Advanced Optimization
Lecture 6: Stochastic Algorithms II
(SGD & CMA-ES)

November 17, 2021

CentraleSupélec / ESSEC Business School

dimo.brockhoff@inria.fr

Dimo Brockhoff

Inria Saclay – Ile-de-France

Course Overview

		Topic
Wed, 13.10.2021	PM	Introduction, examples of problems, problem types
Wed, 20.10.2021	PM	Continuous (unconstrained) optimization: convexity, gradients, Hessian, … [technical test Evalmee]
Wed, 27.10.2021	PM	Continous optimization II: [1st mini-exam] Constrained optimization: Lagrangian, optimality conditions
Wed, 03.11.2021	PM	gradient descent, Newton direction, quasi-Newton (BFGS) Linear programming: duality, maxflow/mincut, simplex algo
Wed, 10.11.2021	PM	derivative-free algorithms: Nelder-Mead and CMA-ES
Wed, 17.11.2021	PM	CMA-ES, Part II, Stochastic Gradient Descent, Bayesian optimization
Wed, 24.11.2021	PM	Benchmarking solvers: runtime distributions, performance profiles **[2nd mini-exam]**
Tue, 30.11.2021	23:59	Deadline open source project (PDF sent by email)
Wed, 01.12.2021	PM	Discrete optimization: branch and bound, branch and cut, k-means clustering
Wed, 15.12.2021	PM	Exam

Details on Continuous Optimization Lectures

Introduction to Continuous Optimization

- examples and typical difficulties in optimization

Mathematical Tools to Characterize Optima

- reminders about differentiability, gradient, Hessian matrix
- unconstraint optimization
  - first and second order conditions
  - convexity
- constraint optimization
  - Lagrangian, optimality conditions

Gradient-based Algorithms

- gradient descent
- quasi-Newton method (BFGS) and invariances

Linear programming, duality

Learning in Optimization / Optimization in Machine Learning

- Stochastic gradient descent (SGD) + Adam
- CMA-ES (adaptive algorithms / Information Geometry)
- Other derivative-free algorithms: Nelder-Mead, Bayesian opt.

back to

Covariance Matrix Adaptation Evolution Strategy (CMA-ES)

The CMA-ES

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

Initialize: $\mathbf{C} = \mathbf{I}$, and $\boldsymbol{p}_\mathbf{c} = \mathbf{0}$, $\boldsymbol{p}_\sigma = \mathbf{0}$,

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

While not terminate

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

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

$$\boldsymbol{p}_\mathbf{c} \leftarrow (1 - c_\mathbf{c})\,\boldsymbol{p}_\mathbf{c} + \mathbb{1}_{\{\|\boldsymbol{p}_\sigma\| < 1.5\sqrt{n}\}} \sqrt{1 - (1 - c_\mathbf{c})^2}\sqrt{\mu_w}\,\boldsymbol{y}_w \qquad \text{cumulation for } \mathbf{C}$$

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

$$\mathbf{C} \leftarrow (1 - c_1 - c_\mu)\,\mathbf{C} + c_1\,\boldsymbol{p}_\mathbf{c}\boldsymbol{p}_\mathbf{c}^{\mathrm{T}} + \ldots \qquad \text{update } \mathbf{C}$$

$$\sigma \leftarrow \sigma \times \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_\sigma\|}{\mathrm{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\|} - 1\right)\right)$$

Not covered on this slide: terminati

encoding

Goal:
Understand the main principles of this state-of-the-art algorithm.

Copyright Notice

- As in the last lecture, slides are taken partly from **http://www.cmap.polytechnique.fr/~nikolaus.hansen/copenhagen-cma-es.pdf** (copyright by Nikolaus Hansen) and from **http://www.cmap.polytechnique.fr/~dimo.brockhoff/optimizationSaclay/2015/slides/20151106-continuousoptIV.pdf** (copyright by Anne Auger)

- Those are referred to as [Hansen, p. X] and [Auger, p. Y] respectively.

CMA-ES: Stochastic Search Template

A stochastic blackbox search template to minimize $f: \mathbb{R}^n \to \mathbb{R}$

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

While happy do:

- Sample distribution $P(\boldsymbol{x}|\theta) \to \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\lambda \in \mathbb{R}^n$
- Evaluate $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\lambda$ on f
- Update parameters $\theta \leftarrow F_\theta(\theta, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\lambda, f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_\lambda))$

For CMA-ES and evolution strategies in general:

sample distributions = multivariate Gaussian distributions

Step-Size Adaptation

Recap CMA-ES: What We Have So Far

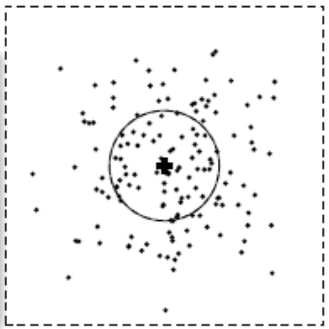
Evolution Strategies

Recalling

New search points are sampled normally distributed

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

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



where

- the mean vector $\boldsymbol{m} \in \mathbb{R}^n$ represents the favorite solution and $\boldsymbol{m} \leftarrow \sum_{i=1}^{\mu} w_i \boldsymbol{x}_{i:\lambda}$
- 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

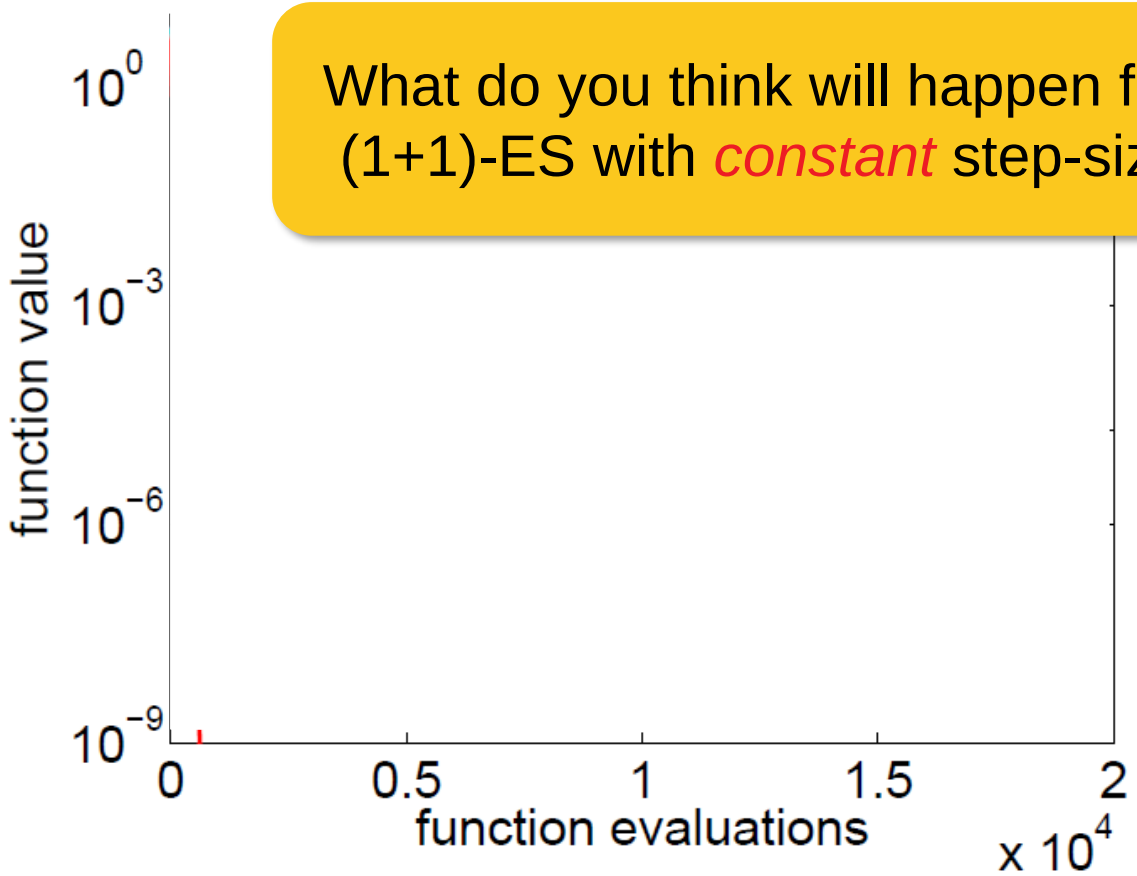
The remaining question is how to update σ and $\mathbf{C}$.

from [Hansen, p. 45]

Why At All Step-Size Adaptation?

Why Step-Size Control?

What do you think will happen for a (1+1)-ES with *constant* step-size?



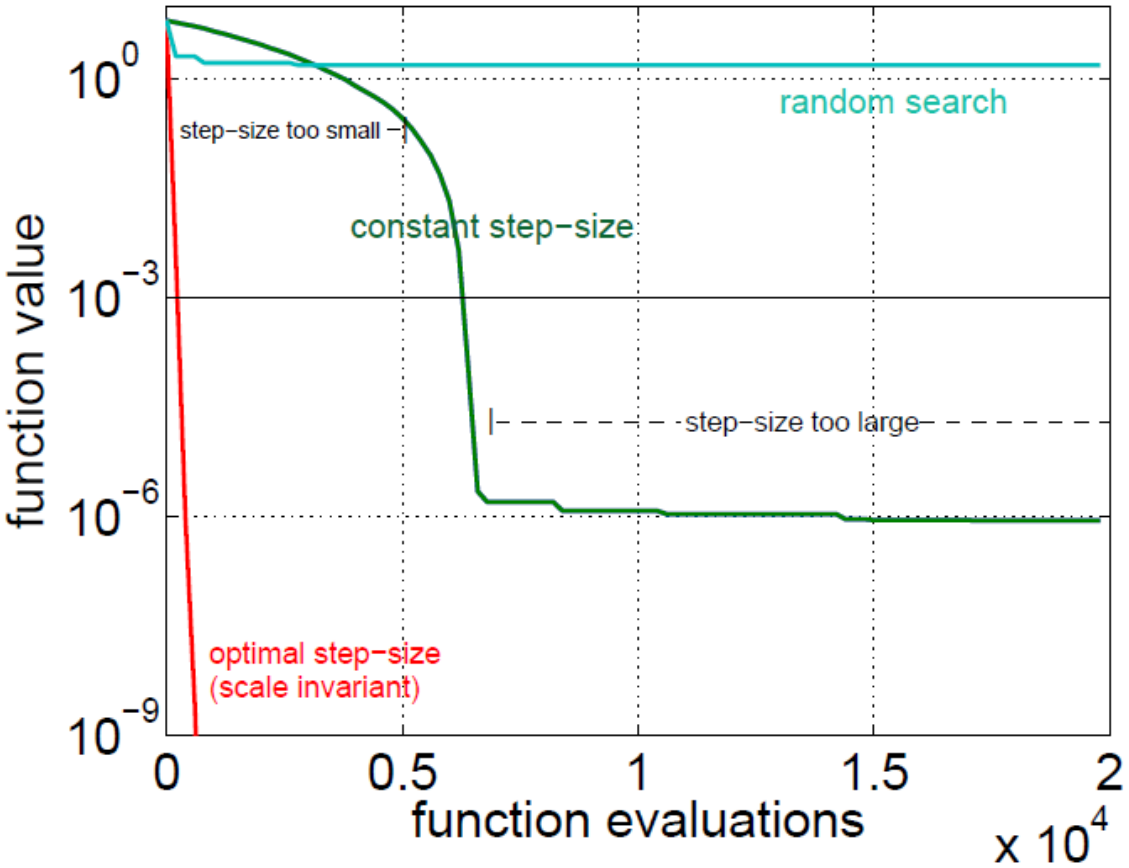
$$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2$$

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

from [Auger, p. 22]

Why Step-Size Adaptation?

Why Step-Size Control?



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

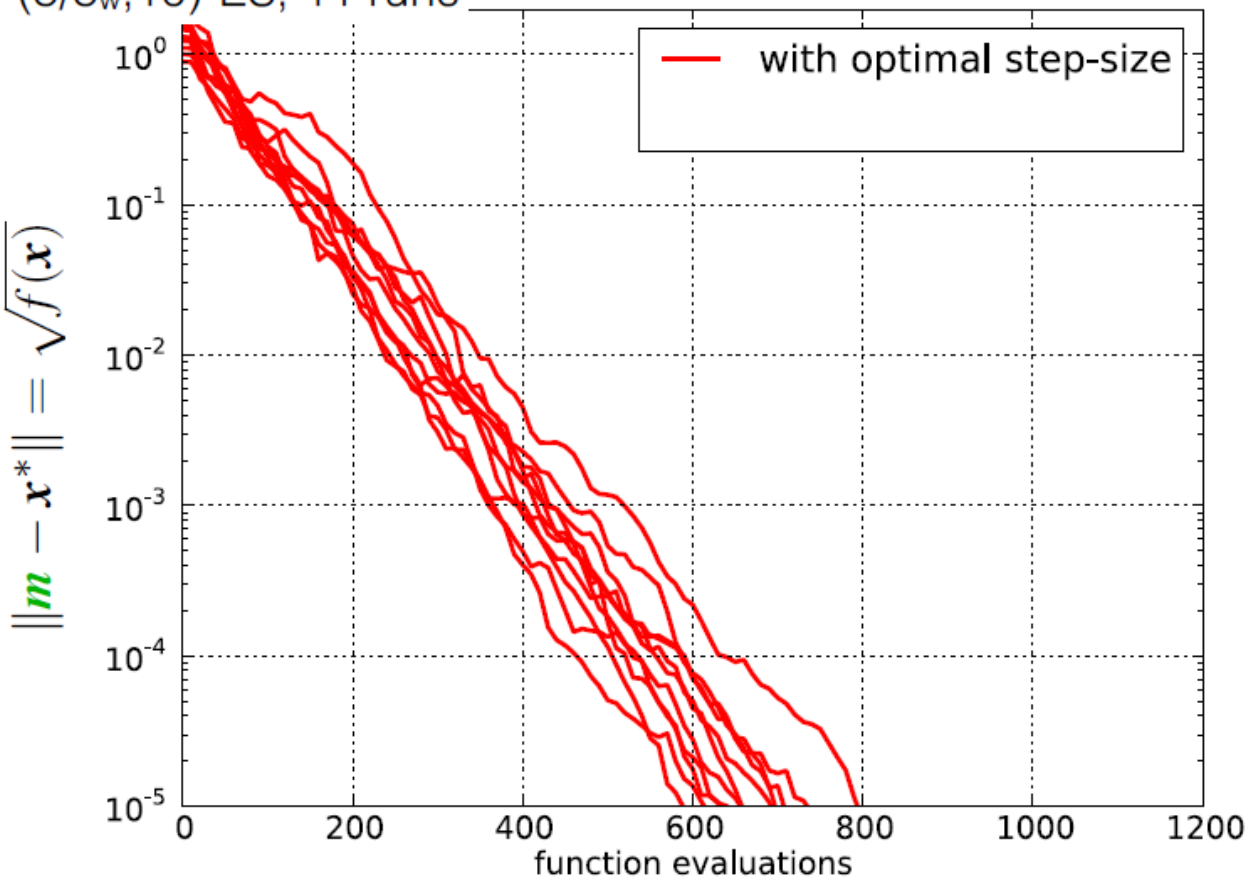
in $[-0.2, 0.8]^n$
for $n = 10$

from [Auger, p. 22]

Optimal Step-Size

Why Step-Size Control?

(5/5$_w$,10)-ES, 11 runs



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

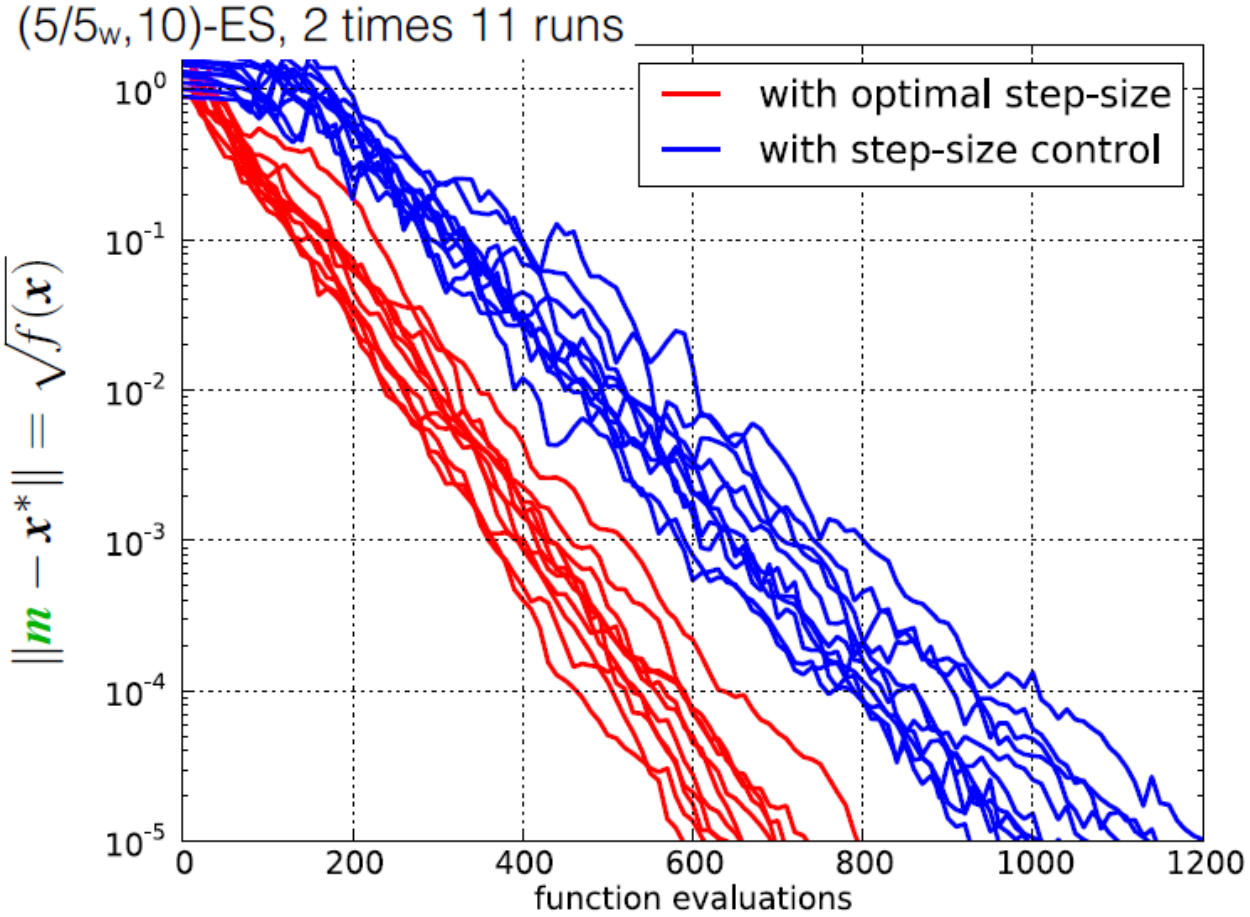
for $n = 10$ and $\boldsymbol{x}^0 \in [-0.2, 0.8]^n$

with optimal step-size σ

from [Hansen, p. 47]

Optimal Step-Size vs. Step-Size Control

Why Step-Size Control?



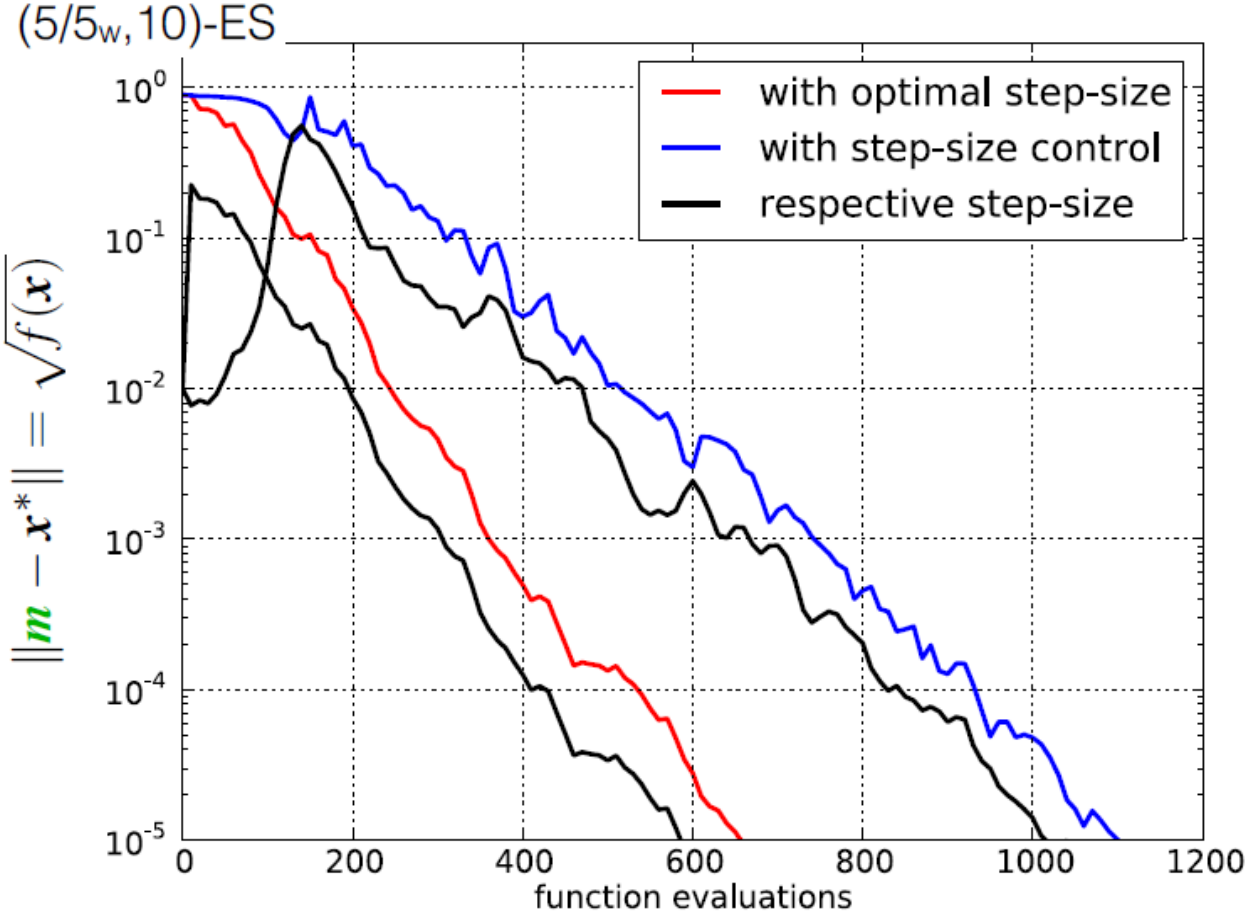
$$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2$$

for $n = 10$ and $\boldsymbol{x}^0 \in [-0.2, 0.8]^n$

with optimal versus adaptive step-size σ with too small initial σ

from [Hansen, p. 48]

Why Step-Size Control?



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

for $n = 10$ and $\boldsymbol{x}^0 \in [-0.2, 0.8]^n$

comparing number of f-evals to reach $\|\boldsymbol{m}\| = 10^{-5}$: $\frac{1100-100}{650} \approx 1.5$

from [Hansen, p. 49]

Adapting the Step-Size

Question:

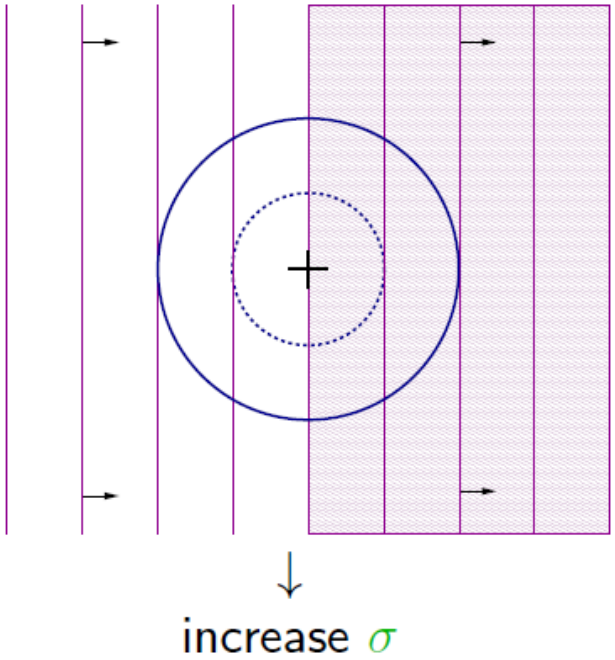
How to actually adapt the step-size during the optimization?

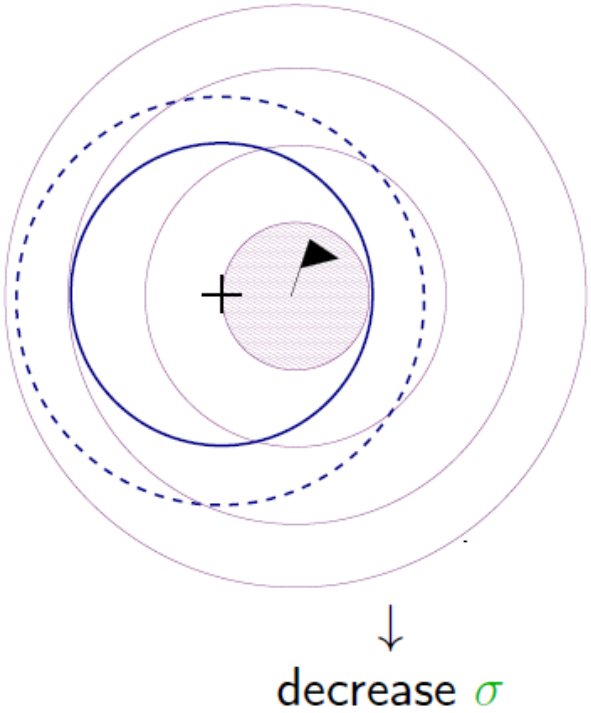
Most common:

- 1/5 success rule
- Cumulative Step-Size Adaptation (CSA, as in standard CMA-ES)
- others possible (Two-Point Adaptation, self-adaptive step-size, ...)

One-Fifth Success Rule

One-fifth success rule

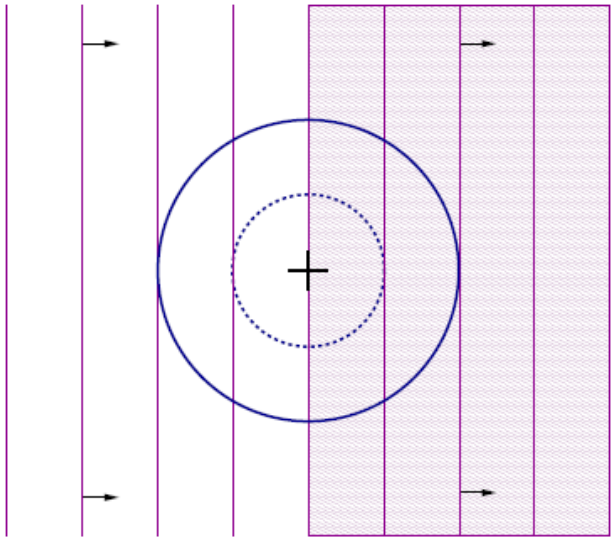




from [Auger, p. 32]

One-Fifth Success Rule

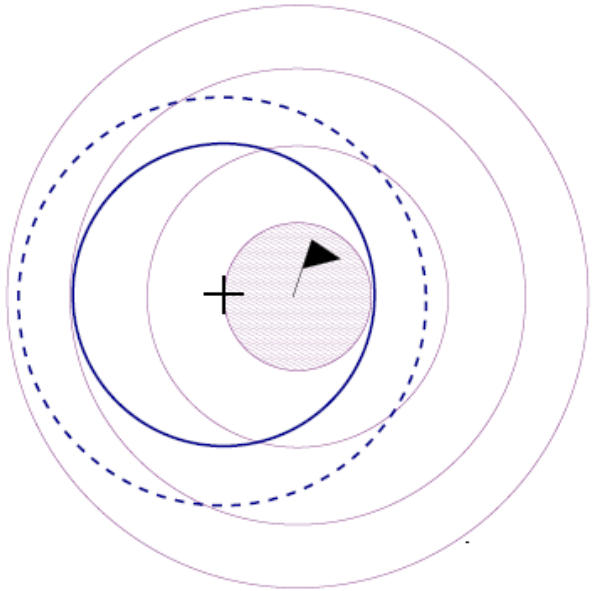
One-fifth success rule



Probability of success (p_s)

$1/2$

$1/5$



Probability of success (p_s)

"too small"

from [Auger, p. 33]

One-Fifth Success Rule

One-fifth success rule

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

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

Increase 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*

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

ELSE

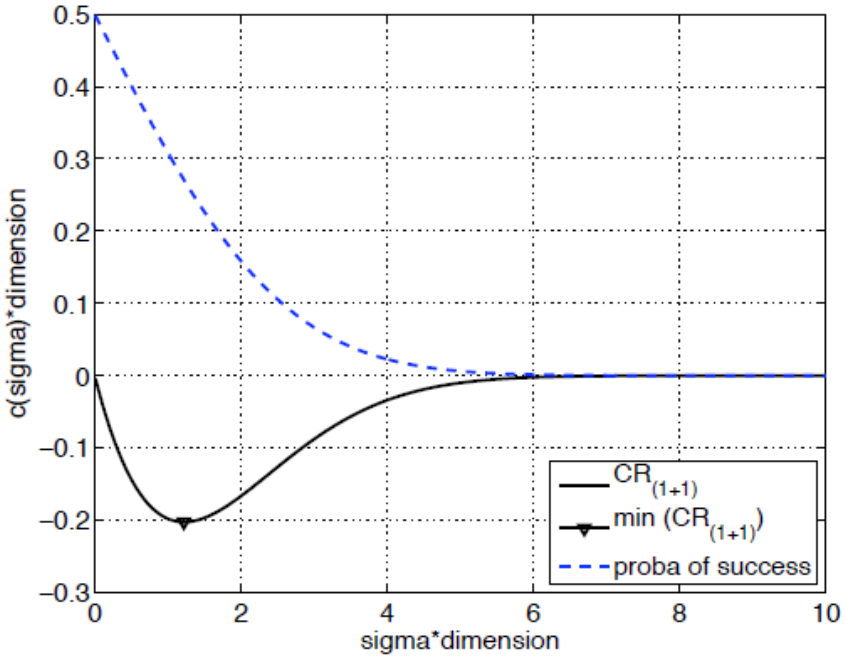
$p_s = 0$, $\sigma \leftarrow \sigma / \exp(1/3)^{1/4}$

from [Auger, p. 34]

One-Fifth Success Rule

Why 1/5?

Asymptotic convergence rate and probability of success of scale-invariant step-size (1+1)-ES



sphere - asymptotic results, i.e. $n = \infty$ (see slides before)

1/5 trade-off of optimal probability of success on the sphere and corridor

from [Auger, p. 35]

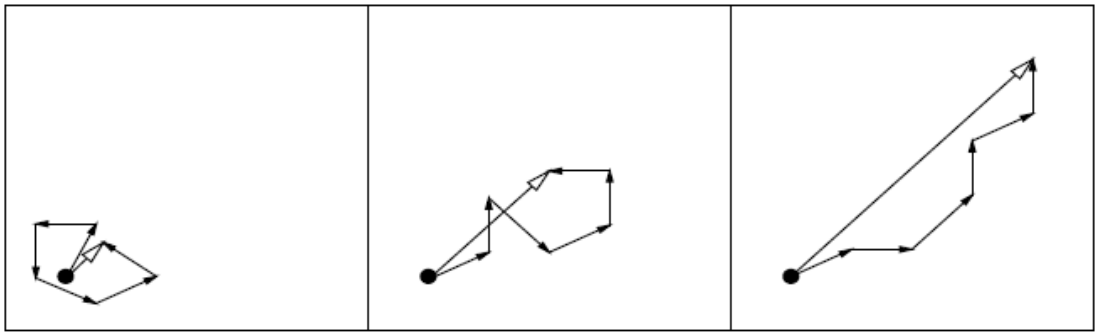
Path Length Control (CSA)

The Concept of Cumulative Step-Size Adaptation

$$x_i = m + \sigma y_i$$
$$m \leftarrow m + \sigma y_w$$

Measure the length of the *evolution path*

the pathway of the mean vector m in the generation sequence



$\Downarrow$ decrease σ

$\Downarrow$ increase σ

from [Auger, p. 36]

Path Length Control (CSA)

The Equations

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

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

$$p_\sigma \leftarrow (1 - c_\sigma)\, p_\sigma + \underbrace{\sqrt{1 - (1 - c_\sigma)^2}}_{\text{accounts for } 1-c_\sigma} \underbrace{\sqrt{\mu_w}}_{\text{accounts for } w_i} y_w$$

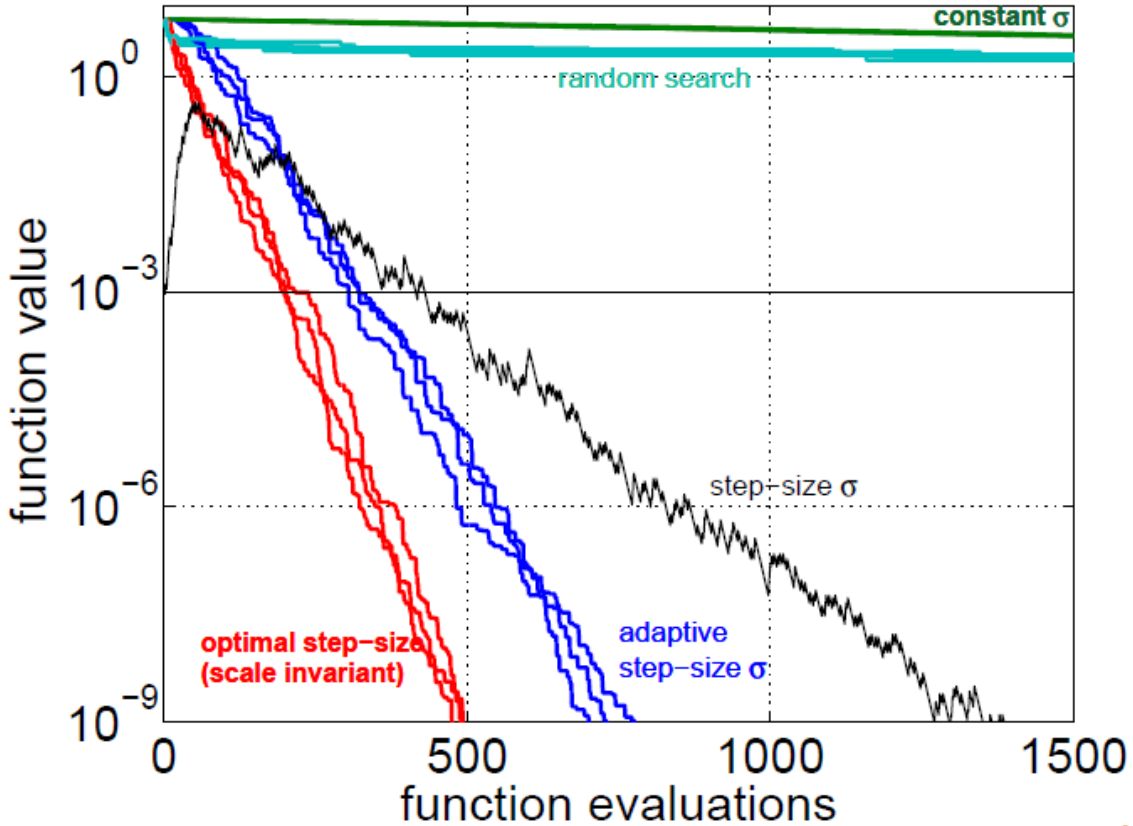
$$\sigma \leftarrow \sigma \times \underbrace{\exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|p_\sigma\|}{\mathrm{E}\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|} - 1\right)\right)}_{>1 \iff \|p_\sigma\| \text{ is greater than its expectation}} \qquad \text{update step-size}$$

from [Auger, p. 37]

Cumulative Step-Size Adaptation (CSA)

Step-size adaptation

What is achived



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

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

Linear convergence

from [Auger, p. 38]

Covariance Matrix Adaptation

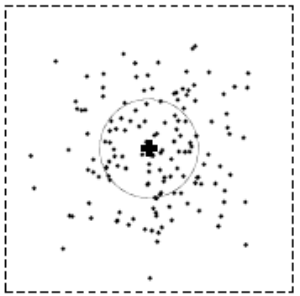
Recap CMA-ES: What We Have So Far

Evolution Strategies

Recalling

New search points are sampled normally distributed

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



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

where

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

The remaining question is how to update C.

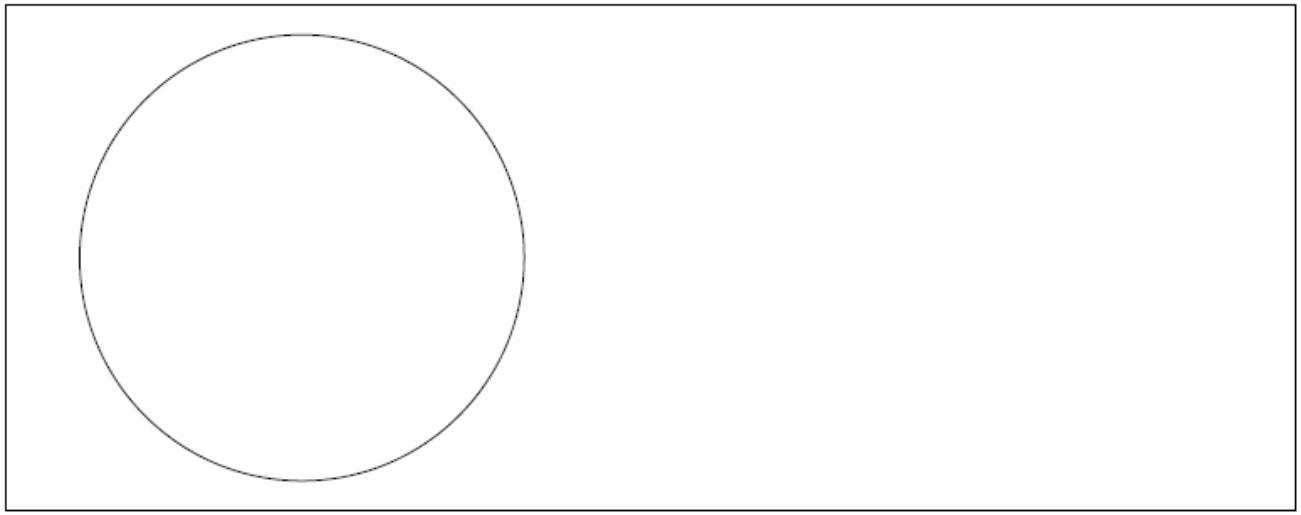
from [Auger, p. 40]

Rank-One Update of Covariance Matrix

Covariance Matrix Adaptation

Rank-One Update

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



initial distribution, $\mathbf{C} = \mathbf{I}$

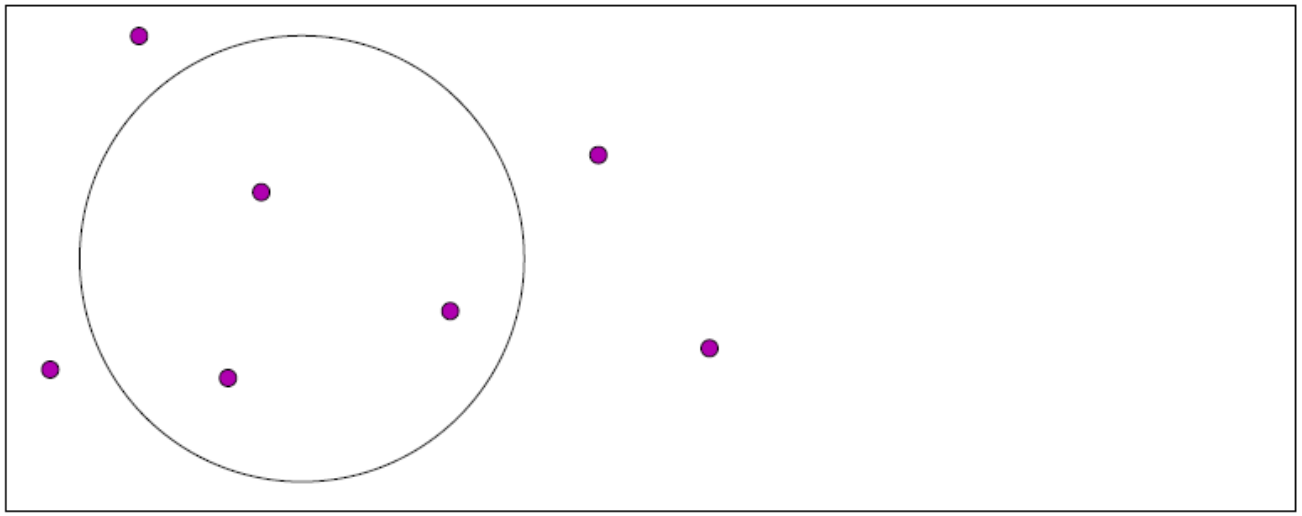
from [Auger, p. 41]

Rank-One Update of Covariance Matrix

Covariance Matrix Adaptation

Rank-One Update

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



initial distribution, $\mathbf{C} = \mathbf{I}$

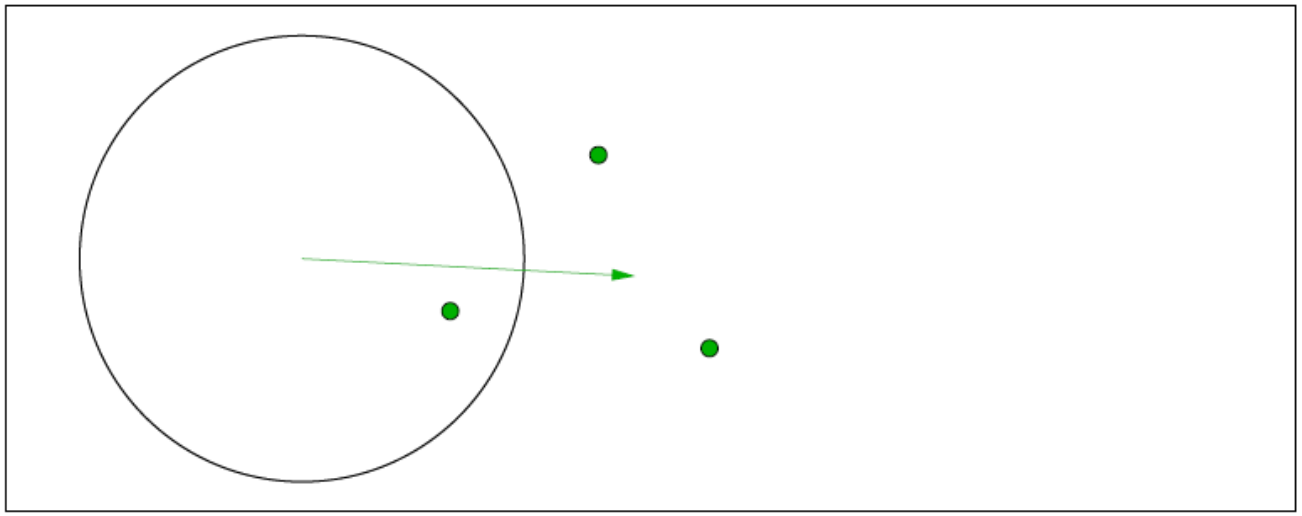
from [Auger, p. 41]

Rank-One Update of Covariance Matrix

Covariance Matrix Adaptation

Rank-One Update

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



y_w, movement of the population mean m (disregarding σ)

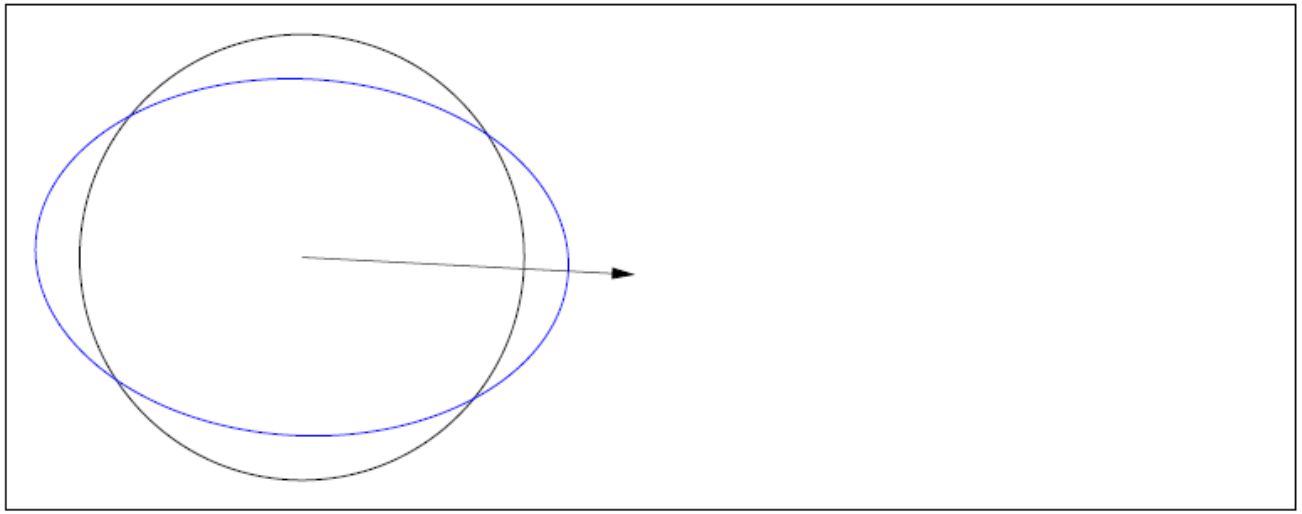
from [Auger, p. 41]

Rank-One Update of Covariance Matrix

Covariance Matrix Adaptation

Rank-One Update

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



mixture of distribution $\mathbf{C}$ and step $\boldsymbol{y}_w$,
$\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$

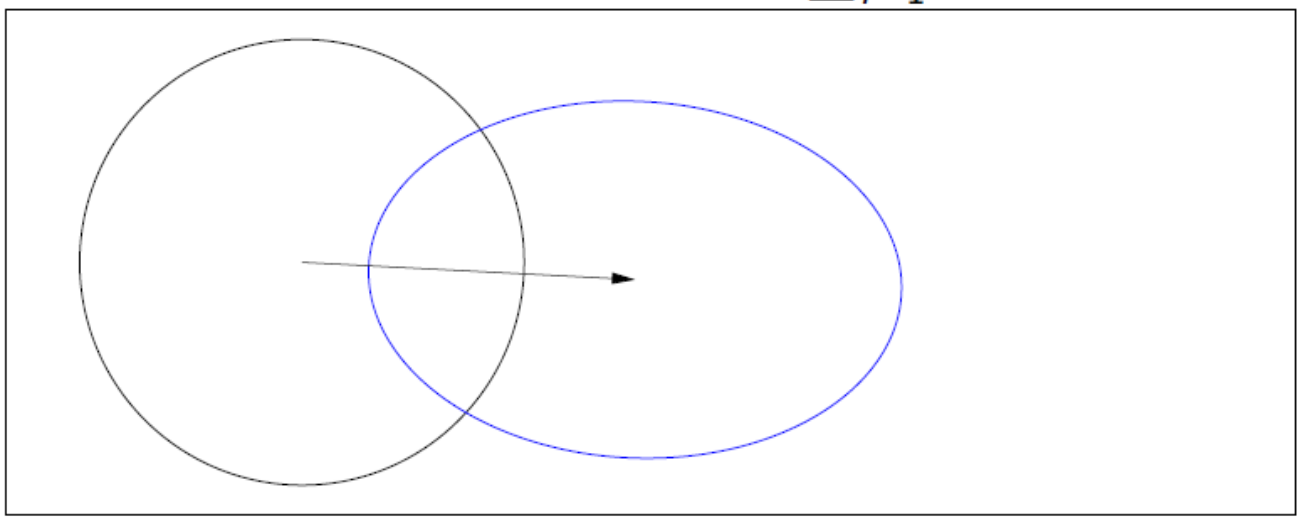
from [Auger, p. 41]

Rank-One Update of Covariance Matrix

Covariance Matrix Adaptation

Rank-One Update

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



new distribution (disregarding σ)

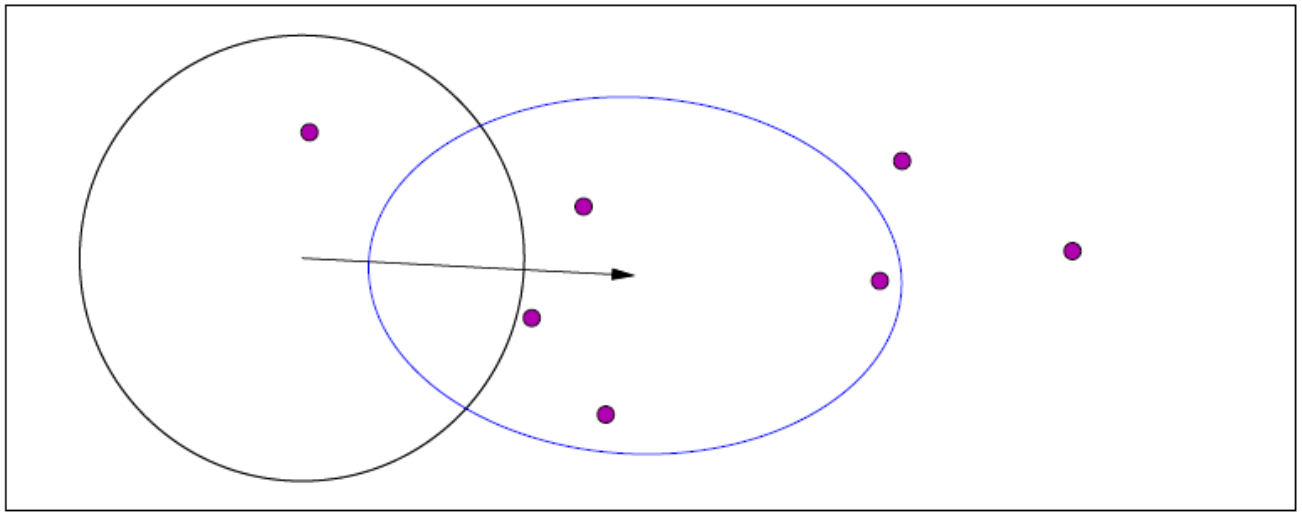
from [Auger, p. 41]

Rank-One Update of Covariance Matrix

Covariance Matrix Adaptation

Rank-One Update

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



new distribution (disregarding σ)

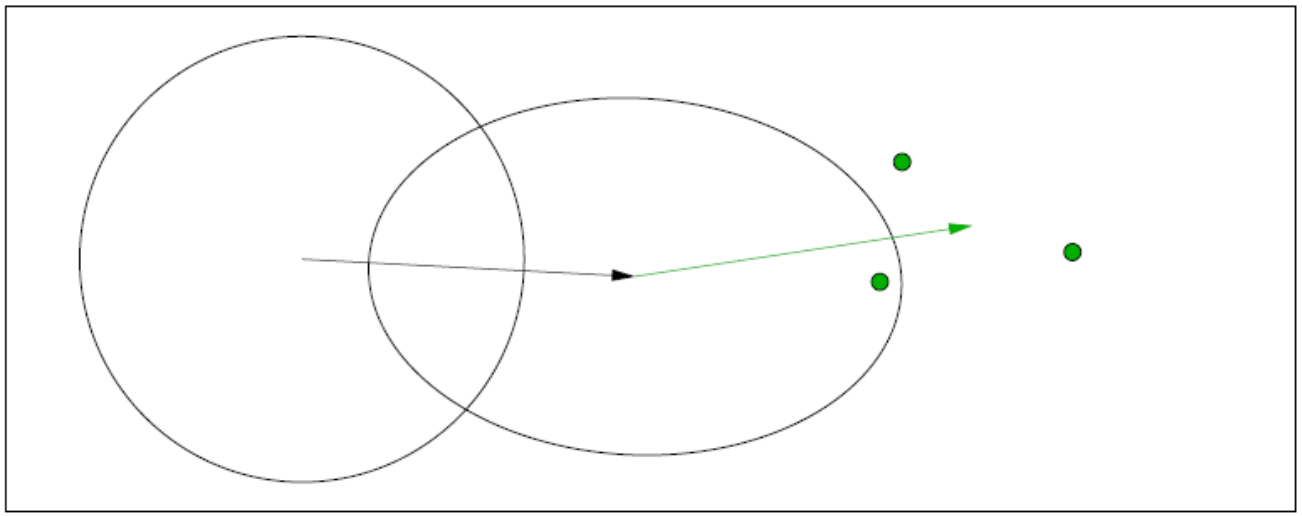
from [Auger, p. 41]

Rank-One Update of Covariance Matrix

Covariance Matrix Adaptation

Rank-One Update

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



movement of the population mean $\boldsymbol{m}$

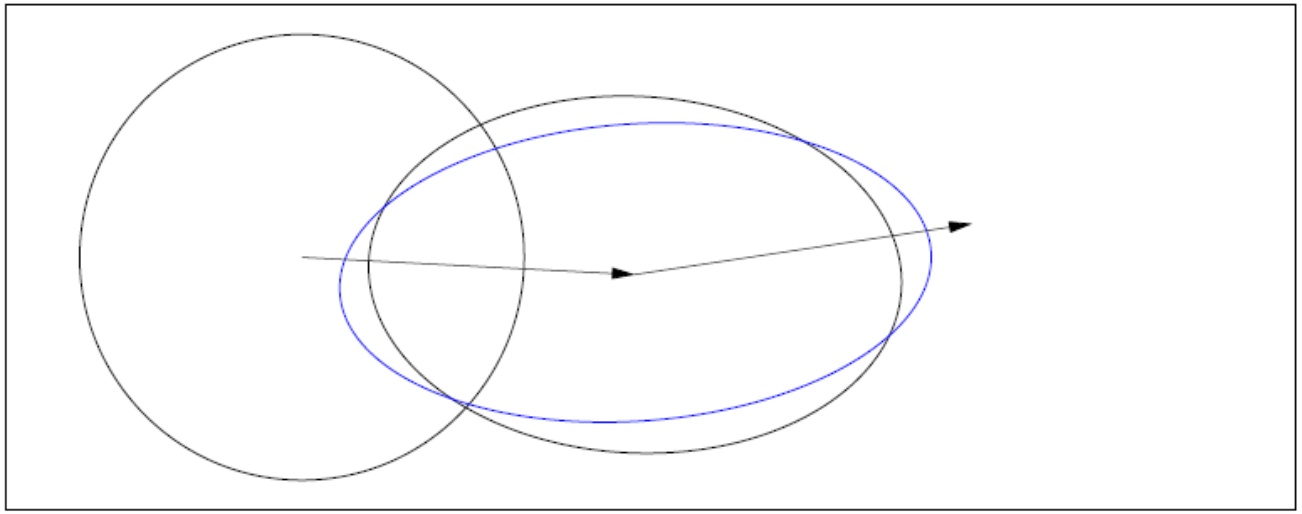
from [Auger, p. 41]

Rank-One Update of Covariance Matrix

Covariance Matrix Adaptation

Rank-One Update

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



mixture of distribution $\mathbf{C}$ and step $\boldsymbol{y}_w$,
$\mathbf{C} \leftarrow 0.8 \times \mathbf{C} + 0.2 \times \boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$

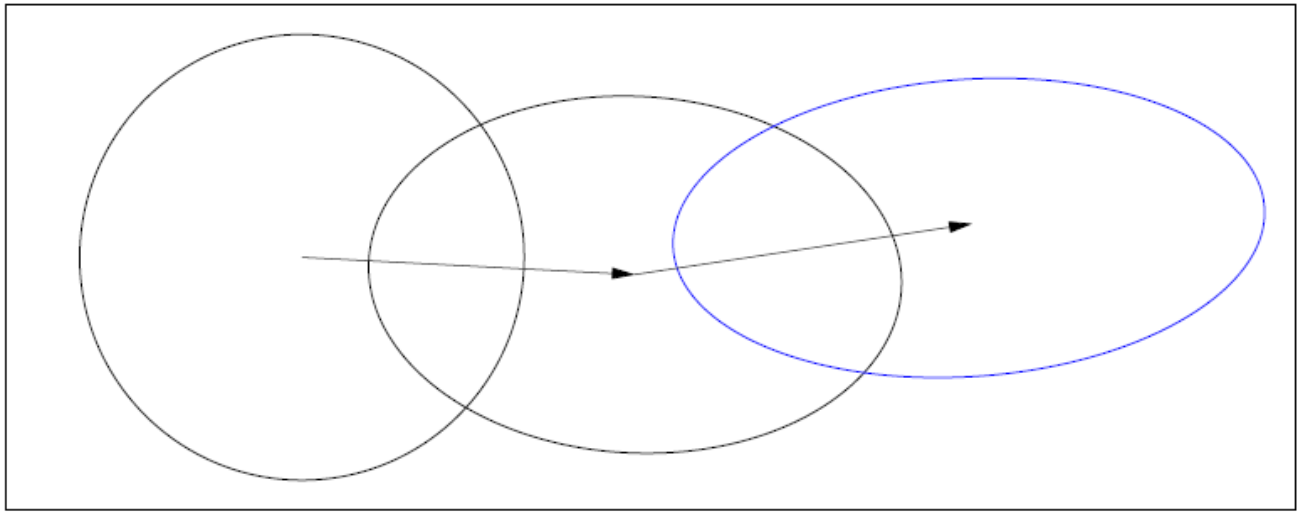
from [Auger, p. 41]

Rank-One Update of Covariance Matrix

Covariance Matrix Adaptation

Rank-One Update

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



new distribution,

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

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

from [Auger, p. 41]

Rank-One Update of Covariance Matrix

Covariance Matrix Adaptation

Rank-One Update

Initialize $\boldsymbol{m} \in \mathbb{R}^n$, and $\mathbf{C} = \mathbf{I}$, set $\sigma = 1$, learning rate $c_{\text{cov}} \approx 2/n^2$
While not terminate

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

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

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

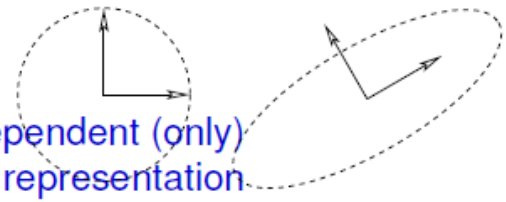
from [Auger, p. 42]

Rank-One Update: Summary

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

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 $\boldsymbol{y}_w$, 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
  components are independent (only) in the new representation



- learns a new (Mahalanobis) metric
  variable metric method
- approximates the inverse Hessian on quadratic functions
  transformation into the sphere function
- for $\mu = 1$: conducts a natural gradient ascent on the distribution $\mathcal{N}$
  entirely independent of the given coordinate system

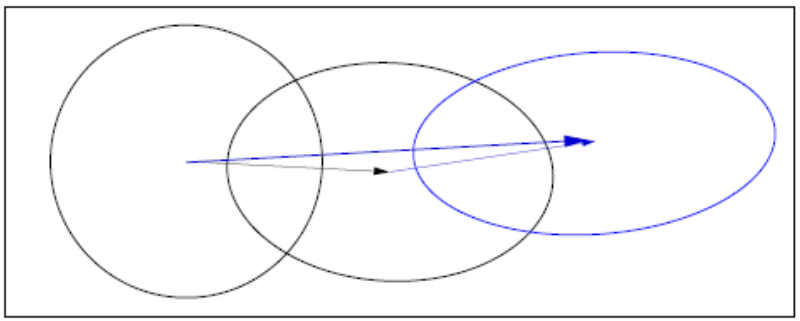
from [Hansen, p. 71]

Cumulation

The Evolution Path

Evolution Path

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



An exponentially weighted sum of steps y_w is used

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

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.

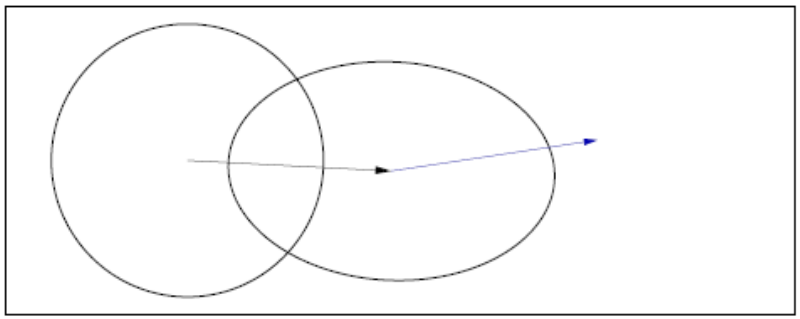
from [Auger, p. 44]

Utilizing the Evolution Path

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 lost.



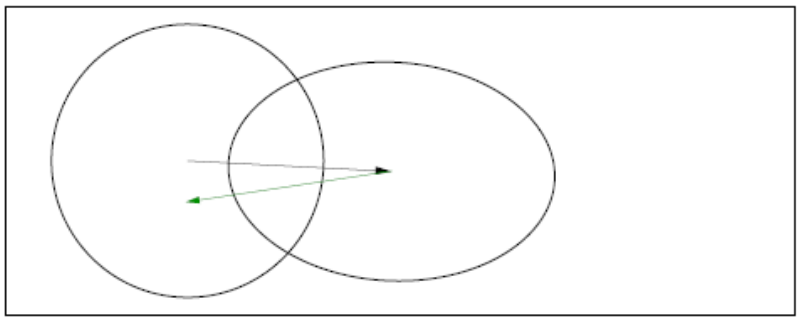
from [Auger, p. 45]

Utilizing the Evolution Path

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 lost.



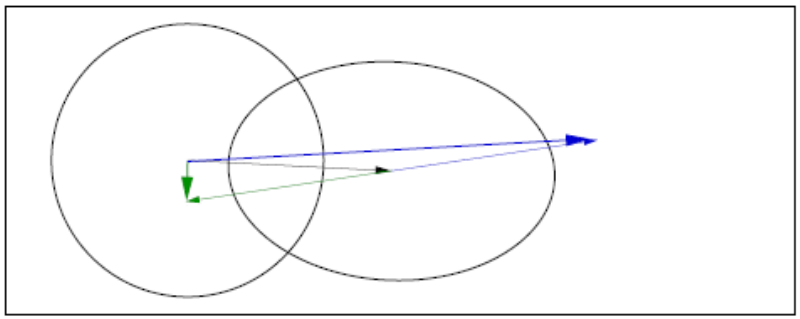
from [Auger, p. 45

Utilizing the Evolution Path

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 lost.



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

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

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

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

from [Auger, p. 45]

Rank-μ Update

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

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.

from [Auger, p. 47]

Rank-μ Update

$$\begin{array}{lcllcl} x_i & = & m + \sigma y_i, & y_i & \sim & \mathcal{N}_i(0, C)\,, \\ m & \leftarrow & m + \sigma y_w & y_w & = & \sum_{i=1}^{\mu} w_i\, y_{i:\lambda} \end{array}$$

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

$$C_\mu = \sum_{i=1}^{\mu} w_i\, y_{i:\lambda} y_{i:\lambda}^{\mathrm{T}}$$

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

from [Auger, p. 47]

Rank-μ Update

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

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

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

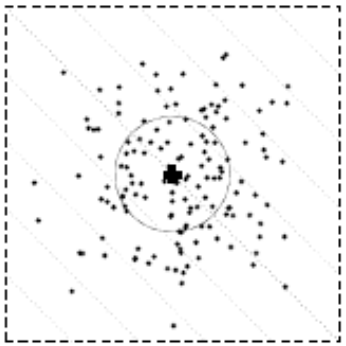
computes a weighted mean of the outer products of the best μ steps and has rank $\min(\mu, n)$ with probability one. The rank-μ update then reads

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

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

from [Auger, p. 47]

Illustration of Rank-μ Update

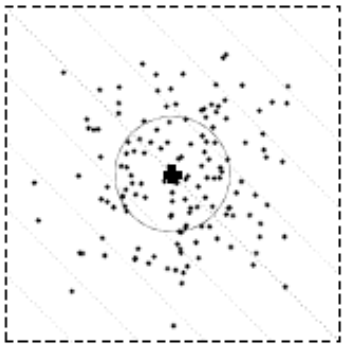


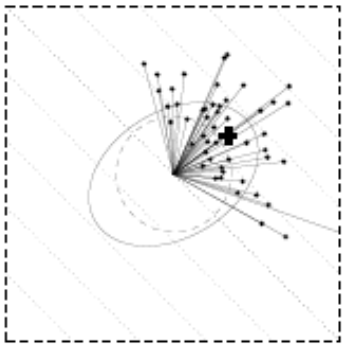
$$x_i = m + \sigma\, y_i, \quad y_i \sim \mathcal{N}(0, C)$$

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

from [Auger, p. 48]

Illustration of Rank-μ Update





$$x_i = m + \sigma y_i, \quad y_i \sim \mathcal{N}(0, C) \qquad C_\mu = \frac{1}{\mu} \sum y_{i:\lambda} y_{i:\lambda}^T$$

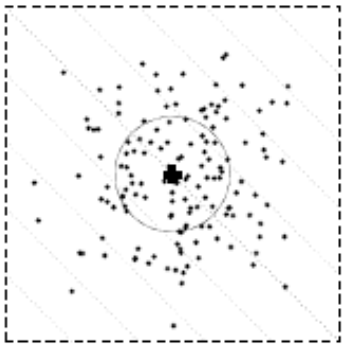
$$C \leftarrow (1-1) \times C + 1 \times C_\mu$$

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

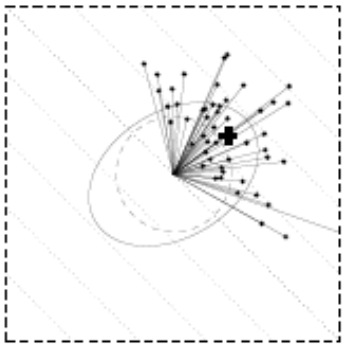
from [Auger, p. 48]

Illustration of Rank-μ Update



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

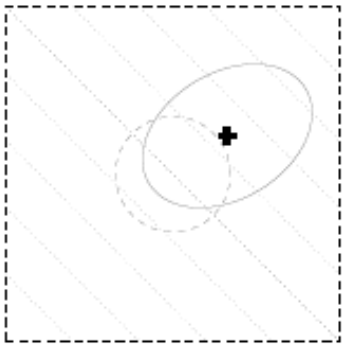
sampling of $\lambda = 150$ solutions where $\mathbf{C} = \mathbf{I}$ and $\sigma = 1$



$$\mathbf{C}_\mu = \frac{1}{\mu} \sum y_{i:\lambda} y_{i:\lambda}^{\mathrm{T}}$$

$$\mathbf{C} \leftarrow (1-1) \times \mathbf{C} + 1 \times \mathbf{C}_\mu$$

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



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

new distribution

from [Auger, p. 48]

Rank-μ Update: Summary

The rank-μ update

- increases the possible learning rate for large populations

  "large" when $\lambda \geq 3n + 10$

- is the primary mechanism whenever a large population size is used
- can be easily combined with rank-one update

The CMA-ES

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

Initialize: $\mathbf{C} = \mathbf{I}$, and $p_\mathbf{c} = \mathbf{0}$,

Set: $c_\mathbf{c} \approx 4/n$, $c_\sigma \approx 4/n$, $c_1 \approx$

and $w_{i=1\ldots\lambda}$ such that $\mu_w = \frac{\mu_w}{n}$,

Promised:
Understand the main principles of this state-of-the-art algorithm.

While not terminate

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

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

$$p_\mathbf{c} \leftarrow (1 - c_\mathbf{c})\, p_\mathbf{c} + \mathbb{1}_{\{\|p_\sigma\| < 1.5\sqrt{n}\}} \sqrt{1 - (1 - c_\mathbf{c})^2}\, \sqrt{\mu_w}\, \boldsymbol{y}_w \qquad \text{cumulation for } \mathbf{C}$$

$$p_\sigma \leftarrow (1 - c_\sigma)\, p_\sigma + \sqrt{1 - (1 - c_\sigma)^2}\, \sqrt{\mu_w}\, \mathbf{C}^{-\frac{1}{2}} \boldsymbol{y}_w \qquad \text{cumulation for } \sigma$$

$$\mathbf{C} \leftarrow (1 - c_1 - c_\mu)\, \mathbf{C} + c_1\, p_\mathbf{c} p_\mathbf{c}^{\mathrm{T}} + c_\mu \textstyle\sum_{i=1}^{\mu} w_i\, \boldsymbol{y}_{i:\lambda} \boldsymbol{y}_{i:\lambda}^{\mathrm{T}} \qquad \text{update } \mathbf{C}$$

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

Not covered on this slide: termination, restarts, useful output, boundaries and encoding

The CMA-ES

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

Initialize: $\mathbf{C} = \mathbf{I}$, and $\boldsymbol{p}_\mathbf{c} = \mathbf{0}$, $\boldsymbol{p}_\sigma = \mathbf{0}$,

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

While not terminate

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

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

$$\boldsymbol{p}_\mathbf{c} \leftarrow (1 - c_\mathbf{c})\,\boldsymbol{p}_\mathbf{c} + \mathbb{1}_{\{\|\boldsymbol{p}_\sigma\| < 1.5\sqrt{n}\}} \sqrt{1 - (1 - c_\mathbf{c})^2}\sqrt{\mu_w}\,\boldsymbol{y}_w \qquad \text{cumulation for } \mathbf{C}$$

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

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

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

Not covered on this slide: termination, restarts, useful output, boundaries and encoding

© Dimo Brockhoff, Inria

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, $\mathbf{C}$, and σ
  - $w_{i=1,\ldots,\mu}$, recombination weights
- related to $\mathbf{C}$-update
  - $c_\mathbf{c}$, decay rate for the evolution path
  - c_1, learning rate for rank-one update of $\mathbf{C}$
  - c_μ, learning rate for rank-μ update of $\mathbf{C}$
- related to σ-update
  - c_σ, decay 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.
Only(?) the population size λ (and the initial σ) might be reasonably varied in a wide range, *depending on the objective function*
Useful: restarts with increasing population size (IPOP)

from [Hansen, p. 90]

Experimental Considerations

Experimentum Crucis (0)

What did we want to achieve?

- reduce any convex-quadratic function

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

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

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

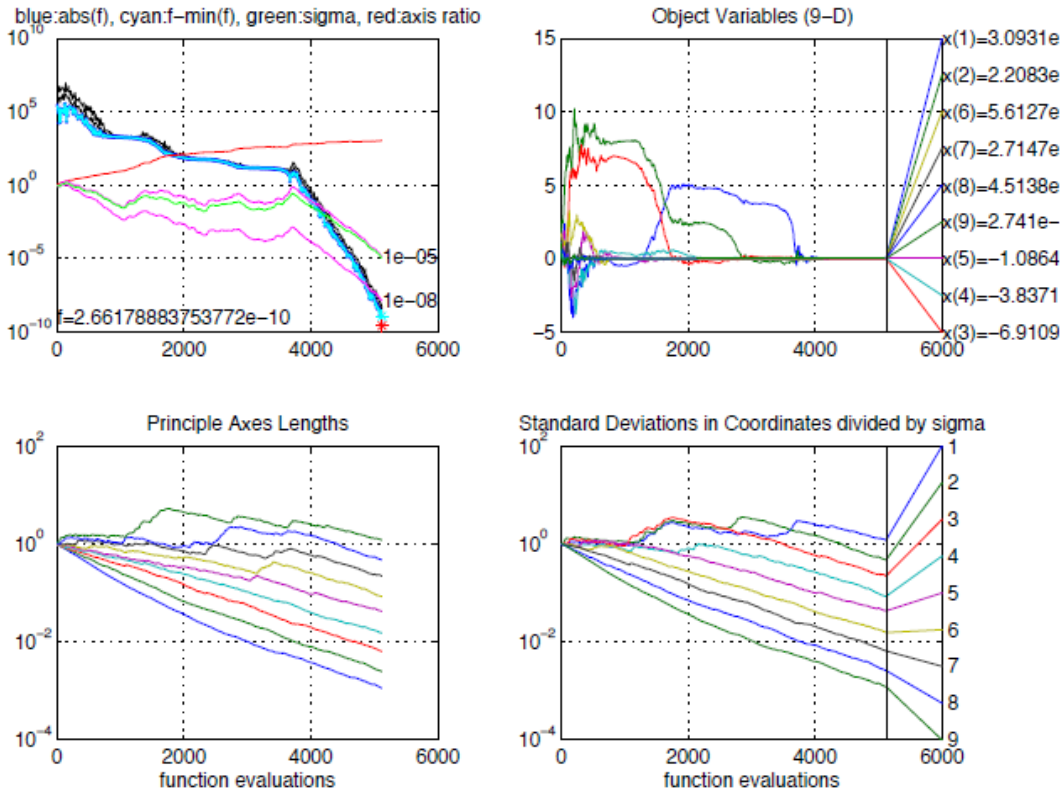
$$\mathbf{C} \propto \boldsymbol{H}^{-1}$$

in a stochastic sense

from [Hansen, p. 91]

Experimentum Crucis (1)

f convex quadratic, separable

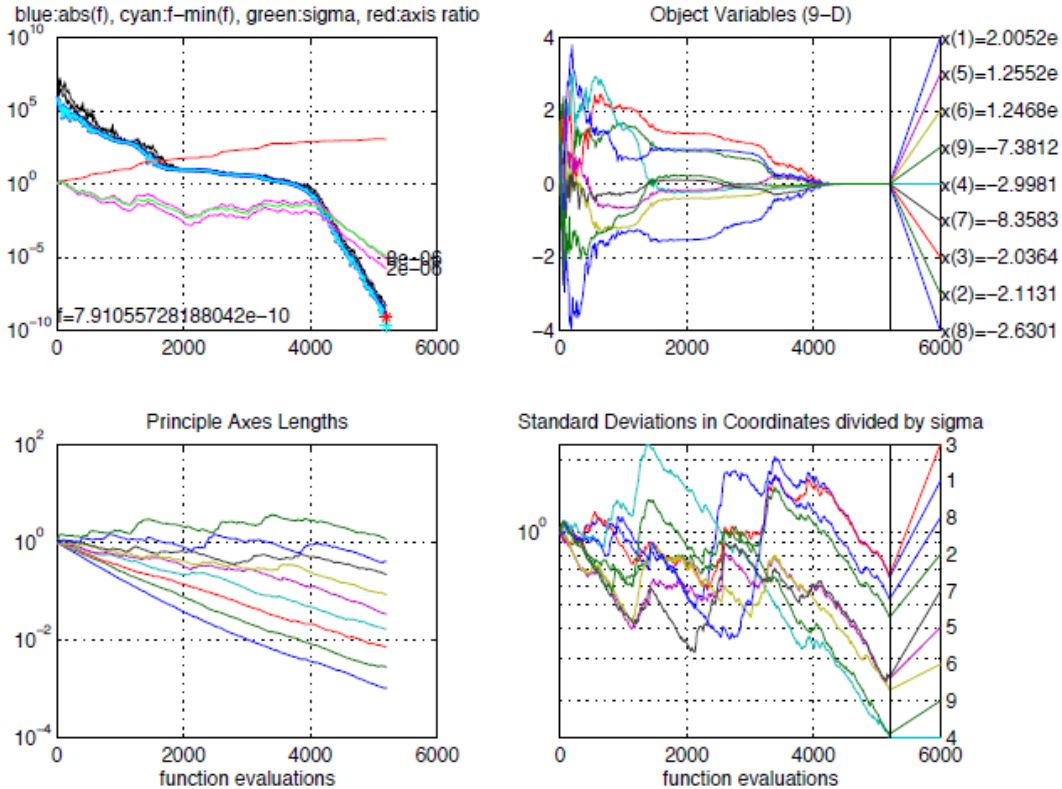


$$f(\boldsymbol{x}) = \sum_{i=1}^{n} 10^{\alpha \frac{i-1}{n-1}} x_i^2, \alpha = 6$$

from [Hansen, p. 92]

Experimentum Crucis (2)

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



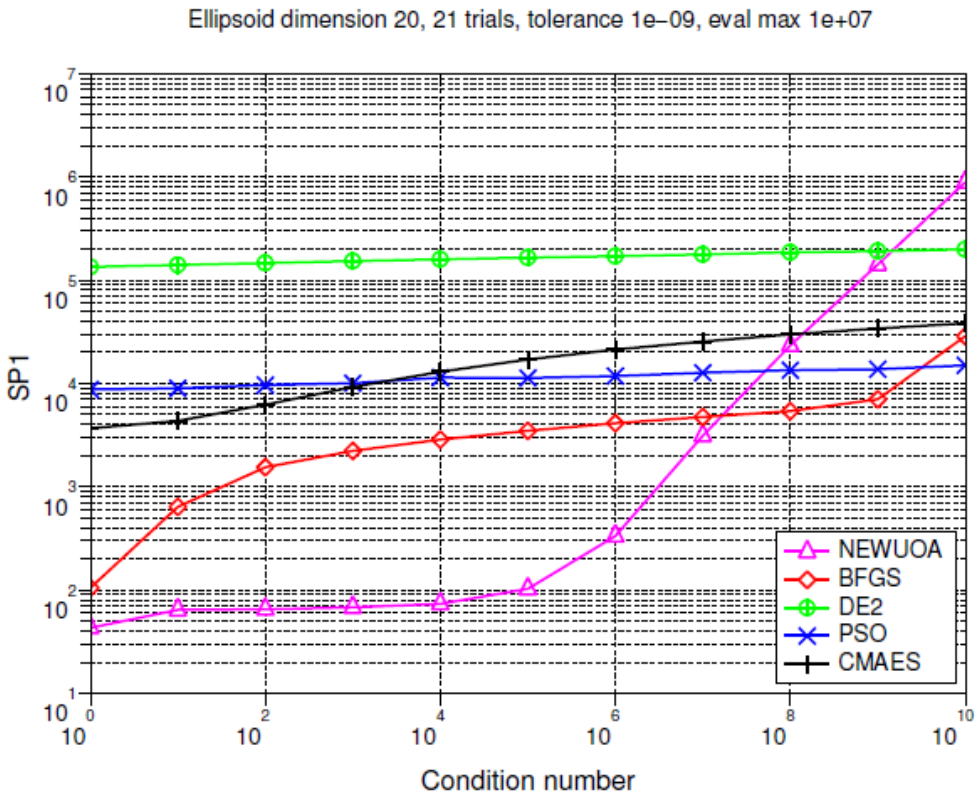
$\mathbf{C} \propto \boldsymbol{H}^{-1}$ for all $g, \mathbf{H}$

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

from [Hansen, p. 93]

Comparison to BFGS, NEWUOA, PSO and DE

f convex quadratic, separable with varying condition number α



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

$f(\boldsymbol{x}) = g(\boldsymbol{x}^{\mathrm{T}}\mathbf{H}\boldsymbol{x})$ with

$\boldsymbol{H}$ diagonal

g identity (for BFGS and NEWUOA)

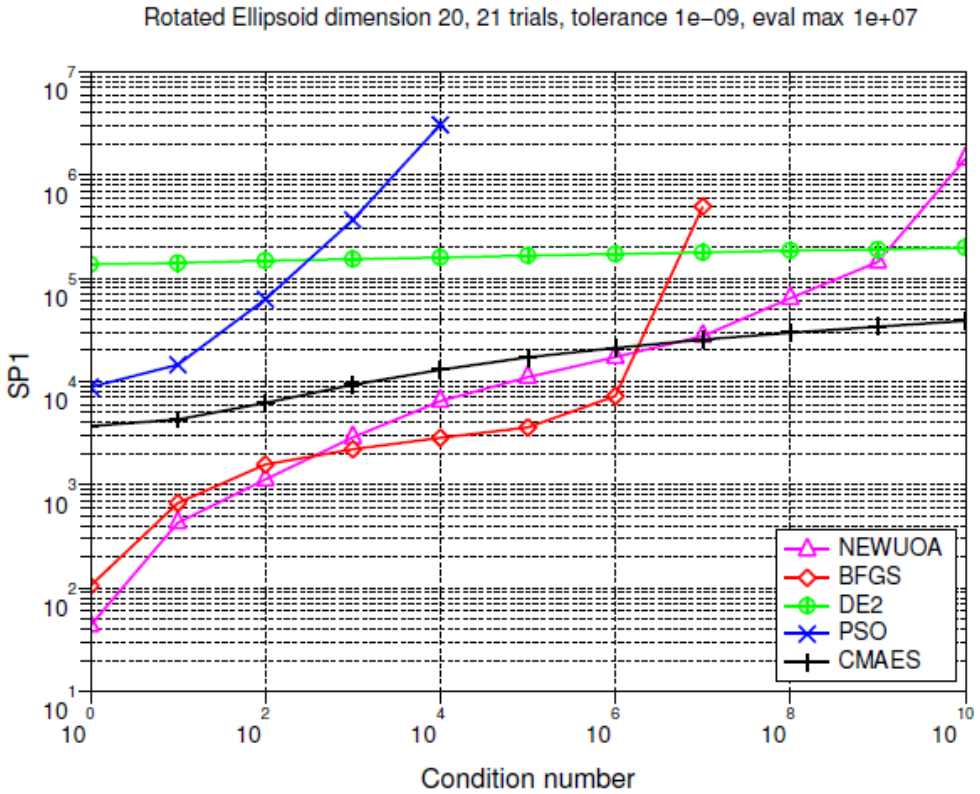
g any order-preserving = strictly increasing function (for all other)

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

[14] Auger et.al. (2009): Experimental comparisons of derivative free optimization algorithms, SEA

Comparison to BFGS, NEWUOA, PSO and DE

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



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

$f(\boldsymbol{x}) = g(\boldsymbol{x}^{\mathrm{T}}\mathbf{H}\boldsymbol{x})$ with

$\boldsymbol{H}$ full

g identity (for BFGS and NEWUOA)

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

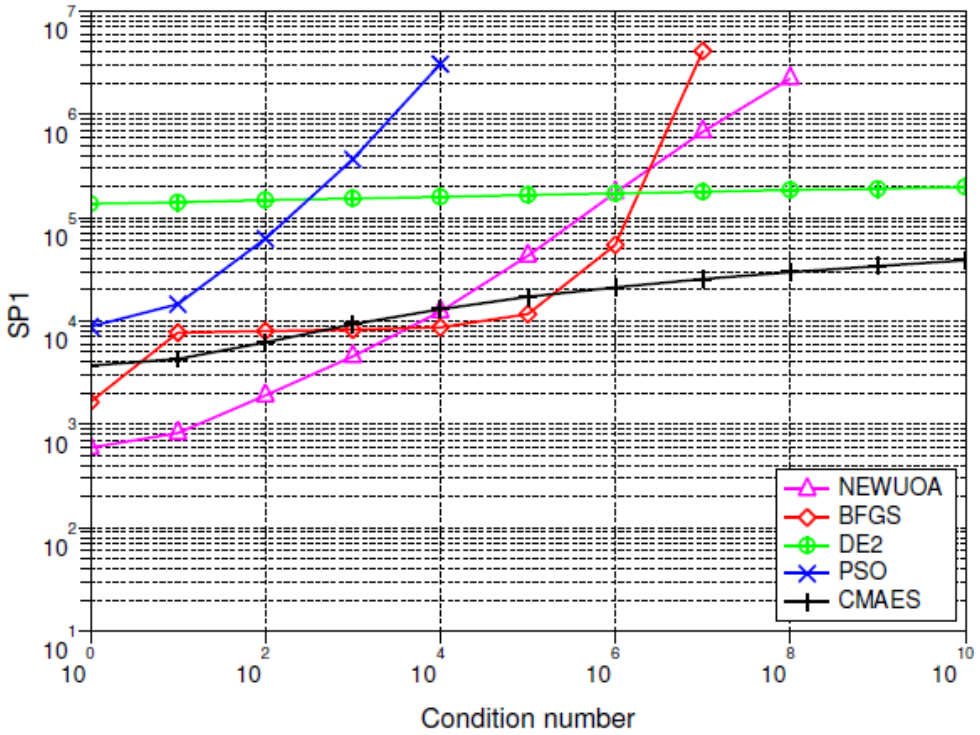
SP1 = average number of objective function evaluations[15] to reach the target function value of $g^{-1}(10^{-9})$

[15] Auger et.al. (2009): Experimental comparisons of derivative free optimization algorithms, SEA

Comparison to BFGS, NEWUOA, PSO and DE

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

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



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

$f(\boldsymbol{x}) = g(\boldsymbol{x}^{\mathrm{T}}\mathbf{H}\boldsymbol{x})$ with

$\boldsymbol{H}$ full

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

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

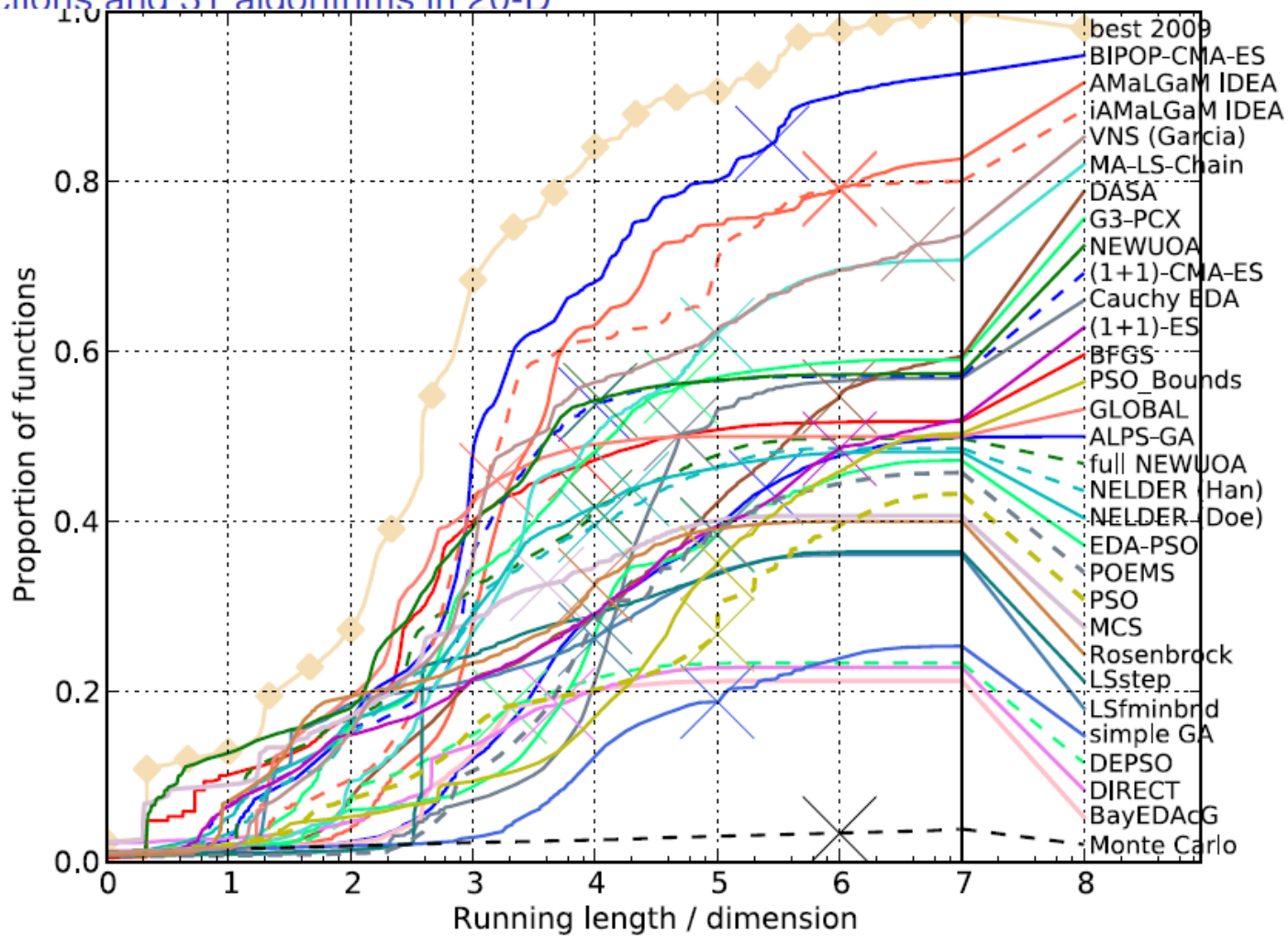
SP1 = average number of objective function evaluations[16] to reach the target function value of $g^{-1}(10^{-9})$

[16] Auger et.al. (2009): Experimental comparisons of derivative free optimization algorithms, SEA

Performance on BBOB Testbed: Data Profile

Comparison during BBOB at GECCO 2009

24 functions and 31 algorithms in 20-D



Summary CMA-ES I

Main Characteristics of (CMA) Evolution Strategies

1. Multivariate normal distribution to generate new search points
   follows the maximum entropy principle
2. Rank-based selection
   implies invariance, same performance on $g(f(\boldsymbol{x}))$ for any increasing g
   more invariance properties are featured
3. Step-size control facilitates fast (log-linear) convergence and possibly linear scaling with the dimension
   in CMA-ES based on an evolution path (a non-local trajectory)
4. *Covariance matrix adaptation (CMA)* increases the likelihood of previously successful steps and can improve performance by orders of magnitude

   the update follows the natural gradient
   $\mathbf{C} \propto \boldsymbol{H}^{-1} \iff$ adapts a variable metric
   $\iff$ new (rotated) problem representation
   $\implies f : \boldsymbol{x} \mapsto g(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{x})$ reduces to $\boldsymbol{x} \mapsto \boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}$

Summary CMA-ES II

Limitations

of CMA Evolution Strategies

- internal CPU-time: $10^{-8}n^2$ seconds per function evaluation on a 2GHz PC, tweaks are available
  1 000 000 f-evaluations in 100-D take 100 seconds *internal* CPU-time
- better methods are presumably available in case of
  - partly separable problems
  - specific problems, for example with cheap gradients
    specific methods
  - small dimension ($n \ll 10$)
    for example Nelder-Mead
  - small running times (number of f-evaluations $< 100n$)
    model-based methods

Conclusions

I hope it became clear...

...that CMA-ES samples according to multivariate normal distributions
...how CMA-ES updates its mean, stepsize, and covariance matrix
...and what are the invariance properties of CMA-ES

Bayesian Optimization

yet another derivative-free approach (but now with surrogates)

Surrogate-Assisted Optimization Idea

When objective function is expensive:

- replace objective function (sometimes) with a (learned) model of it

Questions:

- Which model?
- How to match the model with true evaluations (i.e. how to learn)?
- When to evaluate the objective function, when the model?

- In this context, most algorithms include various heuristic parts…

A Basic Surrogate-Assisted Algorithm Framework

Initial (space-filling) experimental design:

evaluate n_0 points $x_1, \ldots, x_{n_0}$ in the search space as $f(x_1), \ldots, f(x_{n_0})$ to "get a first idea of the landscape"

While happy do:

- Learn surrogate model $\mathcal{M}$ of the true objective function based on (a subset of) all so-far-evaluated search points
- Find (an approximation of) the optimum of the so-called acquisition function $\mathrm{A}_{\mathcal{M}}(x)$, denoted x_n
- Evaluate x_n on the true objective function f

The Framework with Quadratic Model: NEWUOA

(FULL-)NEWUOA by M. J. D. Powell:

- Initial design ($t = 0$): $(n+1)(n+2)/2$ points around x_0 along coordinate axes
- Surrogate model: quadratic, interpolating the current quadratically many points
- Acquisition function: minimum $x_{t+1} = x_t + d$ of quadratic model within a trust region (with $||d||$ smaller than *adapted* trust region radius Δ)
- Evaluate new point x_{t+1} and replace x_t if better, replace interpolation point, furthest to x_{t+1}, with x_{t+1}
- Many more details to make the algorithm work well in practice
- In particular, often using only $2n + 1$ interpolation points

Powell, M. J. D. (November 2004). The NEWUOA software for unconstrained optimization without derivatives (Report). Department of Applied Mathematics and Theoretical Physics, Cambridge University. DAMTP 2004/NA05.

Powell, M. J. D. (2004). "Least Frobenius norm updating of quadratic models that satisfy interpolation conditions". Mathematical Programming 100: 183–215. doi:10.1007/s10107-003-0490-7.

EGO: Gaussian Processes as Probabilistic Model

Bayesian Optimization (Mockus, 1975):

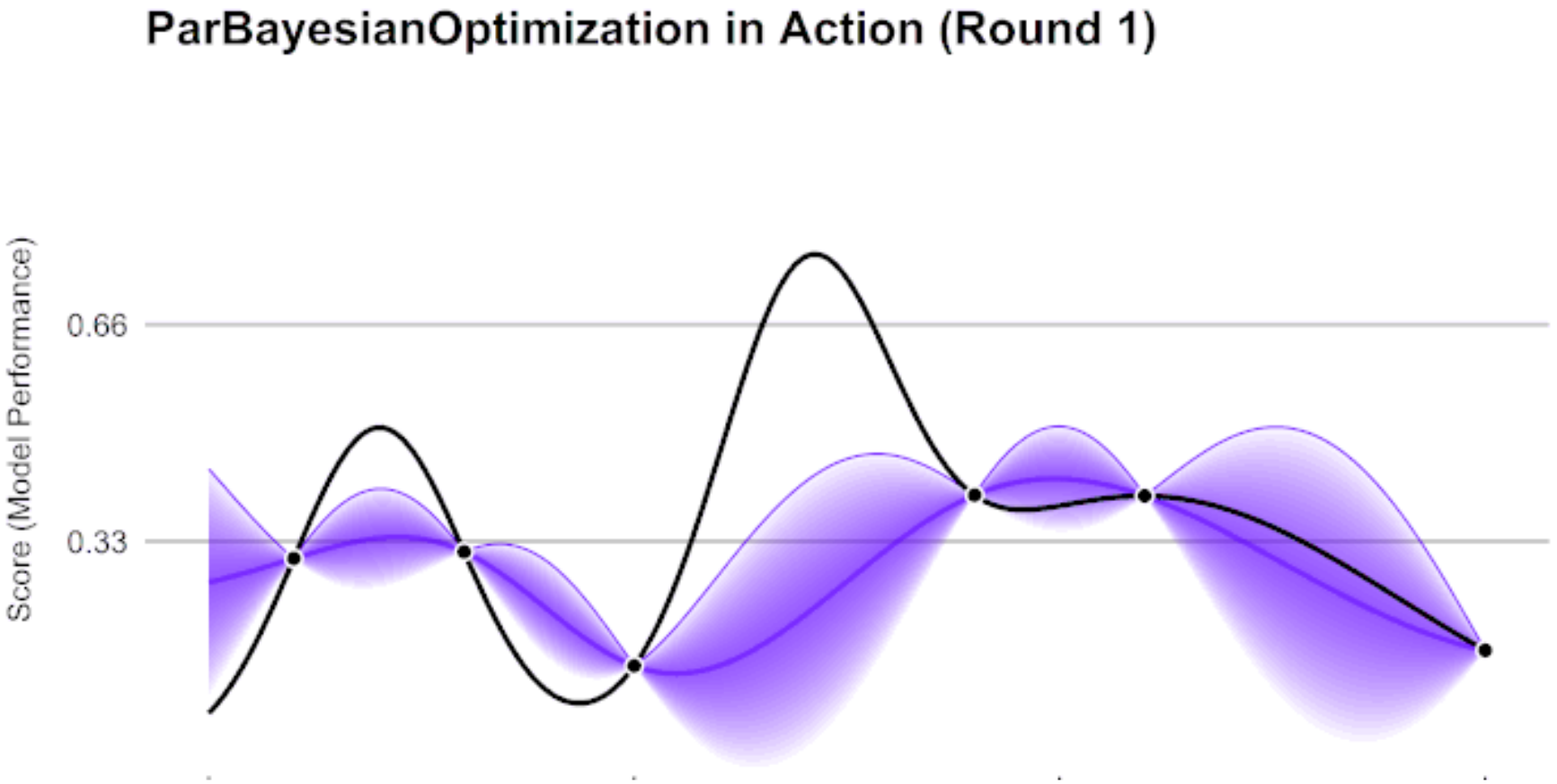
- Model = Gaussian Process
- Gaussian Process aka Kriging: a probabilistic model (see next slide)
- Became popular especially under the name Efficient Global Optimization (EGO, Jones et al., 1998)
- Acquisition function often a trade-off between exploration and exploitation, most common is Expected Improvement

for details, see also https://arxiv.org/pdf/1807.02811.pdf

Mockus, J. (1975). On Bayesian methods for seeking the extremum. In Optimization Techniques IFIP Technical Conference, pages 400–404. Springer

Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. Journal of Global Optimization, 13(4):455–492.

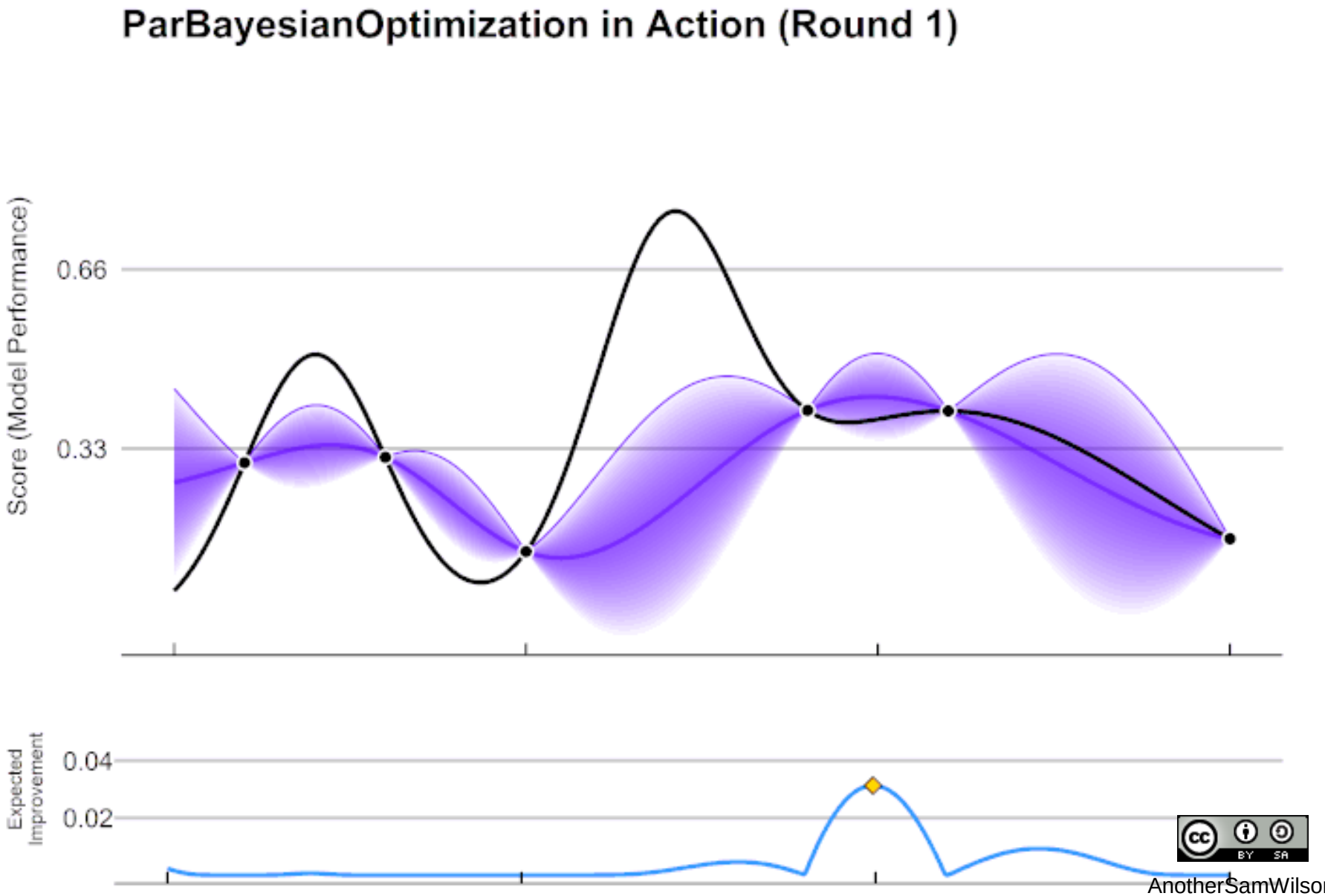
Gaussian Processes



AnotherSamWilson

© Din

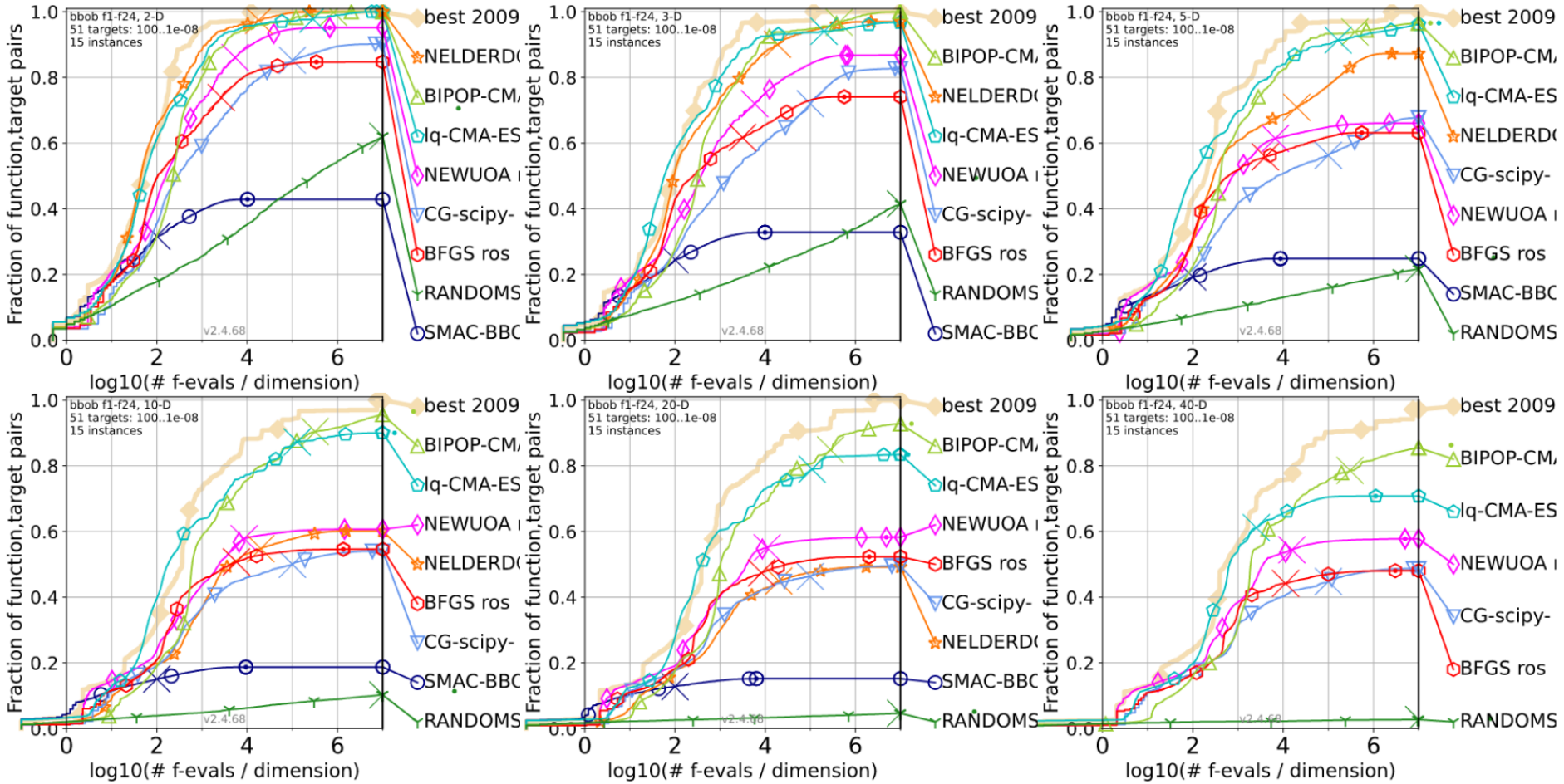
Gaussian Processes



AnotherSamWilson

© Din

Preview: Comparison of Algorithms



addendum:

Stochastic Gradient Descent

Reminder Gradient Descent

General principle

1. choose an initial point $\boldsymbol{x}_0$, set $t = 0$
2. while not happy
   - choose the negative gradient as descent direction: $-\nabla f$
   - line search:
     - choose a step size $\sigma_t > 0$
     - set $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \sigma_t \nabla f(\boldsymbol{x}_t)$
   - set $t = t + 1$

Disadvantage of Basic Gradient Descent

When optimizing weights of a neural net in deep learning:

- we optimize the loss function $\sum_{i=1}^{m}(y_i - \widehat{y_i})$

  y_i: from training data, $\widehat{y_i}$: neural network output, depending on weight vector $\boldsymbol{w}$
- not only $\boldsymbol{w}$ is of high dimension, but also m is large
- consequence: computing gradient $\nabla f(\boldsymbol{w})$ is costly
- Why? → Have to compute many partial derivatives

Idea:

- Compute only an approximation of $\nabla f(\boldsymbol{w})$ with respect to *some* data
- More generally applicable when $f(x) = \sum_{i=1}^{m} f_i(x)$

Stochastic Gradient Descent (SGD)

Idea:

Compute only an approximation of $\nabla f(\boldsymbol{x})$ with respect to *some subfunctions* of $f(x) = \sum_{i=1}^{m} f_i(\boldsymbol{x})$

Instead of $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \sigma_t \nabla f(\boldsymbol{x}_t)$, we update the current iterate only with respect to one (random) subfunction f_i:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \sigma_t \nabla f_i(\boldsymbol{x}_t)$$

Batches, Mini-Batches and Epochs

- To make sure, all subfunctions are used equally, we typically choose a (random) permutation π and use each subfunction once in this order before to redo this step
- Such step is called an *epoch*

- Using single subfunctions is fast, but also highly stochastic

consequence: practical convergence is slower

- Hence, we shall use batches of subfunctions at a time:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \sigma_t \nabla f_{\mathcal{B}_j}(\boldsymbol{x}_t)$$

with $f_{\mathcal{B}_j}(\boldsymbol{x}_t) = \sum_{i \in \mathcal{B}_j} f_i(\boldsymbol{x})$ and $\mathcal{B}_j \subset \{1, \dots, m\}$ the corresponding indices in the jth batch

SGD vs. Adam

Stochastic Gradient Descent

Typically applied with a constant step size in the context of Deep Learning

Adam: Adaptive Moment Adaptation

- decaying average of past gradients $g_t(x) = \nabla f_{\mathcal{B}_j}(x_t)$, also called "momentum":

$$m_t(x) = \beta_1 m_{t-1} + (1-\beta_1) g_t(x)$$

- decaying average of past squared gradients

$$v_t(x) = \beta_2 v_{t-1} + (1-\beta_2) g_t^2(x)$$

- to decrease bias towards 0, we use $\widehat{m_t} = \frac{m_t}{1-\beta_1^t}$ and $\widehat{v_t} = \frac{v_t}{1-\beta_2^t}$
- and then $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \frac{\sigma_t}{\sqrt{\widehat{v_t}(x)+\epsilon}} \widehat{m_t}(x)$