A CONSENSUS-BASED GLOBAL OPTIMIZATION METHOD WITH ADAPTIVE MOMENTUM ESTIMATION

JINGRUN CHEN, SHI JIN, AND LIYAO LYU

ABSTRACT. Objective functions in large-scale machine-learning and artificial intelligence applications often live in high dimensions with strong non-convexity and massive local minima. First-order methods, such as the stochastic gradient method and Adam [12], are often used to find global minima. Recently, the consensus-based optimization (CBO) method has been introduced as one of the gradient-free optimization methods and its convergence is proven with dimension-dependent parameters, which may suffer from the curse of dimensionality. By replacing the isotropic geometric Brownian motion with the component-wise one, the latest improvement of the CBO method [6] is guaranteed to converge to the global minimizer with dimension-independent parameters [9], although the initial data need to be well-chosen. In this paper, based on the CBO method and Adam, we propose a consensus-based global optimization method with adaptive momentum estimation (Adam-CBO). Advantages of the Adam-CBO method include: (1) capable of finding global minima of non-convex objective functions with high success rates and low costs; (2) can handle non-differentiable activation functions and thus approximate low-regularity functions with better accuracy. The former is verified by approximating the 1000 dimensional Rastrigin function with 100% success rate at a cost only growing linearly with respect to the dimensionality. The latter is confirmed by solving a machine learning task for partial differential equations with low-regularity solutions where the Adam-CBO method provides better results than the state-of-the-art method Adam. A linear stability analysis is provided to understand the asymptotic behavior of the Adam-CBO method.

1. INTRODUCTION

The goal of this work is developing consensus-based global optimization methods to solve high dimensional unconstrained optimization problems

$$x^* = \arg\min_{x\in\mathbb{R}^d} f(x),$$

where the target function (loss function) $f(x)$ defined in $\mathbb{R}^d$ achieves a unique global minimizer.

Date: December 9, 2020.

2010 *Mathematics Subject Classification.* 37N40, 90C26.

Key words and phrases. Consensus-based optimization, global optimization, machine learning, curse of dimensionality.

A high-dimensional nonlinear, non-convex optimization is an essential part of machine learning problems, with the target function defined in general as

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} \|\mathcal{N}_x(\hat{x}) - \hat{y}\|,$$

where x is the parameter vector and $\mathcal{N}_x$ represents a neural network representation[1]. $(\hat{x}_i, \hat{y}_i)_{i=1}^n$ is a set of labeled data, and $\|\cdot\|$ is the L^2 distance between a predicted data point and the corresponding labeled data point.

The gradient descent method, most frequently used method in optimization, often updates the parameters by the iteration scheme

$$x^{t+1} = x^t - \alpha \nabla f(x^t)$$

with α being the learning rate. However, for a big labeled data set, i.e., n is tremendously big, computing f in each iteration is time consuming, and the iterations often get stuck at local minima. The stochastic gradient descent (SGD) method [2, 3] instead computes f on a randomly selected subset of the labeled data set, by choosing m points randomly from the labeled data set with $m \ll n$ (The subset needs to be updated at each iteration). The SGD method with momentum term [17] damps oscillations in the SGD method by introducing exponentially weighted moving average as the momentum

$$\begin{aligned} x^{t+1} &= x^t - m^t, \\ m^t &= -\gamma m^{t-1} + \alpha \nabla f(x^t). \end{aligned}$$

The momentum term increases for dimensions whose gradients point toward the same direction and decreases for dimensions whose gradients change directions. Adding the momentum leads to a faster convergence than the SGD method and shows higher possibility to jump out of local minima. However, if the momentum is added too much, the global minimizer will be most likely missed. The iterator typically rolls past the global minimizer, and then rolls backwards but misses it again. Thus, adding too much momentum often generates a sequence that swings back and forward between local minima. Later, the adaptive momentum method (Adam) [12] also adds the estimation of the second order momentum

$$\begin{aligned} x^{t+1} &= x^t - \gamma \frac{\hat{m}^t}{\sqrt{\hat{v}^t} + \epsilon}, \\ m^t &= \beta_1 m^{t-1} + (1-\beta_1)\nabla f(x^t), \quad \hat{m}_t = \frac{m_t}{1-\beta_1^t}, \\ v^t &= \beta_2 v^{t-1} + (1-\beta_2)\nabla^2 f(x^t), \quad \hat{v}_t = \frac{v_t}{1-\beta_1^t}, \end{aligned}$$

where $0 < \beta_1, \beta_2 < 1$. The second order momentum here provides an adaptive adjustment of the learning rate, which has been used in AdaGrad [7], AdaDelta [23], and RMSprop. By combining the advantages of AdaGrad for

[1]Parameters in a neural network are commonly denoted by θ instead in Section 4.2.

dealing with sparse gradients and RMSProp for dealing with non-stationary objectives, the Adam method has been widely used.

However, in many cases the objective function is not differentiable and the training of deep neural networks has the issue of gradient explosion or vanishing [1]. In general, gradient-based methods do not offer a guarantee of global convergence in high dimensional and non-convex problems. Long before machine learning becomes popular, no-convex and nonlinear optimization problems have been considered in some evolutionary computation methods, including the Nelder-Mead method [14, 15], the genetic algorithm [21, 10], the simulated annealing method [19, 13], and the particle swarm optimization [16, 11]. Despite the tremendous empirical success of these techniques, it is often difficult to provide guarantees of robust convergence to the global minimizer.

The focus of the current work is the CBO method, where a particle system consisting of N particles, labeled as $X_t^i, i = 1, \cdots N$, is considered. During the dynamic evolution, the particle system tends to their weighted average, and meanwhile undergoes some fluctuation due to the random noise, such as the isotropic geometric Brownian motion [16, 4]. Ideally, these particles are expected to gather at the global minimizer of the objective function associated to the system. Mathematically, such a convergence was proved in [4] with exponential rate in time under dimension-dependent conditions, i.e., the learning rate depends on the dimension. Therefore, the CBO method may suffer from the curse of dimensionality. To overcome this issue, in [6], Carrillo, Jin, Li, and Zhu proposed to replace the isotropic geometric Brownian motion with the component-wise one. Such a modification leads to the convergence to the global minimizer with dimension-independent parameters, as proved in [9] for well-chosen initial data. From the perspective of efficiency, the idea of random mini-batch is used for quantities involving the summation of individual particle contribution [6], which reduces the computational complexity from $\mathcal{O}(N)$ to $\mathcal{O}(\frac{N}{M})$ with M being the number of particles in each batch. For extremely high dimensional problems, these method require very well-chosen initial data which may be difficult for practical problems.

In this work, we improve the CBO method [6] by adding first and second order momentum terms to damp the oscillation and accelerate the convergence. In general, we emphasize that the Adam-CBO method has the ability to handle non-differentiable object functions, and has the improved possibility to find the global minimizer of high-dimensional and non-convex functions at a cost only growing linearly with respect to the dimensionality. This will be demonstrated by various numerical experiments.

The article is organized by the following structure. In Section 2, we propose the Adam-CBO method together with a brief introduction of the CBO method for completeness. In Section 3, using the example of Rastrigin function, we find that the Adam-CBO method performs better than the CBO method with a higher possibility to find the global minimizer with the same

cost. In Section 4, using the Adam-CBO method to approximate functions, we find that the Adam-CBO method also has the spectral bias [18], or the Frequency principle [22], which is similar to the first-order methods. In addition, the Adam-CBO method is used to solve partial differential equations (PDEs) with low-regularity solutions. By using activation functions that cannot take gradients, the Adam-CBO outperforms Adam in terms of approximation accuracy. The conclusion is drawn in Section 5.

2. A consensus based optimization method with adaptive momentum estimation

In this section, we provide a detailed discussion on the Adam-CBO method and give some theoretical insights on its convergence. For completeness, we first give a brief introduction to the CBO method.

2.1. The CBO method.

The CBO method considers a stochastic interacting system of N particles with position $X_t^i = (x_1^i, \cdots, x_d^i)^T \in \mathbb{R}^d$, whose dynamics can be described as a first order system [5, 4, 6]

$$\dot{X}_t^i = -\lambda(X_t^i - x^*) + \sigma(X_t^i - x^*)\dot{W}_t^i, \quad 1 \le i \le N, \tag{1}$$

where λ represents the learning rate, N is the number of particles, and M is the number of particles in each batch. Here

$$x^* = \sum_{i=1}^{N} X_t^i \frac{\omega_f^\alpha(X_t^i)}{\sum_{j=1}^{N} \omega_f^\alpha(X_t^j)},$$

where ω_f^α is a weight function and can be taken as mode $\omega_f^\alpha = \exp(-\alpha f(x))$ for some appropriately chosen $\alpha > 0$, and $f(x)$ is a given (possibly non-convex) function to be optimized. $\dot{X}$ denotes the temporal derivative of X.

We discretize the system (1) with stepsize 1, and obtain

$$X_{t+1}^i = X_t^i - \lambda(X_t^i - x^*) + \sigma(X_t^i - x^*)\mathrm{d}W_t^i, \tag{2}$$

The component-wise geometric Brownian motion W_t^i is used to replace the noise in the numerical implementation. Details of the algorithm can be found in **Algorithm 1**. Without loss of generality, we assume $N//M$. Note that t_N represents the maximum number of temporal steps, or the final time due to the stepsize 1. If necessary, one can choose a stopping criterion, like $\max_i |X_t^i - x^*| < e$ to stop the update ahead of the final time $t = t_N$.

2.2. The Adam-CBO method.

By introducing an additional momentum M_t^i, we rewrite the first order system (1) in Section 2.1 into

$$\dot{X}_t^i = -\lambda M_t^i + \sigma^t \dot{W}_t^i, \quad i = 1, \cdots, N, \tag{3}$$

$$M_t^i = X_t^i - x^*. \tag{4}$$

Note that the stochastic term in (3) is isotropic since it is found that such a modification leads to a better numerical performance in the Adam-CBO

Algorithm 1: Consensus-based global optimization method.

Input: λ, N, M, t_N

/* λ represents the learning rate, N is the number of particles, M is the number of particles in each batch and t_N is the number of iterations. */

Initial X_0^i, $i = 1, \cdots N$;

for $t = 0$ ***to*** t_N **do**

  Generate an index set P_k by random permutation of $\{1, 2, \cdots, N\}$;

  Generate batch sets of particles in the order of P_k as $B^1, \cdots B^{\frac{N}{M}}$ with each batch having M particles;

  for $j = 1$ ***to*** $\frac{N}{M}$ **do**

    Update $x^* = \sum_{k \in B^j} \frac{X_t^k \mu_t^k}{\sum_{i \in B^j} \mu_t^i}$, where $\mu_t^i = \omega_f^\alpha(X_t^i)$;

    Update X_t^i for $j \in B^j$ as follows

    $X_{t+1}^i = X_t^i - \lambda\gamma_{k,\theta}(X_t^i - x^*) + \sigma_{k,\theta}\sqrt{\gamma_{k,\theta}} \sum_{k=1}^{d} \vec{e}_k(X_t^i - x^*)z_i \quad z_i \sim N(0,1)$.

    /* e_k is the unit vector along the k-th dimension. */

  end

end

Output: $X_{t_N}^i, \quad i = 1 \cdots N$

method, while the anisotropic stochastic term in the CBO method performs better with theoretical guarantees [9]. Discretization of (3) yields

$$X_{t+1}^i = X_t^i - \lambda M_t^i + \sigma^t dW_t^i. \tag{5}$$

By definition (4), we have

$$\begin{aligned} M_{t+1}^i &= X_{t+1}^i - x^* = (X_{t+1}^i - X_t^i) + M_t^i \\ &= (1-\lambda)M_t^i + \sigma^t dW_t^i. \end{aligned}$$

To update the momentum M_t^i adaptively, we borrow the idea from the Adam method [12]. In the asymptotic sense, as $t \to +\infty$, $\sigma^t dW_t^i$ can be represented by λM_{t+1}^i. Thus the above equation can be rewritten as

$$M_{t+1}^i = \beta_1 M_t^i + (1-\beta_1)(X_{t+1}^i - x^*) \tag{6}$$

with $\beta_1 = 1 - \lambda$.

We now show the relationship between M_t^i and the first moment of $X_t^i - x^*$. Using (6) recursively, one gets

$$\begin{aligned} M_t^i &= \beta_1 M_{t-1}^i + (1-\beta_1)(X_t^i - x^*) \\ &= \beta_1(\beta_1 M_{t-2}^i + (1-\beta_1)(X_{t-1}^i - x^*)) + (1-\beta_1)(X_t^i - x^*) \\ &= \cdots \\ &= (1-\beta_1)\sum_{k=0}^{t} \beta_1^{t-k}(X_k^i - x^*). \end{aligned}$$

Assume that $X_k^i - x^*$ is stationary, i.e., they have the same distribution for different k, then

$$\begin{aligned} \mathbb{E}[M_t^i] &= (1-\beta_1)\mathbb{E}[\sum_{k=0}^{t} \beta_1^{t-k}(X_k^i - x^*)] \\ &= (1-\beta_1)\mathbb{E}[X_t^i - x^*]\sum_{k=0}^{t} \beta_1^{t-k} \\ &= (1-\beta_1^t)\mathbb{E}[X_t^i - x^*]. \end{aligned}$$

Therefore, M_t^i gives an estimation of the first moment of $(X_k^i - x^*)$ as $t \to \infty$. To get an unbiased estimation of $(X_k^i - x^*)$ for small t as well, we rescale M_t^i by $(1-\beta_1^t)$ and denote by $\hat{M}_t^i$ in **Algorithm 2**. This argument provides a connection between (5) and (2).

For the second order moment $\mathbb{E}(|X_t^i - x^*|^2)^2$, we define

$$V_t^i = \beta_2 V_{t-1}^i + (1-\beta_2)|X_t^i - x^*|^2. \tag{7}$$

Application of the same argument for $\mathbb{E}[X_t^i]$ yields

$$\mathbb{E}[V_t^i] = (1-\beta_2^t)\mathbb{E}[|X_t^i - x^*|^2], \tag{8}$$

and $\hat{V}_t^i = \frac{V_t^i}{1-\beta_2^t}$ is an unbiased estimation of $\mathbb{E}[|X_t^i - x^*|^2]$. Therefore, we modify (5) by

$$X_{t+1}^i = X_t^i - \frac{\lambda \hat{M}_{t+1}^i}{\sqrt{\hat{V}_{t+1}^i} + \epsilon} + \sigma^t dW_t^i, \tag{9}$$

where ϵ is a small number and typically takes the value $1e-8$ to avoid the vanishing of the denominator. Combining (9), (6), and (7) gives **Algorithm 2**. Although $\beta_1 = 1-\lambda$ in the above derivation, β_1 and β_2 are chosen to be independent of λ. In practice, we set $\beta_1 = 0.9$ and $\beta_2 = 0.99$.

The Adam-CBO method differs from the CBO method in the following aspects. First, it adds estimations of first momentum M_t^i ($\hat{M}_t^i$) and second momentum V_t^i ($\hat{V}_t^i$) into the algorithm without increasing much computational costs. Second, the component-wise geometric Brownian motion term

[2]The square here is defined in the element-wise sense.

$\sum_{k=1}^{d} \vec{e}_k(X_t^i - x^*)z_i$ is replaced by $\sum_{k=1}^{d} \vec{e}_k z_i$, which puts stochastic effects in different dimensions on equal footing. In the case that X_t^i can converge to x^* quickly, so the Adam-CBO method shall have the stronger ability to explore the landscape of the loss function. Note that σ^t in **Algorithm 2** is a decreasing function of t, so the method is expected to converge at the finial time. Typically, $\sigma^t = 0.99^{t/10}$ or $\sigma^t = 0.99^{t/100}$ is used in practice.

Algorithm 2: Consensus-based global optimization method with adaptive momentum estimation.

Input: λ, N, M, t_N, β_1, β_2

`/* ` λ `represents the learning rate, and` β_1, β_2 `are the exponential decay rates for the first and the second order moment estimation, respectively. */`

Initialize X_0^i, $i = 1, \cdots N$ by the uniform distribution;

Initial $M_0^i, V_0^i = 0$; `/* Initialize first order and second order moments. */`

for $t = 0$ ***to*** t_N **do**

  Generate a random permutation of index $\{1, 2, \cdots, N\}$ to form set P_k;

  Generate batch set of particles in order of P_k as $B^1, \cdots B^{\frac{N}{M}}$ with each batch having M particles;

  for $j = 0$ ***to*** $\frac{N}{M}$ **do**

    Update $x^* = \sum_{k \in B^j} \frac{X_t^k \mu_t^k}{\sum_{i \in B^j} \mu_t^i}$, where $\mu_t^i = \omega_f^\alpha(X_t^i)$;

    Update X_t^i for $j \in B^j$ as follows

    $M_{t+1}^i = \beta_1 M_t^i + (1-\beta_1)(X_t^i - x^*) \qquad \hat{M}_{t+1}^i = M_{t+1}^i/(1-\beta_1^t)$;

    $V_{t+1}^i = \beta_2 V_t^i + (1-\beta_2)(X_t^i - x^*)^2 \qquad \hat{V}_{t+1}^i = V_{t+1}^i/(1-\beta_2^t)$;

    $X_{t+1}^i =$

    $X_t^i - \lambda \hat{M}_t^i/(\sqrt{\hat{V}_t^i} + \epsilon) + \sigma^t \sum_{k=1}^{d} \vec{e}_k z_i \quad z_i$ is a random variable.

  end

end

Output: $X_{t_N}^i, \quad i = 1 \cdots N$

Note that the Adam-CBO method is designed to be adaptive by choosing the learning rate (step size) λ automatically rather than empirically. It adapts the learning rate to the parameters, and performs smaller updates (low learning rates) for parameters associated with frequently occurring features, and larger updates (high learning rates) for parameters associated with infrequent features.

2.3. **A linear stability analysis of the Adam-CBO method.** To understand the algorithmic performance, we consider the linearized problems

of both methods at the continuous level and prove their convergences. Note that this does not prove the convergence of the Adam-CBO method, but provides an intuitive understanding of it. We first rewrite **Algorithm 2** into a continuous form and ignore the stochastic term

$$\dot{m} = (\beta_1 - 1)m + (1 - \beta_1)(x - \bar{x}), \tag{10}$$

$$\dot{v} = (\beta_2 - 1)v + (1 - \beta_2)(x - \bar{x})^2, \tag{11}$$

$$\hat{m} = \frac{m}{1 - \beta_1^t} \quad \hat{v} = \frac{v}{1 - \beta_2^t}, \tag{12}$$

$$\dot{x} = -\lambda \frac{\hat{m}}{\sqrt{\hat{v}} + \epsilon}, \tag{13}$$

where $\bar{x}$ is the optimal solution (constant). We shall prove $x \to \bar{x}$ with a convergence rate independent of λ when x is close to $\bar{x}$ by the linear stability analysis. Denote $\tilde{x} = x - \bar{x}$. Linearizing the system (10)-(13) around $m = 0, x = \bar{x}, v = 0$, we have

$$\dot{m} = -(1 - \beta_1)m + (1 - \beta_1)\tilde{x}, \tag{14}$$

$$\dot{v} = -(1 - \beta_2)v, \tag{15}$$

$$\dot{\tilde{x}} = -\frac{\lambda}{(1 - \beta_1^t)\epsilon} m \to -\frac{\lambda}{\epsilon} m = -\mu m \quad (t \to \infty) \tag{16}$$

with $\mu = \lambda/\epsilon$, and in a vector form,

$$\partial_t \begin{pmatrix} m \\ v \\ \tilde{x} \end{pmatrix} = \begin{pmatrix} -(1 - \beta_1) & 0 & 1 - \beta_1 \\ 0 & -(1 - \beta_2) & 0 \\ -\mu & 0 & 0 \end{pmatrix} \begin{pmatrix} m \\ v \\ \tilde{x} \end{pmatrix}. \tag{17}$$

Theorem 1. ***Algorithm 2*** *generates a sequence that converges to the optimal solution with rates independent of the learning rate λ.*

Proof. Eigenvalues of the matrix on the right-hand side are $\beta_2 - 1$ and $\frac{1}{2}(\beta_1 - 1 \pm i\sqrt{1 - \beta_1}\sqrt{\beta_1 - 1 + 4\mu})$ (typically $1 - \beta_1 \ll 4\mu$), respectively. Thus, $m, v, \tilde{x}$ decay to 0 exponentially with rate $\beta_2 - 1$ when $\beta_1 > 2\beta_2 + 1$ and with rate $\frac{1}{2}(\beta_1 - 1)$ when $\beta_1 < 2\beta_2 + 1$ in an oscillatory way. □

The CBO method without random noise can be written into a continuous form

$$\dot{x} = -\lambda(x - \bar{x}). \tag{18}$$

The ODE can be solved analytically with a decay rate $e^{-\lambda t}$ towards the stationary point. Therefore, the decay rate of the CBO method depends exponentially on the learning rate λ.

Remark 1. *Although the above analysis indicates that the decay rate of the Adam-CBO method is independent of λ, λ does control the oscillatory behavior during the iteration. Therefore, we argue that during the initial training*

stage, a large λ is favored to make particles oscillate and escape local minima. During the finial training stage, to make the particles converge to the global minimizer faster, we often set a smaller λ to control the oscillations.

3. The Rastrigin function

In this section, we demonstrate the advantage of the Adam-CBO method by finding the global minimizer of the Rastrigin function

$$f(x) = \frac{1}{d}\sum_{i=1}^{d}\left[(x_i - B)^2 - 10\cos(2\pi(x_i - B)) + 10\right] + C \tag{19}$$

with $B = \arg\min f(x)$ and $C = \min f(x)$. Figure 1 is a visualization of (19) when $d = 2$ and $B = C = 0$.

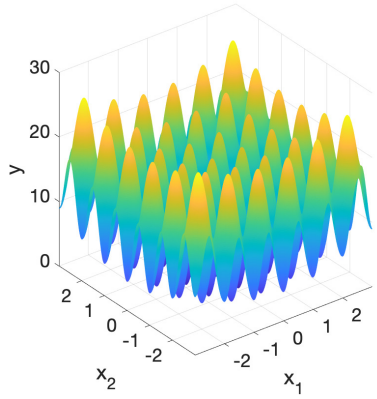


Figure 1. The landscape of the rastrigin function in two dimension with $x \in [-3, 3]^2$ and $B = C = 0$.

Number of local minima of the rastrigin function in terms of dimension when $B = C = 0$ and $x \in (-3, 3)$ is listed in Table 1. The number of local minima is 5^d, which grows exponentially fast in term of the dimensionality. When $d = 1000$, the number of minima is 5^{1000}, approximately 10^{690}.

d	1	2	30	100	1000
Number of local minima	5	5^2	5^{30}	5^{100}	5^{1000}

Table 1. Number of local minima for the Rastrigin function in terms of dimension.

Results of CBO and Adam-CBO methods with several random processes, including uniform, Gaussian, and Levy processes are recorded in Table 2. In all numerical examples, $\beta_1 = 0.9$ and $\beta_2 = 0.99$ in the Adam-CBO method. Here B in (19) is set to be a value between $[-3, 3]$ and $X_0^i,\ i = 1, \cdots, N$ are initialized between $[-3, 3]$. For each case, we run the algorithm 100 times

d	N	M	CBO		
			$\mathcal{N}(0,1)$	$\mathcal{U}(-1,1)$	Wiener process
2	50	40	100%	100%	99%
10	50	40	100%	100%	2%
20	50	40	98%	22%	0%
20	50	20	66%	2%	0%
30	50	40	26%	0%	0%
30	500	5	0%	0%	0%
d	N	M	Adam-CBO		
			$\mathcal{N}(0,1)$	$\mathcal{U}(-1,1)$	Wiener process
30	500	5	99%	100%	0%
100	5000	5	100%	100%	0%
1000	8000	50	92%	20%	0%

TABLE 2. Comparison of CBO and Adam-CBO methods with different random processes. Setup of parameters are: $\lambda = 1, \gamma = 0.01, \sigma = 5.1$ in the CBO method with $\mathcal{N}(0,1)$; $\lambda = 0.01, \gamma = 0.1, \sigma = 3$ in the CBO method with $\mathcal{U}(-1,1)$; $\lambda = 0.5, \gamma = 0.1, \sigma = 0.1$ in the CBO method with Wiener process. In the Adam-CBO method, we set $\lambda = 0.1$ and $\sigma^t = 0.99^{\frac{t}{20}}$ in all cases. For each random process, hyper-parameters have been optimized in order to get the best success rate.

and check the success rate. It is found that the CBO method fails to find the global minimizer when the dimension is over 30, but the Adam-CBO method still has high success rates even when the dimension reaches 1000. Moreover, it is found that the Poisson process almost always has higher success rates than uniform and Levy processes.

Next, we compare the dependence of success rates on the batch number of particles in Table 3. It is observed that the Adam-CBO method usually has higher success rates when the particle batch size M becomes smaller and has higher success rates as the total number of particles grows. Table 4 records the success rate in terms of the number of particles N. As N grows, the success rate increases. One may doubt that the Adam-CBO method shall be sensitive to the initialization. To check this point, instead of choosing initial data randomly, we set initial X_t^i to be 0, i.e., all particles are initially set to be 0. Table 5 shows that the Adam-CBO method still has high success rates.

It is worth mentioning that different choices of stochastic terms are used in CBO and Adam-CBO methods. These choices are purely based on numerical experiences. For the Rastrigin function in high dimensions, we observe that the component-wise geometric Brownian motion term $\sum_{k=1}^{d} \vec{e}_k(X_t^i - x^*)z_i$

d	N	M	Adam-CBO		N	M	Adam-CBO	
			$\mathcal{N}(0,1)$	$\mathcal{U}(-1,1)$			$\mathcal{N}(0,1)$	$\mathcal{U}(-1,1)$
100	1000	5	87%	39%	5000	5	100%	84%
100	1000	10	94%	60%	5000	10	100%	100%
100	1000	20	87%	49%	5000	20	100%	100%
100	1000	25	77%	53%	5000	25	100%	100%
100	1000	50	45%	8%	5000	50	100%	100%
100	1000	100	2%	0%	5000	100	100%	100%

TABLE 3. Comparison of success rates for different batch numbers when the dimension is 100, $\lambda = 0.1$, and $\sigma^t = 0.99^{\frac{t}{20}}$.

d	N	M	Adam-CBO	
			$\mathcal{N}(0,1)$	$\mathcal{U}(-1,1)$
1000	8000	50	92%	20%
1000	10000	50	100%	28%
1000	12000	50	100%	28%
1000	14000	50	100%	32%
1000	16000	50	100%	32%

TABLE 4. Comparison of success rates for different numbers of particles when the dimension is 1000, $\lambda = 0.1$, and $\sigma^t = 0.99^{\frac{t}{20}}$.

d	N	M	Adam-CBO	
			$\mathcal{N}(0,1)$	$\mathcal{U}(-1,1)$
30	500	5	94%	100%
100	5000	5	100%	94%
1000	10000	50	100%	11%

TABLE 5. Comparison of success rates for different dimensions when X_t^i is initialized by 0 ($X_0^i = 0$), $\lambda = 0.1$, and $\sigma^t = 0.99^{\frac{t}{20}}$.

in **Algorithm 1** provides better results for the CBO method, while the term $\sum_{k=1}^{d} \vec{e}_k z_i$ in **Algorithm 2** provides better results for the Adam-CBO method. Similar results are observed when applying both methods to neural networks.

4. Application of the Adam-CBO method on neural networks

In this section we will apply the Adam-CBO method to deep neural networks. For completeness, we briefly introduce deep neural networks (DNNs) and its two applications: approximating functions and solving PDEs. A DNN is constructed by a composition of some basic units which contain

activation function $\sigma(x)$ and linear transform $Wx+b$. More precisely, we define the simplest network

$$\mathbb{D}(x;\theta) = \mathbb{N}_m(\mathbb{N}_{m-1}(\cdots \mathbb{N}_1(x))), \tag{20}$$

where $\mathbb{N}_i(x) = \sigma(W^i x + b^i)$. The linear transform $W^i x + b^i$ can transfer a vector x to any dimension, so the output dimension of $\mathbb{N}_i$ can be different. Typically, for $x \in \mathbb{R}^d$, we fix the width by choosing $W^1 \in \mathbb{R}^{n,d}$, $W^i \in \mathbb{R}^{n,n}$ for $i = 2, \cdots m-1$, and $W^m \in \mathbb{R}^{1,n}$. Therefore, we denote m as the network depth and n as the network width. The parameter set θ consists of W^i and b^i for $i = 1, \cdots, m$, which will be optimized by an optimization method.

For function approximations, we consider to approximate a target function $u(x)$ by a DNN $\mathbb{D}(x)$ over domain Ω. The objective function is defined as

$$f(\theta) = \|\mathbb{D}(x;\theta) - u(x)\|^2_{L^2(\Omega)}. \tag{21}$$

For a Poisson equation, the target solution $u(x)$ is not given in advance, but it satisfies

$$\begin{cases} -\Delta u = f & x \in \Omega \\ u = g & x \in \partial\Omega \end{cases}. \tag{22}$$

By using the Deep Ritz method [20], we define the loss function as

$$f(\theta) = \int_\Omega \frac{1}{2}|\nabla \mathbb{D}(x;\theta)|^2 - f(x)\mathbb{D}(x;\theta)\mathrm{d}x + \eta \int_{\partial\Omega} (\mathbb{D}(x;\theta) - g(x))^2 \mathrm{d}x. \tag{23}$$

For (21) and (23), the goal is to find the global minimizer of the following problem

$$\arg\min_\theta f(\theta). \tag{24}$$

4.1. **Approximating functions.** In this section, we will demonstrate that the Adam-CBO method share the property of spectral bias, or frequency principle as gradient-based methods do [18, 22], i.e., it approximates low-frequency properties of the target function first and high-frequency properties later. Consider two functions

$$u(x) = \sin(2\pi x) + \sin(8\pi x^2), \tag{25}$$

$$u(x) = \begin{cases} 1 & x < -\frac{7}{8}, x > \frac{7}{8}, -\frac{1}{8} < x < \frac{1}{8} \\ -1 & \frac{3}{8} < x < \frac{5}{8}, -\frac{5}{8} < x < -\frac{3}{8} \\ 0 & \text{otherwise} \end{cases}, \tag{26}$$

where the first function is smooth while the second one is not. Here we use the sigmoid function $\sigma(x) = \frac{1}{1+\exp(-x)}$ as the activation function. The training process of the Adam-CBO method is visualized in Figure 2 for (25) and in Figure 3 for (26). Clearly, the low-frequency information is approximated first and the high-frequency one is captured later.

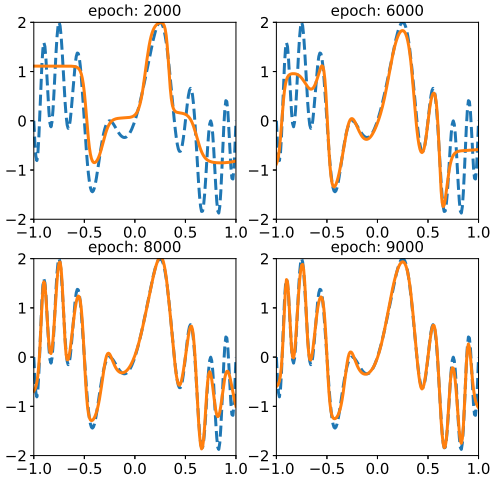


FIGURE 2. Approximating function (25) using a network with $n = 50$, $m = 3$, and 2701 parameters in total. The learning rate is $\lambda = 0.2$. $N = 500$ particles and $M = 5$ particles for each batch are used in the first 50000 iterations. After that, the random term is ignored and $M = 10$ is used for faster convergence to the optimal solution.

4.2. **Deep neural networks.** The commonly used gradient-based method has the issue of gradient vanishing or gradient explosion when the network depth increases. At the formal level, the Adam-CBO method is independent of the gradient of the loss function with respect to the parameters. Thus it is interesting to check its performance for deeper neural networks. We use DNNs with a fixed width 10 and different depths to approximate the function

$$u(x) = \sin(k\pi x^k). \tag{27}$$

Set $N = 500$ particles, $M = 5$ particles for each batch, and the learning rate $\lambda = 0.2$ in the first 30000 epochsiterations. Between 30000 epochs to 80000 epochs, we set $M = 20$ and ignore the random term to accelerate the convergence. Between 80000 epochs to 150000 epochs, we set $M = 100$. After 150000 epochs, we set the learning rate $\lambda = 1e - 2$ to minimize the oscillations. Numerical results are shown in Table 6. For networks with depths $4, 7, 12, 22$, the Adam-CBO method keeps converging to the exact solution. In the implementation, the training process stops after 2×10^6 iterationss. Since the gradient-free method (Adam-CBO) converges slower than the gradient-based method (SGD or Adam), parameters in the neural network fall around the optimal solution but converge to it slowly at the end

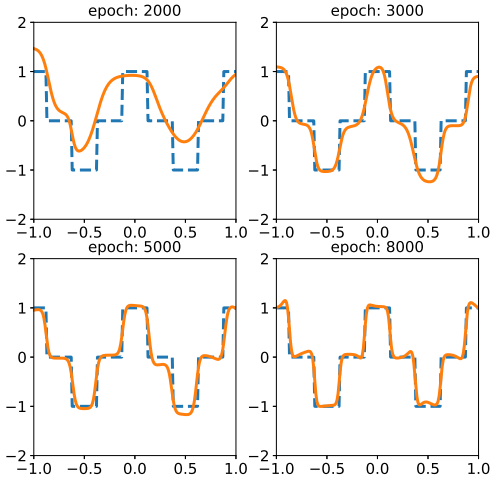


FIGURE 3. Approximating function (26) using a network with $n = 50$, $m = 3$, and 2701 parameters in total. The learning rate is $\lambda = 0.2$. $N = 500$ particles and $M = 5$ particles for each batch are used in the first 50000 iterations. After that, the random term is ignored and $M = 10$ is used for faster convergence to the optimal solution.

depth	Num of parameters	k = 2	k = 3	k = 4
4	141	6.62 e-03	1.32 e-02	1.71 e-01
7	471	4.78 e-03	1.42 e-02	7.54 e-03
12	1021	7.44 e-03	1.30 e-02	5.32 e-02
22	2121	1.00 e-02	1.01 e-02	1.21 e-01

TABLE 6. Dependence of approximation error measured in absolute L^2 norm in terms of network depth for (27) when $k = 2, 3, 4$.

of the training process. Therefore, it is difficult to obtain the convergence rate of the approximation accuracy in terms of the network depth. However, if SGD or Adam is used with parameters initialized by the uniform distribution, neither method converges well (with final error around 0.3 in absolute L^2 norm) when the network depth is 4 and 10, respectively.

4.3. **Solving PDEs with low-regularity solutions.** In this section, we will use the Adam-CBO method to solve PDEs with low-regularity solutions. There has an increasing interest in the development of machine-learning method for solving PDEs; see [8] for review and references therein. For the purpose of low-regularity solutions, we adopt the Deep Ritz method (DRM)

[20], which is based on the variational formulation associated to the PDE. Consider an elliptic PDE

$$\begin{cases} -\nabla\cdot(A(x)\nabla u) = -\sum_{i=1}^{d}\delta(x_i) & x\in\Omega=[-1,1]^d \\ u(x)=g(x) & x\in\partial\Omega \end{cases} \tag{28}$$

with

$$A(x)=\begin{bmatrix} (x_1^2)^{\frac{1}{4}} & & \\ & \ddots & \\ & & (x_d^2)^{\frac{1}{4}} \end{bmatrix}. \tag{29}$$

The exact solution $u(x)=\sum_{i=1}^{d}|x_i|^{\frac{1}{2}}$. One can see that the solution is only in $H^{1/2}(\Omega)$ and has singularities when evaluating its derivative at $x_i=0$. The loss function in DRM reads as

(30)

$$I[u]=\int_\Omega \frac{1}{2}(\nabla u)^T A(x)\nabla u(x)\mathrm{d}x+\sum_{i=1}^{d}\int_{-1}^{1}\delta(x_i)u(x)\mathrm{d}x_i+\eta\int_{\partial\Omega}(u(x)-g(x))^2\mathrm{d}x,$$

where $\eta=500$ is the penalty parameter for the boundary condition.

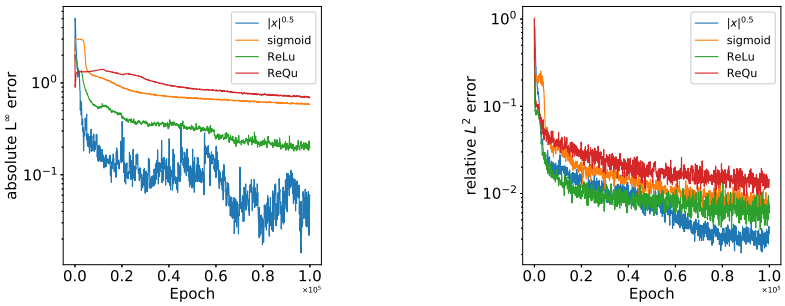
Activation functions used in Adam include ReLu ($\max\{x,0\}$), ReQu ($(\max\{x,0\})^2$), and sigmoid ($\frac{1}{1+\exp(-x)}$). Since the Adam-CBO method is a gradient-free method, to demonstrate its advantage, we use $|x|^{\frac{1}{2}}$ as the activation function. Another reason for choosing this activation function is its low regularity, which leads to superior approximation accuracy in this case. Note that a loss function including this activation function is not differentiable and thus gradient-based methods are not applicable. Numerical results are shown in Table 7. The training process is shown in Figure 4. One-dimensional solution profiles at the intersection where other coordinates are set to be 0 are visualized in Figure 5. It is found that the Adam-CBO method provides better results than Adam with different activation functions. This attributes to the usage of non-differentiable activation functions which better approximate low-regularity PDEs. Moreover, the $|x|^{0.5}$ activation function approximates the low-regularity solution better than any other activation functions near the singularity.

5. Conclustion

In this work, we propose a consensus-based global optimization method with adaptive momentum estimation based on the consensus-based global optimization method and the adaptive momentum estimation. It shows strong abilities to find global minima for high dimensional problems, including given functions in high dimensions and approximation of low-regularity solutions to PDEs by deep neural networks. The computational complexity is found to grow linearly with respect to the dimension of the parameter

d	n	m	Activation-Optimizer	L^2 error	L^∞ error
2	20	2	ReLu-Adam	1.23 e-02	9.91 e-02
			ReQu-Adam	2.22 e-02	4.21 e-01
			sigmoid-Adam	2.19 e-02	3.14 e-01
			$\|x\|^{0.5}$ - Adam-CBO	3.96 e-03	2.09 e-02
4	40	2	ReLu-Adam	6.72 e-03	3.70 e-01
			ReQu-Adam	1.43 e-02	1.10 e -00
			sigmoid-Adam	7.90 e-03	7.66 e -02
			$\|x\|^{0.5}$ -Adam-CBO	3.13 e-03	9.52 e -02

TABLE 7. Errors measured in L^2 and L^∞ norms for (28) by Adam and Adam-CBO methods.

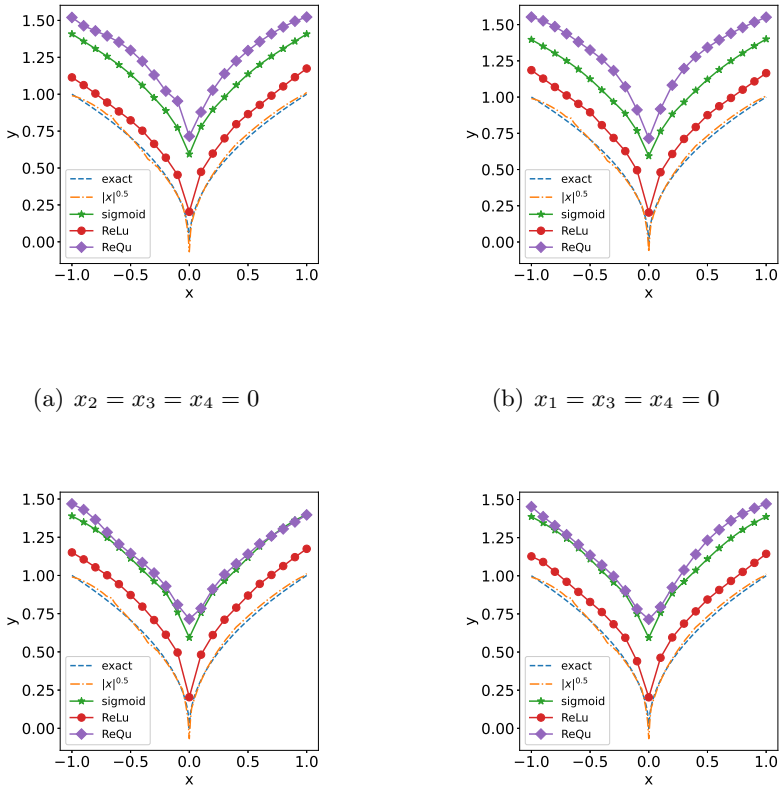


(a) L^∞ error (b) L^2 error

FIGURE 4. Training process of Adam and Adam-CBO methods for (28) when the dimension is 4. (a) L^∞ error; (b) L^2 error.

space. Since it is free of gradient, the Adam-CBO method is a suitable choice for problems where derivatives with respect to parameters do not exist. Therefore, it will be of great interest to find the application of the Adam-CBO method for machine learning tasks where non-differentiable activation functions are needed and the dimensionality of parameter space is high.

Acknowledgment. This work of J. Chen was supported by National Key R&D Program of China (No. 2018YFB0204404) and NSFC grant No. 11971021. The work of S. Jin was supported by NSFC grant No. 11871297.



(a) $x_2 = x_3 = x_4 = 0$

(b) $x_1 = x_3 = x_4 = 0$

(c) $x_1 = x_2 = x_4 = 0$

(d) $x_1 = x_2 = x_3 = 0$

FIGURE 5. One-dimensional solution profiles at the intersection. (a) $x_2 = x_3 = x_4 = 0$; (b) $x_1 = x_3 = x_4 = 0$; (c) $x_1 = x_2 = x_4 = 0$; (d) $x_1 = x_2 = x_3 = 0$.

REFERENCES

[1] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[2] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of COMPSTAT'2010*, pages 177–186, Heidelberg, 2010. Physica-Verlag HD.

[3] Léon Bottou. Stochastic gradient descent tricks. In *Neural Networks, Tricks of the Trade, Reloaded*, volume 7700 of *Lecture Notes in Computer Science (LNCS)*, pages 430–445. Springer, 2012.

[4] José A. Carrillo, Young-Pil Choi, Claudia Totzeck, and Oliver Tse. An analytical framework for consensus-based global optimization method. *Mathematical Models and Methods in Applied Sciences*, 28(06):1037–1066, 2018.

[5] José A. Carrillo, Massimo Fornasier, Jesus Rosado, and Giuseppe Toscani. Asymptotic flocking dynamics for the kinetic Cucker-Smale model. *SIAM Journal on Mathematical Analysis*, 42(1):218–236, 2010.

[6] José A. Carrillo, Shi Jin, Lei Li, and Yuhua Zhu. A consensus-based global optimization method for high dimensional machine learning problems. *arXiv preprint arXiv:1909.09249*, 2019.

[7] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[8] Weinan E, Jiequan Han, and Arnulf Jentzen. Algorithms for solving high dimensional PDEs: From nonlinear Monte Carlo to machine learning. *arXiv preprint arXiv:2008.13333*, 2020.

[9] Seung-Yeal Ha, Shi Jin, and Doheon Kim. Convergence of a first-order consensus-based global optimization algorithm. *arXiv preprint arXiv:1910.08239*, 2019.

[10] Georges R. Harik, Fernando G. Lobo, and David E Goldberg. The compact genetic algorithm. *IEEE transactions on evolutionary computation*, 3(4):287–297, 1999.

[11] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-International Conference on Neural Networks*, volume 4, pages 1942–1948. IEEE, 1995.

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[13] Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[14] Jeffrey C. Lagarias, James A. Reeds, Margaret H. Wright, and Paul E. Wright. Convergence properties of the Nelder–Mead simplex method in low dimensions. *SIAM Journal on Optimization*, 9(1):112–147, 1998.

[15] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 01 1965.

[16] Riccardo Poli, James Kennedy, and Tim Blackwell. Particle swarm optimization. *Swarm intelligence*, 1(1):33–57, 2007.

[17] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1):145–151, Jan 1999.

[18] Nasim Rahaman, Devansh Arpit, Aristide Baratin, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of deep neural networks. *arXiv preprint arXiv:1806.08734*, 2018.

[19] Peter J. M. van Laarhoven and Emile H. L. Aarts. *Simulated Annealing: Theory and Applications*. Springer, Dordrecht, The Netherlands, 1987.

[20] E Weinan and Bing Yu. The deep ritz method: a deep learning-based numerical algorithm for solving variational problems. *Communications in Mathematics and Statistics*, 6(1):1–12, 2018.

[21] Darrell Whitley. A genetic algorithm tutorial. *Statistics and computing*, 4(2):65–85, 1994.

[22] Zhi-Qin John Xu, Yaoyu Zhang, Tao Luo, Yanyang Xiao, and Zheng Ma. Frequency principle: Fourier analysis sheds light on deep neural networks. *arXiv preprint arXiv:1901.06523*, 2019.

[23] Matthew D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

School of Mathematical Sciences and Mathematical Center for Interdisciplinary Research, Soochow University, Suzhou, 215006, China
E-mail address: jingrunchen@suda.edu.cn

School of Mathematical Sciences, Institute of Natural Sciences, and MOE-LSC, Shanghai Jiao Tong University, Shanghai, 200240, China
E-mail address: shijin-m@sjtu.edu.cn

Department of Computational Mathematics, Science, and Engineering, Michigan State University, East Lansing, MI, 48824, USA
E-mail address: lyuliyao@msu.edu