# Black-box Optimizer with Stochastic Implicit Natural Gradient

Yueming Lyu✉ and Ivor W. Tsang

Australian Artificial Intelligence Institute
University of Technology Sydney
15 Broadway, Ultimo NSW 2007, Australia
yueminglyu@gmail.com, Ivor.Tsang@uts.edu.au

**Abstract.** Black-box optimization is primarily important for many computationally intensive applications, including reinforcement learning (RL), robot control, etc. This paper presents a novel theoretical framework for black-box optimization, in which our method performs stochastic updates with an implicit natural gradient of an exponential-family distribution. Theoretically, we prove the convergence rate of our framework with full matrix update for convex functions under Gaussian distribution. Our methods are very simple and contain fewer hyper-parameters than CMA-ES [13]. Empirically, our method with full matrix update achieves competitive performance compared with one of the state-of-the-art methods CMA-ES on benchmark test problems. Moreover, our methods can achieve high optimization precision on some challenging test functions (e.g., $l_1$-norm ellipsoid test problem and Levy test problem), while methods with explicit natural gradient, i.e., IGO [23] with full matrix update can not. This shows the efficiency of our methods.

**Keywords:** Black-box Optimization · Implicit Natural Gradient · Stochastic Optimization.

## 1 Introduction

Given a proper function $f(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$ such that $f(\boldsymbol{x}) > -\infty$, we aim at minimizing $f(\boldsymbol{x})$ by using function queries only, which is known as black-box optimization. It has a wide range of applications, such as automatic hyper-parameters tuning in machine learning and computer vision problems [27], adjusting parameters for robot control and reinforcement learning [9, 18, 7], black-box architecture search in engineering design [30] and drug discovery [21].

Several kinds of approaches have been widely studied for black-box optimization, including Bayesian optimization (BO) methods [29, 8, 20], evolution strategies (ES) [5, 13] and genetic algorithms (GA) [28]. Among them, Bayesian optimization methods are good at dealing with low-dimensional expensive black-box optimization, while ES methods are better for relatively high-dimensional problems with cheaper evaluations compared with BO methods. ES-type algorithms can well support parallel evaluation, and have drawn more and more

attention because of its success in reinforcement learning problems [11, 26, 17], recently.

CMA-ES [13] is one of state-of-the-art ES methods with many successful applications. It uses second-order information to search candidate solutions by updating the mean and covariance matrix of the likelihood of candidate distributions. Despite its successful performance, the update rule combines several sophisticated components, which is not well understood. Wierstra et al. show that directly applying standard reinforce gradient descent is very sensitive to variance in high precision search for black-box optimization [31]. Thus, they propose Natural evolution strategies (NES) [31] to estimate the natural gradient for black-box optimization. However, they use the Monte Carlo sampling to approximate the Fisher information matrix (FIM), which incurs additional error and computation cost unavoidably. Along this line, [1] show the connection between the rank-$\mu$ update of CMA-ES and NES [31]. [23] further show that several ES methods can be included in an unified framework. Despite these theoretical attempts, the practical performance of these methods is still inferior to CMA-ES. Moreover, these works do not provide any convergence rate analysis, which is the key insight to expedite black-box optimizations.

Another line of research for ES-type algorithms is to reduce the variance of gradient estimators. Choromanski et al. [11] proposed to employ Quasi Monte Carlo (QMC) sampling to achieve more accurate gradient estimates. Recently, they further proposed to construct gradient estimators based on active subspace techniques [10]. Although these works can reduce sample complexity, how does the variance of these estimators influence the convergence rate remains unclear.

To take advantage of second-order information for the acceleration of black-box optimizations, we propose a novel theoretical framework: stochastic Implicit Natural Gradient Optimization (INGO) algorithms, from the perspective of information geometry. Raskutti et al. [24] give a method to compute the Fisher information matrix implicitly using exact gradients, which is impossible for black-box optimization; while our methods and analysis focus on black-box optimization. To the best of our knowledge, we are the first to design stochastic implicit natural gradient algorithms w.r.t natural parameters for black-box optimization. Our methods take a stochastic black-box estimate instead of the exact gradient to update. Theoretically, this update is equivalent to a stochastic natural gradient step w.r.t. natural parameters of an exponential-family distribution. Our contributions are summarized as follows:

- To the best of our knowledge, we are the first to design stochastic implicit natural gradient algorithms w.r.t natural parameters for black-box optimization. We propose efficient algorithms for both continuous and discrete black-box optimization. Our methods construct stochastic black-box update without computing the FIM. Our method can adaptively control the stochastic update by taking advantage of the second-order information, which is able to accelerate convergence and is primarily important for ill-conditioned problems. Moreover, our methods have fewer hyperparameters and are much simpler than CMA-ES.

- Theoretically, we prove the convergence rate of our continuous optimization methods for convex functions. We also show that reducing variance of the black-box gradient estimators by orthogonal sampling can lead to a small regret bound.
- Empirically, our continuous optimization method achieves a competitive performances compared with the state-of-the-art method CMA-ES on benchmark problems. We find that our method with full matrix update can obtain higher optimization precision compared with IGO [23] on some challenging problems. We further show the effectiveness of our methods on RL control problems. Moreover, our discrete optimization algorithm outperforms a GA method on a benchmark problem.

## 2 Notation and Symbols

Denote $\|\cdot\|_2$ and $\|\cdot\|_F$ as the spectral norm and Frobenius norm for matrices, respectively. Define $\|Y\|_{tr} := \sum_i |\lambda_i|$, where $\lambda_i$ denotes the $i^{th}$ eigenvalue of matrix $Y$. Notation $\|\cdot\|_2$ will also denote $l_2$-norm for vectors. Symbol $\langle\cdot,\cdot\rangle$ denotes inner product under $l_2$-norm for vectors and inner product under Frobenius norm for matrices. Define $\|\boldsymbol{x}\|_C := \sqrt{\langle \boldsymbol{x}, C\boldsymbol{x}\rangle}$. Denote $\mathcal{S}^+$ and $\mathcal{S}^{++}$ as the set of positive semi-definite matrices and the set of positive definite matrices, respectively. Denote $\Sigma^{\frac{1}{2}}$ as the symmetric positive semi-definite matrix such that $\Sigma = \Sigma^{\frac{1}{2}}\Sigma^{\frac{1}{2}}$ for $\Sigma \in \mathcal{S}^+$.

## 3 Implicit Natural Gradient Optimization

### 3.1 Optimization with Exponential-family Sampling

We aim at minimizing a proper function $f(\boldsymbol{x})$, $\boldsymbol{x} \in \mathcal{X}$ with only function queries, which is known as black-box optimization. Due to the lack of gradient information for black-box optimization, we here present an exponential-family sampling trick to relax any black-box optimization problem. Specifically, the objective is relaxed as the expectation of $f(\boldsymbol{x})$ under a parametric distribution $p(\boldsymbol{x};\eta)$ with parameter $\eta$, i.e., $J(\boldsymbol{\eta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\eta})}[f(\boldsymbol{x})]$ [31]. The optimal parameter $\boldsymbol{\eta}$ is found by minimizing $J(\boldsymbol{\eta})$ as $\min_{\boldsymbol{\eta}} \left\{\mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\eta})}[f(\boldsymbol{x})]\right\}$ This relaxed problem is minimized when the probability mass is all assigned on the minimum of $f(\boldsymbol{x})$. The distribution $p$ is the sampling distribution for black-box function queries. Note, $p$ can be either continuous or discrete.

In this work, we assume that the distribution $p(\boldsymbol{x};\boldsymbol{\eta})$ is an exponential-family distribution:

$$p(\boldsymbol{x};\boldsymbol{\eta}) = h(\boldsymbol{x}) \exp\{\langle \phi(\boldsymbol{x}), \boldsymbol{\eta}\rangle - A(\boldsymbol{\eta})\}, \tag{1}$$

where $\boldsymbol{\eta}$ and $\phi(\boldsymbol{x})$ are the natural parameter and sufficient statistic, respectively. And $A(\eta)$ is the log partition function defined as $A(\eta) = \log \int \exp\{\langle \phi(\boldsymbol{x}), \boldsymbol{\eta}\rangle h(\boldsymbol{x}) d\boldsymbol{x}$.

It is named as minimal exponential-family distribution when there is a one-to-one mapping between the mean parameter $\boldsymbol{m} := \mathbb{E}_p[\phi(\boldsymbol{x})]$ and natural parameter

$\boldsymbol{\eta}$. This one-to-one mapping ensures that we can reparameterize $J(\boldsymbol{\eta})$ as $\tilde{J}(\boldsymbol{m}) = J(\boldsymbol{\eta})$ [3, 14]. $\tilde{J}$ is w.r.t parameter $\boldsymbol{m}$, while $J$ is w.r.t parameter $\boldsymbol{\eta}$.

To minimize the objective $\tilde{J}(\boldsymbol{m})$, we desire the updated distribution lying in a trust region of the previous distribution at each step. Formally, we update the mean parameters by solving the following optimization problem.

$$\boldsymbol{m}_{t+1}=\arg\min_{\boldsymbol{m}}\left\langle \boldsymbol{m},\nabla_{\boldsymbol{m}}\tilde{J}(\boldsymbol{m}_t)\right\rangle+\frac{1}{\beta_t}\mathrm{KL}\left(p_m\|p_{m_t}\right), \tag{2}$$

where $\nabla_{\boldsymbol{m}}\tilde{J}(\boldsymbol{m}_t)$ denotes the gradient at $\boldsymbol{m}=\boldsymbol{m}_t$.

The KL-divergence term measures how close the updated distribution and the previous distribution. For an exponential-family distribution, the KL-divergence term in (2) is equal to Bregman divergence between $\boldsymbol{m}$ and $\boldsymbol{m}_t$ [4]:

$$\mathrm{KL}\left(p_m\|p_{m_t}\right)=A^*(\boldsymbol{m})-A^*(\boldsymbol{m}_t)-\langle \boldsymbol{m}-\boldsymbol{m}_t,\nabla_{\boldsymbol{m}}A^*(\boldsymbol{m}_t)\rangle\,, \tag{3}$$

where $A^*(\boldsymbol{m})$ is the convex conjugate of $A(\boldsymbol{\eta})$. Thus, the problem (2) is a convex optimization problem, and it has a closed-form solution.

### 3.2 Implicit Natural Gradient

**Intractability of Natural Gradient for Black-box Optimization:** Natural gradient [2] can capture information geometry structure during optimization, which enables us to take advantage of the second-order information to accelerate convergence. Direct computation of natural gradient needs the inverse of Fisher information matrix (FIM), which needs to estimate the FIM. The method in [24] provides an alternative way to compute natural gradient without computation of FIM. However, it relies on the exact gradient, which is impossible for black-box optimization.

Hereafter, we propose a novel stochastic implicit natural gradient algorithms for black-box optimization of continuous and discrete variables in Section 4 and Section A (in the supplement), respectively. We first show how to compute the implicit natural gradient. In problem Eq.(2), we take the derivative w.r.t $\boldsymbol{m}$, and set it to zero, also note that $\nabla_{\boldsymbol{m}}A^*(\boldsymbol{m})=\boldsymbol{\eta}$ [24], we can obtain that

$$\boldsymbol{\eta}_{t+1}=\boldsymbol{\eta}_t-\beta_t\nabla_{\boldsymbol{m}}\tilde{J}(\boldsymbol{m}_t). \tag{4}$$

Natural parameters $\boldsymbol{\eta}$ of the distribution lies on a Riemannian manifold with metric tensor specified by the Fisher Information Matrix:

$$\boldsymbol{F}(\boldsymbol{\eta}):=\mathbb{E}_p\left[\nabla_\eta\log p(\boldsymbol{x};\boldsymbol{\eta})\nabla_\eta\log p(\boldsymbol{x};\boldsymbol{\eta})^\top\right]. \tag{5}$$

For exponential-family with the minimal representation, the natural gradient has a simple form for computation.

**Theorem 1.** *[15, 24] For an exponential-family in the minimal representation, the natural gradient w.r.t $\boldsymbol{\eta}$ is equal to the gradient w.r.t. $\boldsymbol{m}$, i.e.,*

$$\boldsymbol{F}(\boldsymbol{\eta})^{-1}\nabla_{\boldsymbol{\eta}}J(\boldsymbol{\eta})=\nabla_{\boldsymbol{m}}\tilde{J}(\boldsymbol{m}). \tag{6}$$

**Remark:** Theorem 1 can be easily obtained by the chain rule and the fact $\boldsymbol{F}(\boldsymbol{\eta}) = \frac{\partial^2 A(\boldsymbol{\eta})}{\partial \boldsymbol{\eta} \partial \boldsymbol{\eta}^\top}$. It enables us to compute the natural gradient implicitly without computing the inverse of the Fisher information matrix. As shown in Theorem 1, the update rule in (4) is equivalent to the natural gradient update w.r.t $\boldsymbol{\eta}$ in (7):

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \boldsymbol{F}(\boldsymbol{\eta}_t)^{-1} \nabla_\eta J(\boldsymbol{\eta}_t). \tag{7}$$

Thus, update rule in (4) selects the steepest descent direction along the Riemannian manifold induced by the Fisher information matrix as natural gradient descent. It can take the second-order information to accelerate convergence.

## 4 Update Rule for Gaussian Sampling

We first present an update method for the case of Gaussian sampling for continuous optimization. For other distributions, we can derive the update rule in a similar manner. We present update methods for discrete optimization in the supplement due to the space limitation.

For a Gaussian distribution $p := \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ with mean $\boldsymbol{\mu}$ and covariance matrix $\Sigma$, the natural parameters $\eta = \{\boldsymbol{\eta}_1, \boldsymbol{\eta}_2\}$ are given as

$$\boldsymbol{\eta}_1 := \Sigma^{-1} \boldsymbol{\mu} \tag{8}$$

$$\boldsymbol{\eta}_2 := -\frac{1}{2} \Sigma^{-1}. \tag{9}$$

The related mean parameters $\boldsymbol{m} = \{\boldsymbol{m}_1, \boldsymbol{m}_2\}$ are given as $\boldsymbol{m}_1 := \mathbb{E}_p[\boldsymbol{x}] = \boldsymbol{\mu}$ and $\boldsymbol{m}_2 := \mathbb{E}_p[\boldsymbol{x}\boldsymbol{x}^\top] = \boldsymbol{\mu}\boldsymbol{\mu}^\top + \Sigma$.

Using the chain rule, the gradient with respect to mean parameters can be expressed in terms of the gradients w.r.t $\boldsymbol{\mu}$ and $\Sigma$ [14, 16] as:

$$\nabla_{\boldsymbol{m}_1} \tilde{J}(\boldsymbol{m}) = \nabla_{\boldsymbol{\mu}} \tilde{J}(\boldsymbol{m}) - 2[\nabla_\Sigma \tilde{J}(\boldsymbol{m})]\boldsymbol{\mu} \tag{10}$$

$$\nabla_{\boldsymbol{m}_2} \tilde{J}(\boldsymbol{m}) = \nabla_\Sigma \tilde{J}(\boldsymbol{m}). \tag{11}$$

It follows that

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\beta_t \nabla_\Sigma \tilde{J}(\boldsymbol{m}_t) \tag{12}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \Sigma_{t+1} \nabla_{\boldsymbol{\mu}} \tilde{J}(\boldsymbol{m}_t). \tag{13}$$

Note that $\tilde{J}(\boldsymbol{m}) = \mathbb{E}_p[f(\boldsymbol{x})]$, the gradients of $\tilde{J}(\boldsymbol{m})$ w.r.t $\mu$ and $\Sigma$ can be obtained by log-likelihood trick as Theorem 2.

**Theorem 2.** *[31] The gradient of the expectation of an integrable function $f(\boldsymbol{x})$ under a Gaussian distribution $p := \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ with respect to the mean $\boldsymbol{\mu}$ and the covariance $\Sigma$ can be expressed as Eq.(14) and Eq.(15), respectively.*

$$\nabla_{\boldsymbol{\mu}} \mathbb{E}_p[f(\boldsymbol{x})] = \mathbb{E}_p\left[\Sigma^{-1}(\boldsymbol{x} - \boldsymbol{\mu}) f(\boldsymbol{x})\right] \tag{14}$$

$$\nabla_\Sigma \mathbb{E}_p[f(\boldsymbol{x})] = \frac{1}{2} \mathbb{E}_p\left[\left(\Sigma^{-1}(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top \Sigma^{-1} - \Sigma^{-1}\right) f(\boldsymbol{x})\right]. \tag{15}$$

Together Theorem 2 with Eq. (12) and (13), we present the update with only function queries as:

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta_t \mathbb{E}_p \left[ \left( \Sigma_t^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_t)(\boldsymbol{x}-\boldsymbol{\mu}_t)^\top \Sigma_t^{-1} - \Sigma_t^{-1} \right) f(\boldsymbol{x}) \right] \tag{16}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \Sigma_{t+1} \mathbb{E}_p \left[ \Sigma_t^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_t) f(\boldsymbol{x}) \right]. \tag{17}$$

**Remark:** Our method updates the inverse of the covariance matrix instead of the covariance matrix itself.

### 4.1 Stochastic Update

The above gradient update needs the expectation of a black-box function. However, this expectation does not have a closed-form solution. Here, we estimate the gradient w.r.t $\mu$ and $\Sigma$ by Monte Carlo sampling. Eq.(16) and (17) enable us to estimate the gradient by the function queries of $f(x)$ instead of $\nabla f(x)$. This property is very crucial for black-box optimization because gradient ($\nabla f(x)$) is not available.

Update rules using Monte Carlo sampling are given as:

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \frac{\beta_t}{N} \sum_{i=1}^{N} \left[ \left( \Sigma_t^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_t)(\boldsymbol{x}_i-\boldsymbol{\mu}_t)^\top \Sigma_t^{-1} - \Sigma_t^{-1} \right) f(\boldsymbol{x}_i) \right] \tag{18}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \frac{\beta_t}{N} \sum_{i=1}^{N} \left[ \Sigma_{t+1} \Sigma_t^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_t) f(\boldsymbol{x}_i) \right]. \tag{19}$$

To avoid the numeric ill-scaling problem, we employ a monotonic transformation $h(\cdot)$ as:

$$h(f(\boldsymbol{x}_i)) = \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{\widehat{\sigma}}. \tag{20}$$

where $\widehat{\mu}$ and $\widehat{\sigma}$ denote mean and stand deviation of function values in a batch of samples. This leads to an unbiased estimator for gradient. The update rule is given as Eq.(21) and Eq.(22). We present our black-box optimization algorithm in Alg.1.

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} \left( \Sigma_t^{-1}(\boldsymbol{x}_i-\boldsymbol{\mu}_t)(\boldsymbol{x}_i-\boldsymbol{\mu}_t)^\top \Sigma_t^{-1} \right) \tag{21}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} \Sigma_{t+1} \Sigma_t^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_t). \tag{22}$$

The update of mean $\boldsymbol{\mu}$ in Alg.1 is properly scaled by $\Sigma$. Moreover, our method updates the inverse of the covariance matrix instead of the covariance matrix itself, which provides us a stable way to update covariance independent of its scale. Thus, our method can update properly when the algorithm adaptively reduces variance for high precision search. In contrast, directly apply standard reinforce type gradient update is unstable as shown in [31].

**Algorithm 1** INGO

**Input:** Number of Samples $N$, step-size $\beta$.
**while** Termination condition not satisfied **do**
  Take i.i.d samples $\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ for $i \in \{1, \cdots N\}$.
  Set $\boldsymbol{x}_i = \boldsymbol{\mu}_t + \Sigma_t^{\frac{1}{2}} \boldsymbol{z}_i$ for $i \in \{1, \cdots N\}$.
  Query the batch observations $\{f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N)\}$
  Compute $\widehat{\sigma} = \text{std}(f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N))$.
  Compute $\widehat{\mu} = \frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{x}_i)$.
  Set $\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i)-\widehat{\mu}}{N\widehat{\sigma}} \Sigma_t^{-\frac{1}{2}} \boldsymbol{z}_i \boldsymbol{z}_i^\top \Sigma_t^{-\frac{1}{2}}$.
  Set $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i)-\widehat{\mu}}{N\widehat{\sigma}} \Sigma_{t+1} \Sigma_t^{-\frac{1}{2}} \boldsymbol{z}_i$
**end while**

### 4.2 Direct Update for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$

We provide an alternative updating equation with simple concept and derivation. The implicit natural gradient algorithms are working on the natural parameter space. Alternatively, we can also directly work on the $\boldsymbol{\mu}$ and $\Sigma$ parameter space. Formally, we derive the update rule by solving the following trust region optimization problem.

$$\boldsymbol{\theta}_{t+1} = \arg\min_{\boldsymbol{\theta}} \left\langle \boldsymbol{\theta}, \nabla_{\boldsymbol{\theta}} \bar{J}(\boldsymbol{\theta}_t) \right\rangle + \frac{1}{\beta_t} \text{KL}\left(p_\theta \| p_{\theta_t}\right), \tag{23}$$

where $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma\}$ and $\bar{J}(\boldsymbol{\theta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})] = J(\boldsymbol{\eta})$.

For Gaussian sampling, the optimization problem in (23) is a convex optimization problem. We can achieve a closed-form update given in Theorem 3:

**Theorem 3.** *For Gaussian distribution with parameter $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma\}$, problem (23) is convex w.r.t $\boldsymbol{\theta}$. The optimum of problem (23) leads to closed-form update (24) and (25):*

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\beta_t \nabla_\Sigma \bar{J}(\boldsymbol{\theta}_t) \tag{24}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \Sigma_t \nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t). \tag{25}$$

**Remark:** Comparing the update rule in Theorem 3 with Eq.(12) and (13), we can observe that the only difference is in the update of $\boldsymbol{\mu}$. In Eq.(25), the update employs $\Sigma_t$, while the update in Eq.(13) employs $\Sigma_{t+1}$. The update in Eq.(13) takes one step look ahead information of $\Sigma$,

We can obtain the black-box update for $\boldsymbol{\mu}$ and $\Sigma$ by Theorem 3 and Theorem 2. The update rule is given as follows:

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta_t \mathbb{E}_p\left[\left(\Sigma_t^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_t)(\boldsymbol{x}-\boldsymbol{\mu}_t)^\top \Sigma_t^{-1} - \Sigma_t^{-1}\right) f(\boldsymbol{x})\right] \tag{26}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \mathbb{E}_p\left[(\boldsymbol{x}-\boldsymbol{\mu}_t) f(\boldsymbol{x})\right]. \tag{27}$$

**Algorithm 2** INGOstep

**Input:** Number of Samples $N$, step-size $\beta$.
**while** Termination condition not satisfied **do**
  Take i.i.d samples $\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ for $i \in \{1, \cdots N\}$.
  Set $\boldsymbol{x}_i = \boldsymbol{\mu}_t + \Sigma_t^{\frac{1}{2}} \boldsymbol{z}_i$ for $i \in \{1, \cdots N\}$.
  Query the batch observations $\{f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N)\}$
  Compute $\widehat{\sigma} = \text{std}(f(\boldsymbol{x}_1), ..., f(\boldsymbol{x}_N))$.
  Compute $\widehat{\mu} = \frac{1}{N}\sum_{i=1}^{N} f(\boldsymbol{x}_i)$.
  Set $\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i)-\widehat{\mu}}{N\widehat{\sigma}} \Sigma_t^{-\frac{1}{2}} \boldsymbol{z}_i \boldsymbol{z}_i^\top \Sigma_t^{-\frac{1}{2}}$.
  Set $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i)-\widehat{\mu}}{N\widehat{\sigma}} \Sigma_t^{\frac{1}{2}} \boldsymbol{z}_i$
**end while**

Using the normalization transformation function $h(f(x)) = (f(x) - \widehat{\mu})/\widehat{\sigma}$, we can obtain Monte Carlo approximation update as

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \beta \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} \left( \Sigma_t^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_t)(\boldsymbol{x}_i - \boldsymbol{\mu}_t)^\top \Sigma_t^{-1} \right) \tag{28}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \sum_{i=1}^{N} \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{N\widehat{\sigma}} (\boldsymbol{x}_i - \boldsymbol{\mu}_t). \tag{29}$$

We present the algorithm in Alg.2 compared with our INGO. The only difference between INGO (Alg.1) and INGOstep (Alg.2) is the update rule of $\boldsymbol{\mu}$. INGO employs information of $\Sigma_{t+1}$, while INGOstep only uses $\Sigma_t$.

## 5 Convergence Rate

We first show a general framework for continuous optimization in Alg.3. Alg.3 employs an unbiased estimator ($\widehat{g}_t$) for gradient $\nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t)$. In contrast, it can employ both the unbiased and biased estimators $\widehat{G}_t$ for update. It is worth noting that $\widehat{g}_t$ can be both the first-order estimate (stochastic gradient) and the zeroth-order estimate (function's value-based estimator).

The update step of $\boldsymbol{\mu}$ and $\Sigma$ is achieved by solving the following convex minimization problem.

$$\boldsymbol{m}^{t+1} = \underset{\boldsymbol{m} \in \boldsymbol{\mathcal{M}}}{\arg\min}\, \beta_t \langle \boldsymbol{m}, \widehat{v}_t \rangle + \text{KL}\left(p_{\boldsymbol{m}} \| p_{\boldsymbol{m}^t}\right). \tag{30}$$

where $\boldsymbol{m} := \{\boldsymbol{m}_1, \boldsymbol{m}_2\} = \{\boldsymbol{\mu}, \Sigma + \boldsymbol{\mu}\boldsymbol{\mu}^\top\} \in \boldsymbol{\mathcal{M}}$, $\boldsymbol{\mathcal{M}}$ denotes a convex set, and $\widehat{v}_t = \{\widehat{g}_t - 2\widehat{G}_t \boldsymbol{\mu}_t, \widehat{G}_t\}$.

The optimum of problem (30) leads to closed-form update (31) and (32):

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\beta_t \widehat{G}_t \tag{31}$$

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta_t \Sigma_{t+1} \widehat{g}_t. \tag{32}$$

**General Stochastic Case:** The convergence rate of Alg.3 is shown in Theorem 4.

**Theorem 4.** *Given a convex function $f(\boldsymbol{x})$, define $\bar{J}(\boldsymbol{\theta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})]$ for Gaussian distribution with parameter $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}}\} \in \boldsymbol{\Theta}$ and $\boldsymbol{\Theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}} \big| \ \boldsymbol{\mu} \in \mathcal{R}^d, \Sigma \in \mathcal{S}^+\}$. Suppose $\bar{J}(\boldsymbol{\theta})$ be $\gamma$-strongly convex. Let $\widehat{G}_t$ be positive semi-definite matrix such that $bI \preceq \widehat{G}_t \preceq \frac{\gamma}{2} I$. Suppose $\Sigma_1 \in \mathcal{S}^{++}$ and $\|\Sigma_1\| \leq \rho$, $\mathbb{E}\widehat{g}_t = \nabla_{\boldsymbol{\mu}=\boldsymbol{\mu}_t} \bar{J}$. Assume furthermore $\|\nabla_{\Sigma=\Sigma_t} \bar{J}\|_{tr} \leq B_1$ and $\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_1\|^2_{\Sigma_1^{-1}} \leq R$, $\mathbb{E}\|\widehat{g}_t\|_2^2 \leq \mathcal{B}$ . Set $\beta_t = \beta$, then Algorithm 3 can achieve*

$$\frac{1}{T}\left[\sum_{t=1}^{T} \mathbb{E} f(\boldsymbol{\mu}_t)\right] - f(\boldsymbol{\mu}^*) \leq \frac{2bR + 2b\beta\rho(4B_1 + \beta\mathcal{B}) + 4B_1(1+\log T) + (1+\log T)\beta\mathcal{B}}{4\beta bT}$$
$$= \mathcal{O}\left(\frac{\log T}{T}\right). \tag{33}$$

**Remark:** Theorem 4 does not require the function $f(\boldsymbol{x})$ be differentiable. It holds for non-smooth function $f(\boldsymbol{x})$. Theorem 4 holds for convex function $f(\boldsymbol{x})$, as long as $\bar{J}(\boldsymbol{\theta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})]$ be $\gamma$-strongly convex. Particularly, when $f(\boldsymbol{x})$ is $\gamma$-strongly convex, we know $\bar{J}(\boldsymbol{\theta})$ is $\gamma$-strongly convex [12]. Thus, the assumption here is weaker than strongly convex assumption of $f(\boldsymbol{x})$. Moreover, Theorem 4 does not require the boundedness of the domain. It only requires the boundedness of the distance between the initialization point and an optimal point. Theorem 4 shows that the bound depends on the bound of $\mathbb{E}\|\widehat{g}_t\|_2^2$, which means that reducing variance of the gradient estimators can leads to a small regret bound.

**Black-box Case:** For black-box optimization, we can only access the function value instead of the gradient. Let $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, we give an unbiased estimator of $\nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t)$ using function values as

$$\widehat{g}_t = \Sigma_t^{-\frac{1}{2}} \boldsymbol{z} \left( f(\boldsymbol{\mu}_t + \Sigma_t^{\frac{1}{2}} \boldsymbol{z}) - f(\boldsymbol{\mu}_t) \right). \tag{34}$$

The estimator $\widehat{g}_t$ is unbiased, i.e., $\mathbb{E}[\widehat{g}_t] = \nabla_{\boldsymbol{\mu}} \bar{J}(\boldsymbol{\theta}_t)$. The proof of unbiasedness of the estimator $\widehat{g}_t$ is given in Lemma 7 in the supplement. With this estimator, we give the convergence rate of Alg.3 for convex black-box optimization as in Theorem 5.

**Theorem 5.** *For a L-Lipschitz continuous convex black box function $f(\boldsymbol{x})$, define $\bar{J}(\boldsymbol{\theta}) := \mathbb{E}_{p(\boldsymbol{x};\boldsymbol{\theta})}[f(\boldsymbol{x})]$ for Gaussian distribution with parameter $\boldsymbol{\theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}}\} \in \boldsymbol{\Theta}$ and $\boldsymbol{\Theta} := \{\boldsymbol{\mu}, \Sigma^{\frac{1}{2}} \big| \ \boldsymbol{\mu} \in \mathcal{R}^d, \Sigma \in \mathcal{S}^+\}$. Suppose $\bar{J}(\boldsymbol{\theta})$ be $\gamma$-strongly convex. Let $\widehat{G}_t$ be positive semi-definite matrix such that $b\boldsymbol{I} \preceq \widehat{G}_t \preceq \frac{\gamma}{2}\boldsymbol{I}$. Suppose $\Sigma_1 \in \mathcal{S}^{++}$ and $\|\Sigma_1\|_2 \leq \rho$, Assume furthermore $\|\nabla_{\Sigma=\Sigma_t} \bar{J}\|_{tr} \leq B_1$ and $\|\boldsymbol{\mu}^* - \boldsymbol{\mu}_1\|^2_{\Sigma_1^{-1}} \leq R$,*

**Algorithm 3** General Framework

**Input:** Number of Samples $N$, step-size $\beta$.
**while** Termination condition not satisfied **do**
  Construct unbiased estimator $\widehat{g}_t$ of gradient w.r.t $\boldsymbol{\mu}$.
  Construct unbiased/biased estimator $\widehat{G}_t \in \mathcal{S}^{++}$ such that $bI \preceq \widehat{G}_t \preceq \frac{\gamma}{2} I$
  Set $\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + 2\beta \widehat{G}_t$.
  Set $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \beta \Sigma_{t+1} \widehat{g}_t$.
**end while**

*and set $\beta_t = \beta$ and employ estimator $\widehat{g}_t$ in Eq.(34), then Algorithm 3 can achieve*

$$\frac{1}{T}\left[\sum_{t=1}^{T} \mathbb{E} f(\boldsymbol{\mu}_t)\right] - f(\boldsymbol{\mu}^*) \leq \frac{bR + b\beta\rho(4B_1 + 2\beta L^2(d+4)^2)}{2\beta bT} + \frac{4B_1(1+\log T) + (1+\log T)\beta L^2(d+4)^2}{4\beta bT} \tag{35}$$

$$= \mathcal{O}\left(\frac{d^2 \log T}{T}\right). \tag{36}$$

**Remark:** Theorem 5 holds for non-differentiable function $f(\boldsymbol{x})$. Thus, Theorem 5 can cover more interesting cases e.g. sparse black box optimization. In contrast, Balasubramanian et al. ([6]) require function $f(\boldsymbol{x})$ has Lipschitz continuous gradients.

Alg.1 employs an unbiased gradient estimator. When further ensure $b\boldsymbol{I} \preceq \widehat{G}_t \preceq \frac{\gamma}{2}\boldsymbol{I}$, Theorem 5 holds for Alg.1 Theorem 5 is derived for single sample per iteration. We can reduce the variance of estimators by constructing a set of structured samples that are conjugate of inverse covariance matrix in a batch, i.e., $\boldsymbol{z}_i \Sigma_t^{-1} \boldsymbol{z}_j = 0, i \neq j$. Particularly, when we use $\widehat{\Sigma}_t = \sigma_t \boldsymbol{I}$, sampling $N = d$ orthogonal samples [11] per iteration can lead to a convergence rate $\mathcal{O}\left(\frac{d \log T}{T}\right)$. For $N > d$ samples, we can use the method in [19] with a random rotation to reduce variance. For very large $N$, we can use the construction in Eq.(23) in [19] to transform the complex sampling matrix [32] onto sphere $\mathbb{S}^{d-1}$, then scale samples by i.i.d variables from Chi distribution. This construction has a bounded mutual coherence.

## 6 Optimization for Discrete Variable

**Binary Optimization:** For function $f(\boldsymbol{x})$ over binary variable $\boldsymbol{x} \in \{0,1\}^d$, we employ Bernoulli distribution with parameter $\boldsymbol{p} = [p_1, \cdots, p_d]^\top$ as the underlying distribution, where $p_i$ denote the probability of $x_i = 1$. Let $\boldsymbol{\eta}$ denote the natural parameter, then we know $\boldsymbol{p} = \frac{1}{1+e^{-\boldsymbol{\eta}}}$. The mean parameter is $\boldsymbol{m} = \boldsymbol{p}$.

From Eq.(4), we know that

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \nabla_{\boldsymbol{p}} \mathbb{E}_{\boldsymbol{p}}[f(\boldsymbol{x})]. \tag{37}$$

Approximate the gradient by Monte Carlo sampling, we obtain that

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \frac{1}{N} \sum_{n=1}^{N} f(\boldsymbol{x}^n) \boldsymbol{h}^n, \tag{38}$$

where $\boldsymbol{h}_i^n = \frac{1}{p_i} \mathbf{1}(\boldsymbol{x}_i^n = 1) - \frac{1}{1-p_i} \mathbf{1}(\boldsymbol{x}_i^n = 0)$.

In order to achieve stable update, we normalize function value by its mean $\widehat{\mu}$ and standard deviation $\widehat{\sigma}$ in a batch. The normalized update is given as follows:

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \sum_{n=1}^{N} \frac{f(\boldsymbol{x}^n) - \widehat{\mu}}{N\widehat{\sigma}} \boldsymbol{h}^n. \tag{39}$$

**General Discrete Optimization:** Similarly, for function $f(\boldsymbol{x})$ over discrete variable $\boldsymbol{x} \in \{1, \cdots, K\}^d$, we employ categorical distribution with parameter $\boldsymbol{P} = [\boldsymbol{p}_1, \cdots, \boldsymbol{p}_d]^\top$ as the underlying distribution, where the $ij$-th element of $\boldsymbol{P}$ ($\boldsymbol{P}_{ij}$) denote the probability of $\boldsymbol{x}_i = j$. Let $\boldsymbol{\eta} \in \mathcal{R}^{d \times K}$ denote the natural parameter, then we know $\boldsymbol{P}_{ij} = \frac{e^{\boldsymbol{\eta}_{ij}}}{\sum_{j=1}^{K} e^{\boldsymbol{\eta}_{ij}}}$. The mean parameter is $\boldsymbol{m} = \boldsymbol{P}$.

From Eq.(4), we know that

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \nabla_{\boldsymbol{P}} \mathbb{E}_{\boldsymbol{P}}[f(\boldsymbol{x})]. \tag{40}$$

Approximate the gradient by Monte Carlo sampling,

$$\boldsymbol{\eta}_{t+1} = \boldsymbol{\eta}_t - \beta_t \frac{1}{N} \sum_{n=1}^{N} f(\boldsymbol{x}^n) \boldsymbol{H}^n, \tag{41}$$

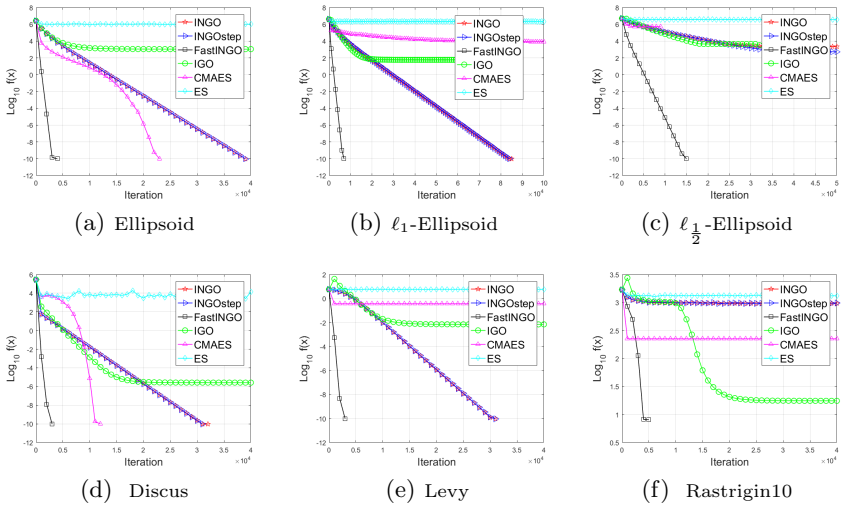
where $\boldsymbol{H}_{ij}^n = \frac{1}{\boldsymbol{P}_{ij}} \mathbf{1}(\boldsymbol{x}_i^n = j)$. We can also normalize the update by the mean $\widehat{\mu}$ and std $\widehat{\sigma}$. More detailed derivation can be found in Appendix H.

## 7 Empirical Study

### 7.1 Evaluation on synthetic continuous test benchmarks

We evaluate the proposed INGO , INGOstep and Fast-INGO (diagonal case of INGO) by comparing with one of the state-of-the-art method CMA-ES [13] and IGO [23] with full covariance matrix update, and vanilla ES with antithetic gradient estimators [26] on several synthetic benchmark test problems. All the test problems are listed in Table 1 in the supplement.

**Parameter Settings:** For INGO, INGOstep and IGO, we use the same normalization transformation $h(f(\boldsymbol{x}_i) = \frac{f(\boldsymbol{x}_i) - \widehat{\mu}}{\widehat{\sigma}}$ and all same hyper-parameters to test the effect of implicit natural gradient. We set step size $\beta = 1/d$ for all of them. For Fast-INGO, we set step size $\beta = 1/\sqrt{d}$, where $d$ is the dimension of the test problems. The number of samples per iteration is set to $N = 2\lfloor 3 + \lfloor 3 \times \ln d \rfloor / 2 \rfloor$ for all the methods, where $\lfloor \cdot \rfloor$ denotes the floor function. This setting ensures $N$ to be an even number. We set $\boldsymbol{\sigma}_1 = 0.5 \times \mathbf{1}$ and sample $\boldsymbol{\mu}_1 \sim Uni[\mathbf{0}, \mathbf{1}]$



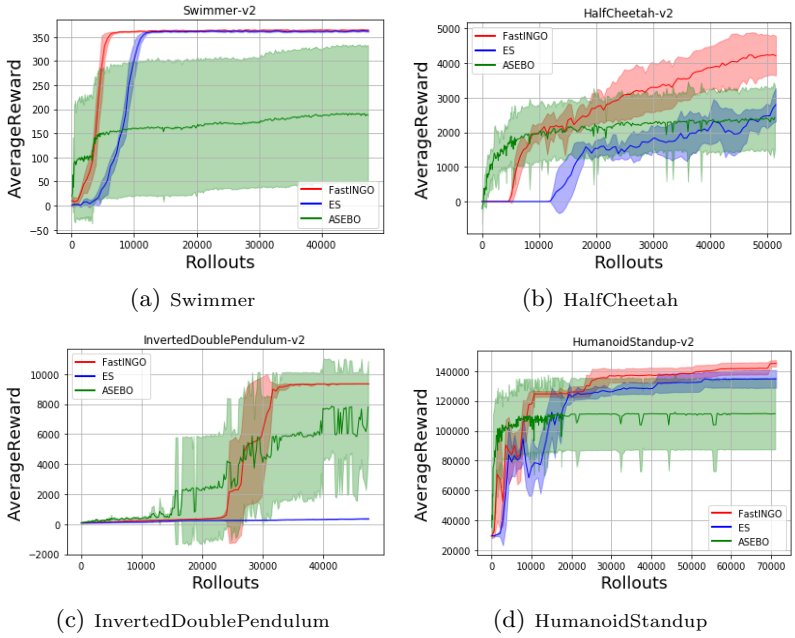

(a) Ellipsoid (b) $\ell_1$-Ellipsoid (c) $\ell_{\frac{1}{2}}$-Ellipsoid

(d) Discus (e) Levy (f) Rastrigin10

**Fig. 1.** Mean value of $f(\boldsymbol{x})$ in $\log_{10}$ scale over 20 independent runs for 100-dimensional problems.

as the same initialization for all the methods, where $Uni[0, 1]$ denotes the uniform distribution in $[0, 1]$. For ES [26], we use the default step-size hyper-parameters.

The mean value of $f(\boldsymbol{x})$ over 20 independent runs for 100-dimensional problems are show in Figure 1. From Figure 1, we can see that INGO, INGOstep and Fast-INGO converge linearly in log scale. Fast-INGO can arrive $10^{-10}$ precision on five cases except the highly non-convex Rastrigin10 problem. Fast-INGO employs the separate structure of the problems, thus it obtains better performance than the other methods with full matrix update. It is worth to note that Fast-INGO is not rotation invariant compared with Full-INGO. The INGO and INGOstep (with full matrix update) can arrive $10^{-10}$ on four cases, while IGO with full matrix update can not achieve high precision. This shows that the update of inverse of covariance matrix is more stable. Moreover, CMA-ES converge linearly in log scale for the convex Ellipsoid problem but slower than Fast-INGO. In addition, CMAES converge slowly on the non-smooth $\ell_1$-Ellipsoid and the non-convex $\ell_{\frac{1}{2}}$-Ellipsoid problem. Furthermore, CMAES fails on the non-convex Levy problem, while INGO, INGOstep and Fast-INGO obtain $10^{-10}$. CMAES converges faster or achieves smaller value than ES. On the non-convex Rastrigin10 problem, all methods fail to obtain $10^{-10}$ precision. Fast-INGO obtains smaller value. The results on synthetic test problems show that methods employing second-order information converge faster than first-order method ES. And employing second-order information is important to obtain high optimization precision, i.e., $10^{-10}$. Moreover, taking stochastic implicit natural gradient update can converge faster than IGO. The test functions are highly ill-conditioned




(a) Swimmer (b) HalfCheetah

(c) InvertedDoublePendulum (d) HumanoidStandup

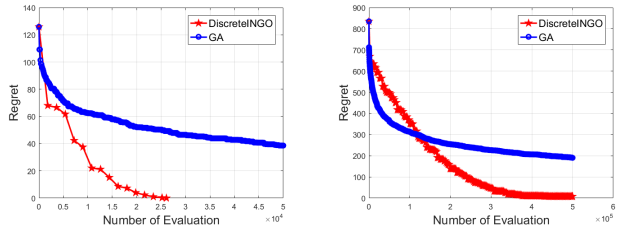
**Fig. 2.** Average Reward over 5 independent runs on benchmark RL environments

and non-convex; the experimental results show that it is challenging for ES to optimize them well without adaptively update covariance and mean.

### 7.2 Evaluation on RL test problems

We further evaluate the proposed Fast-INGO by comparing AESBO [10] and ES with antithetic gradient estimators [26] on MuJoCo control problems: Swimmer, HalfCheetah, HumanoidStandup, InvertedDoublePendulum, in Open-AI Gym environments. CMA-ES is too slow due to the computation of eigendecomposition for high-dimensional problems.

We use one hidden layer feed-forward neural network with tanh activation function as policy architecture. The number of hidden units is set to $h = 16$ for all problems. The goal is to find the parameters of this policy network to achieve large reward. The same policy architecture is used for all the methods on all test problems. The number of samples per iteration is set to $N = 20+4\lfloor\lfloor 3\times\ln d\rfloor/2\rfloor$ for all the methods. For Fast-INGO, we set step-size $\beta = 0.3$ . We set $\boldsymbol{\sigma}_1 = 0.1\times\mathbf{1}$ and $\boldsymbol{\mu}_1 = \mathbf{0}$ as the initialization for both Fast-INGO and ES. For ES [26], we use the default step-size hyper-parameters. Five independent runs are performed. The experimental results are shown in Figure 2. We can observe that Fast-INGO increase AverageReward faster than ES on all four cases. This shows that the update using seconder order information in Fast-INGO can help accelerate convergence.




(a) 100-dimensional problem (b) 500-dimensional problem

(c) 1000-dimensional problem (d) 2000-dimensional problem

**Fig. 3.** Mean value of regret over 10 independent runs for different dimensional discrete optimization problems

### 7.3 Evaluation on discrete test problems

We evaluate our discrete INGO by comparing with GA method on binary reconstruction benchmark problem, i.e., $f(\boldsymbol{x}) := \|\text{sign}(\boldsymbol{x}-0.5)-\boldsymbol{w}\|_2^2-\|\text{sign}(\boldsymbol{w})-\boldsymbol{w}\|_2^2$ with $\boldsymbol{x} \in \{0,1\}^d$ . We construct $\boldsymbol{w}$ by sampling from standard Gaussian. The dimension $d$ of test problem is set to $\{100, 500, 1000, 2000\}$, respectively. For our discrete INGO, we set the step-size $\beta = 1/d$. The number of samples per iteration is same as INGO, i.e., $N = 20 + 4\lfloor 3 + \lfloor 3 \times \ln d \rfloor /2 \rfloor$.

The experimental results are shown in Fig. 3. We can observe that our discrete INGO achieves much smaller regret compared with GA. Our discrete INGO converges to near zero regret on test problems, while GA decrease very slowly after a short initial greedy phase.

## 8 Conclusions

We proposed a novel stochastic implicit natural gradient frameworks for black-box optimization. Under this framework, we presented algorithms for both continuous and discrete black-box optimization. For Gaussian distribution, we proved the $\mathcal{O}(\log T/T)$ convergence rate of our continuous algorithms with stochastic update for convex function under expectation $\gamma$-strongly convex assumption. We proved $\mathcal{O}\left(d^2 \log T/T\right)$ converge rate for black-box function under same assumptions above. For isometric Gaussian case, we proved the $\mathcal{O}(d \log T/T)$ converge rate when using $d$ orthogonal samples per iteration, which well supports parallel evaluation. Our method is very simple, and it contains less hyper-parameters

than CMA-ES. Empirically, our methods obtain a competitive performance compared with CMA-ES. Moreover, our INGO and INGOstep with full matrix update can achieve high precision on Levy test problem and Ellipsoid problems, while IGO [23] with full matrix update can not. This shows the efficiency of our methods. On RL control problems, our algorithms increase average reward faster than ASEBO [10] and ES, which shows employing second order information can help accelerate convergence. Moreover, our discrete algorithm outperforms than GA on test functions.

## Acknowledgements

We would like to thank all anonymous reviewers and the area chair for their valuable comments and suggestions. Yueming Lyu was supported by UTS President Scholarship. Ivor Tsang was supported by the Australian Research Council Grant (DP180100106 and DP200101328).

## References

1. Akimoto, Y., Nagata, Y., Ono, I., Kobayashi, S.: Bidirectional relation between cma evolution strategies and natural evolution strategies. In: International Conference on Parallel Problem Solving from Nature. pp. 154–163. Springer (2010)
2. Amari, S.I.: Natural gradient works efficiently in learning. Neural computation **10**(2), 251–276 (1998)
3. Amari, S.i.: Information geometry and its applications, vol. 194. Springer (2016)
4. Azoury, K.S., Warmuth, M.K.: Relative loss bounds for on-line density estimation with the exponential family of distributions. Machine Learning **43**(3), 211–246 (2001)
5. Back, T., Hoffmeister, F., Schwefel, H.P.: A survey of evolution strategies. In: Proceedings of the fourth international conference on genetic algorithms. vol. 2. Morgan Kaufmann Publishers San Mateo, CA (1991)
6. Balasubramanian, K., Ghadimi, S.: Zeroth-order (non)-convex stochastic optimization via conditional gradient and gradient updates. In: Advances in Neural Information Processing Systems. pp. 3455–3464 (2018)
7. Barsce, J.C., Palombarini, J.A., Martínez, E.C.: Towards autonomous reinforcement learning: Automatic setting of hyper-parameters using bayesian optimization. In: Computer Conference (CLEI), 2017 XLIII Latin American. pp. 1–9. IEEE (2017)
8. Bull, A.D.: Convergence rates of efficient global optimization algorithms. Journal of Machine Learning Research (JMLR) **12**(Oct), 2879–2904 (2011)
9. Choromanski, K., Pacchiano, A., Parker-Holder, J., Tang, Y.: From complexity to simplicity: Adaptive es-active subspaces for blackbox optimization. arXiv:1903.04268 (2019)
10. Choromanski, K., Pacchiano, A., Parker-Holder, J., Tang, Y., Sindhwani, V.: From complexity to simplicity: Adaptive es-active subspaces for blackbox optimization (2019)
11. Choromanski, K., Rowland, M., Sindhwani, V., Turner, R.E., Weller, A.: Structured evolution with compact architectures for scalable policy optimization. In: ICML. pp. 969–977 (2018)

12. Domke, J.: Provable smoothness guarantees for black-box variational inference. arXiv preprint arXiv:1901.08431 (2019)
13. Hansen, N.: The cma evolution strategy: a comparing review. In: Towards a new evolutionary computation, pp. 75–102. Springer (2006)
14. Khan, M.E., Lin, W.: Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. arXiv preprint arXiv:1703.04265 (2017)
15. Khan, M.E., Nielsen, D.: Fast yet simple natural-gradient descent for variational inference in complex models. In: 2018 International Symposium on Information Theory and Its Applications (ISITA). pp. 31–35. IEEE (2018)
16. Khan, M.E., Nielsen, D., Tangkaratt, V., Lin, W., Gal, Y., Srivastava, A.: Fast and scalable bayesian deep learning by weight-perturbation in adam. In: ICML (2018)
17. Liu, G., Zhao, L., Yang, F., Bian, J., Qin, T., Yu, N., Liu, T.Y.: Trust region evolution strategies. In: AAAI (2019)
18. Lizotte, D.J., Wang, T., Bowling, M.H., Schuurmans, D.: Automatic gait optimization with gaussian process regression. In: IJCAI. vol. 7, pp. 944–949 (2007)
19. Lyu, Y.: Spherical structured feature maps for kernel approximation. In: Proceedings of the 34th International Conference on Machine Learning (ICML). pp. 2256–2264 (2017)
20. Lyu, Y., Yuan, Y., Tsang, I.W.: Efficient batch black-box optimization with deterministic regret bounds. arXiv preprint arXiv:1905.10041 (2019)
21. Negoescu, D.M., Frazier, P.I., Powell, W.B.: The knowledge-gradient algorithm for sequencing experiments in drug discovery. INFORMS Journal on Computing **23**(3), 346–363 (2011)
22. Nesterov, Y., Spokoiny, V.: Random gradient-free minimization of convex functions. Foundations of Computational Mathematics **17**(2), 527–566 (2017)
23. Ollivier, Y., Arnold, L., Auger, A., Hansen, N.: Information-geometric optimization algorithms: A unifying picture via invariance principles. The Journal of Machine Learning Research (JMLR) **18**(1), 564–628 (2017)
24. Raskutti, G., Mukherjee, S.: The information geometry of mirror descent. IEEE Transactions on Information Theory **61**(3), 1451–1457 (2015)
25. Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. In: ICML (2014)
26. Salimans, T., Ho, J., Chen, X., Sidor, S., Sutskever, I.: Evolution strategies as a scalable alternative to reinforcement learning. arXiv preprint arXiv:1703.03864 (2017)
27. Snoek, J., Larochelle, H., Adams, R.P.: Practical bayesian optimization of machine learning algorithms. In: NeurIPS. pp. 2951–2959 (2012)
28. Srinivas, M., Patnaik, L.M.: Genetic algorithms: A survey. computer **27**(6), 17–26 (1994)
29. Srinivas, N., Krause, A., Kakade, S.M., Seeger, M.: Gaussian process optimization in the bandit setting: No regret and experimental design. In: ICML (2010)
30. Wang, G.G., Shan, S.: Review of metamodeling techniques in support of engineering design optimization. Journal of Mechanical design **129**(4), 370–380 (2007)
31. Wierstra, D., Schaul, T., Glasmachers, T., Sun, Y., Peters, J., Schmidhuber, J.: Natural evolution strategies. The Journal of Machine Learning Research (JMLR) **15**(1), 949–980 (2014)
32. Xu, Z.: Deterministic sampling of sparse trigonometric polynomials. Journal of Complexity **27**(2), 133–140 (2011)