

Metaheuristic Optimization: Evolutionary Programming (EP)

Adaptive and Cooperative Algorithms (ECE 457A)

ECE, MME, and MSCI Departments,
University of Waterloo, ON, Canada

Course Instructor: Benjamin Ghogh
Fall 2023

Evolutionary Programming: the Idea

- Evolutionary Programming (EP) is one of the evolutionary algorithms. It was invented by Lawrence J. Fogel while serving at the National Science Foundation in 1960 [1, 2, 3].
- Recall genetic algorithm which has both crossover and mutation.
- EP only contains **mutation** and not crossover. It updates the candidate solutions by mutation.
- Initially, the candidate solutions are **randomly initialized** in the optimization landscape.
- Optional: In later iterations, some of the best candidate solutions are selected by **natural selection** among the candidate solutions before and after mutations. We can use any natural selection technique some of which were discussed for genetic algorithm.

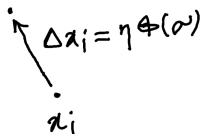
Evolutionary Programming: Mutation

- The update of a selected candidate solution $\mathbf{x}_i \in \mathbb{R}^d$ after natural selection is:

$$\mathbf{x}_i := \mathbf{x}_i + \Delta \mathbf{x}_i, \quad (1)$$

$$\Delta \mathbf{x}_i := \eta \phi(\sigma), \quad (2)$$

where η is the **scaling factor (noise factor)**, $\phi(\cdot)$ is the **scaling function**, and σ is the **strategy parameter**.



Handwritten diagram illustrating the update of a candidate solution \mathbf{x}_i . A point labeled \mathbf{x}_i is shown, and an arrow points from it to a new point, labeled $\Delta \mathbf{x}_i = \eta \phi(\sigma)$.

- Usually the change $\Delta \mathbf{x}_i$ is large in initial iterations for more **exploration** and it gets smaller gradually by iteration for having more **exploitation**.
- The larger the scaling factor η is, the more **exploration** and the less **exploitation** we have. Therefore, the scaling factor can be larger in the initial iterations and it can be decremented gradually so we have more exploration initially and more exploitation later in the iterations.

Evolutionary Programming: η

- The scaling factor (noise factor) is a stochastic variable and can be sampled from various distributions such as:

- ▶ Uniform distribution:

$$\eta \sim \mathbb{P}(\eta) = U(\eta_{\min}, \eta_{\max}). \quad (3)$$

- ▶ Gaussian distribution:

$$\eta \sim \mathcal{N}(0, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{\eta^2}{2\sigma^2}}. \quad (4)$$

- ▶ Cauchy distribution:

$$\eta \sim \mathbb{P}(\eta) = \frac{1}{\pi(1 + \eta^2)}. \quad (5)$$

Evolutionary Programming: $\phi(\cdot)$

- The scaling function can be any of the following functions:
 - ▶ Static scaling function:

$$\phi(\sigma) = \alpha\sigma, \tag{6}$$

where α is a constant scale.

- ▶ Dynamic scaling function: for example, it can be a function of the iteration index and σ .
- ▶ Self-adaptive scaling function: according to the error, it adjusts itself. For example, depending on whether the cost is getting better or worse, it adapts.

Evolutionary Programming: σ

- In general, the strategy parameter can be a function applied on elements of each candidate solution differently. The strategy parameter for the j -th dimension of the i -th candidate solution can be denoted as:

$$\sigma = \sigma_{ij}(t), \quad (7)$$

where t is the iteration index (it is a function of the iteration index).

- The strategy parameter can be any of the following functions:
 - ▶ Static strategy parameter (not very good):

$$\sigma = \sigma_{ij}(t) = \sigma_{ij}. \quad (8)$$

- ▶ Dynamic strategy parameter:

$$\sigma = \sigma_{ij}(t) = \sigma_i(t) = f(\mathbf{x}_i), \quad (9)$$

where \mathbf{x}_i is the i -th candidate solution and $f(\mathbf{x}_i)$ is the cost function at \mathbf{x}_i .

- ★ It makes sense as the update σ will be smaller if the cost is small as we need to exploit there rather than exploring.
- ★ Its flaw is that we are using absolute cost and not the relative cost value.
- ★ We can also use any increasing monotonic function of $f(\mathbf{x}_i)$, e.g., $g(f(\mathbf{x}_i))$.

Evolutionary Programming: σ

- The strategy parameter can be any of the following functions:
 - ▶ Relatively dynamic strategy parameter:

$$\sigma = \sigma_{ij}(t) = \sigma_i(t) = |f(\mathbf{x}_i) - f^*|, \quad (10)$$

where f^* is the best (smallest) cost value found so far and $|\cdot|$ is the absolute value.

- We can also specify the direction of movement (update) to be toward the best solution found so far. For this, we can multiply this direction by any of the above-mentioned update amounts; for example:

$$\sigma = \sigma_i(\mathbf{x}^* - \mathbf{x}_i), \quad \sigma_i \in (0, 1), \quad (11)$$

$$\sigma = \frac{|f(\mathbf{x}_i) - f^*|}{|f(\mathbf{x}_i) + f^*|}(\mathbf{x}^* - \mathbf{x}_i), \quad (12)$$

where \mathbf{x}^* denotes the best candidate solution found so far and $\mathbf{x}^* - \mathbf{x}_i$ is a vector connecting \mathbf{x}_i to \mathbf{x}^* .

Evolutionary Programming: Variants

- Based on the which scaling factor, scaling function, and strategy parameter we choose, we can have various variants of evolutionary programming. Some of the well-known variants are:
 - ▶ Classical Evolutionary Programming (1960) [1, 2]
 - ▶ Fast Evolutionary Programming (1996) [4]
 - ▶ Accelerated Evolutionary Programming (1996) [5]
 - ▶ Exponential Evolutionary Programming (2005-2006) [6, 7]
 - ▶ Momentum Evolutionary Programming (adds a momentum term in the update) (2012) [8]

Evolutionary Programming: Algorithm

Algorithm Evolutionary Programming

Initialize the candidate solutions $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$

while *not converged* **do**

for *each candidate solution* $\mathbf{x}_i \in \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ **do**

$\mathbf{x}_i := \mathbf{x}_i + \eta\phi(\sigma)$

if *better cost* **then**

 └ Update the solution

$\{\mathbf{x}_1, \dots, \mathbf{x}_n\} \leftarrow$ Perform natural selection [Optional]

Return the solution \mathbf{x}

Acknowledgment

- Some slides of this slide deck are inspired by teachings of Prof. Saeed Sharifian at the Amirkabir University of Technology, Department of Electrical Engineering.

References

- [1] D. B. Fogel and L. J. Fogel, "An introduction to evolutionary programming," in *European conference on artificial evolution*, pp. 21–33, Springer, 1995.
- [2] D. B. Fogel, "An overview of evolutionary programming," in *Evolutionary algorithms*, pp. 89–109, Springer, 1999.
- [3] G. B. Fogel, D. Fogel, and L. Fogel, "Evolutionary programming," *Scholarpedia*, vol. 6, no. 4, p. 1818, 2011.
- [4] X. Yao and Y. Liu, "Fast evolutionary programming," *Evolutionary programming*, vol. 3, pp. 451–460, 1996.
- [5] J.-H. Kim, H.-K. Chae, J.-Y. Jeon, and S.-W. Lee, "Identification and control of systems with friction using accelerated evolutionary programming," *IEEE Control systems magazine*, vol. 16, no. 4, pp. 38–47, 1996.
- [6] H. Narihisa, T. Taniguchi, M. Ohta, and K. Katayama, "Evolutionary programming with exponential mutation," *Proceedings of the IASTED artificial intelligence and soft computing, Benidorn, Spain*, pp. 55–50, 2005.
- [7] H. Narihisa, K. Kohmoto, T. Taniguchi, M. Ohta, and K. Katayama, "Evolutionary programming with only using exponential mutation," in *2006 IEEE International Conference on Evolutionary Computation*, pp. 552–559, IEEE, 2006.

References (cont.)

- [8] Y. Alipouri, J. Poshtan, Y. Alipouri, and M. R. Alipour, "Momentum coefficient for promoting accuracy and convergence speed of evolutionary programming," *Applied Soft Computing*, vol. 12, no. 6, pp. 1765–1786, 2012.