

Tutorial—The Covariance Matrix Adaptation Evolution Strategy (CMA-ES)

Nikolaus Hansen

April 8, 2008

Content

- 1 Problem Statement
 - Black Box Optimization and Its Difficulties
 - Non-Separable Problems
 - Ill-Conditioned Problems
- 2 Stochastic Search
 - A Search Template
 - The Normal Distribution
- 3 The CMA Evolution Strategy
 - Sampling New Search Points
 - Covariance Matrix Rank-One Update
 - Cumulation—the Evolution Path
 - Covariance Matrix Rank- μ Update
 - Step-Size Control
 - Summary
- 4 Discussion
 - Evaluation/Selection of Search Algorithms
 - Experimentum Crucis
 - Some Comparisons
 - Strategy Internal Parameters
 - Population Size
 - Invariance
- 5 Empirical Validation
 - A Comparison Study

*Einstein once spoke of the “unreasonable effectiveness of mathematics” in describing how the natural world works. Whether one is talking about basic physics, about the increasingly important environmental sciences, or the transmission of disease, **mathematics is never any more, or any less, than a way of thinking clearly.** As such, it always has been and always will be a valuable tool, but only valuable when it is part of a larger arsenal embracing analytic experiments and, above all, wide-ranging imagination.*

Lord Kay

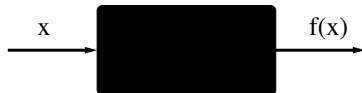
Problem Statement

Continuous Domain Search/Optimization

- Task: **minimize** a **objective function** (*fitness function, loss function*) in continuous domain

$$f : \mathcal{X} \subseteq \mathbb{R}^n \rightarrow \mathbb{R}, \quad \mathbf{x} \mapsto f(\mathbf{x})$$

- **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

Evolutionary Computation

Optimization

individual, offspring, parent	↔	candidate solution decision variables design variables object variables
population	↔	set of candidate solutions
fitness function	↔	objective function loss function cost function
generation	↔	iteration

...function properties

Objective Function Properties

We assume $f : \mathcal{X} \subset \mathbb{R}^n \rightarrow \mathbb{R}$ to have at least moderate dimensionality, say $n \not\ll 10$, 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

- ...

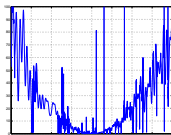
Goal : cope with any of these function properties

they are related to real-world problems

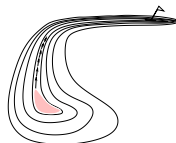
What Makes a Function Difficult to Solve?

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



cut from 3-D example, solvable
with CMA-ES



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.

Consider placing 100 points onto a real interval, say $[-1, 1]$. To get **similar coverage**, in terms of distance between adjacent points, of the 10-dimensional space $[-1, 1]^{10}$ 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

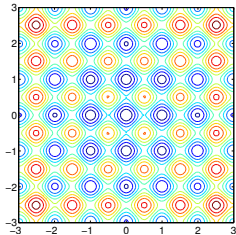
$$\arg \min_{(x_1, \dots, x_n)} f(x_1, \dots, x_n) = \left(\arg \min_{x_1} f(x_1, \dots), \dots, \arg \min_{x_n} f(\dots, x_n) \right)$$

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

Example: Additively decomposable functions

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

Rastrigin function



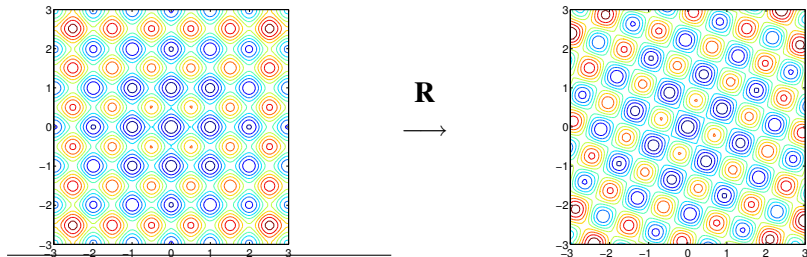
Non-Separable Problems

Building a non-separable problem from a separable one

Rotating the coordinate system

- $f : \mathbf{x} \mapsto f(\mathbf{x})$ separable
- $f : \mathbf{x} \mapsto f(\mathbf{R}\mathbf{x})$ **non-separable**

\mathbf{R} rotation matrix



12

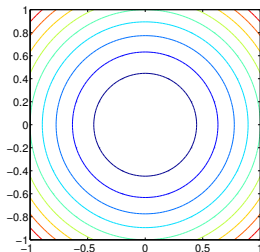
¹ Hansen, Ostermeier, Gawelczyk (1995). On the adaptation of arbitrary normal mutation distributions in evolution strategies: The generating set adaptation. Sixth ICGA, pp. 57-64, Morgan Kaufmann

² Salomon (1996). "Reevaluating Genetic Algorithm Performance under Coordinate Rotation of Benchmark Functions; A survey of some theoretical and practical aspects of genetic algorithms." BioSystems, 39(3):263-278

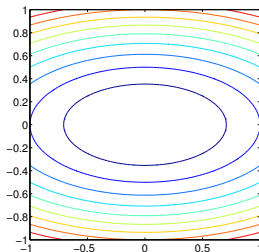
III-Conditioned Problems

If f is quadratic, $f : x \mapsto x^T H x$, ill-conditioned means a high condition number of Hessian Matrix H

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



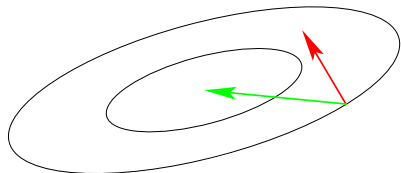
Increased
 →
 condition
 number



consider the curvature of iso-fitness lines

The Benefit of Second Order Information

Consider the convex quadratic function $f(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \mathbf{x}^*)^T \mathbf{H}(\mathbf{x} - \mathbf{x}^*)$



gradient direction $-f'(\mathbf{x})^T$

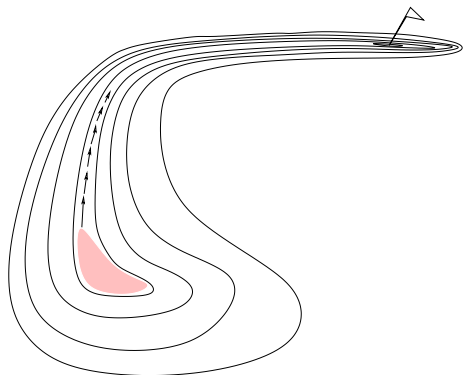
Newton direction $-\mathbf{H}^{-1}f'(\mathbf{x})^T$

Condition number equals nine here. Condition numbers between 100 and even 10^6 can be observed in real world problems.

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

III-Conditioned Problems

Example: A Narrow Ridge



Volume oriented search ends up in the pink area.
To approach the optimum an ill-conditioned problem needs to be solved (e.g. by following the narrow bent ridge).³

³Whitley, Lunacek, Knight 2004. Ruffled by Ridges: How Evolutionary Algorithms Can Fail, *GECCO*

Second Order Approaches

Examples

- quasi-Newton method
- conjugate gradients
- surrogate models, derivative free optimization (DFO)
- linkage learning
- correlated mutations (self-adaptation)
- estimation of distribution algorithms
- covariance matrix adaptation

The mutual idea

capture **dependencies** between variables, a second-order model

...summary

What Makes a Function Difficult to Solve?

... and what can be done

The Problem	What can be done
Ruggedness	<p>non-local policy, large sampling width (step-size) as large as possible while preserving a reasonable convergence speed</p> <p>stochastic, non-elitistic, population-based method recombination operator serves as repair mechanism</p>
Dimensionality, Non-Separability	<p>exploiting the problem structure locality, neighborhood, encoding</p>
Ill-conditioning	<p>second order approach changes the neighborhood metric</p>

- 1 Problem Statement
 - Black Box Optimization and Its Difficulties
 - Non-Separable Problems
 - Ill-Conditioned Problems
- 2 **Stochastic Search**
 - **A Search Template**
 - **The Normal Distribution**
- 3 The CMA Evolution Strategy
 - Sampling New Search Points
 - Covariance Matrix Rank-One Update
 - Cumulation—the Evolution Path
 - Covariance Matrix Rank- μ Update
 - Step-Size Control
 - Summary
- 4 Discussion
 - Evaluation/Selection of Search Algorithms
 - Experimentum Crucis
 - Some Comparisons
 - Strategy Internal Parameters
 - Population Size
 - Invariance
- 5 Empirical Validation
 - A Comparison Study
 - Problem Representation

Stochastic Search

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

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

While not terminate

- 1 **Sample distribution** $P(x|\theta) \rightarrow x_1, \dots, x_\lambda \in \mathbb{R}^n$
- 2 **Evaluate** x_1, \dots, x_λ on f
- 3 **Update parameters** $\theta \leftarrow F_\theta(\theta, x_1, \dots, x_\lambda, f(x_1), \dots, f(x_\lambda))$

Everything depends on the definition of P and F_θ

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*

Stochastic Search

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

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

While not terminate

- ① **Sample distribution** $P(x|\theta) \rightarrow x_1, \dots, x_\lambda \in \mathbb{R}^n$
- ② **Evaluate** x_1, \dots, x_λ on f
- ③ **Update parameters** $\theta \leftarrow F_\theta(\theta, x_1, \dots, x_\lambda, f(x_1), \dots, f(x_\lambda))$

In the following

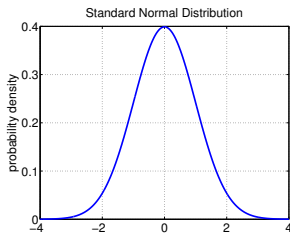
- P is a **multi-variate normal** distribution
 $\mathcal{N}(\mathbf{m}, \sigma^2 \mathbf{C}) \sim \mathbf{m} + \sigma \mathcal{N}(\mathbf{0}, \mathbf{C})$
- $\theta = \{\mathbf{m}, \mathbf{C}, \sigma\} \in \mathbb{R}^n \times \mathbb{R}^{n \times n} \times \mathbb{R}_+$
- $F_\theta = F_\theta(\theta, \mathbf{x}_{1:\lambda}, \dots, \mathbf{x}_{\mu:\lambda})$, where $\mu \leq \lambda$ and $\mathbf{x}_{i:\lambda}$ is the i -th best of the λ points

...why?

Why Normal Distributions?

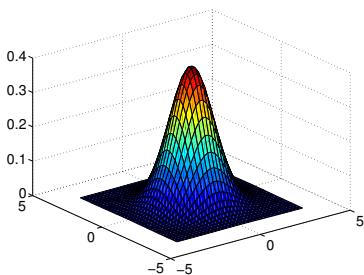
- ① widely observed in nature, for example as phenotypic traits
- ② only stable distribution with finite variance
stable means the sum of normal variates is again normal,
helpful in **design and analysis** of algorithms
- ③ most convenient way to generate **isotropic** search points
the isotropic distribution 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

Normal Distribution



probability density of 1-D standard normal distribution

2-D Normal Distribution



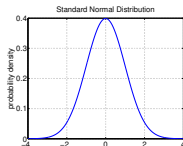
probability density of 2-D normal distribution

The Multi-Variate (n -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} .

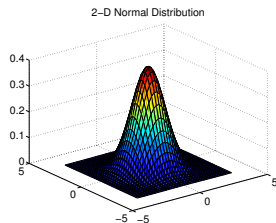
The **mean** value \mathbf{m}

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

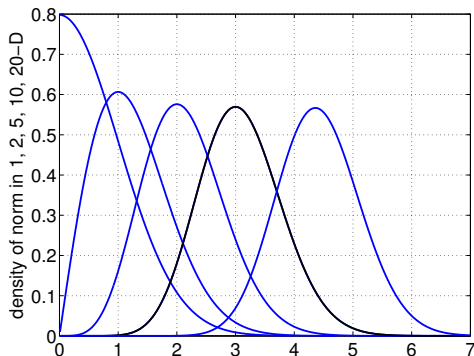


Normal Distribution Revisited

While the maximum likelihood of the multi-variate normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$ is at zero, the distribution of its norm $\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|$ reveals a different, surprising picture.



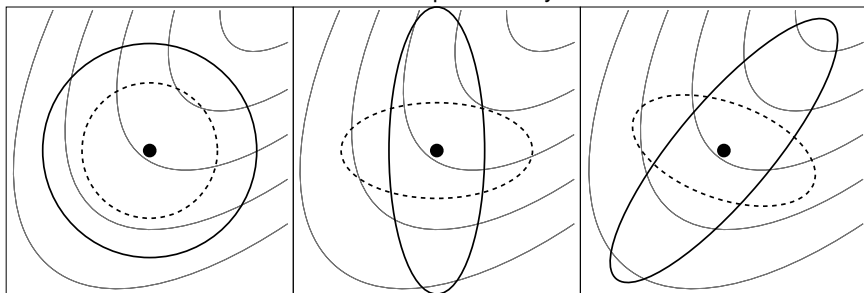
Multi-Variate Normal Distribution



- In 10-D (black) the usual step length is about $3 \times \sigma$ and step lengths smaller than $1 \times \sigma$ virtually never occur
- Remind: this norm-density shape maximizes the distribution entropy

The **covariance matrix \mathbf{C}** determines the shape. It has a valuable **geometrical interpretation**: any covariance matrix can be uniquely identified with the iso-density ellipsoid $\{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{x}^T \mathbf{C}^{-1} \mathbf{x} = 1\}$

Lines of Equal Density



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

one degree of freedom σ
components of $\mathcal{N}(\mathbf{0}, \mathbf{I})$
are independent standard
normally distributed

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

n degrees of freedom
components are
independent, scaled

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

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

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

- 1 Problem Statement
- 2 Stochastic Search
- 3 The CMA Evolution Strategy**
 - Sampling New Search Points
 - Covariance Matrix Rank-One Update
 - Cumulation—the Evolution Path
 - Covariance Matrix Rank- μ Update
 - Step-Size Control
 - Summary
- 4 Discussion
- 5 Empirical Validation

Sampling New Search Points

The Mutation Operator

New search points are sampled normally distributed

$$\mathbf{x}_i \sim \mathbf{m} + \sigma \mathcal{N}_i(\mathbf{0}, \mathbf{C}) \quad \text{for } i = 1, \dots, \lambda$$

as perturbations of \mathbf{m}

where $\mathbf{x}_i, \mathbf{m} \in \mathbb{R}^n$, $\sigma \in \mathbb{R}_+$, and $\mathbf{C} \in \mathbb{R}^{n \times n}$

where

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

The question remains how to update \mathbf{m} , \mathbf{C} , and σ .

Update of the Distribution Mean \mathbf{m}

Selection and Recombination

Given the i -th solution point $\mathbf{x}_i = \mathbf{m} + \sigma \underbrace{\mathcal{N}_i(\mathbf{0}, \mathbf{C})}_{=: \mathbf{y}_i} = \mathbf{m} + \sigma \mathbf{y}_i$

Let $\mathbf{x}_{i:\lambda}$ the i -th **ranked** solution point, such that $f(\mathbf{x}_{1:\lambda}) \leq \dots \leq f(\mathbf{x}_{\lambda:\lambda})$.

The new mean reads

$$\mathbf{m} \leftarrow \sum_{i=1}^{\mu} w_i \mathbf{x}_{i:\lambda} = \mathbf{m} + \sigma \underbrace{\sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda}}_{=: \mathbf{y}_w}$$

where

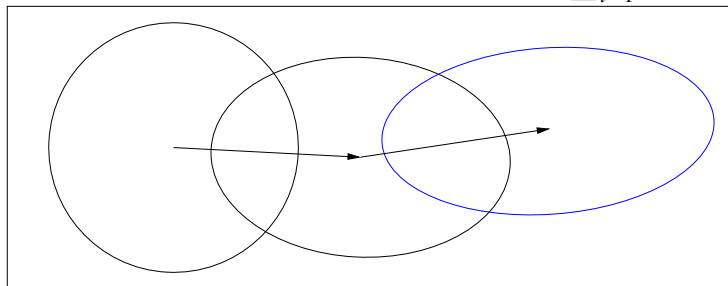
$$w_1 \geq \dots \geq w_{\mu} > 0, \quad \sum_{i=1}^{\mu} w_i = 1$$

The best μ points are selected from the new solutions (non-elitistic) and **weighted intermediate recombination** is applied.

Covariance Matrix Adaptation

Rank-One Update

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



new distribution,

$$\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \mathbf{y}_w \mathbf{y}_w^T$$

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

... equations

Preliminary Set of Equations

Covariance Matrix Adaptation with 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

$$\mathbf{x}_i = \mathbf{m} + \sigma \mathbf{y}_i, \quad \mathbf{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C}),$$

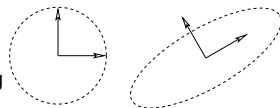
$$\mathbf{m} \leftarrow \mathbf{m} + \sigma \mathbf{y}_w \quad \text{where } \mathbf{y}_w = \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda}$$

$$\mathbf{C} \leftarrow (1 - c_{\text{cov}})\mathbf{C} + c_{\text{cov}} \underbrace{\mu_w \mathbf{y}_w \mathbf{y}_w^T}_{\text{rank-one}} \quad \text{where } \mu_w = \frac{1}{\sum_{i=1}^{\mu} w_i^2} \geq 1$$

$$\mathbf{C} \leftarrow (1 - c_{\text{cov}})\mathbf{C} + c_{\text{cov}}\mu_w \mathbf{y}_w \mathbf{y}_w^T$$

The covariance matrix adaptation

- learns all pairwise dependencies between variables
off-diagonal entries in the covariance matrix reflect the dependencies
- learns a new (rotated) problem representation (according to the principle axes of the mutation ellipsoid)
components are independent (only) in the new representation
- learns a new metric according to the scaling of the independent components
in the new representation
- conducts a principle component analysis (PCA) of steps \mathbf{y}_w , sequentially in time and space
eigenvectors of the covariance matrix \mathbf{C} are the principle components / the principle axes of the mutation ellipsoid
- approximates the inverse Hessian on quadratic functions
overwhelming empirical evidence, proof is in progress
- is equivalent with an adaptive (general) linear encoding⁴



... cumulation, rank- μ , step-size control

⁴Hansen 2000, Invariance, Self-Adaptation and Correlated Mutations in Evolution Strategies, PPSN VI

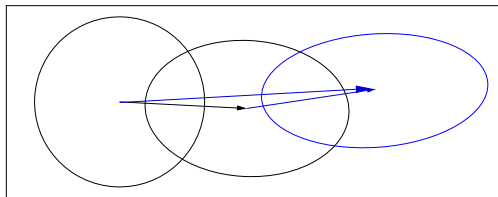
- 1 Problem Statement
- 2 Stochastic Search
- 3 The CMA Evolution Strategy**
 - Sampling New Search Points
 - Covariance Matrix Rank-One Update
 - Cumulation—the Evolution Path
 - Covariance Matrix Rank- μ Update
 - Step-Size Control
 - Summary
- 4 Discussion
- 5 Empirical Validation

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

$$p_c \propto \sum_{i=0}^g \underbrace{(1 - c_c)^{g-i}}_{\text{exponentially fading weights}} y_w^{(i)}$$

The recursive construction of the evolution path (cumulation):

$$p_c \leftarrow \underbrace{(1 - c_c)}_{\text{decay factor}} p_c + \underbrace{\sqrt{1 - (1 - c_c)^2} \sqrt{\mu_w}}_{\text{normalization factor}} \underbrace{y_w}_{\text{input, } \frac{m - m_{\text{old}}}{\sigma}}$$

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 know as

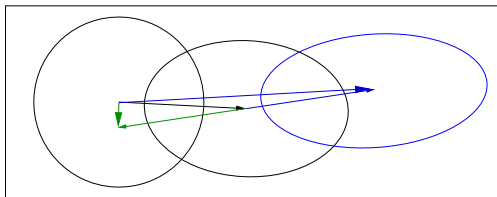
- *exponential smoothing* in time series, forecasting
- exponentially weighted *moving average*
- *iterate averaging* in stochastic approximation
- *momentum* in the back-propagation algorithm for ANNs
- ...

...why?

Cumulation

Utilizing the Evolution Path

We used $\mathbf{y}_w \mathbf{y}_w^T$ for updating \mathbf{C} . Because $\mathbf{y}_w \mathbf{y}_w^T = -\mathbf{y}_w (-\mathbf{y}_w)^T$ the sign of \mathbf{y}_w is neglected. The sign information is (re-)introduced by using the *evolution path*.



$$\mathbf{p}_c \leftarrow \underbrace{(1 - c_c)}_{\text{decay factor}} \mathbf{p}_c + \underbrace{\sqrt{1 - (1 - c_c)^2}}_{\text{normalization factor}} \sqrt{\mu_w} \mathbf{y}_w$$

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

... equations

Preliminary Set of Equations (2)

Covariance Matrix Adaptation, Rank-One Update with Cumulation

Initialize $\mathbf{m} \in \mathbb{R}^n$, $\mathbf{C} = \mathbf{I}$, and $\mathbf{p}_c = \mathbf{0} \in \mathbb{R}^n$,

set $\sigma = 1$, $c_c \approx 4/n$, $c_{cov} \approx 2/n^2$

While not terminate

$$\mathbf{x}_i = \mathbf{m} + \sigma \mathbf{y}_i, \quad \mathbf{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C}),$$

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

$$\mathbf{p}_c \leftarrow (1 - c_c) \mathbf{p}_c + \sqrt{1 - (1 - c_c)^2} \sqrt{\mu_w} \mathbf{y}_w$$

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

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

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

^aHansen, Müller and Koumoutsakos 2003. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). *Evolutionary Computation*, 11(1), pp. 1-18

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

... rank μ update

Rank- μ Update

$$\begin{aligned} \mathbf{x}_i &= \mathbf{m} + \sigma \mathbf{y}_i, & \mathbf{y}_i &\sim \mathcal{N}_i(\mathbf{0}, \mathbf{C}), \\ \mathbf{m} &\leftarrow \mathbf{m} + \sigma \mathbf{y}_w, & \mathbf{y}_w &= \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda} \end{aligned}$$

The rank- μ update extends the update rule for **large population sizes** λ using $\mu > 1$ vectors to update \mathbf{C} at each generation step.

The matrix

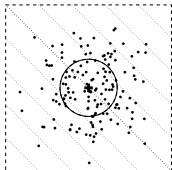
$$\mathbf{C}_{\mu} = \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda} \mathbf{y}_{i:\lambda}^T$$

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

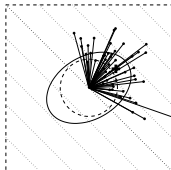
The rank- μ update then reads

$$\mathbf{C} \leftarrow (1 - c_{\text{cov}}) \mathbf{C} + c_{\text{cov}} \mathbf{C}_{\mu}$$

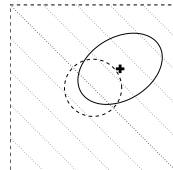
where $c_{\text{cov}} \approx \mu_w/n^2$ and $c_{\text{cov}} \leq 1$.



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



$$\begin{aligned} \mathbf{C}_\mu &= \frac{1}{\mu} \sum z_i: \lambda y_{i: \lambda}^T \\ \mathbf{C} &\leftarrow (1 - \lambda) \times \mathbf{C} + \lambda \times \mathbf{C}_\mu \end{aligned}$$

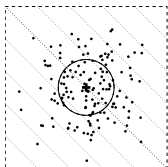


$$m_{\text{new}} \leftarrow m + \frac{1}{\mu} \sum y_{i: \lambda}$$

new distribution

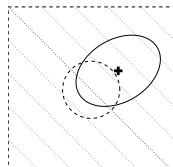
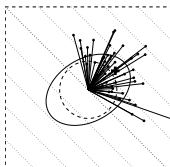
sampling of $\lambda = 150$
solutions where
 $\mathbf{C} = \mathbf{I}$ and $\sigma = 1$

calculating \mathbf{C} where
 $\mu = 50$,
 $w_1 = \dots = w_\mu = \frac{1}{\mu}$,
and $c_{\text{cov}} = 1$

rank- μ CMA versus Estimation of Multivariate Normal Algorithm EMNA_{global}⁵

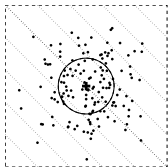
$$x_i = m_{\text{old}} + y_i, \quad y_i \sim \mathcal{N}(0, \mathbf{C})$$

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



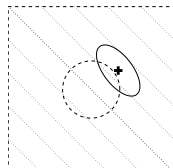
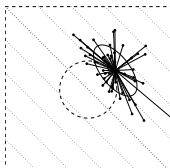
$$m_{\text{new}} = m_{\text{old}} + \frac{1}{\mu} \sum y_{i:\lambda}$$

rank- μ CMA
conducts a
PCA of
steps



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

$$\mathbf{C} \leftarrow \frac{1}{\mu} \sum (x_{i:\lambda} - m_{\text{new}})(x_{i:\lambda} - m_{\text{new}})^T$$



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

EMNA_{global}
conducts a
PCA of
points

sampling of $\lambda = 150$
solutions (dots)

calculating \mathbf{C} from $\mu = 50$
solutions

new distribution

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

⁵ Hansen, N. (2006). The CMA Evolution Strategy: A Comparing Review. In J.A. Lozano, P. Larranga, I. Inza and E. Bengoetxea (Eds.). Towards a new evolutionary computation. Advances in estimation of distribution algorithms. pp. 75-102

The rank- μ update

- increases the possible learning rate in large populations
roughly from $2/n^2$ to μ_w/n^2
- can reduce the number of necessary **generations** roughly from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ ⁶
given $\mu_w \propto \lambda \propto n$

Therefore the rank- μ 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- μ update can be combined...

⁶Hansen, Müller, and Koumoutsakos 2003. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). *Evolutionary Computation*, 11(1), pp. 1-18

Preliminary Set of Equations (3)

Rank-One Update with Cumulation, Rank- μ Update

Initialize $\mathbf{m} \in \mathbb{R}^n$, $\mathbf{C} = \mathbf{I}$, and $\mathbf{p}_c = \mathbf{0}$,

set $\sigma = 1$, $c_c \approx 4/n$, $c_1 \approx 2/N^2$, $c_\mu \approx \mu_w/N^2$, $c_1 + c_\mu \leq 1$

While not terminate

$$\mathbf{x}_i = \mathbf{m} + \sigma \mathbf{y}_i, \quad \mathbf{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C}), \quad \text{sampling}$$

$$\mathbf{m} \leftarrow \mathbf{m} + \sigma \mathbf{y}_w \quad \text{where } \mathbf{y}_w = \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda} \quad \text{update mean}$$


$$\mathbf{p}_c \leftarrow (1 - c_c) \mathbf{p}_c + \sqrt{1 - (1 - c_c)^2} \sqrt{\mu_w} \mathbf{y}_w \quad \text{cumulation for } \mathbf{C}$$

$$\mathbf{C} \leftarrow (1 - c_1 - c_\mu) \mathbf{C} + c_1 \mathbf{p}_c \mathbf{p}_c^T + c_\mu \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda} \mathbf{y}_{i:\lambda}^T \quad \text{update } \mathbf{C}$$

Missing: update of σ

- 1 Problem Statement
- 2 Stochastic Search
- 3 The CMA Evolution Strategy**
 - Sampling New Search Points
 - Covariance Matrix Rank-One Update
 - Cumulation—the Evolution Path
 - Covariance Matrix Rank- μ Update
 - Step-Size Control
 - Summary
- 4 Discussion
- 5 Empirical Validation

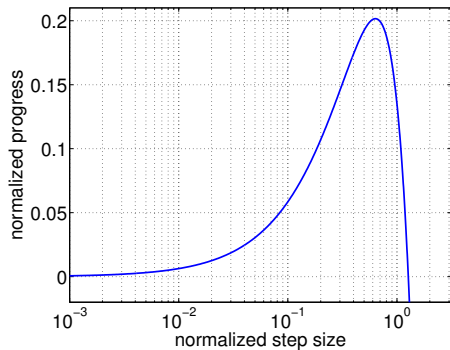
Why Step-Size Control?

- ① the covariance matrix update can hardly **increase the variance** in *all* directions simultaneously
- ② There is a relatively small *evolution window* for the step-size. Given $\mu \not\gg n$ the **optimal step length** remarkably depends on parent number μ . The C-update cannot achieve close to optimal step lengths for a wide range of μ .
 
- ③ The learning rate $c_{\text{cov}} \approx \mu_w/n^2$ does not comply with the requirements of **convergence speed on the sphere model**, $f(\mathbf{x}) = \sum x_i^2$.

Each single reason would be sufficient to ask for additional step-size control

... methods for step-size control

Why Step-Size Control?



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

- **σ -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

- **two-point adaptation**, used in Evolutionary Gradient Search^d

simplified “global” self-adaptation

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

^aRechenberg 1973, *Evolutionsstrategie, Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, Frommann-Holzboog

^bSchumer and Steiglitz 1968. Adaptive step size random search. *IEEE TAC*

^cSchwefel 1981, *Numerical Optimization of Computer Models*, Wiley

^dRechenberg 1993, *Evolutionary Algorithms in the Design of Control Systems*, Wiley

Path Length Control

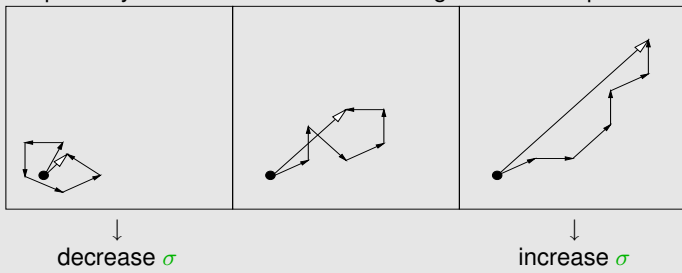
The Concept

$$x_i = m + \sigma y_i$$

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

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

The Equations

Initialize $\mathbf{m} \in \mathbb{R}^n$, $\sigma \in \mathbb{R}_+$, evolution path $\mathbf{p}_\sigma = \mathbf{0}$,
set $\mathbf{C} = \mathbf{I}$, $c_\sigma \approx 4/n$, $d_\sigma \approx 1$.

$$\mathbf{x}_i = \mathbf{m} + \sigma \mathbf{y}_i, \quad \mathbf{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C}), \quad \text{sampling}$$

$$\mathbf{m} \leftarrow \mathbf{m} + \sigma \mathbf{y}_w \quad \text{where } \mathbf{y}_w = \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda} \quad \text{update mean}$$

$$\mathbf{p}_\sigma \leftarrow (1 - c_\sigma) \mathbf{p}_\sigma + \underbrace{\sqrt{1 - (1 - c_\sigma)^2}}_{\text{accounts for } 1 - c_\sigma} \underbrace{\sqrt{\mu_w}}_{\text{accounts for } w_i} \mathbf{C}^{-\frac{1}{2}} \mathbf{y}_w$$

$$\sigma \leftarrow \sigma \times \underbrace{\exp\left(\frac{c_\sigma}{d_\sigma} \left(\frac{\|\mathbf{p}_\sigma\|}{\mathbb{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|} - 1\right)\right)}_{>1 \iff \|\mathbf{p}_\sigma\| \text{ is greater than its expectation}} \quad \text{update step-size}$$

where $\sqrt{\mathbf{C}^{-1}} = \mathbf{C}^{-\frac{1}{2}} = \sqrt{\mathbf{C}^{-1}}$ is symmetric positive definite

Summary

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) in a Nutshell

- 1 Multivariate normal distribution to generate new search points
follows the maximum entropy principle
- 2 Selection only based on the ranking of the f -values, weighted recombination
using only the ranking of f -values preserves invariance
- 3 *Covariance matrix adaptation (CMA)* **increases the probability** to repeat **successful steps**
learning all pairwise dependencies
⇒ conducts an incremental PCA
⇒ new (rotated) problem representation
- 4 An **evolution path** (a trajectory) is exploited in two places
 - enhances the covariance matrix (rank-one) adaptation
yields sometimes linear time complexity
 - controls the **step-size** (step length)
aims at conjugate perpendicularity

Summary of Equations

The Covariance Matrix Adaptation Evolution Strategy

Initialize $\mathbf{m} \in \mathbb{R}^n$, $\sigma \in \mathbb{R}_+$, $\mathbf{C} = \mathbf{I}$, and $\mathbf{p}_c = \mathbf{0}$, $\mathbf{p}_\sigma = \mathbf{0}$,
 set $c_c \approx 4/n$, $c_\sigma \approx 4/n$, $c_1 \approx 2/n^2$, $c_\mu \approx \mu_w/n^2$, $c_1 + c_\mu \leq 1$, $d_\sigma \approx 1 + \sqrt{\frac{\mu_w}{n}}$,
 set λ and $w_i, i = 1, \dots, \mu$ such that $\mu_w \approx 0.3 \lambda$

While not terminate

$$\mathbf{x}_i = \mathbf{m} + \sigma \mathbf{y}_i, \quad \mathbf{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C}), \quad \text{sampling}$$

$$\mathbf{m} \leftarrow \mathbf{m} + \sigma \mathbf{y}_w \quad \text{where } \mathbf{y}_w = \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda} \quad \text{update mean}$$

$$\mathbf{p}_c \leftarrow (1 - c_c) \mathbf{p}_c + \mathbf{1}_{\{\|\mathbf{p}_\sigma\| < 1.5\sqrt{n}\}} \sqrt{1 - (1 - c_c)^2} \sqrt{\mu_w} \mathbf{y}_w \quad \text{cumulation for } \mathbf{C}$$

$$\mathbf{p}_\sigma \leftarrow (1 - c_\sigma) \mathbf{p}_\sigma + \sqrt{1 - (1 - c_\sigma)^2} \sqrt{\mu_w} \mathbf{C}^{-\frac{1}{2}} \mathbf{y}_w \quad \text{cumulation for } \sigma$$

$$\mathbf{C} \leftarrow (1 - c_1 - c_\mu) \mathbf{C} + c_1 \mathbf{p}_c \mathbf{p}_c^T + c_\mu \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda} \mathbf{y}_{i:\lambda}^T \quad \text{update } \mathbf{C}$$

$$\sigma \leftarrow \sigma \times \exp\left(\frac{c_\sigma}{d_\sigma} \left(\frac{\|\mathbf{p}_\sigma\|}{\mathbb{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|} - 1\right)\right) \quad \text{update of } \sigma$$

What was not covered?

- boundary handling
partly part of the problem encoding and parameterization
 - non-linear constraints handling
open question
 - handling of *very high* noise levels and time changing environment
cheap solution available^a
-
- integer variables
feasible

^aHansen et al. (200?). A Method for Handling Uncertainty in Evolutionary Optimization with an Application to Feedback Control of Combustion, *IEEE TEC*

1 Problem Statement

2 Stochastic Search

3 The CMA Evolution Strategy

4 Discussion

- Evaluation/Selection of Search Algorithms
- Experimentum Crucis
- Some Comparisons
- Strategy Internal Parameters
- Population Size
- Invariance

5 Empirical Validation

Evaluation/Selection of Search Algorithms

Evaluation (of the performance) of a search algorithm needs

- meaningful **quantitative measure** on benchmark functions or real world problems
- account for **meta-parameter tuning**
can be quite expensive
- acknowledge **invariance properties** (symmetries)
prediction of performance is based on “similarity”, ideally equivalence classes of functions
- account for **algorithm internal cost**
often negligible, depending on the objective function cost

Experimentum Crucis

What did we specifically want to achieve?

- reduce any convex quadratic function

$$f(\mathbf{x}) = \mathbf{x}^T \mathbf{H} \mathbf{x}$$

to the sphere model

$$f(\mathbf{x}) = \mathbf{x}^T \mathbf{x}$$

without use of derivatives

- lines of equal density align with lines of equal fitness

$$\mathbf{C} \propto \mathbf{H}^{-1}$$

in a stochastic sense

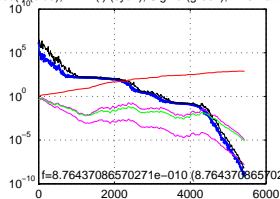
- even true for any $g(f(\mathbf{x})) = g(\mathbf{x}^T \mathbf{H} \mathbf{x})$

$g : \mathbb{R} \rightarrow \mathbb{R}$ strictly monotonic (order preserving)

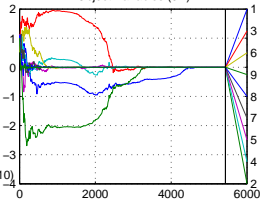
Experimentum Crucis (1)

f convex quadratic, separable

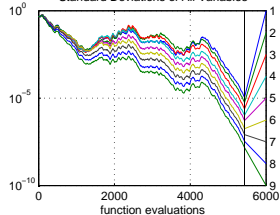
abs(f_0) (blue), $f - \min(f)$ (cyan), Sigma (green), Axis Ratio (red)



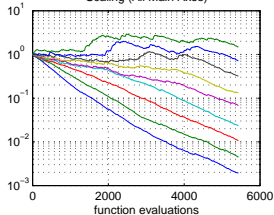
Object Variables (9D)



Standard Deviations of All Variables



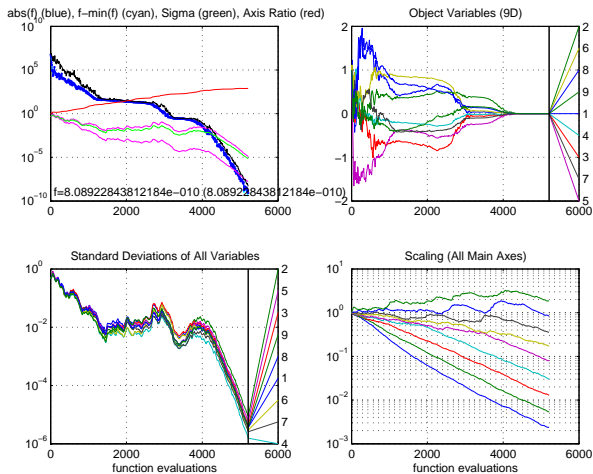
Scaling (All Main Axes)



$$f(\mathbf{x}) = \sum_{i=1}^n 10^{\alpha \frac{i-1}{n-1}} x_i^2, \alpha = 6$$

Experimentum Crucis (2)

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



$C \propto H^{-1}$ for all g, H

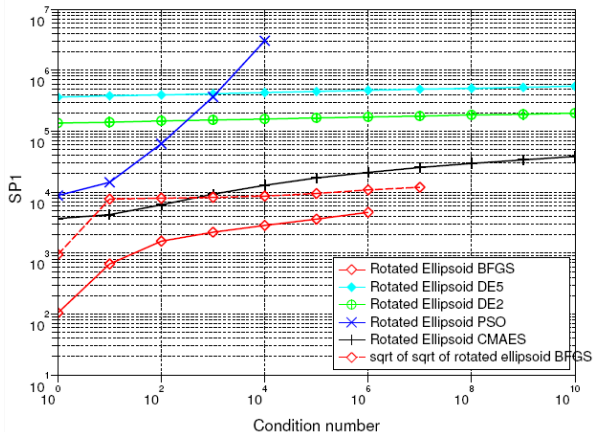
$f(\mathbf{x}) = g(\mathbf{x}^T \mathbf{H} \mathbf{x})$, $g : \mathbb{R} \rightarrow \mathbb{R}$ strictly monotonic

... internal parameters

Comparison to BFGS, PSO and DE

f convex quadratic, non-separable (rotated) with varying α

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



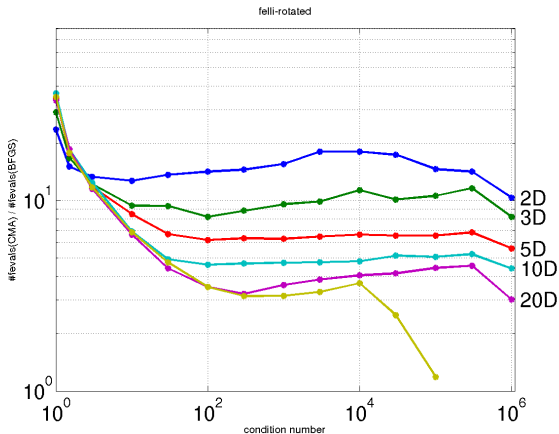
$f(\mathbf{x}) = g(\mathbf{x}^T \mathbf{H} \mathbf{x})$ with
 g identity (BFGS, red) or
 $g(\cdot) = (\cdot)^{1/4}$ (BFGS, red dashed) or
 g any order-preserving = strictly increasing (all other)

SP1 = average number of objective function evaluations to reach the target function value of 10^{-9}

... population size, invariance

Comparison to BFGS

f convex quadratic, non-separable (rotated)

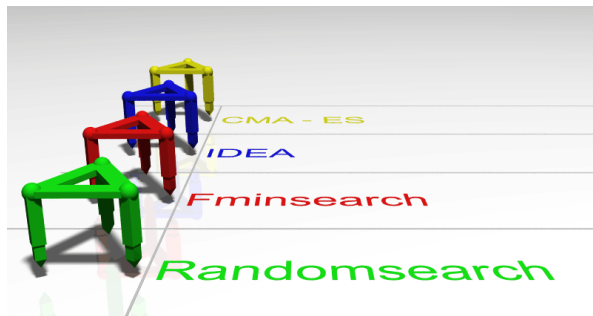


$$f(\mathbf{x}) = \mathbf{x}^T \mathbf{H} \mathbf{x}$$

shown is $\frac{\text{function evaluations CMA-ES}}{\text{function evaluations BFGS}}$ until to reach $f = 10^{-6}$
versus condition number

... population size, invariance

Comparison to IDEA and Simplex-Downhill



IDEA: Iterated Density-Estimation Evolutionary Algorithm⁷

Fminsearch: Nelder-Mead simplex downhill method⁸

just check it out. . .

<http://www.icos.ethz.ch/cse/research/highlights/Race.gif>

⁷ Bosman (2003) Design and Application of Iterated Density-Estimation Evolutionary Algorithms. PhD thesis.

⁸ Nelder and Mead (1965). A simplex method for function minimization. *Computer Journal*.

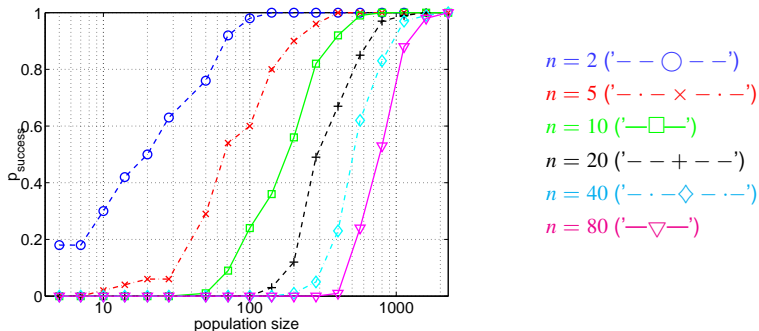
Strategy Internal Parameters

- related to selection and recombination
 - λ , offspring number, new solutions sampled, population size
 - μ , parent number, solutions involved in updates of m , C , and σ
 - $w_{i=1, \dots, \mu}$, recombination weights
 - μ and w_i should be chosen such that the variance effective selection mass $\mu_w \approx \frac{\lambda}{4}$, where $\mu_w := 1 / \sum_{i=1}^{\mu} w_i^2$.
- related to C -update
 - c_{cov} , learning rate for C -update
 - c_c , learning rate for the evolution path
 - μ_{cov} , weight for rank- μ update versus rank-one update
- related to σ -update
 - c_σ , learning rate of the evolution path
 - d_σ , damping for σ -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.** Arguably with the exception of the population size λ that might be reasonably varied in a wide range, *depending on the objective function*

Population Size on Multi-Modal Functions

Success Probability to Find the Global Optimum



Shown: **success rate** to reach the global optimum versus offspring population size on the highly multi-modal Rastrigin function⁹

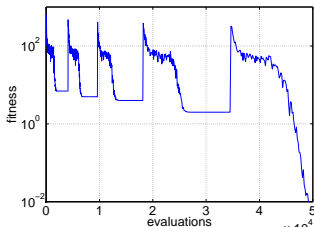
On multi-modal functions increasing the population size can sharply increase the success probability to find the global optimum

⁹ Hansen & Kern 2004. Evaluating the CMA Evolution Strategy on Multimodal Test Functions. PPSN VIII, Springer-Verlag, pp. 282-291.

Multi-Start With Increasing Population Size

Increase by a Factor of Two Each Restart

- ① no performance loss, where small population size is sufficient (e.g. on unimodal functions)
- ② moderate performance loss, if large population size is necessary
loss has, in principle, an upper bound



for a factor between successive runs of ≥ 1.5 we have a performance loss smaller than

$$\sum_{k=0}^{\infty} 1/1.5^k = 3$$

This results in a **quasi parameter free search algorithm**.¹⁰

... empirical evaluation

¹⁰ Auger & Hansen 2005. A Restart CMA Evolution Strategy With Increasing Population Size. IEEE Congress on Evolutionary Computation.

Invariance

Motivation

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

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

base our hypotheses about algorithm performance. They are only useful if they do **generalize** to other problems

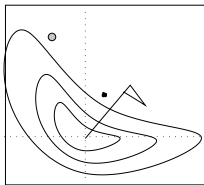
with the smallest number of result we want to cover the greatest number of problems

- **Invariance** is a statement about the feasibility of generalization
 - generalizes performance from a single function to a class of functions*

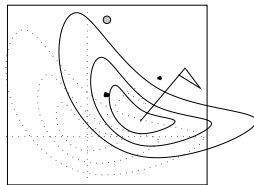
Basic Invariance in Search Space

translation invariance, for example

applies to most optimization algorithms



$$f(x) \leftrightarrow f(x - a)$$



Identical behavior on f and f_a

$$f : \mathbf{x} \mapsto f(\mathbf{x}), \quad \mathbf{x}^{(t=0)} = \mathbf{x}_0$$

$$f_a : \mathbf{x} \mapsto f(\mathbf{x} - \mathbf{a}), \quad \mathbf{x}^{(t=0)} = \mathbf{x}_0 + \mathbf{a}$$

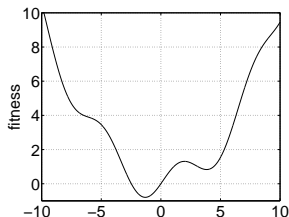
No difference can be observed w.r.t. the argument of f

Only useful if the initial point is not decisive

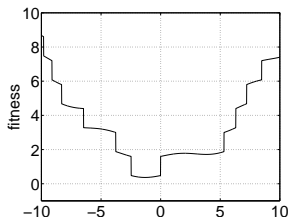
Invariance in Function Space

invariance to order preserving transformations

preserved by ranking based selection



$$f(\mathbf{x}) \leftrightarrow g(f(\mathbf{x}))$$



Identical behavior on f and $g \circ f$ for all order preserving $g : \mathbb{R} \rightarrow \mathbb{R}$ (strictly monotonically increasing g)

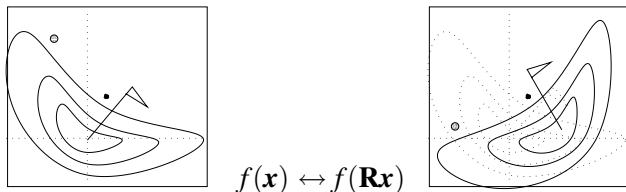
$$\begin{aligned} f &: \mathbf{x} \mapsto f(\mathbf{x}), & \mathbf{x}^{(t=0)} &= \mathbf{x}_0 \\ g \circ f &: \mathbf{x} \mapsto g(f(\mathbf{x})), & \mathbf{x}^{(t=0)} &= \mathbf{x}_0 \end{aligned}$$

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

Rotational Invariance in Search Space

invariance to an orthogonal transformation \mathbf{R} , where $\mathbf{R}\mathbf{R}^T = \mathbf{I}$

e.g. true for simple evolution strategies
recombination operators might jeopardize rotational invariance



Identical behavior on f and $f_{\mathbf{R}}$

$$f : \mathbf{x} \mapsto f(\mathbf{x}), \quad \mathbf{x}^{(t=0)} = \mathbf{x}_0$$

$$f_{\mathbf{R}} : \mathbf{x} \mapsto f(\mathbf{R}\mathbf{x}), \quad \mathbf{x}^{(t=0)} = \mathbf{R}^{-1}(\mathbf{x}_0)$$

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

Invariances in Search Space

- invariance to any rigid (scalar product preserving) transformation in search space $\mathbf{x} \mapsto \mathbf{R}\mathbf{x} - \mathbf{a}$, where $\mathbf{R}\mathbf{R}^T = \mathbf{I}$
e.g. true for simple evolution strategies
- scale invariance (scalar multiplication)
exploited by step-size control
- invariance to a general linear transformation \mathbf{G}
exploited by CMA

Identical behavior on f and $f_{\mathbf{G}}$

$$f : \mathbf{x} \mapsto f(\mathbf{x}), \quad \mathbf{x}^{(t=0)} = \mathbf{x}_0, \quad \mathbf{C}^{(t=0)} = \mathbf{I}$$

$$f_{\mathbf{G}} : \mathbf{x} \mapsto f(\mathbf{G}(\mathbf{x} - \mathbf{b})), \quad \mathbf{x}^{(t=0)} = \mathbf{G}^{-1}\mathbf{x}_0 + \mathbf{b}, \quad \mathbf{C}^{(t=0)} = \mathbf{G}^{-1}\mathbf{G}^{-1T}$$

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

Only useful with an effective adaptation of \mathbf{C}

Invariance of the CMA Evolution Strategy

- The CMA Evolution Strategy **inherits all invariances** from simple evolution strategies

*to rigid transformations of the search space and
to order preserving transformations of the function value*

- The Covariance Matrix Adaptation adds invariance to general linear transformations

*useful **only together** with an effective adaptation of the covariance matrix*

... empirical validation CEC

- 1 Problem Statement
- 2 Stochastic Search
- 3 The CMA Evolution Strategy
- 4 Discussion
- 5 Empirical Validation**
 - A Comparison Study

Comparison of 11 Evolutionary Algorithms

A Performance Meta-Study

- 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 $FE_{\max} = \begin{cases} 10^5 & \text{for } n = 10 \\ 3 \times 10^5 & \text{for } n = 30 \end{cases}$

Remark: the setting of FE_{\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 $FES \geq FE_{\max}$ 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.

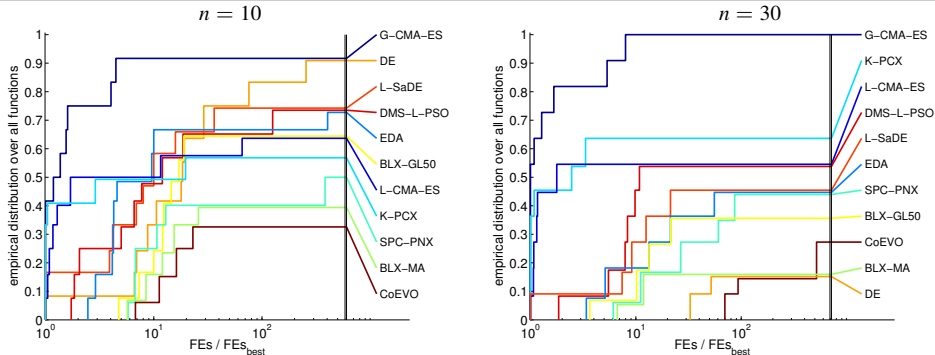
References to Algorithms

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

In: CEC 2005 IEEE Congress on Evolutionary Computation, Proceedings

Summarized Results

Empirical Distribution of Normalized Success Performance



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

Shown: **empirical distribution function** of the Success Performance FE_s divided by FE_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 11×25 experiments).

Small values for FE_s and therefore large (cumulative frequency) values in the graphs are preferable.

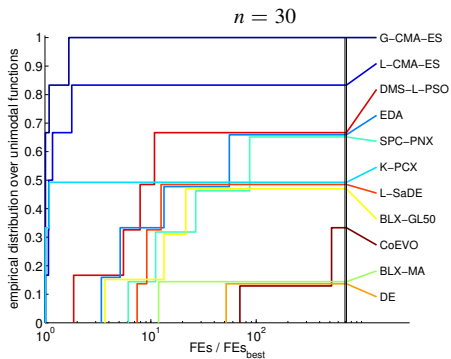
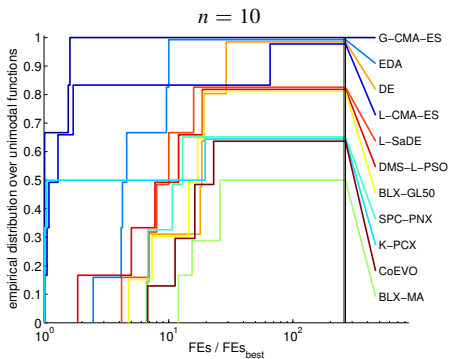
Function Sets

We split the function set into three subsets

- unimodal functions
- solved multimodal functions
at least one algorithm conducted at least one successful run
- unsolved multimodal functions
no single run was successful for any algorithm

Unimodal Functions

Empirical Distribution of Normalized Success Performance



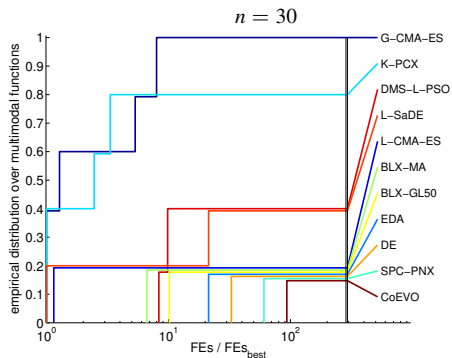
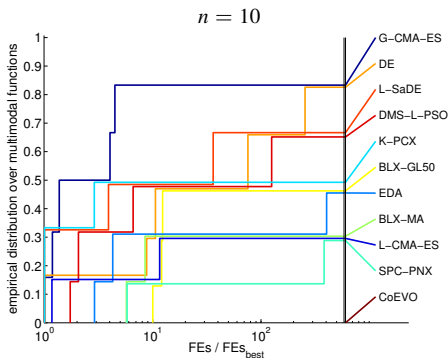
Empirical distribution function of the Success Performance FE_s divided by FE_s of the best algorithm (table entries of last slides).

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

Small values of FE_s and therefore large values in the empirical distribution graphs are preferable.

Multimodal Functions

Empirical Distribution of Normalized Success Performance



Empirical distribution function of the Success Performance FE_s divided by FE_s of the best algorithm (table entries of last slides).

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

Small values of FE_s and therefore large values in the empirical distribution graphs are preferable.

Comparison Study

Conclusion

The CMA-ES with multi-start and increasing population size (G-CMA-ES)

- performs best **over all functions**
- performs best on the **function subsets**
 - unimodal functions
 - solved multimodal functions
 - unsolved multimodal functions
- no **parameter tuning** were conducted
- G-CMA-ES, L-CMA-ES, and EDA have the most **invariance properties**
- on two **separable problems** G-CMA-ES is considerably outperformed

Conclusion

The Take Home Message

Difficulties of a non-linear optimization problem are

- ruggedness
demands a non-local (stochastic?) approach
- dimensionality and non-separability
demands to exploit problem structure, e.g. neighborhood
- ill-conditioning
demands to acquire a second order model

The **CMA-ES** addresses these difficulties and is

- a **robust local search** algorithm
BFGS is roughly ten times faster on convex quadratic f
- a **competitive global search** algorithm
empirically outperforms other plain or hybrid EAs on most functions
- successfully applied to many real-world applications
easily applicable as quasi parameter free

Thank You

<http://www.bionik.tu-berlin.de/user/niko/cmaesintro.html>
or google **NIKOLAUS HANSEN**

Strategy Internal CPU Consumption

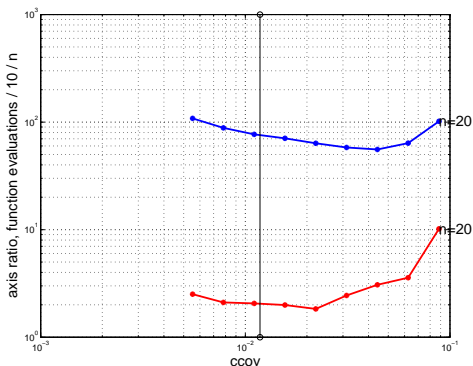
On a 2.5GHz processor our CMA-ES implementation needs

- roughly $3 \times 10^{-8}(n + 4)^2$ seconds per function evaluation
- for one million function evaluations roughly

n	time
10	5s
30	30s
100	300s

Determining Learning Rates

Learning rate for the covariance matrix



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

optimal condition number for \mathbf{C} is one,

initial condition number of \mathbf{C} equals 10^4

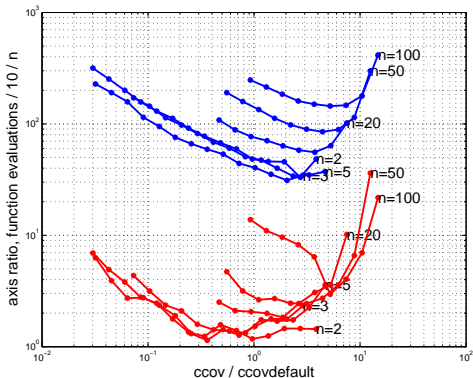
shown are single runs

x-axis: learning rate for the covariance matrix

y-axis: square root of final **condition number** of \mathbf{C} (**red**),
number of **function evaluations** to reach f_{stop} (**blue**)

Determining Learning Rates

Learning rate for the covariance matrix



x-axis: factor for learning rate for the covariance matrix

y-axis: square root of final **condition number** of \mathbf{C} (red),
number of **function evaluations** to reach f_{stop} (blue)

- learning rates can be identified on simple functions

exploiting invariance properties

- the outcome depends on the problem dimensionality
- the specific objective function is rather insignificant

... step size control

EMNA versus CMA

Both algorithms use the same sample distribution

$$\mathbf{x}_i = \mathbf{m} + \sigma \mathbf{y}_i, \quad \mathbf{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C})$$

In EMNA_{global} $\sigma \equiv 1$ and

$$\mathbf{m} \leftarrow \frac{1}{\mu} \sum_{i=1}^{\mu} \mathbf{x}_{i:\lambda}$$

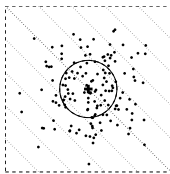
$$\mathbf{C} \leftarrow \frac{1}{\mu} \sum_{i=1}^{\mu} (\mathbf{x}_{i:\lambda} - \mathbf{m})(\mathbf{x}_{i:\lambda} - \mathbf{m})^T$$

In CMA, for $c_{\text{cov}} = 1$, with rank- μ update only

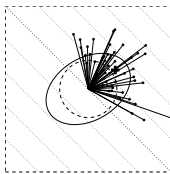
$$\mathbf{C} \leftarrow \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda} \frac{\mathbf{x}_{i:\lambda} - \mathbf{m}}{\sigma} \frac{(\mathbf{x}_{i:\lambda} - \mathbf{m})^T}{\sigma}$$

$$\mathbf{m} \leftarrow \sum_{i=1}^{\mu} w_i \mathbf{x}_{i:\lambda}$$

Remark: order of updates!

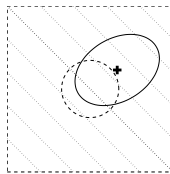


$$x_i = m_{\text{old}} + y_i, \quad y_i \sim \mathcal{N}(0, \mathbf{C})$$



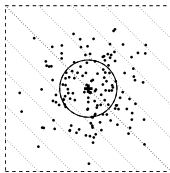
$$\mathbf{C} \leftarrow$$

$$\frac{1}{\mu} \sum (x_{i:\lambda} - m_{\text{old}})(x_{i:\lambda} - m_{\text{old}})^T$$

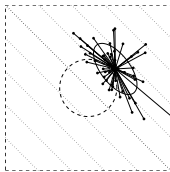


$$m_{\text{new}} = m_{\text{old}} + \frac{1}{\mu} \sum y_{i:\lambda}$$

rank- μ CMA
conducts a
PCA of
steps

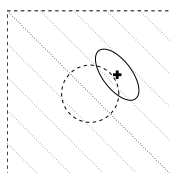


$$x_i = m_{\text{old}} + y_i, \quad y_i \sim \mathcal{N}(0, \mathbf{C})$$



$$\mathbf{C} \leftarrow$$

$$\frac{1}{\mu} \sum (x_{i:\lambda} - m_{\text{new}})(x_{i:\lambda} - m_{\text{new}})^T$$



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

EMNA_{global}
conducts a
PCA of
points

sampling of $\lambda = 150$
solutions (dots) where

$$\mathbf{C} = \mathbf{I} \text{ and } \sigma = 1$$

calculating \mathbf{C} where

$$\mu = 50,$$

$$w_1 = \dots = w_\mu = \frac{1}{\mu}, \text{ and}$$

$$c_{\text{cov}} = 1$$

new distribution

the CMA-update yields a larger variance in particular in gradient direction

is the minimizer for the variance when calculating \mathbf{C}

Population Size on Unimodal Functions

On unimodal functions the performance degrades at most linearly with increasing population size.

most often a small population size, $\lambda \leq 10$, is optimal

Problem Formulation

A real world problem requires

- a representation; the encoding of problem parameters into $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n$
- the definition of a objective function $f : \mathbf{x} \mapsto f(\mathbf{x})$ to be minimized

One might distinguish two approaches

Natural Encoding

Use a “natural” encoding and **design the optimizer** with respect to the problem e.g. use of specific “genetic operators”

frequently done in discrete domain

Concerned Encoding (Pure Black Box)

Put problem specific knowledge into the encoding and use a “**generic**” optimizer

frequently done in continuous domain

Advantage: Sophisticated and well-validated optimizers can be used

...interface to real world problems