Advanced Optimization

Lecture 4: Randomized Algorithms for Continuous Problems

Master AIC

Université Paris-Saclay, Orsay, France

Anne Auger

INRIA Saclay – Ile-de-France

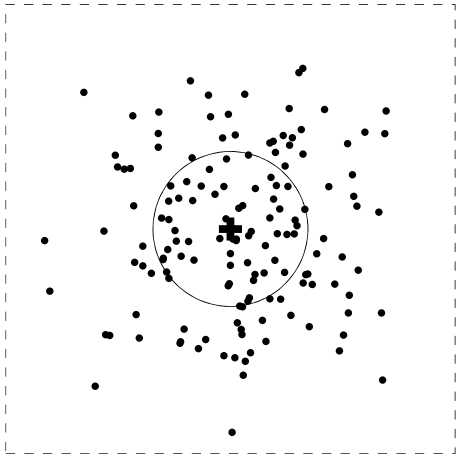
Dimo Brockhoff

INRIA Saclay – Ile-de-France

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

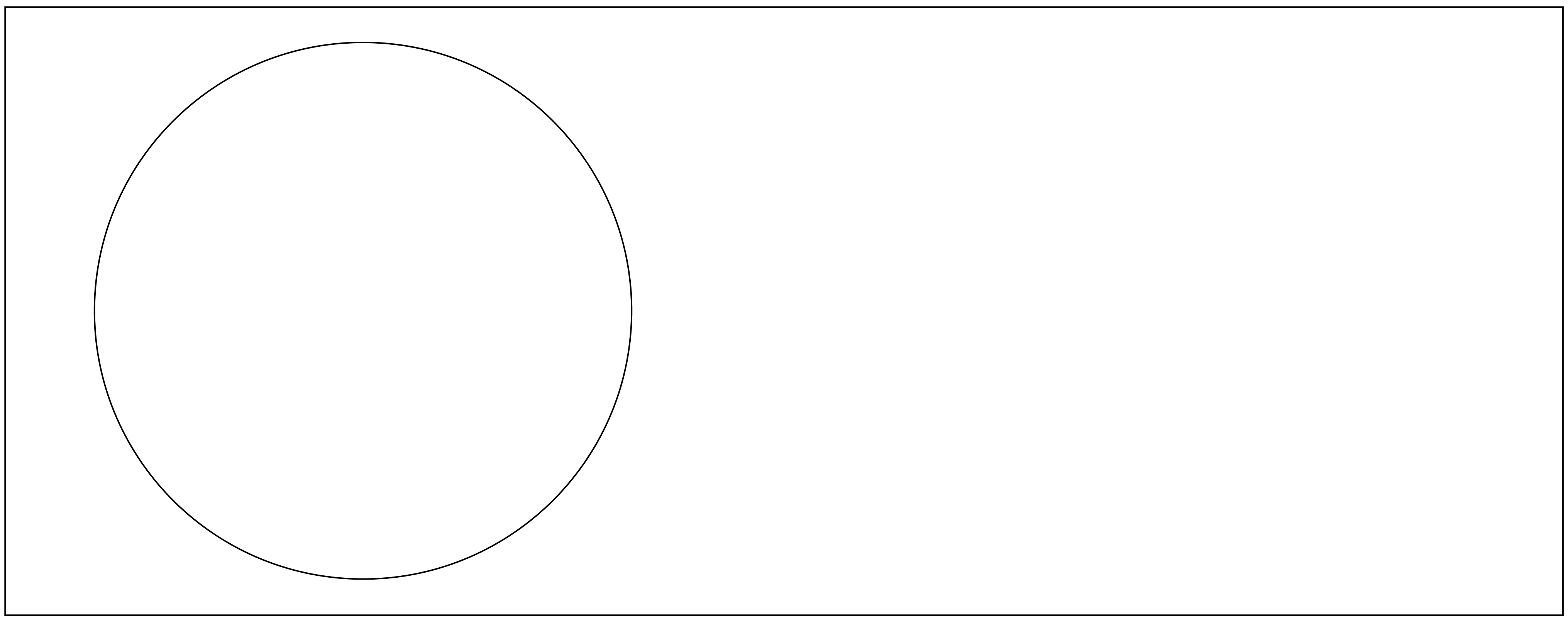
- the mean vector $\boldsymbol{m} \in \mathbb{R}^n$ represents the favorite solution
- the so-called step-size $\sigma \in \mathbb{R}_+$ controls the *step length*
- the covariance matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$ determines the shape of the distribution ellipsoid

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

Covariance Matrix Adaptation

Rank-One Update

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

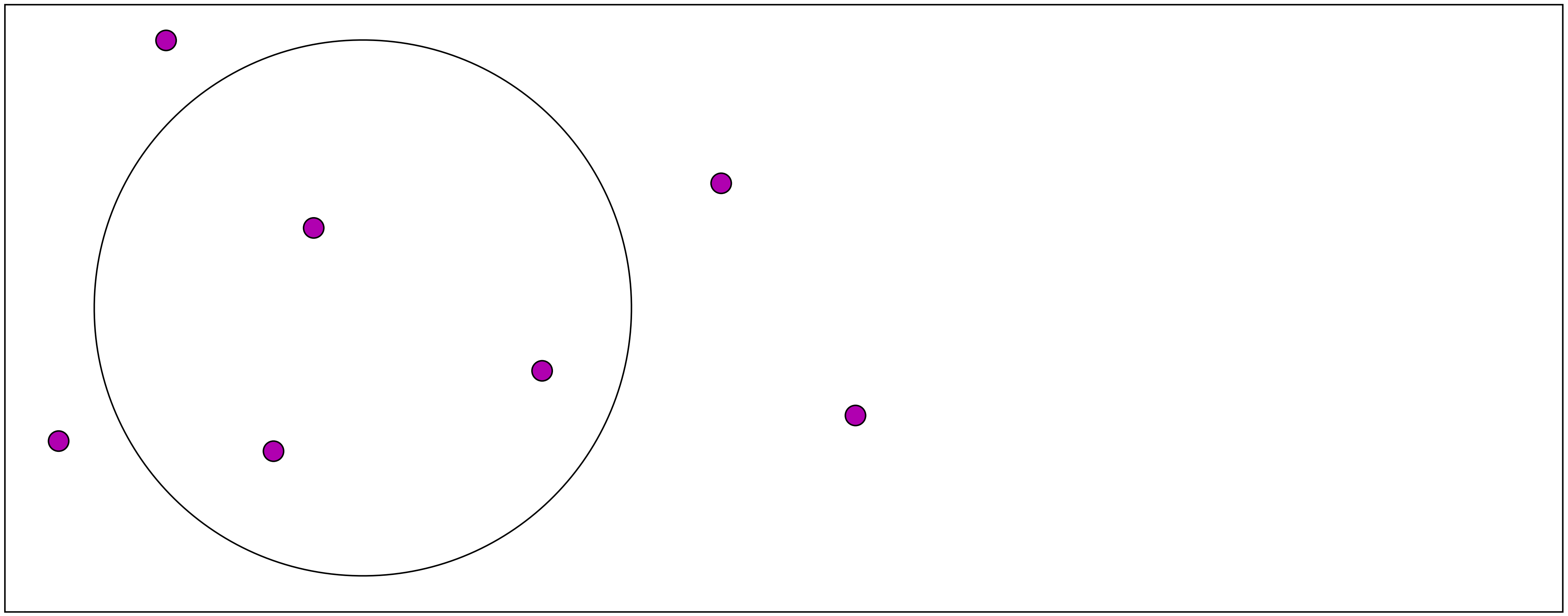


initial distribution, $\mathbf{C} = \mathbf{I}$

Covariance Matrix Adaptation

Rank-One Update

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

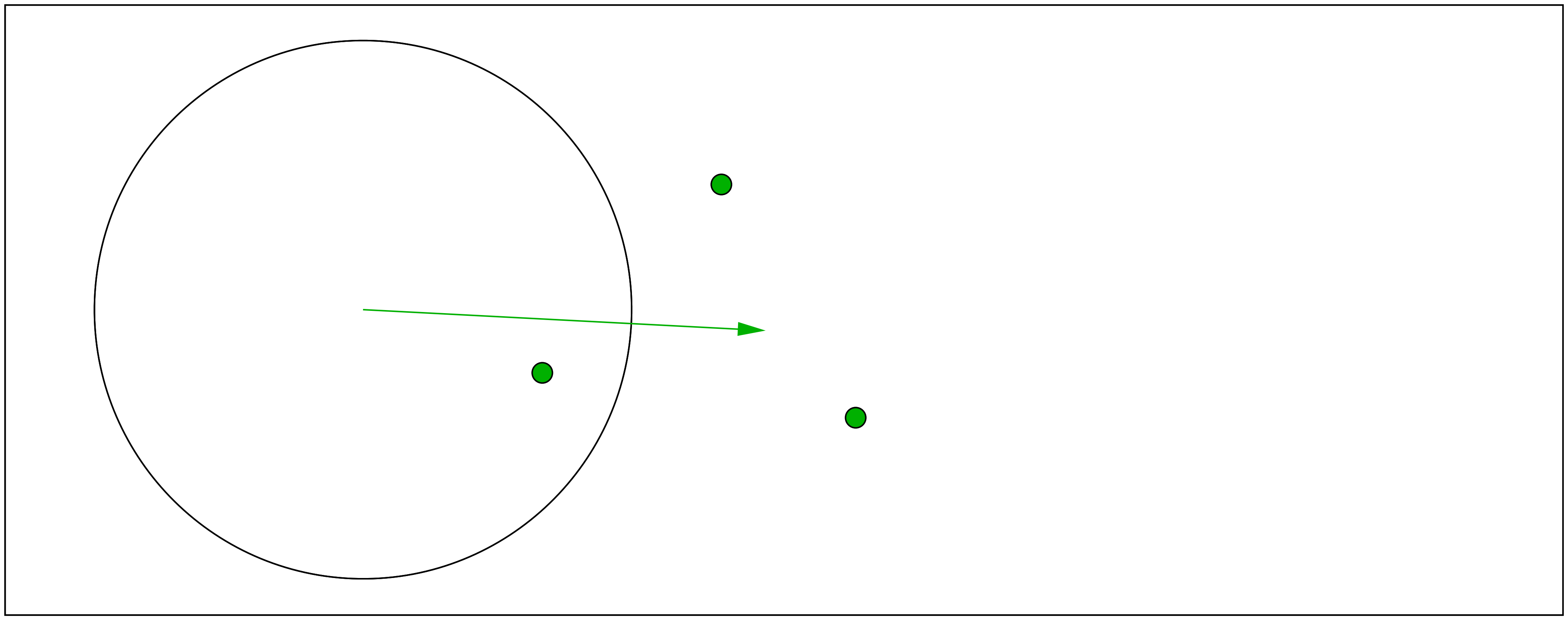


initial distribution, $\mathbf{C} = \mathbf{I}$

Covariance Matrix Adaptation

Rank-One Update

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

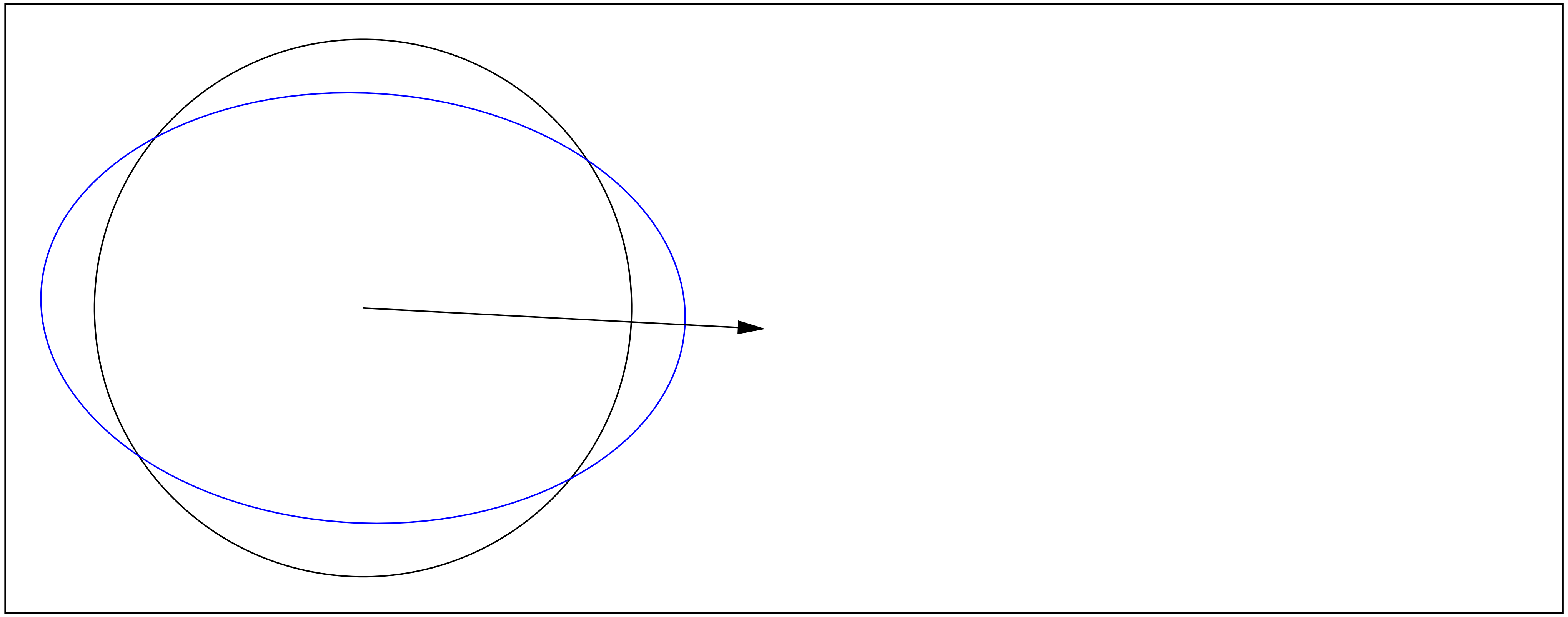


$\boldsymbol{y}_w$, movement of the population mean $\boldsymbol{m}$ (disregarding σ)

Covariance Matrix Adaptation

Rank-One Update

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



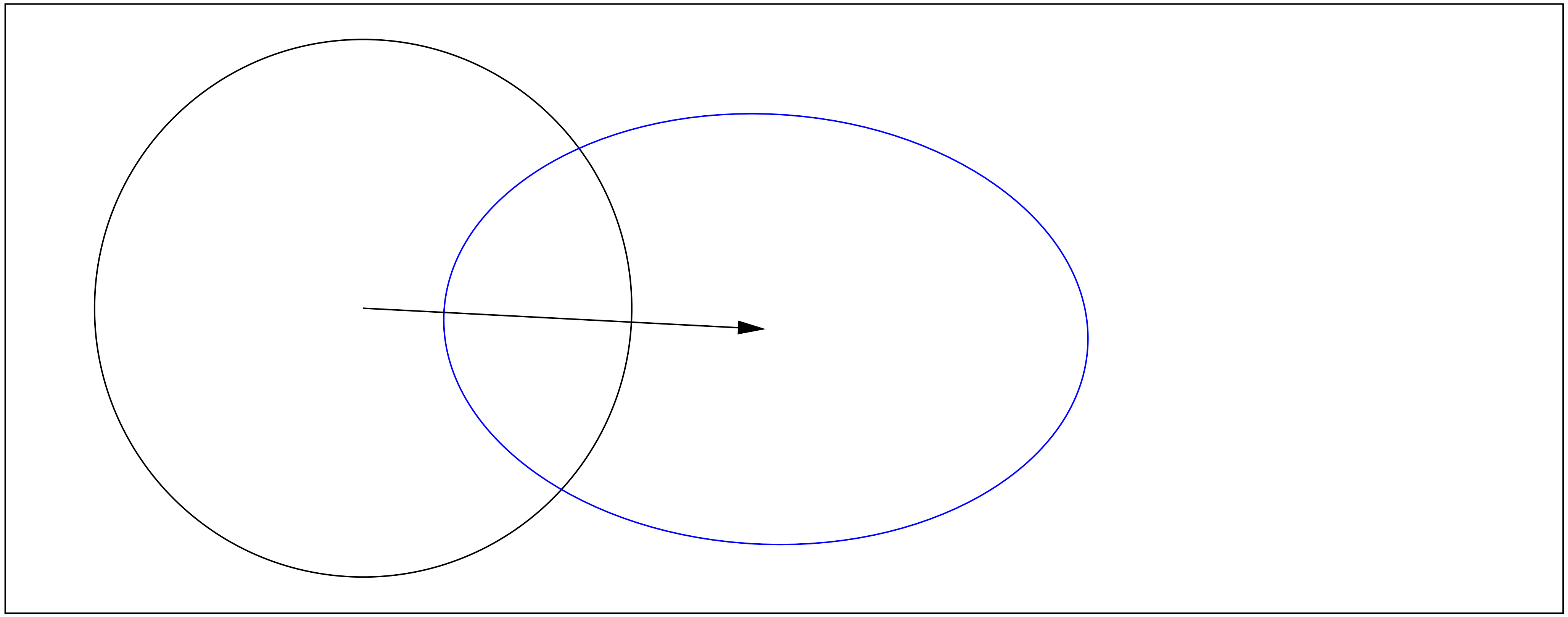
mixture of distribution $\mathbf{C}$ and step $\boldsymbol{y}_w$,

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

Covariance Matrix Adaptation

Rank-One Update

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

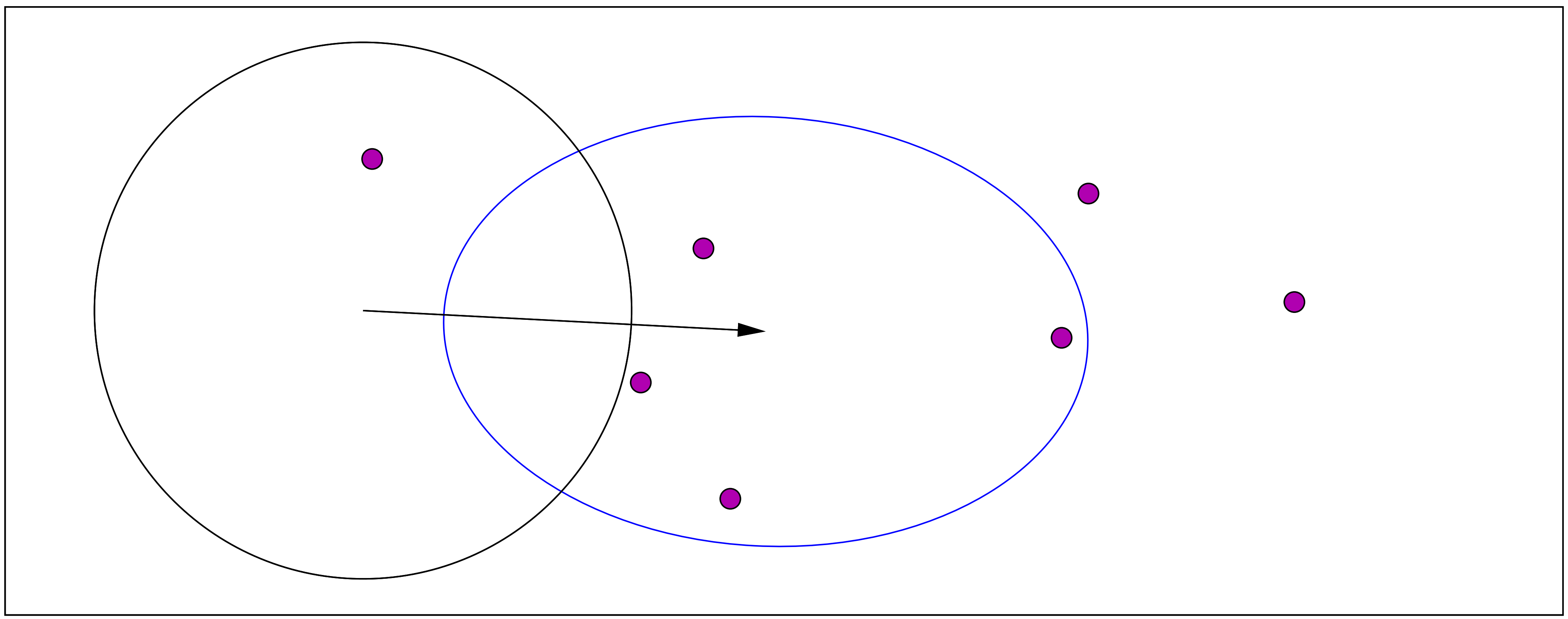


new distribution (disregarding σ)

Covariance Matrix Adaptation

Rank-One Update

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

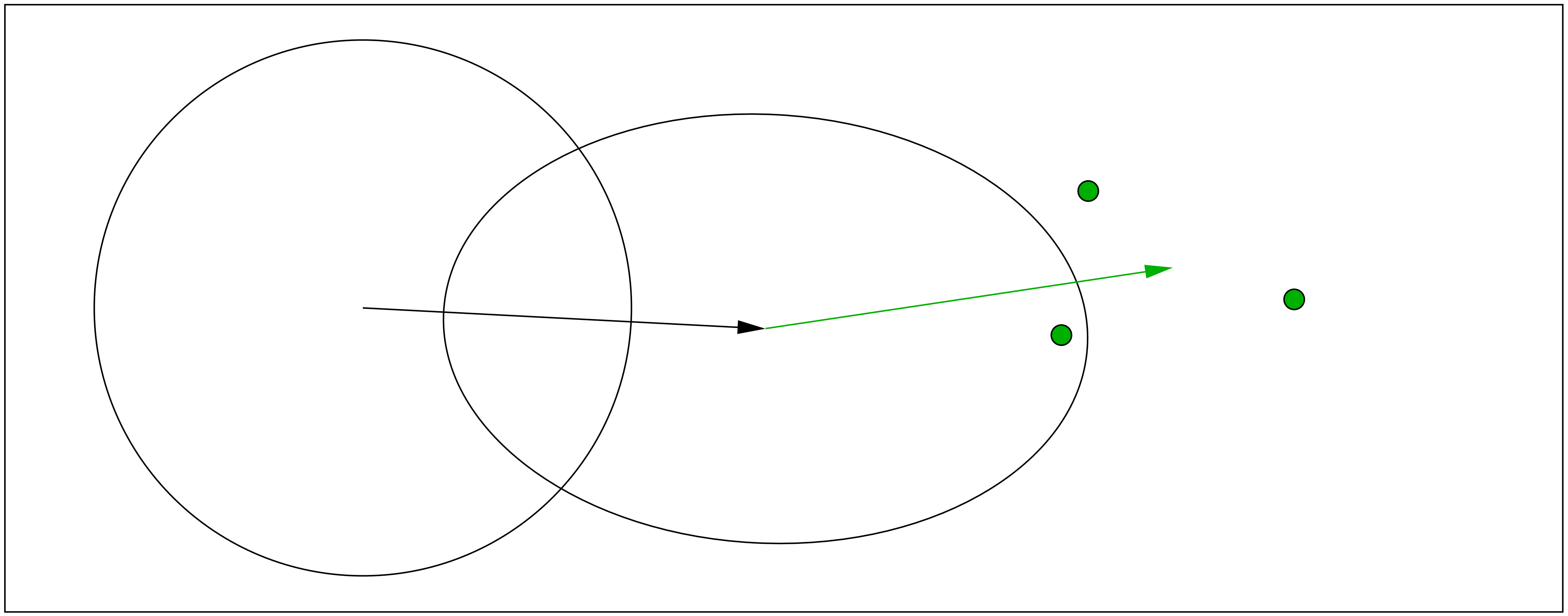


new distribution (disregarding σ)

Covariance Matrix Adaptation

Rank-One Update

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

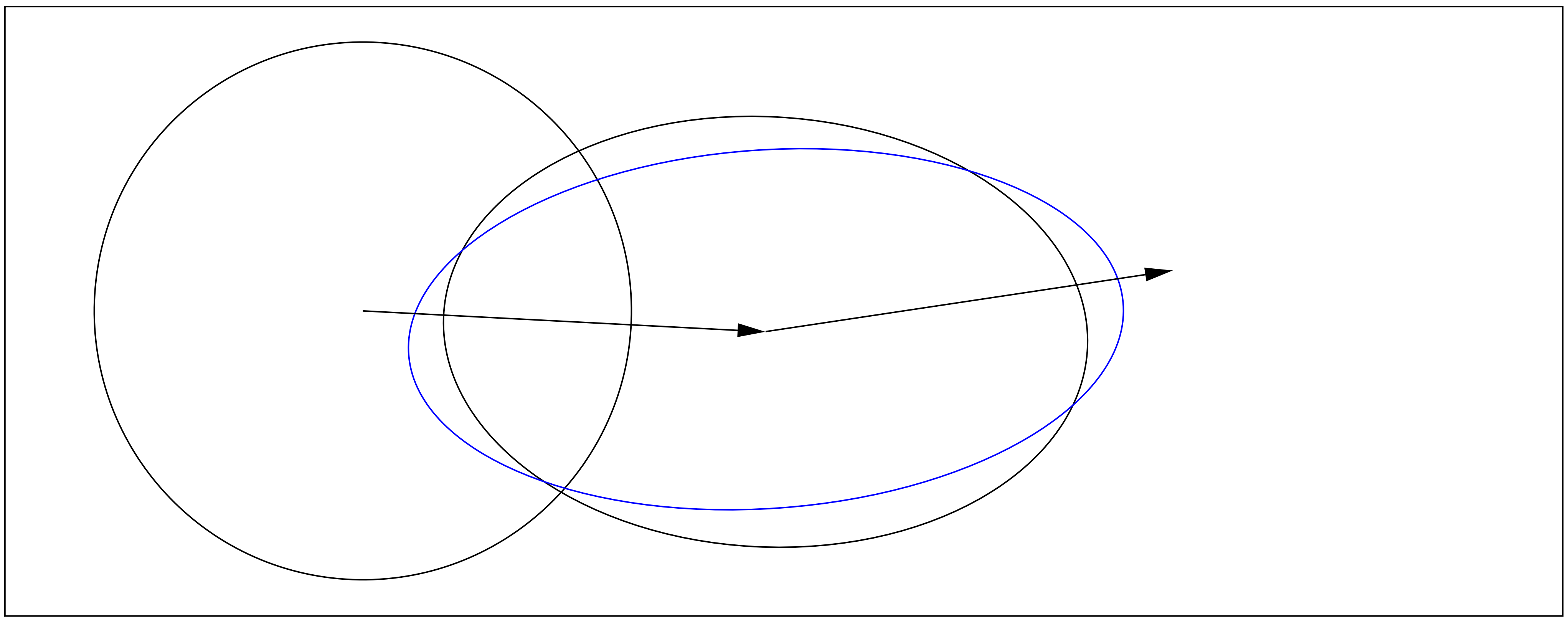


movement of the population mean $\boldsymbol{m}$

Covariance Matrix Adaptation

Rank-One Update

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



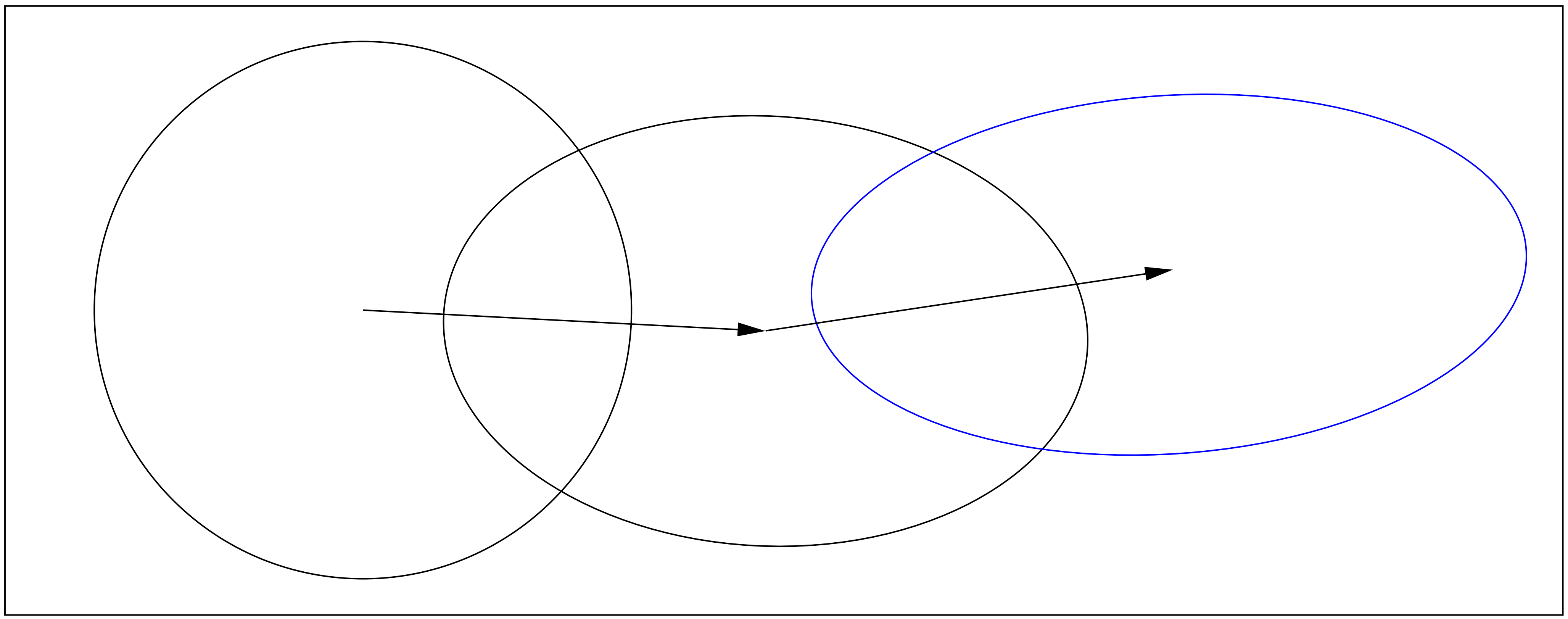
mixture of distribution $\mathbf{C}$ and step $\boldsymbol{y}_w$,

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

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

Covariance Matrix Adaptation

Rank-One Update

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

While not terminate

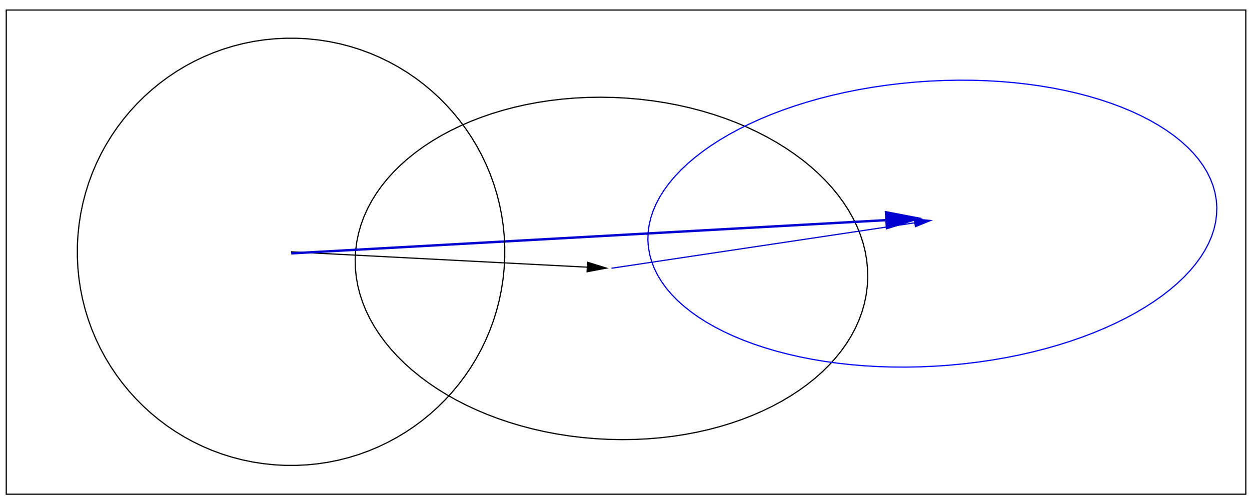
$$\boldsymbol{x}_i = \boldsymbol{m} + \sigma\, \boldsymbol{y}_i, \qquad \boldsymbol{y}_i \sim \mathcal{N}_i(\mathbf{0}, \mathbf{C}),$$

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

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

Evolution Path

Conceptually, the evolution path is the search path the strategy takes over a number of iteration steps. It can be expressed as a sum of consecutive *steps* of the mean $\boldsymbol{m}$.

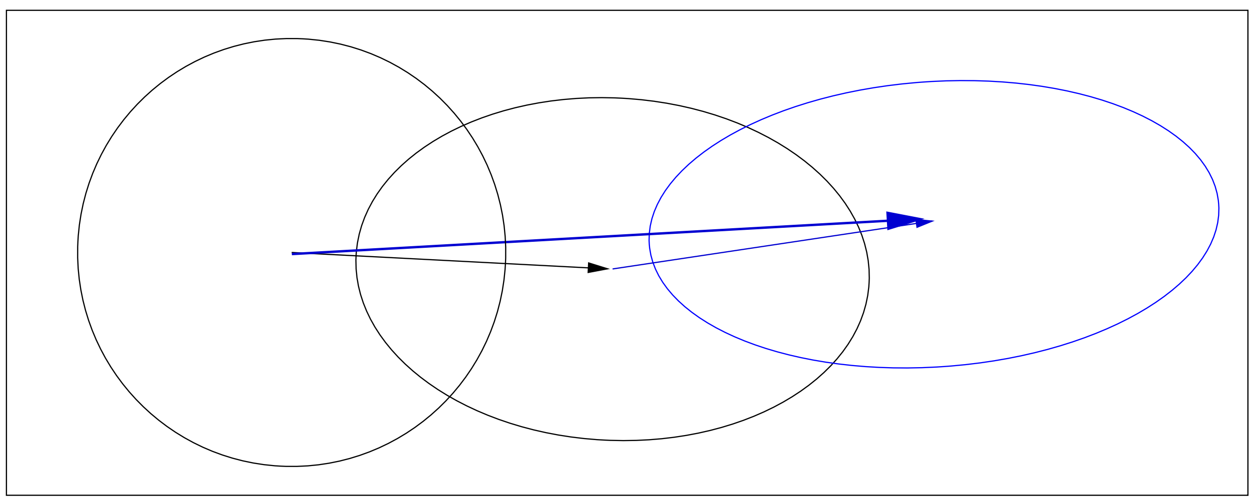


An exponentially weighted sum of steps $\boldsymbol{y}_w$ is used

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

Evolution Path

Conceptually, the evolution path is the search path the strategy takes over a number of iteration 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

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

The recursive construction of the evolution path (cumulation):

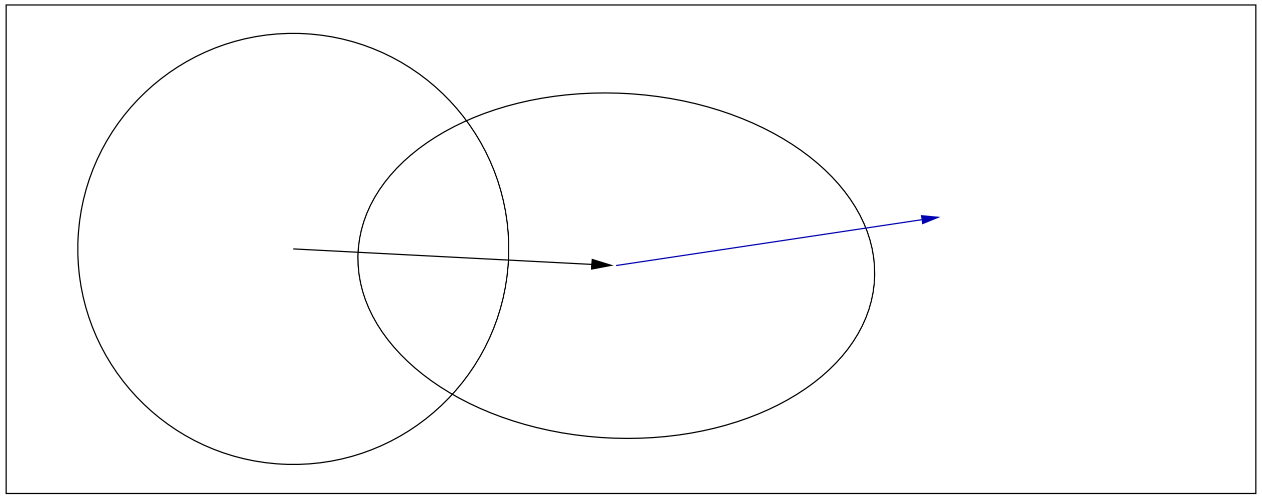
$$\boldsymbol{p}_c \leftarrow \underbrace{(1-c_c)}_{\text{decay factor}} \boldsymbol{p}_c + \underbrace{\sqrt{1-(1-c_c)^2}\sqrt{\mu_w}}_{\text{normalization factor}} \underbrace{\boldsymbol{y}_w}_{\text{input } = \frac{\boldsymbol{m}-\boldsymbol{m}_{\text{old}}}{\sigma}}$$

where $\mu_w = \frac{1}{\sum {w_i}^2}$, $c_c \ll 1$. History information is accumulated in the evolution path.

Cumulation

Utilizing the Evolution Path

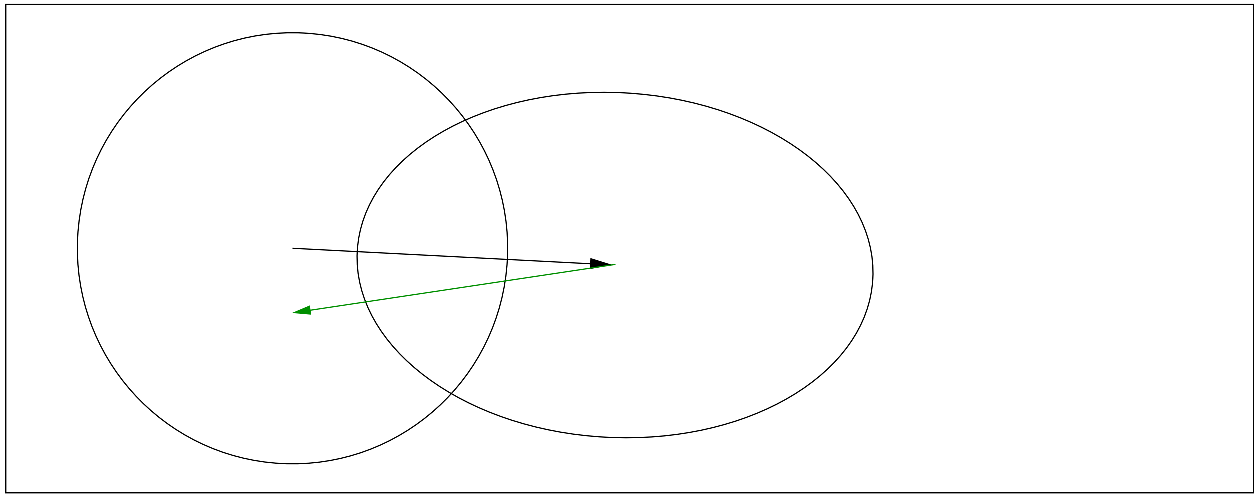
We used $\boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$ for updating $\mathbf{C}$. Because $\boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}} = -\boldsymbol{y}_w(-\boldsymbol{y}_w)^{\mathrm{T}}$ the sign of $\boldsymbol{y}_w$ is lost.



Cumulation

Utilizing the Evolution Path

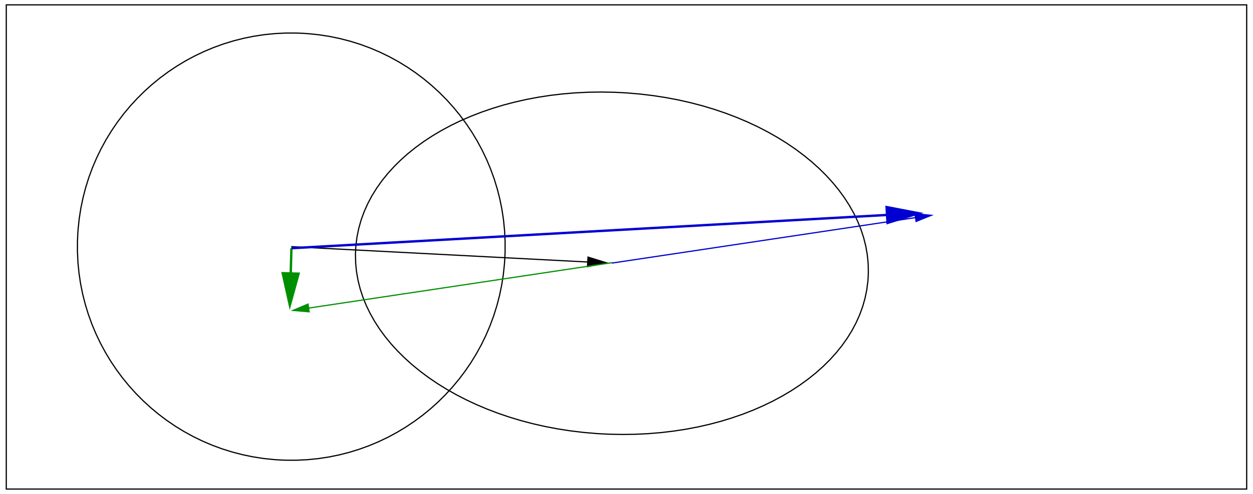
We used $\boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}}$ for updating **C**. Because $\boldsymbol{y}_w \boldsymbol{y}_w^{\mathrm{T}} = -\boldsymbol{y}_w(-\boldsymbol{y}_w)^{\mathrm{T}}$ the sign of $\boldsymbol{y}_w$ is lost.



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

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 $\mathcal{O}(n)$.[3]

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

[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

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 iteration step.

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 iteration 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.

Rank-μ Update

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

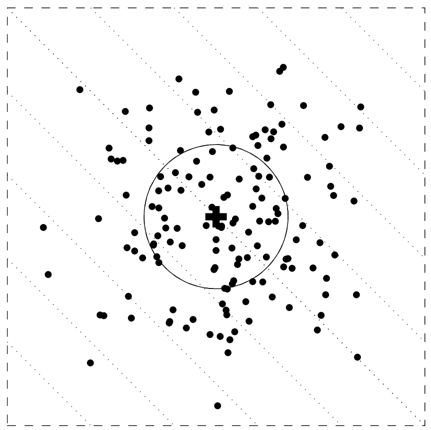
The rank-μ update extends the update rule for large population sizes λ using $\mu > 1$ vectors to update $\mathbf{C}$ at each iteration step. The matrix

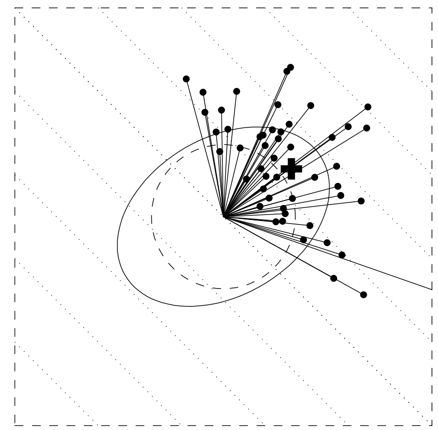
$$\mathbf{C}_\mu = \sum_{i=1}^{\mu} w_i\, \boldsymbol{y}_{i:\lambda} \boldsymbol{y}_{i:\lambda}^{\mathrm{T}}$$

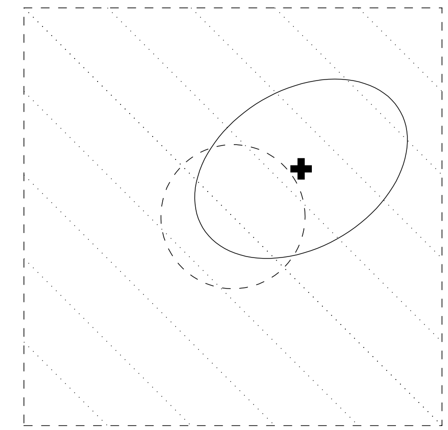
computes a weighted mean of the outer products of the best μ steps and has rank $\min(\mu, n)$ with probability one.
The rank-μ update then reads

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

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







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

$$C_\mu = \frac{1}{\mu} \sum y_{i:\lambda} y_{i:\lambda}^{T}$$
$$C \leftarrow (1-1) \times C + 1 \times C_\mu$$

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

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

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

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 iterations roughly from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ [4]

  given $\mu_w \propto \lambda \propto n$

Therefore the rank-μ update is the primary mechanism whenever a large population size is used

say $\lambda \geq 3\,n + 10$

[4] 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 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 iterations roughly from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ [4]

  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 $\mathcal{O}(n)$.

[4] 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 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 iterations roughly from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ [4]

  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

[4] 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

Summary of Equations

The Covariance Matrix Adaptation Evolution Strategy

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

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

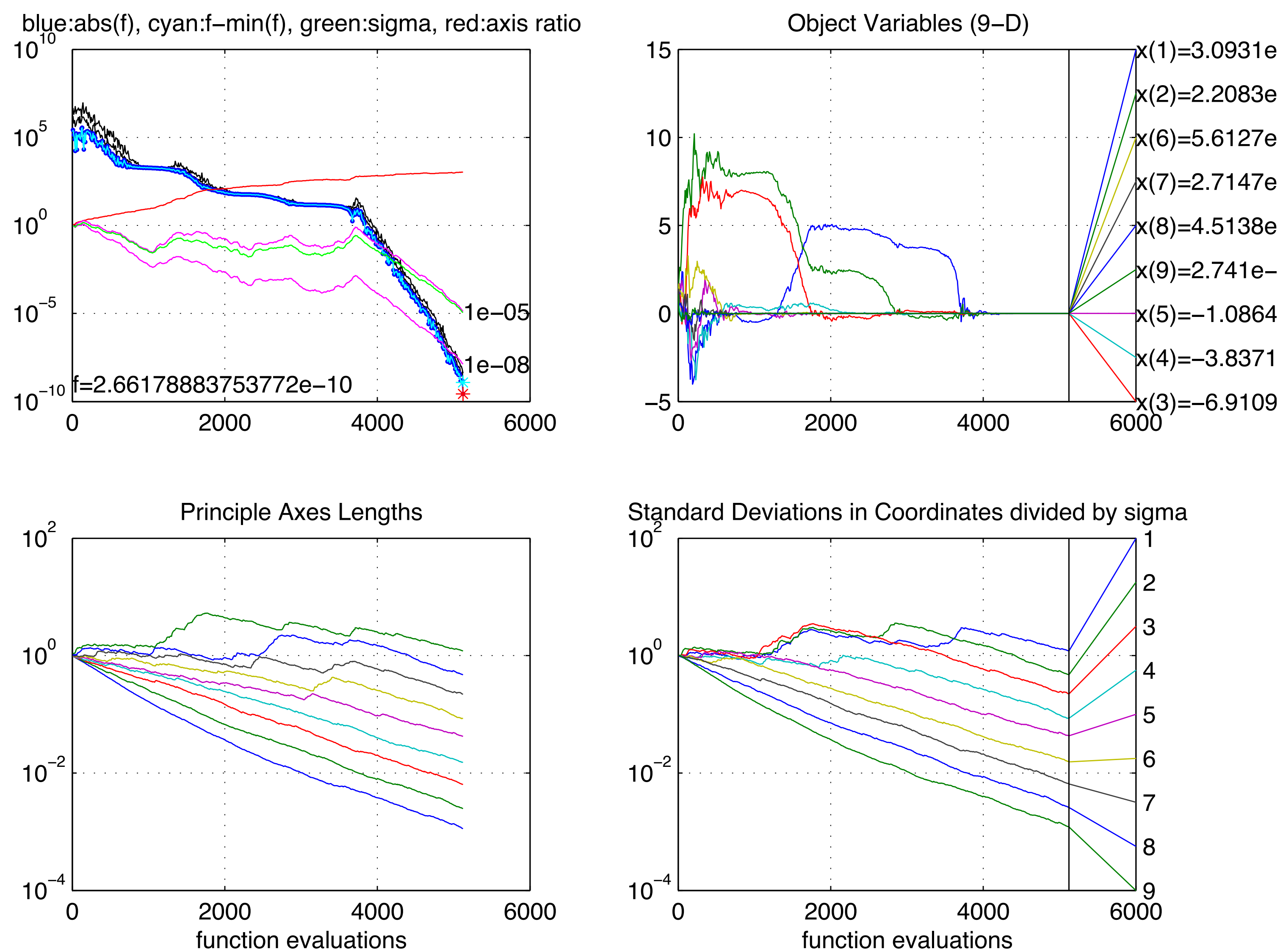
- lines of equal density align with lines of equal fitness

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

in a stochastic sense

Experimentum Crucis (1)

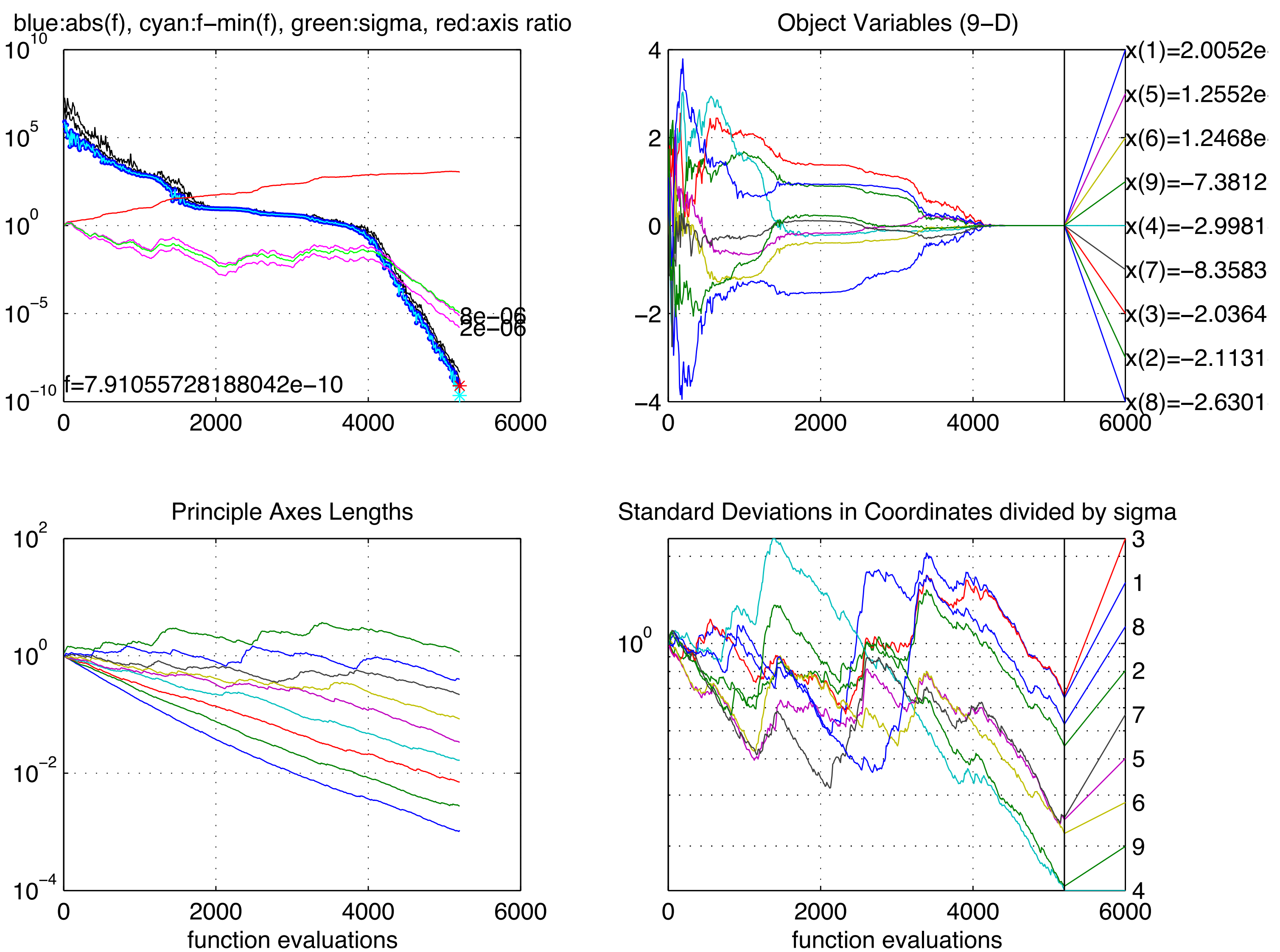
f convex quadratic, separable



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

Experimentum Crucis (2)

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

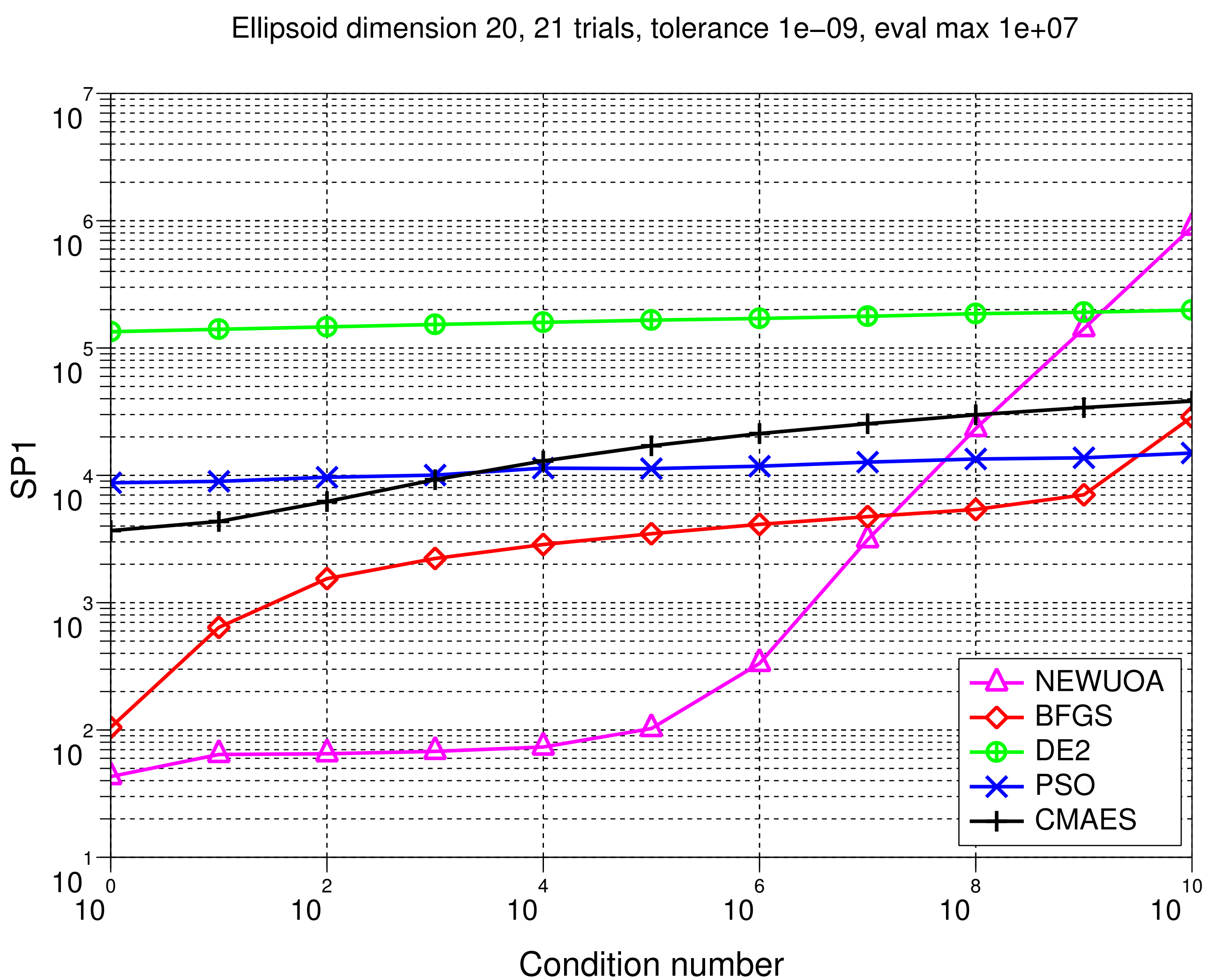


$\mathbf{C} \propto \boldsymbol{H}^{-1}$ for all $g, \mathbf{H}$

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

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)

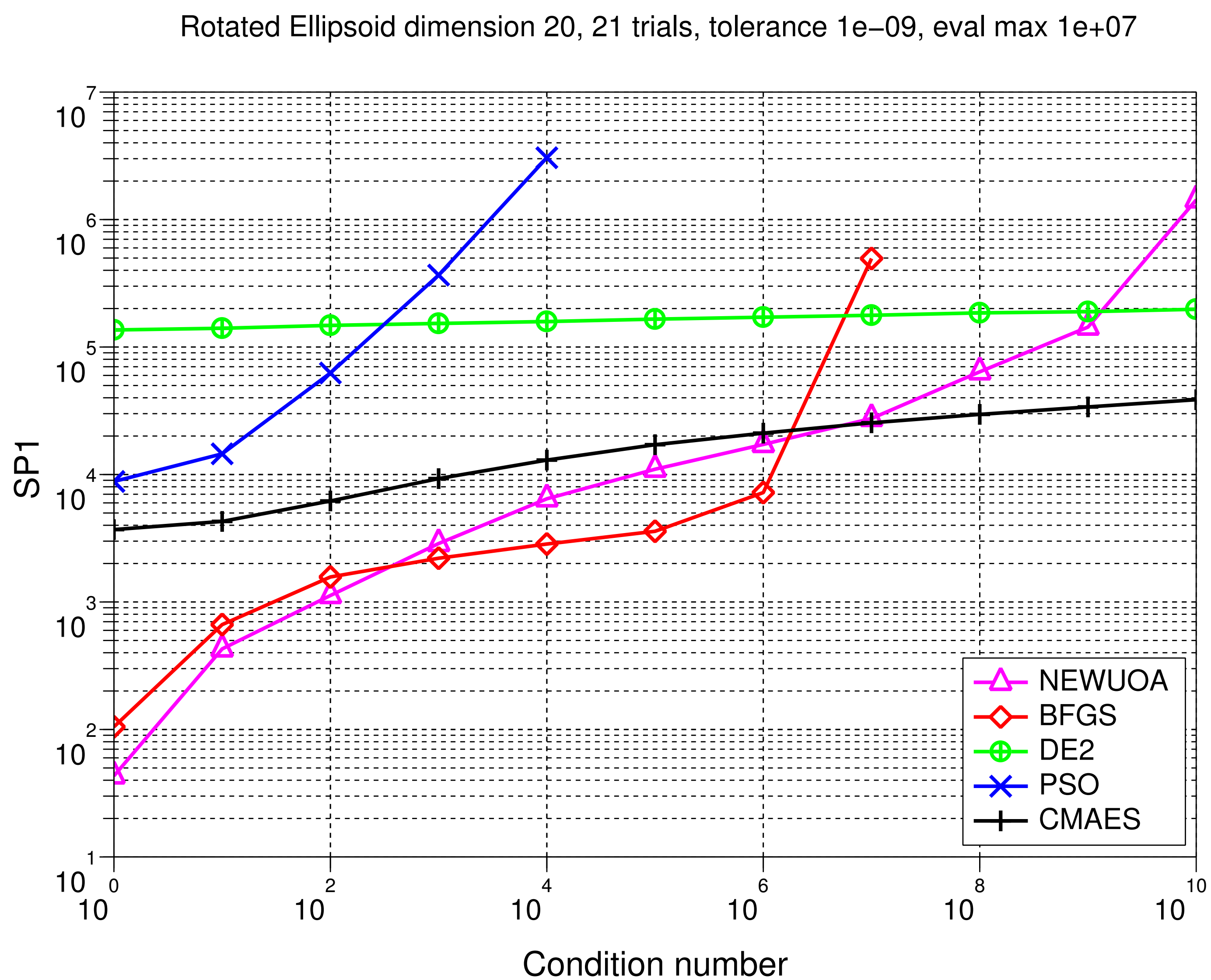
$f(\boldsymbol{x}) = g(\boldsymbol{x}^{\mathrm{T}}\mathbf{H}\boldsymbol{x})$ with
$\boldsymbol{H}$ diagonal
g identity (for BFGS and NEWUOA)
g any order-preserving = strictly increasing function (for all other)

SP1 = average number of objective function evaluations[5] to reach the target function value of $g^{-1}(10^{-9})$

[5] 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)

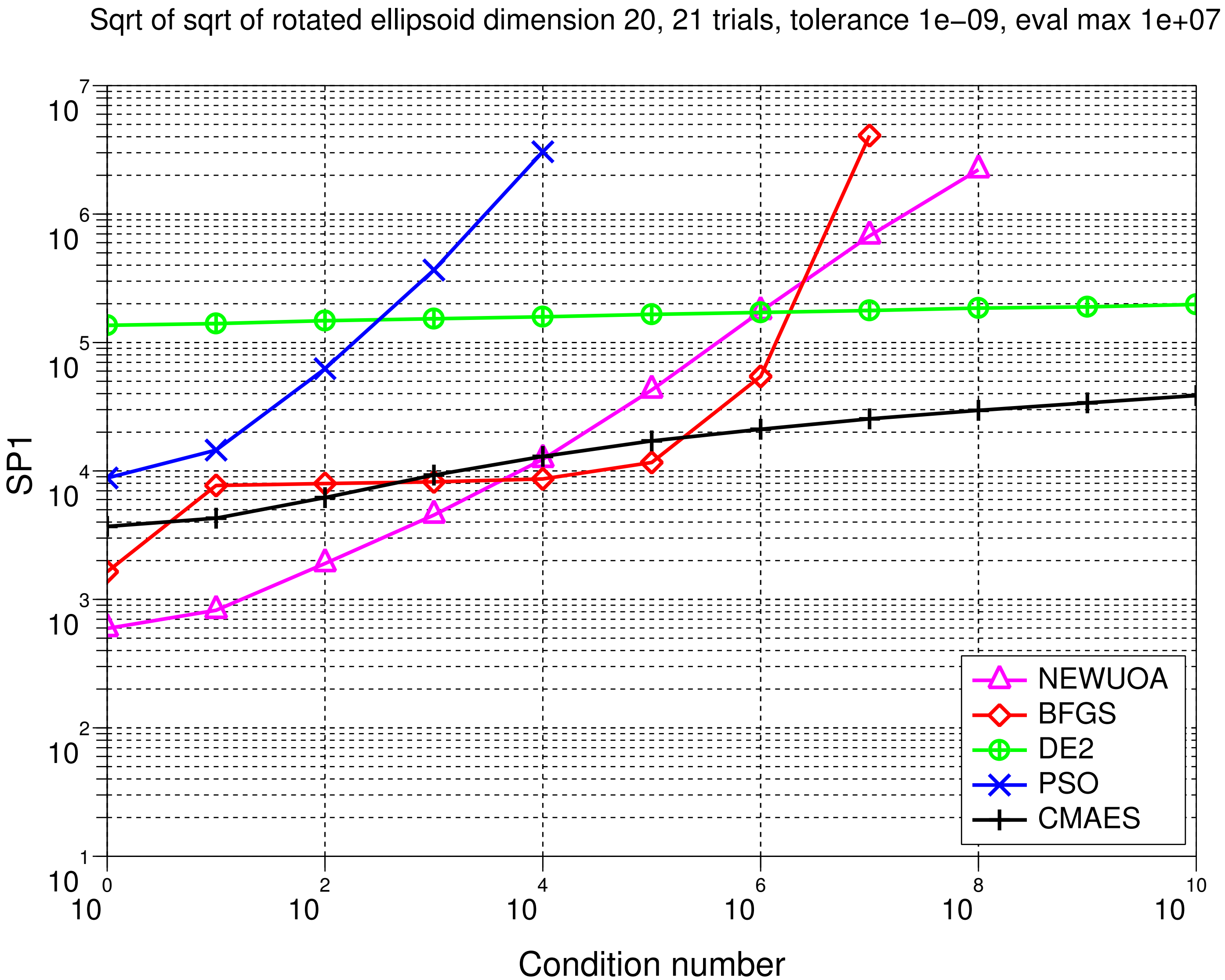
$f(\boldsymbol{x}) = g(\boldsymbol{x}^{\mathrm{T}}\mathbf{H}\boldsymbol{x})$ with
$\boldsymbol{H}$ full
g identity (for BFGS and NEWUOA)
g any order-preserving = strictly increasing function (for all other)

SP1 = average number of objective function evaluations[6] to reach the target function value of $g^{-1}(10^{-9})$

[6] 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



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

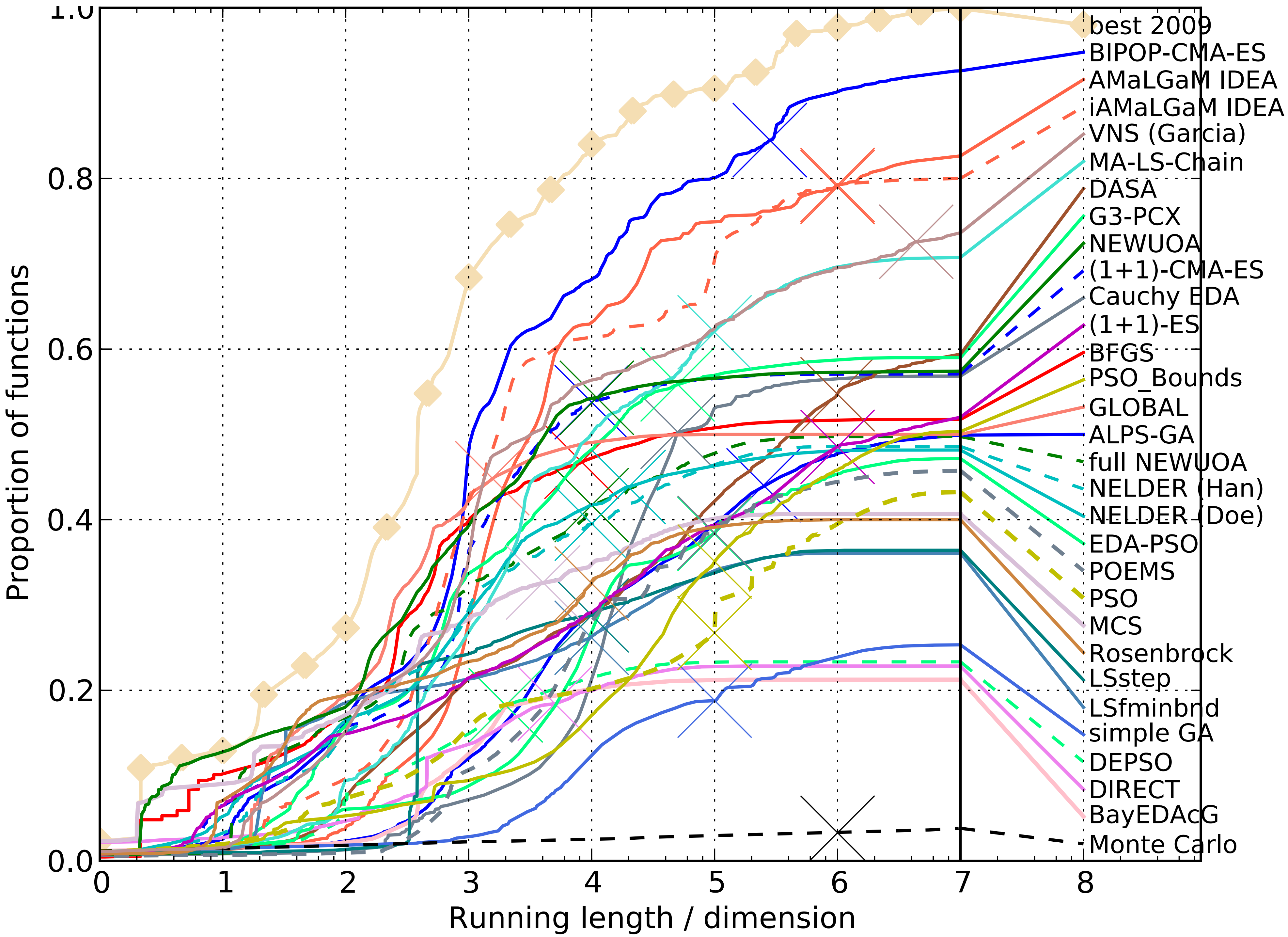
$f(\boldsymbol{x}) = g(\boldsymbol{x}^{\mathrm{T}}\mathbf{H}\boldsymbol{x})$ with
$\boldsymbol{H}$ full
$g : x \mapsto x^{1/4}$ (for BFGS and NEWUOA)
g any order-preserving = strictly increasing function (for all other)

SP1 = average number of objective function evaluations[7] to reach the target function value of $g^{-1}(10^{-9})$

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

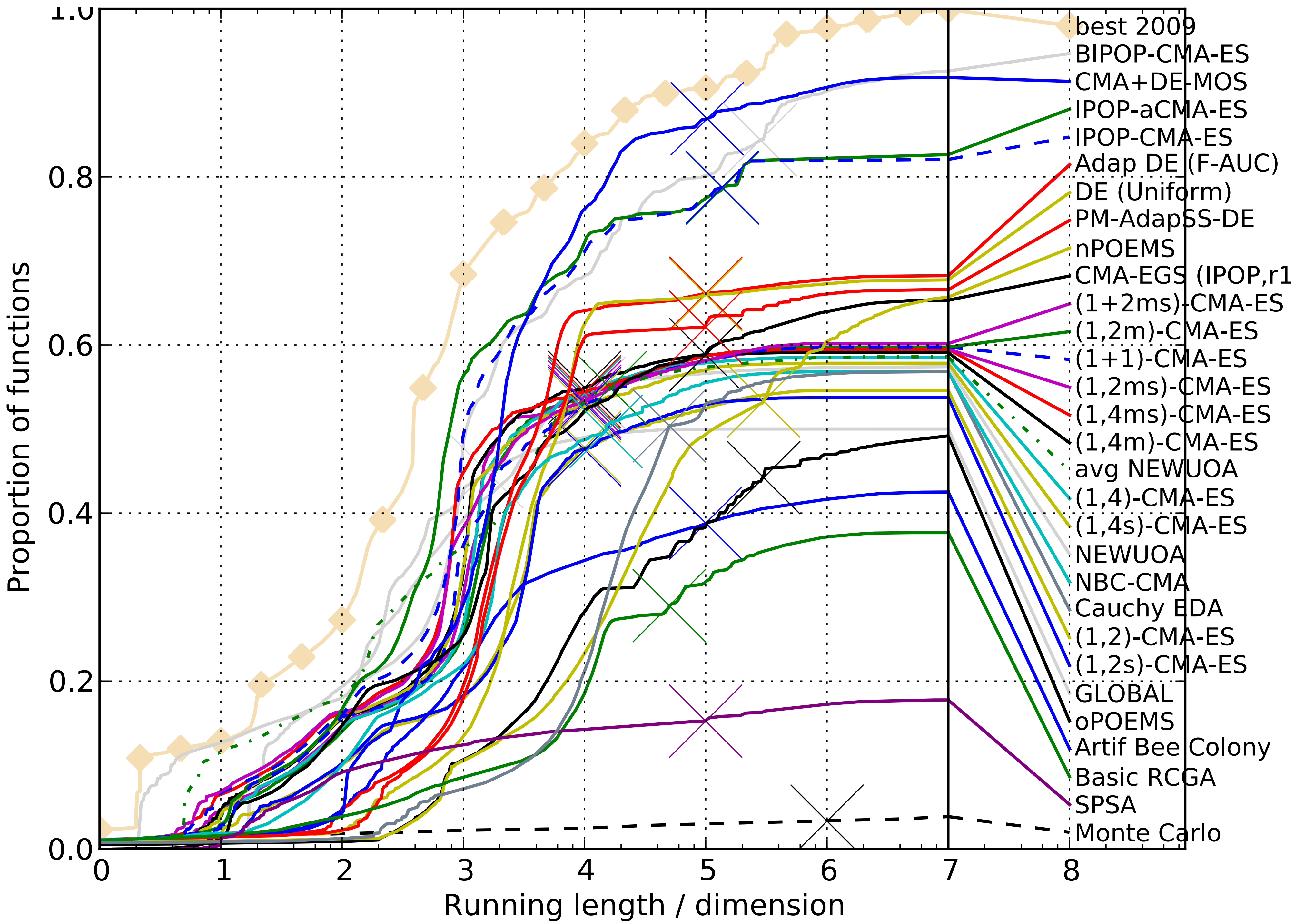
Comparison during BBOB at GECCO 2009

24 functions and 31 algorithms in 20-D



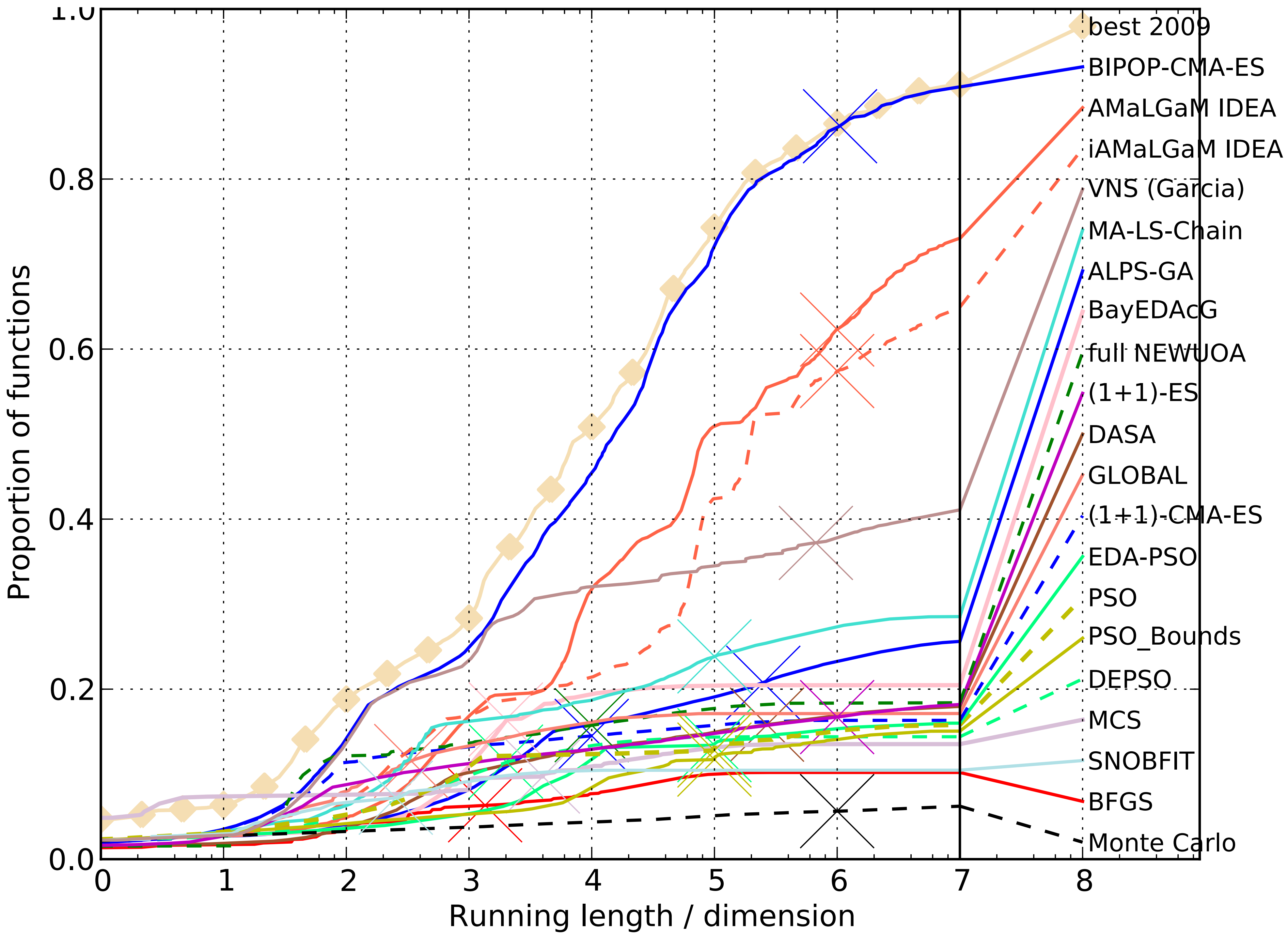
Comparison during BBOB at GECCO 2010

24 functions and 20+ algorithms in 20-D



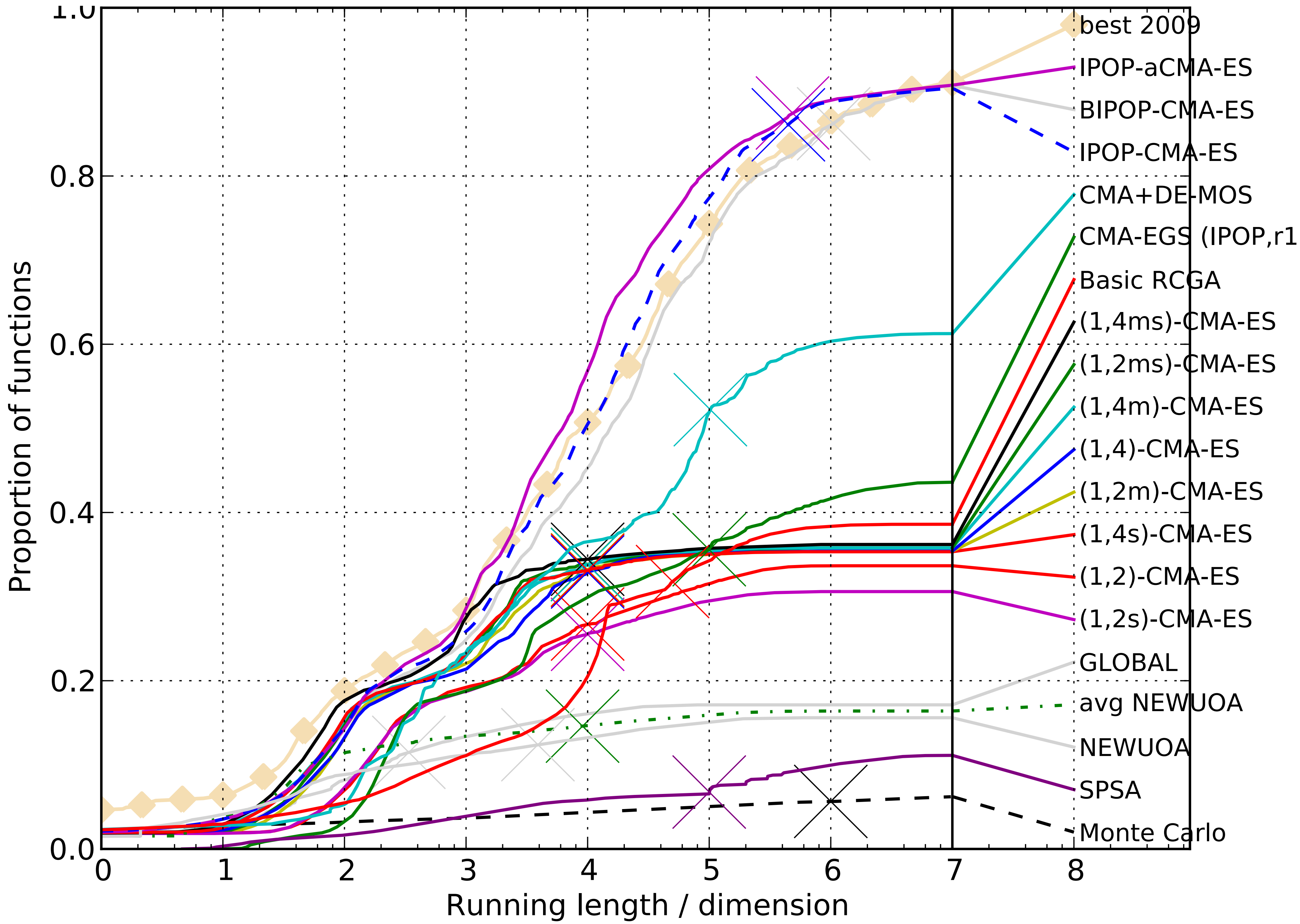
Comparison during BBOB at GECCO 2009

30 noisy functions and 20 algorithms in 20-D



Comparison during BBOB at GECCO 2010

30 noisy functions and 10+ algorithms in 20-D



The Continuous Search Problem

Difficulties of a non-linear optimization problem are

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

Approach: population based stochastic search, coordinate system independent and with second order estimations (covariances)

Main Features 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

   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

   $\mathbf{C} \propto \boldsymbol{H}^{-1} \iff$ adapts a variable metric

   $\iff$ new (rotated) problem representation

   $\implies f(\boldsymbol{x}) = g(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{x})$ reduces to $g(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x})$

Limitations

of CMA Evolution Strategies

- internal CPU-time: $10^{-8}n^2$ seconds per function evaluation on a 2GHz PC, tweaks are available

  100 000 f-evaluations in 1000-D take 1/4 hours
  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