ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

Learning Step Size Adaptation in Evolution Strategies

Oliver Kramer

Computational Intelligence Group
Department of Computing Science
University of Oldenburg
26122 Oldenburg, Germany

Abstract. Step size adaptation is an essential part of successful evolution strategies in continuous solution spaces as they moderate between exploration and exploitation. We propose to learn step sizes evolved with a σ-self-adaptive $(1 + \lambda)$-ES using LSTMs. Based on input sequences of multi-variate distances between best solutions of successive generations and their step sizes a long short-term memory network (LSTM) is trained. The learned distances-step size pairs guide the search of the LSTM-ES, which is a $(1 + \lambda)$-ES with LSTM step size predictions. An experimental analysis illustrates the behavior of the LSTM-ES on the Sphere function with different parameter settings and problem dimensionalities.

1 Introduction

For an efficient search in continuous solution spaces, the adaptation of step sizes has an important part to play. Step sizes, also known as mutation rates, parameterize mutation distributions, e.g., the width of the Gaussian distribution in evolution strategies (ES). Various methods for step size control and adaptation have been introduced for ES in the past decades, from Rechenberg's 1/5th success rule and self-adaptation, its derandomized variant to evolution path control and covariance matrix adaptation techniques.

Learning step size adaptation techniques may allow coping with varying solution space conditions. A first step towards this direction is taken in this paper. We propose to learn step size adaptation strategies with recurrent neural networks, i.e., with long short-term memory networks (LSTMs) [5], which have shown great success in learning time series. We introduce the LSTM-ES that learns sequences of distances in solution space and step size pairs observed during the run of a self-adaptive $(1 + \lambda)$-ES. Trained with this data the LSTM predicts optimal step sizes for the $(1 + \lambda)$-ES solving the target optimization problem.

This paper is structured as follows. You are reading Section 1. In Section 2 the foundations of ES and ES-based step size control are introduced. Related work is discussed in Section 3. Section 4 introduces the LSTM-ES sketching the LSTM algorithm and describing how it is integrated into ES-based search. The LSTM-ES is experimentally analyzed in Section 5. Conclusions are drawn in Section 6, where an outlook to prospective research directions is presented.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

2 $(1+\lambda)$-σ-SA-ES

ES are have been introduced by Rechenberg and Schwefel [1] and have developed to strong blackbox optimization algorithms for continuous solution spaces, i.e., to search for an optimal solution $\mathbf{x}^* \in \mathbb{R}^N$, for which holds

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \quad \forall \quad \mathbf{x} \in \mathbb{R}^N \tag{1}$$

in an N-dimensional solution space.

Employing a parental population of solutions, new offspring solutions are generated based on recombination and mutation. The mutation operator, on which we focus in this work and which is applied in almost all numerical ES-variants is Gaussian mutation. It samples from the Gaussian distribution $\mathcal{N}(\mathbf{0}, \sigma)$ with expectation vector $\mathbf{0}$ and standard deviation σ.

Algorithm 1: $(1+\lambda)$-σ-SA-ES generating training set

```
initialize x, x'
repeat
    for k ∈ {1, ..., λ} do
        σ_k ← mutate σ
        x_k ← mutate x with σ_k
    end for
    x ← select best of x_1, ..., x_λ
    σ ← select best of σ_1, ..., σ_λ
    save (log(|x − x'|), σ) in training set
    x' = x
until termination condition
```

Algorithm 1 shows the ES used to generate a training set, which is a $(1+\lambda)$-σ-SA-ES, i.e., a self-adaptive ES, which adapts step size σ automatically by means of evolution. After initialization, in each generation k step sizes σ_k are generated with log normal mutation. These are employed to generate k mutants $\mathbf{x}_k$. The best solution $\mathbf{x}$ and its corresponding step size σ are selected for the following generation. The logarithm of the absolute value of the difference between the best solution and the best $\mathbf{x}'$ of the previous generation, as well as the corresponding step size:

$$(\log(|\mathbf{x} - \mathbf{x}'|), \sigma) \tag{2}$$

is added to the training set for the LSTM-ES, see Section 4. The ES terminates after a maximum number of generations is reached.

For the step size learning process, the $(1+\lambda)$-σ-SA-ES is repeated multiple times generating the training set of distances and step sizes. For small populations sizes λ, the old parent may be the best solution leading to zero distance vectors. It is reasonable to only put patterns into the training set with positive distance vectors.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2.
Available from http://www.i6doc.com/en/.

3 Related Work

In ES step size adaptation has an important part to play [9]. Constant mutation rates and dynamic changes w.r.t. a fixed scheme lack flexibility. Adapting the step size is advantageous to allow convergence and to moderate between exploration and exploitation. A successful step size adaptation technique is the Rechenberg's 1/5th rule [1], which is based on feedback about the success rate during the search. Rechenberg's rule estimates the number of successful offspring solution and increases the step size in case of success and decreasing it in case of failure.

Closely related is self-adaptation, which lets the evolutionary search take place in the space of step sizes, while at the same time searching in objective space. Derandomized self-adaptation [11] overcomes selection noise, which may occur in case of small population sizes.

Covariance matrix adaptation evolution strategies (CMA-ES) [4] with variants [2] adapt the covariance matrix during the search and thus allow sampling from a flexible multivariate and rotated Gaussian distribution. The CMA-ES also employs the evolution path principle, which uses a record of generational information about the employed step sizes.

Natural evolution strategies (NES) [12] learn a gradient policy for the ES. The NES idea is to maximize the expectation of a solution's fitness and estimate the parameter distribution. Gradient ascent in this parameter space leads to new solution estimates. The Fisher matrix for the natural gradient has to be derived to achieve optimal NES results.

To the best of our knowledge, recurrent networks and LSTMs have not been employed for adapting step sizes yet. An approach learning step sizes like the LSTM-ES is based on reinforcement learning [10]. It employs the fitness as reward signal for policy updates while learning step sizes. Recurrent networks predict optima, e.g., in particle swarm optimization (PSO) based search [8]. The counterpart of employing ES in LSTMs is neuroevolution, where LSTMs are evolved with evolutionary algorithms [6]. The resulting networks' performances share similarities with the standard LSTM.

4 LSTM-ES

Long short-term memory networks (LSTMs) have been introduced by Schmidhuber *et al.* [5] for time series prediction. They are recurrent neural networks with hidden layers and cell states, which are vectors representing the network state and controlling the network behavior. Cell states are influenced by actions of gates. LSTMs use a forget gate, which is a small network to allow forgetting information of past hidden states. An input gate allows writing on the hidden state and selecting particular information. An update gate updates cell states and hidden states and thus influences subsequent time steps. LSTMs are trained with a backpropagation variant called backpropagation through time. Simpler LSTM variants are gated recurrent units (GRU) [3].

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

Algorithm 2: LSTM-ES

initialize $\mathbf{x}, \mathbf{x}', \mathbf{d} = \mathbf{1}$
repeat
  $\sigma \leftarrow \exp(\text{LSTM}(\mathbf{d}))$
  for $k \in \{1, \ldots, \lambda\}$ **do**
    $\mathbf{x}_k \leftarrow$ mutate $\mathbf{x}$ with σ_k
  end for
  $\mathbf{x} \leftarrow$ select best of $\mathbf{x}_1, \ldots, \mathbf{x}_\lambda$
  $\mathbf{d} = (\log(|\mathbf{x} - \mathbf{x}'|)$
  $\mathbf{x}' = \mathbf{x}$
until termination condition

The LSTM-ES is based on a trained LSTM with distance-step size pairs and applies them in an ES-like algorithmic scheme. The pseudocode is depicted in Algorithm 2. After generation of a training set with the ES introduced in Section 2 the LSTM is trained with log-distance-σ patterns. After training, the LSTM runs within the LSTM-ES. A $(1 + \lambda)$-ES can use the LSTM for prediction of optimal step sizes without using extra adaptation or self-adaptation mechanisms, but with the LSTM predictions based on occurring log-distance-σ patterns. The predicted step size is used to parameterize the Gaussian mutation. The novel solution $\mathbf{x}'$ is used as new parent, if its fitness is better than the original parent's fitness. These steps are repeated until a termination condition is reached.

The LSTM-ES can be used with any trained LSTM based on past optimization processes. We recommend using LSTMs trained with step sizes from similar problem classes.

5 Experiments

In the following, we conduct an experimental analysis of the LSTM-ES on a simple test function. We choose a typical benchmark function, i.e., the Sphere function $f(\mathbf{x}) = \mathbf{x}^T\mathbf{x}$ with $N = 2, 5$, and $N = 10$ dimensions. For $N = 2, 5$ the (1+50)-ES runs for 1,000 fitness function evaluations and generates 2-step patterns that are the basis of the training process. The LSTM uses 100 neurons for each layer. The optimal number of neurons depends on the solution space dimensions. For $N < 20$, 100 neurons turned out to be sufficient in experiments, but problem-depending adaptations are recommended. The LSTM employs a

Table 1: Overview of ES and LSTM parameters.

ES	setting	LSTM	setting
pop.	(1+50)	neurons	100
step	σ-SA	opt.	Adam
ffe	1000/10000	epochs	100

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

linear activation function, Adam [7] as optimizer, and is trained for 100 epochs. The LSTM-ES is a (1+100)-ES using the LSTM for step size prediction and running for 1,000 fitness function evaluations. ES and LSTM-ES use the initial starting point $(1.0, \ldots, 1.0)$. The experiments for $N = 10$ use 10,000 and 20,000 fitness function evaluations (ffe) for training set generation and for the LSTM-ES. The ES and LSTM parameters are summarized in Table 1.

Table 2: Experimental comparison of normal ES and LSTM-ES on the Sphere function $f(\mathbf{x}) = \mathbf{x}^T\mathbf{x}$ for $N = 2, 5, 10$.

optimizer	**N**	**rep.**	**mean**	**std**	**min**	**max**
σ-SA-ES	2	100	5.65e-06	2.57e-05	2.73e-11	2.00e-04
LSTM-ES	2	100	4.52e-07	1.20e-06	2.99e-10	4.05e-06
σ-SA-ES	5	100	6.04e-07	2.08e-06	9.06e-10	1.82e-05
LSTM-ES	5	100	3.40e-08	5.47e-08	1.33e-09	1.94e-07
σ-SA-ES	10	100	1.32e-03	1.80e-03	2.98e-05	1.04e-02
LSTM-ES	10	100	1.18e-02	1.93e-02	6.36e-05	6.09e-02
σ-SA-ES	10	200	1.23e-03	1.83e-03	3.17e-05	1.23e-02
LSTM-ES	10	200	4.95e-03	7.35e-03	6.86e-05	2.54e-02

Table 2 shows the experimental results for $N = 2, 5$ and $N = 10$. The results show that the LSTM-ES is able to learn step sizes reaching a similar quality of results like the original ES, in some cases even surpassing the performance of the original ES. The higher the dimensions, the more training data is required: for $N = 10$ 100 repetitions of the $(1 + \lambda)$-σ-SA-ES are not sufficient, but 200 are required to provide an adequate training set.

6 Conclusions

This paper proposes the LSTM-ES, an ES that employs an LSTM trained on past optimization processes for learning step size adaptation strategies. It is based on a training set of log-distances in solution space and step sizes chosen in a σ-self-adaptive ES. After training the LSTM-ES maps log-distances to step sizes for Gaussian mutations.

The experimental part has demonstrated that an LSTM-ES is able to learn efficient step size strategies on the Sphere function and reaches or even overcomes the results of a σ-SA-ES.

Future research will investigate the question, if further types of input features can be employed as extensions to sequences of log-distance patterns. Further, transfer learning from different problem classes will be investigated. For this sake a database of numerical search processes will be build up. A comparison of different step size learning approaches will complement future research.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

Acknowledgements

The authors thank the German Research Foundation (DFG) for supporting their research.

References

[1] H.-G. Beyer and H.-P. Schwefel. Evolution strategies – a comprehensive introduction. *Natural Computing*, 1(1):3–52, March 2002.

[2] H.-G. Beyer and B. Sendhoff. Covariance matrix adaptation revisited - the cmsa evolution strategy -. In *Parallel Problem Solving from Nature (PPSN)*, pages 123–132, 2008.

[3] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Gated Feedback Recurrent Neural Networks. *International Conference on Machine Learning (ICML)*, pages 2067–2075, 2015.

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

[5] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[6] R. Józefowicz, W. Zaremba, and I. Sutskever. An empirical exploration of recurrent network architectures. In *International Conference on Machine Learning (ICML)*, pages 2342–2350, 2015.

[7] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference for Learning Representations (ICLR)*, 2015.

[8] A. Meier and O. Kramer. Recurrent neural network-predictions for PSO in dynamic optimization. In *Genetic and Evolutionary Computation Conference (GECCO)*, pages 29–36, 2018.

[9] S. Meyer-Nieberg and H.-G. Beyer. Self-adaptation in evolutionary algorithms. In F. G. Lobo, C. F. Lima, and Z. Michalewicz, editors, *Parameter Setting in Evolutionary Algorithms*. Springer, Berlin, 2007.

[10] S. D. Muller, N. N. Schraudolph, and P. D. Koumoutsakos. Step size adaptation in evolution strategies using reinforcement learning. In *Congress on Evolutionary Computation (CEC)*, pages 151–156, 2002.

[11] A. Ostermeier, A. Gawelczyk, and N. Hansen. A derandomized approach to self adaptation of evolution strategies. *Evolutionary Computation*, 2(4):369–380, 1994.

[12] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber. Natural evolution strategies. *Journal of Machine Learning Research*, 15(1):949–980, 2014.