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

GECCO'08, July 12, 2008, Atlanta, Georgia, USA.
ACM 978-1-60558-131-6/08/07

Content

Problem Statement

Continuous Domain Search/Optimization

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

$$f : \mathcal{X} \subseteq \mathbb{R}^n \to \mathbb{R}, \qquad \boldsymbol{x} \mapsto f(\boldsymbol{x})$$

- **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

    . . . or to a robust solution $\boldsymbol{x}$
  - solution $\boldsymbol{x}$ with **small function value** with **least search cost**

    there are two conflicting objectives
- Typical Examples
  - shape optimization (e.g. using CFD) — curve fitting, airfoils
  - model calibration — biological, physical
  - parameter calibration — 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 $f : \mathcal{X} \subset \mathbb{R}^n \to \mathbb{R}$ 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

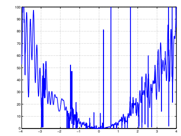
- multimodal
  there are eventually many local optima
- non-smooth
  derivatives do not exist
- discontinuous
- ill-conditioned
- noisy
- . . .

Goal : cope with any of these function properties
they are related to real-world problems

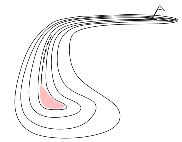
What Makes a Function Difficult to Solve?

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

Separable Problems

Definition (Separable Problem)

A function f is separable if

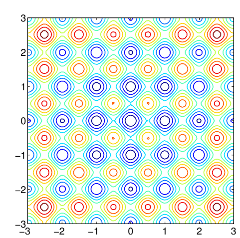
$$\arg\min_{(x_1,\ldots,x_n)} f(x_1,\ldots,x_n) = \left(\arg\min_{x_1} f(x_1,\ldots),\ldots,\arg\min_{x_n} f(\ldots,x_n)\right)$$

⇒ 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



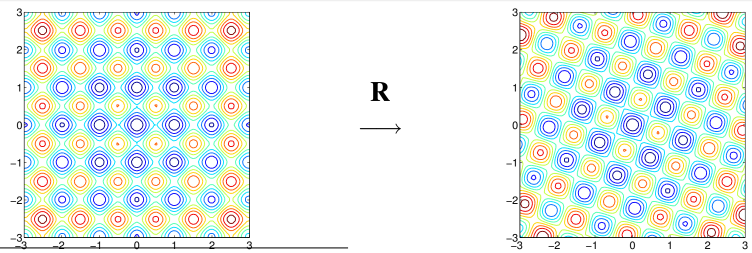
Non-Separable Problems

Building a non-separable problem from a separable one

Rotating the coordinate system

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

$\mathbf{R}$ rotation matrix



[1,2]

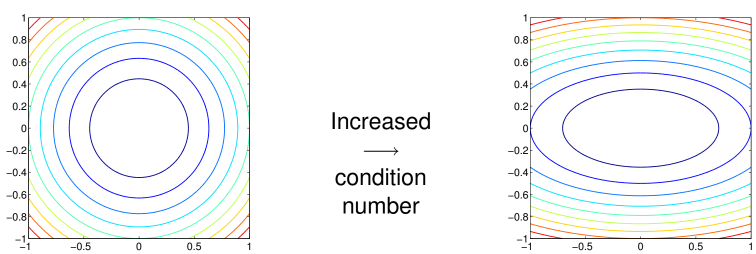
[1] Hansen, Ostermeier, Gawelczyk (1995). On the adaptation of arbitrary normal mutation distributions in evolution strategies: The generating set adaptation. Sixth ICGA, pp. 57-64, Morgan Kaufmann

[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

Ill-Conditioned Problems

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

ill-conditioned means **"squeezed" lines** of equal function value



consider the curvature of iso-fitness lines

What Makes a Function Difficult to Solve?

... and what can be done

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

Stochastic Search

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

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

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

Everything depends on the definition of P and F_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*

Evolution Strategies and Normal Estimation of Distribution Algorithms

New search points are sampled normally distributed

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

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

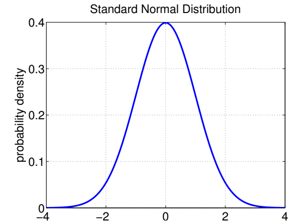
where

- 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 $\mathbf{C} \in \mathbb{R}^{n \times n}$ determines the **shape** of the distribution ellipsoid

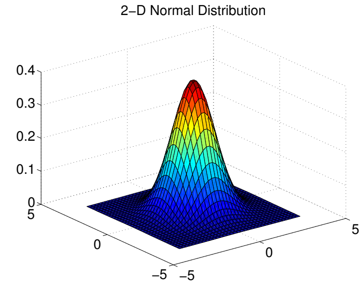
The question remains how to update m, $\mathbf{C}$, and σ.

Normal Distribution

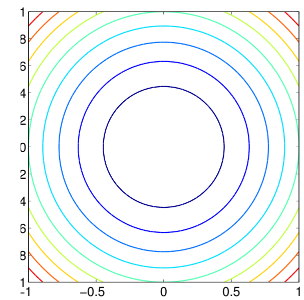
Isotropic Case



probability density of 1-D standard normal distribution



2-D

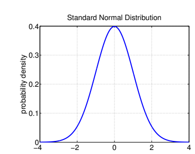


The Multi-Variate (n-Dimensional) Normal Distribution

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

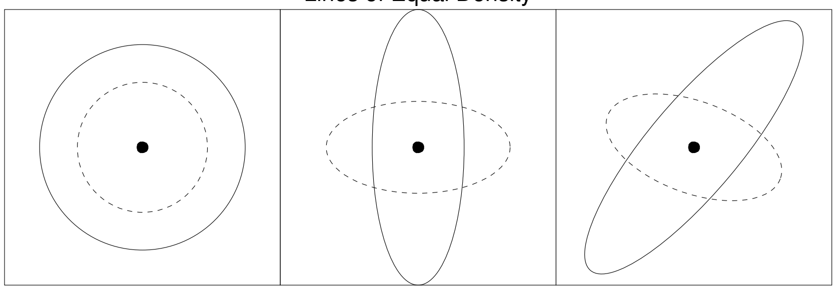
The **mean** value m

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



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

Lines of Equal Density



$\mathcal{N}(\boldsymbol{m}, \sigma^2\mathbf{I}) \sim \boldsymbol{m} + \sigma\mathcal{N}(\mathbf{0}, \mathbf{I})$
one degree of freedom σ
components of $\mathcal{N}(\mathbf{0}, \mathbf{I})$ are independent standard normally distributed

$\mathcal{N}(\boldsymbol{m}, \mathbf{D}^2) \sim \boldsymbol{m} + \mathbf{D}\,\mathcal{N}(\mathbf{0}, \mathbf{I})$
n **degrees of freedom**
components are independent, scaled

$\mathcal{N}(\boldsymbol{m}, \mathbf{C}) \sim \boldsymbol{m} + \mathbf{C}^{\frac{1}{2}}\mathcal{N}(\mathbf{0}, \mathbf{I})$
$(n^2+n)/2$ **degrees of freedom**
components are correlated

...CMA

Evolution Strategies

$(\mu \overset{+}{,} \lambda)$ μ: # parents, λ: # offspring

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

$(1+1)$-ES

Sample one offspring from parent $\boldsymbol{m}$

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

If $\boldsymbol{x}$ better than $\boldsymbol{m}$ select

$$\boldsymbol{m} \leftarrow \boldsymbol{x}$$

...why?

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

Non-elitist selection and intermediate (weighted) recombination

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

Let $\boldsymbol{x}_{i:\lambda}$ the **i-th ranked** solution point, such that $f(\boldsymbol{x}_{1:\lambda}) \leq \cdots \leq f(\boldsymbol{x}_{\lambda:\lambda})$.
The new mean reads

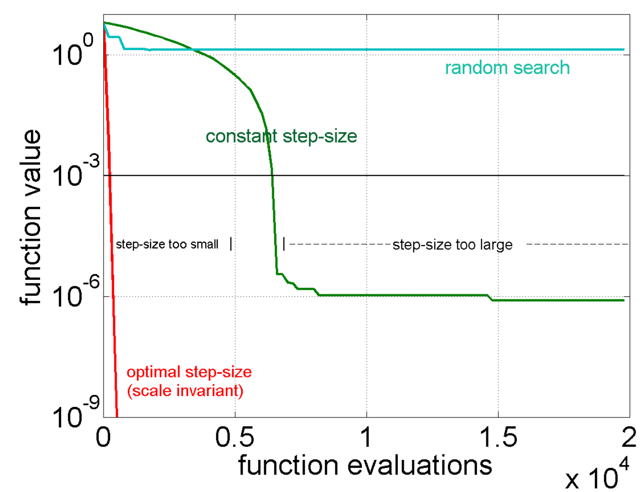
$$\boldsymbol{m} \leftarrow \sum_{i=1}^{\mu} w_i\,\boldsymbol{x}_{i:\lambda} \;=\; \boldsymbol{m} + \sigma \underbrace{\sum_{i=1}^{\mu} w_i\,\boldsymbol{y}_{i:\lambda}}_{=:\boldsymbol{y}_w}$$

where

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

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

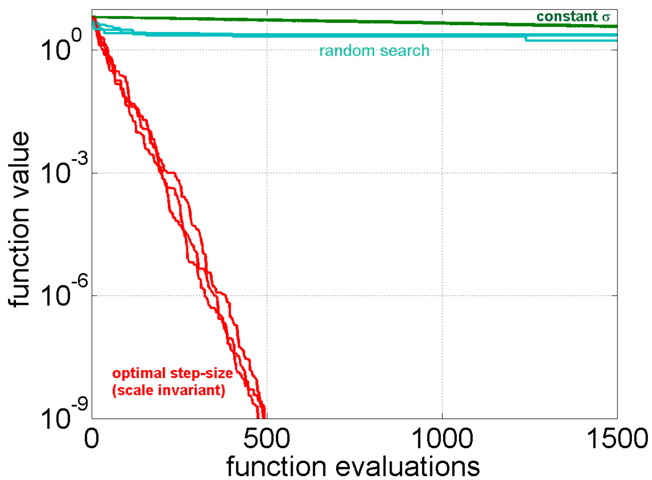
Why Step-Size Control?



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

in $[-0.2, 0.8]^n$
for $n = 10$

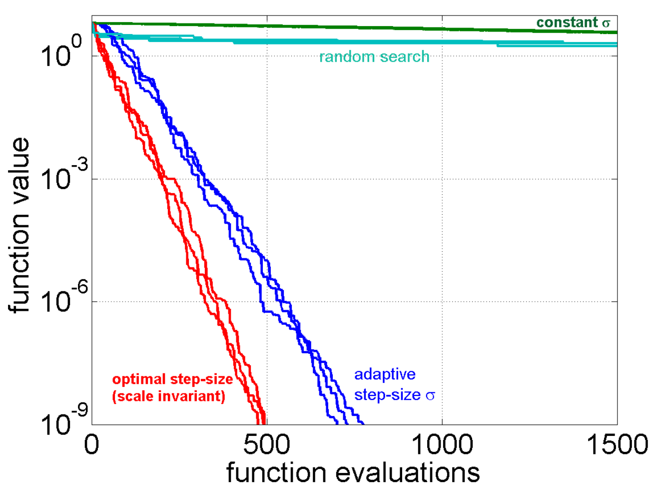
Why Step-Size Control?



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

in $[-0.2, 0.8]^n$
for $n = 10$

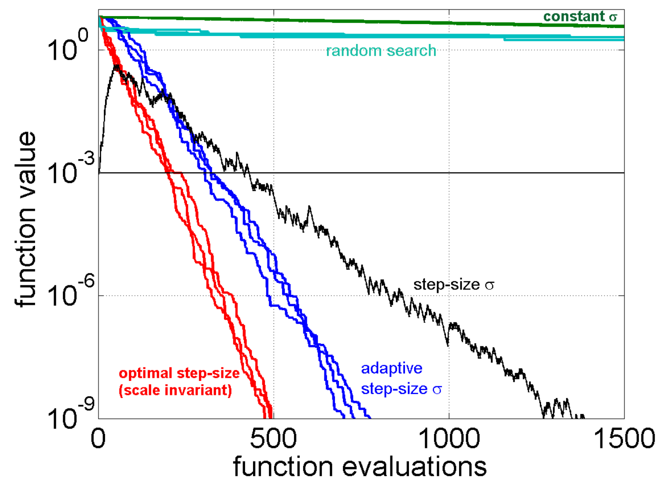
Why Step-Size Control?



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

in $[-0.2, 0.8]^n$
for $n = 10$

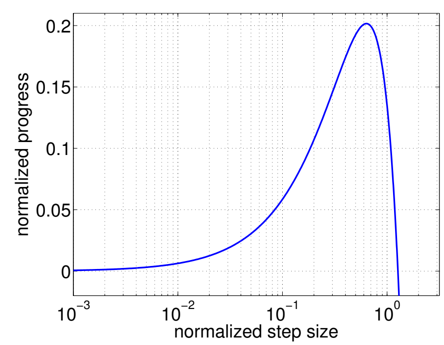
Why Step-Size Control?



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

in $[-0.2, 0.8]^n$
for $n = 10$

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

  simplified "global" self-adaptation

- path length control[d] (Cumulative Step-size Adaptation, CSA)[e], applied with ","-selection

[a] Rechenberg 1973, *Evolutionsstrategie, Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, Frommann-Holzboog

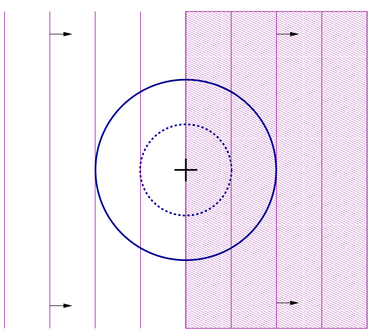
[b] Schumer and Steiglitz 1968. Adaptive step size random search. *IEEE TAC*

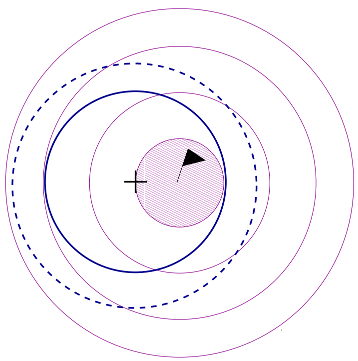
[c] Schwefel 1981, *Numerical Optimization of Computer Models*, Wiley

[d] Hansen & Ostermeier 2001, Completely Derandomized Self-Adaptation in Evolution Strategies, *Evol. Comput. 9(2)*

[e] Ostermeier *et al* 1994, Step-size adaptation based on non-local use of selection information, *PPSN IV*

One-fifth success rule

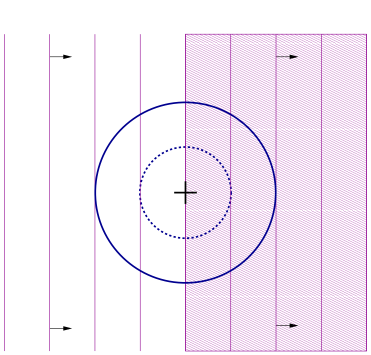


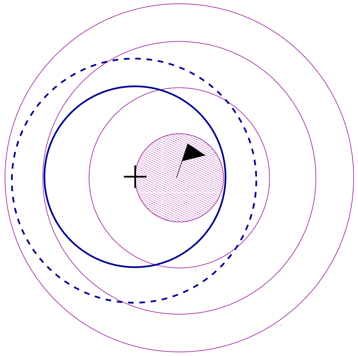


↓ increase σ

↓ decrease σ

One-fifth success rule





Proba of success (p_s) — $1/2$

$1/5$

Proba of success (p_s) — "too small"

One-fifth success rule

Let p_s: $\#$ of successful offspring / generation

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

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

$(1+1)$-ES

$p_{target} = 1/5$
IF *offspring better parent*
  $p_s = 1$
ELSE
  $p_s = 0$

Self-adaptation

in a $(1, \lambda)$-ES

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

step-size $\quad \sigma_i \leftarrow \sigma \exp(\tau N(0,1))$

parent $\quad x_i \leftarrow x + \sigma_i \mathcal{N}(0, \mathbf{I})$

EVALUATE

SELECT

Best offspring x_* with its step-size σ_*

Rationale

Unadapted step-size won't produce successive good individuals

"The step-size are adjusted by the evolution itself"

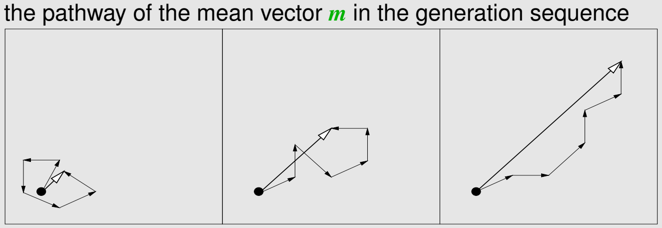
Path Length Control

The Concept

$$\boldsymbol{x}_i = \boldsymbol{m} + \sigma \boldsymbol{y}_i$$
$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \boldsymbol{y}_w$$

Measure the length of the *evolution path*

the pathway of the mean vector $\boldsymbol{m}$ in the generation sequence



↓ decrease σ ↓ increase σ

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

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

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

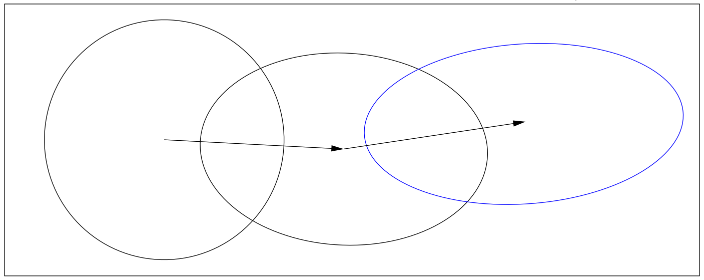
$$\sigma \leftarrow \sigma \times \underbrace{\exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_\sigma\|}{\mathsf{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\|} - 1\right)\right)}_{>1 \iff \|\boldsymbol{p}_\sigma\| \text{ is greater than its expectation}} \qquad \text{update step-size}$$

. . . CMA in a nutshell

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,

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

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

. . . equations

Covariance Matrix Adaptation

Rank-One Update

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

While not terminate

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

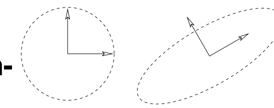
$$\mathbf{m} \leftarrow \mathbf{m} + \sigma \mathbf{y}_w \qquad \text{where } \mathbf{y}_w = \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda}$$

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

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

covariance matrix adaptation

- learns all **pairwise dependencies** between variables
  off-diagonal entries in the covariance matrix reflect the dependencies
- conducts a **principle component analysis** (PCA) of steps $\mathbf{y}_w$, sequentially in time and space
  eigenvectors of the covariance matrix $\mathbf{C}$ are the principle components / the principle axes of the mutation ellipsoid
- learns a new, **rotated problem representation** and a **new metric** (Mahalanobis)
  components are independent (only) in the new representation



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

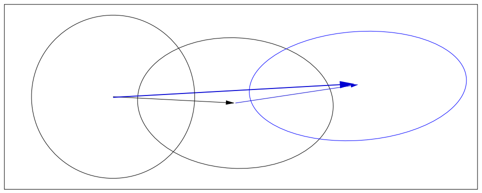
. . . cumulation, rank-μ, step-size control

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 $\boldsymbol{m}$.



An exponentially weighted sum of steps $\boldsymbol{y}_w$ is used

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

The recursive construction of the evolution path (cumulation):

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

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

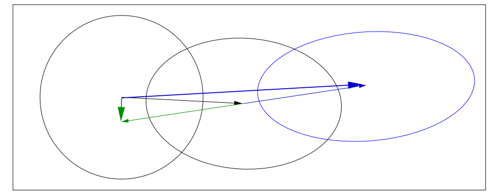
- *exponential smoothing* in time series, forecasting
- exponentially weighted *mooving average*
- *iterate averaging* in stochastic approximation
- *momentum* in the back-propagation algorithm for ANNs
- ...

...why?

Cumulation

Utilizing the Evolution Path

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



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

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

...equations

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]

[a] Hansen, 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

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

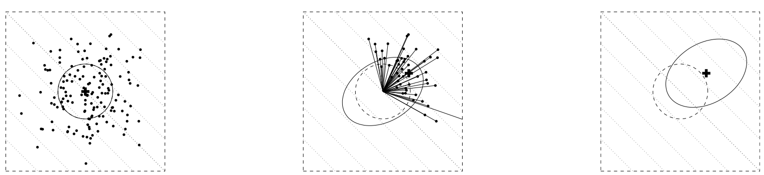
The rank-μ update extends the update rule for **large population sizes** λ using $\mu > 1$ vectors to update $\mathbf{C}$ at each generation step.
The matrix

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

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

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

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



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

$\mathbf{C}_\mu = \frac{1}{\mu} \sum \boldsymbol{y}_{i:\lambda} \boldsymbol{y}_{i:\lambda}^{\mathrm{T}}$
$\mathbf{C} \leftarrow (1 - 1) \times \mathbf{C} + 1 \times \mathbf{C}_\mu$

$\boldsymbol{m}_{\text{new}} \leftarrow \boldsymbol{m} + \frac{1}{\mu} \sum \boldsymbol{y}_{i:\lambda}$

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

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

new distribution

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 $\mathcal{O}(n)$[3]
  given $\mu_w \propto \lambda \propto n$

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

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...

[3] Hansen, 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

Estimation of Distribution Algorithms

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

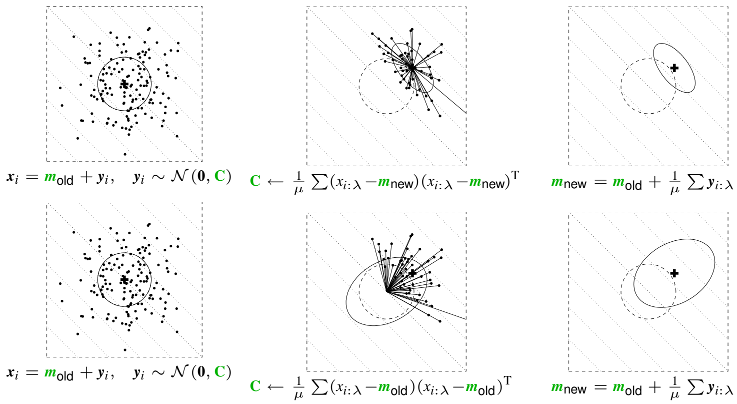
Example: EMNA (Estimation of Multi-variate Normal Algorithm

Initialize $\boldsymbol{m} \in \mathbb{R}^n$, and $\mathbf{C} = \mathbf{I}$
While not terminate

$$\begin{aligned} \boldsymbol{x}_i &= \boldsymbol{m} + \boldsymbol{y}_i, \qquad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C}), \qquad \text{for } i = 1, \ldots, \lambda \\ \mathbf{C} &\leftarrow \sum_{i=1}^{\mu} (\boldsymbol{x}_{i:\lambda} - \boldsymbol{m})(\boldsymbol{x}_{i:\lambda} - \boldsymbol{m})^{\mathrm{T}} \\ \boldsymbol{m} &\leftarrow \sum_{i=1}^{\mu} \boldsymbol{x}_{i:\lambda} \end{aligned}$$

Larrañaga and Lozano 2002. *Estimation of Distribution Algorithms*

Estimation of Multivariate Normal Algorithm EMNA$_{\text{global}}$ versus rank-μ CMA[4]



EMNA$_{\text{global}}$ conducts a PCA of **points**

rank-μ CMA conducts a PCA of **steps**

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 $\mathbf{C}$

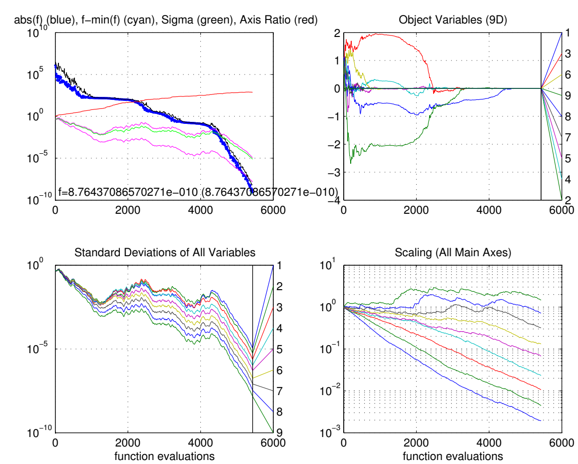
[4] Hansen, N. (2006). The CMA Evolution Strategy: A Comparing Review. In J.A. Lozano, P. Larranga, I. Inza and E. Bengoetxea (Eds.). Towards a new evolutionary computation. Advances in estimation of distribution algorithms. pp. 75-102

What did we achieve?

1. Covariance matrix adaptation: reduce any convex quadratic function
$$f(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{x}$$
to the sphere model
$$f(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}$$
without use of derivatives
lines of equal density align with lines of equal fitness $\mathbf{C} \propto \boldsymbol{H}^{-1}$
2. Step-size control: converge log-linearly on the sphere
3. Rank-based selection: the same holds for any $g(f(\boldsymbol{x})) = g\left(\boldsymbol{x}^{\mathrm{T}}\mathbf{H}\boldsymbol{x}\right)$
$g: \mathbb{R} \to \mathbb{R}$ stricly monotonic (order preserving)

Experimentum Crucis (1)

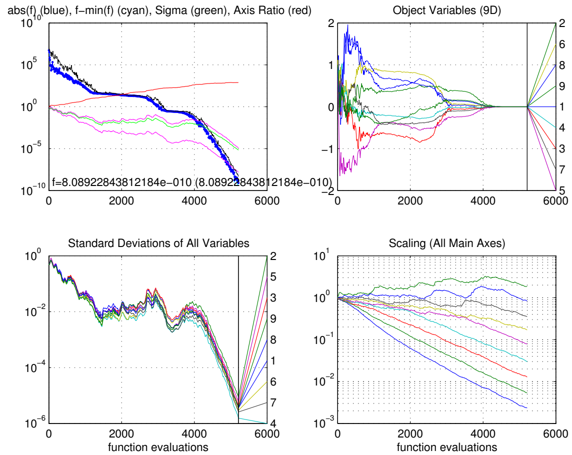
f convex quadratic, separable



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

Experimentum Crucis (2)

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



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

$f(\boldsymbol{x}) = g\left(\boldsymbol{x}^{\mathrm{T}}\mathbf{H}\boldsymbol{x}\right)$, $g : \mathbb{R} \to \mathbb{R}$ stricly monotonic

. . . internal parameters

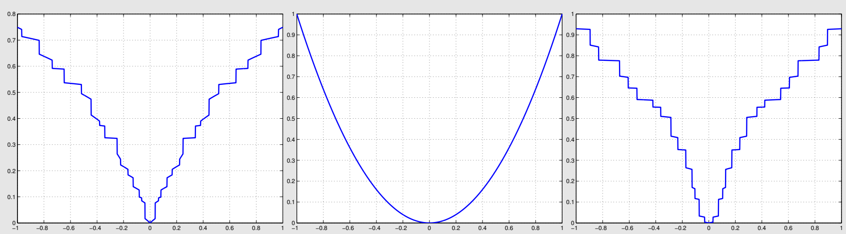
Invariance Under Strictly Monotonically Increasing Functions

Rank-based algorithms

Selection based on the rank:

$$f(x_{1:\lambda}) \leq f(x_{2:\lambda}) \leq ... \leq f(x_{\lambda:\lambda})$$

Update of all parameters uses only the rank



$$g(f(x_{1:\lambda})) \leq g(f(x_{2:\lambda})) \leq ... \leq g(f(x_{\lambda:\lambda}))$$