A Zeroth-Order Block Coordinate Descent Algorithm for Huge-Scale Black-Box Optimization

HanQin Cai [1]   Yuchen Lou [2]   Daniel McKenzie [1]   Wotao Yin[3]

[1]Department of Mathematics, University of California, Los Angeles

[2]Department of Mathematics, The University of Hong Kong

[3]DAMO Academy, AliBaba US

International Conference on Machine Learning
July 22nd, 2021

Authors


HanQin Cai


Yuchen Lou


Dainel McKenzie


Wotao Yin

Zeroth-Order Optimization

Goal

Using only noisy function queries (no gradients) to find $x^\star \approx \arg\min_{x \in \mathbb{R}^d} f(x)$.

Applications:

- Simulation-based optimization.
- Adversarial attacks.
- Hyperparameter tuning.
- Reinforcement learning.
- Climate modelling.

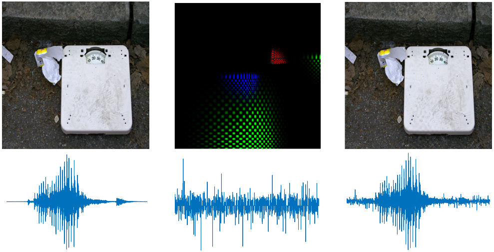

Figure: **Left:** Clean signal. **Middle:** attacking perturbation (scaled up). **Right:** Attacked signal.

In all these applications, function queries are **expensive**.

Scaling to huge dimensions

- Can estimate gradient via finite differencing: $\nabla_i f(x) \approx \dfrac{f(x+\delta e_i) - f(x)}{\delta}$.
- This requires $\mathcal{O}(d)$ queries per iteration — too expensive for large d.
- Recent work[1] uses *compressed sensing* to estimate $\nabla f(x)$:

$$y_i = \frac{f(x+\delta z_i) - f(x)}{\delta} \approx z_i^\top \nabla f(x) \text{ for } i = 1, \cdots, m.$$

$$\nabla f(x) \approx \operatorname*{arg\,min}_{v \in \mathbb{R}^d} \|Zv - y\|_2 \text{ s.t. } \|v\|_0 := \#\{i : v_i \neq 0\} \leq s. \quad (1)$$

- If[2] $\|\nabla f(x)\|_0 \leq s$ then can take $m = \mathcal{O}(s \log d)$. **Query efficient**.
- However, solving (1) is computationally expensive.

Motivating Question

Is there a query **and** computationally efficient Zeroth-Order Opt algorithm?

[1] Wang *et al*, 2018. Cai *et al*, 2020.

[2] Approximate gradient sparsity indeed observed in many applications. See Fig. 1 in (Cai *et al*, 2020).

The ZO-BCD Algorithm

Our algorithm combines:

- Compressed sensing gradient estimator.
- Block coordinate descent using randomized blocks.

Algorithm 1 Zeroth-Order Block Coordinate Descent (ZO-BCD)

$\pi \leftarrow \texttt{randperm}(d)$ ◁ Create random permutation
for $j = 1, \cdots, J$ ◁ Create J blocks
    $x^{(j)} \leftarrow [x_{\pi((j-1)\frac{d}{J}+1)}, \cdots, x_{\pi(j\frac{d}{J}+1)}]$ ◁ Assign variables to blocks
for $k = 1, \cdots, K$ ◁ Do K iterations
    $j \leftarrow \texttt{randint}(\{1, \cdots, J\})$ ◁ Randomly select a block
    for $i = 1, \cdots, m$ ◁ Query objective function
        $y_i = \frac{f(x+\delta z_i) - f(x)}{\delta}$ ◁ Approximate $z_i^\top \nabla f(x)$
    $\hat{g}^{(j)} \leftarrow \arg\min_{v:\|v\|_0 \leq s} \|Zv - y\|_2$ ◁ Approximate block gradient
    $x_{k+1} \leftarrow x_k - \alpha \hat{g}^{(j)}$ ◁ Step of BCD
return x_K ◁ Approximated minimizer

ZO-BCD Is Theoretically Sound

Main Theorem

Assume $\|\nabla f(x)\|_0 \leq s$ for all $x \in \mathbb{R}^d$. Choose $J \ll d$ random blocks. ZO-BCD returns x_K satisfying

$$f(x_K) - f_\star \leq \varepsilon$$

using $\tilde{\mathcal{O}}(s/\varepsilon)$ total queries and $\tilde{\mathcal{O}}(sd/J^2)$ FLOPS per iteration (w.h.p.).

Sketch of proof.

- Randomization ensures $\|\nabla^{(j)} f(x)\|_0 \approx s/J$.
- Compressed sensing[a] guarantees $\hat{g}^{(j)} \approx \nabla^{(j)} f(x)$.
- Apply convergence for inexact[b] block coordinate descent.

$\square$

[a] Needell *et al*, 2008.
[b] Tappenden *et al*, 2016.

A More Efficient Variant: ZO-BCD-RC

- Replace Z with a Rademacher circulant (RC) matrix[3]:

$$\mathcal{C}(z) = \begin{pmatrix} z_1 & z_2 & \cdots & z_{d/J} \\ z_{d/J} & z_1 & \cdots & z_{d/J-1} \\ \vdots & \ddots & \ddots & \vdots \\ z_2 & \cdots & z_{d/J} & z_1 \end{pmatrix}.$$

- Only need to store a vector z. **Memory efficient**.
- Matrix product calculation can be accelerated by fft & ifft. **Even faster**.

$$\mathcal{C}(z) \cdot x = \mathcal{F}\left(\mathcal{F}(z) \cdot \mathcal{F}^{-1}(x)\right)$$

for all z and x.

[3] We actually only use m randomly selected rows from $\mathcal{C}(z)$.

Experimental Results: Synthetic

Benchmarked ZO-BCD on two functions exhibiting gradient sparsity:

- Sparse quadric: $f(x) = \sum_{i=1}^{s} x_i^2$.
- Max-s-squared-sum: $f(x) = \sum_{i=1}^{s} x_{\sigma(i)}^2$ where $|x_{\sigma(1)}| \geq |x_{\sigma(2)}| \geq \cdots$.

ZO-BCD exceeds prior state-of-the-art.

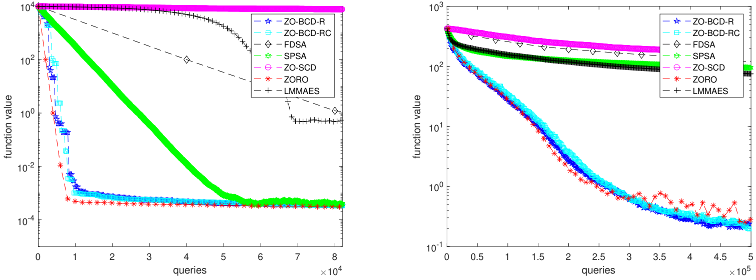

Figure: Comparing ZO-BCD against various SOTA Zeroth-Order Opt algorithms. **Left:** Sparse quadric. **Right:** Max-s-squared-sum.

Experimental Results: Adversarial Attack

- Sparse wavelet transform attack[4]:

$$x_\star = \arg\min_x f(\mathrm{IWT}(\mathrm{WT}(\tilde{x}) + x))$$

$\tilde{x}$ = clean image/audio signal. f = Carlini-Wagner loss function[5].

- **Image Attack.** model: `Inception-v3`[6]. Wavelet: 'db45'. $d \approx 675,000$.
- **Audio Attack.** model: `commandNet`[7]. Wavelet: Morse. $d \approx 1,700,000$.

Image Attack			
METHOD	ASR	ℓ_2 DIST	QUERIES
ZO-SCD	78%	57.5	2400
ZO-SGD	78%	37.9	**1590**
ZO-AdaMM	81%	28.2	1720
ZORO	90%	21.1	2950
ZO-BCD	**96**%	**13.7**	1662

Audio Attack	
METHOD	ASR
Alzantot *et al*, 2018	89.0%
Vadillo *et al*, 2019	70.4%
Li *et al*, 2020	96.8%
Xie *et al*, 2020	97.8%
ZO-BCD	**97.9**%

[4] WT & IWT stand for some fixed wavelet transform & inverse wavelet transform, respectively.

[5] Carlini & Wagner, 2016. Chen *et al*, 2017.

[6] Szegedy *et al*, 2016. Trained on `ImageNet` (Deng *et al* 2009).

[7] `Matlab model trained on SpeechCommand dataset` (Warden *et al*, 2018).