Function-Value-Free Second-Order Stochastic Optimization with CMA-ES

Nikolaus Hansen
Inria
Research Centre Saclay – Île-de-France
Université Paris-Saclay, École Polytechnique, CMAP

http://www.inria.fr

http://www.lri.fr/~hansen

…feel free to ask questions…

Any intelligent fool can make things bigger, more complex, and more violent. It takes a touch of genius, and a lot of courage, to move in the opposite direction.

Albert Einstein

Prediction is very difficult, especially about the future.

Niels Bohr

Black-Box Optimization (Search)

Minimize (or maximize) a continuous domain objective (cost, loss, error, fitness) function

$$f : \mathbb{R}^n \to \mathbb{R}, \quad x \mapsto f(x)$$

in a black-box scenario (direct search)

where

- gradients are not available or useful
- problem specific knowledge is used only *within* the black box, e.g. with an appropriate encoding

The search costs are the number of black-box calls (function evaluations)

Landscape of Continuous Search Methods

Gradient-based (Taylor, local)

- Conjugate gradient methods [Fletcher & Reeves 1964]
- Quasi-Newton methods (BFGS) [Broyden et al 1970]

Derivative-free optimization (DFO)

- Trust-region methods (NEWUOA, BOBYQA) [Powell 2006, 2009]
- Simplex downhill [Nelder & Mead 1965]
- Pattern search [Hooke & Jeeves 1961, Audet & Dennis 2006]

Stochastic (randomized) search methods

- Evolutionary algorithms (broader sense, continuous domain)
  - Differential Evolution [Storn & Price 1997]
  - Particle Swarm Optimization [Kennedy & Eberhart 1995]
  - **Evolution Strategies** [Rechenberg 1965, Hansen & Ostermeier 2001]
- Simulated annealing [Kirkpatrick et al 1983]
- Simultaneous perturbation stochastic approximation (SPSA) [Spall 2000]

Covariance Matrix Adaptation Evolution Strategy (CMA-ES)

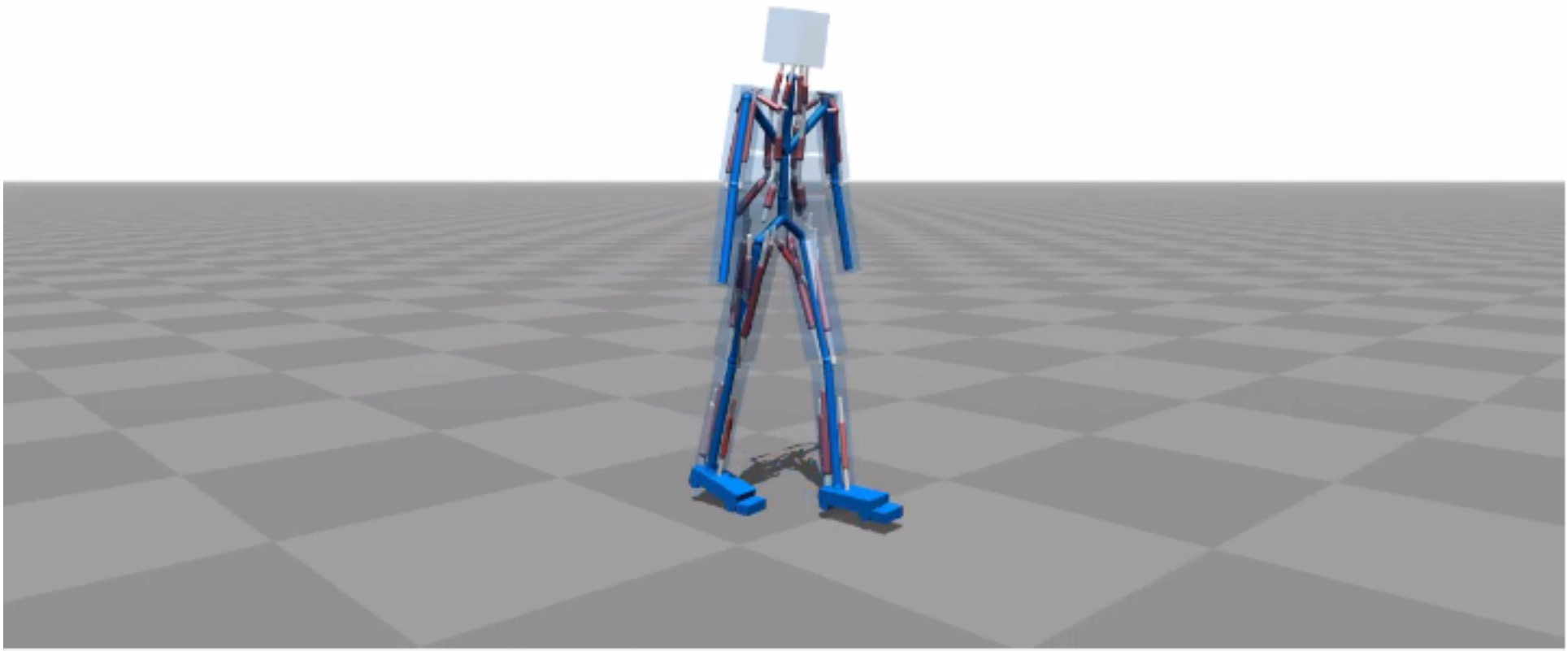
- > 3000 citations to the two main original CMA-ES articles
  $\gg 100$ published applications
- implemented in various software libraries
  - *evolutionary computation:* Open BEAGLE, DEAP, EO, PyEC...
  - *optimization:* Apache Commons Math, NOMADm, OpenOpal
  - *machine learning:* PyBrain, Shark
  - *image processing, robot control & simulation:* PAC, OpenSim
  - *PDE solver:* FreeFem++
  - *water model calibration:* PEST
  - *economics:* AmiBroker, Dynare, parma
- ≈ 2000 monthly page views on Wikipedia and of the source code page
- used by various companies: Alstom, Astrium, Bosch, Honda, Rolls-Royce, Siemens, Storengy, Total,...

Typical Applications

- model/system calibration
  - biological/chemical/physical $\Longrightarrow$ universal constants
  - production process
- optimization of control parameters
  - movements of a robot
  - trajectory of a rocket
  - stability of a gas flame
- shape optimization
  - curve fitting
  - aero- or fluid dynamics design (airfoil, airship)

An Example Application

We present a control system based on 3D muscle actuation



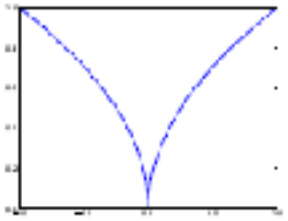
Flexible Muscle-Based Locomotion for Bipedal Creatures

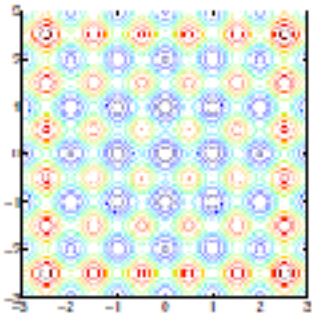
from **John Goatstream** 2 months ago ALL AUDIENCES

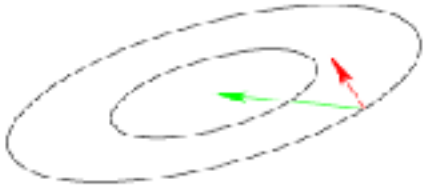
https://youtu.be/pgaEE27nsQw

Difficulties in Black-Box Optimization

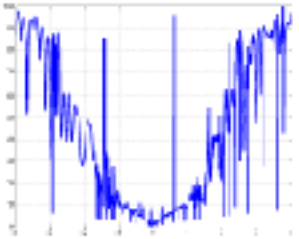
- non-linear, non-quadratic, non-convex
  on linear/quadratic functions better search policies are available
- dimensionality
  (considerably) larger than three
- non-separability
  dependencies between the objective variables
- ill-conditioning
  widely varying sensitivity
- ruggedness
  non-smooth, discontinuous, multimodal, and/or noisy function





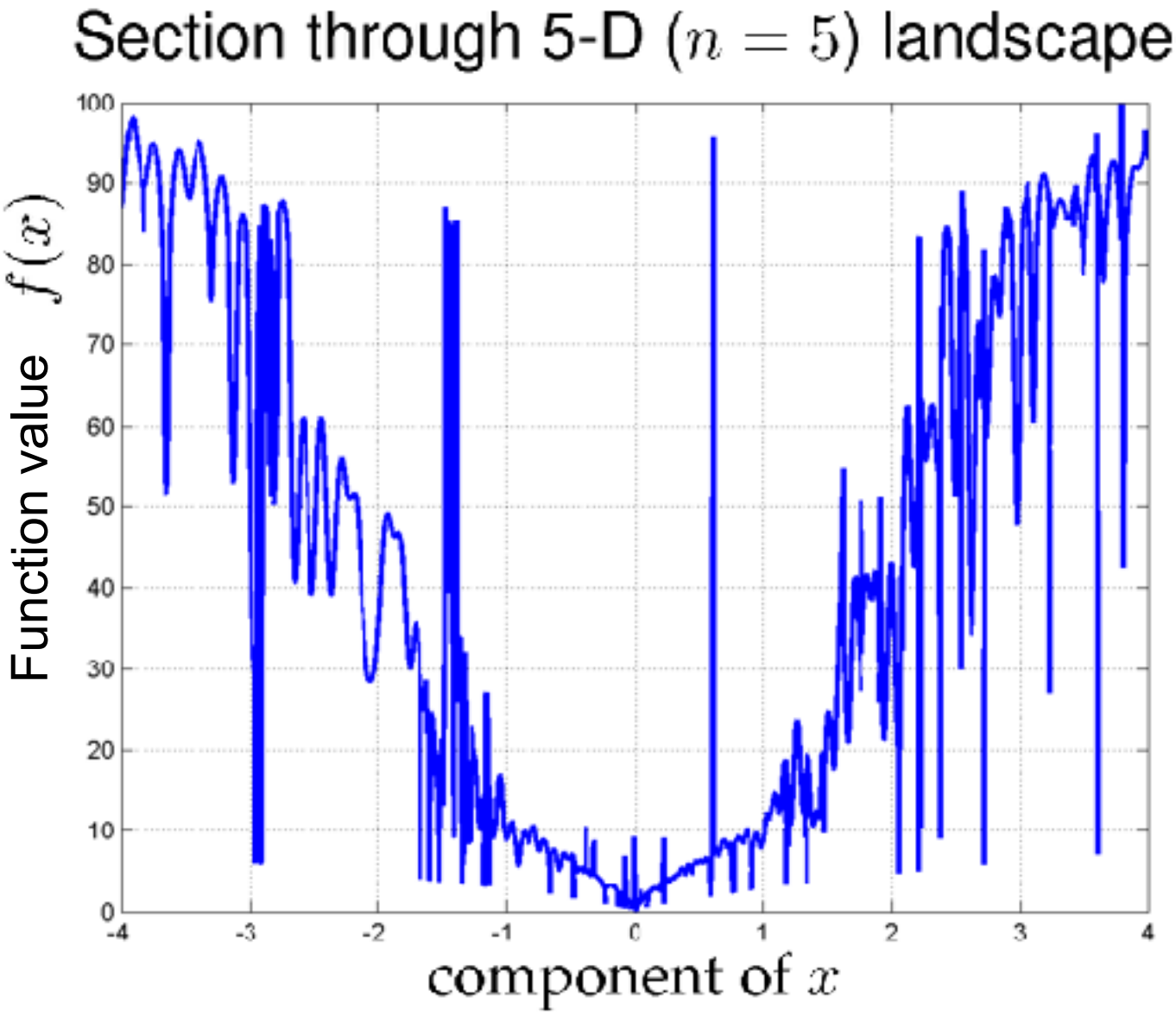


gradient direction Newton direction



in any case, the objective function must be highly regular

Rugged landscape



$$f : \mathbb{R}^n \to \mathbb{R}, \quad x \mapsto f(x)$$

Randomized optimization template

Given: a parametrized distribution $p(.|\theta)$ on $\mathcal{X}$ and $\lambda \in \mathbb{N}$
Initialize: parameter vector θ
While not *happy*

1. Sample $p(x|\theta) \to x_1, \ldots, x_\lambda \in \mathcal{X}$
2. Evaluate $x_1, \ldots, x_\lambda$ on $f \longrightarrow f(x_1), \ldots, f(x_\lambda)$
3. Update parameters

$$\theta \leftarrow \text{Update}(\theta, x_1, \ldots, x_\lambda, f(x_1), \ldots, f(x_\lambda))$$

Return, e.g., the expectation or ML value of $p(.|\theta)$ as given in θ

Randomized optimization template

Given: a parametrized distribution $p(.|\theta)$ on $\mathcal{X}$ and $\lambda \in \mathbb{N}$
Initialize: parameter vector θ
While not *happy*

1. Sample $p(x|\theta) \to x_1, \ldots, x_\lambda \in \mathcal{X}$
2. Evaluate $x_1, \ldots, x_\lambda$ on $f \longrightarrow f(x_1), \ldots, f(x_\lambda)$
3. Update parameters

$$\theta \leftarrow \text{Update}(\theta, x_1, \ldots, x_\lambda, f(x_1), \ldots, f(x_\lambda))$$

Return, e.g., the expectation or ML value of $p(.|\theta)$ as given in θ

How to choose the parametrized distribution $p(.|\theta)$?
How to update θ (point 3.)?

A new search problem

The template replaces the original search problem

$$\arg\min_x (f(x)) \qquad x \in \mathcal{X}$$

with a new search problem on θ-space,

$$\arg\max_\theta (J(\theta)) \quad \text{where} \quad \boxed{J(\theta) = \mathbb{E}_{x\sim p(.|\theta)}\left(W^f_{\theta_t}(f(x))\right)},$$

$$W^f_{\theta_t}(f(x)) := w(\Pr_{y\sim p(.|\theta_t)}(f(y) \le f(x)))$$

where $W^f_{\theta_t}$ is monotonous *decreasing*.

think of $W(f(x))$ as $-f(x)$ for the time being

Both problems have the same solution (same optimum):

$$P(x|\theta^*) = \delta(x - x^*) \quad \text{for all } W^f_{\theta_t}$$

i.e., $\Pr(x = x^* \,|\, \theta = \theta^*) = 1$.

A new search problem

The template replaces the original search problem

$$\arg\min_x \left(f(x)\right) \qquad x \in \mathcal{X}$$

with a new search problem on θ-space,

$$\arg\max_\theta (J(\theta)) \quad \text{where} \quad \boxed{J(\theta) = \mathbb{E}_{x\sim p(.|\theta)}\left(W^f_{\theta_t}(f(x))\right)},$$

$$W^f_{\theta_t}(f(x)) := w(\Pr_{y\sim p(.|\theta_t)}(f(y) \leq f(x)))$$

where $W^f_{\theta_t}$ is monotonous *decreasing*.

think of $W(f(x))$ as $-f(x)$ for the time being

Both problems have the same solution (same optimum):

$$P(x|\theta^*) = \delta(x - x^*) \quad \text{for all } W^f_{\theta_t}$$

i.e., $\Pr(x = x^* \,|\, \theta = \theta^*) = 1$.

Information Geometric Optimization
A Gradient Method in θ-Space

. Taking a gradient gives an update for θ_t:

$$\frac{\mathrm{d}\theta_t}{\mathrm{d}t} = \overbrace{\widetilde{\nabla}_\theta}^{\text{natural gradient}} J(\theta)\Big|_{\theta=\theta_t} = \widetilde{\nabla}_\theta \, \mathbb{E}\Big(\overbrace{W^f_{\theta_t}(f(x))}^{\text{“f-invariant, adaptive” objective}} \Big)\Big|_{\theta=\theta_t}, \quad \overbrace{x \sim p(.|\theta)}^{\text{target distribution}}$$

$$= \quad \ldots \quad = \mathbb{E}\Big(\underbrace{W^f_{\theta_t}(f(x))}_{\text{preference weight}} \underbrace{\overbrace{\widetilde{\nabla}_\theta}^{\text{natural gradient}} \ln p(x|\theta)|_{\theta=\theta_t}}_{\text{intrinsic direction}} \Big)$$

$$\overset{\text{consistent estimator}}{\approx} \underbrace{\frac{1}{Z(\lambda)} \sum_{k=1}^{\lambda}}_{\text{taking the average}} \underbrace{\big(\lambda/2 - \mathrm{rank}(f(x_k))\big)}_{\text{preference weight}} \underbrace{\widetilde{\nabla}_\theta \overbrace{\ln p(x_k|\theta)}^{\text{log-likelihood}}|_{\theta=\theta_t}}_{\text{intrinsic direction}}, \quad x_k \sim \underbrace{p(.|\theta_t)}_{\text{given distribution}}$$

- gradients are based on a metric (inner product): the *natural* gradient $\widetilde{\nabla}_\theta$ is defined from the *Fisher metric*, as parametrization invariant, as compatible with entropy and with KL-divergence
- works also in *discrete* θ-spaces (maximal f-improvement under minimal entropy change)

Randomized optimization template

Given: a parametrized distribution $p(.|\theta)$ on $\mathcal{X}$ and $\lambda \in \mathbb{N}$

Initialize: parameter vector θ_t

While not *happy*

1. Sample distribution $P(x|\theta_t) \to x_1, \ldots, x_\lambda \in \mathcal{X}$
2. Evaluate samples on $f \to f(x_1), \ldots, f(x_\lambda)$
3. Update parameters

$$\theta_{t+1} = \theta_t + \eta \frac{1}{Z(\lambda)} \sum_{k=1}^{\lambda} \left(\lambda/2 - \mathrm{rank}(f(x_k))\right) \widetilde{\nabla}_\theta \ln p(x_k|\theta) \Big|_{\theta=\theta_t}$$

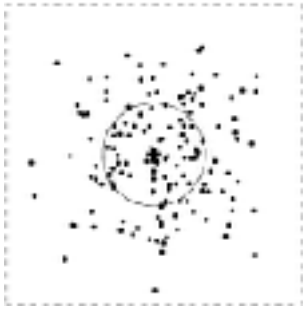
Return, e.g., the expectation or ML value of $p(.|\theta)$ as given in θ_t

How to choose the parametrized distribution $p(.|\theta)$?

How to update θ (point 3.)?

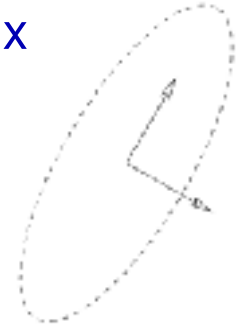
Instantiation in $\mathbb{R}^n$: CMA-ES in a nutshell

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is a second-order method, similar to quasi-Newton methods, however randomized and function-value free



1. $P(x|\theta)$ is a multivariate normal distribution

   θ represents mean and covariance matrix



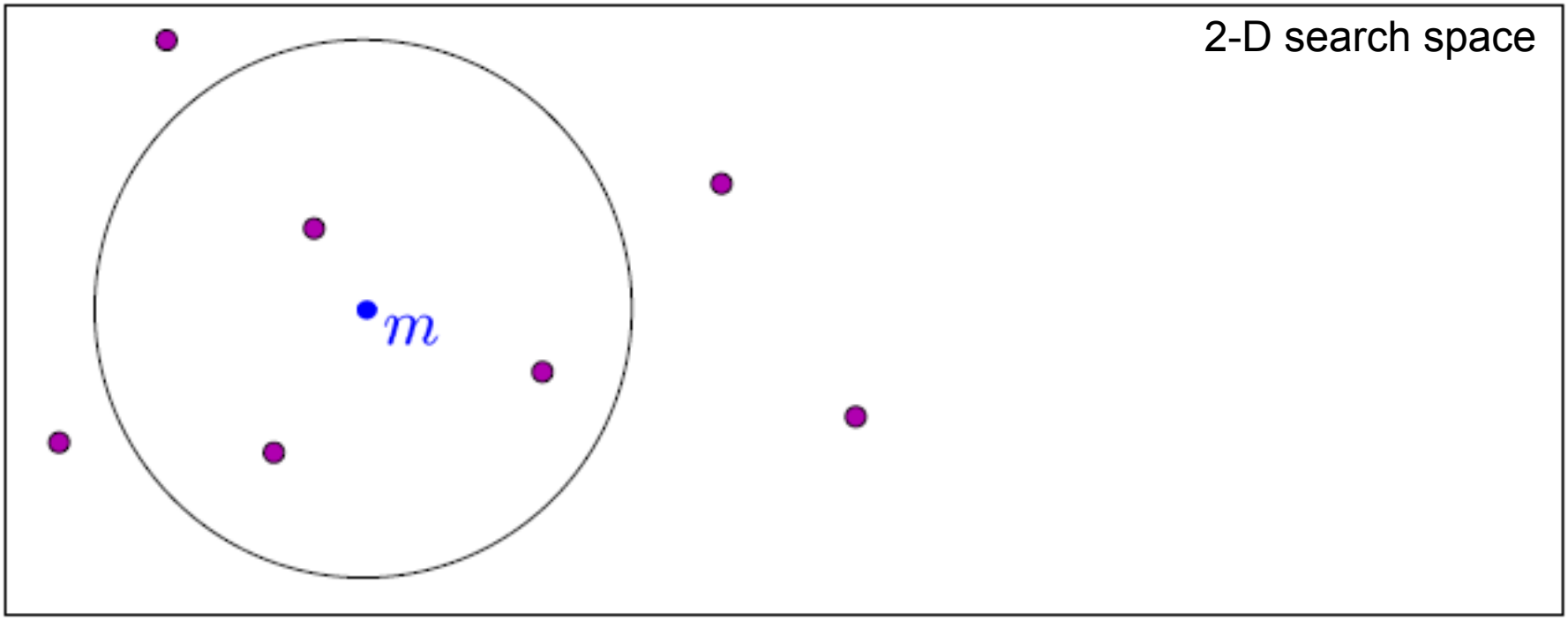
2. the θ-update is a (smoothed) ML-update

   - separate for mean and covariance matrix
   - designed as invariant under linear coordinate system transformations
   - mainly coincides with a natural gradient ascent

3. step-size control thrives for orthogonal steps

   based on non-local information, correlations between steps

Covariance Matrix Adaptation

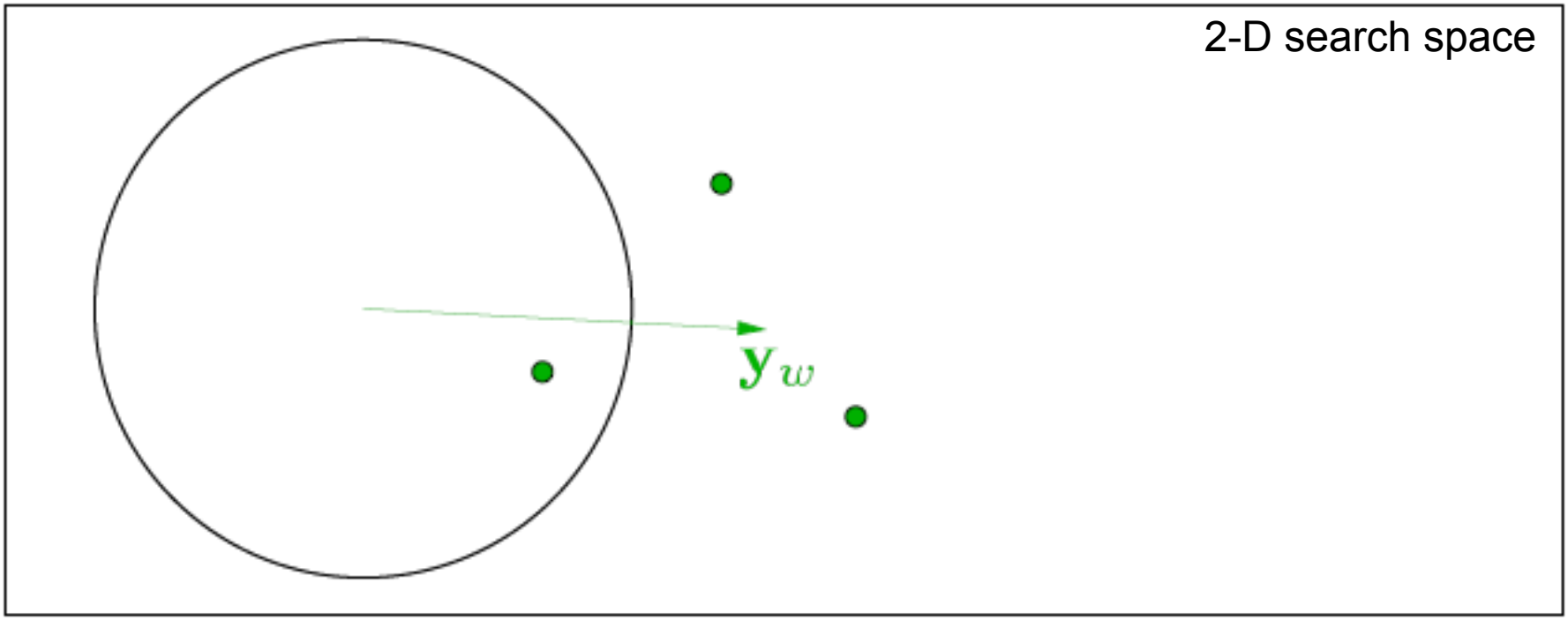


sample from the initial distribution: $\mathcal{N}(m, \mathbf{I})$, $\mathbf{C} = \mathbf{I}$

.

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

Covariance Matrix Adaptation

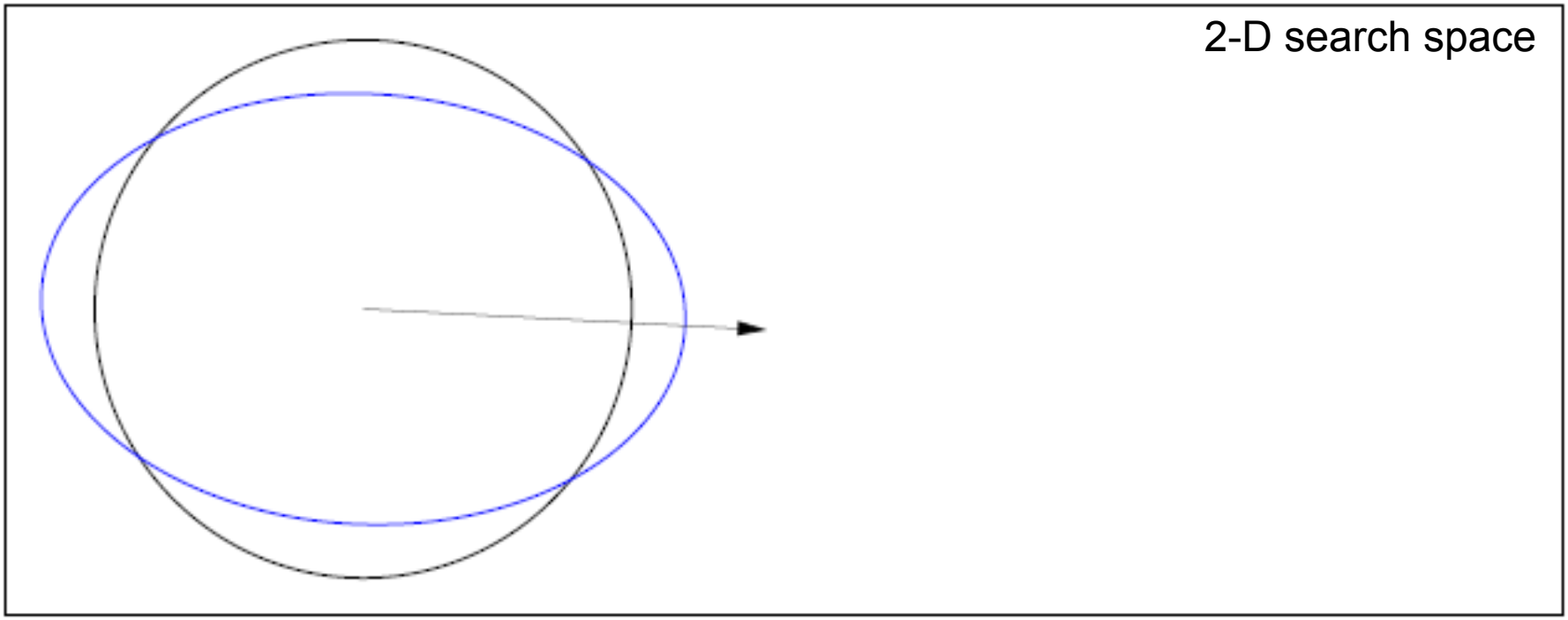


$\mathbf{y}_w$ is the move of the mean m, disregarding σ .

.

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

Covariance Matrix Adaptation



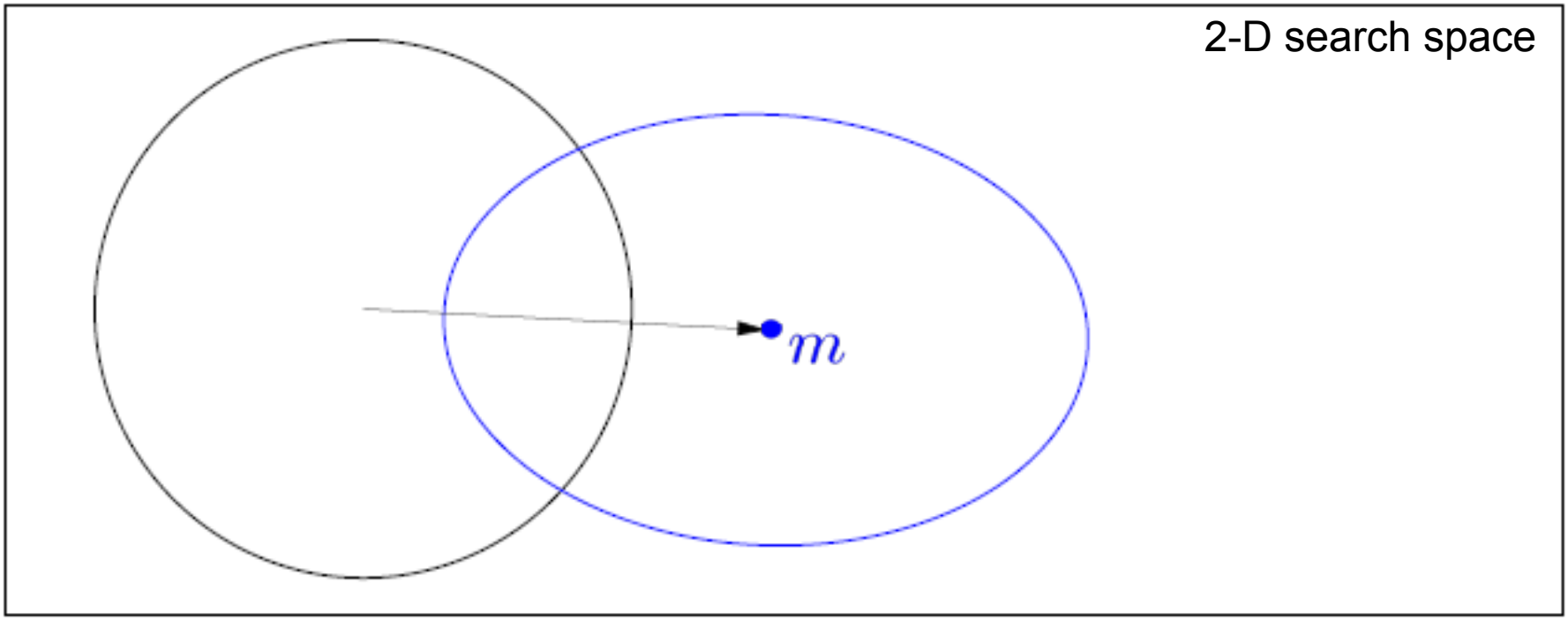
mixture of covariance matrix $\mathbf{C}$ and step $\mathbf{y}_w$

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

.

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

Covariance Matrix Adaptation

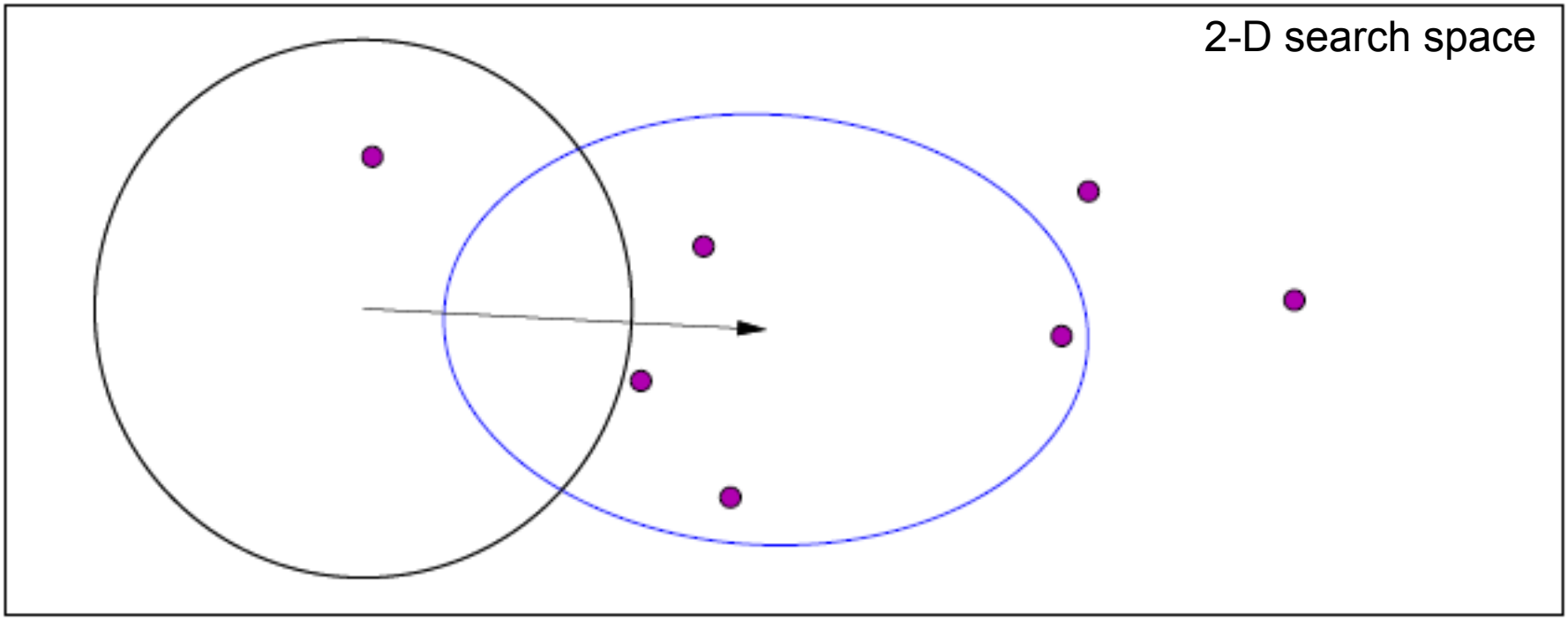


new distribution $\mathcal{N}(m, \mathbf{C})$ (disregarding σ)

.

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

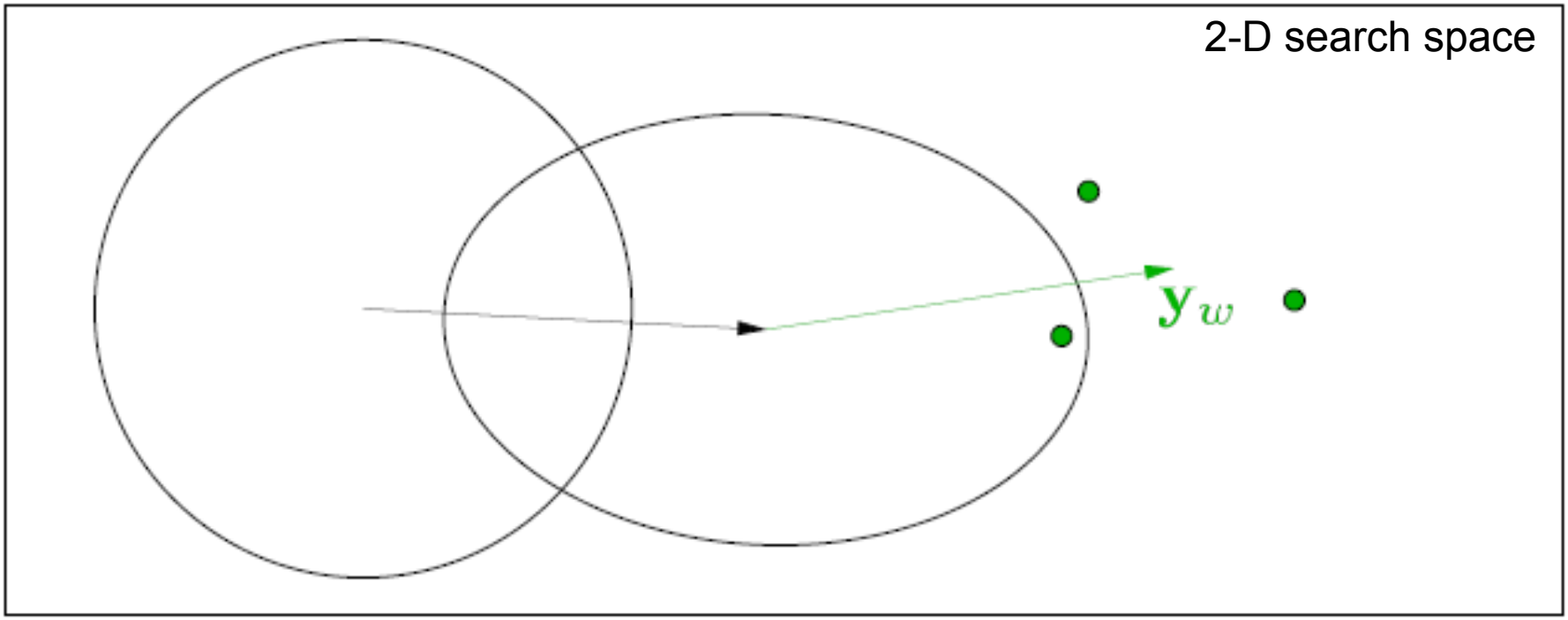
Covariance Matrix Adaptation



another sample $\mathbf{x}_i \sim \mathcal{N}_i(m, \mathbf{C})$

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

Covariance Matrix Adaptation

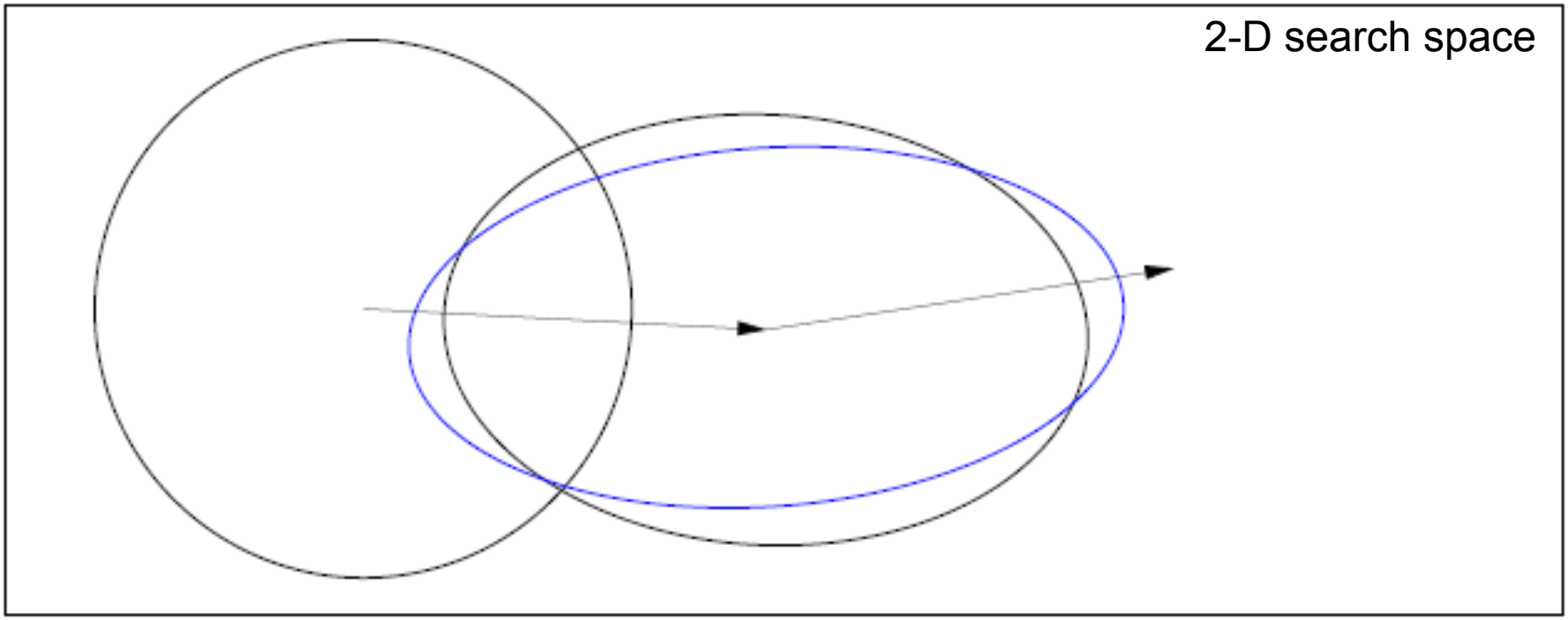


$\mathbf{y}_w$, movement of the population mean m

.

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

Covariance Matrix Adaptation



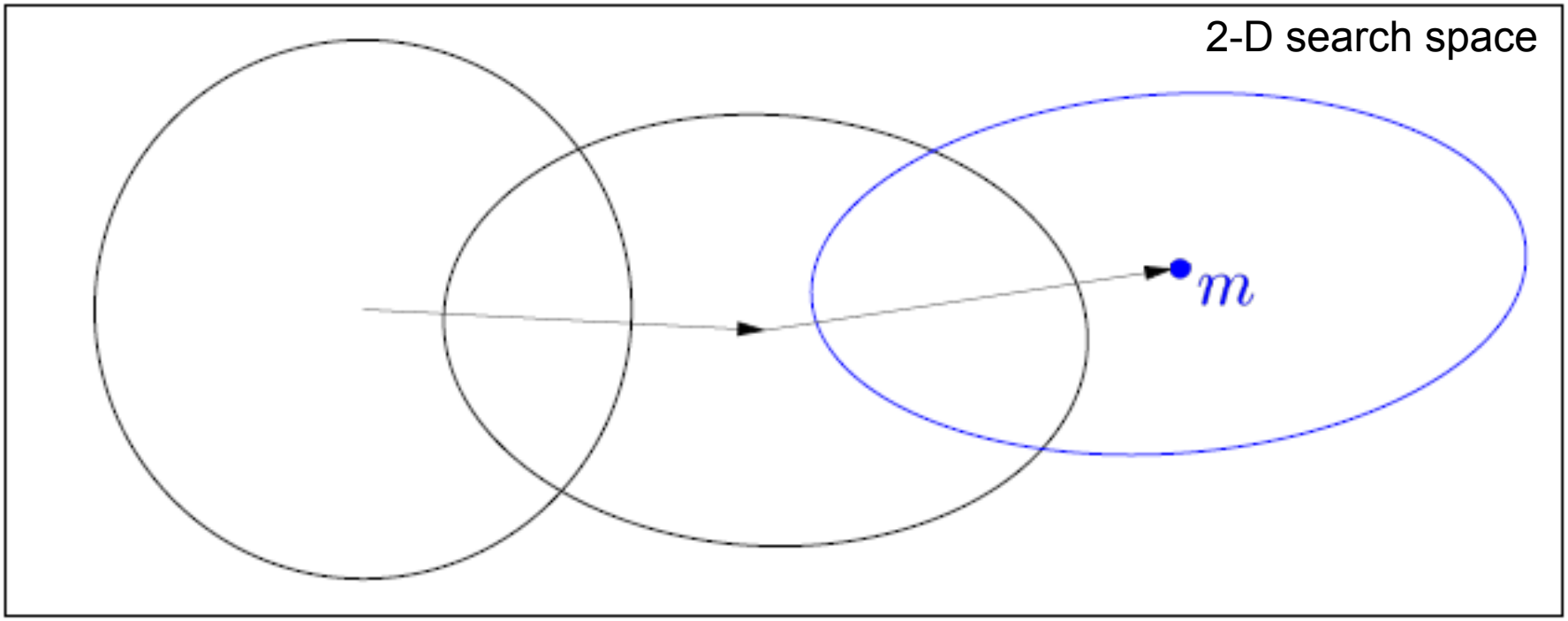
mixture of covariance matrix $\mathbf{C}$ and step $\mathbf{y}_w$

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

.

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

Covariance Matrix Adaptation



new distribution $\mathcal{N}(m, \mathbf{C}) = m + \mathcal{N}(\mathbf{0}, \mathbf{C})$

ruling principles:

- increase the likelihood
  - of successful points by updating $m \leftarrow m + \mathbf{y}_w$
  - of successful steps by updating $\mathbf{C} \leftarrow 0.8\,\mathbf{C} + 0.2\,\mathbf{y}_w\mathbf{y}_w^{\mathrm{T}}$
- increase $\mathbb{E}[w_k(f(x))]$ by natural gradient descent in m and $\mathbf{C}$

[Kjellstroem 1991, Hansen&Ostermeier 1996, Ljung 1999]

CMA-ES (Covariance Matrix Adaptation Evolution Strategy)
= natural gradient ascent + cumulation + step-size control

Input: $m \in \mathbb{R}^n$, $\sigma \in \mathbb{R}_+$, $\lambda \in \mathbb{N}_{\geq 2}$, usually $\lambda \geq 5$ (a default is available)

Set $c_m = 1$, $c_\mu \approx \mu_w/n^2$, $c_c \approx 4/n$, $c_\sigma \approx 4/n$, $c_1 \approx 2/n^2$, $d_\sigma \approx 1$,
set $\widehat{w}_{i=1,\ldots,\lambda}$ decreasing in i, $\sum_i |\widehat{w}_i| = 1$ and $\mu_w^{-1} := \sum_i \widehat{w}_i^2 \approx 3/\lambda$
Initialize $\mathbf{C} = \mathbf{I}$, and $\mathbf{p}_c = \mathbf{0}$, $\mathbf{p}_\sigma = \mathbf{0}$

While not *terminate*

$$\mathbf{x}_i = m + \sigma\, \mathbf{y}_i \sim \mathcal{N}\left(m, \sigma^2 \mathbf{C}\right), \quad \text{for } i = 1, \ldots, \lambda \qquad \text{sampling}$$

$$m \leftarrow m + c_m \sigma \textstyle\sum_i \widehat{w}_{\rho(i)}\, \mathbf{y}_i =: m + c_m \sigma \mathbf{y}_w, \qquad \text{update mean}$$

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

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

$$\mathbf{p}_c \leftarrow (1 - c_c)\, \mathbf{p}_c + \mathbb{1}_{[0,2n]}\left\{\|\mathbf{p}_\sigma\|^2\right\} \sqrt{1 - (1 - c_c)^2}\sqrt{\mu_w}\, \mathbf{y}_w \qquad \text{path for } \mathbf{C}$$

$$\mathbf{C} \leftarrow \mathbf{C} + c_\mu \textstyle\sum_{i=1}^{\lambda} \widehat{w}_{\rho(i)} \left(\mathbf{y}_i \mathbf{y}_i^{\mathrm{T}} - \mathbf{C}\right) + c_1 \left(\mathbf{p}_c\, \mathbf{p}_c^{\mathrm{T}} - \mathbf{C}\right) \qquad \text{update } \mathbf{C}$$

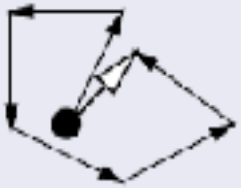
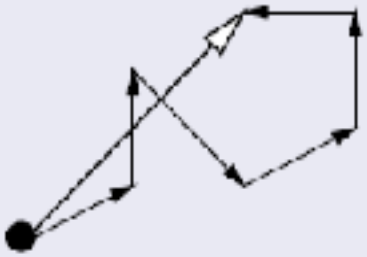
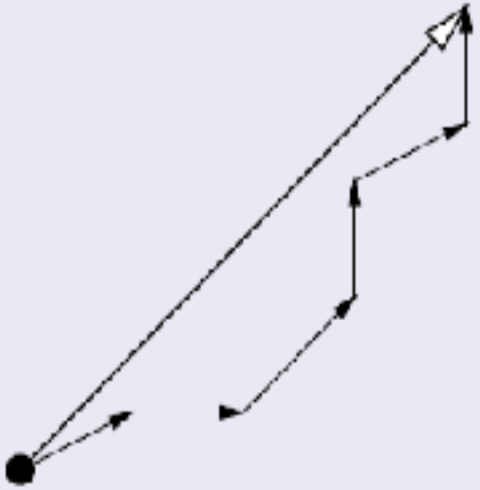
green shade: natural gradient

red shade: not explained by natural gradient

Step-Size Control: The Concept

search paths in 2-D

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

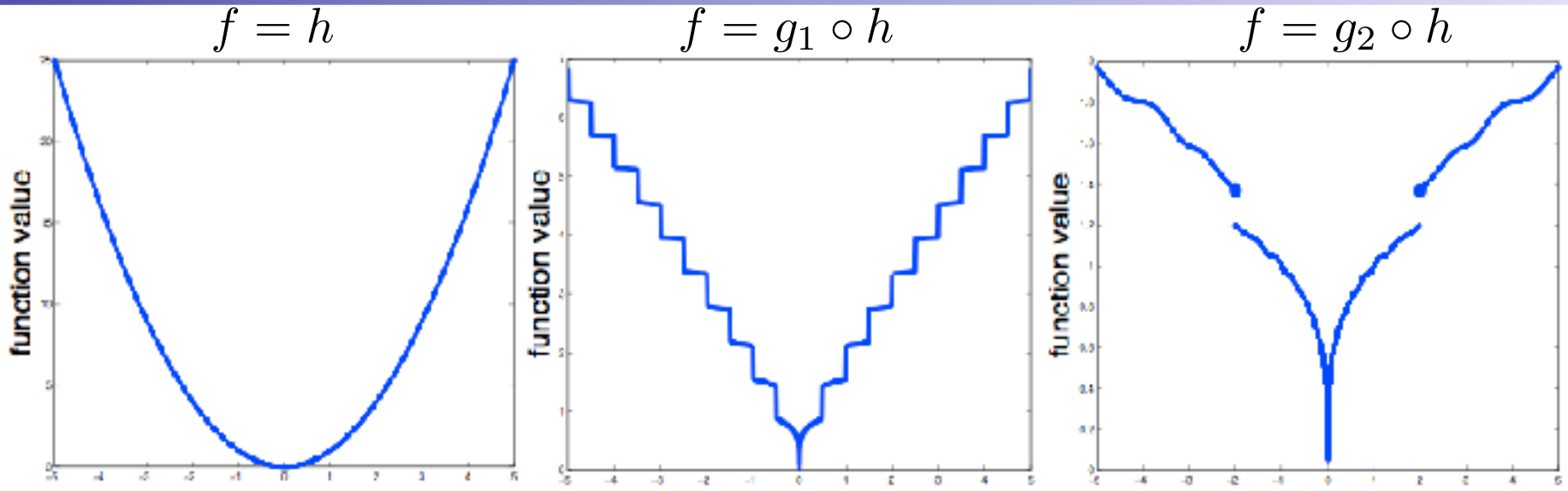
short	expected	long
		
too large step-size	neutral and optimal step-size	too small step-size

If several updates go into the same or similar direction (if they have the same sign) increase the step-size

Design Principles Applied in CMA-ES

- Minimal prior assumptions
  - stochastics helps, maximum entropy distribution
  - improvement only by selection of solutions
  - harder to deceive
- Exploit all available information
  - given remaining design principles (e.g. invariance)
  - e.g. cumulation exploits "sign" information
- Stationarity or unbiasedness
  - parameters remain unchanged under "random" ranking
- Almost parameter-less
  - meaningful parameters whose choice is f-independent,
  - e.g. learning rates (time horizons)
- Retain and introduce invariance properties

Invariance: Function-Value Free Property



Three functions belonging to the same equivalence class

A *function-value free search algorithm* is invariant under the transformation with any order preserving (strictly increasing) g.

Invariances make

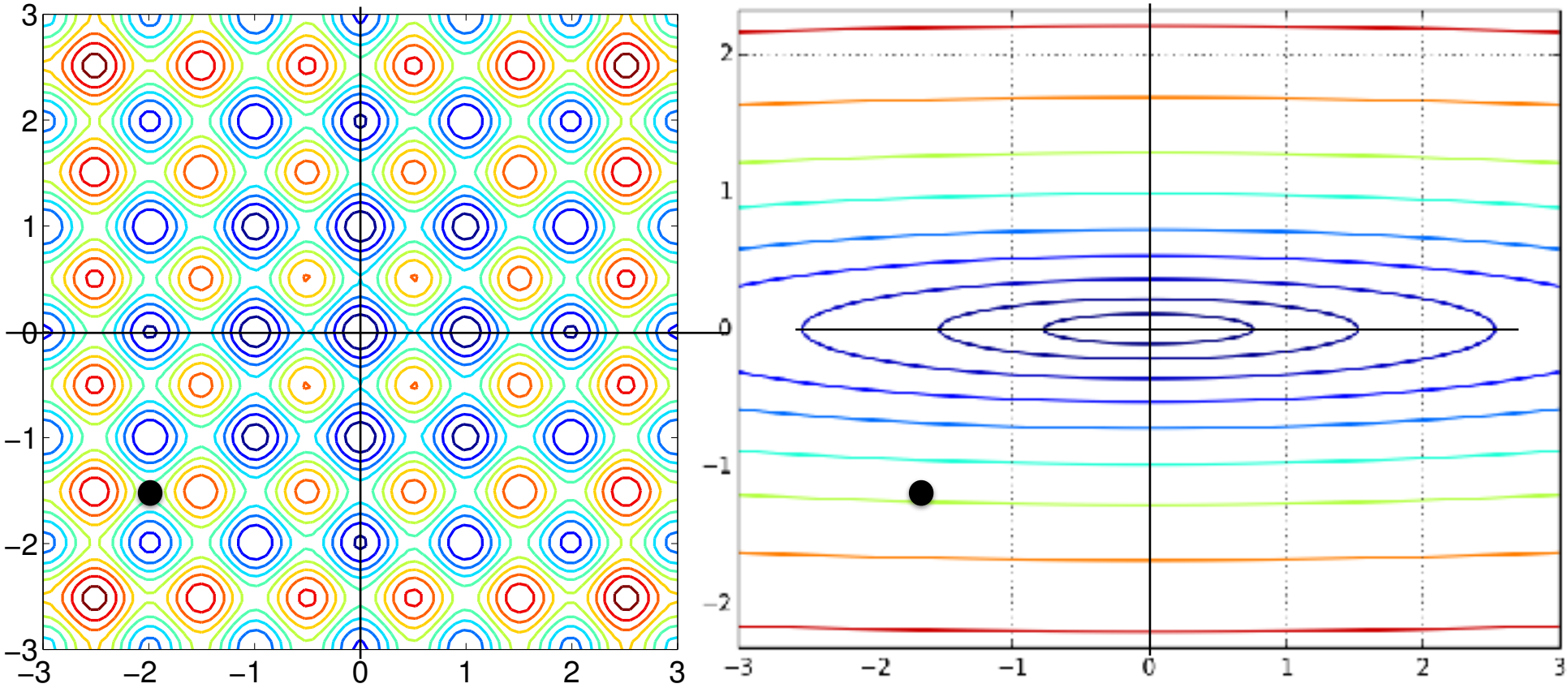
- observations meaningful
- algorithms predictable and/or ”robust”

as a rigorous notion of generalization

28

Invariance Under Rigid Search Space

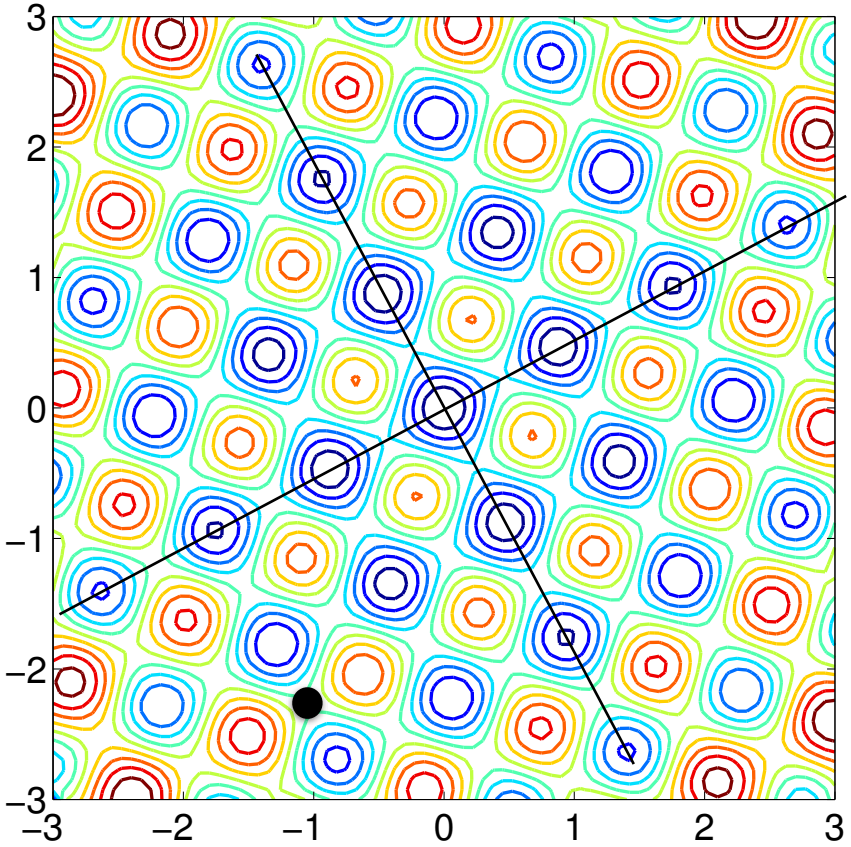
$f = h_{\mathrm{Rast}}$ f-level sets in dimension 2 $f = h$

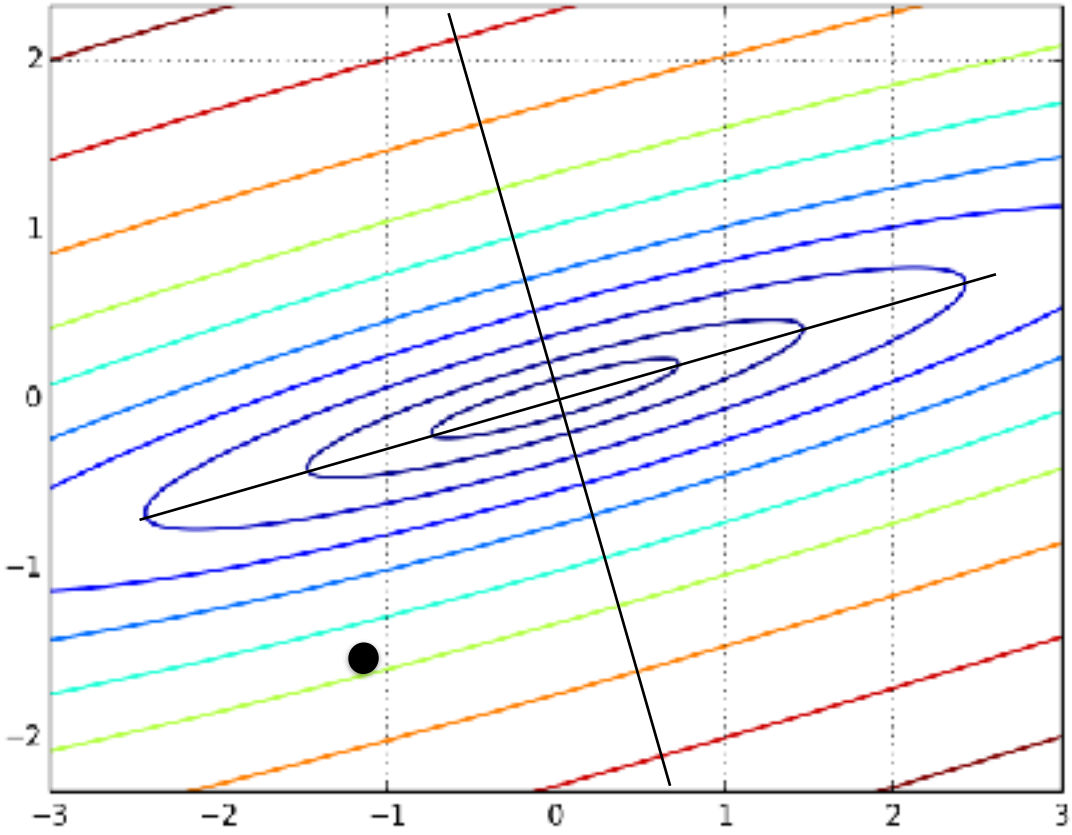


for example, invariance under search space rotation
(separable $\Leftrightarrow$ non-separable)

Invariance Under Rigid Search Space

$f = h_{\mathrm{Rast}} \circ R$ f-level sets in dimension 2 $f = h \circ R$





for example, invariance under search space rotation
(separable ⇔ non-separable)

invariances make observations meaningful

Landscape of Continuous Search Methods

Gradient-based (Taylor, local)

- Conjugate gradient methods [Fletcher & Reeves 1964]
- Quasi-Newton methods (BFGS) [Broyden et al 1970]

Derivative-free optimization (DFO)

- Trust-region methods (NEWUOA, BOBYQA) [Powell 2006, 2009]
- Simplex downhill [Nelder & Mead 1965]
- Pattern search [Hooke & Jeeves 1961, Audet & Dennis 2006]

Stochastic (randomized) search methods

- Evolutionary algorithms (broader sense, continuous domain)
  - Differential Evolution [Storn & Price 1997]
  - Particle Swarm Optimization [Kennedy & Eberhart 1995]
  - **Evolution Strategies** [Rechenberg 1965, Hansen & Ostermeier 2001]
- Simulated annealing [Kirkpatrick et al 1983]
- Simultaneous perturbation stochastic approximation (SPSA) [Spall 2000]

31

Limitations of CMA-ES

- 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 $\ll 100n$)
    model-based methods

Questions?

http://cma.gforge.inria.fr
http://cma.gforge.inria.fr/cmaes_sourcecode_page.html

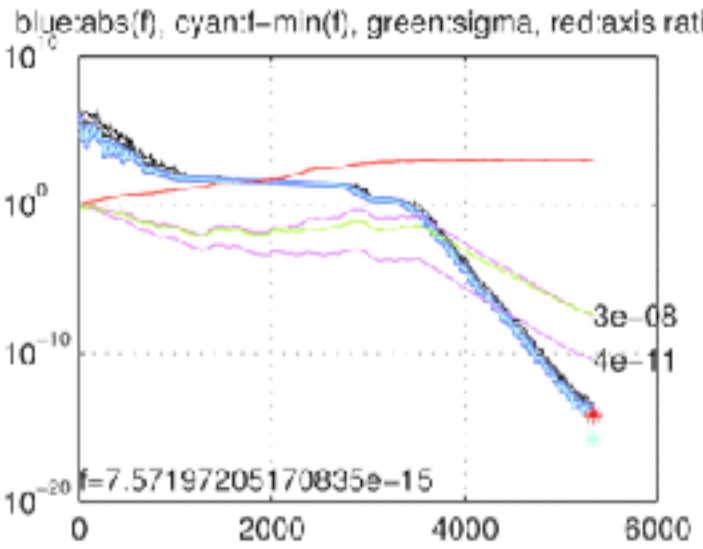
…(experimental) validation...

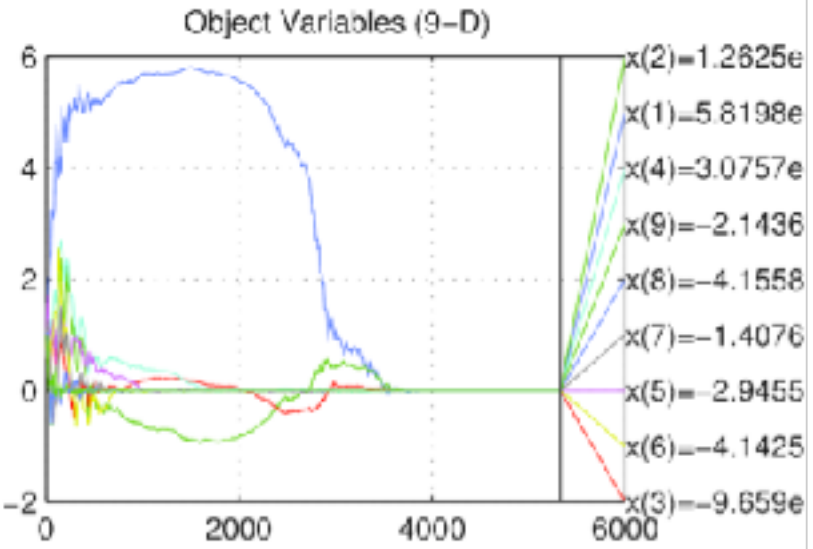
A simple unimodal test function

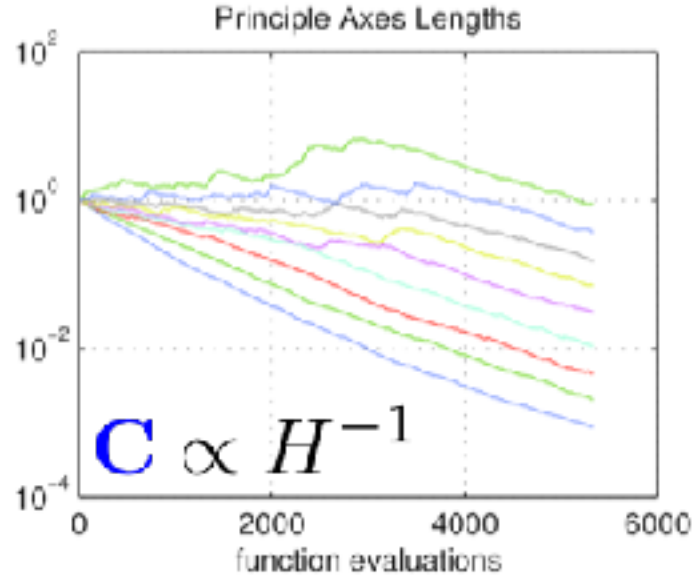
$$f(x) = g\left(\frac{1}{2}x^T H x\right)$$

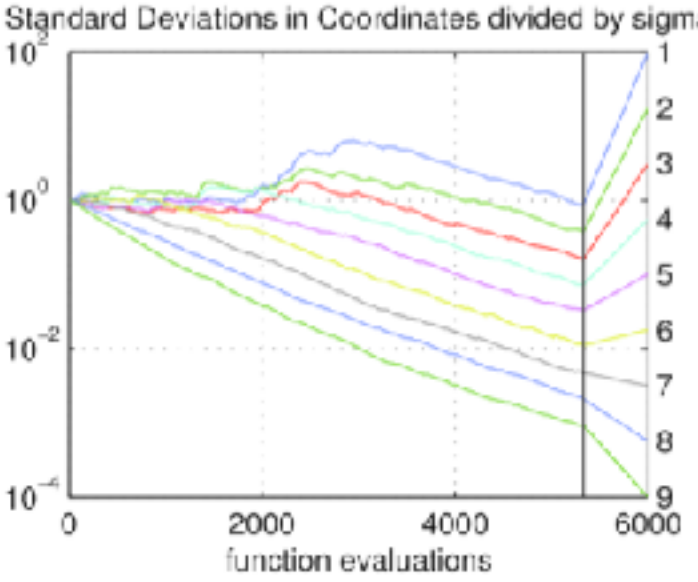
- for different strictly monotonic (i.e. order-preserving) $g : \mathbb{R} \to \mathbb{R}$
- with uniform eigenspectrum of the Hessian H
- with different condition numbers of H (ratio between largest and smallest eigenvalue) between one and 10^{10}
- in dimension 9 and 20

Experimentum crucis







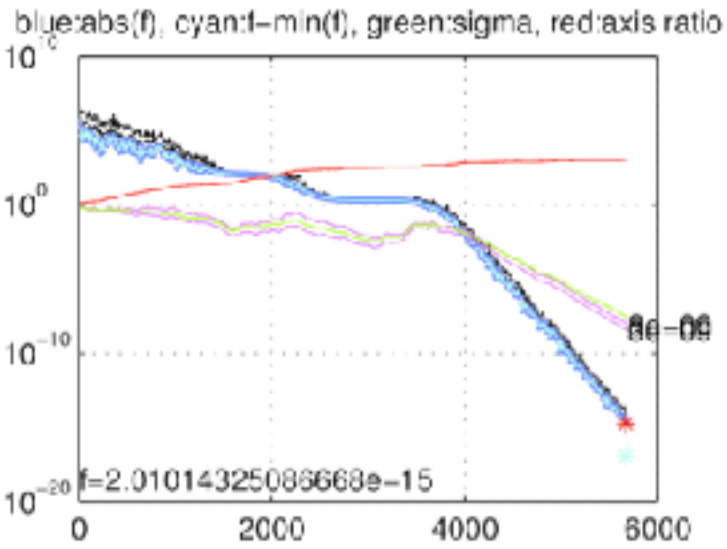


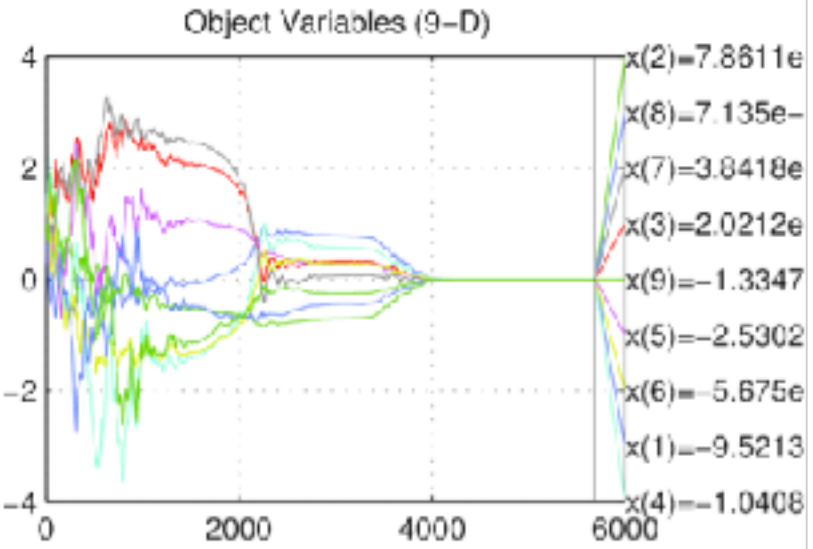
$$f(x) = \sum_{i=1}^{n} \alpha_i x_i^2$$

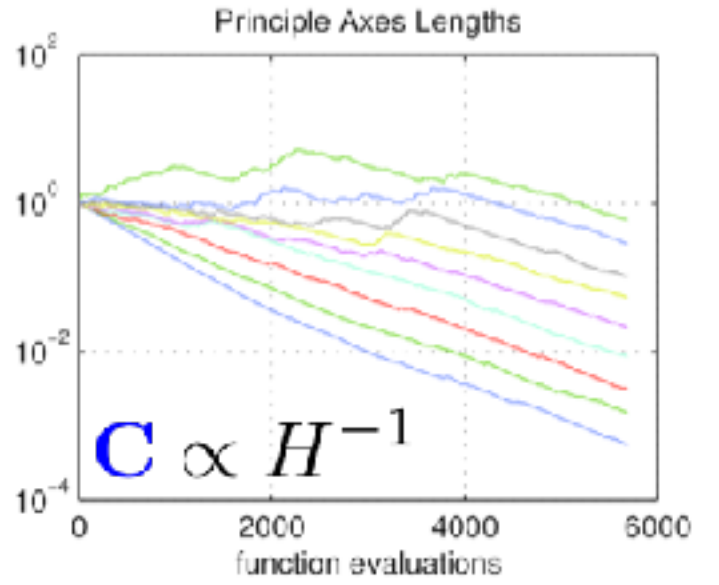
$$\alpha_i = 10^{6\frac{i-1}{n-1}}$$

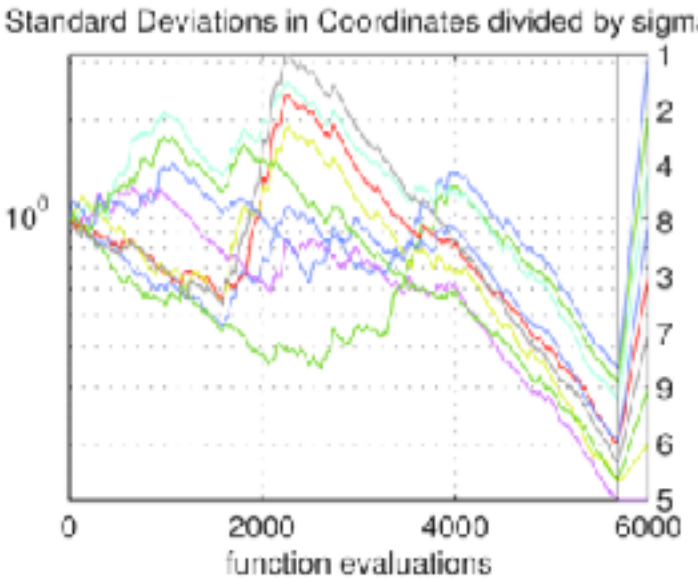
without covariance matrix adaptation it takes 1000 times longer to reach $f = 10^{-10}$

Experimentum crucis









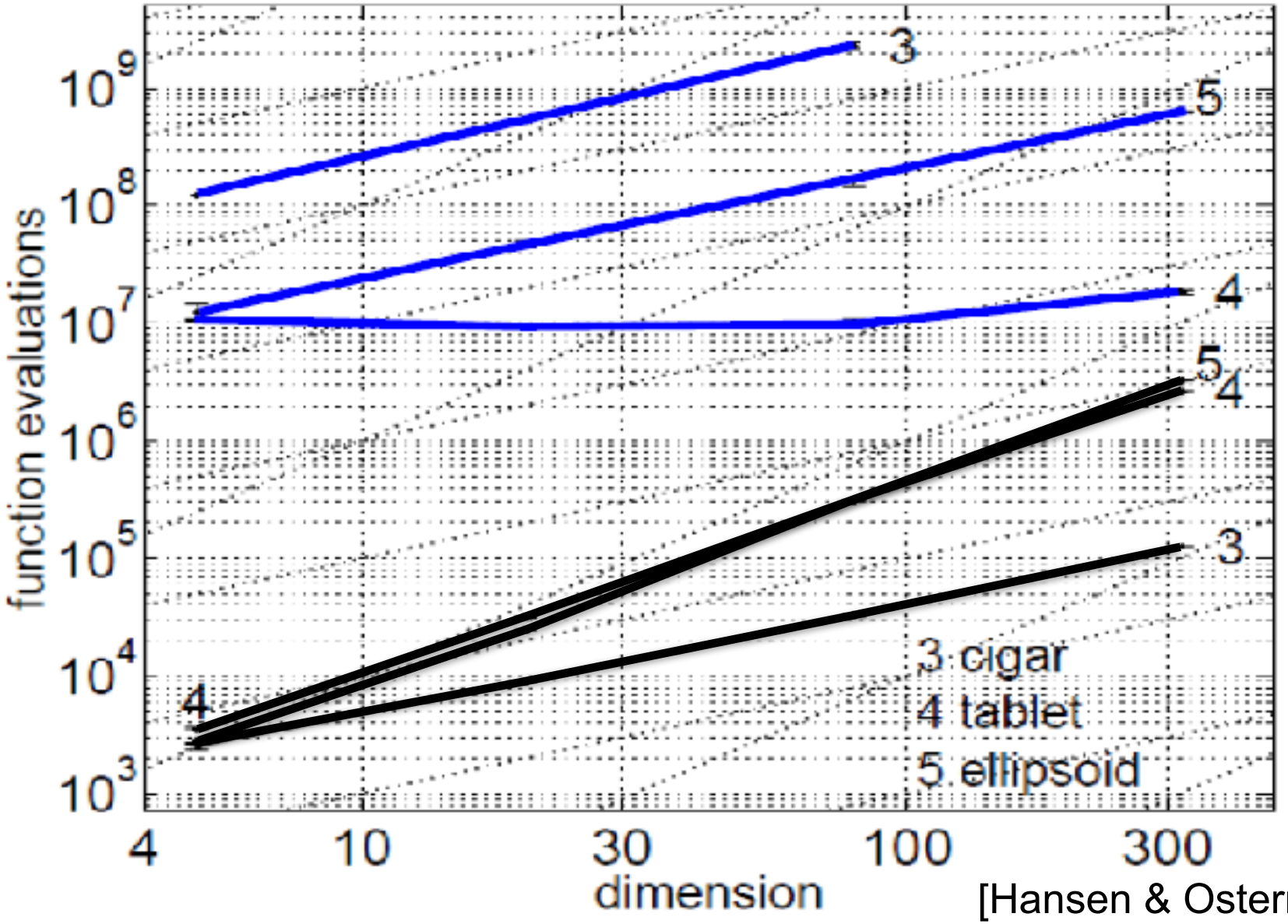
$$f(x) = \sum_{i=1}^{n} \alpha_i y_i^2$$

$$\alpha_i = 10^{6\frac{i-1}{n-1}}$$

$$y = \mathrm{rotation}(x)$$

without covariance matrix adaptation it takes 1000 times longer to reach $f = 10^{-10}$

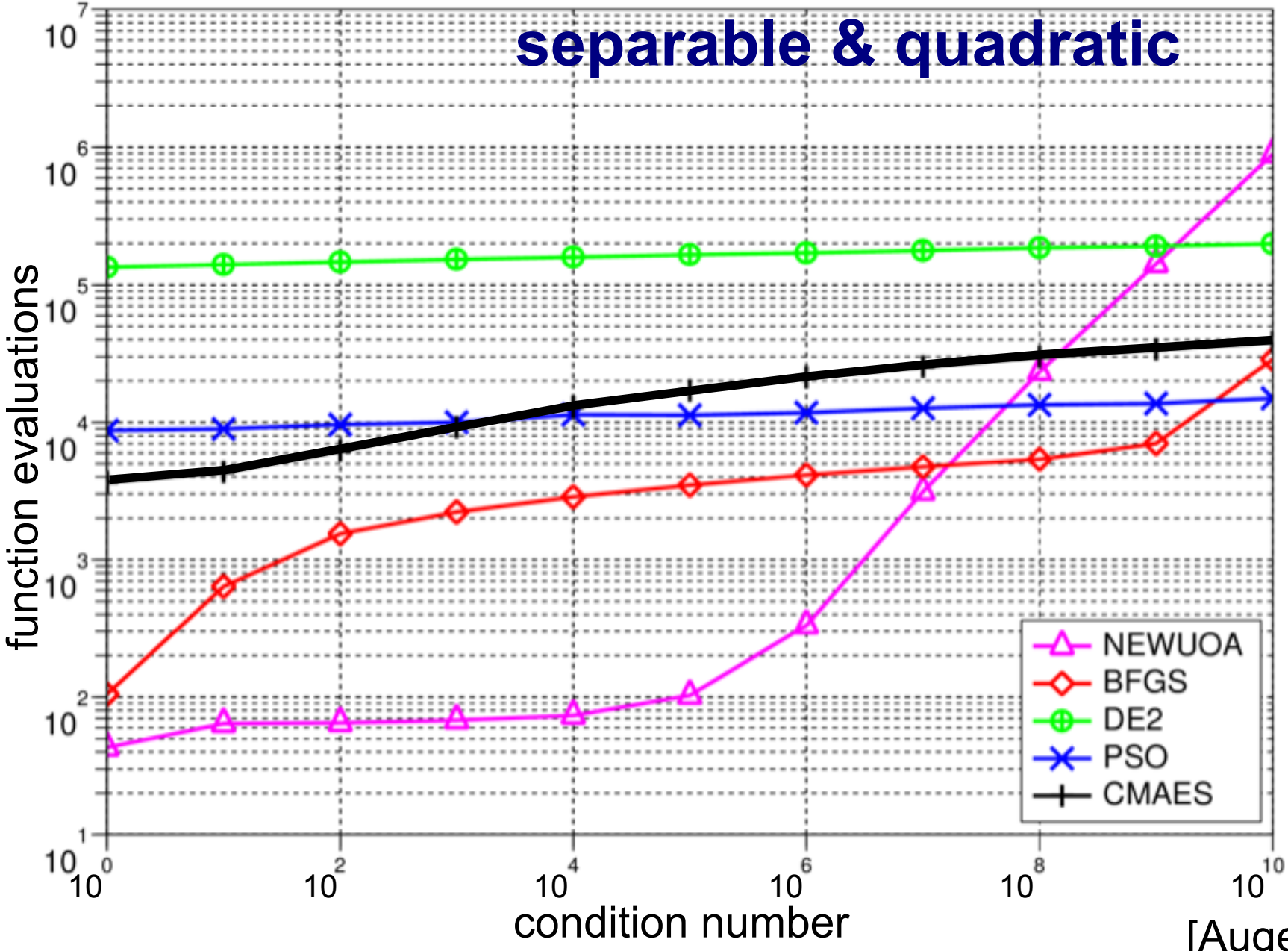
Quantifying the enhancement



[Hansen & Ostermeier 2001]

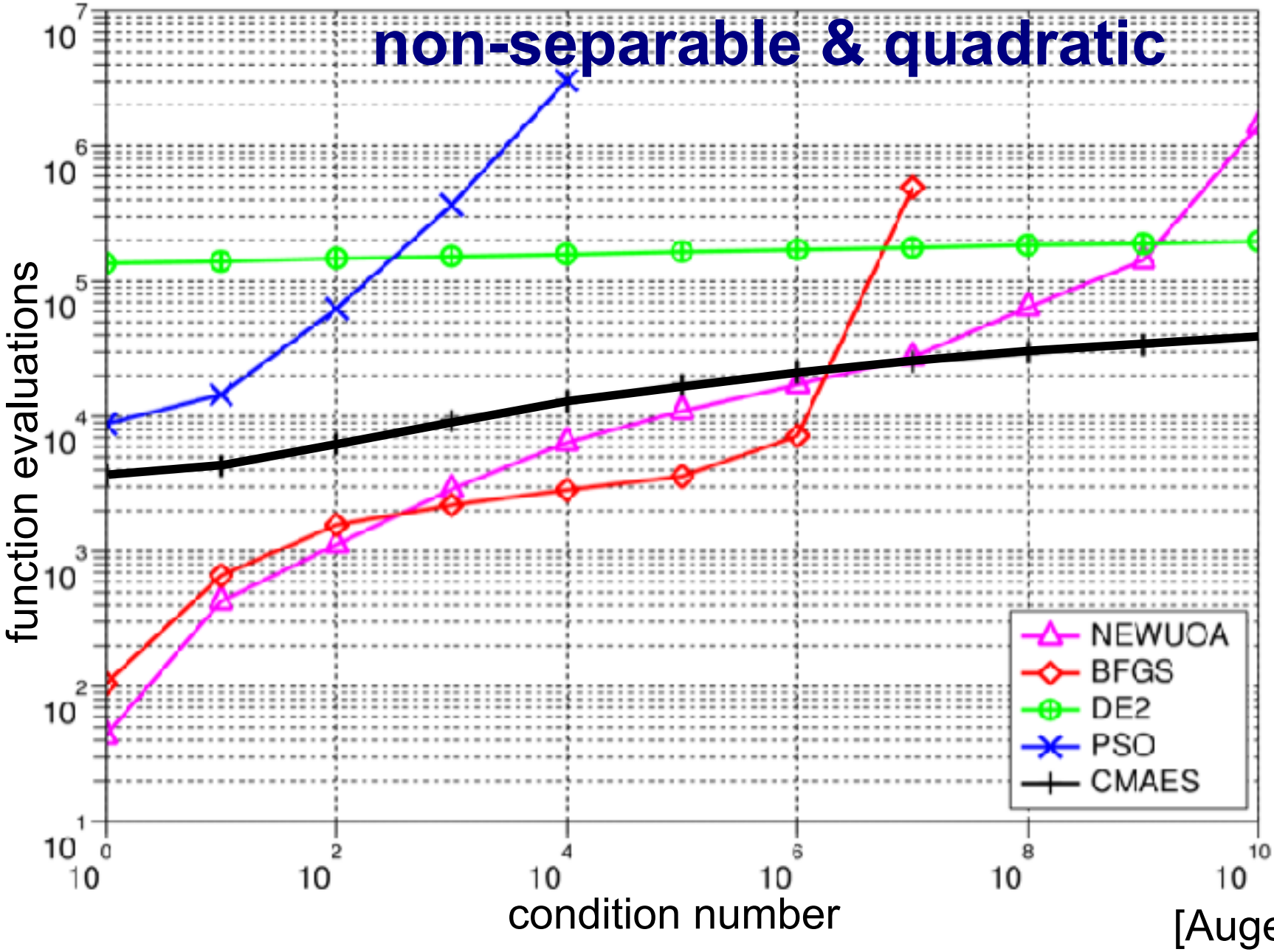
black: CMA-ES ($c_1 \approx 2/n^2$), blue: CSA-ES ($c_1 = 0$)

Runtime versus condition number



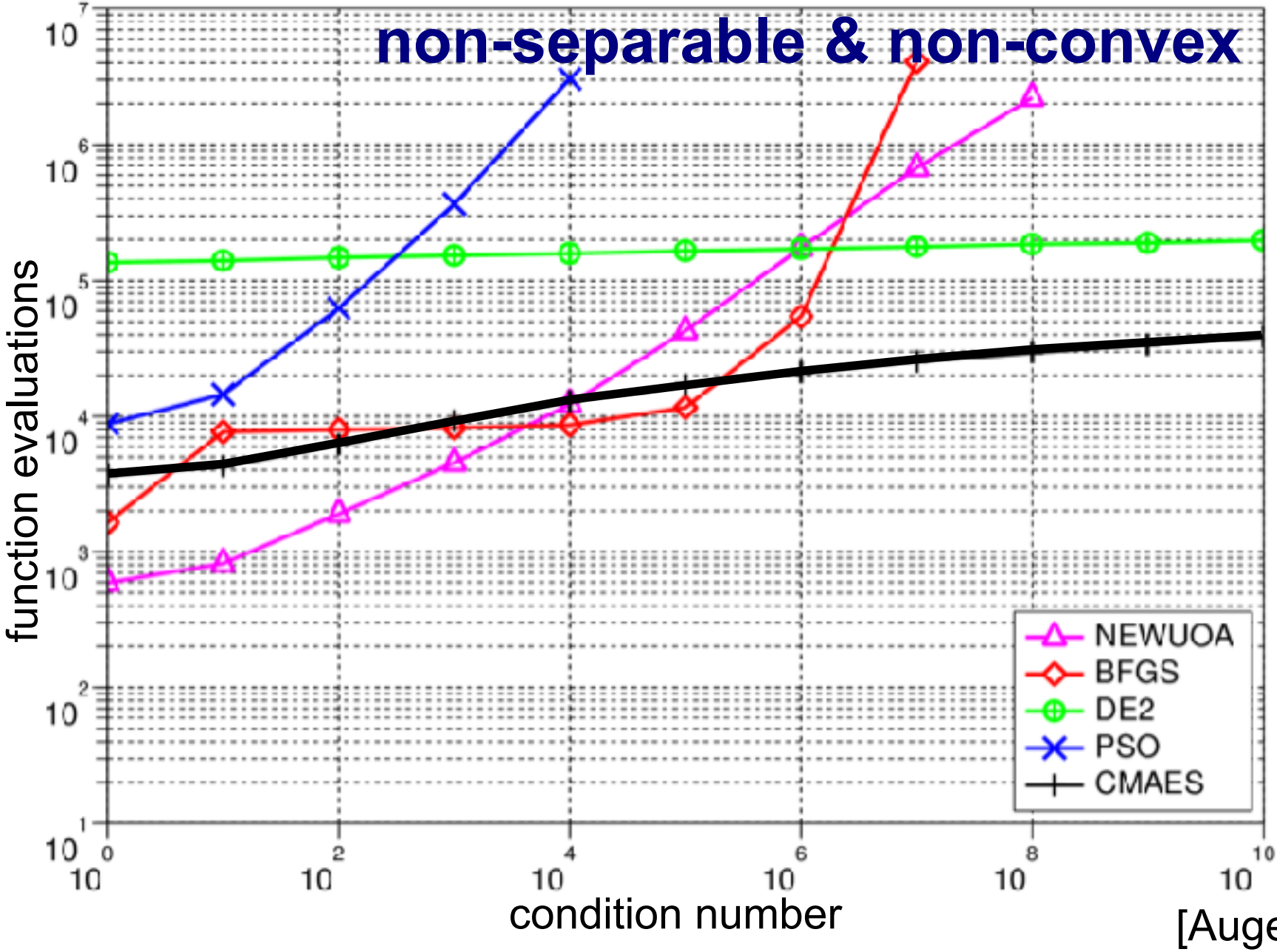
[Auger et al 2009]

Runtime versus condition number



[Auger et al 2009]

Runtime versus condition number



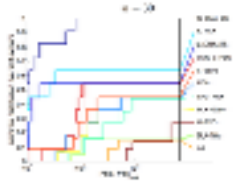
[Auger et al 2009]

Various Benchmarks

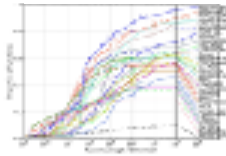
Benchmarks and Applications

- well suited for non-separable, ill-conditioned, rugged problems
  considered as *state-of-the-art*, e.g., Scholarpedia, 2(8):1965, 2007

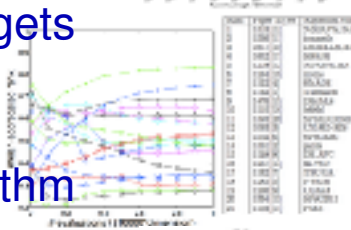
- Used for the RoboCup world champion 2011 & 2012 (team Austin Villa, 3D Nao simulation league)
  in 2013 with a goal difference of 67:1 on 2nd place
- *Benchmarks and Competitions*
  - 2005 IEEE-CEC (Special Session on Real-Parameter Optimization)
    restarts (IPOP-CMA-ES [36]), best algorithm



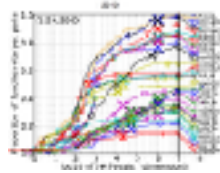
  - 2009 ACM-GECCO (BBOB Black-Box Optimization Benchmarking Workshop)
    restarts (BIPOP-CMA-ES [80,81]), best algorithm for large budgets



  - 2013 IEEE-CEC (Competition on Real-Parameter Single Objective Optimization)
    restarts (NBIPOPaCMA [Loshchilov]), best algorithm

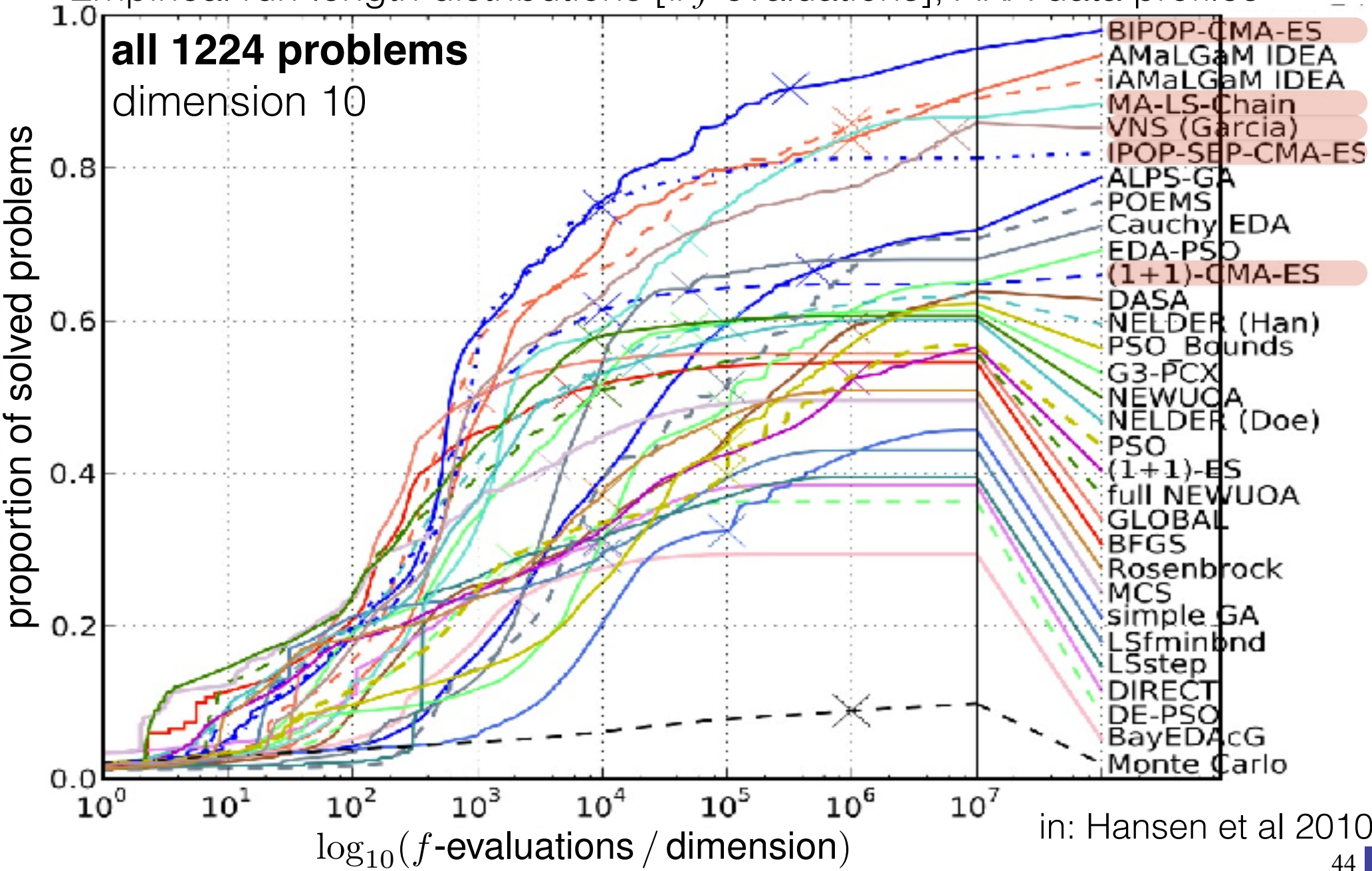


  - 2013 ACM-GECCO (BBOB Black-Box Optimization Benchmarking Workshop)
    portfolio (HCMA [Loshchilov et al]), best algorithm



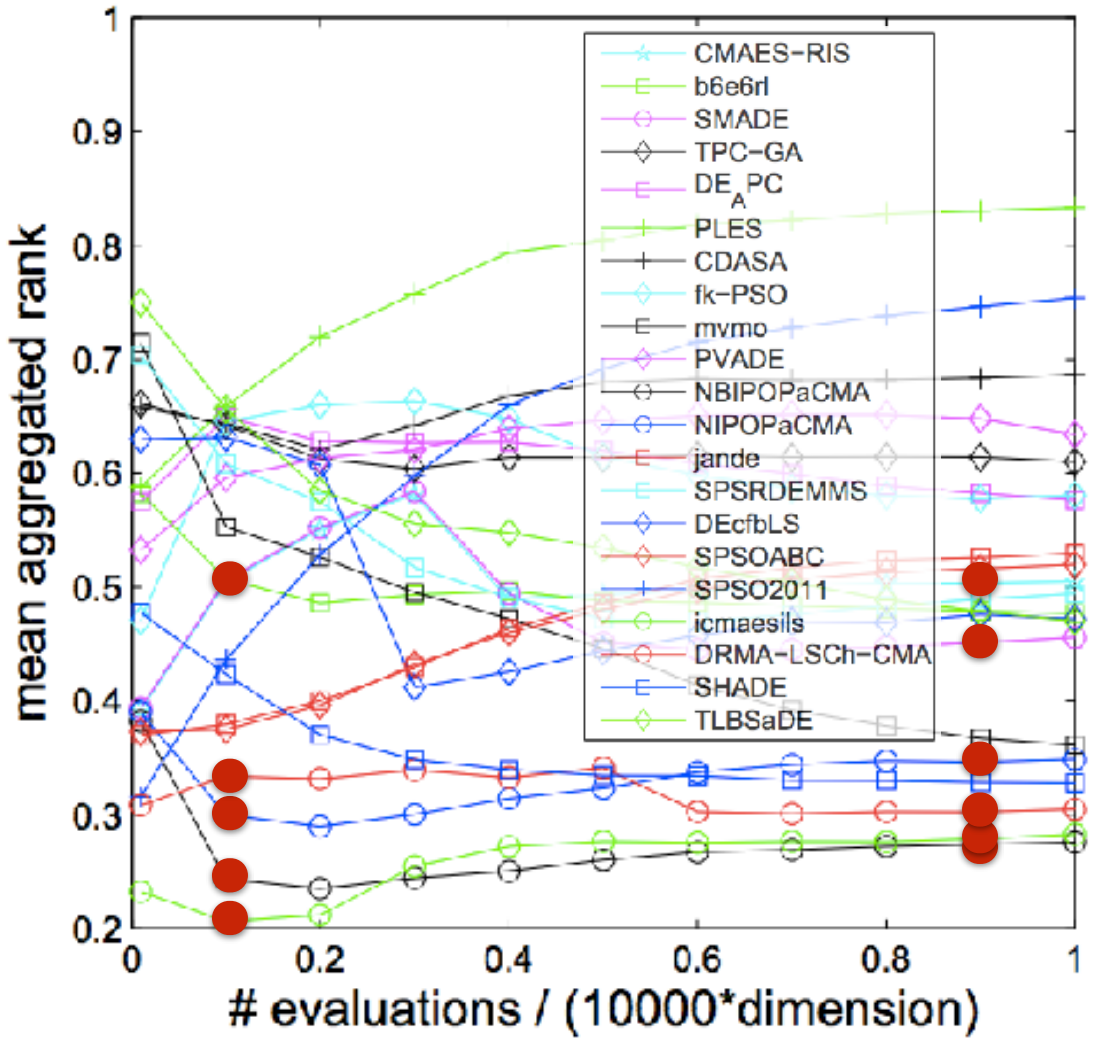
COCO/BBOB: 24 Functions and 51 Target Values

Empirical run-length distributions [# f-evaluations], AKA data profiles



in: Hansen et al 2010

2013 IEEE-CEC Competition



Rank	Algorithm Name	Mean Ranking	Paper ID
1	NBIPOPaCMA	0.27589	1318 [12]
2	icmaesils	0.28289	1566 [11]
3	DRMA-LSCh-CMA	0.30472	1617 [10]
4	SHADE	0.32800	1652 [17]
5	NIPOPaCMA	0.34873	1318 [12]
6	mvmo	0.36127	1284 [16]
7	SMADE	0.45583	1122 [4]
8	TLBSaDE	0.47042	1732 [1]
9	DEcfbLS	0.47222	1476 [15]
10	b6e6rl	0.47687	1110 [18]
11	SPSRDEMMS	0.49421	1393 [20]
12	CMAES-RIS	0.50515	1093 [3]
13	SPSOABC	0.51956	1502 [6]
14	jande	0.52960	1381 [2]
15	DE_APC	0.57617	1148 [8]
16	fk-PSO	0.58058	1267 [13]
17	TPC-GA	0.61008	1132 [7]
18	PVADE	0.63422	1285 [5]
19	CDASA	0.68659	1238 [9]
20	SPSO2011	0.75352	1534 [19]
21	PLES	0.83349	1159 [14]

in: I. Loshchilov , T. Stützle and T. Liao, 2013

28 functions in dimension 10,30,50

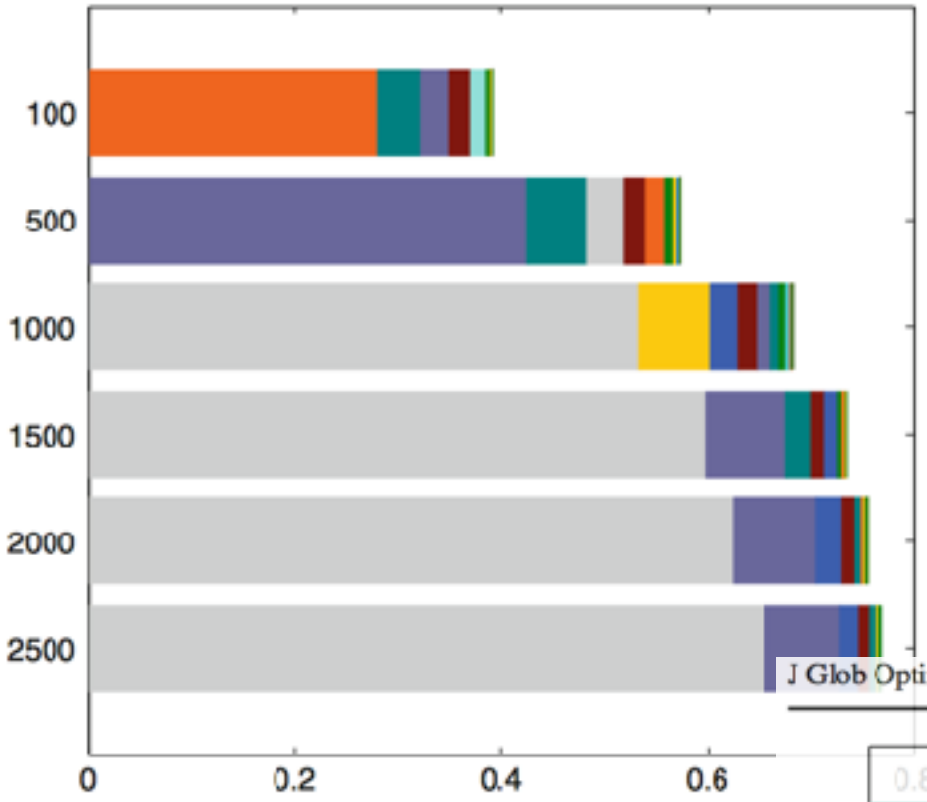


Fig. 21 Minimum number of solvers required to solve all test problems for various limits of function evaluations (best solver performance)

24

n	Number of convex problems smooth	Number of convex problems non-smooth	Number of convex problems total	Number of nonconvex problems smooth	Number of nonconvex problems non-smooth	Number of nonconvex problems total	Total
1–2	0	9	9	86	4	90	99
3–9	6	19	25	97	11	108	133
10–30	30	59	89	27	3	30	119
31–300	42	74	116	35	0	35	153
1–300	78	161	239	245	18	263	502

Table 3 Characteristics of test problems

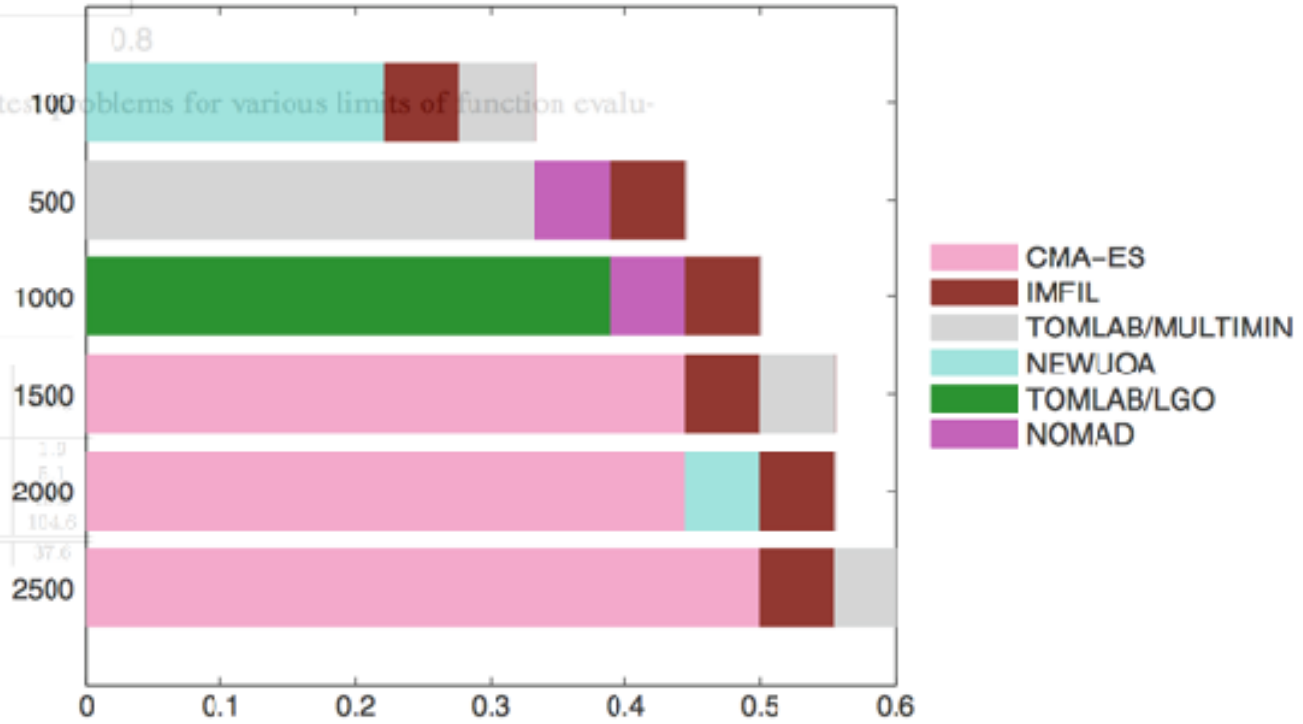


Fig. 20 Minimum number of solvers required to solve nonconvex nonsmooth test problems for various limits of function evaluations (best solver performance)

in L.M. Rios and N.V. Sahinidis, 2013

Questions?

CMA-ES source code: http://www.lri.fr/~hansen/cmaes_inmatlab.html