Generative Evolutionary Strategy For Black-Box Optimizations
supplementary Document

Anonymous Author(s)
Affiliation
Address
email

A Implementation Details

A.1 Neural networks

In order to make a general optimizer, the variable size should be changeable in the last layer of generator networks. In this case, a self-attention network (transformer) can be a simple choice. CNN is advantageous in making large-size output tensors using few parameters. However, CNN has too strong spatial correlations, and also, its output size is not flexible. Fully connected (FC) networks could be easier than CNN to control output tensor size, but they have too many parameters in general.

A multi-head attention structure with 8 heads and $d_{model} = 64$ was used. Position-Wise Feed Forward network (FFNN) has a hidden layer of $d_{ff} = 4 \times d_{model}$ and consists of two FC layers. For activations, we used a $hswish$ function. Dropout is set to zero in the generator to avoid randomness. Non-zero dropout generators are also tested with random feed z and age-evolution to measure the performances of stochastic generators. The self-attention network of this study is a modification of an original transformer. It is divided into a two-level structure, trunk and branches. The trunk-branch structure is ad hoc to reduce the memory usage of the network. The variable length is defined as $n_{variables} = n_{subvar} \times n_{branches}$. For example, $dimension = 8192$ can be defined as $n_{branches} = 4$ and $n_{subvar} = 2048$.

A.2 Optimization

A critic network corresponds to a single-objective target. For example, two critic networks are prepared for (f_1, f_2) of ZDT1, 2, and 3. We used $L1\ loss$ for critic network training to reduce the excessive influence of outlier data. In the experiment, 50 mutations per objective were carried out for every iteration. The pool size was fixed to 500 or $1,000$, and the buffer size was fixed to $10,000$. Mini-batch calculation can be hard to implement due to a lack of GPU memory. Instead, we carried out stochastic gradient descent (SGD).

For neural network calculations, Pytorch 1.7.1 with Python 3.7 was used. For evolution strategies, Pymoo 0.4.2 was used [52]. For a Bayesian optimization (GP), we used an optimization package (non-public) of SAMSUNG-DS.

A.3 Test functions

Scores of test functions were normalized to variable dimensions. For the global minimum of the Styblinski-Tang function, we adjusted its global minimum to 0.0 by adding 39.16617. The formula of Ackley, Rastrigin Rosenbrock, Styblinski-Tang, and ZDT functions are defined in Table 1

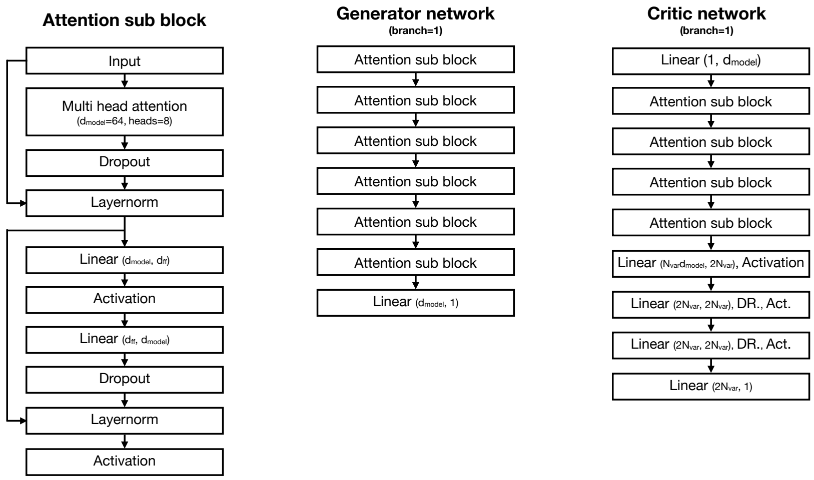

Figure 1: Neural network structures in a single-branch GEO.

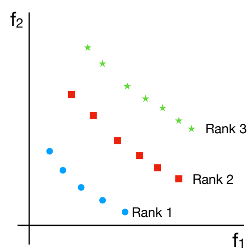

Figure 2: A schematic figure of the Pareto efficiency and ranks.

Cartpole-V1 has (*left, right*) actions for every time step. We set N dimensional real space $(x_1, x_2, ..., x_N)$, and convert it to $left$ and $right$ actions by assuming $left : x_t < 0$ and $right : 0 < x_t$. As a result, the black-box becomes a flat and discontinuous function.

LeNet-5 is a small size CNN model for image classifications. We trained LeNet-5 with MNIST dataset. We assumed predictions of LeNet-5 as scores. Therefore, a softmax function should be added in the final layer to make a score range [0, 1].

A.4 Pareto efficiency

We introduce a simple description of Pareto efficiency in the main paper. Pareto efficiency is the result of a non-dominated sorting, and it evaluates points and their corresponding multi-objective scores by giving ranks. (Figure 2) After obtaining the ranks, the sorting is done as the follow

$$(P_1, P_2, P_3, \cdots)$$

where P_i is rank-i Pareto efficiency. The importance of non-dominated sorting of multi-objective optimizations has also been studied by Tian, Ye, et al. [59].

Like common sorting algorithms, computational speed is considered important in non-dominated sorting algorithm researches. For a single-objective data sorting, there are various methods such as bubble sort, heap sort, quick sort, and Tim sort. The difference between the sorting algorithms are space complexity and time complexity, especially time complexity is considered to be important. It is known that each has advantages and disadvantages, but the quick sort method is widely used since it has $\mathcal{O}(n \log(n))$ complexity in general. Similarly, non-dominated sorting methods have their own advantages and disadvantages. So far, various kinds of methods have been developed [60], but methods based on divide-and-conquer are predominantly preferred. They have computational complexities of $\mathcal{O}(n\log(n))$. It is recommended to use a sorting algorithm with $\mathcal{O}(n\log(n))$ complexity if a very large pool size is desired. However, in this experiment, the pool size is fixed, and the size is small enough. Also, the time consumption of the sorting algorithm is much smaller than that of neural network training. Therefore, in this experiment (pool size=1,000, and function calls = 100,000 case), the choice of sorting algorithm has little effect on the speed of GEO.

Depending on the pool size and the number of mutations, the rank selection strategy of Pareto efficiency may vary. If the number of mutations is much larger than the pool size, the size of the rank-1 Pareto efficiency can be larger than the pool size. In this case, some data of rank-1 Pareto efficiency must be discarded. Therefore, it is recommended not to make too many mutations compared to the pool size. On the other hand, if the number of mutations is too small, the optimization speed may be too slow compared to the amount of network training.

A.5 Exploit & explore

We use the gradient descent technique by the surrogate model, but since this method prioritizes the exploit strategy, the explore strategy may be weak. Additional techniques can be used to supplement the explore strategy.

Typically, random mutations can be used. We can apply random noise to the parameter of the selected generator. Also, the volume of random noise can be selected in various sizes.

Another method is a mixed learning rate of a generator training. A small learning rate contributes a safe exploit strategy. On the other hand, a large learning rate can contribute to the explore strategy. By mixing small learning rates and large learning rates, we can supplement the exploit & explore strategy.

A.6 Observation cost of time-sequential problems

In real-world problems such as electronic device design and mass production processes, observation costs are often high. Sometimes the cost of observation is too high that we have to pay more than the operation of the device. Therefore, it is important to select and observe only the most valuable information. In the real-world problems, the most valuable information is often defined as follows:

$$score = \sum(rewards)$$

,where the score is observable in the final step of a simulation. We can make the most efficient observation by making only one observation at the end of the time step.

Although Cartpole-V1 does not have the observation cost problem, we used it as a toy model to describe real-world problems which have high-cost observations.

B Experimental results

B.1 Single-objective test functions

Table 2-6 shows test function optimization results in low and high dimensions. GEO outperforms other optimizers in high dimensions. However, it tends to be easily trapped at the local optimum in low dimensions. The performance degradation range strongly depends on the test function. In dimension 1,024, however, GEO always shows the best performance.

B.2 Boundary conditions

For ZDT test functions, the boundary of the search domain must be set to [0, 1]. Since the neural network has an open boundary, an additional bounded function is required as an activation of the last layer of generators. We can start with a simple bounded function, $\tanh$. In this case, $x = G(z)$ is defined as

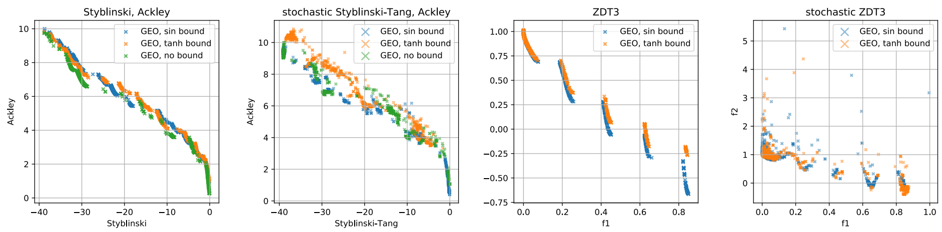

Figure 3: Pareto fronts according to tanh and sin boundary conditions. 100,000 function calls.

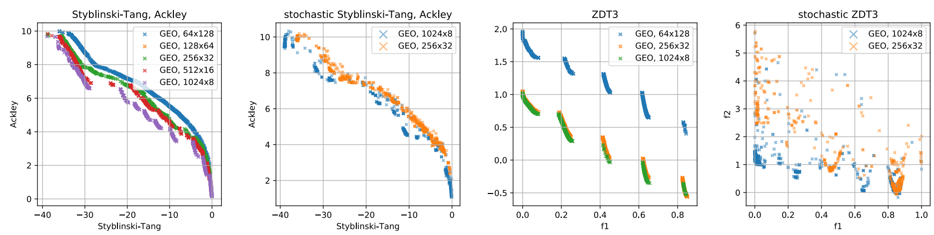

Figure 4: Pareto fronts according to the number of branches ($8,192 = n_{subvar} \times n_{branches}$). 100,000 function calls.

$$x = (bound_{max} - bound_{min}) \frac{\tanh(G(z)) + 1}{2} + bound_{min}$$

However, one problem of $\tanh$ boundary condition is a strong edge bias. With $\tanh$ function, all variables outside of a boundary ($x < bound_{min}, x > bound_{max}$) are mapped to edge of the boundary ($x = bound_{min}, x = bound_{max}$). To avoid the bias problem, we adopted $\sin$ function. Since the $\sin$ function is periodic, we can avoid the edge bias. Figure 3 shows performances according to boundary conditions. In ZDT3, $\sin$ boundary shows better performance than $\tanh$ boundary. In [Styblinski-Tang, Ackley], open boundary shows better performance than others.

B.3 Branches and pool sizes

Figure 4 shows results according to the number of branches. The dimension of variables is fixed to 8,192. As the number of branches increases, optimization performances decrease for both stochastic and non-stochastic functions. The ad hoc trunk-branch structure is memory-efficient, but it can be detrimental to the optimization performance.

Figure 5 shows results according to evolution pool sizes. They have similar performances, but a larger pool tends to find broader range of Pareto-fronts.

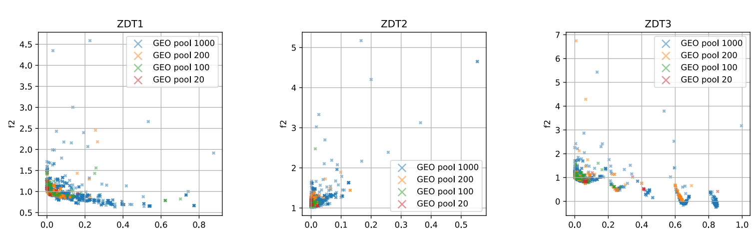

Figure 5: Pareto fronts according to the pool size. 100,000 function calls.

Table 1: Test function definitions

Name	Formula
Ackley	$-20\exp\left[-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N}x_i^2}\right]-\exp\left[\frac{1}{N}\sum_{i=1}^{N}\left(\cos 2\pi x_i\right)\right]+e+20$
Rastrigin	$A+\frac{1}{N}\sum_{i=1}^{N}\left[x_i^2-A\cos\left(2\pi x_i\right)\right]$, where: $A=10$
Rosenbrock	$\frac{1}{N-1}\sum_{i=1}^{N-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(1-x_i\right)^2\right]$
Styblinski-Tang	$\frac{1}{2N}\sum_{i=1}^{N}\left[x_i^4-16x_i^2+5x_i\right]$
ZDT functions	$f_1(x)=x_1$ $f_2(x)=g(x)h(f_1(x),g(x))$ $g(x)=1+\frac{9}{N-1}\sum_{i=2}^{N}x_i$ $0\leq x_i\leq 1$
ZDT1	$h(f_1,g)=1-\sqrt{f_1/g}$
ZDT2	$h(f_1,g)=1-(f_1/g)^2$
ZDT3	$h(f_1,g)=1-\sqrt{f_1/g}-(f_1/g)\sin(10\pi f_1)$

Table 2: Optimization results of Ackley function in low dimensions. 20,000 function calls. 10 repeats.

	Ackley			
Dimension	2	4	8	16
GEO	0.0000 ± 0.0000	0.0071 ± 0.0076	0.1009 ± 0.0432	0.3014 ± 0.2451
GEO 1-layer	0.0000 ± 0.0000	0.0030 ± 0.0023	0.8575 ± 0.7513	1.9020 ± 0.2054
NSGA-II	0.0001 ± 0.0002	0.0016 ± 0.0008	0.0073 ± 0.0033	0.0411 ± 0.0066
CMAES	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000
LSM $\epsilon 1.0$	0.0053 ± 0.0081	0.0261 ± 0.0118	0.1056 ± 0.0292	0.2036 ± 0.1139
LSM $\epsilon 0.2$	0.0006 ± 0.0003	0.6754 ± 1.3312	0.0354 ± 0.0076	0.8967 ± 0.9222
	32	64	128	
GEO	0.0694 ± 0.0576	0.0361 ± 0.0185	0.0296 ± 0.0136	
GEO 1-layer	2.7931 ± 0.1655	3.4449 ± 0.1133	3.8488 ± 0.1002	
NSGA-II	0.1132 ± 0.0163	0.2795 ± 0.0304	0.5510 ± 0.0369	
CMAES	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0001 ± 0.0000	
LSM $\epsilon 1.0$	0.2430 ± 0.1160	0.3432 ± 0.1225	0.8251 ± 0.2573	
LSM $\epsilon 0.2$	2.5080 ± 1.2227	3.4657 ± 0.3694	3.3817 ± 0.3004	

Table 3: Optimization results of Rosenbrock function in low dimensions. 20,000 function calls. 10 repeats.

	Rosenbrock			
Dimension	2	4	8	16
GEO	0.0000 ± 0.0000	0.0543 ± 0.1507	0.5090 ± 0.4545	0.5378 ± 0.5966
GEO 1-layer	0.0000 ± 0.0000	0.3062 ± 0.2330	0.8592 ± 0.2276	3.2036 ± 1.7442
NSGA-II	0.0001 ± 0.0001	0.0888 ± 0.0488	0.5197 ± 0.1102	0.8267 ± 0.0755
CMAES	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000
LSM $\epsilon 1.0$	0.3805 ± 0.3780	0.8514 ± 0.3439	1.3913 ± 0.1683	1.7246 ± 0.1924
LSM $\epsilon 0.2$	0.5814 ± 0.3466	0.5992 ± 0.3083	0.6444 ± 0.3757	0.7882 ± 0.2399
	32	64	128	
GEO	0.1705 ± 0.2817	0.0564 ± 0.0975	0.0164 ± 0.0170	
GEO 1-layer	11.3029 ± 1.3108	32.3401 ± 3.7944	61.4009 ± 4.9140	
NSGA-II	1.7519 ± 0.5589	4.0404 ± 0.4313	5.9195 ± 0.2543	
CMAES	0.6532 ± 0.0278	0.9068 ± 0.0150	0.9734 ± 0.0102	
LSM $\epsilon 1.0$	1.8820 ± 0.5490	2.1025 ± 0.7888	2.0884 ± 0.7405	
LSM $\epsilon 0.2$	1.5623 ± 0.7663	17.4755 ± 25.1455	38.9798 ± 23.4691	

Table 4: Optimization results of Rastrigin function in low dimensions. 20,000 function calls. 10 repeats.

	Rastrigin			
Dimension	2	4	8	16
GEO	0.0000 ± 0.0000	0.1027 ± 0.1635	0.1434 ± 0.2483	0.8459 ± 0.6891
GEO 1-layer	0.0000 ± 0.0000	0.3483 ± 0.1219	0.4758 ± 0.2170	1.2868 ± 0.3246
NSGA-II	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0010 ± 0.0009	0.0127 ± 0.0048
CMAES	0.2985 ± 0.3300	0.4477 ± 0.2168	0.5721 ± 0.2021	0.4166 ± 0.1446
LSM $\epsilon 1.0$	0.5375 ± 0.4406	3.6183 ± 1.9796	5.6259 ± 1.2943	5.5754 ± 1.3568
LSM $\epsilon 0.2$	0.0000 ± 0.0000	0.5076 ± 0.4908	0.3248 ± 0.3222	0.4995 ± 0.2248
	32	64	128	
GEO	1.8010 ± 1.2062	1.9690 ± 1.3727	0.8947 ± 1.1321	
GEO 1-layer	2.8961 ± 0.3151	4.9785 ± 0.2506	6.4839 ± 0.2165	
NSGA-II	0.1580 ± 0.0452	0.5013 ± 0.0478	0.9218 ± 0.0550	
CMAES	0.5006 ± 0.1526	0.5208 ± 0.0961	0.6630 ± 0.0978	
LSM $\epsilon 1.0$	5.1430 ± 1.5115	7.8261 ± 0.8673	8.8664 ± 0.6537	
LSM $\epsilon 0.2$	0.6955 ± 0.3504	4.9844 ± 2.2791	7.5184 ± 1.6464	

Table 5: Optimization results of Styblinski function in low dimensions. 20,000 function calls. 10 repeats.

	Styblinski-Tang			
Dimension	2	4	8	16
GEO	0.0000 ± 0.0000	0.0009 ± 0.0014	0.0023 ± 0.0017	0.0161 ± 0.0176
GEO 1-layer	2.1695 ± 3.2087	8.5127 ± 2.2338	15.9916 ± 1.1828	22.4661 ± 0.5406
NSGA-II	0.7068 ± 2.1205	0.7069 ± 1.4137	3.5346 ± 2.3707	5.2253 ± 0.9198
CMAES	7.0684 ± 5.4751	12.7231 ± 3.2391	10.9560 ± 2.5971	9.7190 ± 2.3708
LSM $\epsilon 1.0$	26.2230 ± 1.9996	7.6978 ± 1.2559	6.4853 ± 3.9141	9.5768 ± 4.0347
LSM $\epsilon 0.2$	27.6515 ± 2.8802	16.1667 ± 1.6806	12.1698 ± 2.0510	19.3622 ± 3.1020
	32	64	128	
GEO	0.0064 ± 0.0092	1.4138 ± 4.2410	0.0000 ± 0.0000	
GEO 1-layer	25.3427 ± 0.5938	27.5193 ± 0.6898	29.3574 ± 0.3288	
NSGA-II	7.9346 ± 1.0712	11.2910 ± 0.5126	18.3960 ± 0.3724	
CMAES	8.4820 ± 2.2159	9.9841 ± 0.6612	9.2883 ± 0.5663	
LSM $\epsilon 1.0$	9.7605 ± 4.3111	8.7600 ± 2.9911	17.5349 ± 4.3426	
LSM $\epsilon 0.2$	18.9152 ± 2.6356	32.2802 ± 2.4185	33.5116 ± 2.6515	

Table 6: Optimization results of test functions in high dimensions. 50,000 function calls. 10 repeats

	Ackley		
Dimension	256	512	1024
GEO	0.0091 ± 0.0036	0.0117 ± 0.0037	0.0084 ± 0.0029
NSGA-II	0.3294 ± 0.0219	0.7342 ± 0.0273	1.4256 ± 0.0477
CMAES	0.0000 ± 0.0000	0.0003 ± 0.0000	0.0291 ± 0.0035
	Rosenbrock		
Dimension	256	512	1024
GEO	0.0006 ± 0.0006	0.0004 ± 0.0003	0.0005 ± 0.0004
NSGA-II	5.0726 ± 0.2486	5.9915 ± 0.2358	6.9435 ± 0.1485
CMAES	0.9742 ± 0.0062	1.0011 ± 0.0337	1.0292 ± 0.0301
	Rastrigin		
Dimension	256	512	1024
GEO	0.2034 ± 0.4046	0.0018 ± 0.0020	0.0034 ± 0.0057
NSGA-II	0.5636 ± 0.0317	0.9810 ± 0.3457	1.7443 ± 0.0638
CMAES	0.9573 ± 0.1040	1.3981 ± 0.1383	3.7305 ± 0.6978
	Styblinski-Tang		
Dimension	256	512	1024
GEO	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000
NSGA-II	14.4536 ± 0.3616	22.3592 ± 0.2035	28.8298 ± 0.1541
CMAES	9.6913 ± 0.4508	9.3711 ± 0.1878	9.2048 ± 0.2450