Tutorial—Evolution Strategies and Related Estimation of Distribution Algorithms

Anne Auger & Nikolaus Hansen

INRIA Saclay - Ile-de-France, project team TAO
Universite Paris-Sud, LRI, Bat. 490
91405 ORSAY Cedex, France

PPSN 2008, September 14, 2008, Dortmund, Germany.
Content

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

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

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

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

5. Experiments

6. Summary
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** with **least search cost**
  - ...or to a robust solution $x$
  - 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
## Metaphors

<table>
<thead>
<tr>
<th>Evolutionary Computation</th>
<th>Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>individual, offspring, parent $\leftrightarrow$ candidate solution</td>
<td></td>
</tr>
<tr>
<td>population               $\leftrightarrow$ set of candidate solutions</td>
<td></td>
</tr>
<tr>
<td>fitness function         $\leftrightarrow$ objective function</td>
<td></td>
</tr>
<tr>
<td>generation               $\leftrightarrow$ iteration</td>
<td></td>
</tr>
<tr>
<td>decision variables       $\leftrightarrow$ design variables</td>
<td></td>
</tr>
<tr>
<td>design variables         $\leftrightarrow$ object variables</td>
<td></td>
</tr>
<tr>
<td>object variables         $\leftrightarrow$ cost function</td>
<td></td>
</tr>
<tr>
<td>function properties      $\leftrightarrow$ loss function</td>
<td></td>
</tr>
</tbody>
</table>
We assume \( f : \mathcal{X} \subset \mathbb{R}^n \rightarrow \mathbb{R} \) to have at least moderate dimensionality, say \( n \ll 10 \), and to be non-linear, non-convex, and non-separable. Additionally, \( f \) can be

- multimodal
  - there are eventually 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?

- **ruggedness**
  - non-smooth, discontinuous, multimodal, and/or noisy function
- **dimensionality**
  - (considerably) larger than three
- **non-separability**
  - dependencies between the objective variables
- **ill-conditioning**

![Plot](image.png)  
**Cut from 3-D example, solvable with an evolution strategy**

![Graph](image.png)  
**A narrow ridge**
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.

Consequently, 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

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

⇒ 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

\[ f : x \mapsto f(x) \text{ separable} \]
\[ f : x \mapsto f(Rx) \text{ non-separable} \]

\( R \) rotation matrix

---


Ill-Conditioned Problems

- If $f$ is convex quadratic, $f : x \mapsto \frac{1}{2} x^T H x (= \frac{1}{2} \sum_i h_{i,i} x_i^2 + \frac{1}{2} \sum_{i \neq j} h_{i,j} x_i x_j)$, with $H$ positive, definite, symmetric matrix

- ill-conditioned means a high condition number of Hessian Matrix $H$

ill-conditioned means “squeezed” lines of equal function value

- consider the curvature of iso-fitness lines

---

$H$ is Hessian matrix of $f$
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>Ruggedness</td>
<td>non-local policy, large sampling width (step-size) as large as possible while preserving a reasonable convergence speed</td>
</tr>
<tr>
<td></td>
<td>stochastic, non-elitistic, population-based method</td>
</tr>
<tr>
<td></td>
<td>recombination operator serves as repair mechanism</td>
</tr>
<tr>
<td>Dimensionality, Non-Separability</td>
<td>exploiting the problem structure locality, neighborhood, encoding</td>
</tr>
<tr>
<td>Ill-conditioning</td>
<td>second order approach changes the neighborhood metric</td>
</tr>
</tbody>
</table>
1 Problem Statement
   - Black Box Optimization and Its Difficulties
   - Non-Separable Problems
   - Ill-Conditioned Problems

2 Evolution Strategies and EDAs
   - A Search Template
   - The Normal Distribution
   - Invariance

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

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

5 Experiments

6 Summary
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 Evolutionary Algorithms the distribution \( P \) is often implicitly defined
via **operators on a population**, in particular, selection, recombination
and mutation

Natural template for *Estimation of Distribution Algorithms*
Evolution Strategies and Normal Estimation of Distribution Algorithms

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 \( x_i, m \in \mathbb{R}^n, \sigma \in \mathbb{R}_+, \text{ and } C \in \mathbb{R}^{n \times n} \)

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 \( m, C, \text{ and } \sigma \).
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
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
The Multi-Variate \((n\text{-Dimensional})\) Normal Distribution

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

The **mean** value \(\mathbf{m}\)

- determines the displacement (translation)
- is the value with the largest density (modal value)
- the distribution is symmetric about the distribution mean
The covariance matrix $C$ determines the shape. It has a valuable geometrical interpretation: any covariance matrix can be uniquely identified with the iso-density ellipsoid $\{x \in \mathbb{R}^n \mid x^T C^{-1} x = 1\}$

Lines of Equal Density

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

one degree of freedom $\sigma$

components of $\mathcal{N}(0, I)$

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
Evolution Strategies

\[(\mu \div \lambda) \mu: \# \text{parents}, \lambda: \# \text{offspring}\]

+ selection in \{\text{parents}\} \cup \{\text{offspring}\}
, selection in \{\text{offspring}\}

\[(1 + 1)\text{-ES}\]

Sample one offspring from parent \(m\)

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

If \(x\) better than \(m\) select

\[m \leftarrow x\]

...why?
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

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

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

Motivation

- 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 the feasibility of generalization
  generalizing (identical) performance from a single function to a whole class of functions

consequently, invariance is important for the evaluation of search algorithms
Invariance Under Strictly Monotonically Increasing Functions

Rank-based algorithms

Selection based on the rank:

\[ f(x_{1:λ}) \leq f(x_{2:λ}) \leq \ldots \leq f(x_{λ:λ}) \]

Update of all parameters uses only the rank

\[ g(f(x_{1:λ})) \leq g(f(x_{2:λ})) \leq \ldots \leq g(f(x_{λ:λ})) \]
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 an orthogonal transformation \( R \), 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 \( f_R \)

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

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


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

2 Evolution Strategies and EDAs
   - A Search Template
   - The Normal Distribution
   - Invariance

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

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

5 Experiments

6 Summary
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?

The diagram illustrates the performance of different methods in minimizing a function. The function is given by:

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

in the range \([-0.2, 0.8]^n\) for \(n = 10\). The plot shows the function evaluations on the x-axis and the function value on the y-axis. The lines represent different methods:

- **Random Search**
- **Constant \(\sigma\)**
- **Optimal Step-Size (Scale Invariant)**

The optimal step-size is scale invariant, while the constant \(\sigma\) method does not adapt to the scale of the problem, leading to suboptimal performance compared to random search and the scale-invariant optimal step-size method.
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?

The evolution window

The evolution window for the step-size on the sphere function

*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 one, according to the objective function value, is selected

  simplified “global” self-adaptation

- **path length control**\(^d\) (Cumulative Step-size Adaptation, CSA)\(^e\), applied with “,”-selection

---


\(b\) 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

Let $p_s$: # of successful offspring / # offspring (per generation)

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

Increase $\sigma$ if $p_s > p_{target}$
Decrease $\sigma$ if $p_s < p_{target}$

(1 + 1)-ES

$p_{target} = 1/5$

IF offspring better parent

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

ELSE

$p_s = 0$, $\sigma \leftarrow \sigma / \exp(1/3)$
Self-adaptation in a $(1, \lambda)$-ES

MUTATE for $i = 1, \ldots \lambda$

\begin{align*}
\text{step-size} & \quad \sigma_i \leftarrow \sigma \exp(\tau N_i(0, 1)) \\
\text{parent} & \quad x_i \leftarrow x + \sigma_i N(0, I)
\end{align*}

EVALUATE

SELECT

Best offspring $x_*$ with its step-size $\sigma_*$

Rationale

Unadapted step-size won’t produce successive good individuals

"The step-size are adjusted by the evolution itself"
Path Length Control (CSA)

The Concept

\[ x_i = m + \sigma y_i \]

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

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

\[
\begin{align*}
m &\leftarrow m + \sigma y_w \quad \text{where } y_w = \sum_{i=1}^{\mu} w_i y_i : \lambda \\
p_\sigma &\leftarrow (1 - c_\sigma) p_\sigma + \sqrt{1 - (1 - c_\sigma)^2} \sqrt{\mu_w} y_w \\
\sigma &\leftarrow \sigma \times \exp \left( \frac{c_\sigma}{d_\sigma} \left( \frac{||p_\sigma||}{\mathbb{E}||\mathcal{N}(0, I)||} - 1 \right) \right)
\end{align*}
\]

update mean

accounts for $1 - c_\sigma$

accounts for $w_i$

update step-size

$>1 \iff ||p_\sigma||$ 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 $$n = 10$$
1 Problem Statement

2 Evolution Strategies and EDAs

3 Step-Size Control

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

5 Experiments

6 Summary
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 \( x_i, m \in \mathbb{R}^n, \sigma \in \mathbb{R}_+, \) and \( C \in \mathbb{R}^{n \times n} \)

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

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

new distribution,

\[ \mathbf{C} \leftarrow (1 - c_{\text{cov}}) \times \mathbf{C} + c_{\text{cov}} \times \mathbf{y}_w \mathbf{y}_w^T \]

the ruling principle: the adaptation increases the probability of successful steps, \( \mathbf{y}_w \), to appear again
Covariance Matrix Adaptation

Rank-One Update

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

While not terminate

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

\[
\mathbf{m} \leftarrow \mathbf{m} + \sigma \mathbf{y}_w \quad \text{where} \quad \mathbf{y}_w = \sum_{i=1}^{\mu} w_i \mathbf{y}_i : \lambda
\]

\[
\mathbf{C} \leftarrow (1 - c_{\text{cov}})\mathbf{C} + c_{\text{cov}} \mu_w \begin{pmatrix} \mathbf{y}_w \mathbf{y}_w^T \end{pmatrix} \quad \text{where} \quad \mu_w = \frac{1}{\sum_{i=1}^{\mu} w_i^2} \geq 1
\]
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
Problem Statement

Evolution Strategies and EDAs

Step-Size Control

Covariance Matrix Adaptation

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

Experiments

Summary
Cumulation
The Evolution Path

Evolution Path

Conceptually, the evolution path is the 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:

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

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
- ...

...why?
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 neglected. 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 $\mu_w = \frac{1}{\sum w_i^2}$, $c_c \ll 1$. 

...resulting in
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 $n$.  

...rank $\mu$ update
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 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 $c_{\text{cov}} \leq 1$. 
\[ x_i = m + \sigma y_i, \quad y_i \sim \mathcal{N}(0, C) \]

\[ C_\mu = \frac{1}{\mu} \sum y_i: \lambda y_i^T: \lambda \]

\[ C \leftarrow (1 - \frac{1}{\mu}) \times C + 1 \times C_\mu \]

\[ m_{\text{new}} \leftarrow m + \frac{1}{\mu} \sum y_i: \lambda \]

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 \( c_{\text{cov}} = 1 \)
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 \(\mathcal{O}(n)\) \(^3\)

Therefore the rank-\(\mu\) update is the primary mechanism whenever a large population size is used given \(\mu_w \propto \lambda \propto n\)

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...

---

1 Problem Statement

2 Evolution Strategies and EDAs

3 Step-Size Control

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

5 Experiments

6 Summary
Estimation of Distribution Algorithms

- Estimate a distribution that (re-)samples the parental population.
- All parameters of the distribution $\theta$ are estimated from the given population.

Example: EMNA (Estimation of Multi-variate Normal Algorithm)

Initialize $m \in \mathbb{R}^n$, and $C = I$

While not terminate

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

\[
    m \leftarrow \frac{1}{\mu} \sum_{i=1}^{\mu} x_{i: \lambda}
\]

\[
    C \leftarrow \sum_{i=1}^{\mu} (x_{i: \lambda} - m)(x_{i: \lambda} - m)^T
\]

Estimation of Multivariate Normal Algorithm $\text{EMNA}_{\text{global}}$ versus rank-$\mu$ CMA$^4$

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

1. Problem Statement
2. Evolution Strategies and EDAs
3. Step-Size Control
4. Covariance Matrix Adaptation
5. Experiments
6. Summary
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)

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

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

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

$\ell$ condition number

$\Delta_{i}^{(\text{NEWUOA})}$

$\Delta_{i}^{(\text{BFGS})}$

$\Delta_{i}^{(\text{DE5})}$

$\Delta_{i}^{(\text{DE2})}$

$\Delta_{i}^{(\text{PSO})}$

$\Delta_{i}^{(\text{CMA-ES})}$

$\Delta_{i}^{(\text{BFGS})}$

$\ell$ identity ($\text{BFGS}$, red) or

$\ell$ any order-preserving = strictly increasing (all other)

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

$\ell (x) = g(x^{T}Hx)$ with

$g(.) = (.)^{1/4}$ ($\text{BFGS}$, red dashed)

or

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

CEC 2005
Comparison of 11 Evolutionary Algorithms

- Task: black-box optimization of 25 benchmark functions and submission of results to the Congress of Evolutionary Computation

- Performance measure: cost (number of function evaluations) to reach the target function value, where the maximum number of function evaluations was $F\text{E}_{\text{max}} = \begin{cases} 10^5 & \text{for } n = 10 \\ 3 \times 10^5 & \text{for } n = 30 \end{cases}$

  Remark: the setting of $F\text{E}_{\text{max}}$ has a remarkable influence on the results, if the target function value can be reached only for a (slightly) larger number of function evaluations with a high probability. Where $F\text{E}_{\text{s}} \geq F\text{E}_{\text{max}}$ the result must be taken with great care.

- The competitors included Differential Evolution (DE), Particle Swarm Optimization (PSO), real-coded GAs, Estimation of Distribution Algorithm (EDA), and hybrid methods combined e.g. with quasi-Newton BFGS.
# References to Algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLX-GL50</td>
<td>García-Martínez and Lozano (Hybrid Real-Coded...)</td>
</tr>
<tr>
<td>BLX-MA</td>
<td>Molina et al. (Adaptive Local Search...)</td>
</tr>
<tr>
<td>CoEVO</td>
<td>Pošík (Real-Parameter Optimization...)</td>
</tr>
<tr>
<td>DE</td>
<td>Rönkkönen et al. (Real-Parameter Optimization...)</td>
</tr>
<tr>
<td>DMS-L-PSO</td>
<td>Liang and Suganthan (Dynamic Multi-Swarm...)</td>
</tr>
<tr>
<td>EDA</td>
<td>Yuan and Gallagher (Experimental Results...)</td>
</tr>
<tr>
<td>G-CMA-ES</td>
<td>Auger and Hansen (A Restart CMA...)</td>
</tr>
<tr>
<td>K-PCX</td>
<td>Sinha et al. (A Population-Based...)</td>
</tr>
<tr>
<td>L-CMA-ES</td>
<td>Auger and Hansen (Performance Evaluation...)</td>
</tr>
<tr>
<td>L-SaDE</td>
<td>Qin and Suganthan (Self-Adaptive Differential...)</td>
</tr>
<tr>
<td>SPC-PNX</td>
<td>Ballester et al. (Real-Parameter Optimization...)</td>
</tr>
</tbody>
</table>

In: CEC 2005 IEEE Congress on Evolutionary Computation, Proceedings
Summarized Results

Empirical Distribution of Normalized Success Performance

$F E_s = \text{mean}(\#fevals) \times \frac{\# \text{all runs (25)}}{\# \text{successful runs}}$, where $\#fevals$ includes only successful runs.

Shown: empirical distribution function of the Success Performance $F E_s$ divided by $F E_s$ of the best algorithm on the respective function.

Results of all functions are used where at least one algorithm was successful at least once, i.e. where the target function value was reached in at least one experiment (out of $11 \times 25$ experiments).

Small values for $F E_s$ and therefore large (cumulative frequency) values in the graphs are preferable.
<table>
<thead>
<tr>
<th></th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Problem Statement</td>
</tr>
<tr>
<td>2</td>
<td>Evolution Strategies and EDAs</td>
</tr>
<tr>
<td>3</td>
<td>Step-Size Control</td>
</tr>
<tr>
<td>4</td>
<td>Covariance Matrix Adaptation</td>
</tr>
<tr>
<td>5</td>
<td>Experiments</td>
</tr>
<tr>
<td>6</td>
<td>Summary</td>
</tr>
</tbody>
</table>
Main Features of Evolution Strategies

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

2. Step-size control: converge log-linearly on the sphere

3. Covariance matrix adaptation: reduce any convex quadratic function
   \[ f(x) = x^T H x \]
   to the sphere model
   \[ f(x) = x^T x \]
   without use of derivatives

   lines of equal density align with lines of equal fitness $C \propto H^{-1}$