Derivative free optimization via repeated classification

Tatsunori B. Hashimoto **Steve Yadlowsky** **John C. Duchi**
Department of Statistics, Stanford University, Stanford, CA, 94305
{thashim, syadlows, jduchi}@stanford.edu

Abstract

We develop an algorithm for minimizing a function using n batched function value measurements at each of T rounds by using classifiers to identify a function's sublevel set. We show that sufficiently accurate classifiers can achieve linear convergence rates, and show that the convergence rate is tied to the difficulty of active learning sublevel sets. Further, we show that the bootstrap is a computationally efficient approximation to the necessary classification scheme.

The end result is a computationally efficient derivative-free algorithm requiring no tuning that consistently outperforms other approaches on simulations, standard benchmarks, real-world DNA binding optimization, and airfoil design problems whenever batched function queries are natural.

1 Introduction

Consider the following abstract problem: given access to a function $f : \mathcal{X} \to \mathbf{R}$, where $\mathcal{X}$ is some space, find $x \in \mathcal{X}$ minimizing $f(x)$. We study an instantiation of this problem that trades sequential access to f for large batches of parallel queries—one can query f for its value over n points at each of T rounds. In this setting, we propose a general algorithm that effectively optimizes f whenever there is a family of classifiers $h : \mathcal{X} \to [0, 1]$ that can predict sublevel sets of f with high enough accuracy.

Our main motivation comes from settings in which n is large—on the order of hundreds to thousands—while possibly small relative to the size of $\mathcal{X}$. These types of problems occur in biological assays [21], physical simulations [27], and reinforcement learning problems [33] where parallel computation or high-throughput measurement systems allow efficient collection of large batches of data. More concretely, consider the optimization of protein binding affinity to DNA sequence targets from biosensor data [11, 21, 38]. In this case, assays measure binding of $n \geq 1000$s of sequences and are inherently parallel due to the fixed costs of setting up an experiment, while the time to measure a collection of sequences makes multiple sequential tests prohibitively time-consuming (so T must be small). In such problems, it is typically difficult to compute the gradients of f (if they even exist); consequently, we focus on derivative-free optimization (DFO, also known as zero-order optimization) techniques.

1.1 Problem statement and approach

The batched derivative free optimization problem consists of a sequence of rounds $t = 1, 2, \ldots, T$ in which we propose a distribution $p^{(t)}$, draw a sample of n candidates $X_i \overset{\text{iid}}{\sim} p^{(t)}$, and observe $Y_i = f(X_i)$. The goal is to find at least one example X_i for which the gap

$$\min_i f(X_i) - \inf_{x \in \mathcal{X}} f(x)$$

is small.

Our basic idea is conceptually simple: In each round, fit a classifier h predicting whether $Y_i \lesseqgtr \alpha^{(t)}$ for some threshold $\alpha^{(t)}$. Then, upweight points x that h predicts as $f(x) < \alpha^{(t)}$ and downweight the other points x for the proposal distribution $p^{(t)}$ for the next round.

This algorithm is inspired by classical cutting-plane algorithms [30, Sec. 3.2], which remove a constant fraction of the remaining feasible space at each iteration, and is extended into the stochastic setting based on multiplicative weights algorithms [25, 3]. We present the overall algorithm as Algorithm 1.

Proceedings of the 21^{st} International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84.

Algorithm 1 Cutting-planes using classifiers

Require: Objective f, Action space $\mathcal{X}$, hypothesis class $\mathcal{H}$.

Set $p^{(0)}(x) = 1/|\mathcal{X}|$
Draw $X^{(0)} \sim p^{(0)}$.
Observe $Y^{(0)} = f(X^{(0)})$
for $t \in \{1 \dots T\}$ **do**
  Set $\alpha^{(t)} = \text{median}(\{Y_i^{(t)}\}_{i=1}^n)$
  Set $h^{(t)} \in \mathcal{H}$ as the loss minimizer of L over $(X^{(0)}, Y^{(0)} > \alpha^{(t)}) \dots (X^{(t-1)}, Y^{(t-1)} > \alpha^{(t)})$.
  Set $p^{(t)}(x) \propto p^{(t-1)}(x)(1 - \eta h^{(t)}(x))$
  Draw $X^{(t)} \sim p^{(t)}$
  Observe $Y^{(t)} = f(X^{(t)})$.
end for
Set $i^* = \arg\min_i Y_i^{(T)}$
return $X_{i^*}^{(T)}$.

1.2 Related work

When, as is typical in optimization, one has substantial *sequential* access to f, meaning that T can be large, there are a number of major approaches to optimization. Bayesian optimization [34, 7] and kernel-based bandits [9] construct an explicit surrogate function to minimize; often, one assumes it is possible to perfectly model the function f. Local search algorithms [12, 26] emulate gradient descent via finite-difference and local function evaluations. Our work differs conceptually in two ways: first, we think of T as being small, while n is large, and second, we represent a function f by approximating its sublevel sets. Existing batched derivative-free optimizers encounter computational difficulties for batch sizes beyond dozens of points [16]. Our sublevel set approach scales to large batches of queries by simply sampling from the current sublevel set approximation.

While other researchers have considered level set estimation in the context of Bayesian optimization [17, 7] and evolutionary algorithms [29], these use the level set to augment a traditional optimization algorithm. We show good sublevel set predictions alone are sufficient to achieve linear convergence. Moreover, given the extraordinary empirical success of modern classification algorithms, e.g. deep networks for image classification [22], it is natural to develop algorithms for derivative-free optimization based on fitting a sequence of classifiers. Yu et al. [40] also propose classification based on optimization, but their approach assumes a classifier constrained to never misclassify near the optimum, making the problem trivial.

1.3 Contributions

We present Algorithm 1 and characterize its convergence rate with appropriate classifiers and show how it relates to measures of difficulty in active learning. We extend this basic approach, which may be computationally challenging, to an approach based on bootstrap resampling that is empirically quite effective and—in certain nice-enough scenarios—has provable guarantees of convergence.

We provide empirical results on a number of different tasks: random (simulated) problems, airfoil (device) design based on physical simulators, and finding strongly-binding proteins based on DNA assays. We show that a black-box approach with random forests is highly effective within a few rounds T of sequential classification; this approach provides advantages in the large batch setting.

The approach to optimization via classification has a number of practical benefits, many of which we verify experimentally. It is possible to incorporate prior knowledge in DFO through domain-specific classifiers, and in more generic optimization problems one can use black-box classifiers such as random forests. Any sufficiently accurate classifier guarantees optimization performance and can leverage the large-batch data collection biological and physical problems essentially necessitate. Finally, one does not even need to evaluate f: it is possible to apply this framework with pairwise comparison or ordinal measurements of f.

2 Cutting planes via classification

Our starting point is a collection of "basic" results that apply to classification-based schemes and associated convergence results. Throughout this section, we assume we fit classifiers using pairs (x, z), where z is a 0/1 label of negative (low $f(x)$) or positive (high $f(x)$) class. We begin by demonstrating that two quantities govern the convergence of the optimizer: (1) the frequency with which the classifier misclassifies (and thus downweights) the optimum x^* relative to the multiplicative weight η, and (2) the fraction of the feasible space each iteration removes.

If the classifier $h^{(t)}(x)$ exactly recovers the sublevel set ($h^{(t)}(x) < 0$ iff $f(x) < \alpha^{(t)}$), $\alpha^{(t)}$ is at most the population median of $f(X^{(t)})$, and $\mathcal{X}$ is finite, the basic cutting plane bound immediately implies that

$$\log\left[\mathbf{P}_{x\sim p^{(T)}}\left(f(x) = \min_{x^*\in\mathcal{X}} f(x^*)\right)\right] \geq \min\left(T\log\left(\frac{2}{2-\eta}\right) - \log(|\mathcal{X}|), 0\right).$$

It is not obvious that such a guarantee continues to hold for inaccurate $h^{(t)}$: it may accidentally misclassify the optimum x^*, and the thresholds $\alpha^{(t)}$ may not rapidly decrease the function value. To address these issues, we provide a careful analysis in the coming sections: first, we show the convergence guarantees implied by Algorithm 1 as a function of classification errors (Theorem 1), after which we propose a classification strategy directly controlling errors (Sec. 2.2), and finally we give a computationally tractable approximation (Sec. 3).

2.1 Cutting plane style bound

We begin with our basic convergence result. Letting $p^{(t)}$ and $h^{(t)}$ be a sequence of distributions and classifiers on $\mathcal{X}$, the convergence rate depends on two quantities: the coverage (number of items cut)

$$\sum_{x\in\mathcal{X}} h^{(t)}(x)p^{(t-1)}(x)$$

and the number of times a hypothesis downweights item x (because $f(x)$ is too large), which we denote $M_T(x) := \sum_{t=1}^T h^{(t)}(x)$. We have the following

Theorem 1. *Let $\gamma > 0$ and assume that for all t,*

$$\sum_{x\in\mathcal{X}} h^{(t)}(x)p^{(t-1)}(x) \geq \gamma$$

where $p^{(t)}(x) \propto p^{(t-1)}(x)(1-\eta h^{(t)}(x))$ as in Alg. 1. Let $\eta \in [0, 1/2]$ and $p^{(0)}$ be uniform. Then for all $x \in \mathcal{X}$,

$$\log p^{(T)}(x) \geq \frac{\gamma\eta}{\eta+2}T - \eta(\eta+1)M_T(x) - \log(2|\mathcal{X}|).$$

The theorem follows from a modification of standard multiplicative weight algorithm guarantees [3]; see supplemental section A.1 for a full proof.

We say that our algorithm converges *linearly* if $\log p^{(t)}(x) \gtrsim t$. In the context of Theorem 1, choice of η maximizing $-(\eta^2+\eta)M_T(x^*) + \frac{\eta}{\eta+2}\gamma T$ yields such convergence, as picking η sufficiently small that

$$T - \frac{(\eta+1)(\eta+2)}{\gamma}M_T(x^*) = \Omega(T)$$

guarantees linear convergence if $2M_T(x^*) < T\gamma$.

A simpler form of the above bound for a fixed η shows the linear convergence behavior.

Corollary 1. *Let $x \in \mathcal{X}$, where $q_T(x) := \frac{M_T(x)}{\gamma T} \leq 1/4$. Under the conditions of Theorem 1,*

$$\log(p^{(T)}(x)) \geq \min\left(\frac{1}{5}, \frac{1}{3} - \frac{4q_T(x)}{3}\right)\frac{\gamma T}{2} - \log(2|\mathcal{X}|)$$

and

$$\frac{1}{4} - \frac{\log(2|\mathcal{X}|)}{2\gamma T} \leq q_T(x).$$

The condition $q_T(x) \geq \frac{1}{4} - \frac{1}{2\gamma T}\log(2|\mathcal{X}|)$ arises because if $M_T(x)$ is small, then eventually we must have $p^{(T)}(x) \geq 1-\gamma$, and any classifier h which fulfils the condition $\sum_{x\in\mathcal{X}} h^{(t)}(x)p^{(t-1)}(x) \geq \gamma$ in Thm. 1 must downweight x. At this point, we can identify the optimum exactly with $O(1/(1-\gamma))$ additional draws.

The corollary shows that if $M_T(x^*) = 0$ and $\gamma = (1-1/e) - 1/2 < 0$, we recover a linear cutting-plane-like convergence rate [cf. 30], which makes constant progress in volume reduction in each iteration.

2.2 Consistent selective strategy for strong control of error

The basic guarantee of Theorem 1 requires relatively few mistakes on x^*, or at least on a point x with $f(x) \approx f(x^*)$, to achieve good performance in optimization. It is thus important to develop careful classification strategies that are conservative: they do not prematurely cut out values x whose performance is uncertain. With this in mind, we now show how consistent selective classification strategies [15] (related to active learning techniques, and which abstain on "uncertain" examples similar to the Knows-What-It-Knows framework [23, 2]) allow us to achieve linear convergence when the classification problems are realizable using a low-complexity hypothesis class.

The central idea is to only classify an example if all zero-error hypotheses agree on the label, and otherwise abstain. Since any hypothesis achieving zero population error must have zero training set errors, we will only label points in a way consistent with the true labels. El-Yaniv and Wiener [15] define the following *consistent selective strategy* (CSS).

Definition 1 (Consistent selective strategy)**.** *For a hypothesis class $\mathcal{H}$ and training sample S, the* version space $\mathsf{VS}_{\mathcal{H},S_m} \subset \mathcal{H}$ *is the set of all hypotheses which perfectly classify S_m. The* consistent selective strategy *is the classifier*

$$h(x) = \begin{cases} 1 & \text{if } \forall g \in \mathsf{VS}_{\mathcal{H},S_m}, g(x) = 1 \\ 0 & \text{if } \forall g \in \mathsf{VS}_{\mathcal{H},S_m}, g(x) = 0 \\ \textit{no decision} & \textit{otherwise.} \end{cases}$$

Applied to our optimizer, this strategy enables safely downweighting examples whenever they are classified as being outside the sublevel set. Optimization performance guarantees then come from demonstrating that at each iteration the selective strategy does not abstain on too many examples.

The rate of abstention for a selective classifier is related to the difficulty of disagreement based active learning, controlled by the disagreement coefficient [18].

Definition 2. *The* disagreement ball *of a hypothesis class $\mathcal{H}$ for distribution P is*

$$B_{\mathcal{H},P}(h,r) := \{h' \in \mathcal{H} \mid P(h(X) \neq h'(X)) \leq r\}.$$

The disagreement region of a subset $\mathcal{G} \subset \mathcal{H}$ *is*

$$\mathsf{Dis}(\mathcal{G}) := \{x \in \mathcal{X} \mid \exists h_1, h_2 \in \mathcal{G} \text{ s.t. } h_1(x) \neq h_2(x)\}.$$

The disagreement coefficient Δ_h *of the hypothesis class $\mathcal{H}$ for the distribution P is*

$$\Delta_h := \sup_{r>0} \frac{P(X \in \mathsf{Dis}(B_{\mathcal{H},P}(h,r)))}{r}.$$

The disagreement coefficient directly bounds the abstention rate as a function of generalization error.

Theorem 2. *Let h be the CSS classifier in definition 1, and let $h^* \in \mathcal{H}$ be a classifier achieving zero risk. If $\mathbf{P}(g(X) \neq h^*(X)) < \epsilon$ for all $g \in \mathsf{VS}_{\mathcal{H},S_m}$, then CSS achieves coverage*

$$\mathbf{P}(h(X) = \textit{no decision}) \leq \Delta_{h^*}\epsilon$$

This follows from the definition of the disagreement coefficient, and the size of the version space (Supp. section A.1 contains a full proof).

The dependence of our results on the disagreement coefficient implies a reduction from zeroth order optimization to disagreement based active learning [15] and selective classification [39] over sublevel sets.

Implementing the CSS classifier may be somewhat challenging: given a particular point x, one must verify that all hypotheses consistent with the data classify it identically. In many cases, this requires training a classifier on the current training sample $S^{(t)}$ at iteration t, coupled with x labeled positively, and then retraining the classifier with x labeled negatively [39]. This cost can be prohibitive. (Of course, implementing the multiplicative weights-update algorithm over $x \in \mathcal{X}$ is in general difficult as well, but in a number of application scenarios we know enough about $\mathcal{H}$ to be able to approximate sampling from $p^{(t)}$ in Alg. 1.)

A natural strategy is to use the CSS classifier as part of Algorithm 1, setting all `no decision` outputs to the zero class, only removing points confidently above the level set $\alpha^{(t)}$. That is, in round t of the algorithm, given samples $S = (X^{(t)}, Z^{(t)})$, we define

$$h^{(t)}(x) = \begin{cases} 1 & \text{if } \forall g \in \mathsf{VS}_{\mathcal{H},S}, g(x) = 1 \\ 0 & \text{if } \forall g \in \mathsf{VS}_{\mathcal{H},S}, g(x) = 0 \\ 0 & \text{otherwise.} \end{cases}$$

There is some tension between classifying examples correctly and cutting out bad $x \in \mathcal{X}$, which the next theorem shows we can address by choosing large enough sample sizes n.

Theorem 3. *Let $\mathcal{H}$ be a hypothesis class containing indicator functions for the sublevel sets of f, with VC-dimension V and disagreement coefficient Δ_h. There exists a numerical constant $C < \infty$ such that for all $\delta \in [0,1]$, $\epsilon \in [0,1]$, and $\gamma \in (\Delta_h\epsilon, \frac{1}{2})$, and*

$$n \geq \max\Big\{C\epsilon^{-1}[V\log(\epsilon^{-1}) + \log(\delta^{-1}) + \log(2T)], \\ \frac{1}{2(\gamma-0.5)^2}(\log(\delta^{-1}) + \log(2T))\Big\},$$

with probability at least $1-\delta$

$$\log(p^{(T)}(x^*)) \geq \min\Big\{(\gamma - \Delta_h\epsilon)\frac{\eta}{\eta+2}T - \log(2|\mathcal{X}|), \\ \log(1-\gamma)\Big\}$$

after T rounds of Algorithm 1.

The proof follows from combining the selective classification bound with standard VC dimension arguments to obtain the sample size requirement (Supp. A.1 contains a full proof).

Thus if Δ_h is small, such as $\log(|\mathcal{X}|)$, then choosing $\epsilon = \Delta_h^{-1}$ achieves exponential improvements over random sampling. In the worst case, $\Delta_h = O(|\mathcal{X}|)$, but small Δ_h are known for many problems, for example for linear classification with continuous $\mathcal{X}$ over densities bounded away from zero, $\Delta_h = \text{poly}(\log(\text{Vol}(\mathcal{X})))$, which would result in linear convergence rates (Theorem 7.16, [18]).

Using recent bounds for the disagreement coefficient for linear separators [5], we can show that for linear optimization over a convex domain, the CSS based optimization algorithm above achieves linear convergence with $O(d^{3/2}\log(d^{1/2}) - d^{1/2}\log(3T\delta))$ samples with probability at least $1-\delta$ (for lack of space, we present this as Theorem A.2 in the supplement.)

When the classification problem is non-realizable, but the Bayes-optimal hypothesis does not misclassify x^*, an analogous result holds through the agnostic selective classification framework of Wiener and El-Yaniv [39]. The full result is in supplemental Theorem A.7.

3 Computationally efficient approximations

While selective classification provides sufficient control of error for linear convergence, it is generally computationally intractable. However, a bootstrap resampling algorithm [14] approximates selective classification well enough to provide finite sample guarantees in parametric settings. Our analysis provides intuition

for the empirical observation that selective classification via the bootstrap works well in many real-world problems [1].

Formally, consider a parametric family $\{P_\theta\}_{\theta\in\Theta}$ of conditional distributions $Z \mid X \in [0,1]$ with compact parameter space Θ. Given n samples $X_1, \ldots, X_n$, we observe $Z_i|X_i \sim P_{\theta^*}$ with $\theta^* \in \operatorname{int}\Theta$.

Let $\ell_\theta(x,z) = -\log(P_\theta(z|x))$ be the negative log likelihood of z, which majorizes the 0-1 loss of the linear hypothesis class $\ell_\theta(x,z) \geq \mathbf{1}_{\{(2z-1)x^\top\theta<0\}}$.

Define the weighted likelihood

$$L_n(\theta, u) \equiv \frac{1}{n}\sum_{i=1}^{n}(1+u_i)\ell_\theta(X_i, Z_i),$$

and consider the following multiplier bootstrap algorithm [14, 35], parameterized by $B \in \mathbf{N}$ and variance σ^2. σ adds *additional* variation in the estimates to increase parameter coverage.

1. Draw $\{(X_i, Z_i)\}_{i=1}^n$ from $\mathbf{P}$.
2. Compute $\theta_n = \arg\min_\theta L_n(\theta, 0)$.
3. For $b = 1$ to B,
   (a) Draw $u_b \overset{\text{iid}}{\sim} \text{Uni}[-1, 1]$.
   (b) Compute
   $$\theta^\circ_{u_b} = \sigma(\arg\min_\theta L_n(\theta, u_b) - \theta_n) + \theta_n.$$
4. Define the estimator
   $$h^\circ(x) = \begin{cases} 1 & \text{if } \forall b \in [B], x^\top\theta^\circ_{u_b} > 0 \\ 0 & \text{if } \forall b \in [B], x^\top\theta^\circ_{u_b} \leq 0 \\ \text{no decision} & \text{otherwise.} \end{cases}.$$

For linear classifiers with strongly convex losses, this algorithm obtains selective classification guarantees under appropriate regularity conditions as presented in the following theorem.

Theorem 4. *Assume ℓ_θ is twice differentiable and fulfils $\|\nabla\ell_\theta(X,Z)\| \leq R$, and $\left\|\nabla^2\ell_\theta(X,Z)\right\|_{op} \leq S$ almost surely. Additionally, assume $L_n(\theta, 1)$ is γ-strongly convex and that $\nabla^2 L_n(\theta, 1)$ is M-Lipschitz with probability one.*

For h° defined above and $x \in \mathcal{X}$,

$$P(x^\top\theta^* \leq 0 \text{ and } h^\circ_u(x) = 1) < \delta.$$

Further, the abstention rate is bounded by

$$\int_{x\in\mathbf{R}^d} \mathbf{1}_{\{h^\circ_u(x)=\emptyset\}} p(x)dx \leq \epsilon\Delta_h$$

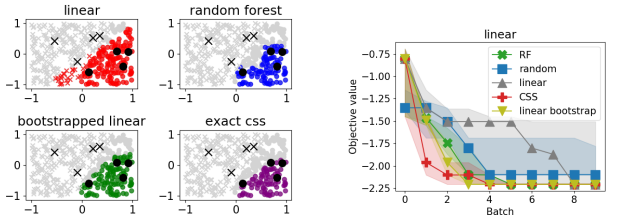
with probability $1-\delta$ whenever

$$B \geq 15\log(3/\delta),$$
$$\sigma = O(d^{1/2} + \log(1/\delta)^{1/2} + n^{-1/2}),$$
$$\epsilon = O\left(\sigma^2 n^{-1}\log(B/\delta)\right),$$

and

$$n \geq 2\log(2d/\delta)S/\gamma^2.$$

Due to length, the proof and full statement with constants appears in the appendix as Theorem A.4, with a sketch provided here: we first show that a given quadratic version space and a multivariate Gaussian sample θ^{quad} obtains the selective classification guarantees (Lemmas A.3,A.4,A.5). We then show that $\theta^\circ \approx \theta^{\text{quad}}$ to order n^{-1} which is sufficient to recover Theorem A.4.



(a) Classification confidences formed by bootstrapping approximate selective classification.

(b) Bootstrapping results in more consistent identification of minima.

Figure 1. Bootstrap consensus provides more conservative classification boundaries which prevents repeatedly misclassifying the minimum, compared to direct loss minimization (panel b, triangle).

The $d\Delta_h$ abstention rate in this bound is d times the original selective classification result. This additional factor of d appearing in σ^2 arises from the difference between finding an optimum within a ball and randomly sampling it: random vectors concentrate within $O(1/d)$ of the origin, while the maximum possible value is 1. This gap forces us to scale the variance in the decision function by σ (step 3b). We present selective classification approximation bounds analogous to Theorem 3 for linear optimization in the Appendix as Theorem A.5.

To illustrate our results through simulations, consider a optimizing a two-dimensional linear function in the unit box. Figure 1a shows the set of downweighted points (colored points) for various algorithms on classifying a single superlevel set based on eight observations (black points). Observe how linear downweights many points (colored 'x'), in contrast to exact CSS, which only downweights points guaranteed to be in the superlevel set. Errors of this type combined with Alg. 1 result in optimizers which fail to find the true

minimum depending on initialization (Figure 1b). The bootstrapped linear classifier behaves similarly to CSS, but is looser due to the non-asymptotic setting. Random forests, another type of bootstrapped classifier is surprisingly good at approximating CSS, despite not making use of the linearity of the decision boundary.

4 Partial order based optimization

One benefit of optimizing via classification is that the algorithm only requires total ordering amongst the elements. Specifically, step 6 of Algorithm 1 only requires threshold comparisons against a percentile selected in step 5. This enables optimization under pairwise comparison feedback. At each round, instead of observing $f(X^{(t)})$, we observe $g(X_i^{(t)}, X_j^{(t)}) = 1_{f(X_i^{(t)}) < f(X_j^{(t)})}$, which is a natural form of feedback in domains such as human surveys [31] or matched biological experiments [19].

Given the pairwise comparison function g, the threshold $f(X^{(t)}) < \alpha^{(t)}$ can be replaced with the following stochastic quantile estimator:

$$\hat{f}(X_i^{(t)}) = \sum_{k=1}^{c} g(X_{I_k}^{(t)}, X_i^{(t)}) \leq 0.5, \qquad (1)$$

where $I_k \sim \text{Unif}(\{1, 2 \dots c\})$ with cn total pairwise comparisons. We show that $c > 10$ seems to work well in practice, and more sophisticated preference aggregation algorithms may reduce the number of comparisons even further.

5 Experimental evidence

We evaluate Algorithm 1 as a DFO algorithm across a few real-world experimental design benchmarks, common synthetic toy optimization problems, and benchmarks that allow only pairwise function value comparisons. The small-batch (n = 1-10) nature of hyperparameter optimization problems is outside the scope of our work, even though they are common DFO problems.

For constructing the classifier in Algorithm 1, we apply ensembled decision trees with a consensus decision defined as 75% of trees agreeing on the label (referred to as CLASSIFY-RF). This particular classifier works in a black-box setting, and is highly effective across all problem domains with no tuning. We also empirically investigate the importance of well-specified hypotheses and consensus ensembling and show improved results for ensembles of linear classifiers and problem specific classifiers, which we call CLASSIFY-TUNED.

In order to demonstrate that no special tuning is necessary, the same constants are used in the optimizer for all experiments, and the classifiers use off-the-shelf implementations from SCIKIT-LEARN with no tuning.

For sampling points according to the weighted distribution in Algorithm 1, we enumerate for discrete action spaces $\mathcal{X}$, and for continuous $\mathcal{X}$ we perturb samples from the previous rounds using a Gaussian and use importance sampling to approximate the target distribution. Although exact sampling for the continuous case would be time-consuming, the Gaussian perturbation heuristic is fast, and seems to work well enough for the functions tested here.

As a baseline, we compare to the following algorithms

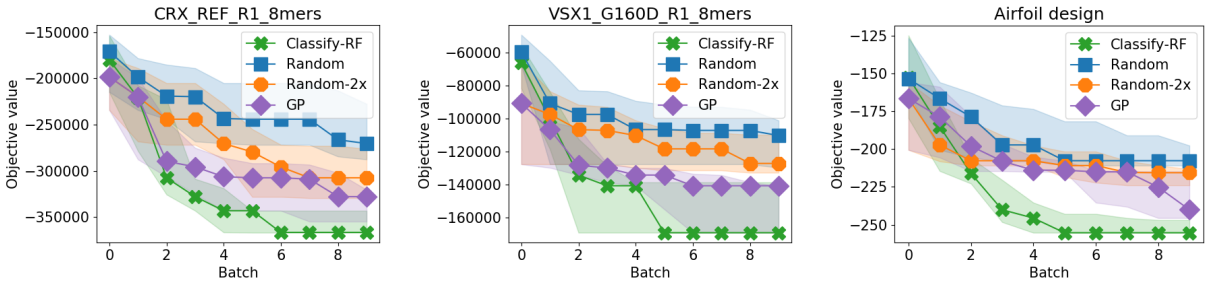
- Random sampling (RANDOM)
- Randomly sampling double the batch size (RANDOM-2X), which is a strong baseline recently shown to outperform many derivative-free optimizers [24].
- The evolutionary strategy (CMA-ES) for continuous problems, due to its high-performance in black box optimization competitions as well as inherent applicability to the large batch setting [26]
- The Bayesian optimization algorithm provided by GPYOPT[4] (GP) for both continuous and discrete problems, using expected improvement as the acquisition function. We use the 'random' evaluator which implements an epsilon-greedy batching strategy, since the large batch sizes (100-1000) makes the use of more sophisticated evaluators completely intractable. The default RBF kernel was used in all experiments presented here. The $^3/_2$- and $^5/_2$-Matern kernels and string kernels were tried where appropriate, but did not provide any performance improvements.

In terms of runtime, all computations for CLASSIFY-RF take less than 1 second per iteration compared to 0.1s for CMA-ES and 1.5 minutes for GPYOPT. All experiments were replicated fifteen times to measure variability with respect to initialization.

All new benchmark functions and reference implementations are made available at `http://bit.ly/2FgiIxA`.

5.1 Designing optimal DNA sequences

The publicly available protein binding microarray (PBM) dataset consisting of 201 separate assays [6] allows us to accurately benchmark the optimization protein binding over DNA sequences. In each assay, the binding affinity between a particular DNA-binding protein (transcription factor) and all 8-base DNA sequences are measured using a microarray.



(a) Binding to the CRX protein (b) Binding to the VSX1 protein (c) High-lift airfoil design

Figure 2. Performance on two types of real-world batched zeroth-order optimization tasks. CLASSIFY-RF consistently outperforms baselines and even randomly sampling twice the batch size. The line shows median function value over runs, shaded area is quartiles.

This dataset defines 201 separate discrete optimization problems. For each protein, the objective function is the negative binding affinity (as measured by fluorescence), the batch size is 100 (corresponding roughly to the size of a typical 96-well plate), across ten rounds. Each possible action corresponds to measuring the binding affinity of a particular 8-base DNA sequence exactly. The actions are featurized by considering the binary encoding of whether a base exists in a position, resulting in a 32-dimensional space. This emulates the task of finding the DNA binding sequence of a protein using purely low-throughput methods.

Figure 2a,2b shows the optimization traces of two randomly sampled examples, where the lines indicate median achieved function value over 15 random initializations, and the shading indicates quartiles. CLASSIFY-RF shows consistent improvements over all discrete action space baselines. For evaluation, we further sample 20 problems and find that the median binding affinity found across replicates is strictly better on 16 out of 20, and tied with the Gaussian process on 2.

In this case, the high performance of random forests is relatively unsurprising, as random forests are known to be high-performance classifiers for DNA sequence recognition tasks [10, 21].

5.2 Designing high-lift airfoils

Airfoil design, and other simulator-based objectives are well-suited to the batched, classification based optimization framework, as 30-40 simulations can be run in parallel on modern multicore computers. In the airfoil design case, the simulator is a 2-D aerodynamics simulator for airfoils [13].

The objective function is the negative of lift divided by drag (with a zero whenever the simulator throws an error) and the action space is the set of all common airfoils (NACA-series 4 airfoils). The airfoils are featurized by taking the coordinates around the perimeter of the airfoil as defined in the Selig airfoil format. This results in a highly-correlated two hundred dimensional feature space. The batch size is 30 (corresponding to the number of cores in our machine) and $T = 10$ rounds of evaluations are performed.

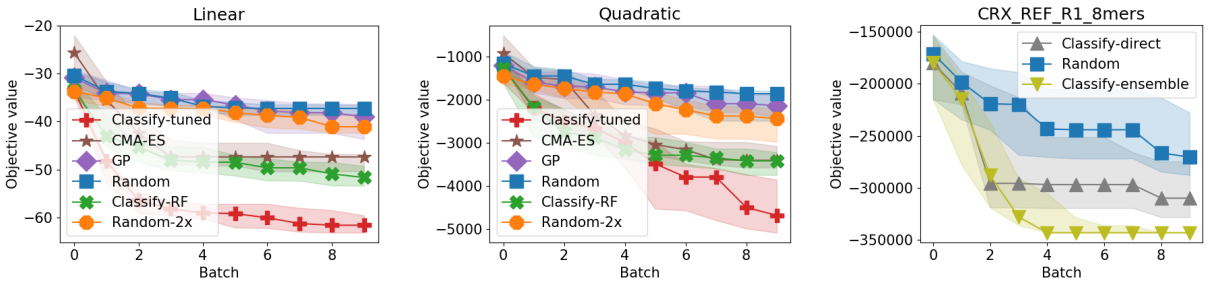
We find in Figure 2c that the CLASSIFY-RF algorithm converges to the optimal airfoil in only five rounds, and does so consistently, unlike the baselines. The Gaussian process beat the twice-random baseline, since the radial basis kernel is well-suited for this task (as lift is relatively smooth over ℓ_2 distance between airfoils) but did not perform as well as the CLASSIFY-RF algorithm.

5.3 Gains from designed classifiers and ensembles

Matching the classifier and objective function generally results in large improvements in optimization performance. We test two continuous optimization problems in $[-1, 1]^{300}$, optimizing a random linear function, and optimizing a random sum of a quadratic and linear functions. For this high dimensional task, we use a batch size of 1000. In both cases we compare continuous baselines with CLASSIFY-RF and CLASSIFY-TUNE which uses a linear classifier.

We find that the use of the correct hypothesis class gives dramatic improvements over baseline in the linear case (Figure 3a) and continues to give substantial improvements even when a large quadratic term is added, making the hypothesis class misspecified (Figure 3b). The CLASSIFY-RF does not do as well as this custom classifier, but continues to do as well as the best baseline algorithm (CMA-ES).

We also find that using an ensembled classifier is an important for optimization. Figure 3c shows an example run on the DNA binding task comparing the consensus of an ensemble of logistic regression classifiers against a single logistic regression classifier. Although both algorithms perform well in early iterations, the



(a) Random linear function (b) Linear+quadratic function (c) Ensembling classifiers improves optimization performance

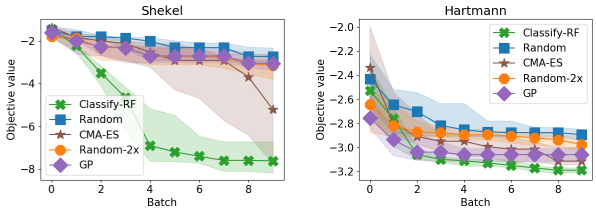
Figure 3. Testing the importance of ensembling and well-specified hypothesis class in synthetic data where the hypothesis for CLASSIFY-TUNED exactly matches level set (panel a), matches level sets with some error (panel b). Ensembling also consistently improves performance, and reduces dependence on initialization (panel c)

single logistic regression algorithm gets 'stuck' earlier and finds a suboptimal local minima, due to an accumulation of errors. Ensembling consistently reduces such behavior.

5.4 Low-dimensional synthetic benchmarks

We additionally evaluate on two common synthetic benchmarks (Figure 4a,4b). Although these tasks are not the focus of the work, we show that the CLASSIFY-RF is surprisingly good as a general black box optimizer when the batch sizes are large.

We consider a batch size of 500 and ten steps due to the moderate dimensionality and multi-modality relative to the number of steps. We find qualitatively similar results to before, with CLASSIFY-RF outperforming other algorithms and CMA-ES as the best baseline.



(a) Shekel (4d) (b) Hartmann (6d)

Figure 4. CLASSIFY-RF outperforms baselines on synthetic benchmark functions with large batches

5.5 Optimizing with pairwise comparisons

Finally, we demonstrate that we can optimize a function using only pairwise comparisons. In Figure 5 we show the optimization performance when using the ordering estimator from equation 1.

For small numbers of comparisons per element ($c = 5$) we find substantial loss of performance, but once we observe at least 10 pairwise comparisons per proposed action, we are able to reliably optimize as well as the full function value case. This suggests that classification based optimization can handle pairwise feedback with little loss in efficiency.

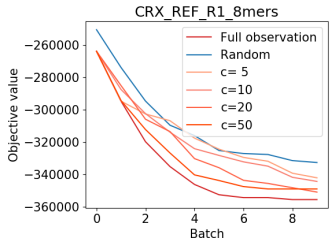


Figure 5. Optimization with pairwise comparisons between each action and a small set of (c) randomly selected actions. Between 10-20 pairwise comparisons per action gives sufficient information to fully optimize the function.

6 Discussion

Our work demonstrates that the classification-based approach to derivative-free optimization is effective and principled, but leaves open several theoretical and practical questions. In terms of theory, it is not clear whether a modified algorithm can make use of empirical risk minimizers instead of perfect selective classifiers. In practice, we have left the question of tractably sampling from $p^{(t)}$, as well as how to appropriately handle smaller-batch settings of $d > n$.

References

[1] N. Abe and M. Hiroshi. Query learning strategies using boosting and bagging. In *Proceedings of the Fifteenth International Conference on Machine Learning*, volume 1, 1998.

[2] J. Abernethy, K. Amin, M. Draief, and M. Kearns. Large-scale bandit problems and KWIK learning. In *Proceedings of the 30th International Conference on Machine Learning*, 2013.

[3] S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[4] T. G. authors. GPyOpt: A bayesian optimization framework in python. `http://github.com/SheffieldML/GPyOpt`, 2016.

[5] M.-F. Balcan and P. M. Long. Active and passive learning of linear separators under log-concave distributions. In *Proceedings of the Twenty Sixth Annual Conference on Computational Learning Theory*, pages 288–316, 2013.

[6] L. A. Barrera, A. Vedenko, J. V. Kurland, J. M. Rogers, S. S. Gisselbrecht, E. J. Rossin, J. Woodard, L. Mariani, K. H. Kock, S. Inukai, et al. Survey of variation in human transcription factors reveals prevalent dna binding changes. *Science*, 351(6280):1450–1454, 2016.

[7] I. Bogunovic, J. Scarlett, A. Krause, and V. Cevher. Truncated variance reduction: A unified approach to bayesian optimization and level-set estimation. In *Advances in Neural Information Processing Systems 29*, pages 1507–1515, 2016.

[8] S. Boucheron, G. Lugosi, and P. Massart. *Concentration Inequalities: a Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

[9] S. Bubeck and R. Eldan. Multi-scale exploration of convex functions and bandit convex optimization. In *Proceedings of the Twenty Ninth Annual Conference on Computational Learning Theory*, pages 583–589, 2016.

[10] X. Chen and H. Ishwaran. Random forests for genomic data analysis. *Genomics*, 99(6):323–329, 2012.

[11] A. Chevalier, D.-A. Silva, G. J. Rocklin, D. R. Hicks, R. Vergara, P. Murapa, S. M. Bernard, L. Zhang, K.-H. Lam, G. Yao, et al. Massively parallel de novo protein design for targeted therapeutics. *Nature*, 2017.

[12] A. Conn, K. Scheinberg, and L. Vicente. *Introduction to Derivative-Free Optimization*, volume 8 of *MPS-SIAM Series on Optimization*. SIAM, 2009.

[13] M. Drela. Xfoil: An analysis and design system for low reynolds number airfoils. In *Low Reynolds number aerodynamics*, pages 1–12. Springer, 1989.

[14] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, 1993.

[15] R. El-Yaniv and Y. Wiener. Active learning via perfect selective classification. *Journal of Machine Learning Research*, 13(Feb):255–279, 2012.

[16] J. González, Z. Dai, P. Hennig, and N. Lawrence. Batch bayesian optimization via local penalization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 648–657, 2016.

[17] A. Gotovos, N. Casati, G. Hitz, and A. Krause. Active learning for level set estimation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 1344–1350, 2013.

[18] S. Hanneke. Theory of disagreement-based active learning. *Foundations and Trends in Machine Learning*, 7(2-3):131–309, 2014.

[19] C. Harwood and A. Wipat. *Microbial Synthetic Biology*, volume 40. Elsevier, 2013.

[20] M. J. Kearns and U. V. Vazirani. *An introduction to computational learning theory*. MIT press, 1994.

[21] C. G. Knight, M. Platt, W. Rowe, D. C. Wedge, F. Khan, P. J. Day, A. McShea, J. Knowles, and D. B. Kell. Array-based evolution of dna aptamers allows modelling of an explicit sequence-fitness landscape. *Nucleic Acids Research*, 37(1):e6–e6, 2008.

[22] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[23] L. Li, M. Littman, and T. Walsh. Knows what it knows: A framework for self-aware learning. In *Proceedings of the 26th International Conference on Machine Learning*, 2009.

[24] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar. Hyperband: bandit-based configuration evaluation for hyperparameter optimization. In *Proceedings of the Fourth International Conference on Learning Representations*, 2016.

[25] N. Littlestone. Redundant noisy attributes, attribute errors, and linear-threshold learning using winnow. In *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, pages 147–156, 1991.

[26] I. Loshchilov. Cma-es with restarts for solving cec 2013 benchmark problems. In *Evolutionary Computation (CEC), 2013*, pages 369–376, 2013.

[27] A. L. Marsden, M. Wang, J. E. Dennis, and P. Moin. Optimal aeroacoustic shape design using the surrogate management framework. *Optimization and Engineering*, 5(2):235–262, 2004.

[28] V. K. S. Mendelson. Bounding the smallest singular value of a random matrix without concentration. *arXiv:1312.3580 [math.PR]*, 2013.

[29] R. S. Michalski. Learnable evolution model: Evolutionary processes guided by machine learning. *Machine Learning*, 38(1):9–40, 2000.

[30] Y. Nesterov. *Introductory Lectures on Convex Optimization*. Kluwer Academic Publishers, 2004.

[31] A. S. Phelps, D. M. Naeger, J. L. Courtier, J. W. Lambert, P. A. Marcovici, J. E. Villanueva-Meyer, and J. D. MacKenzie. Pairwise comparison versus likert scale for biomedical image assessment. *American Journal of Roentgenology*, 204 (1):8–14, 2015.

[32] A. Rakhlin and K. Sridharan. On equivalence of martingale tail bounds and deterministic regret inequalities. *arXiv:1510.03925 [math.PR]*, 2015.

[33] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1889–1897, 2015.

[34] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

[35] V. Spokoiny. Parametric estimation. finite sample theory. *The Annals of Statistics*, 40(6):2877–2909, 2012.

[36] J. A. Tropp. An introduction to matrix concentration inequalities. *Foundations and Trends in Machine Learning*, 8(1-2):1–230, 2015.

[37] J. Van Hemmen and T. Ando. An inequality for trace ideals. *Communications in Mathematical Physics*, 76(2):143–148, 1980.

[38] J. Wang, Q. Gong, N. Maheshwari, M. Eisenstein, M. L. Arcila, K. S. Kosik, and H. T. Soh. Particle display: A quantitative screening method for generating high-affinity aptamers. *Angewandte Chemie International Edition*, 53(19):4796–4801, 2014.

[39] Y. Wiener and R. El-Yaniv. Agnostic selective classification. In *Advances in Neural Information Processing Systems 24*, pages 1665–1673, 2011.

[40] Y. Yu, H. Qian, and Y.-Q. Hu. Derivative-free optimization via classification. In *Proceedings of the Thirty Second National Conference on Artificial Intelligence*, pages 2286–2292, 2016.