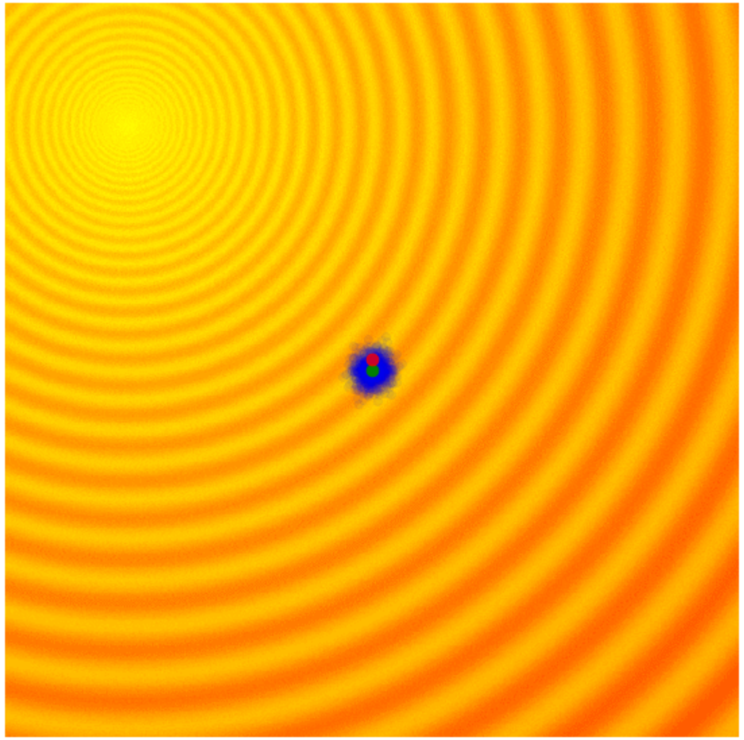
ROBOT INTELLIGENCE LAB

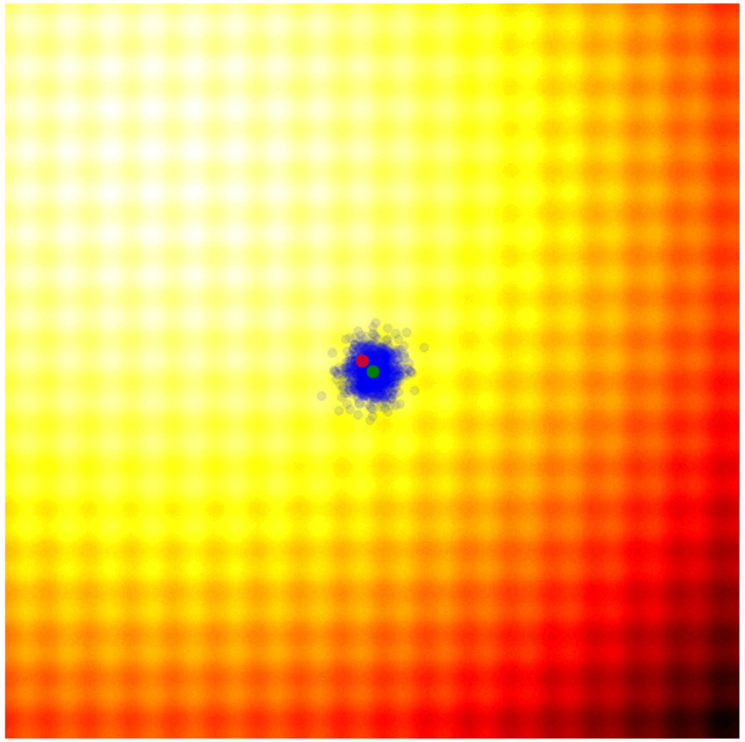
Reinforcement Learning

Lecture 5. Population-based Methods

Sungjoon Choi, Korea University

Population-based Search



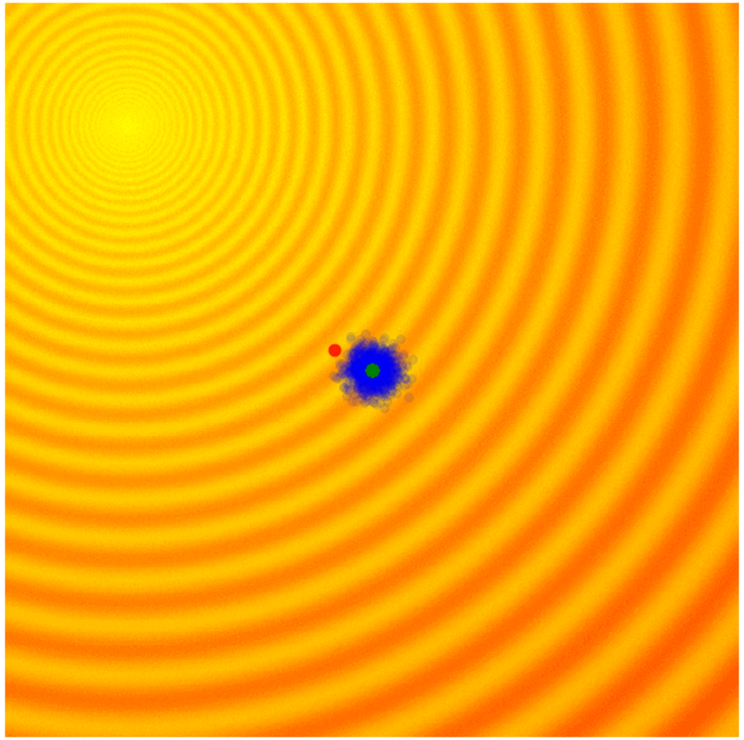


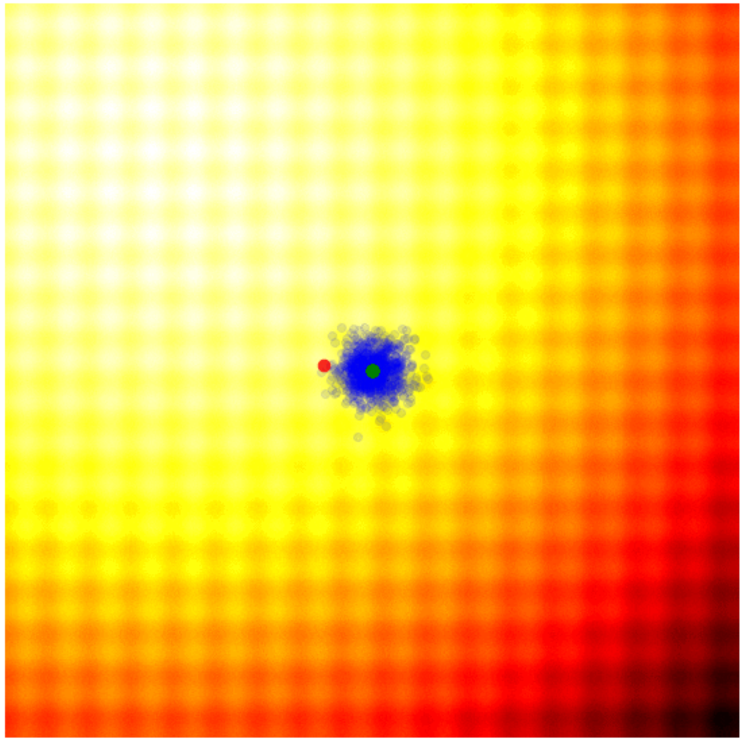
- The green dot indicates the mean of the distribution at each generation, the blue dots are the sampled solutions, and the red dot is the best solution so far.

https://blog.otoro.net/2017/10/29/visual-evolution-strategies/

Cross Entropy Method (CEM)

Genetic Algorithm





- The green dot indicates the **elite population** from the previous generation, the blue dots are the offsprings to form the set of candidate solutions, and the red dot is the best solution so far.

https://blog.otoro.net/2017/10/29/visual-evolution-strategies/

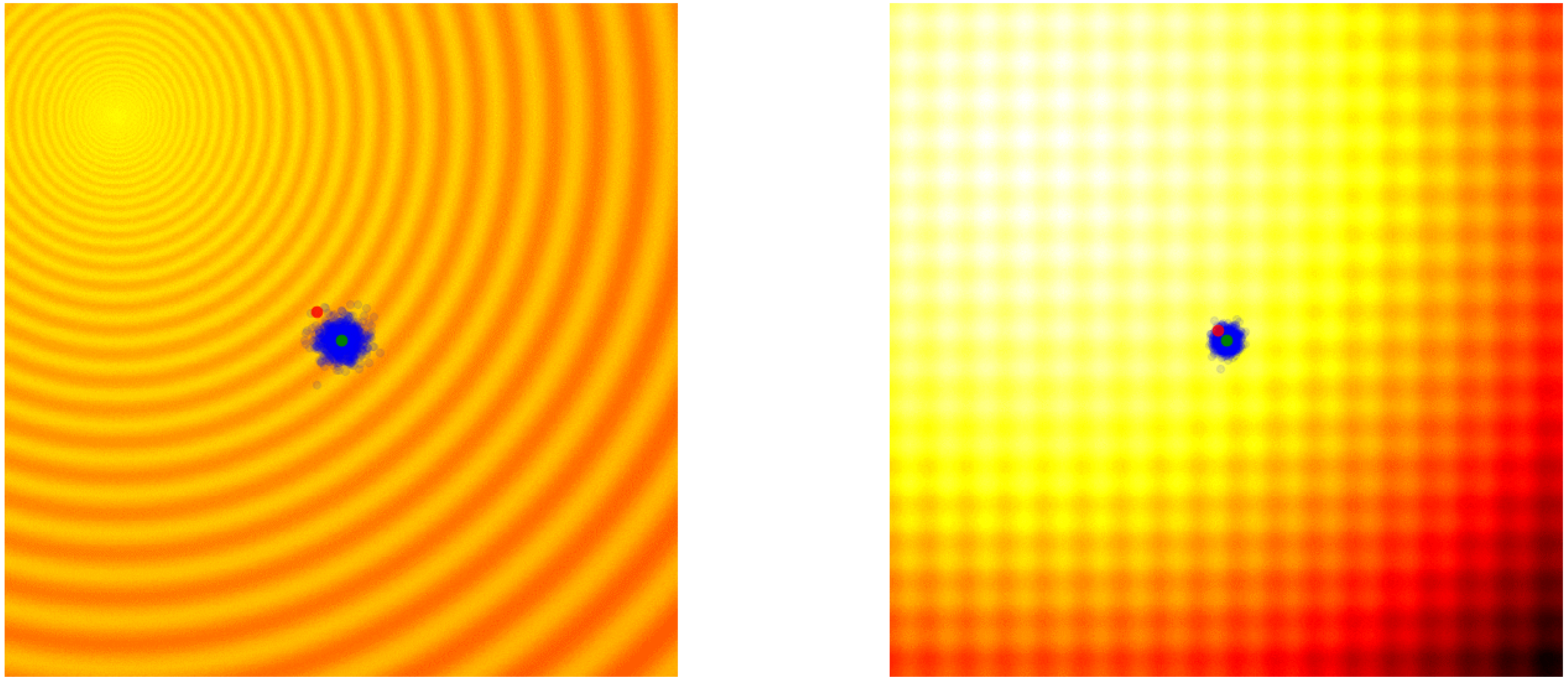
Cross Entropy Method (CEM)

- Initialize $\mu \in \mathbb{R}^d$ and $\sigma \in \mathbb{R}^d$
- **for** iteration = 1, 2, …
  - Collect n samples of $\theta_i \sim \mathcal{N}(\mu, \mathrm{diag}(\sigma))$
  - Perform a noisy evaluation $R_i \sim \theta_i$
  - Select the top $p\,\%$ of samples (aka the **elite set**)
  - Fit a Gaussian distribution (with diagonal covariance) to the **elite set**, obtaining a new μ and σ
- **end for**
- Return the final μ

CMA-ES

"The CMA Evolution Strategy: A Tutorial," 2016

CMA-ES



- The green dot indicates the **elite population** from the previous generation, the blue dots are the offsprings to form the set of candidate solutions, and the red dot is the best solution so far.
- CMA-ES can **adaptively** increase or decrease the search space!

https://blog.otoro.net/2017/10/29/visual-evolution-strategies/

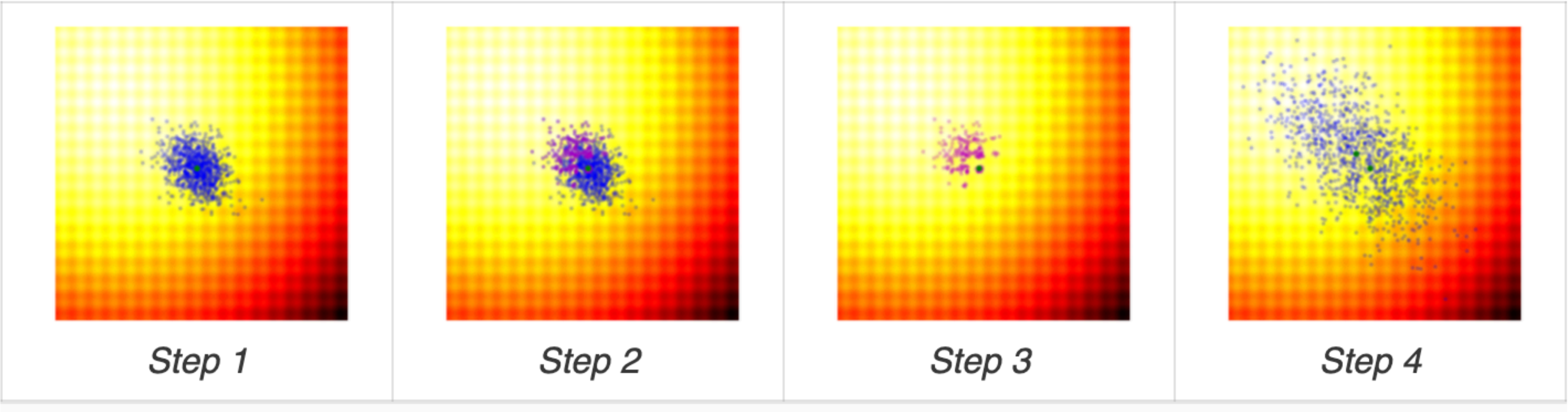
CMA-ES

ROBOT INTELLIGENCE LAB

- Initialize $\mu \in \mathbb{R}^d$ and $\sigma \in \mathbb{R}^d$
- **for** iteration g = 1, 2, …
  - Collect n samples of $\theta_i \sim \mathcal{N}(\mu, \text{diag}(\sigma))$
  - Perform a noisy evaluation $R_i \sim \theta_i$
  - Select the top $p\,\%$ of samples (aka the **elite set**)
  - Fit $\mu^{(g)}$ to the **elite set** by $\mu^{(g)} = \frac{1}{N_{\text{best}}} \sum_{i}^{N_{\text{best}}} \theta_i$
  - The trick is to update the covariance using the previous mean $\mu^{(g-1)}$.
    - For 1-dimensional case, $\sigma_x^{2,(g+1)} = \frac{1}{N_{\text{best}}} \sum_{i=1}^{N_{\text{best}}} \left(x_i - \mu_x^{(g)}\right)^2$
- **end for**
- Return the final μ

https://blog.otoro.net/2017/10/29/visual-evolution-strategies/

The Effect of CMA



https://blog.otoro.net/2017/10/29/visual-evolution-strategies/

Augmented Random Search (ARS)

"Simple random search provides a competitive approach to reinforcement learning," 2018

Augmented Random Search

- Augmented Random Search (ARS) is a population-based, derivative-free reinforcement learning method.
  - Derivative-free methods such as a **cross-entropy method** (CEM) or **covariance matrix adaptation** (CMA) treat the return as as a black box function to be optimized in terms of the policy parameters.
  - From the current policy parameter θ^t, multiple parameters $\{\tilde{\theta}_i^t\}_{i=1}^N$ are sampled and the returns $\{\eta(\pi_{\tilde{\theta}_i^t})\}_{i=1}^N$ of the corresponding policies $\{\pi_{\tilde{\theta}_i^t}\}_{i=1}^N$ evaluated by rollouts.
- ARS utilizes a simple linear policy and searches over the space of matrices.

Basic Random Search

Algorithm 1 Basic Random Search (BRS)

Hyperparameters: step-size α, number of directions sampled per iteration N, standard deviation of the exploration noise ν

Initialize: $\theta_0 = \mathbf{0}$, and $j = 0$.

while ending condition not satisfied **do**

  Sample $\delta_1, \delta_2, \ldots, \delta_N$ of the same size as θ_j, with i.i.d. standard normal entries.

  Collect $2N$ rollouts of horizon H and their corresponding rewards using the policies

$$\underbrace{\pi_{j,k,+}(x) = \pi_{\boxed{\theta_j+\nu\delta_k}}(x)}_{\text{Positively perturbed policy}} \quad \text{and} \quad \underbrace{\pi_{j,k,-}(x) = \pi_{\boxed{\theta_j-\nu\delta_k}}(x)}_{\text{Negatively perturbed policy}},$$

  with $k \in \{1, 2, \ldots, N\}$.

  Make the update step:

$$\theta_{j+1} = \theta_j + \overset{\text{Stepsize}}{\boxed{\frac{\alpha}{N}}} \sum_{k=1}^{N} \overset{\text{Score of the perturbation}}{\boxed{[r(\pi_{j,k,+}) - r(\pi_{j,k,-})]}} \underset{\text{Perturbation}}{\boxed{\delta_k}}.$$

  $j \leftarrow j + 1$.

end while

Augmented Random Search

- Drawbacks of Basic Random Search
  - Input normalization is often necessary for high-dimensional inputs.
  - Not all perturbations (directions) are useful.

Augmented Random Search

ROBOT INTELLIGENCE LAB

Algorithm 2 Augmented Random Search (ARS): four versions **V1**, **V1-t**, **V2** and $\boxed{\textbf{V2-t}}$

Hyperparameters: step-size α, number of directions sampled per iteration N, standard deviation of the exploration noise ν, $\boxed{\text{number of top-performing directions to use } b}$ ($b < N$ is allowed only for **V1-t** and **V2-t**)

Use top b search directions.

Initialize: $M_0 = \mathbf{0} \in \mathbb{R}^{p\times n}$, $\mu_0 = \mathbf{0} \in \mathbb{R}^n$, and $\Sigma_0 = \mathbf{I}_n \in \mathbb{R}^{n\times n}$, $j = 0$.

while ending condition not satisfied **do**

Sample $\delta_1, \delta_2, \dots, \delta_N$ in $\mathbb{R}^{p\times n}$ with i.i.d. standard normal entries.

Collect $2N$ rollouts of horizon H and their corresponding rewards using the $2N$ policies

$$\textbf{V1:}\quad \begin{cases} \pi_{j,k,+}(x) = (M_j + \nu\delta_k)x \\ \pi_{j,k,-}(x) = (M_j - \nu\delta_k)x \end{cases}$$

$$\textbf{V2:}\quad \begin{cases} \pi_{j,k,+}(x) = (M_j + \nu\delta_k)\boxed{\operatorname{diag}(\Sigma_j)^{-1/2}(x-\mu_j)} \\ \pi_{j,k,-}(x) = (M_j - \nu\delta_k)\boxed{\operatorname{diag}(\Sigma_j)^{-1/2}(x-\mu_j)} \end{cases}$$

Input normalization

for $k \in \{1, 2, \dots, N\}$.

$\boxed{\text{Sort the directions } \delta_k \text{ by } \max\{r(\pi_{j,k,+}), r(\pi_{j,k,-})\}}$ denote by $\delta_{(k)}$ the k-th largest direction, and by $\pi_{j,(k),+}$ and $\pi_{j,(k),-}$ the corresponding policies.

Make the update step:

$$M_{j+1} = M_j + \frac{\alpha}{\boxed{b}\boxed{\sigma_R}}\sum_{k=1}^{\boxed{b}}\left[r(\pi_{j,(k),+}) - r(\pi_{j,(k),-})\right]\delta_{(k)},$$

where $\boxed{\sigma_R \text{ is the standard deviation of the } 2b \text{ rewards}}$ used in the update step.

$\boxed{\textbf{V2}: \text{Set } \mu_{j+1}, \Sigma_{j+1} \text{ to be the mean and covariance of the } 2NH(j+1) \text{ states encountered from the start of training.}^2}$

$j \leftarrow j + 1$

end while

ROBOT INTELLIGENCE LAB