Evolution Strategies and Covariance Matrix Adaptation

Cours Contrôle Avancé - Ecole Centrale Paris

Anne Auger
January 31, 2013

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

Slides from A. Auger, N. Hansen GECCO 2013 Tutorial on ES and CMA-ES
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. CMA-ES Summary

6. Theoretical Foundations

7. Comparing Experiments

8. 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
  
  ... or to a robust solution $x$

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
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
  
- **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
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

Typical Examples

- shape optimization (e.g. using CFD)
- model calibration
- parameter calibration

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 $f : \mathcal{X} \subset \mathbb{R}^n \to \mathbb{R}$ to be non-linear, non-separable and to have at least moderate dimensionality, say $n \ll 10$.

Additionally, $f$ can be

- non-convex
- multimodal
- non-smooth
- discontinuous, plateaus
- ill-conditioned
- noisy
- ...

**Goal**: cope with any of these function properties they are related to real-world problems
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 \( n \lesssim 10 \). Additionally, \( f \) can be

- non-convex
- multimodal (there are possibly many local optima)
- non-smooth (derivatives do not exist)
- discontinuous, plateaus
- 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
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)

Consequence: a search policy (e.g. exhaustive search) that is valuable in small dimensions might be useless in moderate or large dimensional search spaces.
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)

Consequence: a search policy (e.g. exhaustive search) that is valuable in small dimensions might be useless in moderate or large dimensional search spaces.
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)

Consequence: a *search policy* (e.g. exhaustive search) that is valuable in small dimensions might be useless in moderate or large dimensional search spaces.
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)

Consequence: a *search policy* (e.g. exhaustive search) that is valuable in small dimensions *might be useless* in moderate or large dimensional search spaces.
Separable Problems

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

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

$\Rightarrow$ 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

\[ R \text{ rotation matrix} \]

---


---

---
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.
## What Makes a Function Difficult to Solve?

... and what can be done

<table>
<thead>
<tr>
<th>The Problem</th>
<th>Possible Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dimensionality</strong></td>
<td>exploiting the problem structure</td>
</tr>
<tr>
<td>Ill-conditioning</td>
<td>separability, locality/neighborhood, encoding</td>
</tr>
<tr>
<td>Ruggedness</td>
<td>second order approach</td>
</tr>
<tr>
<td></td>
<td>changes the neighborhood metric</td>
</tr>
<tr>
<td></td>
<td>non-local policy, large sampling width (step-size)</td>
</tr>
<tr>
<td></td>
<td>as large as possible while preserving a reasonable convergence speed</td>
</tr>
<tr>
<td></td>
<td>population-based method, stochastic, non-elitistic</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
### What Makes a Function Difficult to Solve?

...and what can be done

<table>
<thead>
<tr>
<th>The Problem</th>
<th>Possible Approaches</th>
</tr>
</thead>
</table>
| Dimensionality       | exploiting the problem structure
dergability, locality/neighborhood, encoding                                                   |
| Ill-conditioning     | second order approach                                                                                                                                     |
|                      | changes the neighborhood metric                                                                                                                       |
| Ruggedness           | non-local policy, large sampling width (step-size)                                                                                                     |
|                      | as large as possible while preserving a reasonable convergence speed                                                                                 |
|                      | population-based method, stochastic, non-elitistic                                                                                                      |
|                      | recombination operator                                                                                                                                     |
|                      | serves as repair mechanism                                                                                                                               |
|                      | restarts                                                                                                                                               |

...metaphors
## What Makes a Function Difficult to Solve?

... and what can be done

<table>
<thead>
<tr>
<th>The Problem</th>
<th>Possible Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimensionality</td>
<td>exploiting the problem structure</td>
</tr>
<tr>
<td></td>
<td>separability, 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>non-local policy, large sampling width (step-size)</td>
</tr>
<tr>
<td></td>
<td>as large as possible while preserving a reasonable convergence speed</td>
</tr>
<tr>
<td></td>
<td>population-based method, stochastic, non-elitistic</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/Nonlinear Programming</th>
</tr>
</thead>
<tbody>
<tr>
<td>individual, offspring, parent</td>
<td>candidate solution</td>
</tr>
<tr>
<td>population</td>
<td>decision variables</td>
</tr>
<tr>
<td>fitness function</td>
<td>design variables</td>
</tr>
<tr>
<td>generation</td>
<td>objective function</td>
</tr>
<tr>
<td></td>
<td>set of candidate solutions</td>
</tr>
<tr>
<td></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></td>
<td>iteration</td>
</tr>
</tbody>
</table>

...methods: ESs
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-µ Update

5. CMA-ES Summary

6. Theoretical Foundations

7. Comparing Experiments

8. Summary and Final Remarks
Stochastic Search

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 (incremental) Estimation of Distribution Algorithms
Stochastic Search

A black box search template to minimize \( f : \mathbb{R}^n \rightarrow \mathbb{R} \)

1. Initialize distribution parameters \( \theta \), set population size \( \lambda \in \mathbb{N} \)
2. While not terminate
   1. Sample distribution \( P(x|\theta) \rightarrow x_1, \ldots, x_\lambda \in \mathbb{R}^n \)
   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 (incremental) Estimation of Distribution Algorithms
A black box search template to minimize $f : \mathbb{R}^n \rightarrow \mathbb{R}$

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

While not terminate

1. Sample distribution $P(x|\theta) \rightarrow x_1, \ldots, x_\lambda \in \mathbb{R}^n$
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 (incremental) Estimation of Distribution Algorithms
Stochastic Search

A black box search template to minimize $f : \mathbb{R}^n \rightarrow \mathbb{R}$

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

1. Sample distribution $P(x|\theta) \rightarrow x_1, \ldots, x_\lambda \in \mathbb{R}^n$
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 (incremental) Estimation of Distribution Algorithms
Evolution Strategies

A Search Template

Stochastic Search

A black box search template to minimize \( f : \mathbb{R}^n \rightarrow \mathbb{R} \)

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

While not terminate

1. Sample distribution \( P(x|\theta) \rightarrow x_1, \ldots, x_\lambda \in \mathbb{R}^n \)
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 (incremental) Estimation of Distribution Algorithms
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 $P(x|\theta) \rightarrow x_1, \ldots, x_\lambda \in \mathbb{R}^n$
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 (incremental) Estimation of Distribution Algorithms
Stochastic Search

A black box search template to minimize $f : \mathbb{R}^n \rightarrow \mathbb{R}$

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

While not terminate

1. Sample distribution $P(x|\theta) \rightarrow x_1, \ldots, x_\lambda \in \mathbb{R}^n$
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 (incremental) Estimation of Distribution Algorithms
A black box search template to minimize $f : \mathbb{R}^n \rightarrow \mathbb{R}$

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

While not terminate

1. Sample distribution $P(x|\theta) \rightarrow x_1, \ldots, x_{\lambda} \in \mathbb{R}^n$
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 (incremental) *Estimation of Distribution Algorithms*
The CMA-ES

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

$sampling$

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

$update mean$

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

$cumulation for C$

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

$cumulation for \sigma$

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

$update C$

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

$update of \sigma$

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

Not covered on this slide: termination, restarts, useful output, boundaries and encoding
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 step length
- the covariance matrix \( C \in \mathbb{R}^{n \times n} \) determines the shape of the distribution ellipsoid

here, all new points are sampled with the same parameters

The question remains how to update \( m, C, \) and \( \sigma \).
Evolution Strategies

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

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 that the sum of normal variates is again normal:
   \[ \mathcal{N}(x, A) + \mathcal{N}(y, B) \sim \mathcal{N}(x + y, A + B) \]
   helpful in design and analysis of algorithms related to the central limit theorem
3. most convenient way to generate isotropic search points
   the isotropic distribution does not favor any direction, rotational invariant
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
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 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
\[
\{ x \in \mathbb{R}^n | (x - m)^T C^{-1} (x - m) = 1 \}
\]

**Lines of Equal Density**

\[ N(m, \sigma^2 I) \sim m + \sigma N(0, I) \]

one degree of freedom \( \sigma \)

components are independent standard normally distributed

\[ N(m, D^2) \sim m + D N(0, I) \]

\( n \) degrees of freedom components are independent, scaled

\[ N(m, C) \sim m + C^{\frac{1}{2}} N(0, I) \]

\( (n^2 + n)/2 \) degrees of freedom components are correlated

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

\[ \mathcal{N}(m, \sigma^2 I) \sim m + \sigma \mathcal{N}(0, I) \]

-one degree of freedom \( \sigma \)

components are independent standard normally distributed

\[ \mathcal{N}(m, D^2) \sim m + D \mathcal{N}(0, I) \]

- \( n \) degrees of freedom

components are independent, scaled

\[ \mathcal{N}(m, C) \sim m + C^{1/2} \mathcal{N}(0, I) \]

- \( n^2 + n \)/2 degrees of freedom

components are correlated

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 \( A \).
...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

\[ \mathcal{N}(m, \sigma^2 I) \sim m + \sigma \mathcal{N}(0, I) \]

one degree of freedom \( \sigma \)

components are independent standard normally distributed

\[ \mathcal{N}(m, D^2) \sim m + D \mathcal{N}(0, I) \]

\( n \) degrees of freedom

components are independent, scaled

\[ \mathcal{N}(m, C) \sim m + C^{\frac{1}{2}} \mathcal{N}(0, I) \]

\( (n^2 + n)/2 \) degrees of freedom

components are correlated

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 \( A \).
Effect of Dimensionality

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

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

yet: maximum entropy distribution
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 $m$

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

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 $x_i = m + \sigma N_i(0, C) = m + \sigma y_i$

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

$$m \leftarrow \sum_{i=1}^{\mu} w_i x_i:\lambda$$

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.
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

$$m \leftarrow \sum_{i=1}^{\mu} w_i x_{i:\lambda}$$

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.
The \((\mu/\mu, \lambda)\)-ES

Non-elitist selection and intermediate (weighted) recombination

Given the \(i\)-th solution point \(x_i = m + \sigma N_i(0, C) = m + \sigma y_i =: y_i\)

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

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

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

\[ f(x_1;\lambda) \leq f(x_2;\lambda) \leq \ldots \leq f(x_\lambda;\lambda) \]

\[ g(f(x_1;\lambda)) \leq g(f(x_2;\lambda)) \leq \ldots \leq g(f(x_\lambda;\lambda)) \quad \forall g \]

\( g \) 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

is true for most optimization algorithms

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

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

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

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

- Invariance to orthogonal (rigid) transformations $R$, where $RR^T = I$
  
  - e.g. true for simple evolution strategies
  
  - Recombination operators might jeopardize rotational invariance

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

Identical behavior on $f$ and $f_R$

\[
\begin{align*}
  f &: \quad x \mapsto f(x), \quad x^{(i=0)} = x_0 \\
  f_R &: \quad x \mapsto f(Rx), \quad x^{(i=0)} = R^{-1}(x_0)
\end{align*}
\]

No difference can be observed w.r.t. the argument of $f$

---


Invariance

*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 CMA-ES Summary

6 Theoretical Foundations

7 Comparing Experiments

8 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?

(1+1)-ES (red & green)

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

in \([-2.2, 0.8]^n\] for \(n = 10\)
Why Step-Size Control?

(5/5w, 10)-ES, 11 runs

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

for \( n = 10 \) and \( x^0 \in [-0.2, 0.8]^n \)

with optimal step-size \( \sigma \)
Why Step-Size Control?

(5/5\textsubscript{w}, 10)-ES, 2\times11 runs

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

for \( n = 10 \) and \( x^0 \in [-0.2, 0.8]^n \)

with optimal versus adaptive step-size \( \sigma \) with too small initial \( \sigma \)
Why Step-Size Control?

\((5/5w, 10)-ES\)

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

for \(n = 10\) and \(x^0 \in [-0.2, 0.8]^n\)

comparing number of \(f\)-evals to reach \(\|m\| = 10^{-5}: \frac{1100 - 100}{650} \approx 1.5\)
Why Step-Size Control?

$(5/5_w, 10)$-ES

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

in $[-0.2, 0.8]^n$

for $n = 10$

Comparing optimal versus default damping parameter $d_\sigma$: $\frac{1700}{1100} \approx 1.5$
Why Step-Size Control?

\[ \sigma \leftarrow \sigma_{opt} \parallel \text{parent} \parallel \]

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

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

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

- **\(\sigma\)-self-adaptation**\(^b\), 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

\( p_s : \) number of successful offspring / number of 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 $c_\sigma \approx 4/n$, $d_\sigma \approx 1$.

- $m \leftarrow m + \sigma y_w$, where $y_w = \sum_{i=1}^{\mu} w_i y_{i:}\lambda$, update mean
- $p_\sigma \leftarrow (1 - c_\sigma) p_\sigma + \sqrt{1 - (1 - c_\sigma)^2} \sqrt{\mu_w} y_w$, accounts for $1 - c_\sigma$ and $w_i$, update evolution path
- $\sigma \leftarrow \sigma \times \exp \left( \frac{c_\sigma}{d_\sigma} \left( \frac{\|p_\sigma\|}{E\|\mathcal{N}(0, I)\|} - 1 \right) \right)$, update step-size

$> 1 \iff \|p_\sigma\|$ is greater than its expectation
Path Length Control (CSA)

The Equations

Initialize $\mathbf{m} \in \mathbb{R}^n$, $\sigma \in \mathbb{R}_+$, evolution path $\mathbf{p}_\sigma = 0$.

Set $c_\sigma \approx 4/n$, $d_\sigma \approx 1$.

\[
\mathbf{m} \leftarrow \mathbf{m} + \sigma \mathbf{y}_w \quad \text{where} \quad \mathbf{y}_w = \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda} \\
\mathbf{p}_\sigma \leftarrow (1 - c_\sigma) \mathbf{p}_\sigma + \sqrt{1 - (1 - c_\sigma)^2} \sqrt{\mu_w} \mathbf{y}_w \\
\sigma \leftarrow \sigma \times \exp \left( \frac{c_\sigma}{d_\sigma} \left( \frac{\|\mathbf{p}_\sigma\|}{\mathbb{E}\|\mathcal{N}(0, I)\|} - 1 \right) \right) \quad \text{update step-size}
\]

$>1 \iff \|\mathbf{p}_\sigma\|$ is greater than its expectation.

update mean

accounts for $1 - c_\sigma$

accounts for $\mu_w$

accounts for $w_i$
(5/5, 10)-CSA-ES, default parameters

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

in \([-0.2, 0.8]^n\) for \(n = 30\)
Covariance Matrix Adaptation

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. CMA-ES Summary
6. Theoretical Foundations
7. Comparing Experiments
8. Summary and Final Remarks
Evolution Strategies

Recalling

New search points are sampled normally distributed

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

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

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

initial distribution, \( \mathbf{C} = \mathbf{I} \)
Covariance Matrix Adaptation

Rank-One Update

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

initial distribution, \( C = I \)
Covariance Matrix Adaptation

Rank-One Update

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

\( y_w \), movement of the population mean \( m \) (disregarding \( \sigma \))
Covariance Matrix Adaptation

Rank-One Update

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

mixture of distribution \( C \) and step \( y_w \),
\[ C \leftarrow 0.8 \times C + 0.2 \times y_w y_w^T \]
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 (disregarding $\sigma$)
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 (disregarding \( \sigma \))
Covariance Matrix Adaptation

Rank-One Update

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

\[
y_w = \sum_{i=1}^{\mu} w_i y_i : \lambda,
\]

\[
y_i \sim \mathcal{N}_i(0, C)
\]

movement of the population mean \(m\)
Covariance Matrix Adaptation

Rank-One Update

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

mixture of distribution \( C \) and step \( y_w \),

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

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

$$m \leftarrow m + \sigma y_w$$

where $y_w = \sum_{i=1}^{\mu} w_i y_{i:}\lambda$

$$C \leftarrow (1 - c_{\text{cov}})C + c_{\text{cov}} \mu_w \left[ y_w y_w^T \right]_\text{rank-one}$$

where $\mu_w = \frac{1}{\sum_{i=1}^{\mu} w_i^2} \geq 1$

The rank-one update has been found independently in several domains\textsuperscript{6 7 8 9}

\textsuperscript{6} Kjellström\&Taxén 1981. Stochastic Optimization in System Design, IEEE TCS
\textsuperscript{7} Hansen\&Ostermeier 1996. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation, ICEC
\textsuperscript{8} Ljung 1999. System Identification: Theory for the User
\textsuperscript{9} Haario et al 2001. An adaptive Metropolis algorithm, JSTOR
CMA-ES

Covariance Matrix Adaptation

Covariance Matrix Rank-One Update

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

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

- learns a new rotated problem representation
  - components are independent (only) in the new representation

- learns a new (Mahalanobis) metric

- approximates the inverse Hessian on quadratic functions
  - transformation into the sphere function

- for \( \mu = 1 \): conducts a natural gradient ascent on the distribution \( \mathcal{N} \)
  - entirely independent of the given coordinate system
## Problem Statement

## Evolution Strategies

## Step-Size Control

## Covariance Matrix Adaptation

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

## CMA-ES Summary

## Theoretical Foundations

## Comparing Experiments

## Summary and Final Remarks
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

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

exponentially fading weights

The recursive construction of the evolution path (cumulation):

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

decay factor

normalization factor

input $= \frac{m - m_{old}}{\sigma}$

where $\mu_w = \frac{1}{\sum w_i^2}$, $c_c \ll 1$. History information is accumulated in the 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:

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

exponentially fading weights

The recursive construction of the evolution path (cumulation):

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

decay factor

normalization factor

input $= \frac{m - m_{\text{old}}}{\sigma}$

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 *moving average*
- *iterate averaging* in stochastic approximation
- *momentum* in the back-propagation algorithm for ANNs
- ...

“Cumulation” conducts a *low-pass* filtering, but there is more to it...
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 (signifying correlation between steps) is (re-)introduced by using the evolution path.

$$
p_{c} \leftarrow \underbrace{(1 - c_c)}_{\text{decay factor}} p_{c} + \underbrace{\sqrt{1 - (1 - c_c)^2}}_{\text{normalization factor}} \sqrt{\mu_w} y_w
$$

$$
C \leftarrow \underbrace{(1 - c_{cov})}_{} C + \underbrace{c_{cov}}_{\text{rank-one}} p_{c} p_{c}^T
$$

where $\mu_w = \frac{1}{\sum w_i^2}$, $c_{cov} \ll c_c \ll 1$ such that $1/c_c$ is the “backward time horizon”.
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 (signifying correlation between steps) 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 $\mu_w = \frac{1}{\sum w_i^2}$, $c_{cov} \ll c_c \ll 1$ such that $1/c_c$ is the “backward time horizon”.

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** (signifying correlation *between* steps) is (re-)introduced by using the *evolution path*.

\[
\begin{align*}
p_c & \leftarrow (1 - c_e) p_c + \sqrt{1 - (1 - c_e)^2} \sqrt{\mu_w} y_w \\
C & \leftarrow (1 - c_{cov}) C + c_{cov} p_c p_c^T
\end{align*}
\]

where $\mu_w = \frac{1}{\sum w_i^2}$, $c_{cov} \ll c_e \ll 1$ such that $1 / c_e$ is the “backward time horizon”.
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 about $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$.\(^{(a)}\)


Number of $f$-evaluations divided by dimension on the cigar function $f(x) = x_1^2 + 10^6 \sum_{i=2}^{n} x_i^2$

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

Rank-$\mu$ Update

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

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 weighted empirical covariance 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.

with $\mu = \lambda$ weights can be negative $^{10}$

The rank-$\mu$ update then reads

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

where $c_{cov} \approx \mu w / n^2$ and $c_{cov} \leq 1$. 

Rank-\(\mu\) Update

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

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 weighted empirical covariance 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.

with \(\mu = \lambda\) weights can be negative

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 \(c_{\text{cov}} \leq 1\).
Rank-$\mu$ Update

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

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 weighted empirical covariance 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.

with $\mu = \lambda$ weights can be negative

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 $c_{\text{cov}} \leq 1$.

Jastrebski and Arnold (2006). Improving evolution strategies through active covariance matrix adaptation. CEC.
Covariance Matrix Adaptation

Covariance Matrix Rank-μ Update

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

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

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

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

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

\[ C \leftarrow \frac{1}{\mu} \sum (x_i; \lambda - m_{old})(x_i; \lambda - m_{old})^T \]

\[ m_{\text{new}} = m_{old} + \frac{1}{\mu} \sum y_i; \lambda \]

**Sampling of $\lambda = 150$ solutions (dots)**

**Calculating $C$ from $\mu = 50$ solutions**

**New distribution**

$m_{\text{new}}$ 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 $O(n)$ \(^{(12)}\)

Therefore the rank-$\mu$ update is the primary mechanism whenever a large population size is used

\[ \text{say } \lambda \geq 3n + 10 \]

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

---

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 $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ \(^{(12)}\)

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 $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$.

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

---

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 $O(n)$ \(^{(12)}\)

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:** \( C = I, \text{ and } p_c = 0, p_\sigma = 0, \)

**Set:** \( c_c \approx 4/n, c_\sigma \approx 4/n, c_1 \approx 2/n^2, c_\mu \approx \mu_w/n^2, c_1 + c_\mu \leq 1, d_\sigma \approx 1 + \sqrt{\frac{\mu_w}{n}}, \)

and \( w_i = 1 \ldots \lambda \) such that \( \mu_w = \frac{1}{\sum_{i=1}^{\lambda} w_i^2} \approx 0.3 \lambda \)

**While not terminate**

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

\[
m \leftarrow \sum_{i=1}^{\lambda} w_i x_i : \lambda = m + \sigma y_w \quad \text{where } y_w = \sum_{i=1}^{\lambda} w_i y_i : \lambda
\]

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

\[
p_\sigma \leftarrow (1 - c_\sigma) p_\sigma + \sqrt{1 - (1 - c_\sigma)^2} \sqrt{\mu_w} C^{-1/2} y_w
\]

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

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

Not covered on this slide: termination, restarts, useful output, boundaries and encoding
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 = arx(:,arindex(1:mu)) * weights; % recombination, new mean value 

% Cumulation: Update evolution paths 
ps = (1-cs)'*ps ... 
    + sqrt(cs*(2-cs)*mueff) * invsqrtC * (xmean-xold) / sigma; 
hsig = norm(ps)/sqrt(1-(1-cs)^2*(2*counteval/lambda))/chIN < 1.4 + 2/(N+1); 
pC = (1-cC)'*pc ... 
    + hsig * sqrt(cC*(2-cC)*mueff) * (xmean-xold) / sigma; 

% Adapt covariance matrix C 
artmp = (1/sigma) * (arx(:,arindex(1:mu)) - remat(xold,1,mu)); 
C = (1-c1-cmu) * C ... % regard old matrix 
    + c1 * (pc'*pc) ... % plus rank one update 
    + (1-hsig) * cmu*(2-cmu)*C ... % minor correction if hsig==0 
    + cmu * artmp * diag(weights') * artmp' ; % 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 - eigeneval > lambda/(c1+cmu)/N/10 % to achieve O(N^2) 
eigeneval = 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';
Strategy Internal Parameters

- related to selection and recombination
  - $\lambda$, offspring number, new solutions sampled, population size
  - $\mu$, parent number, solutions involved in updates of $m, C, \sigma$
  - $w_{i=1,...,\mu}$, recombination weights

- related to $C$-update
  - $c_{c}$, decay rate for the evolution path
  - $c_{1}$, learning rate for rank-one update of $C$
  - $c_{\mu}$, learning rate for rank-$\mu$ update of $C$

- related to $\sigma$-update
  - $c_{\sigma}$, decay rate of the evolution path
  - $d_{\sigma}$, damping for $\sigma$-change

Parameters were identified in carefully chosen experimental set ups. Parameters do not in the first place depend on the objective function and are not meant to be in the users choice. Only(?) the population size $\lambda$ might be reasonably varied in a wide range, depending on the objective function.

Useful: restarts with increasing population size (IPOP)
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)**

\[ f \text{ convex quadratic, separable} \]

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

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

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

\[
C \propto H^{-1} \text{ for all } g, H
\]
1. Problem Statement
2. Evolution Strategies
3. Step-Size Control
4. Covariance Matrix Adaptation
5. CMA-ES Summary
6. Theoretical Foundations
7. Comparing Experiments
8. Summary and Final Remarks
Consider arg min \( \theta \) \( E(f(x)|\theta) \) under the sampling distribution \( x \sim p(.|\theta) \)
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, therefore

Consider the natural gradient of the expected transformed fitness
\[
\tilde{\nabla}_{\theta} E(w \circ P_f(f(x))|\theta) = F_{\theta}^{-1} \nabla_{\theta} E(w \circ P_f(f(x))|\theta)
= E(w \circ P_f(f(x))F_{\theta}^{-1} \nabla_{\theta} \ln p(x|\theta))
\]
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
\[
\tilde{\nabla}_{\theta} \hat{E}(\hat{w}(f(x))|\theta) = \sum_{i=1}^{\lambda} w_i F_{\theta}^{-1} \nabla_{\theta} \ln p(x_i;|\theta), \quad w_i = \hat{w}(f(x_i;|\theta))
\]
Consider $\arg\min_{\theta} \mathbb{E}(f(x)|\theta)$ under the sampling distribution $x \sim p(\cdot|\theta)$ we could improve $\mathbb{E}(f(x)|\theta)$ by following the gradient $\nabla_{\theta} \mathbb{E}(f(x)|\theta)$:

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

$\nabla_{\theta}$ depends on the parameterization of the distribution, therefore consider the natural gradient of the expected transformed fitness

$$\tilde{\nabla}_{\theta} \mathbb{E}(w \circ P_f(f(x))|\theta) = F_{\theta}^{-1} \nabla_{\theta} \mathbb{E}(w \circ P_f(f(x))|\theta)$$

$$= \mathbb{E}(w \circ P_f(f(x)) F_{\theta}^{-1} \nabla_{\theta} \ln p(x|\theta))$$

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

$$\tilde{\nabla}_{\theta} \mathbb{E}(\hat{w}(f(x))|\theta) = \sum_{i=1}^{\lambda} w_i F_{\theta}^{-1} \nabla_{\theta} \ln p(x_i;\lambda|\theta), \quad w_i = \hat{w}(f(x_i;\lambda)|\theta)$$
Consider $\arg \min_{\theta} E(f(x)|\theta)$ under the sampling distribution $x \sim p(\cdot|\theta)$ 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, therefore

Consider the natural gradient of the expected transformed fitness

$$\tilde{\nabla}_{\theta} E(w \circ P_f(f(x))|\theta) = F_{\theta}^{-1} \nabla_{\theta} E(w \circ P_f(f(x))|\theta)$$

$$= E(w \circ P_f(f(x)) F_{\theta}^{-1} \nabla_{\theta} \ln p(x|\theta))$$

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

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

A Monte-Carlo approximation reads

$$\tilde{\nabla}_{\theta} \hat{E}(\hat{w}(f(x))|\theta) = \sum_{i=1}^{\lambda} w_i F_{\theta}^{-1} \nabla_{\theta} \ln p(x_i;\lambda|\theta), \quad w_i = \hat{w}(f(x_i;\lambda)|\theta)$$
Natural Gradient Descend

Consider \( \arg \min_{\theta} \mathbb{E}(f(x)|\theta) \) under the sampling distribution \( x \sim p(.|\theta) \)
we could improve \( \mathbb{E}(f(x)|\theta) \) by following the gradient \( \nabla_{\theta} \mathbb{E}(f(x)|\theta) \):
\[
\theta \leftarrow \theta - \eta \nabla_{\theta} \mathbb{E}(f(x)|\theta), \quad \eta > 0
\]
\( \nabla_{\theta} \) depends on the parameterization of the distribution, therefore

Consider the natural gradient of the expected transformed fitness
\[
\tilde{\nabla}_{\theta} \mathbb{E}(w \circ P_{f}(f(x))|\theta) = F_{\theta}^{-1} \nabla_{\theta} \mathbb{E}(w \circ P_{f}(f(x))|\theta)
\]
\[
= \mathbb{E}(w \circ P_{f}(f(x))F_{\theta}^{-1} \nabla_{\theta} \ln p(x|\theta))
\]
using the Fisher information matrix \( F_{\theta} = \left( \mathbb{E} \frac{\partial^2 \log p(x|\theta)}{\partial \theta_i \partial \theta_j} \right)_{ij} \) of the density \( p \).
The natural gradient is invariant under re-parameterization of the distribution.

A Monte-Carlo approximation reads
\[
\tilde{\nabla}_{\theta} \hat{\mathbb{E}}(\hat{w}(f(x))|\theta) = \sum_{i=1}^{\lambda} w_i F_{\theta}^{-1} \nabla_{\theta} \ln p(x_{i:\lambda}|\theta), \quad w_i = \hat{w}(f(x_{i:\lambda})|\theta)
\]
Consider \( \arg \min_{\theta} E(f(x) | \theta) \) under the sampling distribution \( x \sim p( . | \theta) \) 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, therefore

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

\[
\tilde{\nabla}_{\theta} E(w \circ P_f(f(x)) | \theta) = F_{\theta}^{-1} \nabla_{\theta} E(w \circ P_f(f(x)) | \theta)
\]

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

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

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

A **Monte-Carlo approximation** reads

\[
\tilde{\nabla}_{\theta} \hat{E}(\hat{w}(f(x)) | \theta) = \sum_{i=1}^{\lambda} w_i F_{\theta}^{-1} \nabla_{\theta} \ln p(x_i: \lambda | \theta), \quad w_i = \hat{w}(f(x_i: \lambda) | \theta)
\]
Rewriting the update of the distribution mean

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

natural gradient for mean \( \frac{\partial}{\partial m} \widehat{E}(w \circ P_f(f(x))|m, C) \)

Rewriting the update of the covariance matrix\(^\text{13}\)

\[ C_{\text{new}} \leftarrow C + c_1 (p_c p_c^T - C) \]

\[ + \frac{c_2}{\sigma^2} \sum_{i=1}^{\mu} w_i \left( (x_{i:}\lambda - m)(x_{i:}\lambda - m)^T - \sigma^2 C \right) \]

natural gradient for covariance matrix \( \frac{\partial}{\partial C} \widehat{E}(w \circ P_f(f(x))|m, C) \)

\(^\text{13}\) Akimoto et.al. (2010): Bidirectional Relation between CMA Evolution Strategies and Natural Evolution
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

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

$p_N$ is the density of the multi-variate normal distribution
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

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

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

$p_N$ is the density of the multi-variate normal distribution
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:

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

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

\[ g : \mathbb{R} \rightarrow \mathbb{R} \text{ is strictly increasing} \]
Evolution Strategies converge with probability one on, e.g., $g\left(\frac{1}{2}x^THx\right)$ like

$$\|m_k - x^*\| \propto e^{-ck}, \quad c \leq \frac{0.25}{n}$$

Monte Carlo pure random search converges like

$$\|m_k - x^*\| \propto k^{-c} = e^{-c \log k}, \quad c = \frac{1}{n}$$
Comparing Experiments

1. Problem Statement
2. Evolution Strategies
3. Step-Size Control
4. Covariance Matrix Adaptation
5. CMA-ES Summary
6. Theoretical Foundations
7. Comparing Experiments
8. Summary and Final Remarks
Comparing Experiments

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

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>1000</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>10000</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>100000</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>1000000</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>

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

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

SP1 = average number of objective function evaluations\(^{14}\) to reach the target function value of \( g^{-1}(10^{-9}) \)

\(^{14}\) Auger et.al. (2009): Experimental comparisons of derivative free optimization algorithms, SEA
Comparing 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 $1e+07$

Condition number

SP1

NEWUOA
BFGS
DE
PSO
CMAES

$BFGS$ (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

$H$ full

$g$ identity (for BFGS and NEWUOA)

$g$ any order-preserving = strictly increasing function (for all other)

$SP1 = \text{average number of objective function evaluations}^{15} \text{ to reach the target function value of } g^{-1}(10^{-9})$

---

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

Comparison to BFGS, NEWUOA, PSO and DE

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

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

$H$ full

$g: x \mapsto x^{1/4}$ (for BFGS and NEWUOA)

g any order-preserving = strictly increasing function (for all other)

SP1 = average number of objective function evaluations\(^{16}\) to reach the target function value of $g^{-1}(10^{-9})$

---

\(^{16}\) 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
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

Proportion of functions

Running length / dimension
Comparison during BBOB at GECCO 2010

30 noisy functions and 10+ algorithms in 20-D
1. Problem Statement
2. Evolution Strategies
3. Step-Size Control
4. Covariance Matrix Adaptation
5. CMA-ES Summary
6. Theoretical Foundations
7. Comparing Experiments
8. 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
cave: design of benchmark functions

- ill-conditioning
  demands to acquire a second order model

- ruggedness
  demands a non-local (stochastic? population based?) approach
Main Characteristics 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 in CMA-ES 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

\[
C \propto H^{-1} \iff \text{adapts a variable metric} \\
\iff \text{new (rotated) problem representation} \\
\iff f : x \mapsto g(x^T H x) \text{ reduces to } x \mapsto x^T x
\]
**Limitations of CMA Evolution Strategies**

- **internal CPU-time:** $10^{-8} n^2$ seconds per function evaluation on a 2GHz PC, tweaks are available. 1 million $f$-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 specific methods
  - small dimension ($n \ll 10$) for example Nelder-Mead
  - small running times (number of $f$-evaluations $< 100n$) model-based methods
Thank You

Source code for CMA-ES in C, Java, Matlab, Octave, Python, Scilab is available at http://www.lri.fr/~hansen/cmaes_inmatlab.html