1 CMA-ES: Covariance Matrix Adaptation Evolution Strategy

1.1 Goal

To understand the basic idea behind CMA-ES, a powerful algorithm for finding the minimum (or maximum) of complex functions, especially when we don't know the gradient.

1.2 What are Evolutionary Algorithms (EAs)?

EAs are inspired by biological evolution. They work with a population of candidate solutions. In each "generation" (iteration):

1. Good solutions are selected (survival of the fittest).
2. New solutions are created based on the good ones (reproduction, mutation).
3. This process repeats, hopefully evolving the population towards better and better solutions.

1.3 What is CMA-ES?

The Core Idea (główna idea): Adapting the Search Distribution (dostosowywanie rozkładu wyszukiwania)

The "magic" of CMA-ES lies in how it adapts its search strategy. It uses a multivariate normal distribution (think of an ellipse or ellipsoid in multiple dimensions) to generate new candidate points.

1. **Mean** – μ
   This is the center of the distribution.
2. **Step-size** – σ
   This controls the overall (ogólny) size of (rozmiar) the search ellipse. If steps are consistently successful (konsekwentnie skuteczny), it might increase σ to explore further; if it seems to be overshooting or oscillating, it might decrease σ.

3. **Covariance Matrix – C**
   This controls the shape and orientation of the search ellipse. This is the most sophisticated (zaawansowany) part. CMA-ES learns correlations between variables. If the valley (dolina) is a long, narrow ridge (grzbietu) , the covariance matrix will adapt to make the search ellipse long and narrow and align it with (ustawi się wzdłuż) the ridge, making the search much more efficient.

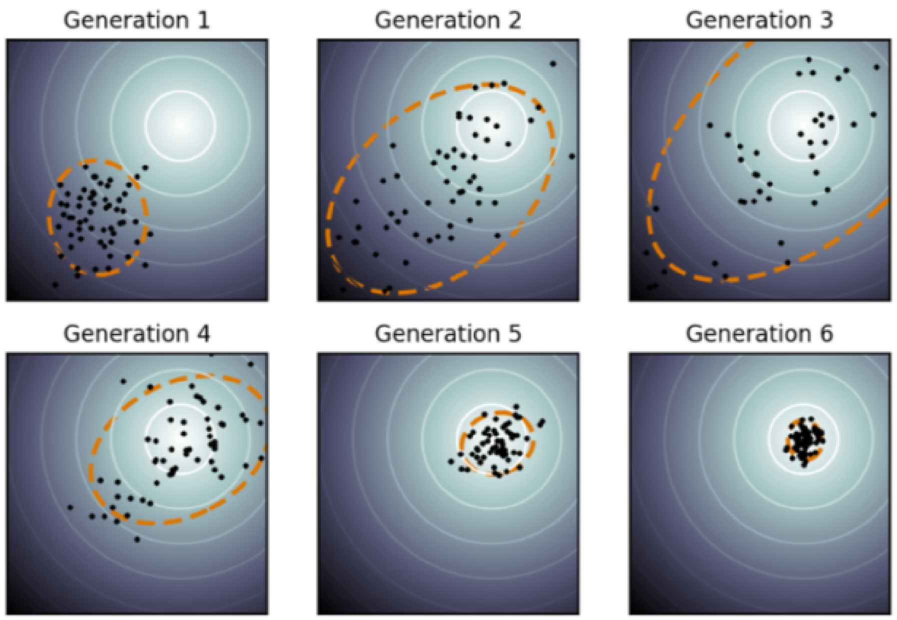
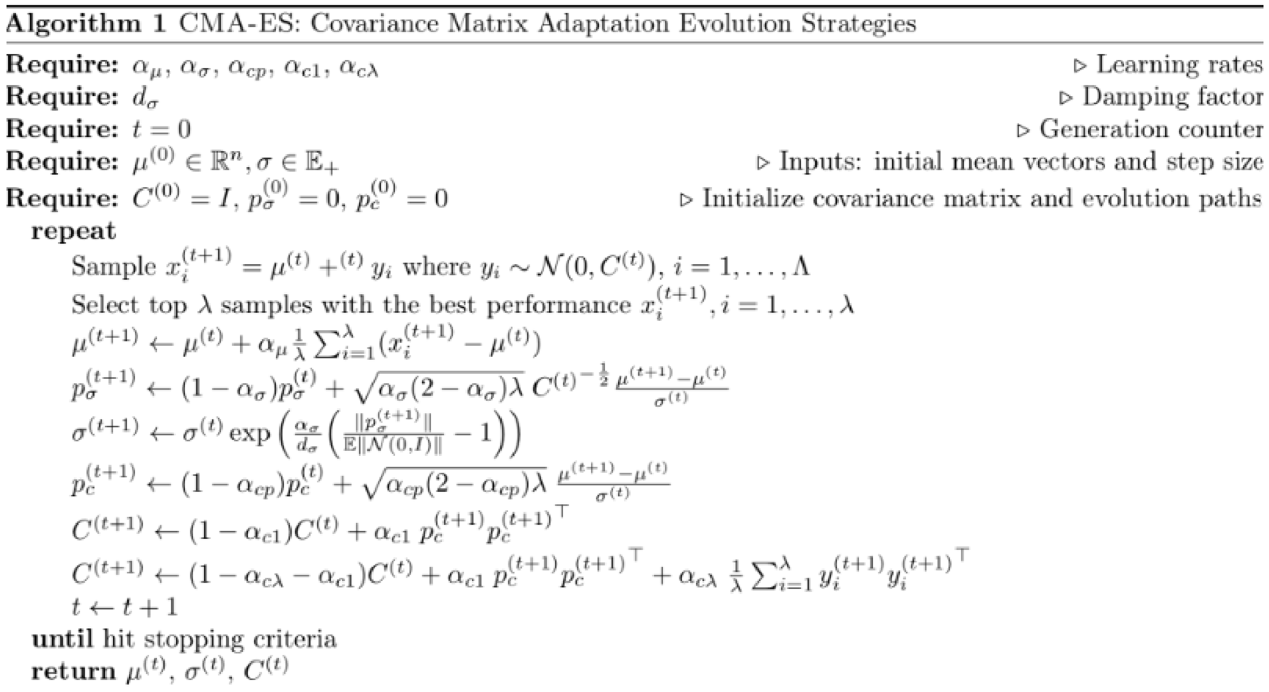


Figure 1: Visualization of how CMA-ES adapts its search distribution to the landscape of the objective function across generations.



```
Algorithm 1 CMA-ES: Covariance Matrix Adaptation Evolution Strategies
Require: $\alpha_\mu$, $\alpha_\sigma$, $\alpha_{cp}$, $\alpha_{c1}$, $\alpha_{c\lambda}$                                ▷ Learning rates
Require: $d_\sigma$                                                       ▷ Damping factor
Require: $t = 0$                                                          ▷ Generation counter
Require: $\mu^{(0)} \in \mathbb{R}^n, \sigma \in \mathbb{E}_+$                      ▷ Inputs: initial mean vectors and step size
Require: $C^{(0)} = I$, $p_\sigma^{(0)} = 0$, $p_c^{(0)} = 0$               ▷ Initialize covariance matrix and evolution paths
  repeat
    Sample $x_i^{(t+1)} = \mu^{(t)} +^{(t)} y_i$ where $y_i \sim \mathcal{N}(0, C^{(t)})$, $i = 1, \ldots, \Lambda$
    Select top $\lambda$ samples with the best performance $x_i^{(t+1)}, i = 1, \ldots, \lambda$
    $\mu^{(t+1)} \leftarrow \mu^{(t)} + \alpha_\mu \frac{1}{\lambda} \sum_{i=1}^{\lambda} (x_i^{(t+1)} - \mu^{(t)})$
    $p_\sigma^{(t+1)} \leftarrow (1 - \alpha_\sigma) p_\sigma^{(t)} + \sqrt{\alpha_\sigma (2 - \alpha_\sigma) \lambda}\, {C^{(t)}}^{-\frac{1}{2}} \frac{\mu^{(t+1)} - \mu^{(t)}}{\sigma^{(t)}}$
    $\sigma^{(t+1)} \leftarrow \sigma^{(t)} \exp\left(\frac{\alpha_\sigma}{d_\sigma}\left(\frac{\|p_\sigma^{(t+1)}\|}{\mathbb{E}\|\mathcal{N}(0,I)\|} - 1\right)\right)$
    $p_c^{(t+1)} \leftarrow (1 - \alpha_{cp}) p_c^{(t)} + \sqrt{\alpha_{cp}(2 - \alpha_{cp})\lambda}\, \frac{\mu^{(t+1)} - \mu^{(t)}}{\sigma^{(t)}}$
    $C^{(t+1)} \leftarrow (1 - \alpha_{c1}) C^{(t)} + \alpha_{c1}\, p_c^{(t+1)} {p_c^{(t+1)}}^\top$
    $C^{(t+1)} \leftarrow (1 - \alpha_{c\lambda} - \alpha_{c1}) C^{(t)} + \alpha_{c1}\, p_c^{(t+1)} {p_c^{(t+1)}}^\top + \alpha_{c\lambda}\, \frac{1}{\lambda} \sum_{i=1}^{\lambda} y_i^{(t+1)} {y_i^{(t+1)}}^\top$
    $t \leftarrow t + 1$
  until hit stopping criteria
  return $\mu^{(t)}$, $\sigma^{(t)}$, $C^{(t)}$
```

Figure 2: Image: Pseudocode of CMA-ES algorithm.

Kroki działania algorytmu CMA-ES (opis słowny)

1. Generujemy próbki z rozkładu normalnego o środku μ i kowariancji C.
2. Wybieramy najlepsze λ osobników, czyli tych, które mają najlepsze wartości funkcji celu.
3. Nowe centrum rozkładu (średnia) przesuwa się w stronę najlepszych osobników.
4. Aktualizujemy ścieżkę ewolucji p_σ, która zapamiętuje kierunek optymalizacji.
5. Na podstawie ścieżki p_σ dostosowujemy długość kroku σ (czyli jak daleko próbkujemy w kolejnych iteracjach).
6. Aktualizujemy ścieżkę ewolucji p_c, która służy do modyfikacji macierzy kowariancji.

7. Aktualizujemy macierz kowariancji C, aby dopasować kształt rozkładu do kierunków, w których osobniki miały najlepsze wyniki.

8. Zwiększamy licznik pokolenia t.

9. Sprawdzamy warunek stopu — jeśli nie jest spełniony, wracamy do kroku 1.

1.4 Strengths of CMA-ES:

- Very effective on non-linear, non-convex (niekonweksowa), rugged (chropowata), ill-conditioned problems (źle uwarunkowane problemy).
- Handles (uwzględnia) correlations between variables automatically.
- Robust to noise (odporny na szum) in the function evaluation. (ocenie funkcji)
- Requires relatively (stosunkowo) few parameter settings (defaults often work well).
- Doesn't need the function's gradient/derivative.

1.5 Weaknesses of CMA-ES:

- Computationally more expensive per generation than simpler EAs due to matrix operations (especially in high dimensions).
- Can be slower than gradient-based methods if gradients are available and reliable. (niezawodny)
- Performance can degrade significantly in very high dimensions (e.g., »100).

1.6 Recommended Reading

1. Hansen, Nikolaus. *The CMA evolution strategy: A tutorial.*

2. Lil'Log. *Evolution Strategies*

3. Such, Felipe Petroski, et al. *Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning.*

4. OpenAI. *Evolution strategies as a scalable alternative to reinforcement learning.*