# Evolutionary Gradient Descent for Non-convex Optimization

Ke Xue[1] ,Chao Qian[1*], Ling Xu[2], Xudong Fei[2]
Email: {xuek, qianc}@lamda.nju.edu.cn

[1]LAMDA Group, Nanjing University, China
[2]Huawei Technologies, China

# Outline

- Background
- Motivation
- EGD algorithm
- Theoretical analysis
- Experiments
- Conclusion

# Background

## Non-convex optimization

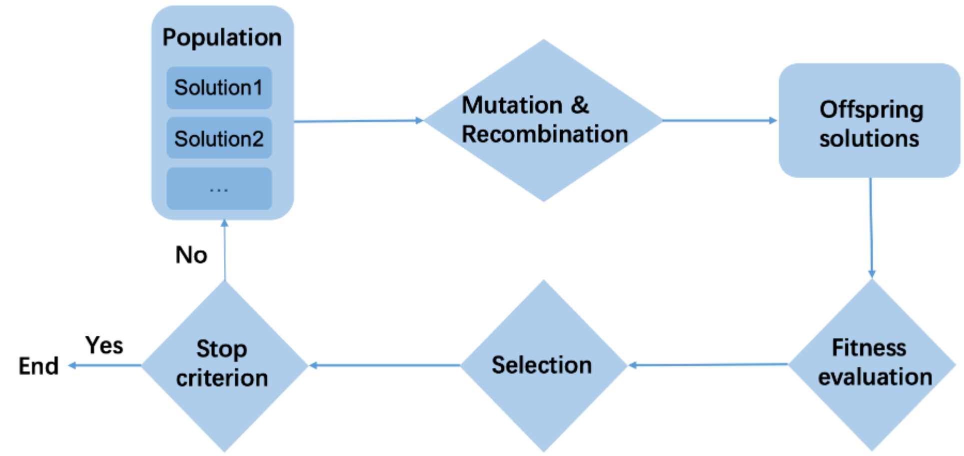
- popular in many real-world tasks
- harder to solve, contrast to convex optimization.
  - First order stationary point is global minima in convex optimization.
  - However, it maybe saddle point in non-convex optimization.

How to efficiently escape saddle points and find second order stationary point is the key issue in non-convex optimization.

# Background

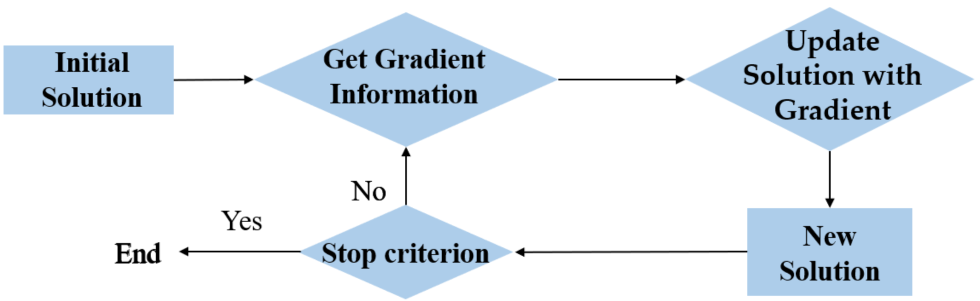
## Evolutionary Algorithms

- Global convergence
- Low efficiency, especially in high dimension




## Gradient Descent

- Perform well in high dimension and large scale tasks
- Converge to local optima generally




EA and GD each has its advantages and disadvantages.

# Motivation

A natural question:

*Can we get better algorithm for non-convex optimization by combining the merits of EAs and GD?*

Previous work:

- only combine few mechanism of EAs and GD.
- no theoretical guarantee on the convergence rate.

Our work:

- gradient + mutation + population + selection
- show the superior performance from both theoretical and experimental results.

# PGD algorithm

**Algorithm 1** PGD algorithm

**Parameter**: learning rate $\eta$, mutation strength $r$, time interval $L$ for mutation, tolerance $\epsilon$, number $T$ of iterations

**Process**:

Initialize the solution $\boldsymbol{x}_0$, set $i = i_{\text{mutate}} = 0$;
**while** $i \le T$ **do**
    **if** $\|\nabla f(\boldsymbol{x}_i)\| \le \epsilon$ and $i - i_{\text{mutate}} > L$ **then**
        $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \boldsymbol{\xi}_i$, $\boldsymbol{\xi}_i \sim \text{Uniform}(B(\mathbf{0}, r))$;
        $i_{\text{mutate}} \leftarrow i$
    **else**
        $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i - \eta \nabla f(\boldsymbol{x}_i)$
    **end if**
    $i \leftarrow i + 1$
**end while**

**Mutation**

**Gradient descent**

# EGD algorithm

**Algorithm 2** EGD algorithm

**Parameter**: learning rate $\eta$, population size $N$, mutation strength $\{r^{(p)}\}_{p=1}^{N}$, time interval $L$ for mutation, tolerance $\epsilon$, $\epsilon'$, number $T$ of iterations

**Process**:

```
Initialize the population {x_0^(p)}_{p=1}^N, set i = i_mutate = 0,
  update^(p) = 1 for p ∈ [N];
while i ≤ T do
  for p = 1 : N do
    if update^(p) = 1 then
      if ‖∇f(x_i^(p))‖ ≤ ε and i − i_mutate > L then
        update^(p) = 0
      else
        x_{i+1}^(p) ← x_i^(p) − η∇f(x_i^(p))
      end if
    end if
  end for
  if ∀p ∈ [N] : update^(p) = 0 then
    Apply Algorithm 3 for mutation and selection;
    Set update^(p) = 1 for p ∈ [N];
    i ← i + L
  end if
  i ← i + 1
end while
```

**Gradient descent update or wait for mutation**

**Mutation and selection**

# EGD algorithm

**Algorithm 3** Mutation and Selection

**for** $p = 1 : N$ **do**
  $\boldsymbol{x}_{i+1}^{(p)} \leftarrow \boldsymbol{x}_{i}^{(p)} + \boldsymbol{\xi}_{i}^{(p)}$, $\boldsymbol{\xi}_{i}^{(p)} \sim \text{Uniform}(B(\mathbf{0}, r^{(p)}))$;
  $i_{\text{mutate}} \leftarrow i$;
  **for** $j = (i+1) : (i+L)$ **do**
    $\boldsymbol{x}_{j+1}^{(p)} \leftarrow \boldsymbol{x}_{j}^{(p)} - \eta \nabla f(\boldsymbol{x}_{j}^{(p)})$
  **end for**
  **if** $f(\boldsymbol{x}_{i+1+L}^{(p)}) + \epsilon' < f(\boldsymbol{x}_{i}^{(p)})$ **then**
    $escape^{(p)} = 1$
  **else**
    $escape^{(p)} = 0$, $\boldsymbol{x}_{i+1+L}^{(p)} \leftarrow \boldsymbol{x}_{i}^{(p)}$
  **end if**
**end for**
$f_{\text{mean}} = \frac{1}{N} \sum_{p=1}^{N} f(\boldsymbol{x}_{i+1+L}^{(p)})$;
$\boldsymbol{x}_{\text{best}} = \arg\min_{\boldsymbol{x} \in \{\boldsymbol{x}_{i+1+L}^{(p)}\}_{p=1}^{N}} f(\boldsymbol{x})$;
**for** $p = 1 : N$ **do**
  **if** $escape^{(p)} = 0$ and $f(\boldsymbol{x}_{i+1+L}^{(p)}) \geq f_{\text{mean}}$ **then**
    $\boldsymbol{x}_{i+1+L}^{(p)} \leftarrow \boldsymbol{x}_{\text{best}}$
  **end if**
**end for**

**Mutation and update for *L* iterations**

**Selection**

# Theoretical analysis

**Assumption 1.** *The function $f$ is $\ell$-gradient Lipschitz and $\rho$-Hessian Lipschitz.*

**Definition 1.** *A differentiable function $f$ is $\ell$-gradient Lipschitz if*

$$\forall \boldsymbol{x}, \boldsymbol{y} : \|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| \leq \ell \cdot \|\boldsymbol{x} - \boldsymbol{y}\|.$$

*A twice-differentiable function $f$ is $\rho$-Hessian Lipschitz if*

$$\forall \boldsymbol{x}, \boldsymbol{y} : \|\nabla^2 f(\boldsymbol{x}) - \nabla^2 f(\boldsymbol{y})\| \leq \rho \cdot \|\boldsymbol{x} - \boldsymbol{y}\|.$$

**Definition 3.** *For a twice-differentiable function $f$, $\boldsymbol{x}$ is a second-order stationary point if*

$$\|\nabla f(\boldsymbol{x})\| = 0, \textit{and } \lambda_{\min}(\nabla^2 f(\boldsymbol{x})) \geq 0.$$

*For a $\rho$-Hessian Lipschitz function $f$, $\boldsymbol{x}$ is an $\epsilon$-second-order stationary point if*

$$\boxed{\|\nabla f(\boldsymbol{x})\| \leq \epsilon, \textit{and } \lambda_{\min}(\nabla^2 f(\boldsymbol{x})) \geq -\sqrt{\rho\epsilon},}$$

*where $\epsilon \geq 0$.*

# Theoretical analysis

**Theorem 1.** *[Jin et al., 2019] Let $f$ satisfy Assumption 1. Let the parameters of PGD satisfy that $\eta = \frac{1}{\ell}$, $r = \frac{\epsilon}{400\iota^3}$ and $L = \frac{\ell}{\sqrt{\rho\epsilon}} \cdot \iota$, where $\iota = c\log(\frac{d\ell(f(\boldsymbol{x}_0)-f^*)}{\rho\epsilon\delta})$, and $c$ is an absolute constant. Then for any $\epsilon, \delta > 0$, after running*

$$\boxed{\tilde{O}(\ell(f(\boldsymbol{x}_0) - f^*)/\epsilon^2)}$$

*iterations, PGD will find an $\epsilon$-second-order stationary point with probability at least $1 - \delta_{pgd}$, where*

$$\boxed{\delta_{pgd} = \frac{T^*}{4L} \cdot \frac{\ell\sqrt{d\epsilon}}{r\sqrt{\rho}} \frac{1}{\sqrt{\pi}2^{\iota}\iota} \leq \delta,}$$

*and $T^* = 8\max\{50\ell(f(\boldsymbol{x}_0) - f^*) \cdot \iota^4, \ell(f(\boldsymbol{x}_0) - f^*)\}/\epsilon^2$.*

**Theorem 1 give the iterations and probability of PGD to find $\epsilon$-second-order stationary point**

# Theoretical analysis

**Theorem 2.** *Let $f$ satisfies Assumption 1. Let the parameters of EGD satisfy that* $\eta = \frac{1}{\ell}$, $L = \frac{\ell}{\sqrt{\rho\epsilon}} \cdot \iota$ *and* $\epsilon' = \frac{1}{100\iota^3}\sqrt{\frac{\epsilon^3}{\rho}}$, *where* $\iota = c\log(\frac{d\ell(f(\boldsymbol{x}_0)-f^*)}{\rho\epsilon\delta})$, *and $c$ is an absolute constant. Then for any* $\epsilon, \delta > 0$, *after running*

$$\boxed{\tilde{O}(\ell(f(\boldsymbol{x}_0) - f^*)/\epsilon^2)}$$

*iterations, EGD will find an $\epsilon$-second-order stationary point with probability at least* $1 - \delta_{egd}$, *where*

$$\boxed{\delta_{egd} = \frac{T^*}{4L} \cdot \prod_{p=1}^{N} \frac{\ell\sqrt{d\epsilon}}{r^{(p)}\sqrt{\rho}} \frac{1}{\sqrt{\pi}2^{\iota}\iota},}$$

*and* $T^* = 8\max\{50\ell(f(\boldsymbol{x}_0) - f^*) \cdot \iota^4, \ell(f(\boldsymbol{x}_0) - f^*)\}/\epsilon^2$.

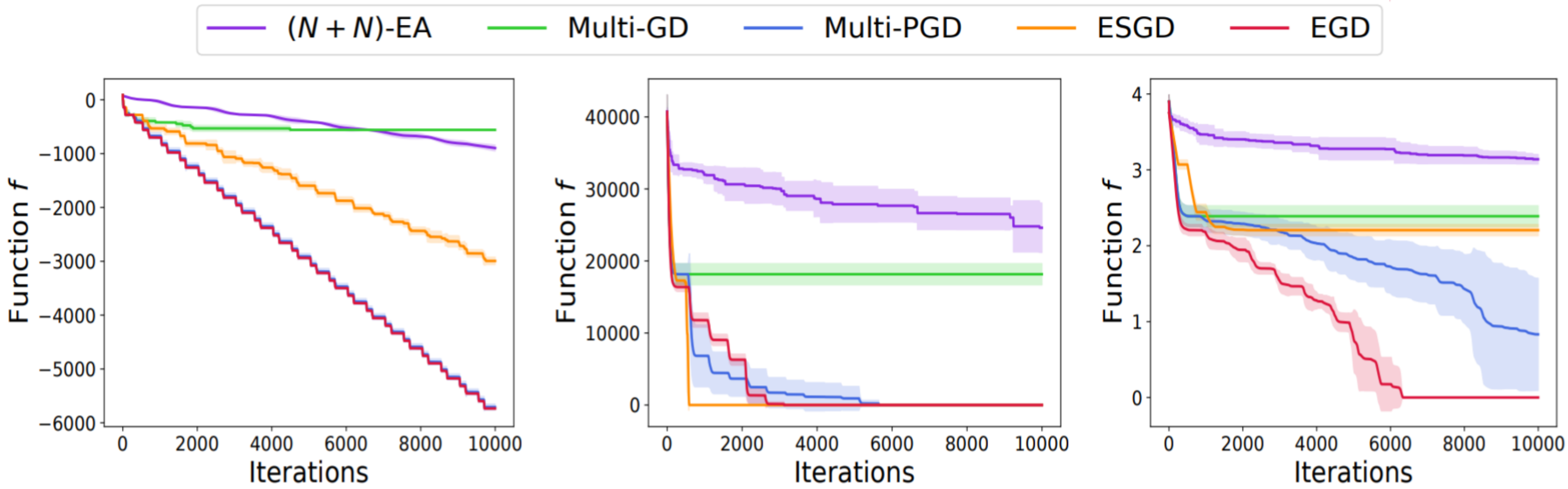
**Theorem 2 give the iterations and probability of EGD to find $\epsilon$-second-order stationary point**

**Remark 1.** *EGD will have more advantage over Multi-PGD,*

*(1) when the problem dimension $d$, the Lipschitz parameters $\ell$ and $\rho$ are larger, implying that the problem is more challenging;*

*(2) when the population size $N$ is larger.*

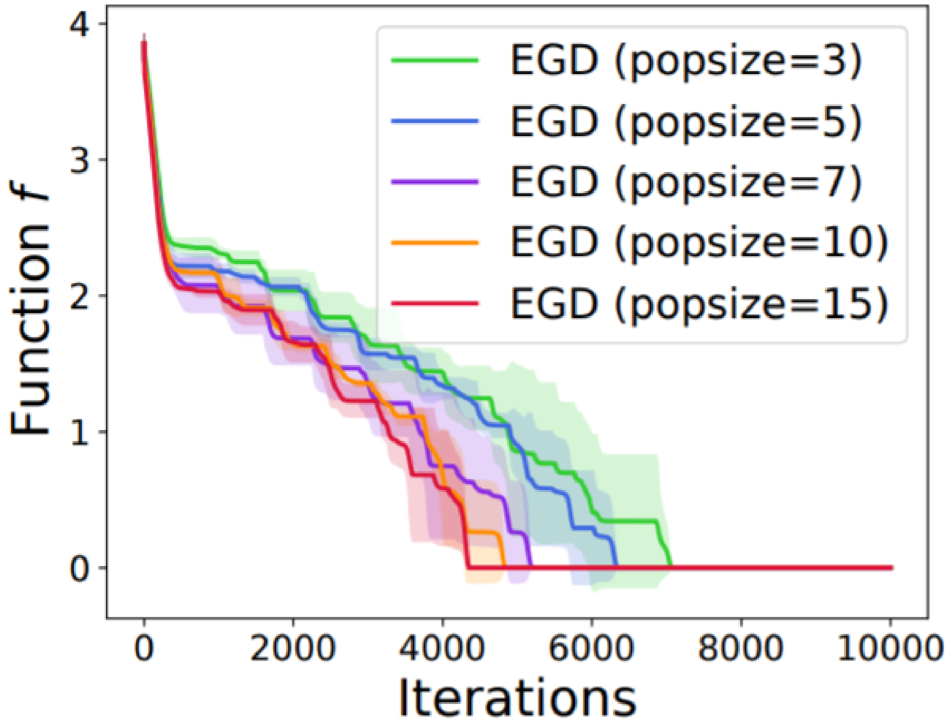
# Experiments



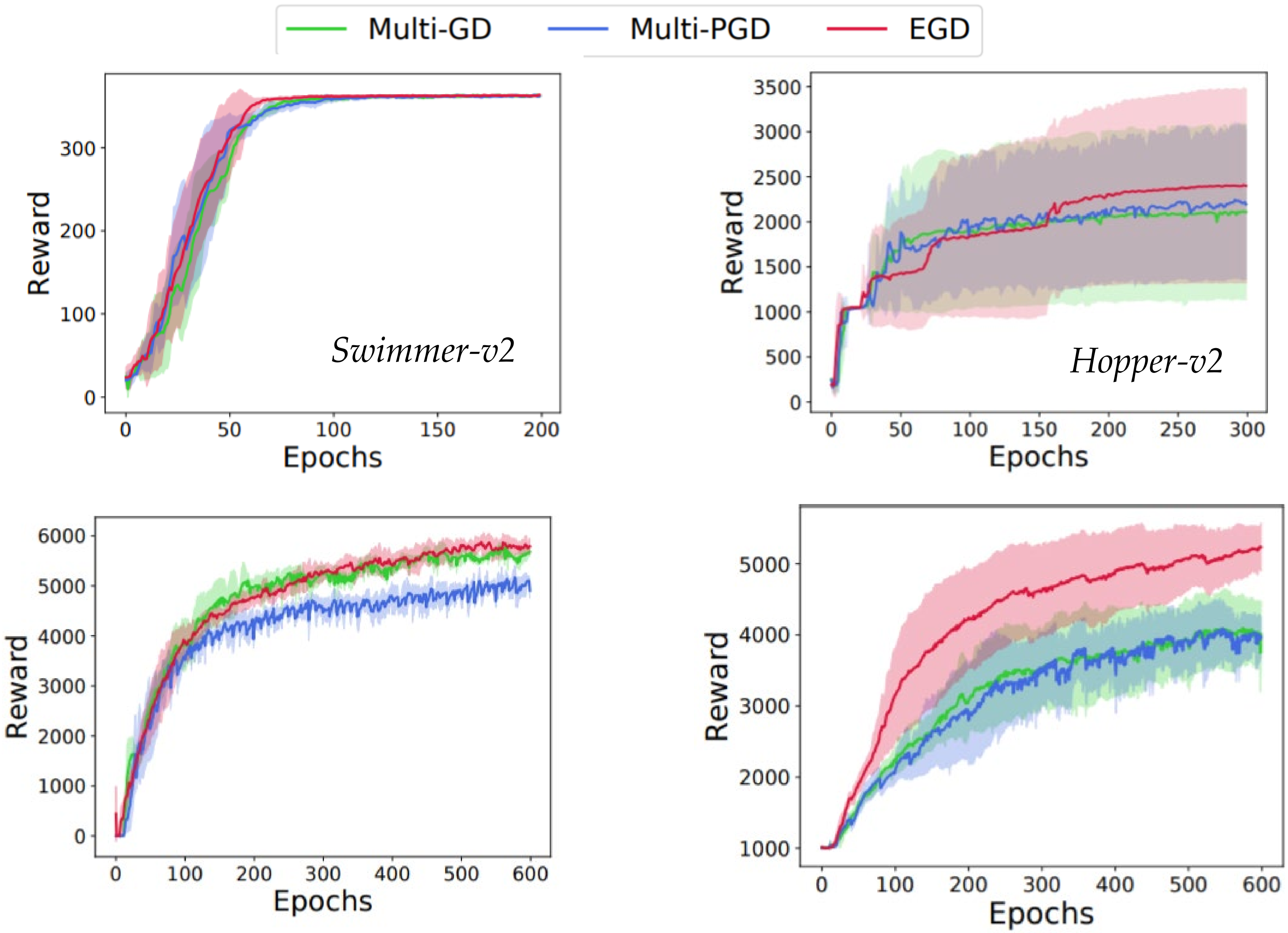

(a) *Octopus* (b) *Schwefel* (c) *Ackley*

# Experiments

| Dimension $d$ | 200 | 400 | 600 | 800 | 1000 |
|---|---|---|---|---|---|
| $f(\boldsymbol{x}) < 2$ | 1.79 | 2.05 | 2.71 | 2.78 | 2.88 |
| $f(\boldsymbol{x}) < 1$ | 1.70 | 2.04 | 2.24 | 2.30 | 2.34 |
| $f(\boldsymbol{x}) < 0.1$ | 1.51 | 1.88 | 2.10 | 2.20 | 2.24 |



# Experiments



# Conclusion and future work

- We propose a new algorithm EGD for non-convex optimization.
  - In theory, EGD can converge to a second-order stationary point more efficiently than previous algorithms.
  - In experiments, EGD shows the superior performance on non-convex optimization tasks, including synthetic benchmark functions and RL tasks.

- Future work.
  - Incorporate crossover operators into EGD.
  - Diversity of EGD.
  - Combine with advanced variants of GD.

# Thank you!

Ke Xue[1] ,Chao Qian[1*], Ling Xu[2], Xudong Fei[2]
Email: {xuek, qianc}@lamda.nju.edu.cn

[1]LAMDA Group, Nanjing University, China
[2]Huawei Technologies, China