ZOERTH-ORDER OPTIMIZATION: COSINE MEASURE IN HIGH DIMENSIONAL SPACE

ZHEKAI LIU

Abstract. Positive spanning is a crucial definition in derivative-free optimization (DFO) problems. A popular way to measure the quality of positive spanning is using the cosine measure. However, in reality, how to measure the quality of the different positive spannings in high-dimensional space becomes a challenging nonconvex optimization problem. In this paper, we introduce a novelty **z**ero-**o**rder stochastic algorithm to tackle this **c**osine **m**easure **a**lgorithm problem (**ZO-CMA**). **ZO-CMA shows a significant improvement compared with the Central simplex gradient algorithm in high-dimensional space.** The code links: Zeroth-Order-Cosine-measure-algorithm

1. Introduction

Positive spanning[8] is an essential definition in derivative-free optimization (DFO) [3], it's usually used in GPS [1] [13][10] [16], and MADS [2][6]. The Positive spanning set is always deemed to be potential poll directions for the next iteration. So a common question is **how to measure the positive spanning.**

The most famous way is using the cosine measure [9] to compare different positive spannings:

$$\min_{u\in\mathbb{R}^n,\|u\|=1} \max_{d\in\mathbb{D}} \frac{u^\top d}{\|d\|}.$$

The cosine measure is a Min-Max problem in reality, which means that we could not use a traditional gradient-based method to optimize it. So, the problem becomes a DFO problem [7].

However, in reality, how to measure the quality of the different positive spannings in high-dimensional space becomes a challenging non-convex optimization problem [4].

In recent years, machine learning has been growing rapidly; adversarial attacks [11][5], and neural network defenses are becoming a popular topic [15]. In computer vision's perspective, a vector u is just like an image with a noise. Our goal is to let the pre-trained neural network or convolutional network model have a wrong classification of this image. The original image without noise is exactly like the origin point in the cosine measure definition, the noise can be deemed to be the u in the cosine definition analogously.

In this paper, we introduce a novelty **z**ero-**o**rder stochastic algorithm to tackle this **c**osine **m**easure **a**lgorithm problem (**ZO-CMA**). We create two methods, namely the ZO-CMA rectangular coordinates vision and the ZO-CMA spheric coordinates vision.

- ZO-CMA rectangular coordinates vision: optimize the rectangular coordinates position and project it on the unit sphere surface back for each iteration.
- ZO-CMA spheric coordinates vision: optimize the spheric coordinates position directly, and because we only optimize the angle of the vector u, it is always on the unit sphere surface.

Date: Feb. 2025.

2. Main Problem

Let $\mathbb{D}$ be a non-empty subset of non-zero vectors in $\mathbb{R}^n$. The **cosine measure**[9] of $\mathbb{D}$ is defined as:

$$\mathrm{cm}(\mathbb{D}) = \min_{u\in\mathbb{R}^n,\|u\|=1} \max_{d\in\mathbb{D}} \frac{u^\top d}{\|d\|}.$$

So, giving a positive spanning set $\mathbb{D}$, our problem becomes:

$$\begin{aligned} \min_{u,\alpha} \quad & \alpha \\ \text{s.t.} \quad & \frac{u^\top d_i}{\|d_i\|} \le \alpha, \quad \forall d_i \in \mathbb{D}, \\ & \|u\| = 1. \end{aligned}$$

Here are several methods to solve that in a low-dimensional space.

- Solve this by hand (Solve this non-linear programming problem and also need to enumerate different situations).
- Solve by some package in Python (e.g. CVXPY).

The advantage of the first method is that we could avoid the local minimum in lower dimensions, so it is a promising method to tackle a low-dimensional space problem. For instance, your question:

$$\mathbb{D} = \{e_1, e_2, e_1 + e_2 + e_3, e_4, -e_1 - e_2, -e_3 - e_4\},$$

however, because we need to consider and enumerate all situations during this non-linear programming problem, things will get harder in high-dimensional space.

Furthermore, because of the nonconvex property of the objective function in high-dimensional space, CVXPY(usually used to solve convex problems) isn't a good choice.

3. High-dimensional Space Situation

3.1. **Methodology.** Give this optimization problem below:

$$\min_{u\in\mathbb{R}^n,\|u\|=1} \max_{d\in\mathbb{D}} \frac{u^\top d}{\|d\|},$$

where $\mathbb{D}$ is a positive spanning, for abbreviation, we denote our object function as $f(u)$. This optimization problem is a Min-Max problem, which means that we could not use the traditional optimization method (white-box gradient-based optimization) to solve it.

Building on these insights, several popular approaches have emerged [12],[3],[5]. In this paper, we used the Zoerth-order gradient estimation in the [12]:

$$\hat{g}_u = \frac{f(u + \mu\mathbf{u}) - f(u - \mu\mathbf{u})}{2\mu}\mathbf{u},$$

where $\mu > 0$ is the smoothing parameter, which is a small positive constant, and $\mathbf{u} \in \mathbb{R}^d$ is distributed in $\mathcal{N}(0, \mathbf{I}_d)$.

Actually, this is a really common problem in machine learning, and the problem is called adversarial attack. In computer vision's perspective, a vector u is just like an image with a noise. Our goal is to let the pre-trained neural network or convolutional network model have a wrong classification of this image [5][11].

Now, let us introduce the ZO-CMA rectangular coordinates vision (Figure 1, Algorithm 1), in the algorithm, we optimize the rectangular position in the whole space and project it on the unit sphere surface back at the end of each iteration.

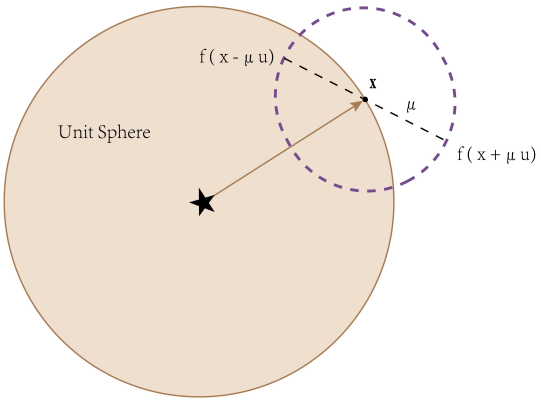


FIGURE 1. ZO-CMA Rectangular Coordinates Vision

In this picture, we add forward and backward noises at the vector x rectangular coordinates position (we use x instead of u in the figure 1 for abbreviation), and the noise radius is μ.

Algorithm 1 ZO-CMA: Rectangular Coordinates Vision

Input: Positive spanning set $\mathbb{D}$, Q, I, T are two constants, μ is also a constant in Zoerth-order gradient, learning rate η, smooth parameter μ, d represents dimensions, and the object function f.

Initialization:

1: Randomly sample $r_1, \cdots, r_Q$ from a Uniform distribution on unit shpere $\mathcal{S}^d(0,1)$.

2: $u^0 = \arg\min_{\{r_1,\cdots,r_Q\}} f(r_i);\ i = 1, \cdots, Q.$

Optimization:

3: **for** t in 0 to $T-1$ **do**

4: **for** i in 1 to I **do**

5: $g_i^t = \dfrac{f(u^t + \mu \mathbf{u}_i) - f(u^t - \mu \mathbf{u}_i)}{2\mu} \mathbf{u}_i$, where $\mathbf{u}_i \sim \mathcal{S}^d(0,1)$.

6: $g_u^t = \dfrac{1}{I} \sum_{i=1}^{I} g_i^t.$

7: **end for**

8: $\hat{u}^{t+1} = u^t - \eta g_u^t.$

9: $u^{t+1} = \dfrac{\hat{u}^{t+1}}{\|u^{t+1}\|}$

10: **end for**

11: **return** u^T

Explanation: This version of the algorithm optimizes the coordinate position of vector u directly. We normalize the vector in step 9 for each iteration to ensure that the vector is always a unit vector. The reason we do not use the normal distribution in step 5 is that we aim to reduce the oscillation.

Then, let us introduce the ZO-CMA spheric coordinates vision (Figure 2, Algorithm 2), we optimize spherical coordinates directly, so the vector u is always on the unit sphere surface so we don't need to normalize it. However, in order to make sure the polar position is illegal ($\theta \in (-\pi, \pi]^d$), we should check the spherical position value during the optimization process, please see the details in the codes.

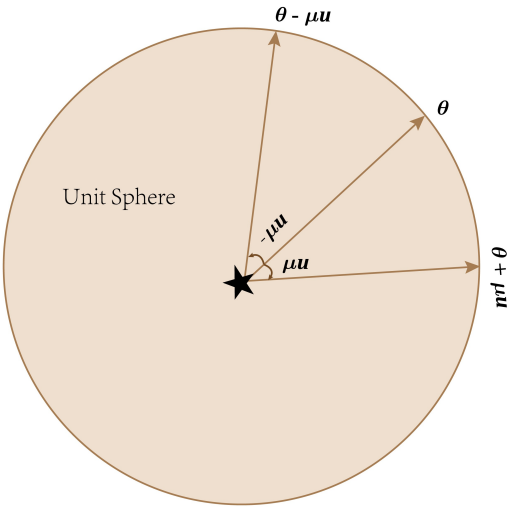


FIGURE 2. ZO-CMA Spheric Coordinates Vison

Algorithm 2 ZO-CMA: Spherical Coordinates Vision

Input: Positive spanning set $\mathbb{D}$, Q, I, T are two constants, μ is also a constant in Zoerth-order gradient, learning rate η, smooth parameter μ, d represents dimensions and the object function f.

Initialization:

Randomly sample $r_1, \cdots, r_Q$ from a Uniform distribution between $(-\pi, \pi]^d$.

$\theta^0 = \arg\min_{\{r_1,\cdots,r_Q\}} f(cos(r_i));\ i = 1, \cdots, Q.$

Optimization:

for t in 0 to $T-1$ **do**

  for i in 1 to I **do**

    $g_i^t = \dfrac{f(cos(\theta^t + \mu \mathbf{u}_i)) - f(cos(\theta^t - \mu \mathbf{u}_i))}{2\mu} \mathbf{u}_i$, where $\mathbf{u}_i \sim \mathcal{S}(0, 1)$.

    $g_\theta^t = \dfrac{1}{I} \sum_{i=1}^{I} g_i^t.$

  end for

  $\theta^{t+1} = \theta^t - \eta g_\theta^t.$

end for

return θ^T

4. EXPERIMENTS

Clearification: In the section, **some examples do not use the positive spanning $\mathbb{D}$, in order to make sure it's easy to calculate**. This action won't hurt the performance of the ZO-CMA on the positive spanning.

Problem 1: $\mathbb{D} = \{e_1, e_2\}$, it's easy to calculate that the cosine measure of this set is:

$$f(u) = cos(135^\circ) \approx= \textbf{-0.707}$$

By using **ZO-CMA** we have following result after :

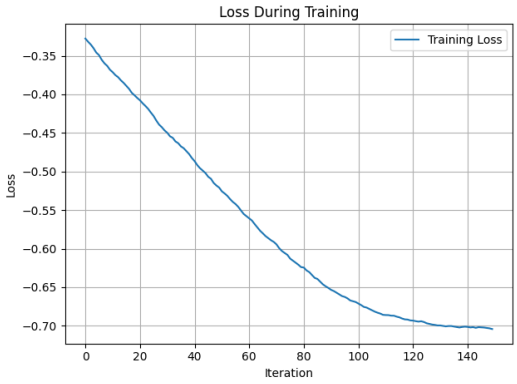


FIGURE 3.
ZO-CMA Rectangular Vision: $I = 10, Q = 2, T = 150, \mu = 0.1, \eta = 0.01$

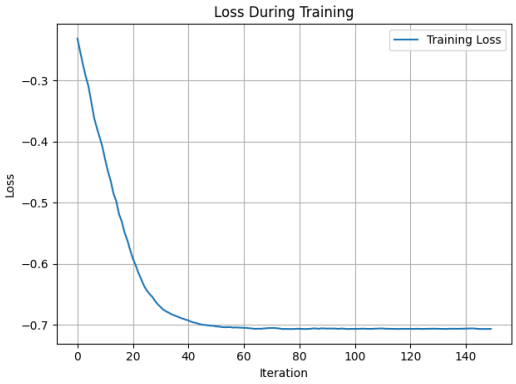


FIGURE 4.
ZO-CMA Spheric Vision: $I = 10, Q = 2, T = 150, \mu = 0.1, \eta = 0.01$

In this experiment 3, we only use 150 iterations with $3,002$ function evaluations($3,000+2$, 2 for initialization). After 150 iterations, $f(u) \approx$ **-0.704**.

In experiment 4, we only use 150 iterations with $3,002$ function evaluations($3,000+2$, 2 for initialization). After 150 iterations, $f(u) \approx$ **-0.707**.

Problem 2: $\mathbb{D} = \{e_1, e_2, e_1+e_2+e_3, e_4, -e_1-e_2, -e_3-e_4\}$. Firstly, we random sample vector on the unit sphere for $10,000$ times, and the final result is $f(u) \approx$ **0.1745**.(I need to apologize that I didn't calculate it by hand.)

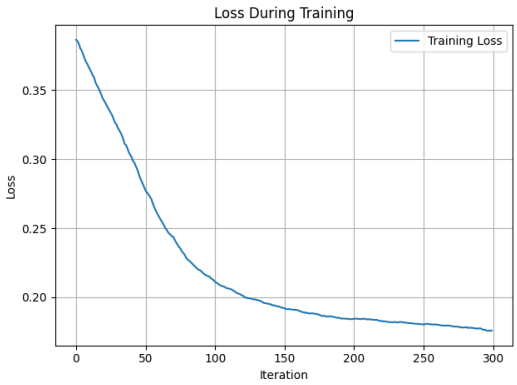


FIGURE 5.
ZO-CMA Rectangular Vision: $I = 10, Q = 30, T = 300, \mu = 0.1, \eta = 0.01$

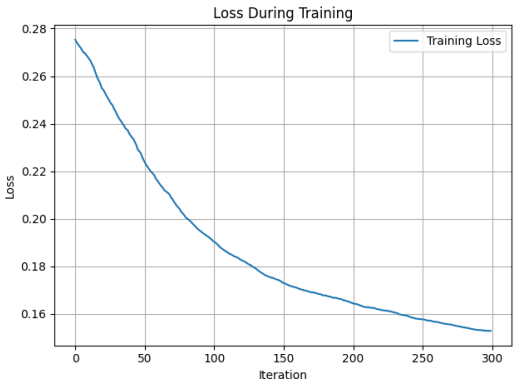


FIGURE 6.
ZO-CMA Spheric Vision: $I = 10, Q = 30, T = 300, \mu = 0.1, \eta = 0.01$

In this experiment 5, we use 300 iterations with $6,030$ function evaluations($6,000+30$, $6,030$ for initialization). After 300 iterations, $f(u) \approx$ **0.1759**.

In spheric experiment 6, we use 300 iterations with $6,030$ function evaluations($6,000+30$, $6,030$ for initialization). After 300 iterations, $f(u) \approx$ **0.1729**.

Problem 3: $\mathbb{D} = \{e_i; i = 1, 2, \cdots, 8\}$. Firstly, we random sample vector on the unit sphere for **1,000,000** times, and the final result is $f(u) \approx$ **-0.3084**. Actually, this is

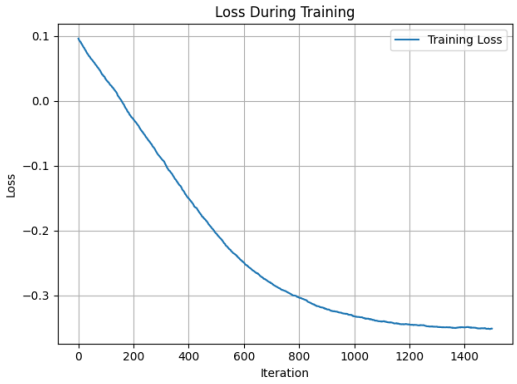


FIGURE 7. ZO-CMA Rectangular: $I = 10, Q = 30, T = 1,500, \mu = 0.1, \eta = 0.01$

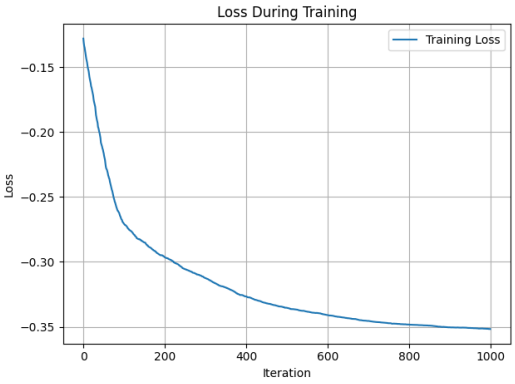


FIGURE 8. ZO-CMA Spheric: $I = 10, Q = 30, T = 1,000, \mu = 0.1, \eta = 0.01$

NOT the correct anwer, because there are enormous situations. So, here we use another method:

To minimize the maximum component of u, an optimal choice is to distribute the components of u evenly with equal negative values:

$$u = \left(-\frac{1}{\sqrt{8}}, -\frac{1}{\sqrt{8}}, \ldots, -\frac{1}{\sqrt{8}}\right).$$

Therefore, the cosine measure of $\mathbb{D}$ is:

$$\mathrm{cm}(\mathbb{D}) = -\frac{1}{\sqrt{8}} \approx \textbf{-0.3536}.$$

In our experiment 7, we use $1,500$ iterations with $30,030$ function evaluations($30,000 +$ 30, 30 for initialization). After $1,500$ iterations, $f(u) \approx$ **-0.3511**.

In spheric coordinates vision experiments 8, we use $1,000$ iterations with $21,000$ function evaluations($20,000 + 1,000$, $1,000$ for initialization). After $1,000$ iterations, $f(u) \approx$ **-0.3517**.

In these experiments, we found that initialization is a crucial part of the algorithm, a relatively large number of initialization could reduce the possibility of getting stuck in the local minimum.

Furthermore, I rarely tuned the hyperparameter in these experiments, actually, the function evaluation times could dramatically decrease by raising the learning rate η, however raising the algorithm's variance instead [14].

5. Comparision between ZO-CMA and Central simplex gradient

5.1. **Centered simplex gradient.** According to the Definition 9.12 (Centered Simplex Gradient in [3]). Let $f : \mathbb{R}^n \to \mathbb{R}$ and let $\mathbb{Y}^+ = \{y^0, y^1, \ldots, y^n\} \subset \mathbb{R}^n$ be poised for linear interpolation. Define $d^i = y^i - y^0$ for $i = 1, 2, \ldots, n$ and $\mathbb{Y}^- = \{y^0, y^0 - d^1, y^0 - d^2, \ldots, y^0 - d^n\}$, and set $\mathbb{Y} = \mathbb{Y}^+ \cup \mathbb{Y}^-$. The Centred Simplex Gradient of f over $\mathbb{Y}$, denoted $\nabla_{CS} f(\mathbb{Y})$, is given by

$$\nabla_{CS} f(\mathbb{Y}) := \frac{1}{2}\left(\nabla_S f(\mathbb{Y}^-) + \nabla_S f(\mathbb{Y}^+)\right).$$

Now, let's explain why this is a high-dimensional centered finite difference gradient. After observation, we could find:

$$\mathbb{Y}^+ = \{y^0, y^1, \dots, y^n\} = \{y^0 + (y^i - y^0); i = 0, 1, \cdots, n\},$$

and

$$\mathbb{Y}^- = \{2y^0 - y^i; i = 0, 1, \cdots, n\} = \{y^0 - (y^i - y^0); i = 0, 1, \cdots, n\}.$$

So, we could already find that this is just like the finite difference gradient but $h \in \{y^i - y^0; i = 1, \cdots, n\}$ belongs in various directions. Then,

$$2(y^i - y^0)^\top \nabla_{CS} f(y^0) = f(y^i) - f(y^0 - d^i).$$

Theorem 1. *(Proposition 9.11 in [3]) Let $f : \mathbb{R}^n \to \mathbb{R}$ and let $\mathbb{Y}^+ = \{y^0, y^1, \dots, y^n\} \subset \mathbb{R}^n$ be poised for linear interpolation. Define $d^i := y^i - y^0$ for $i = 1, 2, \dots, n$,*

$$L := \begin{bmatrix} y^1 - y^0 & y^2 - y^0 & \dots & y^n - y^0 \end{bmatrix},$$

and

$$\delta_{CS}^{f(\mathbb{Y})} := \frac{1}{2} \begin{bmatrix} f(y^1) - f(y^0 - d^1) \\ f(y^2) - f(y^0 - d^2) \\ \vdots \\ f(y^n) - f(y^0 - d^n) \end{bmatrix}.$$

Then the centered simplex gradient $\beta = \nabla_{CS} f(\mathbb{Y})$ is the solution of the linear system

$$L^\top \beta = \delta_{CS}^{f(\mathbb{Y})}.$$

In this section, we compared ZO-CMA with traditional central simplex gradient descent (CSGD). The pseudo-code is below:

Algorithm 3 CSGD

Input: Positive spanning set $\mathbb{D}$, learning rate η, d represents dimensions, and the object function f.

Initialization:

Randomly sample $r_1, \cdots, r_Q$ from a Uniform distribution on unit shpere $\mathcal{S}^d(0, 1)$.

$u^0 = \arg\min_{\{r_1, \cdots, r_Q\}} f(r_i);\ i = 1, \cdots, Q.$

Optimization:

for t in 0 to $T - 1$ **do**

  Generare $\mathbb{Y}_t^+ = \{u^t, u^t + \mu \mathbf{u}_1, \cdots, u^t + \mu \mathbf{u}_d\}$ where $\mathbf{u}_i \sim \mathcal{S}^d(0, 1)$.

  Solve $\nabla_{CS} f(u^t)$ from $L^\top \nabla_{CS} f(u^t) = \delta_{CS}^{f(\mathbb{Y}_t^+)}$

  $\hat{u}^{t+1} = u^t - \eta \nabla_{CS} f(u^t)$.

  $u^{t+1} = \dfrac{\hat{u}^{t+1}}{\|u^{t+1}\|}$

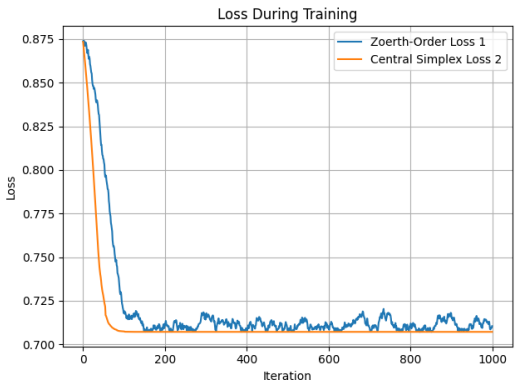
end for

return u^T

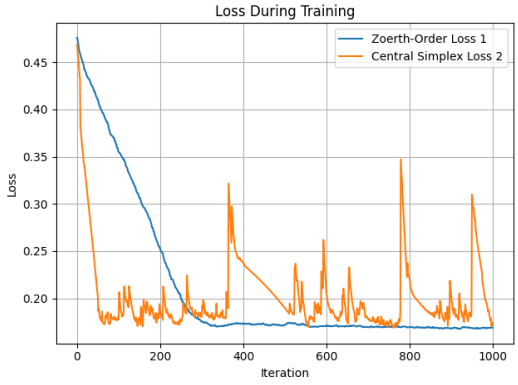
To ensure a fair comparison, we need to use the same function evaluations and sampling positions as closely as possible. Using CSGD 3, in $\mathbb{R}^d$ needs $2d + 1$ function evaluations, so in ZO-CMA, we also sample $\mathbf{u}_i$ d times from the unit sphere $\mathcal{S}(0, 1)$. Aiming to make a fair comparison, we use the **same sampling points** in both algorithms, which means that we use the same $\{\mathbf{u}_i; \mathbf{u}_i \sim \mathcal{N}(0, \mathcal{I}_d), i = 1, \cdots, d\}$ in both ZO-CMA and CSGD algorithms for each iteration.

In the following experiments, we also used the same vector sets $\mathbb{D}$ compared with experiment section 4. We didn't use positive spanning because these examples are easier to check whether they are correct or not.

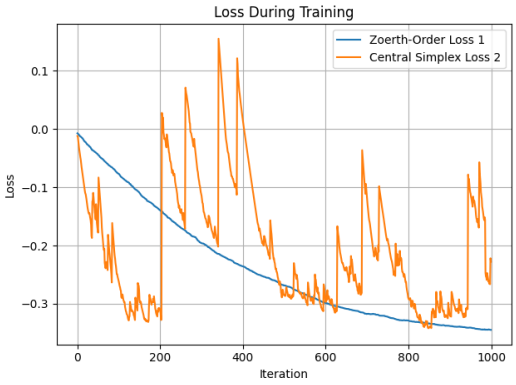
The experiment results are 9a 9b 9c; we also used the same hyperparameters to ensure fairness.



(A) $\mathbb{D}_1 = \{e_1, e_2\}$



(B) $\mathbb{D}_2 = \{e_1, e_2, \sum_{i=1}^{3} e_i, e_4, -e_1 - e_2, e_3 - e_4\}$



(C) $\mathbb{D} = \{e_1, e_2, \cdots, e_8\}$

FIGURE 9. Comparison of different problems

In these experiments, we found that the Central simplex gradient performs really well in low-dimensional space, but becomes worse in higher dimensions which suffers a high variance and low accuracy during the experiments. In contrast, ZO-CMA performs more steadily than CSGD in high dimensions. After observing the experiments, I suppose the reason CSGD is not stable is that its gradient may **explode** during the iterations. So we normalized the Central simplex gradient below:

$$\text{Solve } \hat{\nabla}_{CS} f(u^t) \text{ from } L^\top = \delta_{CS}^{f(\mathbb{Y}^+)},$$

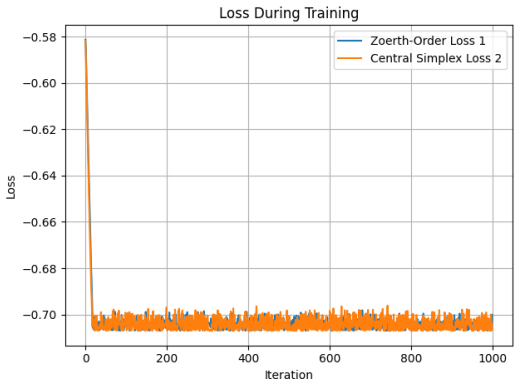
$$\nabla_{CS} f(u^t) = \frac{\hat{\nabla} f(u^t)}{\|\hat{\nabla} f(u^t)\|}.$$

For fairness, we also normalize the Zoerth-Order gradient:

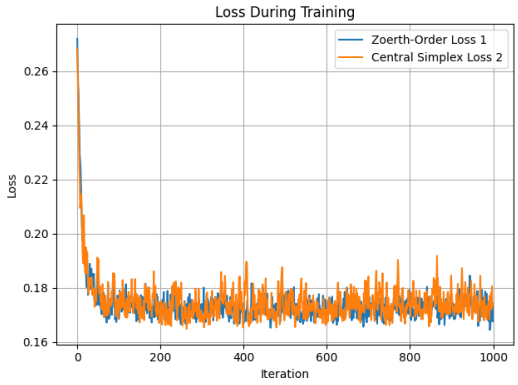
$$\hat{g}_u^t = \frac{1}{d} \sum_{i=1}^{d} g_i^t,$$

$$g_u^t = \frac{\hat{g}_u^t}{\|\hat{g}_u^t\|}.$$

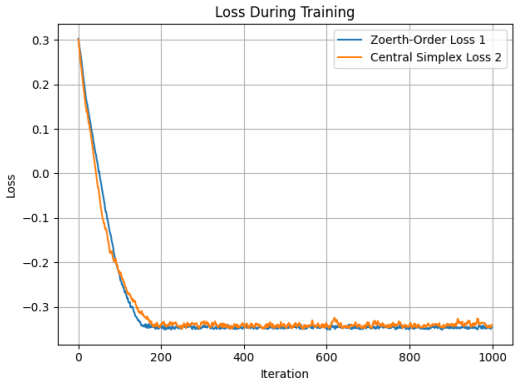
After normalizing the ZO and CSG, we re-do the above experiments, the results are below:



(A) $\mathbb{D}_1 = \{e_1, e_2\}$



(B) $\mathbb{D}_2 = \{e_1, e_2, \sum_{i=1}^{3} e_i, e_4, -e_1 - e_2, e_3 - e_4\}$



(C) $\mathbb{D} = \{e_1, e_2, \cdots, e_8\}$

FIGURE 10. Comparison of different problems

In these experiments 10a 10b 10c, we set the same default hyperparameters and sampling points for both the Zoerth-Order gradient and CSGD. According to Figure 9, we normalize both Zoerth-Order and CSG before iteration for fairness. So, **this is a fair comparison**.

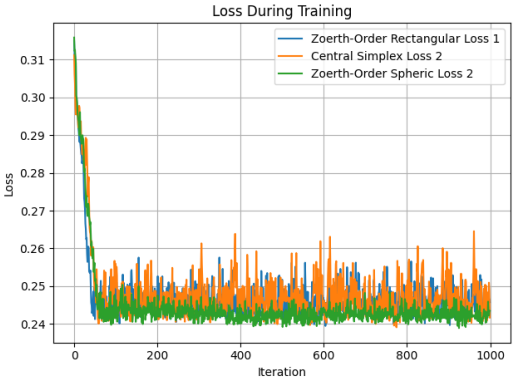
These experiments showed the **ZO-CMA Rectangular, Spheric Vision, and Central simplex gradient have a similar performance in the low dimensional space.**

6. FINAL COMPARISON

In the end, we compared the ZO-CMA Rectangular, Spheric Vision, and Central simplex gradient algorithm (Figure 11). We used the same default hyperparameters and sampling points and normalized all three gradients (the ZO Rectangular, Spheric, and Central simplex gradient) to ensure fairness.

The reason the experiment for $\mathbb{D}_2$ suffers a high variance 11b is that the learning rate is relatively large for this experiment. Solving this problem is easy, setting the learning rate $\eta = 0.001$ is fine. However, in order to show our algorithms are low-sensitivity regarding the hyperparameters, we didn't tune any parameter during the experiments.

(A) $\mathbb{D}_1 = \{e_1, e_2\}$



(B) $\mathbb{D}_2 = \{e_1, e_2, \sum_{i=1}^{3} e_i, e_4, -e_1 - e_2, e_3 - e_4\}$

(C) $\mathbb{D} = \{e_1, e_2, \cdots, e_8\}$

FIGURE 11. Comparison of different problems

6.1. **Spotlight results.** We tried to compare ZO-CMA rectangular, spheric vision, and the Central simplex gradient in **high-dimensional space**, namely:

$$\mathbb{D}^{100} = \{e_i, i = 1, 2, \cdots, 100\}.$$
$$\mathbb{D}^{225} = \{e_i, i = 1, 2, \cdots, 225\}.$$

The true answers are:

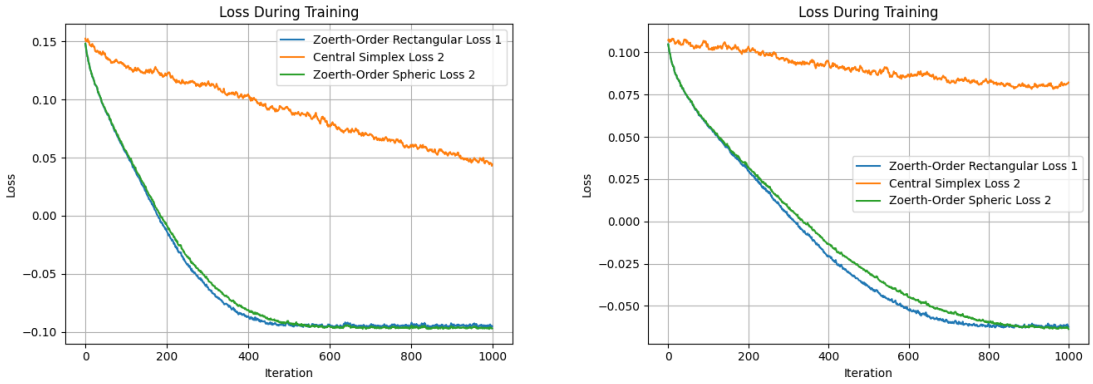
$$cm(\mathbb{D}^{100}) = -\frac{1}{\sqrt{100}} = \textbf{-0.1},$$
$$cm(\mathbb{D}^{225}) = -\frac{1}{\sqrt{225}} = -\frac{1}{15} = \textbf{-0.067}.$$

We compared three algorithms with default hyper-parameters, and the results are below Figures 12a 12b.

The left figure experiment costs $10min$ CPU time 12a, and the right experiment spends $1hour$ CPU time 12b; the ZO-CMA algorithms demonstrate a significant improvement compared with the Central simplex gradient algorithm.

7. CONCLUSION

In this work, we introduce a novelty **a**lgorithm framework by using the **z**eroth-**o**rder gradient to estimate the **c**osine **m**easure (ZO-CMA). We used both experiments and theory to prove the effectiveness of our method. Furthermore, **we compared ZO-CMA with the Central simplex gradient and showed their significantly improved performance in high dimensional space.**



(A) $\mathbb{D}^{100} = \{e_1, \cdots, e_{100}\}$ (B) $\mathbb{D}^{225} = \{e_1, \cdots, e_{225}\}$

FIGURE 12. High-dimensional space comparison

8. LIMITATION

The **ZO-CMA** could only find the local minimum solution in reality. During the experiments, we need to reimplement the codes several times to achieve better performance. Furthermore, our algorithm is almost a monotonic decrease but may increase at some spots.

REFERENCES

[1] Mark A. Abramson, Charles Audet, and J. E. Dennis. "Generalized pattern searches with derivative information". In: *Math. Program.* 100.1 (May 2004), pp. 3–25. ISSN: 0025-5610.

[2] C. Audet and J. E. Dennis Jr. "Mesh adaptive direct search algorithms for constrained optimization". In: *SIAM J. Optim.* 17.1 (2006), pp. 188–217. DOI: 10.1137/040603371.

[3] Charles Audet and Warren Hare. *Derivative-Free and Blackbox Optimization*. Cham, Switzerland: Springer, 2017. DOI: 10.1007/978-3-319-68913-5. URL: https://dx.doi.org/10.1007/978-3-319-68913-5.

[4] Charles Audet, Warren Hare, and Gabriel Jarry-Bolduc. "The cosine measure relative to a subspace". In: *arXiv preprint arXiv:2401.09609* (2024).

[5] Minhao Cheng et al. "Sign-opt: A query-efficient hard-label adversarial attack". In: *arXiv preprint arXiv:1909.10773* (2019).

[6] F. H. Clarke. *Optimization and Nonsmooth Analysis*. Reissued in 1990 by SIAM Publications, Philadelphia, as vol. 5 in the series Classics in Applied Mathematics. New York: Wiley, 1983.

[7] A. R. Conn, K. Scheinberg, and L. N. Vicente. *Introduction to Derivative-Free Optimization*. MOS-SIAM Series on Optimization. Philadelphia: SIAM, 2009.

[8] Chandler Davis. "THEORY OF POSITIVE LINEAR DEPENDENCE." In: *American Journal of Mathematics* 76 (1954), p. 733. URL: https://api.semanticscholar.org/CorpusID:125070227.

[9] Tamara G. Kolda, Robert Michael Lewis, and Virginia Torczon. "Optimization by Direct Search: New Perspectives on Some Classical and Modern Methods". In: *SIAM Review* 45.3 (2003), pp. 385–482. DOI: 10.1137/S003614450242889.

[10] R.M. Lewis and V. Torczon. *Rank ordering and positive bases in pattern search algorithms*. Tech. rep. 96-71. Mail Stop 132C, NASA Langley Research Center, Hampton, Virginia 23681-2199: Institute for Computer Applications in Science and Engineering, 1996.

[11] Linyang Li, Demin Song, and Xipeng Qiu. "Text Adversarial Purification as Defense against Adversarial Attacks". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Toronto, Canada: Association for Computational Linguistics, 2023, pp. 338–350. DOI: 10.18653/v1/2023.acl-long.20.

[12] Yurii Nesterov and Vladimir Spokoiny. "Random gradient-free minimization of convex functions". In: *Foundations of Computational Mathematics* 17.2 (2017), pp. 527–566.

[13] C. J. Price and I. D. Coope. "Frames and Grids in Unconstrained and Linearly Constrained Optimization: A Nonsmooth Approach". In: *SIAM Journal on Optimization* 14.2 (2003), pp. 415–438. DOI: 10.1137/S1052623402407084. eprint: https://doi.org/10.1137/S1052623402407084. URL: https://doi.org/10.1137/S1052623402407084.

[14] D.H. Wolpert and W.G. Macready. "No free lunch theorems for optimization". In: *IEEE Transactions on Evolutionary Computation* 1.1 (1997), pp. 67–82. DOI: 10.1109/4235.585893.

[15] S. Yang and C. Xu. "One size does not fit all: data-adaptive adversarial training". In: *European Conference on Computer Vision* (2022), pp. 70–85.

[16] W.-C. Yu. "Positive basis and a class of direct search techniques". In: *Sci. Sinica* 1 (1979), pp. 53–67.

Appendix A. Appendix

Appendix B. Convergence Analysis

Theorem 2. *Under the **local** gradient's Lipschitz assumption $f \in l^{1+}, x \in B_\Delta$, let learning rate $\eta = \frac{1}{L^2}$, when $T = \mathcal{O}(\frac{2L^2}{\varepsilon^2})$, $\mu = \frac{\varepsilon}{\sqrt{DLd}}, I = \mathcal{O}(\frac{\sigma^2}{2L^2\varepsilon^2}), \mu = \mathcal{O}(\sqrt{\frac{\varepsilon}{Ld}})$, where L represents Lipchitz constant and d represents the dimensions, for $\varepsilon > 0$, we have:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(u^t)\|^2 \leq \mathcal{O}(\varepsilon^2)$$

The vector u will converge to the critical position (this point will change $d \in \mathbb{D}$ to measure cosine measure), so the local gradient Lipschitz is a reasonable assumption for this problem.

Proof. **Assumption 1** [L-Smoothness] For convenience, denote:

$$f(u) := \min_{u\in\mathbb{R}^n, \|u\|=1} \max_{\mathbf{d}\in\mathbb{D}} \frac{u^\top \mathbf{d}}{\|\mathbf{d}\|}.$$

Because the object function is $\min\max cos(\cdot,\cdot)$, it is reasonable to assume that the object function f is a **local** Lipchitz function.

$$\|\nabla f(u) - \nabla f(u')\| \leq L\|u - u'\|,\ u, u' \in B_\Delta. \tag{1}$$

Where, $u \in \mathbb{R}^d$, L represents the Lipchitz constant, and B_Δ is a trust region. This condition is equivalent to the following inequality:

$$|f(u') - f(u) - \langle \nabla f(u), u' - u\rangle| \leq \frac{L^2}{2}\|u' - u\|^2, \quad u, u' \in B_\Delta. \tag{2}$$

Lemma 1. *(Theorem 1 and Lemma 3 in [12]) If $f \in C^{1,1}(\mathbb{R}^d)$ with constant L, and $x \in \mathbb{R}^d$ then for the bias with respect to the zeroth-order function:*

$$|f_\mu(x) - f(x)| \leq \frac{\mu^2}{2}Ld. \tag{3}$$

For the bias with respect to the zeroth-order gradient:

$$\|\nabla f_\mu(x) - \nabla f(x)\| \leq \frac{\mu}{2}L(d+3)^{3/2}, \tag{4}$$

where:

$$f_\mu(x) = \frac{1}{\kappa}\int_{\mathbb{R}^d} f(x + \mu u)e^{-\frac{1}{2}\|u\|^2}\mathrm{d}u;$$

$$\nabla f_\mu(x) = \frac{1}{\kappa}\int_{\mathbb{R}^d} \frac{f(x+\mu u) - f(x-\mu u)}{2\mu}e^{-\frac{1}{2}\|u\|^2}u\mathrm{d}u.$$

According to Eq.(12) in [12], we can now that f_μ is Lipschitz smooth:

$$\|\nabla f_\mu(u) - \nabla f_\mu(u')\| \leq L\|u - u'\|, \tag{5}$$

where $\nabla f_\mu(u) = \frac{1}{\kappa}\int_{\mathbb{R}^d} \frac{f(x+\mu u) - f(x-\mu u)}{2\mu}e^{-\frac{1}{2}\|\mathbf{u}\|^2}u\mathrm{d}u$, and $\kappa = \int_{\mathbb{R}^d} e^{-\frac{1}{2}\|u\|^2}\mathrm{d}u$.

Convergence analysis:
According to the local gradient smoothness property:

$$|f(u') - f(u) - \langle \nabla f(u), u' - u\rangle| \leq \frac{L^2}{2}\|u' - u\|^2, \quad u, u' \in B_\Delta.$$

Firstly, because of the smoothness of the $f(\cdot)$ with respect to u from the inequality (2) and the bias from the zeroth-order gradient from the inequality (6):

$$\begin{aligned} f(u^{t+1}) - f(u^t) \leq & \left(f_\mu(u^{t+1}) + \frac{\mu^2}{2} Ld \right) - \left(f_\mu(u^t) - \frac{\mu^2}{2} Ld \right) \\ = & \ f_\mu(u^{t+1}) - f_\mu(u^t) + \mu^2 Ld \\ \leq & \langle \nabla f_\mu(u^t), u^{t+1} - u^t \rangle + \frac{L^2}{2} \|u^{t+1} - u^t\|^2 + \mu^2 Ld. \end{aligned}$$

Let $\eta = \frac{1}{L^2}$, denote the variance $\frac{\sigma^2}{I} = \mathbb{E}_u \|\hat{g}_\mu(u) - \nabla f_\mu(u^t)\|^2$,

$$\begin{aligned} \mathbb{E} f(u^{t+1}) - f(u^t) \leq & -\frac{1}{L^2} \mathbb{E} \|\hat{g}_\mu(u^t)\|^2 + \frac{1}{2L^2} (\mathbb{E} \|\hat{g}_\mu(u^t) - \nabla f_\mu(u^t)\|^2 + \mathbb{E} \|\nabla f_\mu(u^t)\|^2) + \mu^2 Ld \\ \leq & -\frac{1}{2L^2} \|\nabla f(u^t)\|^2 + \frac{\sigma^2}{2L^2 I} + 2\mu^4 L^2 d^2 \end{aligned}$$

At this moment denote $C = 2\mu^4 L^2 (d)^2$:

$$\begin{aligned} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E} \|\nabla f(u^t)\|^2 \leq & 2L^2 \frac{1}{T} \sum_{t=0}^{T-1} (f(u^0) - f^*) + C \\ = & \mathcal{O}(\varepsilon^2) \end{aligned}$$

When $T = \mathcal{O}(\frac{2L^2}{\varepsilon^2})$, $\mu = \frac{\varepsilon}{\sqrt{DLd}}, I = \mathcal{O}(\frac{\sigma^2}{2L^2\varepsilon^2}), \mu = \mathcal{O}(\sqrt{\frac{\varepsilon}{Ld}})$, and f^* represents minimum valueof the object function $f(u)$. □