear and logistic regression, $f = X\theta$ is a linear. thus for linear regression, $\ell(z) = \frac{1}{2}\|z - y\|_2^2$ is convex, $L = 1$ and $L_\theta = \lambda_{\max}[X^\top X]$ (where $\lambda_{\max}[A]$ is the maximum eigenvalue of matrix A).

Iterative first-order methods such as gradient descent (GD) or stochastic gradient descent (SGD) are typically used to minimize $h(\theta)$. At iteration $t \in [T]$, the *parametric GD* update is: $\theta_{t+1} = \theta_t - \eta \nabla h_t(\theta_t)$ where η is the step-size to be selected or tuned according to the properties of h. If h is L_θ-smooth, then each iteration of parametric GD can be viewed as exactly minimizing the quadratic surrogate function derived from the smoothness condition. Specifically, $\theta_{t+1} := \arg\min g_t^\theta = \left[h(\theta_t) + \langle \nabla h(\theta_t), \theta - \theta_t\rangle + \frac{1}{2\eta}\|\theta - \theta_t\|_2^2\right]$ where g_t^θ is referred to as the *parametric smoothness surrogate*. The surrogate is tight at θ_t i.e $h(\theta_t) = g_t^\theta(\theta_t)$ and becomes loose away from θ_t. For $\eta \le \frac{1}{L_\theta}$, the surrogate is a global (for all θ) upper-bound on h i.e. $g_t^\theta(\theta) \ge h(\theta)$. Minimizing the global upper-bound results in descent on h since, $h(\theta_t) = g_t^\theta(\theta_t) \ge g_t^\theta(\theta_{t+1}) \ge h_t(\theta_{t+1})$. Note that GD does not exploit the finite-sum or composite structure of h.

On the other hand, (parametric) SGD exploits the finite-sum structure in h, and at iteration t, SGD selects a function h_t (corresponding to the index $i_t \in [n]$ sampled typically uniformly), computes its gradient and takes a descent step in that direction. The update is given as: $\theta_{t+1} = \theta_t - \eta_t \nabla h_t(\theta_t)$ where $\mathbb{E}_i\left[\nabla h_t(\theta)|\theta_t\right] = \nabla h(\theta)$ and η_t is typically a decreasing step-size. Similar to GD, each iteration of SGD can be viewed as exactly minimizing a *stochastic parametric smoothness surrogate*: $\tilde{g}_t^\theta(\theta) := \left[h_t(\theta_t) + \langle \nabla h_t(\theta_t), \theta - \theta_t\rangle + \frac{1}{2\eta_t}\|\theta - \theta_t\|_2^2\right]$.

These algorithms exploit smoothness in the parameter space to form the surrogate function. However, none of these methods exploit the composite structure of the loss functions to form surrogates and design better algorithms. In the next section we exploit this structure and introduce a majorization-minimization algorithm in the target space.

3. Methodology

We first consider minimizing the composite function h in the deterministic setting Section 3.1, and study the stochastic finite-sum setting in Section 3.2.

3.1. Deterministic setting

We take advantage of the composite structure in h and use the smoothness of ℓ in the target space $\mathcal{Z}$ in order to form the surrogate function to be minimized at each iteration. At iteration t, for a step-size η to be determined, we define the *target smoothness surrogate* analogous to g_t^θ as $\ell(z_t) + \langle \nabla_z \ell(z_t), z - z_t\rangle + \frac{1}{2\eta}\|z - z_t\|_2^2$. Since $z = f(\theta)$, the surrogate can be written as: a function of θ as:

$$g_t^z(f(\theta)) = \ell(z_t) + \langle \nabla_z \ell(z_t), f(\theta) - f(\theta_t)\rangle + \frac{1}{2\eta}\|f(\theta) - z_t\|_2^2 . \tag{1}$$

For example, for linear regression, $g_t^z(\theta) = \frac{1}{2}\|X\theta_t - y\|_2^2 + \left\langle [X\theta_t - y], X(\theta - \theta_t)\right\rangle + \frac{1}{2\eta}\|X(\theta - \theta_t)\|_2^2$. Similar to GD, $h(\theta_t) = g_t^z(f(\theta_t))$ and if ℓ is L-smooth in the target space $\mathcal{Z}$, then for $\eta \leq \frac{1}{L}$, $g_t^z(f(\theta)) \geq h(\theta)$ for all θ i.e. the surrogate is a global upper-bound on h. Since g_t^z is a global upper-bound on h, similar to GD, we can minimize h by minimizing the surrogate at each iteration, which suggests the following algorithm: $\theta_{t+1} := \arg\min_\theta g_t^z(\theta)$. However, unlike GD, in general, there is no closed form solution for the minimizer of g_t^z, and we will consider minimizing it approximately. This results in the following meta-algorithm: For $t \in [T]$, at iterate θ_t, form the surrogate g_t^z and compute θ_{t+1} by (approximately) minimizing $g_t^z(\theta)$. The meta-algorithm enables the use of any black-box algorithm to minimize the surrogate at each iteration. We can instantiate this meta-algorithm by considering minimizing $g_t^z(\theta)$ using $m \geq 1$ steps of gradient descent. This results in Algorithm 1 in Appendix B. For $m = 1$, Algorithm 1 results in the following update: $\theta_{t+1} = \theta_t - \alpha\nabla g_t^z(\theta_t) = \theta_t - \alpha\nabla h(\theta_t)$ (since $g_t^z(\theta_t) = h(\theta_t)$. Hence, for 1 GD step on g_t^z, Algorithm 1 is equivalent to parametric GD with a step-size α.

Again, let us instantiate the above meta-algorithm for linear regression. For $m = 1$, we recover GD on the least squares objective. On the other hand, minimizing the surrogate exactly (corresponding to $m = \infty$) to compute θ_{t+1} results in the following equation: $X^\top[X\theta_t - y] + X^\top X(\theta_{t+1} - \theta_t) = 0 \implies \theta_{t+1} = (X^\top X)^{-1}[X^\top(X\theta_t - y)]$, and recovers the Newton update in the parameter space. Hence, for linear regression, approximately minimizing g_t^z (using $m \in (1, \infty)$) interpolates between a first and second-order method in the parameter space. In Lemma 1, we show that Algorithm 1 with any value of $m \geq 1$ and an appropriate step-size α can result in descent on the surrogate g_t^z and hence on the composite function h. Importantly, this result does not require the function f or the surrogate g_t^z to be convex in θ and is true for non-convex models such as neural networks. Hence, by iteratively forming and approximately minimizing g_t^z, we are guaranteed convergence to a stationary point of $h(\theta)$. This general idea has been recently explored in the context of designing efficient off-policy algorithms for reinforcement learning [23].

3.2. Stochastic setting

In order to minimize the finite-sum in the stochastic setting, a natural idea is to form a stochastic surrogate similar to SGD. In particular, at iteration t, for the randomly sampled loss function ℓ_t, we form the *stochastic target surrogate* as follows:

$$\tilde{g}_t^z(\theta) := \ell_t(z_t) + \langle \nabla_f \ell_t(z_t), f(\theta) - z_t\rangle + \frac{1}{2\eta_t}\|f(\theta) - z_t\|_2^2 \tag{2}$$

where η_t is the iteration dependent step-size. In order to instantiate $\tilde{g}_t^z$ for linear regression, define $z^j = X_j\theta$ as coordinate j of $z \in \mathbb{R}^n$ and $z_t^j := X_j\theta_t$. Using this definition, $\ell_i(z) = \sum_{j=1}^n \left[\frac{\mathcal{I}\{j=i\}}{2}(z^j - y_i)^2\right]$ where $\mathcal{I}$ is the indicator function, and $\ell(z) = \sum_{i=1}^n \ell_i(z)$. In this case, if $(X_t, y_t) \in \mathbb{R}^d \times \mathbb{R}$ is the input-label pair corresponding to the loss function sampled at iteration t, then, for linear regression, $\tilde{g}_t^z(\theta) = \frac{1}{2}(X_t\theta_t - y_t)^2 + \left\langle [X_t\theta_t - y], X_t(\theta - \theta_t)\right\rangle +$

$\frac{1}{2\eta_t}\|X_t(\theta-\theta_t)\|_2^2$. Note that writing ℓ_i in this form allows us to leverage the separable structure of ℓ w.r.t to each z_i by only requiring that we sample ℓ_i, and not z. In other words we can use full vector z in each update, but the gradient of ℓ_i w.r.t. z_j $j \neq i$ is zero. Algorithmically, we can form the surrogate $\tilde{g}_t^z$ at iteration t, and minimize it approximately by using any black-box algorithm. Similar to the deterministic setting, we can minimize $\tilde{g}_t^z$ using m steps of GD. The resulting algorithm is the same as Algorithm 1 but using $\tilde{g}_t^z$ instead of g_t. It is important to note that the surrogate is formed by selecting function ℓ_t randomly and is therefore random, however, once the surrogate is formed, it can be minimized using any deterministic algorithm i.e. there is no additional randomness in the inner-loop in Algorithm 1. Moreover, for the special case of $m = 1$, Algorithm 1 has the same update as parametric stochastic gradient descent. Hence, our framework encompasses the standard SGD approach.

Theoretically, for $\eta_t \leq \frac{1}{L}$ for all t, $\tilde{g}_t^z$ is a global upper-bound on h_t, and minimizing $\tilde{g}_t$ guarantees descent on h_t. However, this is not enough to guarantee descent on h. In order to prove theoretical guarantees, we interpret minimizing $\tilde{g}_t^z$ as projected SGD in the target space. The corresponding update can be written as:

$$z_{t+1/2} = z_t - \eta_t \nabla_f \ell_t(z_t) \quad ; \quad \bar{z}_{t+1} = \arg\min_{z\in\mathcal{Z}} \frac{1}{2}\left\|z_{t+1/2} - z\right\|_2^2. \tag{3}$$

Starting from iterate $z_t \in \mathcal{Z}$, the SGD update might result in an iterate $z_{t+1/2}$ that is not realizable by the class of parametric functions f. In other words, there may not exist a θ for which $z_{t+1/2} = f(\theta)$, and hence we need to project onto $\mathcal{Z}$. Recall that for all $z \in \mathcal{Z}$, $\exists\theta$ s.t. $z = f(\theta)$ and that $\mathcal{Z}$ is related to the parameterization used. For example, when using a linear parameterization, for a feature matrix $X \in \mathbb{R}^{n\times d}$, the set $\mathcal{Z} = \{z \in \mathbb{R}^n | \exists\theta \in \Theta \text{ s.t. } z = X^\top\theta\}$. Clearly, for linear parameterization, the set $\mathcal{Z}$ is convex, and the Euclidean projection onto $\mathcal{Z}$ is unique. In general, set $\mathcal{Z} = \{z \in \mathbb{R}^n | \exists\theta \in \Theta \text{ s.t. } z = f(\theta)\}$ i.e. the set $\mathcal{Z}$ consists of realizations when using a class of parametric functions (parameterized by θ). The update in Eq. (3) can be equivalently written as:

$$\bar{z}_{t+1} = \arg\min_{z\in\mathcal{Z}} \left[\langle \nabla_z \ell_t(z_t), z - z_t\rangle + \frac{1}{2\eta_t}\|z - z_t\|_2^2\right] = f\left(\arg\min_{\theta\in\Theta} \tilde{g}_t^z(\theta)\right). \tag{4}$$

The first equality is true in general (see [2]), while the second equality uses the definition of $\mathcal{Z}$. Hence minimizing $\tilde{g}_t^z$ is equivalent to projected stochastic gradient descent in the target space. This equivalence implies that the inexact minimization of $\tilde{g}_t^z$ (for example, using m steps of GD in Algorithm 1) can be interpreted as an inexact projection onto $\mathcal{Z}$. Note that since we are approximately minimizing $\tilde{g}_t^z(\theta)$, the resulting iterate $z_{t+1} = f(\theta_{t+1}) \in \mathcal{Z}$.

For non-convex f, the set $\mathcal{Z}$ can be non-convex and the projection is not well-defined. However, algorithmically, the surrogate $\tilde{g}_t^z$ can still be minimized, albeit without any guarantees on the convergence. In order to fully instantiate the stochastic surrogate, we need to specify the step-size sequence $\{\eta_t\}_{t=1}^T$. Since the resulting algorithm is equivalent to projected SGD on a (strongly-) convex function $\ell(z)$, we can use the existing literature to set

η_t. For example, if $\ell(z)$ is convex, the step-size $\eta_t = O\left(1/\sqrt{t}\right)$ in order to ensure convergence to the minimizer [16]. In the next section, we will prove guarantees for convergence of our algorithm to the minimizer when $\ell(z)$ is strongly-convex, and f is a linear map and show that we match the convergence rate for parametric SGD on smooth, strongly-convex problems.

A similar approach has been explored in [8] for which the updates in the simplest setting are given as: $z_{t+1} = \arg\min_{z\in\mathbb{R}^n}\left[\langle\nabla_z\ell(z_t), z - z_t\rangle + \frac{1}{2\eta}\left\|z - z_t\right\|_2^2\right]$ and $\theta_{t+1} = \arg\min_{\theta\in\Theta}\left[\frac{1}{2}\left\|f(\theta) - z_{t+1}\right\|_2^2 + R(\theta)\right]$. While our framework uses SGD in the target space and GD to minimize $\tilde{g}_t^z$ in the parametric space, Johnson and Zhang [8] uses GD in the target space and SGD in the parametric space. Unlike Johnson and Zhang [8], our framework allows for efficient optimization in the parametric space (e.g., by using Armijo line-search).

4. Theoretical Results

4.1. Convergence Analysis

We will only focus on GD in the target space where each ℓ_i is strongly convex and ϵ_{t+1} represents the projection error i.e. $\|z_{t+1} - \bar{z}_{t+1}\| \leq \epsilon_{t+1}$ where z_{t+1} is the inexact projection onto $\mathcal{Z}$. For the subsequent theoretical results, we assume that the choice of f ensures that that the projection is well-defined (for example, when f is linear in θ).

Proposition 1 *Assuming that (i) ℓ_t is L-smooth and μ-strongly convex, (ii) the projection error at iteration t is ϵ_t, the sub-optimality for the updates in Eq. (3) can be bounded as,*

$$\mathbb{E}\left\|z_T - z^*\right\|_2^2 \leq \left(\Pi_{i=1}^T\rho_i\right)\mathbb{E}\left\|z_0 - z^*\right\|_2^2 + \frac{3}{2}\sum_{t=1}^T\Pi_{i=t+1}^T\rho_i\epsilon_t^2 + 2\sigma^2\sum_{i=1}^T\Pi_{i=t+1}^T\rho_i\eta_t^2\,,$$

where $\sigma^2 := \mathbb{E}\left[\left\|\nabla\ell(z^*) - \nabla\ell_t(z^*)\right\|_2^2\right]$ and $\rho_t = (1 - 2\mu\eta_t(1 - \mu\eta_t))$.

In Theorem 1 (proof in Appendix D.1), we prove an bound on the suboptimality for z_{t+1}. Theorem 1 shows that the upper bound depends on the noise σ^2 and the accumulation of projection error ϵ_t^2. If we can control the projection errors such that $\sum_{t=1}^T\Pi_{i=t+1}^T\rho_i\epsilon_t^2 \in \mathcal{O}(\rho^T)$, then we can recover the GD rate for strongly convex functions. In the following corollary we fix the step size and assume that the projection error is bounded i.e. $\epsilon_t^2 \leq \epsilon^2$.

Corollary 2 *Under the same assumptions as Proposition 1 and (iii) $\epsilon_t \leq \epsilon\ \forall t$, if $\eta_t = \eta = \frac{1}{L}$, then,*

$$\mathbb{E}\left\|z_T - z^*\right\|_2^2 \leq \left(1 - \frac{\mu}{2L}\right)^T\left\|z_0 - z^*\right\|_2^2 + 3\kappa\,\epsilon^2 + \frac{\sigma^2}{\mu L}.$$

The iterate converges to the neighbourhood of the minimizer in the target space. However, one can set η_t using other techniques such as exponential step-sizes [24], we can achieve convergence to the exact minimizer at an $O\left(\exp\left(-T\right) + {}^{\sigma^2}/_{T}\right)$ rate.

4.2. Controlling the Projection Error

Since iteration t of projected SGD is equivalent to minimizing $g_t(\theta)$, we can control the projection error ϵ_{t+1} by minimizing $g_t(\theta)$ to the desired tolerance. Specifically, we assume that the function f is L_f-Lipschitz in θ. In the case where f is linear, $L_f = \|X\|$. Recall that $z_{t+1} = f(\theta_{t+1})$ and $\bar{z}_{t+1} = f(\bar{\theta}_{t+1})$ where $\bar{\theta}_{t+1} = \arg\min_\theta g_t(\theta)$ and θ_{t+1} is obtained by approximately minimizing $g_t(\theta)$. Using the Lipschitz continuity of f, $\epsilon_{t+1}^2 = \|\bar{z}_{t+1} - z_{t+1}\|_2^2 \leq L_f^2 \left\|\bar{\theta}_{t+1} - \theta_{t+1}\right\|_2^2$. The following proposition (proof in Appendix D.2) demonstrates that in order to control the projection error i.e. $\epsilon_t \leq \epsilon$, we need m_t (the number of iterations required to minimize the surrogate at iteration t) to be $O(\log(1/\epsilon))$.

Proposition 3 *Assuming that (i) each ℓ_i is μ-strongly convex, (ii) f is L_θ-smooth, (iii) g is μ_g strongly convex and L_g-smooth with $\kappa_g = L_g/\mu_g$. In order to ensure that $\epsilon_t \leq \epsilon$ for a given $\epsilon > 0$ we require*

$$m_t \geq \kappa_g \log\left(\frac{L_f^2 L\, L_\theta \left[\exp\left(-t/2\kappa\right) L_f^2 \left\|\theta_0 - \theta^*\right\|_2^2 + 3\kappa\, \epsilon^2 + \frac{\sigma^2}{\mu L} + \frac{\sigma_\theta^2}{L\, L_\theta}\right]}{\mu_g^2\, \epsilon^2}\right).$$

Hence, we only need $O\left(\log(1/\epsilon)\right)^2$ gradient calls to converge to a $O(\epsilon+\sigma)$-neighbourhood of the minimizer. With the same number of gradient calls and the exponential step-size [24], we can get to an $O(\epsilon)$ neighbourhood of the minimizer. Note that in the above bound when t increases the lower-bound of m_t decreases. Note that we have two different notions of noise: σ which is the noise in the stochastic gradients of ℓ w.r.t. z and σ_θ which is noise in the stochastic gradients of h w.r.t. θ.

5. Experimental Evaluation

In this section, we experimentally evaluate our framework on LibSVM benchmarks [3], a standard suite of convex-optimization problems. We consider three datasets – `mushrooms`, `ijcnn` and `rcv1`, two losses – squared loss and logistic loss and three mini-batch sizes – {25, 125, 625, full-batch}. We only consider the linear parametrization setting where $f = X\theta$. Each optimization algorithm is run for 500 epochs (full passes over the data) and 3 independent runs. We compare target optimization referred to as stochastic surrogate optimization or `SSO` in the plots against SGD with the standard theoretical $1/2L_\theta$ step-size (referred to as `SGD` in the plots) and SGD with the step-size set according to a stochastic line-search [22] with the default hyper-parameters (referred to as `SLS`). Since `SSO` is equivalent to projected SGD in the target space, we set η (in the surrogate definition) to $1/2L$ where L is the smoothness of ℓ w.r.t z. For `SSO`, since the surrogate optimization is a deterministic problem, we use the standard back-tracking Armijo line-search [1].

Fig. 1 for the squared loss (the corresponding plot for the logistic loss is included in Appendix E) shows that (i) `SSO` improves over `SGD` when the step-sizes are chosen theoretically, (ii) `SSO` is competitive with `SLS` and (iii) as m increases, on average, the

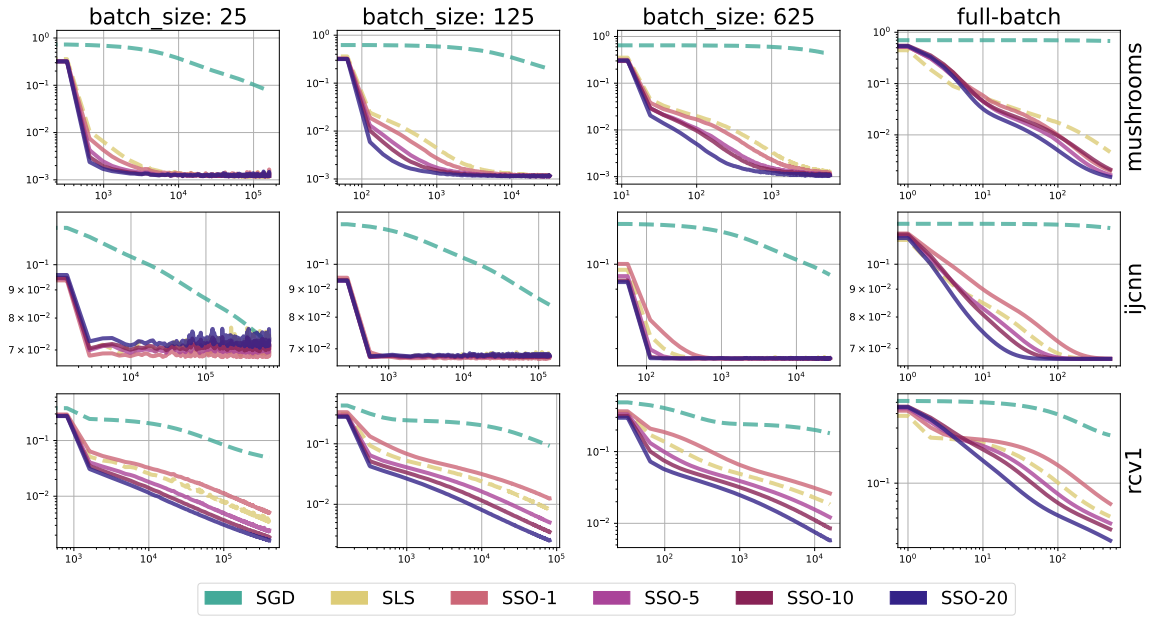

Figure 1: Comparison in terms of average loss of stochastic gradient decent, stochastic line-search, and finally stochastic surrogate optimization using the mean squared error loss. Each run was evaluated over three random seeds following the same initialization scheme. All plots are in log-log space to make trends between optimization algorithms more apparent. We note the SGD fails to perform on par with a more sophisticated algorithm like SLS when using its near optimal step-size proposed by theory. On the other-hand, using its own proposed theoretical step-size, stochastic surrogate optimization remains on par or better than SLS for all m. We note that in the full-batch setting, SGD reduces to GD, while SLS reduces to GD with a back-tracking line-search.

performance of `SSO` improves (since the projection error decreases). Additionally, we show in Appendix E that taking $m \geq 1$ steps on the surrogate requires negligible additional compute. This indicates that offline updates (that do not require additional access to the stochastic gradient oracle) can improve the empirical performance.

6. Discussion

In the future, we aim to extend our theoretical results to a broader class of functions, and empirically evaluate target optimization for more complex models such as deep neural networks. Moreover, since our framework allows us the use of any optimizer in the target space, we plan to explore other optimization algorithms in order to construct better surrogates.

References

[1] Larry Armijo. Minimization of functions having lipschitz continuous first partial derivatives. *Pacific Journal of mathematics*, 16(1):1–3, 1966.

[2] Sébastien Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.

[3] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[4] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems, NeurIPS*, 2014.

[5] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[6] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 2011.

[7] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems, NeurIPS*, 2013.

[8] Rie Johnson and Tong Zhang. Guided learning of nonconvex models through successive functional gradient optimization. In *International Conference on Machine Learning*, pages 4921–4930. PMLR, 2020.

[9] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[10] Jonathan Wilder Lavington, Sharan Vaswani, and Mark Schmidt. Improved policy optimization for online imitation learning. *arXiv preprint arXiv:2208.00088*, 2022.

[11] Julien Mairal. Stochastic majorization-minimization algorithms for large-scale optimization. *Advances in Neural Information Processing Systems*, 26, 2013.

[12] Julien Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. *SIAM Journal on Optimization*, 25(2):829–855, 2015.

[13] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.

[14] Lam M Nguyen, Trang H Tran, and Marten van Dijk. Finite-sum optimization: A new perspective for convergence to a global solution. *arXiv preprint arXiv:2202.03524*, 2022.

[15] Francesco Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

[16] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[17] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. *Journal of machine learning research*, pages 627–635, 2011.

[18] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112, 2017.

[19] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*, pages 1889–1897, 2015.

[20] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[21] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1057–1063, 2000.

[22] Sharan Vaswani, Aaron Mishkin, Issam Laradji, Mark Schmidt, Gauthier Gidel, and Simon Lacoste-Julien. Painless stochastic gradient: Interpolation, line-search, and convergence rates. In *Advances in Neural Information Processing Systems*, pages 3727–3740, 2019.

[23] Sharan Vaswani, Olivier Bachem, Simone Totaro, Robert Mueller, Matthieu Geist, Marlos C Machado, Pablo Samuel Castro, and Nicolas Le Roux. A functional mirror ascent view of policy gradient methods with function approximation. *arXiv preprint arXiv:2108.05828*, 2021.

[24] Sharan Vaswani, Benjamin Dubois-Taine, and Reza Babanezhad. Towards noise-adaptive, problem-adaptive (accelerated) stochastic gradient descent. In *International Conference on Machine Learning*, pages 22015–22059. PMLR, 2022.

[25] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Supplementary material

Organization of the Appendix

A Definitions

B Algorithm

C Proofs in the deterministic setting

D Proofs in the stochastic setting

E Additional Experimental Results

Appendix A. Definitions

Our main assumptions are that each individual function f_i is differentiable, has a finite minimum f_i^*, and is L_i-smooth, meaning that for all v and w,

$$f_i(v) \leq f_i(w) + \langle \nabla f_i(w),\, v - w \rangle + \frac{L_i}{2} \|v - w\|_2^2 \,, \qquad \text{(Individual Smoothness)}$$

which also implies that f is L-smooth, where L is the maximum smoothness constant of the individual functions. A consequence of smoothness is the following bound on the norm of the stochastic gradients,

$$\|\nabla f_i(w)\|^2 \leq 2L(f_i(w) - f_i^*). \tag{5}$$

We also assume that each f_i is convex, meaning that for all v and w,

$$f_i(v) \geq f_i(w) + \langle \nabla f_i(w),\, v - w \rangle, \qquad \text{(Convexity)}$$

Depending on the setting, we will also assume that f is μ strongly-convex, meaning that for all v and w,

$$f(v) \geq f(w) + \langle \nabla f(w),\, v - w \rangle + \frac{\mu}{2} \|v - w\|_2^2 \,, \qquad \text{(Strong Convexity)}$$

Appendix B. Algorithm

Algorithm 1 Surrogate optimization

Input: θ_0 (initialization), T (Number of iterations), m_t (number of inner-loops), η (step-size to form the surrogate), α (step-size for the parametric update)

for $t \leftarrow 0$ **to** $T-1$ **do**
  Access the gradient oracle and define $g_t^z(\theta)$ Eq. (1)
  Initialize inner-loop: $\omega_0 = \theta_t$
  for $k \leftarrow 0$ **to** m_t **do**
    $\omega_{k+1} = \omega_k + \alpha \nabla_\omega g_t^z(\omega_k)$
  end
  $\theta_{t+1} = \omega_m$
  $z_{t+1} = f(\theta_{t+1})$
end
Return θ_T

Appendix C. Proofs in the deterministic setting

Lemma 1 *Assuming that $g_t^z(\theta)$ is β-smooth w.r.t. the Euclidean norm and $\eta \leq \frac{1}{L}$, then, for $\alpha = 1/\beta$, iteration t of Algorithm 1 guarantees that $h(\theta_{t+1}) \geq h(\theta_t)$ for any number $m \geq 1$ of surrogate steps. In this setting, under the additional assumption that h is lower-bounded by h^*, then Algorithm 1 results in the following guarantee,*

$$\min_{t \in \{0,\ldots,T-1\}} \|\nabla h(\theta_t)\|_2^2 \leq \frac{2\beta\left[h(\theta_0) - h^*\right]}{T}.$$

Proof Using the update in Algorithm 1 with $\alpha = \frac{1}{\beta}$ and the β-smoothness of $g_t^z(\theta)$, for all $k \in [m-1]$,

$$g_t^z(\omega_{k+1}) \leq g_t^z(\omega_k) - \frac{1}{2\beta}\|\nabla g_t^z(\omega_k)\|_2^2$$

After m steps,

$$g_t^z(\omega_m) \leq g_t^z(\omega_0) - \frac{1}{2\beta}\sum_{k=0}^{m-1}\|\nabla g_t^z(\omega_k)\|_2^2$$

Since $\theta_{t+1} = \omega_m$ and $\omega_0 = \theta_t$ in Algorithm 1,

$$\implies g_t^z(\theta_{t+1}) \leq g_t^z(\theta_t) - \frac{1}{2\beta}\|\nabla g_t^z(\theta_t)\|_2^2 - \sum_{k=1}^{m-1}\|\nabla g_t^z(\omega_k)\|_2^2$$

Note that $h(\theta_t) = g_t^z(\theta_t)$ and if $\eta \leq \frac{1}{L}$, then $h(\theta_{t+1}) \leq g_t^z(\theta_{t+1})$. Using these relations,

$$h(\theta_{t+1}) \leq h(\theta_t) - \left[\underbrace{\frac{1}{2\beta}\left\|\nabla g_t^z(\theta_t)\right\|_2^2 + \sum_{k=1}^{m-1}\left\|\nabla g_t^z(\omega_k)\right\|_2^2}_{+\text{ve}}\right] \implies h(\theta_{t+1}) \leq h(\theta_t).$$

This proves the first part of the Lemma. Since $\sum_{k=1}^{m-1}\left\|\nabla g_t^z(\omega_k)\right\|_2^2 \geq 0$,

$$\begin{aligned} & h(\theta_{t+1}) \leq h(\theta_t) - \frac{1}{2\beta}\left\|\nabla g_t^z(\theta_t)\right\|_2^2 \\ \implies & \left\|\nabla h(\theta_t)\right\|_2^2 \leq 2\beta\left[h(\theta_t) - h(\theta_{t+1})\right] \qquad \text{(Since } \nabla h(\theta_t) = \nabla g_t^z(\theta_t)\text{)} \end{aligned}$$

Summing from $k = 0$ to $T - 1$, and dividing by T,

$$\begin{aligned} \frac{\left\|\nabla h(\theta_t)\right\|_2^2}{T} &\leq \frac{2\beta\left[h(\theta_t) - h^*\right]}{T} \\ \implies \min_{t\in\{0,\ldots,T-1\}}\left\|\nabla h(\theta_t)\right\|_2^2 &\leq \frac{2\beta\left[h(\theta_t) - h^*\right]}{T} \end{aligned}$$

■

Appendix D. Proofs in the stochastic setting

D.1. Proof for strongly-convex functions

We consider the case where $\ell(f)$ is strongly-convex and the set $\mathcal{F}$ is convex. We will focus on GD in the functional space, and consider the following updates:

$$\begin{aligned} z_{t+1/2} &= z_t - \eta_t \nabla \ell_t(z_t) \\ \bar{z}_{t+1} &= \arg\min_{f\in\mathcal{F}} \frac{1}{2}\left\|f - z_{t+1/2}\right\|_2^2 = f\left(\arg\min_{\theta} g_t(\theta)\right) \\ \left\|z_{t+1} - \bar{z}_{t+1}\right\| &\leq \epsilon_{t+1} \end{aligned}$$

Lemma 2 *For any vectors a, b, c, we can bound the term $\langle b - a, a - c\rangle$ as follows:*

$$\langle b - a, a - c\rangle \leq \frac{1}{2}\left\|b - c\right\|_2^2 - \frac{1}{2}\left\|a - c\right\|_2^2 \tag{6}$$

Proof

$$\begin{aligned} \langle b - a, a - c\rangle &= \langle b - c + c - a, a - c\rangle = \langle b - c, a - c\rangle + \langle c - a, a - c\rangle \\ &= \langle b - c, a - c\rangle - \left\|a - c\right\|_2^2 \end{aligned}$$

$$\leq \frac{\|b-c\|_2^2}{2} + \frac{\|a-c\|_2^2}{2} - \|a-c\|_2^2 \qquad \text{(Using Young inequality)}$$
$$= \frac{\|b-c\|_2^2}{2} - \frac{\|a-c\|_2^2}{2}$$

■

In the following lemma we bound the suboptimality of z_{t+1} based on $z_{t+1/2}$ and the projection error.

Lemma 3 *One can bound the approximate sub-optimality based on exact sub-optimality as follows:*

$$\|z_{t+1} - z^*\|_2^2 \leq \frac{3}{2}\epsilon_{t+1}^2 + \|\bar{z}_{t+1} - z^*\|_2^2 \leq \frac{3}{2}\epsilon_{t+1}^2 + \left\|z_{t+1/2} - z^*\right\|_2^2 \tag{7}$$

Proof

$$\begin{aligned}
\|z_{t+1} - z^*\|_2^2 &= \|z_{t+1} - \bar{z}_{t+1} + \bar{z}_{t+1} - z^*\|_2^2 \\
&= \|z_{t+1} - \bar{z}_{t+1}\|_2^2 + \|\bar{z}_{t+1} - z^*\|_2^2 - 2\langle z_{t+1} - \bar{z}_{t+1}, \bar{z}_{t+1} - z^*\rangle \\
&= \|z_{t+1} - \bar{z}_{t+1}\|_2^2 + \|\bar{z}_{t+1} - z^*\|_2^2 - 2\langle z_{t+1} - \bar{z}_{t+1}, \bar{z}_{t+1} - z_{t+1} + z_{t+1} - z^*\rangle \\
&= \|z_{t+1} - \bar{z}_{t+1}\|_2^2 + \|\bar{z}_{t+1} - z^*\|_2^2 - 2\langle z_{t+1} - \bar{z}_{t+1}, \bar{z}_{t+1} - z_{t+1}\rangle \\
&- 2\langle z_{t+1} - \bar{z}_{t+1}, z_{t+1} - z^*\rangle \\
&\leq 3\,\|z_{t+1} - \bar{z}_{t+1}\|_2^2 + \|\bar{z}_{t+1} - z^*\|_2^2 - 2\langle z_{t+1} - \bar{z}_{t+1}, z_{t+1} - z^*\rangle \\
&\leq 3\epsilon_{t+1}^2 + \|\bar{z}_{t+1} - z^*\|_2^2 - 2\langle z_{t+1} - \bar{z}_{t+1}, z_{t+1} - z^*\rangle \\
&\qquad \text{(Projection error upper-bound)} \\
&\leq 3\epsilon_{t+1}^2 + \|\bar{z}_{t+1} - z^*\|_2^2 + \|\bar{z}_{t+1} - z^*\|_2^2 - \|z_{t+1} - z^*\|_2^2 \\
&\qquad \text{(Using Lemma 2 } (a = z_{t+1}, b = \bar{z}_{t+1}, c = z^*)) \\
&\leq 3\epsilon_{t+1}^2 + 2\left\|z_{t+1/2} - z^*\right\|_2^2 - \|z_{t+1} - z^*\|_2^2 \\
&\qquad \text{(Since projections are non-expansive)}
\end{aligned}$$

$$\implies \|z_{t+1} - z^*\|_2^2 \leq \frac{3}{2}\epsilon_{t+1}^2 + \|\bar{z}_{t+1} - z^*\|_2^2 \leq \frac{3}{2}\epsilon_{t+1}^2 + \left\|z_{t+1/2} - z^*\right\|_2^2$$

■

Now we need to bound the exact sub-optimality at iteration $t+1$ by the inexact sub-optimality at iteration t to get a recursion.

Lemma 4 *One can bound the exact sub-optimality at iteration $t+1$ by the inexact sub-optimality at iteration t as follows:*

$$\|\bar{z}_{t+1} - z^*\|_2^2 \leq \left(1 - \frac{2\eta_t\,\mu L}{\mu + L}\right)\|z_t - z^*\|_2^2 + \left(2{\eta_t}^2 - \frac{2\eta_t}{\mu + L}\right)\|\nabla\ell_t(z^*) - \nabla\ell_t(z_t)\|_2^2 \tag{8}$$
$$+ 2{\eta_t}^2\,\|\nabla\ell_t(z^*) - \nabla\ell(z^*)\|_2^2 + A_t \tag{9}$$

where $A_t = 2\eta_t\langle z_t - z^*, \nabla\ell(z^*) - \nabla\ell_t(z^*)\rangle$.

Proof Since $z^* \in \mathcal{F}$ and optimal, $z^* = \Pi_{\mathcal{F}}[z^* - \eta_t \nabla \ell(z^*)]$.

$$\begin{aligned}
\|\bar{z}_{t+1} - z^*\|_2^2 &= \|\Pi_{\mathcal{F}}[z_t - \eta_t \nabla \ell_t(z_t)] - \Pi_{\mathcal{F}}[z^* - \eta_t \nabla \ell(z^*)]\|_2^2 \\
&\leq \|[z_t - \eta_t \nabla \ell_t(z_t)] - [z^* - \eta_t \nabla \ell(z^*)]\|_2^2 \\
&\qquad\qquad\qquad\qquad\qquad\qquad \text{(Since projections are non-expansive)} \\
&= \|[z_t - \eta_t \nabla \ell_t(z_t)] - [z^* - \eta_t \nabla \ell_t(z^*)] + \eta_t \left[\nabla \ell(z^*) - \nabla \ell_t(z^*)\right]\|_2^2 \\
&= \|[z_t - \eta_t \nabla \ell_t(z_t)] - [z^* - \eta_t \nabla \ell_t(z^*)]\|_2^2 + {\eta_t}^2 \|\nabla \ell(z^*) - \nabla \ell_t(z^*)\|_2^2 \\
&+ \underbrace{2\eta_t \langle z_t - z^*, \nabla \ell(z^*) - \nabla \ell_t(z^*)\rangle}_{A_t} + 2{\eta_t}^2 \langle \nabla \ell_t(z^*) - \nabla \ell_t(z_t), \nabla \ell(z^*) - \nabla \ell_t(z^*)\rangle \\
&\leq \|[z_t - \eta_t \nabla \ell_t(z_t)] - [z^* - \eta_t \nabla \ell_t(z^*)]\|_2^2 + {\eta_t}^2 \|\nabla \ell(z^*) - \nabla \ell_t(z^*)\|_2^2 + A_t \\
&+ {\eta_t}^2 \|\nabla \ell_t(z^*) - \nabla \ell_t(z_t)\|_2^2 + {\eta_t}^2 \|\nabla \ell(z^*) - \nabla \ell_t(z^*)\|_2^2 \qquad \text{(Young inequality)} \\
&= \|[z_t - \eta_t \nabla \ell_t(z_t)] - [z^* - \eta_t \nabla \ell_t(z^*)]\|_2^2 + 2{\eta_t}^2 \|\nabla \ell(z^*) - \nabla \ell_t(z^*)\|_2^2 + A_t \\
&+ {\eta_t}^2 \|\nabla \ell_t(z^*) - \nabla \ell_t(z_t)\|_2^2
\end{aligned}$$

Let us bound the first term in the above equation,

$$\begin{aligned}
\|z_t - \eta_t \nabla \ell_t(z_t)] - [z^* - \eta_t \nabla \ell_t(z^*)]\|_2^2 &= \|z_t - z^* - \eta_t \left[\nabla \ell_t(z_t) - \nabla \ell_t(z^*)\right]\|_2^2 \\
&\leq \|z_t - z^*\|_2^2 - 2\eta_t \langle z_t - z^*, \nabla \ell_t(z_t) - \nabla \ell_t(z^*)\rangle \\
&+ {\eta_t}^2 \|\nabla \ell_t(z^*) - \nabla \ell_t(z_t)\|_2^2
\end{aligned}$$

At this point, let us assume ℓ_t is μ strongly-convex to make the proof easier. Since ℓ_t is μ-SC and L-smooth,

$$\langle z_t - z^*, \nabla \ell_t(z_t) - \nabla \ell_t(z^*)\rangle \geq \frac{\mu L}{\mu + L} \|z_t - z^*\|_2^2 + \frac{1}{\mu + L} \|\nabla \ell_t(z^*) - \nabla \ell_t(z_t)\|_2^2$$

Using the above inequality,

$$\begin{aligned}
\|z_t - \eta_t \nabla \ell_t(z_t)] - [z^* - \eta_t \nabla \ell_t(z^*)]\|_2^2 \leq &\left(1 - \frac{2\eta_t\, \mu L}{\mu + L}\right) \|z_t - z^*\|_2^2 \\
&+ \left({\eta_t}^2 - \frac{2\eta_t}{\mu + L}\right) \|\nabla \ell_t(z^*) - \nabla \ell_t(z_t)\|_2^2
\end{aligned}$$

Using the above bounds we have

$$\begin{aligned}
\|\bar{z}_{t+1} - z^*\|_2^2 \leq &\left(1 - \frac{2\eta_t\, \mu L}{\mu + L}\right) \|z_t - z^*\|_2^2 + \left(2{\eta_t}^2 - \frac{2\eta_t}{\mu + L}\right) \|\nabla \ell_t(z^*) - \nabla \ell_t(z_t)\|_2^2 \\
&+ A_t + 2{\eta_t}^2 \|\nabla \ell(z^*) - \nabla \ell_t(z^*)\|_2^2
\end{aligned}$$

■

Proposition 4 *Assuming that (i) ℓ_t is L-smooth and μ-strongly convex, (ii) the projection error at iteration t is ϵ_t, the sub-optimality for the updates in Eq. (3) can be bounded as,*

$$\mathbb{E}\left\|z_T - z^*\right\|_2^2 \leq \left(\Pi_{i=1}^T \rho_i\right) \mathbb{E}\left\|z_0 - z^*\right\|_2^2 + \frac{3}{2}\sum_{t=1}^T \Pi_{i=t+1}^T \rho_i \epsilon_t^2 + 2\sigma^2 \sum_{i=1}^T \Pi_{i=t+1}^T \rho_i \eta_t^2\,,$$

where $\sigma^2 := \mathbb{E}\left[\left\|\nabla \ell(z^*) - \nabla \ell_t(z^*)\right\|_2^2\right]$ and $\rho_t = (1 - 2\mu\eta_t(1-\mu\eta_t))$.

Proof Using Lemma 3 we get

$$\left\|z_{t+1} - z^*\right\|_2^2 \leq \frac{3}{2}\epsilon_{t+1}^2 + \left\|\bar{z}_{t+1} - z^*\right\|_2^2$$

Taking expectation,

$$\mathbb{E}\left[\left\|z_{t+1} - z^*\right\|_2^2\right] \leq \frac{3}{2}\mathbb{E}\left[\epsilon_{t+1}^2\right] + \mathbb{E}\left[\left\|\bar{z}_{t+1} - z^*\right\|_2^2\right]$$

Using Lemma 4 we get a bound for $\|\bar{z}_{t+1} - z^*\|$.

$$\begin{aligned}\left\|\bar{z}_{t+1} - z^*\right\|_2^2 \leq & \underbrace{\left(1 - \frac{2\eta_t\,\mu L}{\mu + L}\right)\left\|z_t - z^*\right\|_2^2 + \left(2{\eta_t}^2 - \frac{2\eta_t}{\mu + L}\right)\left\|\nabla \ell_t(z^*) - \nabla \ell_t(z_t)\right\|_2^2}_{B_t} \\ & + 2{\eta_t}^2\left\|\nabla \ell_t(z^*) - \nabla \ell(z^*)\right\|_2^2 + A_t\end{aligned}$$

Now we want $\left(2{\eta_t}^2 - \frac{2\eta_t}{\mu+L}\right) \leq 0$. This is true when $\eta_t \leq \frac{1}{\mu+L}$, and hence true when $\eta_t \leq \frac{1}{2L}$. Hence, we can use strong-convexity to bound the last term

$$\begin{aligned}B_t &\leq \left(1 - \frac{2\eta_t\,\mu L}{\mu + L}\right)\left\|z_t - z^*\right\|_2^2 + \left(2\mu^2{\eta_t}^2 - \frac{2\eta_t\,\mu^2}{\mu + L}\right)\left\|z^* - z_t\right\|_2^2 \\ &= \left\|z_t - z^*\right\|_2^2\left[1 + 2\mu^2{\eta_t}^2 - 2\mu\eta_t(\frac{L}{\mu + L} + \frac{\mu}{\mu + L})\right] \\ &= \left\|z_t - z^*\right\|_2^2 \underbrace{(1 - 2\mu\eta_t(1 - \mu\eta_t))}_{\rho_t}\end{aligned}$$

Define $\sigma^2 := \mathbb{E}\left[\left\|\nabla \ell(z^*) - \nabla \ell_t(z^*)\right\|_2^2\right]$. Putting together the above relations and taking expectation we have,

$$\mathbb{E}\left\|z_{t+1} - z^*\right\|_2^2 \leq \rho_t \mathbb{E}\left\|z_t - z^*\right\|_2^2 + \frac{3}{2}\mathbb{E}[\epsilon_{t+1}^2] + 2{\eta_t}^2\sigma^2 + \mathbb{E}[A_t]$$

$$= \rho_t \mathbb{E}\left\|z_t - z^*\right\|_2^2 + \frac{3}{2}\mathbb{E}[\epsilon_{t+1}^2] + 2{\eta_t}^2\sigma^2 \qquad (\mathbb{E}[A_t] = 0)$$

Unrolling the recursion we get,

$$\mathbb{E}\left\|z_T - z^*\right\|_2^2 \leq \left(\prod_{t=0}^{T-1}\rho_t\right)\mathbb{E}\left\|z_0 - z^*\right\|_2^2 + \frac{3}{2}\sum_{t=0}^{T-1}\epsilon_t^2\prod_{i=t+1}^{T-1}\rho_i + 2\sigma^2\sum_{t=0}^{T-1}\eta_t^2\prod_{i=t+1}^{T-1}\rho_i \tag{10}$$

■

Corollary 5 *Under the same assumptions as Proposition 1 and (iii) $\epsilon_t \leq \epsilon\ \forall t$, if $\eta_t = \eta = \frac{1}{L}$, then,*

$$\mathbb{E}\left\|z_T - z^*\right\|_2^2 \leq \left(1 - \frac{\mu}{2L}\right)^T\left\|z_0 - z^*\right\|_2^2 + 3\kappa\,\epsilon^2 + \frac{\sigma^2}{\mu L}.$$

Proof If $\eta_t = \eta$, then for $\rho := 1 - 2\mu\eta(1 - \mu\eta)$ and using the recursion in Proposition 1, one gets

$$\mathbb{E}\left\|z_T - z^*\right\|_2^2 \leq \left(\prod_{t=0}^{T-1}\rho\right)\mathbb{E}\left\|z_0 - z^*\right\|_2^2 + \frac{3}{2}\sum_{t=0}^{T-1}\epsilon_t^2\prod_{i=t+1}^{T-1}\rho^i + 2\sigma^2\sum_{t=0}^{T-1}\eta_t^2\prod_{i=t+1}^{T-1}\rho^i \tag{11}$$

$$= \rho^T\left\|z_0 - z^*\right\|_2^2 + \frac{3}{2}\sum_{t=0}^{T-1}\epsilon_t^2\,\rho^{T-t-1} + 2\sigma^2\sum_{t=0}^{T-1}\eta^2\rho^{T-t-1} \tag{12}$$

$$= \rho^T\left\|z_0 - z^*\right\|_2^2 + \frac{3}{2}\sum_{t=0}^{T-1}\epsilon_{T-t-1}^2\,\rho^t + 2\sigma^2\eta^2\sum_{t=0}^{T-1}\rho^t \tag{13}$$

If we control the projection error such that $\epsilon_t = \epsilon$ for all t, then,

$$\mathbb{E}\left\|z_T - z^*\right\|_2^2 \leq \rho^T\left\|z_0 - z^*\right\|_2^2 + \frac{3\epsilon^2}{2}\frac{1}{1-\rho} + \frac{2\sigma^2\eta^2}{1-\rho} \tag{14}$$

Setting $\eta = \frac{1}{2L}$,

$$\rho = 1 - \frac{\mu}{L}\left(1 - \frac{\mu}{2L}\right) = 1 - \frac{\mu}{L} + \frac{\mu^2}{2L^2} \leq 1 - \frac{\mu}{L} + \frac{\mu}{2L} = 1 - \frac{\mu}{2L} \tag{15}$$

With this simplification, we get our final bound,

$$\mathbb{E}\left\|z_T - z^*\right\|_2^2 \leq \left(1 - \frac{\mu}{2L}\right)^T\left\|z_0 - z^*\right\|_2^2 + 3\kappa\,\epsilon^2 + \frac{\sigma^2}{\mu L} \tag{16}$$

■

D.2. Controlling the projection error

Proposition 6 *Assuming that (i) each ℓ_i is μ-strongly convex, (ii) f is L_θ-smooth, (iii) g is μ_g strongly convex and L_g-smooth with $\kappa_g = L_g/\mu_g$. In order to ensure that $\epsilon_t \leq \epsilon$ for a given $\epsilon > 0$ we require*

$$m_t \geq \kappa_g \log\left(\frac{L_f^2 L\, L_\theta \left[\exp\left(-t/2\kappa\right) L_f^2 \left\|\theta_0 - \theta^*\right\|_2^2 + 3\kappa\, \epsilon^2 + \frac{\sigma^2}{\mu L} + \frac{\sigma_\theta^2}{L\, L_\theta}\right]}{\mu_g^2\, \epsilon^2}\right).$$

Proof If we minimize $g_t(\theta)$ using m_t iterations of GD starting from θ_t, then using the convergence guarantees of gradient descent [13],

$$\left\|\bar{\theta}_{t+1} - \theta_{t+1}\right\|_2^2 \leq \exp\left(-m_t/\kappa_\theta\right) \left\|\bar{\theta}_{t+1} - \theta_t\right\|_2^2$$

where, $\kappa_g = \frac{L_\theta}{\mu_g}$ is the condition number of $g_t(\theta)$. Putting together the above relations,

$$\begin{aligned}
\epsilon_{t+1}^2 &\leq L_f^2 \exp\left(-m_t/\kappa_g\right) \left\|\bar{\theta}_{t+1} - \theta_t\right\|_2^2 \\
&\leq \frac{L_f^2}{\mu_\theta^2} \exp\left(-m_t/\kappa_g\right) \left\|\nabla g_t(\theta_t) - \nabla g_t(\bar{\theta}_{t+1})\right\|_2^2 && (\mu_g\text{-strong convexity of } g_t) \\
&= \underbrace{\frac{L_f^2}{\mu_g^2} \exp\left(-m_t/\kappa_g\right)}_{:=C_t} \left\|\nabla h_t(\theta_t)\right\|_2^2 && (\text{Since } \nabla g_t(\bar{\theta}_{t+1}) = 0 \text{ and } \nabla g_t(\theta_t) = \nabla h_t(\theta_t))
\end{aligned}$$

Since g_t depends on ℓ_t, it is a random quantity. Taking expectation w.r.t the randomness at iteration t,

$$\mathbb{E}[\epsilon_{t+1}^2] \leq C_t \mathbb{E} \left\|\nabla h_t(\theta_t)\right\|_2^2$$

Using the growth condition on the stochastic gradients of h_t, $\mathbb{E}\left\|\nabla h_t(\theta)\right\|_2^2 \leq \rho \left\|\nabla h(\theta)\right\|_2^2 + \sigma_\theta^2$,

$$\leq C_t \left[\left\|\nabla h(\theta_t)\right\|_2^2 + \sigma_\theta^2\right]$$

We can use the bound in Theorem 2 to get a handle on the sub-optimality in h. Defining θ_T, θ^* s.t $f_T = f(\theta_T)$ and $z^* = f(\theta^*)$, and using that h is L_θ smooth,

$$\begin{aligned}
\left\|\nabla h(\theta_T)\right\|_2^2 &\leq 2L_\theta \left[h(\theta_T) - h(\theta^*)\right] = 2L_\theta \left[\ell(z_T) - \ell(z^*)\right] \\
&\leq L\, L_\theta \left\|f_t - z^*\right\|_2^2 \leq L\, L_\theta \left[\left(1 - \frac{\mu}{2L}\right)^T \left\|f_0 - z^*\right\|_2^2 + 3\kappa\, \epsilon^2 + \frac{\sigma^2}{\mu L}\right] \qquad (17)
\end{aligned}$$

The above inequality is true for all T, and we will use this fact subsequently. Using Eq. (17) for iteration t:

$$\implies \mathbb{E}[\epsilon_{t+1}^2] \leq C_t L\, L_\theta \left[\left(1 - \frac{\mu}{2L}\right)^t \|f_0 - z^*\|_2^2 + 3\kappa\, \epsilon^2 + \frac{\sigma^2}{\mu L} + \frac{\sigma_\theta^2}{L\, L_\theta} \right]$$

In order to control the projection error, we want $\mathbb{E}[\epsilon_{t+1}^2] \leq \epsilon^2$. To ensure this, we need to set m_t s.t. the RHS of the above equation is less than ϵ^2.

$$\frac{\mu_g^2}{L_f^2} \exp\left({}^{m_t}/_{\kappa_g}\right) \geq \frac{L\, L_\theta \left[\left(1 - \frac{\mu}{2L}\right)^t \|f_0 - z^*\|_2^2 + 3\kappa\, \epsilon^2 + \frac{\sigma^2}{\mu L} + \frac{\sigma_\theta^2}{L\, L_\theta} \right]}{\epsilon^2}$$

$$\implies \exp\left({}^{m_t}/_{\kappa_g}\right) \geq \frac{L_f^2 L\, L_\theta \left[\exp\left({}^{-t}/_{2\kappa}\right) L_f^2 \|\theta_0 - \theta^*\|_2^2 + 3\kappa\, \epsilon^2 + \frac{\sigma^2}{\mu L} + \frac{\sigma_\theta^2}{L\, L_\theta} \right]}{\mu_g^2\, \epsilon^2}$$

$$\implies m_t \geq \kappa_g \log \left(\frac{L_f^2 L\, L_\theta \left[\exp\left({}^{-t}/_{2\kappa}\right) L_f^2 \|\theta_0 - \theta^*\|_2^2 + 3\kappa\, \epsilon^2 + \frac{\sigma^2}{\mu L} + \frac{\sigma_\theta^2}{L\, L_\theta} \right]}{\mu_g^2\, \epsilon^2} \right)$$

■

Appendix E. Additional Experimental Results

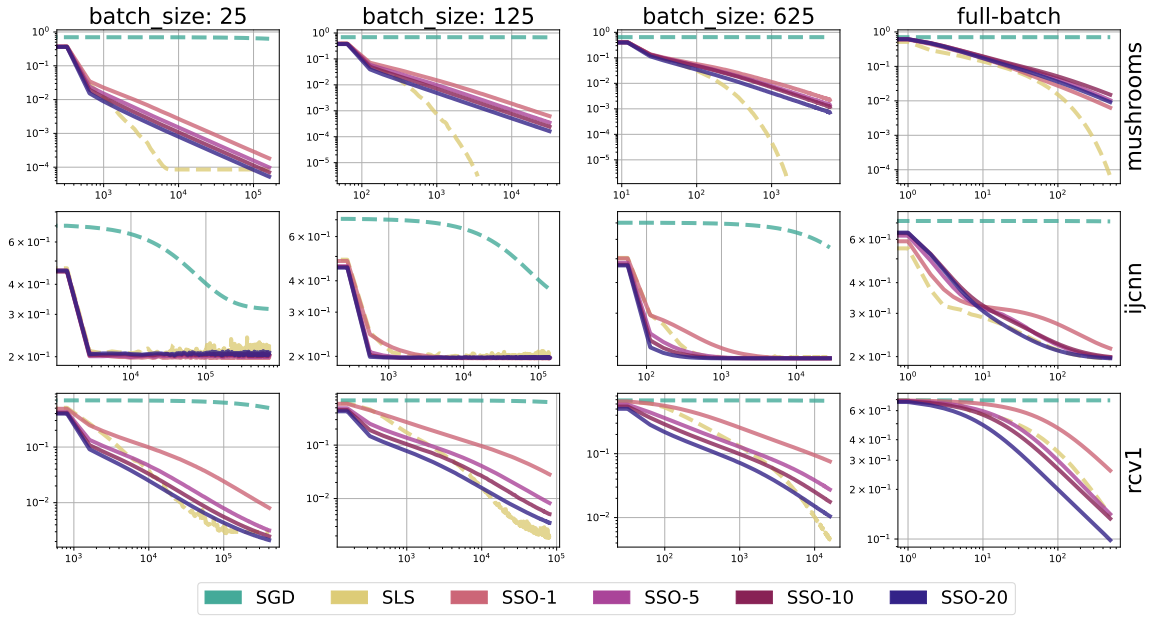

Figure 2: Comparison in terms of average loss of stochastic gradient decent, stochastic line-search, and finally stochastic surrogate optimization using the logistic loss. Each run was evaluated over three random seeds following the same initialization scheme. All plots are in log-log space to make trends between optimization algorithms more apparent. We note the SGD fails to perform on par with a more sophisticated algorithm like SLS when using its near optimal step-size proposed by theory. Similar to Figure 1, stochastic surrogate optimization remains on-par with SLS using its theoretically set step-size for almost all benchmarks, with the exception being mushrooms. We believe this is a by-product of separability in the mushrooms dataset, as well as the expanding step-size heuristic used by SLS. We note that in the full-batch setting, SGD reduces to GD, while SLS reduces to GD with a back-tracking line-search.

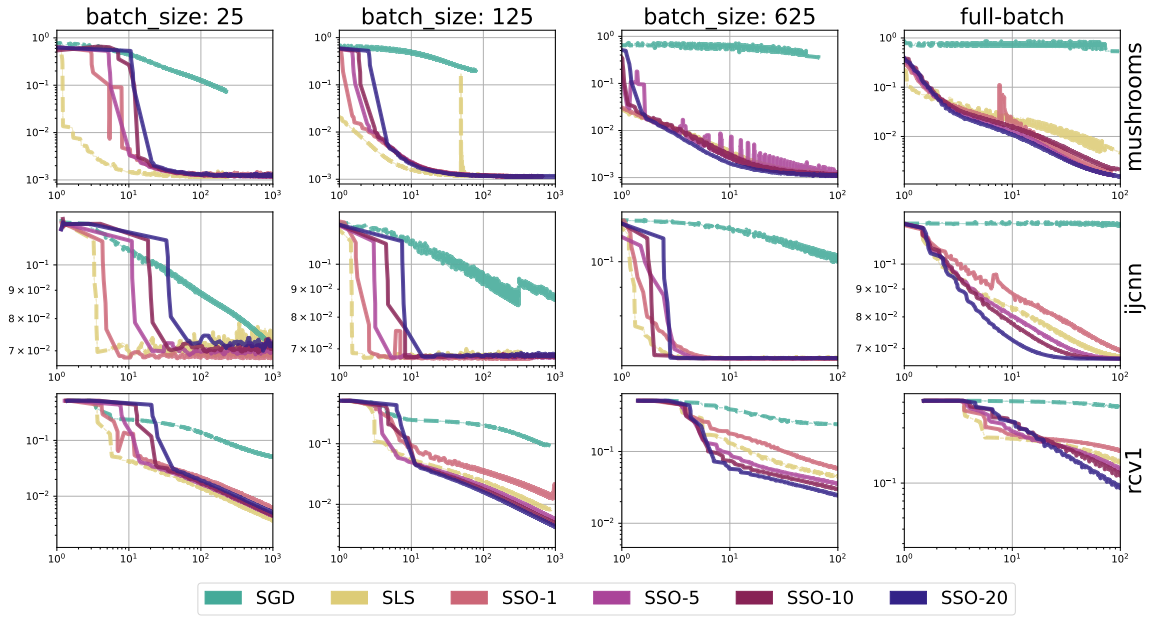

Figure 3: Comparison of the average loss as a function of time-elapsed between stochastic gradient decent, stochastic line-search, and finally stochastic surrogate optimization using the mean-squared error. All plots are in log-log space to make trends between optimization algorithms more apparent. The important thing to note is that most of the trends from Figure 1 given in the main text continue to hold. Namely, that SSO performs as well or better than SLS and SGD over all benchmarks and batch-sizes. Additionally, it seems that multiple steps of the surrogate can in some cases improve performance, as in the case of the full-batch setting. We also note that because examples were gathered at irregular time-intervals, simple confidence intervals are not included, and instead Gaussian smoothing is applied to the sequence of observed losses.

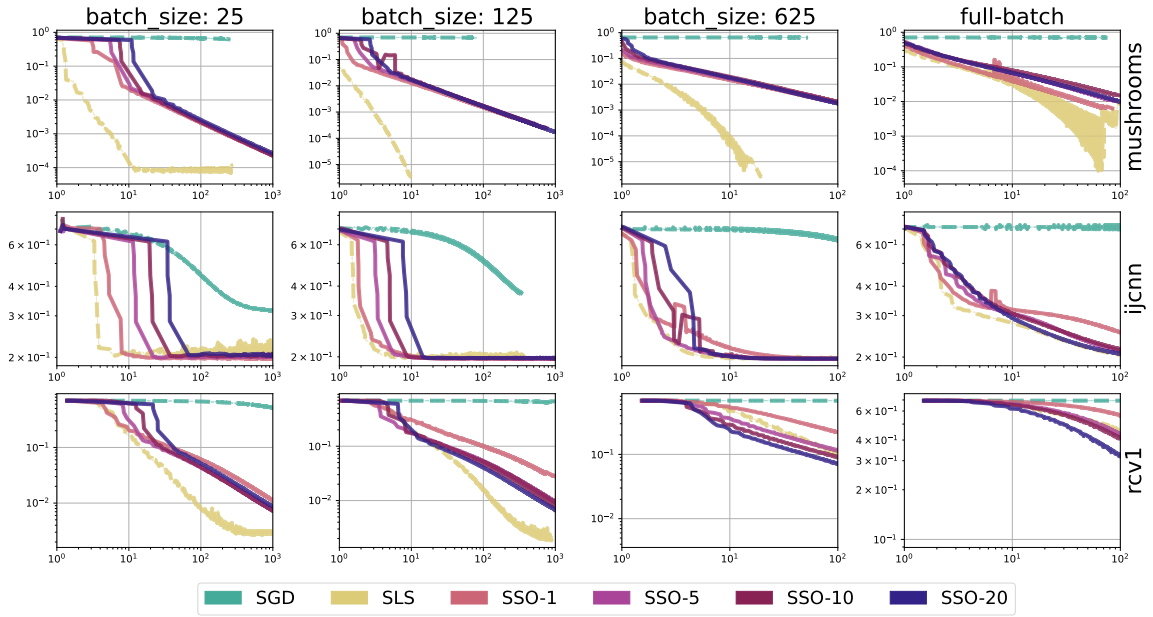

Figure 4: Comparison of the average loss as a function of time-elapsed between stochastic gradient decent, stochastic line-search, and finally stochastic surrogate optimization using the logistic loss. All plots are in log-log space to make trends between optimization algorithms more apparent. Here again, all trends hold from their associated optimization steps counterpart, Figure 2. Namely, that SSO performs as well or better than SLS and SGD over all benchmarks and batch-sizes. Additionally, it seems that multiple steps of the surrogate can in some cases improve performance, as in the case of the full-batch setting. Again, because examples were gathered at irregular time-intervals, simple confidence intervals are not included, and instead Gaussian smoothing is applied to the sequence of observed losses.

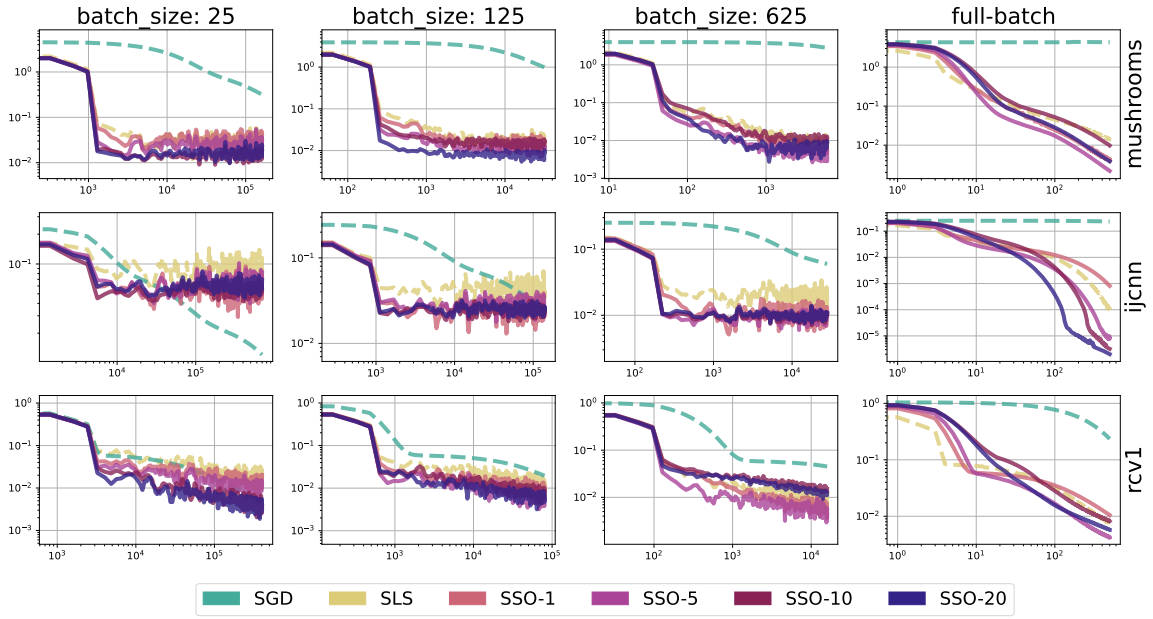

Figure 5: Comparison of the average gradient norm (of the original loss, and not the surrogate) between stochastic gradient decent, stochastic line-search, and finally stochastic surrogate optimization using the mean squared error loss. All plots are in log-log space to make trends between optimization algorithms more apparent. Here we see that all algorithms drive the gradient norm of their lower-bounded loss to reasonably high precision.

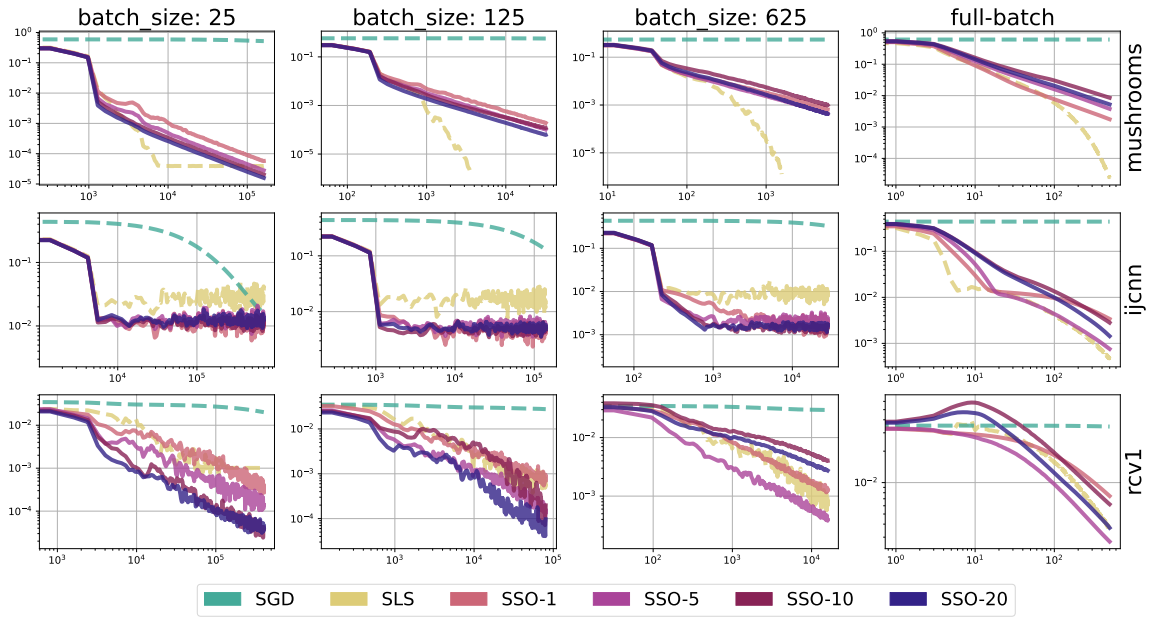

Figure 6: Comparison of the average gradient norm (of the original loss, and not the surrogate) between stochastic gradient decent, stochastic line-search, and finally stochastic surrogate optimization using the mean squared error loss. All plots are in log-log space to make trends between optimization algorithms more apparent. Here we see that all algorithms drive the gradient norm of their lower-bounded loss to reasonably high precision. We note that in the full-batch setting certain m, briefly increase in the gradient norm of the original loss-function, but eventually drive the loss, and the gradient norm towards zero.