University of Paris Sud 11, France
Department of computer science
Machine learning, information and contents (AIC)
Stochastic optimization class

# Theoretical foundation for covariance matrix adaptation evolutionary strategy (CMA-ES) from information geometry perspective

Presented by : Ahmed MAZARI

20 February 2017

# Context and motivation

**Context :**

Continuous domain optimization

**Target :**

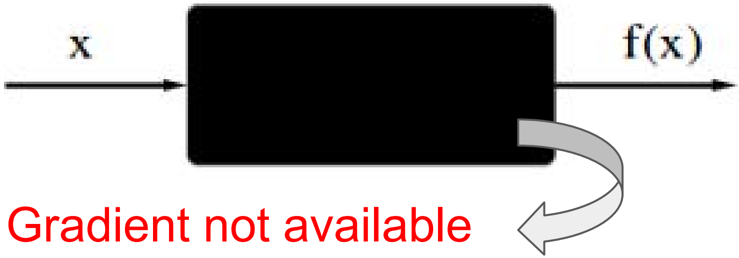
Minimize an objective function in a continuous domain

$$f: \mathbb{X} \subseteq \mathbb{R}^n \longrightarrow \mathbb{R}, \qquad x \longmapsto f(x)$$

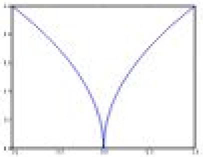
# Context and motivation

**Black box scenario :**

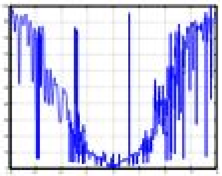
How to minimize f in a black-box scenario ?
Why it's difficult to solve ?



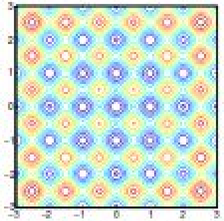

- Gradient not available
- Non-convex, noisy, rugged, high dimensional
- Analytic form is not known
- Time evaluation ...



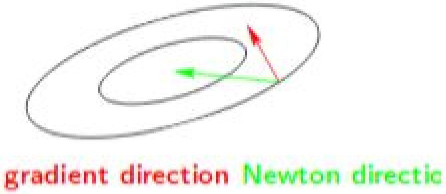
non-convex



ruggedness



non-separability




ill-conditioning

# Context and motivation

**Goal :**

Convergence to the global optimum.

**As fast as possible**

**Problem :**

- Exhaustive search is infeasible
- Naive random and deterministic  search take too long time

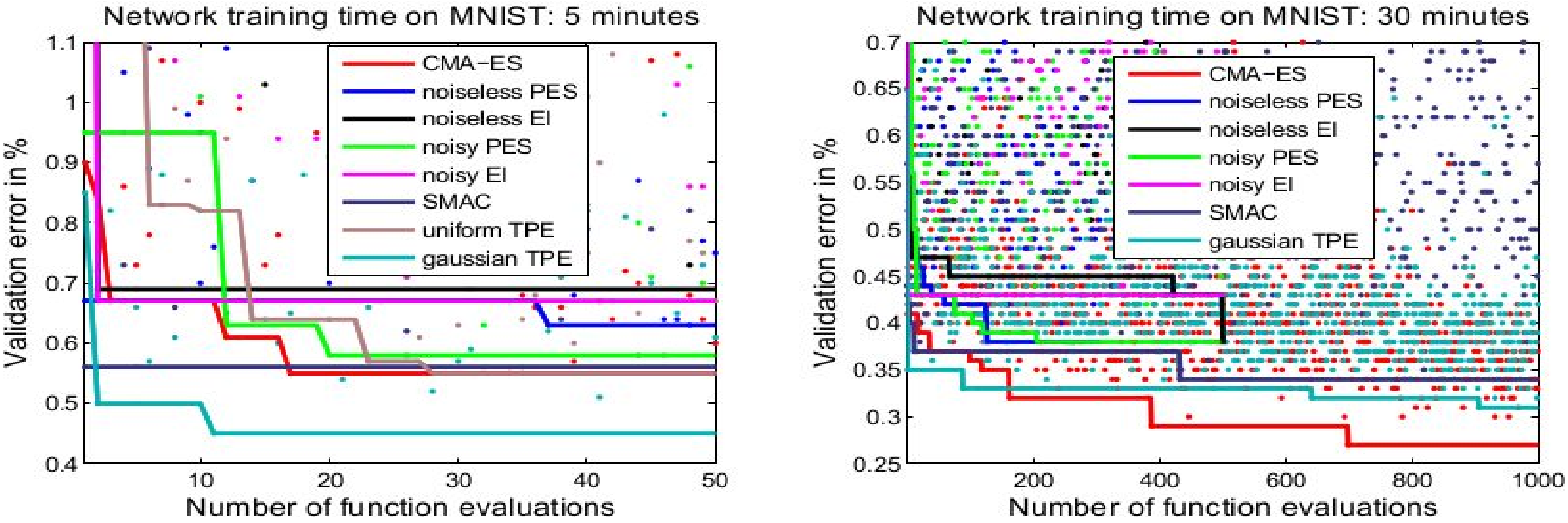
# CMA-ES for deep learning hyperparameter optimization

| name | description | transformation | range |
|---|---|---|---|
| $x_1$ | selection pressure at $e_0$ | $10^{-2+10^{2x_1}}$ | $[10^{-2}, 10^{98}]$ |
| $x_2$ | selection pressure at $e_{end}$ | $10^{-2+10^{2x_2}}$ | $[10^{-2}, 10^{98}]$ |
| $x_3$ | batch size at $e_0$ | $2^{4+4x_3}$ | $[2^4, 2^8]$ |
| $x_4$ | batch size at $e_{end}$ | $2^{4+4x_4}$ | $[2^4, 2^8]$ |
| $x_5$ | frequency of loss recomputation $r_{freq}$ | $2x_5$ | $[0, 2]$ |
| $x_6$ | alpha for batch normalization | $0.01 + 0.2x_6$ | $[0.01, 0.21]$ |
| $x_7$ | epsilon for batch normalization | $10^{-8+5x_7}$ | $[10^{-8}, 10^{-3}]$ |
| $x_8$ | dropout rate after the first Max-Pooling layer | $0.8x_8$ | $[0, 0.8]$ |
| $x_9$ | dropout rate after the second Max-Pooling layer | $0.8x_9$ | $[0, 0.8]$ |
| $x_{10}$ | dropout rate before the output layer | $0.8x_{10}$ | $[0, 0.8]$ |
| $x_{11}$ | number of filters in the first convolution layer | $2^{3+5x_{11}}$ | $[2^3, 2^8]$ |
| $x_{12}$ | number of filters in the second convolution layer | $2^{3+5x_{12}}$ | $[2^3, 2^8]$ |
| $x_{13}$ | number of units in the fully-connected layer | $2^{4+5x_{13}}$ | $[2^4, 2^9]$ |
| $x_{14}$ | Adadelta: learning rate at $e_0$ | $10^{0.5-2x_{14}}$ | $[10^{-1.5}, 10^{0.5}]$ |
| $x_{15}$ | Adadelta: learning rate at $e_{end}$ | $10^{0.5-2x_{15}}$ | $[10^{-1.5}, 10^{0.5}]$ |
| $x_{16}$ | Adadelta: $\rho$ | $0.8 + 0.199x_{16}$ | $[0.8, 0.999]$ |
| $x_{17}$ | Adadelta: $\epsilon$ | $10^{-3-6x_{17}}$ | $[10^{-9}, 10^{-3}]$ |
| $x_{14}$ | Adam: learning rate at $e_0$ | $10^{-1-3x_{14}}$ | $[10^{-4}, 10^{-1}]$ |
| $x_{15}$ | Adam: learning rate at $e_{end}$ | $10^{-3-3x_{15}}$ | $[10^{-6}, 10^{-3}]$ |
| $x_{16}$ | Adam: $\beta_1$ | $0.8 + 0.199x_{16}$ | $[0.8, 0.999]$ |
| $x_{17}$ | Adam: $\epsilon$ | $10^{-3-6x_{17}}$ | $[10^{-9}, 10^{-3}]$ |
| $x_{18}$ | Adam: $\beta_2$ | $1 - 10^{-2-2x_{18}}$ | $[0.99, 0.9999]$ |
| $x_{19}$ | adaptation end epoch index $e_{end}$ | $20 + 200x_{19}$ | $[20, 220]$ |

**Hyperparameters to optimize**

Ilya Loshchilov & Frank Hutter. CMA-ES for hyperparameter optimization of deep neural networks. ICLR 2016

# CMA-ES for deep learning hyperparameter optimization




Comparison of optimizers for Adam with batch selection when solutions are evaluated sequentially for 5 minutes each (left), and in parallel for 30 minutes each (right).

Ilya Loshchilov & Frank Hutter. CMA-ES for hyperparameter optimization of deep neural networks. ICLR 2016

# Covariance matrix adaptation evolutionary strategy (CMA-ES) algorithm

## The CMA-ES

**Input:** $\boldsymbol{m} \in \mathbb{R}^n$, $\sigma \in \mathbb{R}_+$, $\lambda$

**Initialize:** $\mathbf{C} = \mathbf{I}$, and $\boldsymbol{p}_c = \mathbf{0}$, $\boldsymbol{p}_\sigma = \mathbf{0}$,

**Set:** $c_c \approx 4/n$, $c_\sigma \approx 4/n$, $c_1 \approx 2/n^2$, $c_\mu \approx \mu_w/n^2$, $c_1 + c_\mu \leq 1$, $d_\sigma \approx 1 + \sqrt{\frac{\mu_w}{n}}$,
and $w_{i=1\ldots\lambda}$ such that $\mu_w = \frac{1}{\sum_{i=1}^{\mu} {w_i}^2} \approx 0.3\,\lambda$

**While not terminate**

| | |
|---|---|
| $\boldsymbol{x}_i = \boldsymbol{m} + \sigma\,\boldsymbol{y}_i, \quad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})\,, \quad$ for $i = 1, \ldots, \lambda$ | sampling |
| $\boldsymbol{m} \leftarrow \sum_{i=1}^{\mu} w_i\,\boldsymbol{x}_{i:\lambda} = \boldsymbol{m} + \sigma\boldsymbol{y}_w \quad$ where $\boldsymbol{y}_w = \sum_{i=1}^{\mu} w_i\,\boldsymbol{y}_{i:\lambda}$ | update mean |
| $\boldsymbol{p}_c \leftarrow (1 - c_c)\,\boldsymbol{p}_c + \mathbb{1}_{\{\lVert\boldsymbol{p}_\sigma\rVert < 1.5\sqrt{n}\}} \sqrt{1 - (1 - c_c)^2}\sqrt{\mu_w}\,\boldsymbol{y}_w$ | cumulation for $\mathbf{C}$ |
| $\boldsymbol{p}_\sigma \leftarrow (1 - c_\sigma)\,\boldsymbol{p}_\sigma + \sqrt{1 - (1 - c_\sigma)^2}\sqrt{\mu_w}\,\mathbf{C}^{-\frac{1}{2}}\boldsymbol{y}_w$ | cumulation for $\sigma$ |
| $\mathbf{C} \leftarrow (1 - c_1 - c_\mu)\,\mathbf{C} + c_1\,\boldsymbol{p}_c\boldsymbol{p}_c^{\mathrm{T}} + c_\mu \sum_{i=1}^{\mu} w_i\,\boldsymbol{y}_{i:\lambda}\boldsymbol{y}_{i:\lambda}^{\mathrm{T}}$ | update $\mathbf{C}$ |
| $\sigma \leftarrow \sigma \times \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\lVert\boldsymbol{p}_\sigma\rVert}{\mathsf{E}\lVert\mathcal{N}(\mathbf{0},\mathbf{I})\rVert} - 1\right)\right)$ | update of $\sigma$ |

**Not covered** on this slide: termination, restarts, useful output, boundaries and encoding

Anne Auger & Dimo Brockhoff (Introduction to optimization class)

# Rank-u update in CMA-ES

- Increases the possible  learning rate in large populations

- Uses the evolution path and reduces the number of necessary function evaluations

# Properties of CMA-ES

- Learns the dependencies between variables
- Invariant for any increasing function g : g(f(x))
- increases the likelihood of previously successful steps
- Approximates  the inverse of  hessian on quadratic functions

# Problems of CMA-ES

- O(n^2) time and space complexities :

  -To store and update  C in R^(n by n)

  - To compute the eigendecomposition of C

- Internal CPU-time  10^(-8) n^(2) secondes per function evaluation on a 2GHZ pc, tweaks are available

  -1 000 000 f -evaluations in 100-D take 100 seconds internal CPU-time

**Youhei Akimoto, Anne Auger and Nikolaus Hansen .  In CMA-ES and Advanced Adaptation Mechanisms talk**

# Covariance matrix adaptation evolutionary strategy (CMA-ES)

Initialize distribution parameters $\{m, C\}$, set population size $\lambda \in \mathbb{N}$
While not terminate

1- Sample $\lambda$ independant points $x_1 \ldots . x_\lambda$ from $P(x|\theta)$ such that
$P(x|\theta)$ multivariate gaussian distribution
$\theta = \{m, \sigma^2, C\}$
$m$ : mean vector, $m \in \mathbb{R}^n$
$\sigma$ : global step size controls step length, $\sigma \in \mathbb{R}_+$
$C$ : Covariance matrix (symmetric, positive definite) determines the shape of the distribution ellipsoid, $C \in \mathbb{R}^{nxn}$

2- Evaluate the fitness values $f(x_1), \ldots, f(x_\lambda)$

3-Update the parameters of $\theta$

$$m^{t+1} = m^t + \eta_m \sum_{i=1}^{\lambda} W_{R_i}(x_i - m^t) \quad (1)$$

$$C^{t+1} = C^t + \eta_C \sum_{i=1}^{\lambda} W_{R_i}((x_i - m^t)(x_i - m^t)^T - C^t) \quad (2)$$

where $\eta_m$ and $\eta_C$ are learning rate parameters
and $W_{R_i}$ : the weight for the $R_i^{th}$ highest point

# Rank-u update CMA-ES and Natural Gradient Ascent

1- Expected Fitness

$$J(\theta) = \mathbb{E}[f(x);\theta] = \int f(x)P(x;\theta)dx \tag{3}$$

$J(\theta)$ a function on a Riemannian manifold.

2-Natural Gradient

$$\tilde{\nabla} J(\theta) = F_{\theta}^{-1} \nabla J(\theta) \tag{4}$$

2-A $F_\theta$ : Fisher metric for $\theta$ such that

$$\begin{aligned} F_\theta &= \int \frac{\partial lnP(x;\theta)}{\partial\theta} \left(\frac{\partial lnP(x;\theta)}{\partial\theta}\right)^T P(x;\theta)dx \\ &= \mathbb{E}\left[\frac{\partial lnP(x;\theta)}{\partial\theta} \left(\frac{\partial lnP(x;\theta)}{\partial\theta}\right)^T\right] \end{aligned} \tag{5}$$

2-B $\nabla J(\theta)$ gradient of $J$

$$\begin{aligned} \nabla J(\theta) &= \nabla \int f(x)P(x,\theta)\nabla lnP(x,\theta)dx \\ &= \mathbb{E}[f(x)\nabla lnP(x,\theta)] \end{aligned} \tag{6}$$

Under some regularity conditions which are derived from Lebsgue's dominated convergence theorem 16.3

Billingsley, Probability and Measure. 1995

# Rank-u update CMA-ES and Natural Gradient Ascent

3- Monte carlo approximation of the natural gradient : fitness function is unknown

$$\delta(\theta|\{x_i\}) = \sum_{i=1}^{\lambda} \frac{f(x_i)}{\lambda} \mathbf{F}^{-1}(\theta) \nabla ln P(x_i; \theta) \tag{7}$$

Circumvent the computation of the inverse of the information matrix by theorem 4.1

$$\tilde{\delta}(\theta|\{x_i\}) = \sum_{i=1}^{\lambda} \frac{f(x_i)}{\lambda} \begin{bmatrix} x_i - m(\theta^t) \\ vech((x_i - m(\theta^t))(x_i - m(\theta^t))^T - C(\theta^t)) \end{bmatrix} \tag{8}$$

4- Update rule for natural gradient learning :

update $\theta$ such that : $\theta = \theta + \eta \tilde{\delta}(\theta)$

$$m^{t+1} = m^t + \eta \sum_{i=1}^{\lambda} \frac{f(x_i)}{\lambda} (x_i - m^t) \tag{9}$$

$$C^{t+1} = C^t + \eta \sum_{i=1}^{\lambda} \frac{f(x_i)}{\lambda} ((x_i - m^t)(x_i - m^t)^T - C^t) \tag{10}$$



$\eta_C = \eta_m = \eta$

# Important property of Natural Gradient

**The natural gradient equipped with the Fisher metric on the density p is invariant under re-parameterization of the distribution**

$$F_\theta = \int \frac{\partial \ln P(x;\theta)}{\partial \theta} \left( \frac{\partial \ln P(x;\theta)}{\partial \theta} \right)^T P(x;\theta) dx$$

$$= \mathbb{E}\left[ \frac{\partial \ln P(x;\theta)}{\partial \theta} \left( \frac{\partial \ln P(x;\theta)}{\partial \theta} \right)^T \right]$$

# Importance of invariance

Why it is important to build invariants ?

# Natural gradient ascent along with rank-u CMA-ES update

**Natural gradient ascent with monte-carlo approximation ⇔ rank-u CMA-ES**

From equation (1), (2) of CMA-ES

$$m^{t+1} = m^t + \eta_m \sum_{i=1}^{\lambda} W_{R_i}(x_i - m^t) \qquad (1)$$

$$C^{t+1} = C^t + \eta_C \sum_{i=1}^{\lambda} W_{R_i}((x_i - m^t)(x_i - m^t)^T - C^t) \qquad (2)$$

And from (9), (10) of natural gradient

$W_{R_i} = \sum_{i=1}^{\lambda} \frac{f(x_i)}{\lambda}$

$$m^{t+1} = m^t + \eta \sum_{i=1}^{\lambda} \frac{f(x_i)}{\lambda}(x_i - m^t) \qquad (9)$$

$$C^{t+1} = C^t + \eta \sum_{i=1}^{\lambda} \frac{f(x_i)}{\lambda}((x_i - m^t)(x_i - m^t)^T - C^t) \qquad (10)$$

# Building invariants

**How to build invariants ?**

# Building invariants

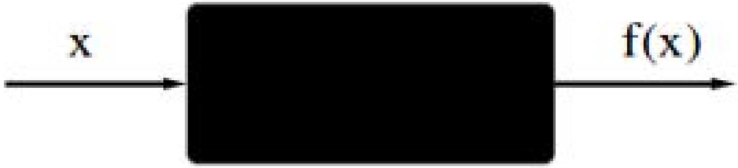
## How to build invariants ?

- Riemannian geometry
- Information theory
- Group theory
- Approximation theory
- Theory of complexity

# Perspectives and open problems

- Theoretical foundation for the evolution path and  cumulation
- Evaluation of the estimated natural gradient (Coefficients)
- Stability of CMA-ES
- Fitness shaping exploration
- Build a group of invariants

# Limit of CMA-ES




**What if x is on an arbitrary space ?**

# Extended results from Information-Geometric Optimization and group theory

Perspective from Information geometric optimization :

- Optimization on arbitrary space
- Invariance principles for generalization

**f-invariance , theta-invariance, X-invariance**

- Quantile rewriting of the objective function

**Yann Ollivier, Ludovic Arnold, Anne Auger and Nikolaus Hansen. Information-Geometric Optimization Algorithms: A Unifying Picture via Invariance Principles. 2011**

# Extended results from Information-Geometric Optimization and group theory

Perspective from group theory :

- Learning group of invariants (translation, rotation, elastic deformation)
- Linearization of symmetries in high dimension without a loss of information
- Retrieve more invariants from the inner structure of information

**Stephane Mallat. Understanding deep convolutional neural networks. 2016**

# My intuition

- Building invariants in CMA-ES can also lead to theoretical grounding of deep neural networks
- Both CMA-ES and  neural networks are black-box architectures
- Rely on neural networks to learn invariants for CMA.
- Apply CMA-ES to optimize the hyperparameters of deep architecture to learn the complex inner structure
- Dialectic relationship between CMA-ES and deep neural networks

# References


- Youhei Akimoto, Yuichi Nagata, Isao Ono, Shigenobu Kobayashi. Theoretical foundation for CMA-ES from information geometric perspective. 2012
- Yann Ollivier, Ludovic Arnold, Anne Auger and Nikolaus Hansen. Information-Geometric Optimization Algorithms: A Unifying Picture via Invariance Principles. 2011
- Stephane Mallat. Understanding deep convolutional neural networks. 2016
- Tom Schaul . Studies in Continuous Black-box Optimization, PHD thesis. Technical university of Munich. 2011
- Ilya Loshchilov & Frank Hutter. CMA-ES for hyperparameter optimization of deep neural networks. ICLR 2016
- Yann Ollivier. Optimization and recurrent neural networks training talk. College de France, 2016
- Stephane Mallat. Mathematical mysteries of deep neural networks. College de France, 2016
- Anne Auger. How information theory sheds new light on black-box optimization talk. Institut Henré Poincaré, France, 2016
- Anne Auger, Youhei Akimoto and Nikolaus Hansen. Miscellaneous talks.
- Anne Auger and Dimo Brockhoff. Introduction and advanced optimization class (M2 AIC). 2016-2017