A Linear Time Natural Evolution Strategy for Non-Separable Functions

Yi Sun
Google Inc., USA
rosun@google.com

Tom Schaul
New York University, USA
tom@idsia.ch

Faustino Gomez
IDSIA, Manno, Switzerland
tino@idsia.ch

Jürgen Schmidhuber
IDSIA, Manno, Switzerland
juergen@idsia.ch

ABSTRACT

We present a novel Natural Evolution Strategy (NES) variant, the Rank-One NES (R1-NES), which uses a low-rank approximation of the search distribution covariance matrix. The algorithm allows computation of the natural gradient with cost linear in the dimensionality of the parameter space, and excels in solving high-dimensional non-separable problems.

Categories and Subject Descriptors

I.2 [**Computing Methodologies**]: Artificial Intelligence; G.1.6 [**Mathematics of Computing**]: Numerical Analysis—*Optimization*

Keywords

evolution strategies, covariance matrix adaptation, natural gradient

1. INTRODUCTION

The state-of-the-art continuous black-box optimization algorithms, such as xNES [2] and CMA-ES [5], are all based on the same principle: a Gaussian search distribution is repeatedly adjusted according to the objective function evaluated at sampled points. Usually the full covariance matrix is updated, allowing the distribution to adapt to the curvature of the objective function. However, this implies that the number of parameters scales *quadratically* in the number of dimensions, and the cost of updating the search distribution can become the computational bottleneck. One possible remedy is to restrict the covariance matrix to be diagonal [6], which reduces the computation per function evaluation to $O(d)$. Unfortunately, this "diagonal" approach performs poorly when the problem is non-separable.

We propose a new variant of the natural evolution strategy family [7], termed Rank One NES (R1-NES). This algorithm stays within the general NES framework in that the search distribution is adjusted according to the natural gradient [1], but it uses a novel parameterization of the covariance matrix,

$$\mathbf{\Sigma} = e^{2s}(\mathbf{I} + \mathbf{v}\mathbf{v}^\top), \tag{1}$$

where s and $\mathbf{v}$ are the parameters to be adjusted. This parameterization allows $\mathbf{v}$, the predominant eigen-direction of $\mathbf{\Sigma}$, to be oriented in any direction, enabling the algorithm to tackle non-separable problems while maintaining $O(d)$ parameters. R1-NES scales well to high dimensions, and dramatically outperforms diagonal covariance matrix algorithms on some non-separable objective functions. As an example, R1-NES reliably solves the non-convex Rosenbrock function up to 512 dimensions.

GECCO'13 Companion, July 6–10, 2013, Amsterdam, The Netherlands.
ACM 978-1-4503-1964-5/13/07.

2. THE ALGORITHM

Natural Evolution Strategy (NES) are a class of evolutionary algorithms for real-valued optimization that maintain a search distribution, and adapt the distribution parameters by following the *natural* gradient of the expected function value. At each time step, the algorithm samples n new samples $\mathbf{x}_1, \ldots, \mathbf{x}_n \sim \pi(\cdot|\theta)$, with $\pi(\cdot|\theta)$ being the search distribution parameterized by θ. Let $f : \mathbb{R}^d \mapsto \mathbb{R}$ be the objective function to maximize. The expected function value under the search distribution is

$$J(\theta) = \mathbb{E}_\theta[f(\mathbf{x})] = \int f(\mathbf{x})\pi(\mathbf{x}|\theta)\, d\mathbf{x},$$

whose natural gradient can be computed (e.g., see [7]) by

$$\nabla_\theta J = \mathbb{E}_\theta[f(\mathbf{x})\, \tilde{\nabla}_\theta L(\mathbf{x})] \approx \frac{1}{\lambda}\sum_{i=1}^{\lambda} f(\mathbf{x}_i)\, \tilde{\nabla}_\theta L(\mathbf{x}_i). \tag{2}$$

where $L(\mathbf{x}) = \log \pi(\mathbf{x}|\theta)$ is the log-likelihood, $\tilde{\nabla}_\theta \log \pi(\mathbf{x}|\theta) = \mathbf{F}_\theta^{-1}\nabla_\theta \log \pi(\mathbf{x}|\theta)$ is its natural gradient, and $\mathbf{F}$ is the Fisher information matrix.

In R1-NES, the search distribution is $\mathcal{N}(\mu, \mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is given by Eq.1. Let $\mathbf{y} \sim \mathcal{N}(0, \mathbf{I})$, $\alpha \sim \mathcal{N}(0, 1)$ and $\mathbf{w} = \mathbf{y} + \alpha\mathbf{v}$, then

$$\mathbf{x} = \boldsymbol{\mu} + e^s\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma}),$$

and the natural gradients w.r.t. μ, s and $\mathbf{v}$ are

$$\widetilde{\nabla}_\mu L(\mathbf{x}) = \mathbf{x} - \boldsymbol{\mu} \tag{3}$$

$$\widetilde{\nabla}_s L(\mathbf{x}) = \frac{1}{2(d-1)}[(\mathbf{w}^\top\mathbf{w} - d) - ((\mathbf{w}^\top\mathbf{z})^2 - 1)] \tag{4}$$

$$\widetilde{\nabla}_\mathbf{v} L(\mathbf{x}) = \frac{(r^2 - d + 2)(\mathbf{w}^\top\mathbf{z})^2 - (r^2+1)\mathbf{w}^\top\mathbf{w}}{2r(d-1)}\mathbf{z} + \frac{\mathbf{w}^\top\mathbf{z}}{r}\mathbf{w} \tag{5}$$

Here $r = \|\mathbf{v}\|$ and $\mathbf{z} = r^{-1}\mathbf{v}$. Note that computing both $\widetilde{\nabla}_\sigma L(\mathbf{x})$ and $\widetilde{\nabla}_\mathbf{v} L(\mathbf{x})$ takes $O(d)$ storage and time.

The natural gradient above is obtained with respect to $\mathbf{v}$. However, when $\widetilde{\nabla}_\mathbf{v} L(\mathbf{x})$ is large and in the opposite direction of $\mathbf{v}$, the gradient update could flip the direction of

Algorithm 1: R1-NES($\lambda, \eta, \eta_{\boldsymbol{\mu}}, \mathbf{v}$)

while *not terminate* **do**
  for $i = 1$ *to* λ **do**
    $\mathbf{y}_i \leftarrow \mathcal{N}(0, \mathbf{I})$
    $\alpha_i \leftarrow \mathcal{N}(0, 1)$
    $\mathbf{x}_i \leftarrow \boldsymbol{\mu} + e^s(\mathbf{y}_i + \alpha_i \mathbf{v})$ `//generate samples`
    evaluate $f(\mathbf{x}_i)$
  end
  compute the natural gradient for $\boldsymbol{\mu}$, s, $\mathbf{v}$, c, and $\mathbf{z}$ according to Eq.3, 4, 5, 6, and 7, and combine them using Eq.2
  $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} + \eta \widetilde{\nabla}_{\boldsymbol{\mu}} J$
  $s \leftarrow s + \eta \widetilde{\nabla}_s J$
  if $\widetilde{\nabla}_c J < 0$ **then**
    $c \leftarrow c + \eta \widetilde{\nabla}_c J$ `//multiplicative update`
    $\mathbf{z} \leftarrow \frac{\mathbf{z} + \eta \widetilde{\nabla}_{\mathbf{z}} J}{\|\mathbf{z} + \eta \widetilde{\nabla}_{\mathbf{z}} J\|}$
    $\mathbf{v} \leftarrow e^c \mathbf{z}$
  else
    $\mathbf{v} \leftarrow \mathbf{v} + \eta \widetilde{\nabla}_{\mathbf{v}} J$ `//additive update`
    $c \leftarrow \log \|\mathbf{v}\|$
    $\mathbf{z} \leftarrow \frac{\mathbf{v}}{\|\mathbf{v}\|}$
  end
end

$\mathbf{v}$ and grow its length, causing instability. A remedy is to re-parameterize $\mathbf{v} = e^c \mathbf{z}$, where $\|\mathbf{z}\| = 1$ and $e^c = r$, so that the update on c will never flip $\mathbf{v}$. The natural gradient w.r.t. c and $\mathbf{z}$ is given by

$$\widetilde{\nabla}_c L(\mathbf{x}) = r^{-1} (\widetilde{\nabla}_{\mathbf{v}} L(\mathbf{x}))^\top \mathbf{z} \tag{6}$$

$$\widetilde{\nabla}_{\mathbf{z}} L(\mathbf{x}) = r^{-1} [\widetilde{\nabla}_{\mathbf{v}} L(\mathbf{x}) - ((\widetilde{\nabla}_{\mathbf{v}} L(\mathbf{x}))^\top \mathbf{z}) \mathbf{z}], \tag{7}$$

Algorithm 1 shows the complete R1-NES algorithm in pseudocode.

3. EXPERIMENT

We show comparison between R1-NES and xNES [2], SNES on eight noise-free unimodal functions in the 'Black-Box Optimization Benchmarking' collection [4], with problem dimensions d vary from 2 to 512 (xNES was only run up to $d = 64$.) In order to make the results comparable those of other methods, the setup in [3] was used, which transforms the pure benchmark functions to make the parameters non-separable (for some) and avoid trivial optima at the origin.

Figure 1 shows eight functions where R1-NES achieves good performance. However, it must be pointed out that out of the twelve benchmark functions, R1-NES failed to solve four of them (f_2, f_7, f_{10}, f_{11}), due to the limitation in its parameterization.

4. REFERENCES

[1] S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

[2] T. Glasmaches, T. Schaul, Y. Sun, and J. Schmidhuber. Exponential natural evolution strategies. In *GECCO'10*, pages 393–400, 2010.

[3] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2010: Experimental setup. Technical Report RR-7215, INRIA, 2010.

[4] N. Hansen, S. Finck, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2010: Noiseless functions definitions. Technical Report RR-6829, INRIA, 2010.

[5] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[6] T. Schaul, T. Glasmachers, and J. Schmidhuber. High dimensions and heavy tails for natural evolution strategies. In *GECCO'11*, pages 845–852, 2011.

[7] D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber. Natural evolution strategies. In *CEC'08*, pages 3381–3387, 2008.

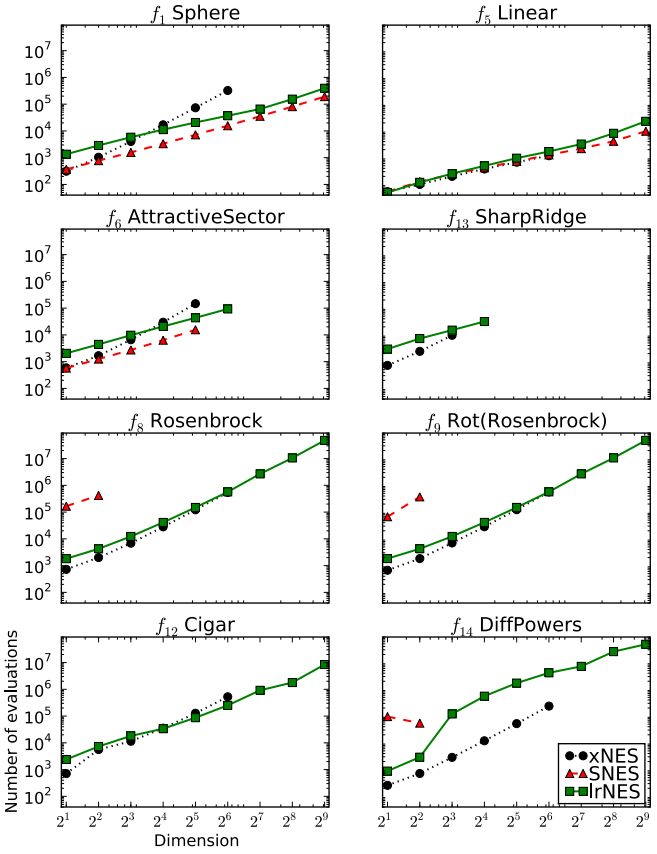

Figure 1: Performance comparison on BBOB unimodal functions. Log-log plot of the median number of evaluations (over 20 trials) required to reach the target fitness of -10^{-8} for functions for which R1-NES is well suited (cases for which 90% or more of the runs converged prematurely are not shown). Note that, with exception of f_6 and f_{13}, xNES consistently solves all benchmarks on small dimensions (≤ 64), with a scaling factor that is almost the same over all functions.