Tutorial: CMA-ES — Evolution Strategies and Covariance Matrix Adaptation

Anne Auger & Nikolaus Hansen

INRIA Research Centre Saclay – Île-de-France
Project team TAO
University Paris-Sud, LRI (UMR 8623), Bat. 490
91405 ORSAY Cedex, France

GECCO’11, July 12–16, 2011, Dublin, Ireland.
get the slides: google "Nikolaus Hansen"... under Publications click Invited talks, tutorials...
Content

1. Problem Statement
   - Black Box Optimization and Its Difficulties
   - Non-Separable Problems
   - Ill-Conditioned Problems

2. Evolution Strategies
   - A Search Template
   - The Normal Distribution
   - Invariance

3. Step-Size Control
   - Why Step-Size Control
   - One-Fifth Success Rule
   - Path Length Control (CSA)

4. Covariance Matrix Adaptation
   - Covariance Matrix Rank-One Update
   - Cumulation—the Evolution Path
   - Covariance Matrix Rank-$\mu$ Update

5. Theoretical Foundations

6. Experiments

7. Summary and Final Remarks
Problem Statement
Continuous Domain Search/Optimization

- **Task:** minimize an objective function (*fitness function*, *loss* function) in continuous domain
  \[ f : \mathcal{X} \subseteq \mathbb{R}^n \rightarrow \mathbb{R}, \quad x \mapsto f(x) \]

- **Black Box** scenario (direct search scenario)
  - gradients are not available or not useful
  - problem domain specific knowledge is used only within the black box, e.g. within an appropriate encoding

- **Search costs:** number of function evaluations
Problem Statement
Continuous Domain Search/Optimization

Goal
- fast convergence to the global optimum
- solution $x$ with **small function value** $f(x)$ with **least search cost**
- there are two conflicting objectives

Typical Examples
- shape optimization (e.g. using CFD)
- model calibration
- parameter calibration
- curve fitting, airfoils
- biological, physical
- controller, plants, images

Problems
- exhaustive search is infeasible
- naive random search takes too long
- deterministic search is not successful / takes too long

**Approach**: stochastic search, Evolutionary Algorithms
Objective Function Properties

We assume \( f : \mathcal{X} \subset \mathbb{R}^n \rightarrow \mathbb{R} \) to be non-linear, non-separable and to have at least moderate dimensionality, say \( n \ll 10 \). Additionally, \( f \) can be

- non-convex
- multimodal
  - there are possibly many local optima
- non-smooth
  - derivatives do not exist
- discontinuous
- ill-conditioned
- noisy
- . . .

**Goal**: cope with any of these function properties
they are related to real-world problems
What Makes a Function Difficult to Solve?

Why stochastic search?

- non-linear, non-quadratic, non-convex
  on linear and quadratic functions much better
  search policies are available

- ruggedness
  non-smooth, discontinuous, multimodal, and/or
  noisy function

- dimensionality (size of search space)
  (considerably) larger than three

- non-separability
  dependencies between the objective variables

- ill-conditioning

---

gradient direction Newton direction

Anne Auger & Nikolaus Hansen ()
CMA-ES
July, 2011 6 / 80
Ruggedness
non-smooth, discontinuous, multimodal, and/or noisy

cut from a 5-D example, (easily) solvable with evolution strategies
Curse of Dimensionality

The term *Curse of dimensionality* (Richard Bellman) refers to problems caused by the **rapid increase in volume** associated with adding extra dimensions to a (mathematical) space.

Example: Consider placing 100 points onto a real interval, say $[0, 1]$. To get **similar coverage**, in terms of distance between adjacent points, of the 10-dimensional space $[0, 1]^{10}$ would require $100^{10} = 10^{20}$ points. A 100 points appear now as isolated points in a vast empty space.

Remark: **distance measures** break down in higher dimensionalities (the central limit theorem kicks in)

Consequently, a **search policy** (e.g. exhaustive search) that is valuable in small dimensions **might be useless** in moderate or large dimensional search spaces.
Effect of Dimensionality: Example

\[ \| \mathcal{N}(0, I) - \mathcal{N}(0, I) \| / \sqrt{2} \sim \| \mathcal{N}(0, I) \| \rightarrow \mathcal{N} \left( \sqrt{n - 1/2}, 1/2 \right), \]

with modal value: \( \sqrt{n - 1} \)
Separable Problems

Definition (Separable Problem)
A function $f$ is separable if

$$\arg\min_{(x_1, \ldots, x_n)} f(x_1, \ldots, x_n) = \left( \arg\min_{x_1} f(x_1, \ldots), \ldots, \arg\min_{x_n} f(\ldots, x_n) \right)$$

⇒ it follows that $f$ can be optimized in a sequence of $n$ independent 1-D optimization processes

Example: Additively decomposable functions

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{n} f_i(x_i)$$

Rastrigin function
Non-Separable Problems

Building a non-separable problem from a separable one \(^{(1,2)}\)

Rotating the coordinate system

- \( f : x \mapsto f(x) \) separable
- \( f : x \mapsto f(Rx) \) non-separable

\( R \) rotation matrix

---

\(^1\) Hansen, Ostermeier, Gawelczyk (1995). On the adaptation of arbitrary normal mutation distributions in evolution strategies: The generating set adaptation. Sixth ICGA, pp. 57-64, Morgan Kaufmann

Ill-Conditioned Problems

Curvature of level sets

Consider the convex-quadratic function

\[ f(x) = \frac{1}{2}(x - x^*)^T H(x - x^*) = \frac{1}{2} \sum_i h_{i,i} x_i^2 + \frac{1}{2} \sum_{i \neq j} h_{i,j} x_i x_j \]

- \( H \) is Hessian matrix of \( f \) and symmetric positive definite
- gradient direction \(-f'(x)^T\)
- Newton direction \(-H^{-1}f'(x)^T\)

Ill-conditioning means **squeezed level sets** (high curvature).
Condition number equals nine here. Condition numbers up to \(10^{10}\) are not unusual in real world problems.

If \( H \approx I \) (small condition number of \( H \)) first order information (e.g. the gradient) is sufficient. Otherwise **second order information** (estimation of \( H^{-1} \)) is necessary.
## Problem Statement

### Ill-Conditioned Problems

What Makes a Function Difficult to Solve?  
... and what can be done

<table>
<thead>
<tr>
<th>The Problem</th>
<th>The Approach in ESs and continuous EDAs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimensionality, Non-Separability</td>
<td>exploiting the problem structure</td>
</tr>
<tr>
<td></td>
<td>locality, neighborhood, encoding</td>
</tr>
<tr>
<td>Ill-conditioning</td>
<td>second order approach</td>
</tr>
<tr>
<td></td>
<td>changes the neighborhood metric</td>
</tr>
<tr>
<td>Ruggedness</td>
<td><strong>non-local</strong> policy, large sampling width (step-size)</td>
</tr>
<tr>
<td></td>
<td>as large as possible while preserving a</td>
</tr>
<tr>
<td></td>
<td>reasonable convergence speed</td>
</tr>
<tr>
<td></td>
<td>stochastic, non-elitistic, <strong>population-based</strong> method</td>
</tr>
<tr>
<td></td>
<td>recombination operator</td>
</tr>
<tr>
<td></td>
<td>serves as repair mechanism</td>
</tr>
<tr>
<td></td>
<td>restarts</td>
</tr>
</tbody>
</table>

... metaphors
### Metaphors

<table>
<thead>
<tr>
<th>Evolutionary Computation</th>
<th>Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>individual, offspring, parent</td>
<td>candidate solution</td>
</tr>
<tr>
<td></td>
<td>decision variables</td>
</tr>
<tr>
<td></td>
<td>design variables</td>
</tr>
<tr>
<td></td>
<td>object variables</td>
</tr>
<tr>
<td>population</td>
<td>set of candidate solutions</td>
</tr>
<tr>
<td>fitness function</td>
<td>objective function</td>
</tr>
<tr>
<td></td>
<td>loss function</td>
</tr>
<tr>
<td></td>
<td>cost function</td>
</tr>
<tr>
<td></td>
<td>error function</td>
</tr>
<tr>
<td>generation</td>
<td>iteration</td>
</tr>
</tbody>
</table>

*methods: ESs*
Evolution Strategies

1. Problem Statement
   - Black Box Optimization and Its Difficulties
   - Non-Separable Problems
   - Ill-Conditioned Problems

2. Evolution Strategies
   - A Search Template
   - The Normal Distribution
   - Invariance

3. Step-Size Control
   - Why Step-Size Control
   - One-Fifth Success Rule
   - Path Length Control (CSA)

4. Covariance Matrix Adaptation
   - Covariance Matrix Rank-One Update
   - Cumulation—the Evolution Path
   - Covariance Matrix Rank-$\mu$ Update

5. Theoretical Foundations

6. Experiments

7. Summary and Final Remarks
A black box search template to minimize $f : \mathbb{R}^n \to \mathbb{R}$

Initialize distribution parameters $\theta$, set population size $\lambda \in \mathbb{N}$

While not terminate

1. Sample distribution $P(x|\theta) \to x_1, \ldots, x_\lambda \in \mathbb{R}^n$
2. Evaluate $x_1, \ldots, x_\lambda$ on $f$
3. Update parameters $\theta \leftarrow F_\theta(\theta, x_1, \ldots, x_\lambda, f(x_1), \ldots, f(x_\lambda))$

Everything depends on the definition of $P$ and $F_\theta$

deterministic algorithms are covered as well

In many Evolutionary Algorithms the distribution $P$ is implicitly defined via operators on a population, in particular, selection, recombination and mutation

Natural template for *Estimation of Distribution Algorithms*
Evolution Strategies

New search points are sampled normally distributed

\[ x_i \sim m + \sigma \mathcal{N}_i(0, C) \quad \text{for } i = 1, \ldots, \lambda \]

as perturbations of \( m \), where \( x_i, m \in \mathbb{R}^n \), \( \sigma \in \mathbb{R}_+ \), \( C \in \mathbb{R}^{n \times n} \)

where

- the mean vector \( m \in \mathbb{R}^n \) represents the favorite solution
- the so-called step-size \( \sigma \in \mathbb{R}_+ \) controls the \textit{step length}
- the covariance matrix \( C \in \mathbb{R}^{n \times n} \) determines the \textit{shape} of the distribution ellipsoid

here, all new points are sampled with the same parameters

The question remains how to update \( m \), \( C \), and \( \sigma \).
Why Normal Distributions?

1. widely observed in nature, for example as phenotypic traits
2. only stable distribution with finite variance
   stable means the sum of normal variates is again normal, helpful in design and analysis of algorithms
collection to central limit theorem
3. most convenient way to generate isotropic search points
   the isotropic distribution does not favor any direction (unfoundedly), supports rotational invariance
4. maximum entropy distribution with finite variance
   the least possible assumptions on $f$ in the distribution shape
Normal Distribution

probability density of the 1-D standard normal distribution

probability density of a 2-D normal distribution
The Multi-Variate \((n\text{-Dimensional})\) Normal Distribution

Any multi-variate normal distribution \(\mathcal{N}(m, C)\) is uniquely determined by its mean value \(m \in \mathbb{R}^n\) and its symmetric positive definite \(n \times n\) covariance matrix \(C\).

The **mean value** \(m\)

- determines the displacement (translation)
- value with the largest density (modal value)
- the distribution is symmetric about the distribution mean

The **covariance matrix** \(C\)

- determines the shape

- **geometrical interpretation**: any covariance matrix can be uniquely identified with the iso-density ellipsoid \(\{x \in \mathbb{R}^n \mid (x - m)^T C^{-1} (x - m) = 1\}\)
...any covariance matrix can be uniquely identified with the iso-density ellipsoid 
\( \{ x \in \mathbb{R}^n \mid (x - m)^T C^{-1} (x - m) = 1 \} \)

Lines of Equal Density

\[
\begin{align*}
\mathcal{N}(m, \sigma^2 I) & \sim m + \sigma \mathcal{N}(0, I) & \text{one degree of freedom} \quad \sigma \\
\mathcal{N}(m, D^2) & \sim m + D \mathcal{N}(0, I) & n \text{ degrees of freedom} \\
\mathcal{N}(m, C) & \sim m + C^{\frac{1}{2}} \mathcal{N}(0, I) & (n^2 + n)/2 \text{ degrees of freedom}
\end{align*}
\]

where \( I \) is the identity matrix (isotropic case) and \( D \) is a diagonal matrix (reasonable for separable problems) and \( A \times \mathcal{N}(0, I) \sim \mathcal{N}(0, AA^T) \) holds for all \( A \).
Evolution Strategies

Terminology

Let $\mu$: # of parents, $\lambda$: # of offspring

Plus (elitist) and comma (non-elitist) selection

$(\mu + \lambda)$-ES: selection in $\{\text{parents}\} \cup \{\text{offspring}\}$

$(\mu, \lambda)$-ES: selection in $\{\text{offspring}\}$

$(1 + 1)$-ES

Sample one offspring from parent $m$

$$x = m + \sigma \mathcal{N}(0, C)$$

If $x$ better than $m$ select

$$m \leftarrow x$$
The \((\mu/\mu, \lambda)\)-ES

Non-elitist selection and intermediate (weighted) recombination

Given the \(i\)-th solution point \(x_i = m + \sigma \mathcal{N}_i(0, C) = m + \sigma y_i\)

Let \(x_{i:\lambda}\) the \(i\)-th ranked solution point, such that \(f(x_{1:\lambda}) \leq \cdots \leq f(x_{\lambda:\lambda})\).

The new mean reads

\[
m \leftarrow \sum_{i=1}^{\mu} w_i x_{i:\lambda} = m + \sigma \sum_{i=1}^{\mu} w_i y_{i:\lambda}
\]

where

\[
w_1 \geq \cdots \geq w_\mu > 0, \quad \sum_{i=1}^{\mu} w_i = 1, \quad \frac{1}{\sum_{i=1}^{\mu} w_i^2} =: \mu_w \approx \frac{\lambda}{4}
\]

The best \(\mu\) points are selected from the new solutions (non-elitistic) and weighted intermediate recombination is applied.
Invariance Under Monotonically Increasing Functions

Rank-based algorithms

Update of all parameters uses only the ranks

\[ f(x_1:λ) \leq f(x_2:λ) \leq \ldots \leq f(x_λ:λ) \]

\[ g(f(x_1:λ)) \leq g(f(x_2:λ)) \leq \ldots \leq g(f(x_λ:λ)) \quad \forall g \]

\( g \) is strictly monotonically increasing
\( g \) preserves ranks

---

3. Whitley 1989. The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best, ICGA
Basic Invariance in Search Space

- translation invariance

\[ f(x) \leftrightarrow f(x - a) \]

Identical behavior on \( f \) and \( f_a \)

\[
\begin{align*}
  f : & \quad x \mapsto f(x), & \quad x^{(t=0)} = x_0 \\
  f_a : & \quad x \mapsto f(x - a), & \quad x^{(t=0)} = x_0 + a
\end{align*}
\]

No difference can be observed w.r.t. the argument of \( f \)
Rotational Invariance in Search Space

- invariance to orthogonal (rigid) transformations \( \mathbf{R} \), where \( \mathbf{RR}^T = \mathbf{I} \)
  
  e.g. true for simple evolution strategies recombination operators might jeopardize rotational invariance

\[
f(x) \leftrightarrow f(Rx)
\]

Identical behavior on \( f \) and \( f_R \)

\[
f: \quad x \mapsto f(x), \quad x^{(t=0)} = x_0 \\
f_R: \quad x \mapsto f(Rx), \quad x^{(t=0)} = R^{-1}(x_0)
\]

No difference can be observed w.r.t. the argument of \( f \)

---


The grand aim of all science is to cover the greatest number of empirical facts by logical deduction from the smallest number of hypotheses or axioms.

— Albert Einstein

- Empirical performance results, for example
  - from benchmark functions
  - from solved real world problems

  are only useful if they do generalize to other problems

- Invariance is a strong non-empirical statement about generalization
  generalizing (identical) performance from a single function to a whole class of functions

consequently, invariance is important for the evaluation of search algorithms
1 Problem Statement
   • Black Box Optimization and Its Difficulties
   • Non-Separable Problems
   • Ill-Conditioned Problems

2 Evolution Strategies
   • A Search Template
   • The Normal Distribution
   • Invariance

3 Step-Size Control
   • Why Step-Size Control
   • One-Fifth Success Rule
   • Path Length Control (CSA)

4 Covariance Matrix Adaptation
   • Covariance Matrix Rank-One Update
   • Cumulation—the Evolution Path
   • Covariance Matrix Rank-$\mu$ Update

5 Theoretical Foundations

6 Experiments

7 Summary and Final Remarks
Evolution Strategies

Recalling

New search points are sampled normally distributed

\[ x_i \sim m + \sigma N_i(0, C) \quad \text{for } i = 1, \ldots, \lambda \]

as perturbations of \( m \), where \( x_i, m \in \mathbb{R}^n, \sigma \in \mathbb{R}_+, C \in \mathbb{R}^{n \times n} \)

where

- the mean vector \( m \in \mathbb{R}^n \) represents the favorite solution and \( m \leftarrow \sum_{i=1}^{\mu} w_i x_i; \lambda \)
- the so-called step-size \( \sigma \in \mathbb{R}_+ \) controls the step length
- the covariance matrix \( C \in \mathbb{R}^{n \times n} \) determines the shape of the distribution ellipsoid

The remaining question is how to update \( \sigma \) and \( C \).
Why Step-Size Control?

The step-size control is crucial in optimization algorithms to ensure efficient exploration of the search space. The diagram illustrates the behavior of the function $f(x)$ defined as:

$$f(x) = \sum_{i=1}^{n} x_i^2$$

within the interval $[-0.2, 0.8]^n$ for $n = 10$.

- **Step-size too small**: The algorithm might stagnate due to insufficient exploration.
- **Step-size too large**: The algorithm might overshoot the optimal solution.
- **Random search**: This approach does not consider the scale of the problem and can be inefficient.
- **Optimal step-size**: This is a scale-invariant parameter that adapts to the problem's scale, ensuring efficient convergence.

Graphically, the function evaluations and function values are shown, highlighting the effectiveness of the optimal step-size control.
Why Step-Size Control?

\[ f(x) = \sum_{i=1}^{n} x_i^2 \]

in \([-0.2, 0.8]^n\)

for \(n = 10\)
Why Step-Size Control?

\[ f(x) = \sum_{i=1}^{n} x_i^2 \]

in \([-0.2, 0.8]^n\) for \(n = 10\)
Why Step-Size Control?

\[
f(x) = \sum_{i=1}^{n} x_i^2
\]

in \([-0.2, 0.8]^n\)

for \(n = 10\)
Why Step-Size Control?

\[ \sigma \leftarrow \sigma^* \| \text{parent} \| \]

\[ \frac{\varphi^*}{n} \]

*evolution window* refers to the step-size interval (-----) where reasonable performance is observed.
Methods for Step-Size Control

- **1/5-th success rule**\(^{ab}\), often applied with “+”-selection
  
  increase step-size if more than 20% of the new solutions are successful, decrease otherwise

- **\(\sigma\)-self-adaptation**\(^c\), applied with “,”-selection

  mutation is applied to the step-size and the better, according to the objective function value, is selected

  simplified “global” self-adaptation

- **path length control**\(^d\) (Cumulative Step-size Adaptation, CSA)**\(^e\)

  self-adaptation derandomized and non-localized

---


\(^{b}\) Schumer and Steiglitz 1968. Adaptive step size random search. *IEEE TAC*


\(^{e}\) Ostermeier et al 1994, Step-size adaptation based on non-local use of selection information, *PPSN IV*
One-fifth success rule

increase $\sigma$

decrease $\sigma$
One-fifth success rule

Probability of success \((p_s)\)

1/2

1/5

“too small”
One-fifth success rule

\( p_s : \# \text{ of successful offspring} / \# \text{ offspring (per generation)} \)

\[
\sigma \leftarrow \sigma \times \exp \left( \frac{1}{3} \times \frac{p_s - p_{\text{target}}}{1 - p_{\text{target}}} \right)
\]

Increase \( \sigma \) if \( p_s > p_{\text{target}} \)

Decrease \( \sigma \) if \( p_s < p_{\text{target}} \)

(1 + 1)-ES

\( p_{\text{target}} = 1/5 \)

IF offspring better parent

\( p_s = 1, \sigma \leftarrow \sigma \times \exp(1/3) \)

ELSE

\( p_s = 0, \sigma \leftarrow \sigma / \exp(1/3)^{1/4} \)
Path Length Control (CSA)

The Concept of Cumulative Step-Size Adaptation

\[ x_i = m + \sigma y_i \]

\[ m \leftarrow m + \sigma y_w \]

Measure the length of the *evolution path*

the pathway of the mean vector \( m \) in the generation sequence

loosely speaking steps are

- perpendicular under random selection (in expectation)
- perpendicular in the desired situation (to be most efficient)
Path Length Control (CSA)

The Equations

Initialize \( m \in \mathbb{R}^n \), \( \sigma \in \mathbb{R}_+ \), evolution path \( p_\sigma = 0 \), set \( c_\sigma \approx \frac{4}{n} \), \( d_\sigma \approx 1 \).

- **Update Mean**
  \[
  m \leftarrow m + \sigma y_w
  \text{ where } y_w = \sum_{i=1}^{\mu} w_i y_i : \lambda
  \]

- **Update Evolution Path**
  \[
  p_\sigma \leftarrow (1 - c_\sigma) p_\sigma + \sqrt{1 - (1 - c_\sigma)^2} \sqrt{\mu} w \sqrt{\mu} y_w
  \text{ accounts for } 1 - c_\sigma
  \]

- **Update Step-size**
  \[
  \sigma \leftarrow \sigma \times \exp \left( \frac{c_\sigma}{d_\sigma} \left( \frac{\|p_\sigma\|}{\mathbb{E}\|\mathcal{N}(0, 1)\|} - 1 \right) \right)
  \]
  \( > 1 \iff \|p_\sigma\| \text{ is greater than its expectation} \)
Step-Size Control

Path Length Control (CSA)

\[ f(x) = \sum_{i=1}^{n} x_i^2 \]

in \([-0.2, 0.8]^n\)

for \(n = 10\)
Problem Statement

Evolution Strategies

Step-Size Control

Covariance Matrix Adaptation

- Covariance Matrix Rank-One Update
- Cumulation—the Evolution Path
- Covariance Matrix Rank-$\mu$ Update

Theoretical Foundations

Experiments

Summary and Final Remarks
Recalling

New search points are sampled normally distributed

\[ x_i \sim m + \sigma N_i(0, C) \quad \text{for } i = 1, \ldots, \lambda \]

as perturbations of \( m \), where \( x_i, m \in \mathbb{R}^n, \sigma \in \mathbb{R}_+, C \in \mathbb{R}^{n \times n} \)

where

- the mean vector \( m \in \mathbb{R}^n \) represents the favorite solution
- the so-called step-size \( \sigma \in \mathbb{R}_+ \) controls the step length
- the covariance matrix \( C \in \mathbb{R}^{n \times n} \) determines the shape of the distribution ellipsoid

The remaining question is how to update \( C \).
Covariance Matrix Adaptation

Rank-One Update

\[ m \leftarrow m + \sigma y_w, \quad y_w = \sum_{i=1}^{\mu} w_i y_i : \lambda, \quad y_i \sim \mathcal{N}_i(0, C) \]

new distribution,

\[ C \leftarrow 0.8 \times C + 0.2 \times y_w y_w^T \]

the ruling principle: the adaptation increases the likelihood of successful steps, \( y_w \), to appear again

another viewpoint: the adaptation follows a natural gradient approximation of the expected fitness

... equations
Covariance Matrix Adaptation

Rank-One Update

Initialize $m \in \mathbb{R}^n$, and $C = I$, set $\sigma = 1$, learning rate $c_{\text{cov}} \approx 2/n^2$

While not terminate

$$x_i = m + \sigma y_i, \quad y_i \sim \mathcal{N}_i(0, C),$$

$$m \leftarrow m + \sigma y_w \quad \text{where} \quad y_w = \sum_{i=1}^{\mu} w_i y_{i: \lambda}$$

$$C \leftarrow (1 - c_{\text{cov}})C + c_{\text{cov}} \mu_w y_w y_w^T \quad \text{where} \quad \mu_w = \frac{1}{\sum_{i=1}^{\mu} w_i^2} \geq 1$$
covariance matrix adaptation

- learns all **pairwise dependencies** between variables
  - off-diagonal entries in the covariance matrix reflect the dependencies

- conducts a **principle component analysis (PCA)** of steps $y_w$, sequentially in time and space
  - eigenvectors of the covariance matrix $C$ are the principle components / the principle axes of the mutation ellipsoid, rotational invariant

- learns a new, **rotated problem representation** and a **new metric** (Mahalanobis)
  - components are independent (only) in the new representation
  - rotational invariant

- approximates the **inverse Hessian** on quadratic functions
  - overwhelming empirical evidence, proof is in progress

- is entirely **independent** of the given coordinate system
  - for $\mu = 1$: natural gradient ascent on $\mathcal{N}$
1. Problem Statement
2. Evolution Strategies
3. Step-Size Control
4. Covariance Matrix Adaptation
   - Covariance Matrix Rank-One Update
   - Cumulation—the Evolution Path
   - Covariance Matrix Rank-$\mu$ Update
5. Theoretical Foundations
6. Experiments
7. Summary and Final Remarks
Cumulation
The Evolution Path

Evolution Path

Conceptually, the evolution path is the search path the strategy takes over a number of generation steps. It can be expressed as a sum of consecutive steps of the mean \( m \).

An exponentially weighted sum of steps \( y_w \) is used

\[
p_c \propto \sum_{i=0}^{g} (1 - c_c)^{g-i} y_w^{(i)}
\]

where \( c_c \ll 1 \).

The recursive construction of the evolution path (cumulation):

\[
p_c \leftarrow (1 - c_c) p_c + \sqrt{1 - (1 - c_c)^2} \sqrt{\mu_w} y_w
\]

where \( \mu_w = \frac{1}{\sum w_i^2}, c_c \ll 1 \). History information is accumulated in the evolution path.
“Cumulation” is a widely used technique and also known as

- *exponential smoothing* in time series, forecasting
- exponentially weighted *mooving average*
- *iterate averaging* in stochastic approximation
- *momentum* in the back-propagation algorithm for ANNs
- ...
Cumulation
Utilizing the Evolution Path

We used $y_w y_w^T$ for updating $C$. Because $y_w y_w^T = -y_w (-y_w)^T$ the sign of $y_w$ is lost.

The sign information is (re-)introduced by using the *evolution path*.

\[
p_c \leftarrow (1 - c_c) p_c + \sqrt{1 - (1 - c_c)^2} \sqrt{\mu_w} y_w
\]

\[
C \leftarrow (1 - c_{cov}) C + c_{cov} p_c p_c^T
\]

where $\mu_w = \frac{1}{\sum w_i^2}$, $c_c \ll 1$ such that $1/c_c$ is the “backward time horizon”.

\[\text{decay factor} \hspace{2cm} \text{normalization factor} \]

\[\text{rank-one} \]
Using an **evolution path** for the **rank-one update** of the covariance matrix reduces the number of function evaluations to adapt to a straight ridge from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$.\(^{(a)}\)


The overall model complexity is $n^2$ but important parts of the model can be learned in time of order $n$. 

![Graph showing the number of f-evaluations divided by dimension on the cigar function.](image)
Rank-$\mu$ Update

\[ x_i = m + \sigma y_i, \quad y_i \sim N_i(0, C), \]
\[ m \leftarrow m + \sigma y_w \quad y_w = \sum_{i=1}^{\mu} w_i y_{i:}\lambda \]

The rank-$\mu$ update extends the update rule for large population sizes $\lambda$ using $\mu > 1$ vectors to update $C$ at each generation step. The matrix

\[ C_\mu = \sum_{i=1}^{\mu} w_i y_{i:}\lambda y_{i:}\lambda^T \]

computes a weighted mean of the outer products of the best $\mu$ steps and has rank $\min(\mu, n)$ with probability one. The rank-$\mu$ update then reads

\[ C \leftarrow (1 - c_{\text{cov}}) C + c_{\text{cov}} C_\mu \]

where $c_{\text{cov}} \approx \mu_w / n^2$ and $c_{\text{cov}} \leq 1$. 
\[ x_i = m + \sigma y_i, \quad y_i \sim \mathcal{N}(0, C) \]

\[ C_\mu = \frac{1}{\mu} \sum_{i=1}^{\lambda} y_i y_i^T \]

\[ m_{new} \leftarrow m + \frac{1}{\mu} \sum_{i=1}^{\lambda} y_i \]

Sampling of \( \lambda = 150 \) solutions where \( C = I \) and \( \sigma = 1 \)

Calculating \( C \) where \( \mu = 50 \),

\[ w_1 = \cdots = w_{\mu} = \frac{1}{\mu}, \]

and \( c_{cov} = 1 \)
Rank-$\mu$ CMA versus Estimation of Multivariate Normal Algorithm EMNA$_{\text{global}}$°

$x_i = m_{\text{old}} + y_i, \quad y_i \sim \mathcal{N}(0, C)$

$C \leftarrow \frac{1}{\mu} \sum (x_i: \lambda - m_{\text{old}})(x_i: \lambda - m_{\text{old}})^T$

$m_{\text{new}} = m_{\text{old}} + \frac{1}{\mu} \sum y_i: \lambda$

The CMA-update yields a larger variance in particular in gradient direction, because $m_{\text{new}}$ is the minimizer for the variances when calculating $C$

The **rank-**$\mu$** update**

- increases the possible learning rate in large populations roughly from $2/n^2$ to $\mu_w/n^2$
- can reduce the number of necessary *generations* roughly from $O(n^2)$ to $O(n)$ \(^7\) given $\mu_w \propto \lambda \propto n$

Therefore the rank-$\mu$ update is the primary mechanism whenever a large population size is used say $\lambda \geq 3n + 10$

The **rank-one update**

- uses the evolution path and reduces the number of necessary *function evaluations* to learn straight ridges from $O(n^2)$ to $O(n)$ .

Rank-one update and rank-$\mu$ update can be combined

---

Summary of Equations

The Covariance Matrix Adaptation Evolution Strategy

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

**Initialize:** $C = I$, and $p_c = 0$, $p_\sigma = 0$

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

**While not terminate**

$$x_i = m + \sigma y_i, \quad y_i \sim \mathcal{N}_i(0, C), \quad \text{for } i = 1, \ldots, \lambda$$

sampling

$$m \leftarrow \sum_{i=1}^{\mu} w_i x_{i:\lambda} = m + \sigma y_w \quad \text{where } y_w = \sum_{i=1}^{\mu} w_i y_{i:\lambda}$$

update mean

$$p_c \leftarrow (1 - c_c) p_c + \mathbb{1}_{\|p_\sigma\| < 1.5\sqrt{n}} \sqrt{1 - (1 - c_c)^2} \sqrt{\mu_w} y_w$$

cumulation for $C$

$$p_\sigma \leftarrow (1 - c_\sigma) p_\sigma + \sqrt{1 - (1 - c_\sigma)^2} \sqrt{\mu_w} C^{-\frac{1}{2}} y_w$$

cumulation for $\sigma$

$$C \leftarrow (1 - c_1 - c_\mu) C + c_1 p_c p_c^T + c_\mu \sum_{i=1}^{\mu} w_i y_{i:\lambda} y_{i:\lambda}^T$$

update $C$

$$\sigma \leftarrow \sigma \times \exp \left( \frac{c_\sigma}{d_\sigma} \left( \frac{\|p_\sigma\|}{\mathbb{E}\|\mathcal{N}(0, I)\|} - 1 \right) \right)$$

update of $\sigma$

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

```matlab
counteval = 0; % the next 40 lines contain the 20 lines of interesting code
while counteval < stopeval

% Generate and evaluate lambda offspring
for k=1:lambda,
    arx(:,k) = xmean + sigma * B * (D .* randn(N,1)); % m + sig * Normal(0,C)
    arfitness(k) = feval(strfitnessfct, arx(:,k)); % objective function call
    counteval = counteval+1;
end

% Sort by fitness and compute weighted mean into xmean
[arfitness, arindex] = sort(arfitness); % minimization
xold = xmean;
xmean = arxiv(:,arindex(1:mu))*weights; % recombination, new mean value

% Cumulation: Update evolution paths
ps = (1-csl)*ps ... 
    + sqrt(cs*(2-csl)*muEff) * invsqrtC * (xmean-xold) / sigma;
hSig = norm(ps)/sqrt(1-(1-csl)*(2*counteval/lambda))/(chiN < 1.4 + 2/(N+1));
ps = (1-csl)*ps ... 
    + hSig * sqrt(cs*(2-csl)*muEff) * (xmean-xold) / sigma;

% Adapt covariance matrix C
arCtmp = (1/sigma) * (arxiv(:,arindex(1:mu))-repmat(xold,1,mu));
C = (1-csl-cmu) * C ... % regard old matrix
    + cl * (pc*pc' ... % plus rank one update
    + (1-hSig) * cs*(2-csl) * C) ... % minor correction if hSig==0
    + cmu * arctmp * diag(weights) * arctmp'; % plus rank mu update

% Adapt step size sigma
sigma = sigma * exp((cs/demps)*(norm(ps)/chiN - 1));

% Decomposition of C into B(diag(D.^2)*B' (diagonalization)
if counteval - eigenscale > lambda/(cl+cmu)/N/10 % to achieve O(N^2)
    eigenscale = counteval;
    C = triu(C) + triu(C,1)'; % enforce symmetry
    [B,D] = eig(C); % eigen decomposition, B==normalized eigenvectors
    D = sqrt(diag(D)); % D is a vector of standard deviations now
    invsqrtC = B * diag(D.^(-1)) * B';
end
```
Evolution Strategies in a Nutshell

1. **Sampling** from a multi-variate normal distribution with maximum entropy

2. **Rank-based selection**: same performance on $g(f(x))$ for any $g$
   
   $g : \mathbb{R} \rightarrow \mathbb{R}$ strictly monotonic (order preserving)

3. **Step-size control**: converge log-linearly on the sphere function and many others possibly with linear scaling in dimension $n$

4. **Covariance matrix adaptation**: reduce any convex quadratic, $g$-transformed function
   
   $f(x) = g(x^T H x)$

   to the sphere function
   
   $f(x) = x^T x$

   without use of derivatives

   lines of equal density align with lines of equal fitness $C \propto H^{-1}$
Theoretical Foundations
Natural Gradient Descend

Consider \( \arg \min_{\theta} E(f(x)|\theta) \) under the sampling distribution \( p(.|\theta) \) we could improve \( E(f(x)|\theta) \) by following the gradient \( \nabla_{\theta} E(f(x)|\theta) \):

\[
\theta \leftarrow \theta - \eta \nabla_{\theta} E(f(x)|\theta), \quad \eta > 0
\]

\( \nabla_{\theta} \) depends on the parameterization of the distribution, specifically

Consider the **natural gradient** of the expected weighted fitness

\[
\nabla \hat{E}(w_g(f(x))|\theta) = F_{\theta}^{-1} \nabla_{\theta} E(w_g(f(x))|\theta)
\]

\[
= E(w_g(f(x))F_{\theta}^{-1} \nabla_{\theta} \ln p(x|\theta))
\]

using the Fisher information matrix \( F_{\theta} = \left( \frac{\partial^2 \log p(x|\theta)}{\partial \theta_i \partial \theta_j} \right)_{ij} \) of the density \( p \).

The natural gradient is **invariant under re-parameterization** of the distribution.

A **Monte-Carlo approximation** reads

\[
\nabla \hat{E}(\hat{w}(f(x))|\theta) = \sum_{i=1}^{\lambda} w_i F_{\theta}^{-1} \nabla_{\theta} \ln p(x_i:|\theta), \quad w_i = \hat{w}(f(x_i:)|\theta)
\]
CMA-ES = Natural Evolution Strategy + Cumulation

Natural gradient descend using the MC approximation and the normal distribution

- Rewriting the update of the distribution mean

\[ \mathbf{m}_{\text{new}} \leftarrow \sum_{i=1}^{\mu} w_i \mathbf{x}_{i: \lambda} = \mathbf{m} + \sum_{i=1}^{\mu} w_i (\mathbf{x}_{i: \lambda} - \mathbf{m}) \]

natural gradient for mean \( \frac{\partial}{\partial \mathbf{m}} \mathbb{E}(w_g(f(x)) | \mathbf{m}, \mathbf{C}) \)

- Rewriting the update of the covariance matrix\(^8\)

\[ \mathbf{C}_{\text{new}} \leftarrow \mathbf{C} + c_1 (\mathbf{p}_c \mathbf{p}_c^T - \mathbf{C}) \]

\[ + \frac{c_\mu}{\sigma^2} \sum_{i=1}^{\mu} w_i \left( (\mathbf{x}_{i: \lambda} - \mathbf{m})(\mathbf{x}_{i: \lambda} - \mathbf{m})^T - \sigma^2 \mathbf{C} \right) \]

natural gradient for covariance matrix \( \frac{\partial}{\partial \mathbf{C}} \mathbb{E}(w_g(f(x)) | \mathbf{m}, \mathbf{C}) \)

\(^8\) Akimoto et.al. (2010): Bidirectional Relation between CMA Evolution Strategies and Natural Evolution

Anne Auger & Nikolaus Hansen ()
Theoretical Foundations

Maximum Likelihood Update

The new distribution mean \( m \) maximizes the log-likelihood

\[
m_{\text{new}} = \arg \max_m \sum_{i=1}^{\mu} w_i \log p_N(x_i; \lambda | m)
\]

independently of the given covariance matrix

The rank-\( \mu \) update matrix \( C_\mu \) maximizes the log-likelihood

\[
C_\mu = \arg \max_C \sum_{i=1}^{\mu} w_i \log p_N \left( \frac{x_i; \lambda - m_{\text{old}}}{\sigma} \right | m_{\text{old}}, C)
\]

\[
\log p_N(x | m, C) = -\frac{1}{2} \log \det(2\pi C) - \frac{1}{2} (x - m)^T C^{-1} (x - m)
\]

\( p_N \) is the density of the multi-variate normal distribution.

Anne Auger & Nikolaus Hansen ()
CMA-ES
July, 2011 62 / 80
Variable Metric

On the function class

\[ f(x) = g \left( \frac{1}{2} (x - x^*) H (x - x^*)^T \right) \]

the covariance matrix approximates the inverse Hessian up to a constant factor, that is:

\[ C \propto H^{-1} \quad \text{(approximately)} \]

In effect, ellipsoidal level-sets are transformed into spherical level-sets.

\[ g : \mathbb{R} \rightarrow \mathbb{R} \text{ is strictly increasing} \]
On Convergence

Evolution Strategies converge with probability one on, e.g., \( g\left(\frac{1}{2}x^T H x\right) \) like

\[
\left\| m_k - x^* \right\| \propto e^{-ck}, \quad c \leq \frac{0.25}{n}
\]

Monte Carlo pure random search converges like

\[
\left\| m_k - x^* \right\| \propto k^{-c} = e^{-c \log k}, \quad c = \frac{1}{n}
\]
1. Problem Statement
2. Evolution Strategies
3. Step-Size Control
4. Covariance Matrix Adaptation
5. Theoretical Foundations
6. Experiments
7. Summary and Final Remarks
Experimentum Crucis (0)

What did we want to achieve?

- reduce any convex-quadratic function
  \[ f(x) = x^T H x \]
  e.g. \( f(x) = \sum_{i=1}^{n} 10^6 \frac{i-1}{n-1} x_i^2 \)
  to the sphere model
  \[ f(x) = x^T x \]
  without use of derivatives

- lines of equal density align with lines of equal fitness
  \[ C \propto H^{-1} \]
  in a stochastic sense
Experimentum Crucis (1)

\( f \) convex quadratic, separable

\[
f(x) = \sum_{i=1}^{n} 10^{\alpha \frac{i-1}{n-1}} x_i^2, \quad \alpha = 6
\]
Experimentum Crucis (2)

$f$ convex quadratic, as before but non-separable (rotated)

\[ f(x) = g(x^T H x), \quad g : \mathbb{R} \rightarrow \mathbb{R} \text{ strictly increasing} \]

\[ C \propto H^{-1} \text{ for all } g, H \]
Comparison to BFGS, NEWUOA, PSO and DE

$f$ convex quadratic, separable with varying condition number $\alpha$

Ellipsoid dimension 20, 21 trials, tolerance 1e−09, eval max 1e+07

- **BFGS** (Broyden et al 1970)
- **NEWUOA** (Powell 2004)
- **DE** (Storn & Price 1996)
- **PSO** (Kennedy & Eberhart 1995)
- **CMA-ES** (Hansen & Ostermeier 2001)

$f(x) = g(x^T H x)$ with

- $H$ diagonal
- $g$ identity (for BFGS and NEWUOA)
- $g$ any order-preserving = strictly increasing function (for all others)

**SP1** = average number of objective function evaluations\(^9\) to reach the target function value of $g^{-1}(10^{-9})$

---

\(^9\) Auger et.al. (2009): Experimental comparisons of derivative free optimization algorithms, SEA
Experiments

Comparison to BFGS, NEWUOA, PSO and DE

\( f \) convex quadratic, non-separable (rotated) with varying condition number \( \alpha \)

Rotated Ellipsoid dimension 20, 21 trials, tolerance 1e−09, eval max 1e+07

\[
f(x) = g(x^T H x) \text{ with } 
\]

- **BFGS** (Broyden et al 1970)
- **NEWUOA** (Powell 2004)
- **DE** (Storn & Price 1996)
- **PSO** (Kennedy & Eberhart 1995)
- **CMA-ES** (Hansen & Ostermeier 2001)

- \( g \) identity (for BFGS and NEWUOA)
- \( g \) any order-preserving = strictly increasing function (for all other)

\[ SP1 = \text{average number of objective function evaluations}^{10} \text{ to reach the target function value of } g^{-1}(10^{-9}) \]

---

10 Auger et.al. (2009): Experimental comparisons of derivative free optimization algorithms, SEA
Comparison to BFGS, NEWUOA, PSO and DE

$f$ non-convex, non-separable (rotated) with varying condition number $\alpha$

SQRT of SQRT of rotated ellipsoid dimension 20, 21 trials, tolerance 1e−09, eval max 1e+07

BFGS (Broyden et al 1970)
NEWUOA (Powell 2004)
DE (Storn & Price 1996)
PSO (Kennedy & Eberhart 1995)
CMA-ES (Hansen & Ostermeier 2001)

\[ f(x) = g(x^T H x) \text{ with} \]
\[ H \text{ full} \]
\[ g : x \mapsto x^{1/4} \text{ (for BFGS and NEWUOA)} \]
\[ g \text{ any order-preserving = strictly increasing function (for all other)} \]

SP1 = average number of objective function evaluations\(^{11}\) to reach the target function value of \(g^{-1}(10^{-9})\)

\(^{11}\) Auger et.al. (2009): Experimental comparisons of derivative free optimization algorithms, SEA
Comparison during BBOB at GECCO 2009

24 functions and 31 algorithms in 20-D

Anne Auger & Nikolaus Hansen
CMA-ES
July, 2011
Comparison during BBOB at GECCO 2010

24 functions and 20+ algorithms in 20-D
Comparison during BBOB at GECCO 2009

30 noisy functions and 20 algorithms in 20-D
Comparison during BBOB at GECCO 2010

30 noisy functions and 10+ algorithms in 20-D
Summary and Final Remarks
The Continuous Search Problem

**Difficulties** of a non-linear optimization problem are

- dimensionality and non-separability
  - demands to exploit problem structure, e.g. neighborhood

- ill-conditioning
  - demands to acquire a second order model

- ruggedness
  - demands a non-local (stochastic? population based?) approach
Main Features of (CMA) Evolution Strategies

1. Multivariate normal distribution to generate new search points follows the maximum entropy principle.

2. Rank-based selection implies invariance, same performance on \( g(f(x)) \) for any increasing \( g \) more invariance properties are featured.

3. Step-size control facilitates fast (log-linear) convergence and possibly linear scaling with the dimension based on an evolution path (a non-local trajectory).

4. **Covariance matrix adaptation (CMA)** increases the likelihood of previously successful steps and can improve performance by orders of magnitude. The update follows the natural gradient

   \[
   \mathbf{C} \propto \mathbf{H}^{-1} \iff \text{adapts a variable metric}
   \]

   \[
   \iff \text{new (rotated) problem representation}
   \]

   \[
   \implies f(x) = g(x^T \mathbf{H} x) \text{ reduces to } g(x^T x)
   \]
Limitations of CMA Evolution Strategies

- **internal CPU-time**: $10^{-8} n^2$ seconds per function evaluation on a 2GHz PC, tweaks are available. 1 000 000 $f$-evaluations in 100-D take 100 seconds internal CPU-time.

- Better methods are presumably available in case of:
  - partly separable problems
  - specific problems, for example with cheap gradients
  - small dimension ($n \ll 10$)
  - small running times (number of $f$-evaluations $\ll 100n$)

Specific methods:
- for example Nelder-Mead
- model-based methods
Source code for CMA-ES in C, Java, Matlab, Octave, Scilab, Python is available at

http://www.lri.fr/~hansen/cmaes_inmatlab.html