A comparative introduction to two optimization classics: the Nelder-Mead and the CMA-ES algorithms

Rodolphe Le Riche[1,2]

[1] Ecole des Mines de Saint Etienne, France
[2] CNRS LIMOS France

March 2018
MEXICO network research school on sensitivity analysis, metamodeling and optimization of complex models
La Rochelle, France

Foreword

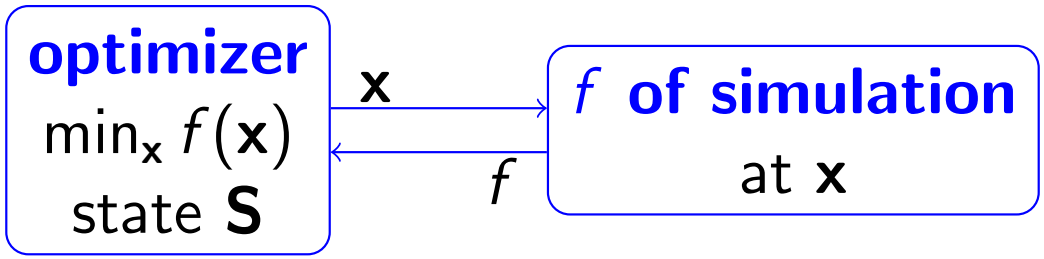
- In this 1h30 course + 1h30 hands-on class, we address unconstrained optimization with continuous variables

$$\text{find one or many} \quad \mathbf{x}^* \equiv \arg \min_{\mathbf{x} \in \mathcal{S} \in \mathbb{R}^d} f(\mathbf{x}) \tag{1}$$

  where $\mathcal{S}$ is typically the compact $[\mathbf{x}^{\text{LB}}, \mathbf{x}^{\text{UB}}] \in \mathbb{R}^d$.
- We cover the principles (the optimization "engines") of two algorithms, Nelder-Mead and CMA-ES. These algorithms have the same goal, but one is deterministic and one is stochastic.
- Both algorithms do not use metamodels: because they do not model f, they are insensitive to a monotonic transformation of $f()$ (e.g., they perform the same on $f()$, $f^3()$, $\exp(f())$); they do not build any approximation of $f()$ valid in the entire $\mathcal{S}$ (like EGO does).

Optimizers are iterative algorithms

We are going to approximate the solution(s) to Problem (1) using iterative optimization algorithms (optimizers) that exchange information with the objective function code (that wraps the often costly simulation).



or in a state-space representation,

$$\mathbf{x}^{t+1} = \phi\left(\mathbf{S}^{t+1}\right) \tag{2}$$

$$\mathbf{S}^{t+1} = \psi\left(\mathbf{S}^{t}, (\mathbf{x}^{t}, f(\mathbf{x}^{t}))\right) \tag{3}$$

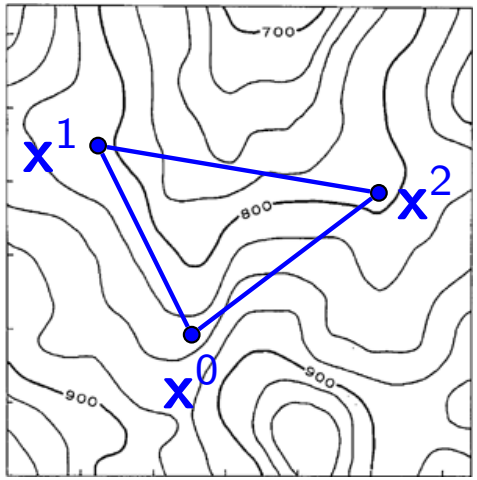
The optimizer is defined by $\mathbf{S}$ (state vector), ϕ (next iterate equation) and ψ (state equation).

Content of the talk

- A pattern search method: the Nelder-Mead algorithm
- A stochastic optimizer: CMA-ES

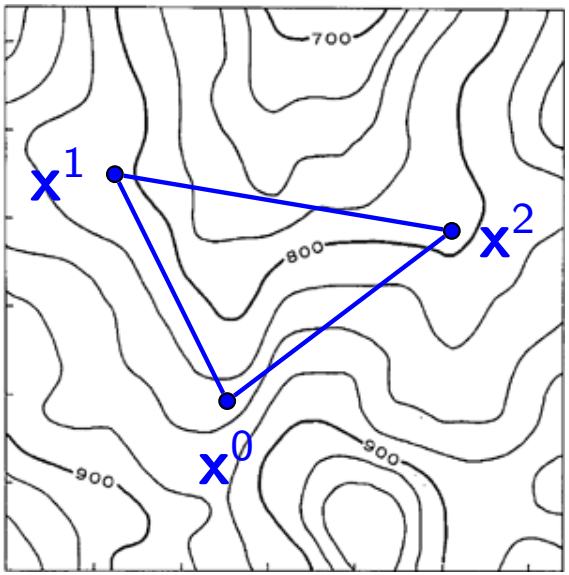
The Nelder-Mead algorithm

- 1965. A great compromise between simplicity and efficiency: [14] is cited about 27000 times on Google Scholar.
- It belongs to the class of pattern search methods [2], i.e., methods that propose new points to evaluate based on the deformation of a geometrical pattern.
- Sometimes also called "simplex" (not to be mistaken with that of linear programming) because the pattern is made of a simplex: $d + 1$ points in a d-dimensional space



Simplex properties (1)

Notation: $f(\mathbf{x}^0) \leq f(\mathbf{x}^1) \leq \ldots \leq f(\mathbf{x}^d)$



$(\mathbf{x}^1 - \mathbf{x}^0, \mathbf{x}^2 - \mathbf{x}^0)$ is a basis of $\mathbb{R}^d$.

$(\mathbf{x}^1 - \mathbf{x}^0, \mathbf{x}^2 - \mathbf{x}^0, -(\mathbf{x}^1 - \mathbf{x}^0), -(\mathbf{x}^2 - \mathbf{x}^0))$ a positive set ($-\nabla f(\mathbf{x}^0)$ can be approximated as a positive linear combination of the vectors, a condition for convergence with shrink steps, see later).

$d + 1$ is the minimum number of points to construct a linear interpolation.

Simplex properties (2)

$d+1$ is the minimum number of points to construct a linear interpolation:

$$\widehat{f}(\mathbf{x}^i) = \alpha_0 + \alpha_1 x_1^i + \ldots + \alpha_d x_d^i = f(\mathbf{x}^i) \quad , \quad i = 0, d$$

$$\begin{bmatrix} 1 & x_1^0 & \ldots & x_d^0 \\ 1 & x_1^1 & \ldots & x_d^1 \\ & & \ldots & \\ 1 & x_1^d & \ldots & x_d^d \end{bmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \\ \alpha_d \end{pmatrix} = \begin{pmatrix} f(\mathbf{x}^0) \\ f(\mathbf{x}^1) \\ \\ f(\mathbf{x}^d) \end{pmatrix}$$

$$\Leftrightarrow \begin{bmatrix} 1 & x_1^0 & \ldots & x_d^0 \\ 0 & x_1^1 - x_1^0 & \ldots & x_d^1 - x_d^0 \\ & & \ldots & \\ 0 & x_1^d - x_1^0 & \ldots & x_d^d - x_d^0 \end{bmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \\ \alpha_d \end{pmatrix} = \begin{pmatrix} f(\mathbf{x}^0) \\ f(\mathbf{x}^1) - f(\mathbf{x}^0) \\ \\ f(\mathbf{x}^d) - f(\mathbf{x}^0) \end{pmatrix}$$

which has a unique solution iff lower right sub-matrix $\mathbf{L}$ is invertible, or better, well-conditioned.

Simplex properties (2)

$d+1$ is the minimum number of points to construct a linear interpolation:

$$\widehat{f}(\mathbf{x}^i) = \alpha_0 + \alpha_1 x_1^i + \ldots + \alpha_d x_d^i = f(\mathbf{x}^i) \quad , \quad i = 0, d$$

$$\begin{bmatrix} 1 & x_1^0 & \ldots & x_d^0 \\ 1 & x_1^1 & \ldots & x_d^1 \\ & & \ldots & \\ 1 & x_1^d & \ldots & x_d^d \end{bmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \\ \alpha_d \end{pmatrix} = \begin{pmatrix} f(\mathbf{x}^0) \\ f(\mathbf{x}^1) \\ \\ f(\mathbf{x}^d) \end{pmatrix}$$

$$\Leftrightarrow \begin{bmatrix} 1 & x_1^0 & \ldots & x_d^0 \\ 0 & x_1^1 - x_1^0 & \ldots & x_d^1 - x_d^0 \\ & & \ldots & \\ 0 & x_1^d - x_1^0 & \ldots & x_d^d - x_d^0 \end{bmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \\ \alpha_d \end{pmatrix} = \begin{pmatrix} f(\mathbf{x}^0) \\ f(\mathbf{x}^1) - f(\mathbf{x}^0) \\ \\ f(\mathbf{x}^d) - f(\mathbf{x}^0) \end{pmatrix}$$

which has a unique solution iff lower right sub-matrix **L** is invertible, or better, well-conditioned.

Simplex properties (3)

- The diameter of a simplex $\left\{\mathbf{x}^0, \ldots, \mathbf{x}^d\right\}$ is its largest edge,

$$\text{diam}(\{\mathbf{x}^0, \ldots, \mathbf{x}^d\}) = \max_{0 \leq i < j \leq d} \left\|\mathbf{x}^i - \mathbf{x}^j\right\|$$

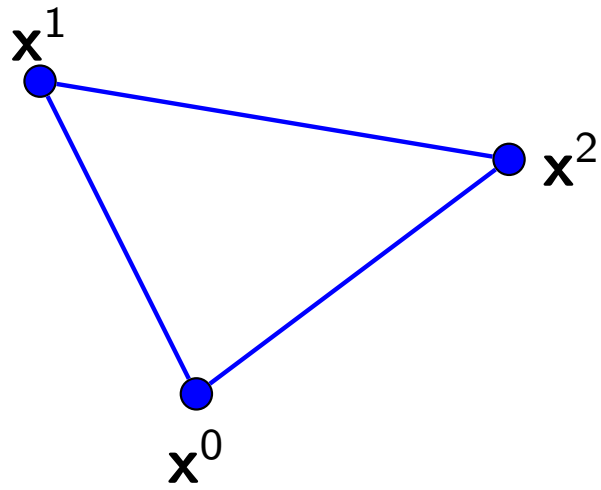
- A measure of how well the edges of a simplex span $\mathbb{R}^d$ is its normalized volume (noting $s \equiv \text{diam}(\{\mathbf{x}^0, \ldots, \mathbf{x}^d\})$),

$$\overline{\text{vol}}(\{\mathbf{x}^0, \ldots, \mathbf{x}^d\}) = \text{vol}\left(\frac{\mathbf{x}^0}{s}, \ldots, \frac{\mathbf{x}^0}{s}\right) = \frac{\det(\mathbf{L})}{d! s^d},$$

  which is a scale invariant measure of the volume of the simplex.

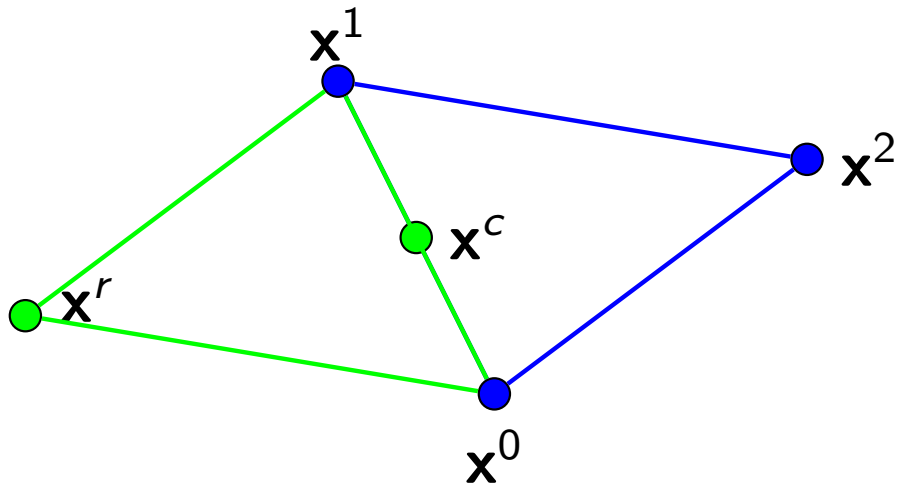
A Nelder-Mead iteration

New points are generated through simple geometrical transformations. Remember $f(\mathbf{x}^0) \leq f(\mathbf{x}^1) \leq \ldots \leq f(\mathbf{x}^d)$.



A Nelder-Mead iteration

Reflection: take a step from the worst to its symmetric w.r.t. the cendroid of the d better points $\approx$ a descent step, whose size and direction come from the current simplex,

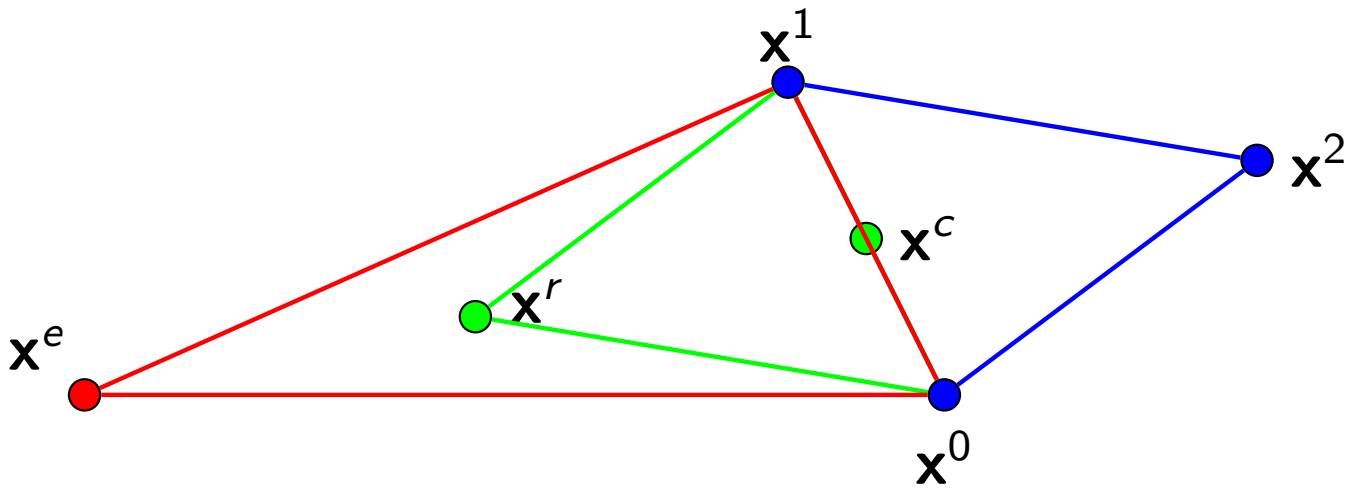


$$\mathbf{x}^r = \mathbf{x}^c + \delta_r(\mathbf{x}^c - \mathbf{x}^d) \text{ where } \delta_r = 1 \text{ and } \mathbf{x}^c = \frac{1}{d}\sum_{i=0}^{d-1} \mathbf{x}^i$$

calculate $f(\mathbf{x}^r)$ (cost 1 call to f).
If $f(\mathbf{x}^0) \leq f(\mathbf{x}^r) < f(\mathbf{x}^{d-1})$, $\mathbf{x}^d \leftarrow \mathbf{x}^r$, end iteration.

A Nelder-Mead iteration

Expansion: if $f(\mathbf{x}^r) < f(\mathbf{x}^0)$, progress was made, increase step size.

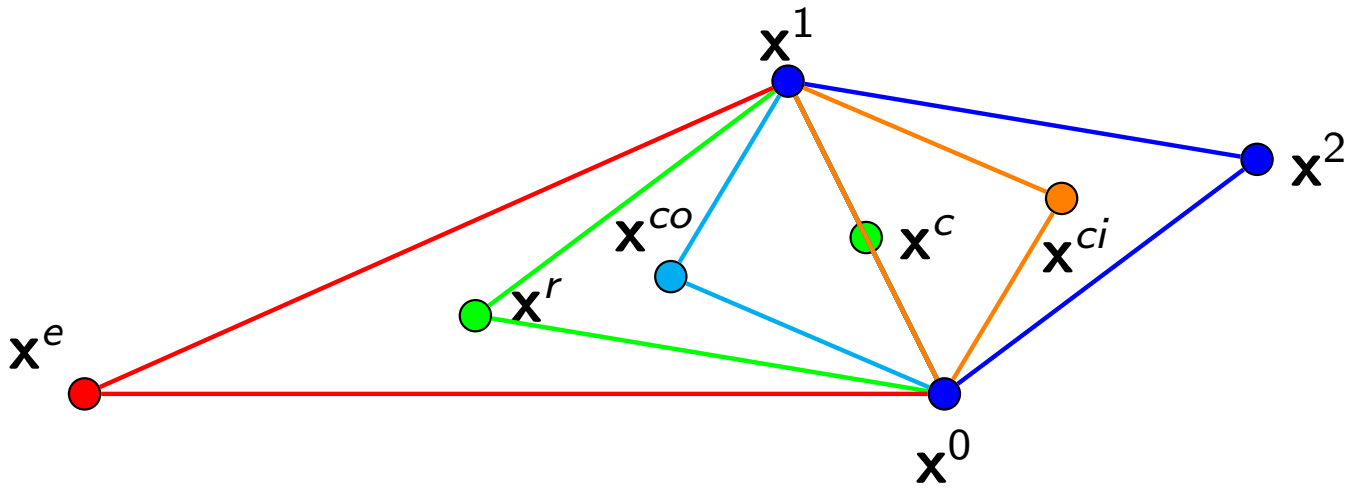


$$\mathbf{x}^e = \mathbf{x}^c + \delta_e(\mathbf{x}^c - \mathbf{x}^d) \text{ where } \delta_e = 2$$

If $f(\mathbf{x}^e) \leq f(\mathbf{x}^r)$, $\mathbf{x}^d \leftarrow \mathbf{x}^e$, end iteration.
Otherwise, $\mathbf{x}^d \leftarrow \mathbf{x}^r$, end iteration.

A Nelder-Mead iteration

Contraction: if $f(\mathbf{x}^r) \geq f(\mathbf{x}^{d-1})$, little progress, reduce step size: **Outside contraction** if $f(\mathbf{x}^r) < f(\mathbf{x}^d)$; **Inside contraction** if $f(\mathbf{x}^r) \geq f(\mathbf{x}^d)$.

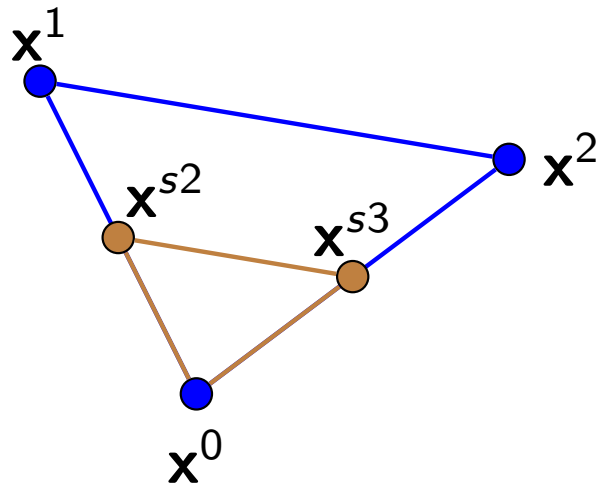


$$\mathbf{x}^{c\{i,o\}} = \mathbf{x}^c + \delta_{c\{i,o\}}(\mathbf{x}^c - \mathbf{x}^d) \text{ where } \delta_{ci} = -1/2 \text{ and } \delta_{co} = 1/2$$

If $f(\mathbf{x}^{co}) \leq f(\mathbf{x}^r)$, $\mathbf{x}^d \leftarrow \mathbf{x}^{co}$; Else $\mathbf{x}^d \leftarrow \mathbf{x}^r$; End iteration.
If $f(\mathbf{x}^{ci}) < f(\mathbf{x}^d)$, $\mathbf{x}^d \leftarrow \mathbf{x}^{ci}$ and end iteration; Else shrink.

A Nelder-Mead iteration

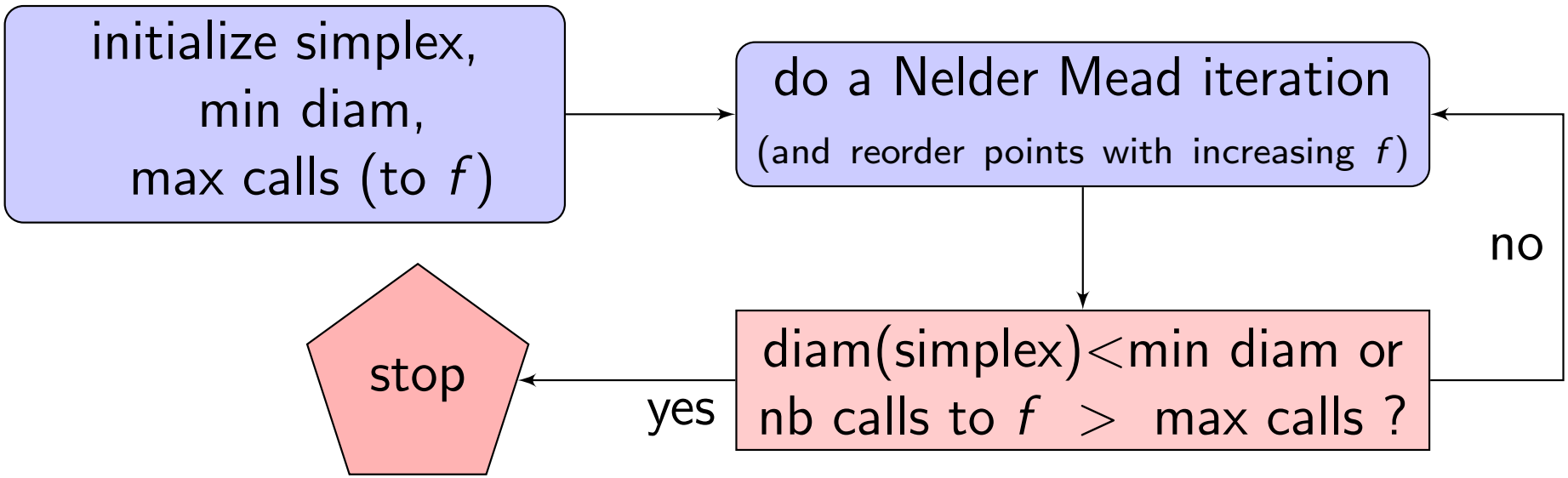
Shrink: if $f(\mathbf{x}^{ci}) > f(\mathbf{x}^{d})$, no progress, reduce simplex size (direction more influenced by $\mathbf{x}^0$ and smaller step size).



$$\mathbf{x}^{s\{1,\dots,d\}} = \mathbf{x}^0 + \gamma_s(\mathbf{x}^{\{1,\dots,d\}} - \mathbf{x}^0) \text{ where } \gamma_s = 1/2$$

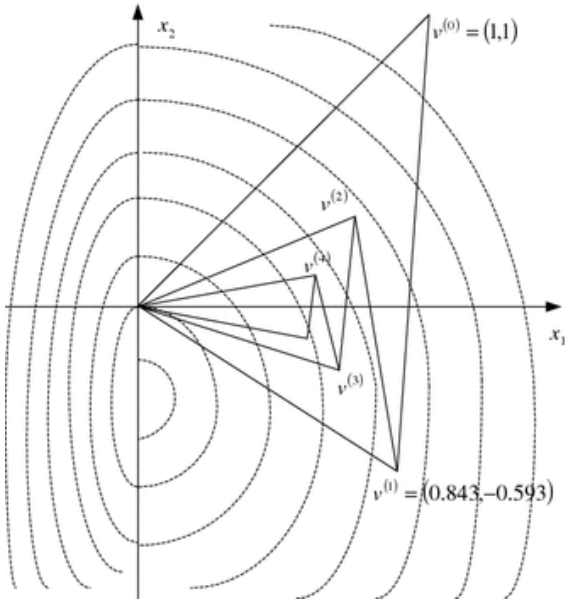
Evaluate f at $\mathbf{x}^{s1}, \dots, \mathbf{x}^{sd}$ (cost d), end iteration.

Flow chart of the Nelder-Mead algorithm



Properties of the Nelder-Mead algorithm

- An iteration may cost 1 call to f if end at reflection, or 2 calls (reflection + expansion; reflection + contraction) or $d + 2$ calls (reflection + expansion + shrink).
- No shrink step is ever taken on a convex function (proof in [2] p. 148).
- It always converges to a stationary point (where $\nabla f(\mathbf{x}) = 0$) when $d = 1$ (proof in [8]) and f is continuously differentiable.
- However it may converge to nonstationary points:



Nelder-Mead operates a series of contractions and converges to a nonstationary point on McKinnon functions which are convex and continuously differentiable (plot from [12]).

Practicalities

- **Restarts**: after each iteration, add a restart test, if $\overline{\text{vol}} < \varepsilon$ (say 10^{-4}), restart the simplex at $\mathbf{x}^0$.
- **Bound constraints handling**, $\mathbf{x}^{\text{LB}} \leq \mathbf{x} \leq \mathbf{x}^{\text{UB}}$: the solution used in the `nlopt` software ([6], a `R` version exists) is to project points onto the constraints whenever a Nelder Mead transformation creates out-of-bounds points. This may induce a collapse of the simplex onto a subspace and prevent convergence.
- Works often well when $d \leq 10$ even on discontinuous, moderately multi-modal, moderately noisy functions. Nelder-Mead seems to contain a minimal set of operations to rapidly adapt to f local curvatures.

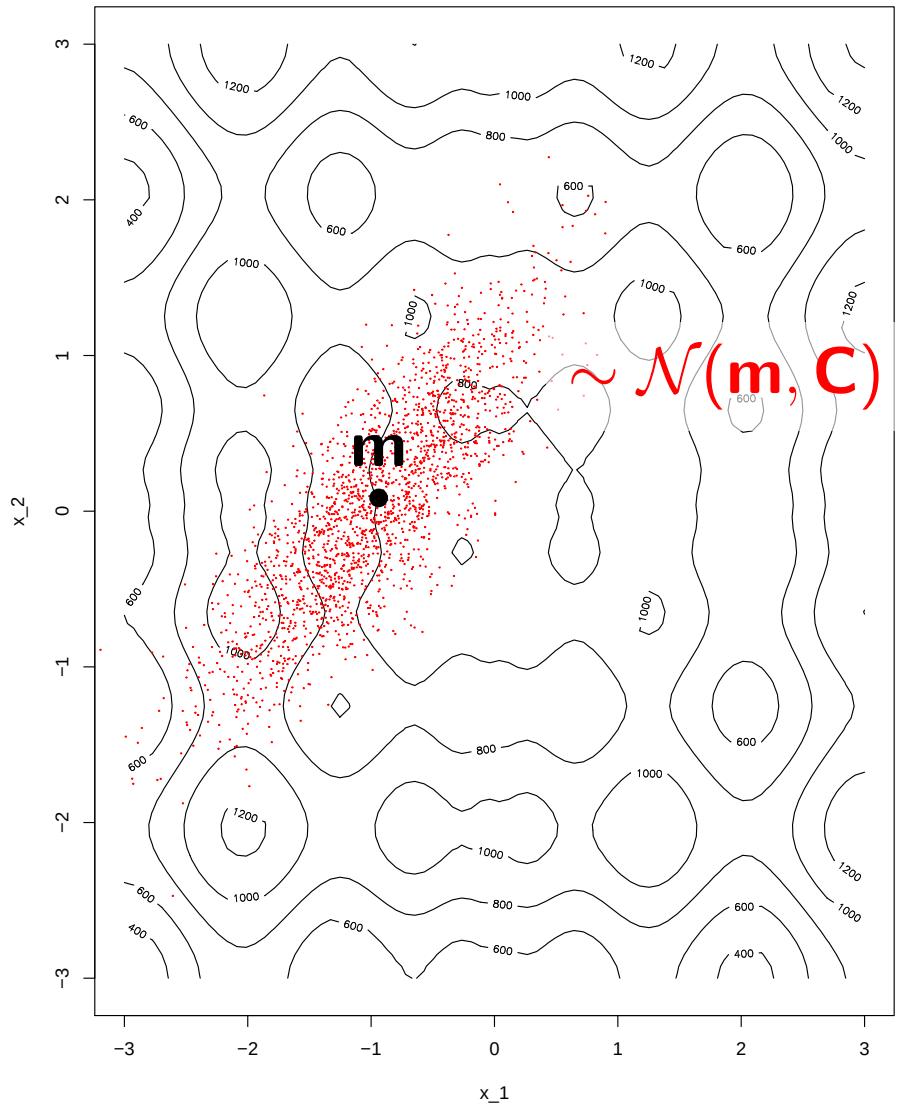
Converging extensions to the Nelder-Mead algorithm

- Do a 180° rotation of the simplex when the reflection does not work ([2, 16]): ensures that a descent direction can be found if the simplex is small enough (either $d^\top \nabla f(\mathbf{x}^0) < 0$ or $-d^\top \nabla f(\mathbf{x}^0) < 0$, unless at a stationary point).
- A version with space filling restarts and bound constraints, [11].
- . . . (393 articles with Nelder Mead in the title on Google Scholar).

The CMA-ES algorithm – Introduction (1)

A stochastic way of searching for good points. Replace the previous (simplex / simplex transformations) by (multivariate normal distribution / sampling and estimation).

$\mathcal{N}(\mathbf{m}, \mathbf{C})$ represent our belief about the location of the optimum $\mathbf{x}^\star$.



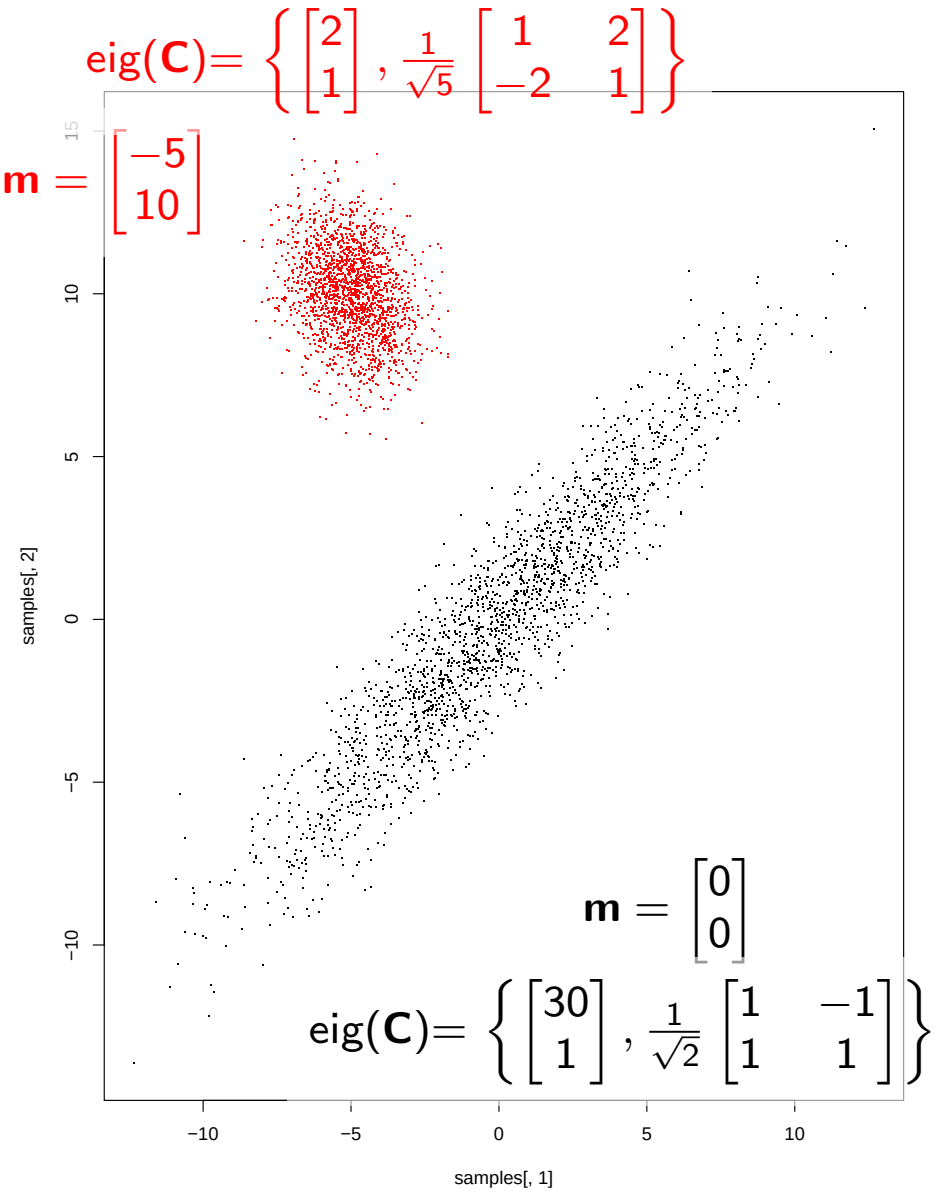
Multivariate normal

A d-dimensional Gaussian is fully described by its mean and covariance $\Rightarrow$ the central question is "how to adapt the mean and covariance of $\mathcal{N}$ to optimize efficiently?".

Sample generation:

$\mathbf{L} \leftarrow \text{cholesky}(\mathbf{C})$
$\mathbf{U} \sim \mathcal{N}(0, \mathbf{I})$ {rnorm in R}
$\mathbf{x} \leftarrow \mathbf{m} + \mathbf{L} \times \mathbf{U}$

because if $\text{Cov}(\mathbf{U}) = \mathbf{I}$ and $\mathbf{x} = \mathbf{Lu}$, then $\text{Cov}(\mathbf{X}) = \mathbf{L}\text{Cov}(\mathbf{U})\mathbf{L}^\top = \mathbf{L}\mathbf{L}^\top = \mathbf{C}$



The CMA-ES algorithm – Introduction (2)

- Originated in the evolutionary computation community with the work of Niko Hansen et al. in 1996 [4, 5], now turning more mathematical (cf. work by Anne Auger, [1, 15], Dimo Brockhoff, ...).
- A practical tutorial is [3].
- The presentation is restricted to the main concepts. To start with, ES-(1+1) with constant step.

The constant step size ES-(1+1)

Algorithm 1 constant circular step ES-(1+1)

Require: $\mathbf{m} \in \mathcal{S}$, σ^2, $t^{\max}$

calculate $f(\mathbf{m})$, $t \leftarrow 1$
while $t < t^{\max}$ **do**
  sample: $\mathbf{x} \sim \mathcal{N}(\mathbf{m}, \sigma^2 \mathbf{I})$
  calculate $f(\mathbf{x})$, $t \leftarrow t + 1$
  if $f(\mathbf{x}) < f(\mathbf{m})$ **then**
    $\mathbf{m} \leftarrow \mathbf{x}$
  end if
end while

The simplest stochastic optimizer: the covariance matrix is $\mathbf{C} = \sigma^2 \mathbf{I}$.

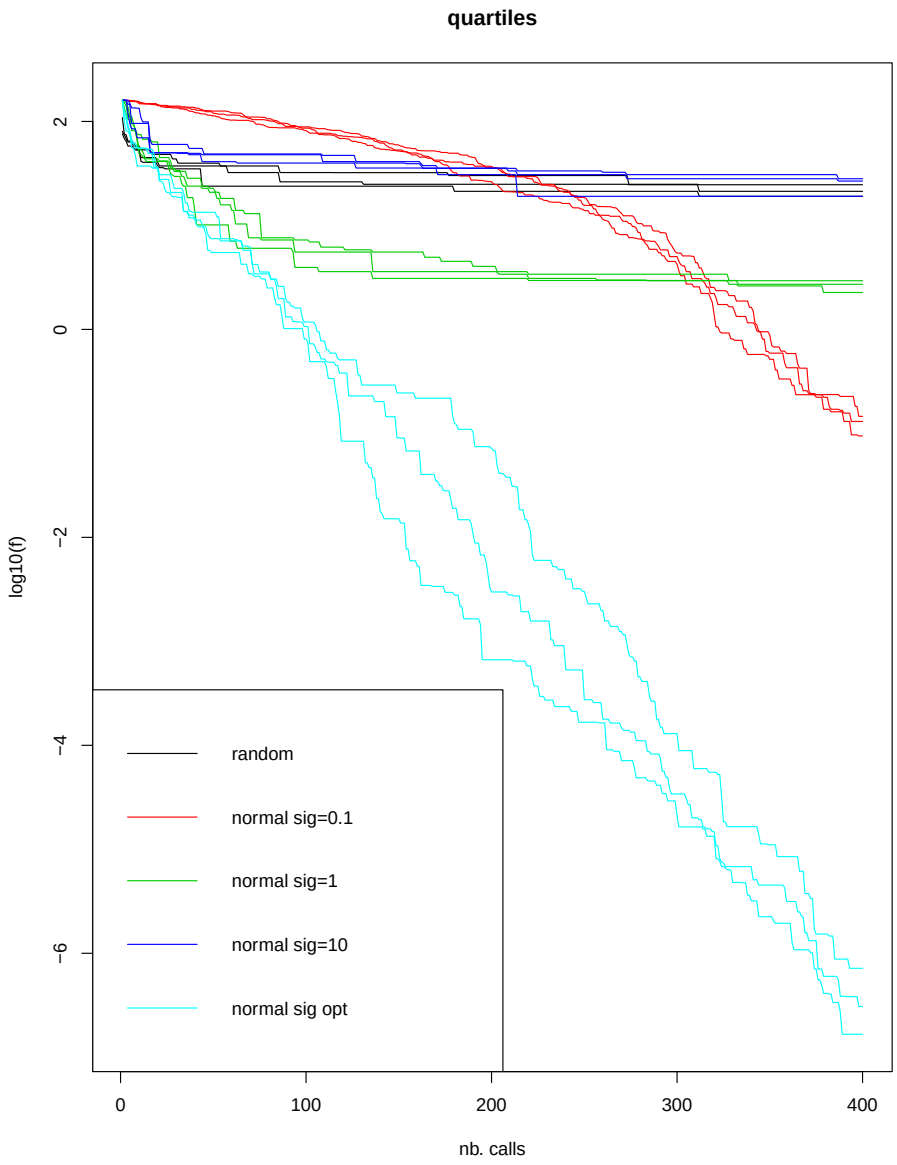
The choice of σ^2 is critical

Minimize the sphere function $f(\mathbf{x}) = \|x\|^2$, $\mathbf{x} \in \mathbb{R}^{10}$, starting from $\mathbf{m} = (4, \ldots, 4)^\top$ with ES-(1+1) constant circular step size, 10 repetitions. Plot shows the 25%, 50% and 75% quantiles.

A constant step size may go from too small to too large during convergence.

The optimal step size for the sphere function is (proof in a later slide)

$$\sigma^\star \approx 1.22 \frac{\|x\|}{d}$$



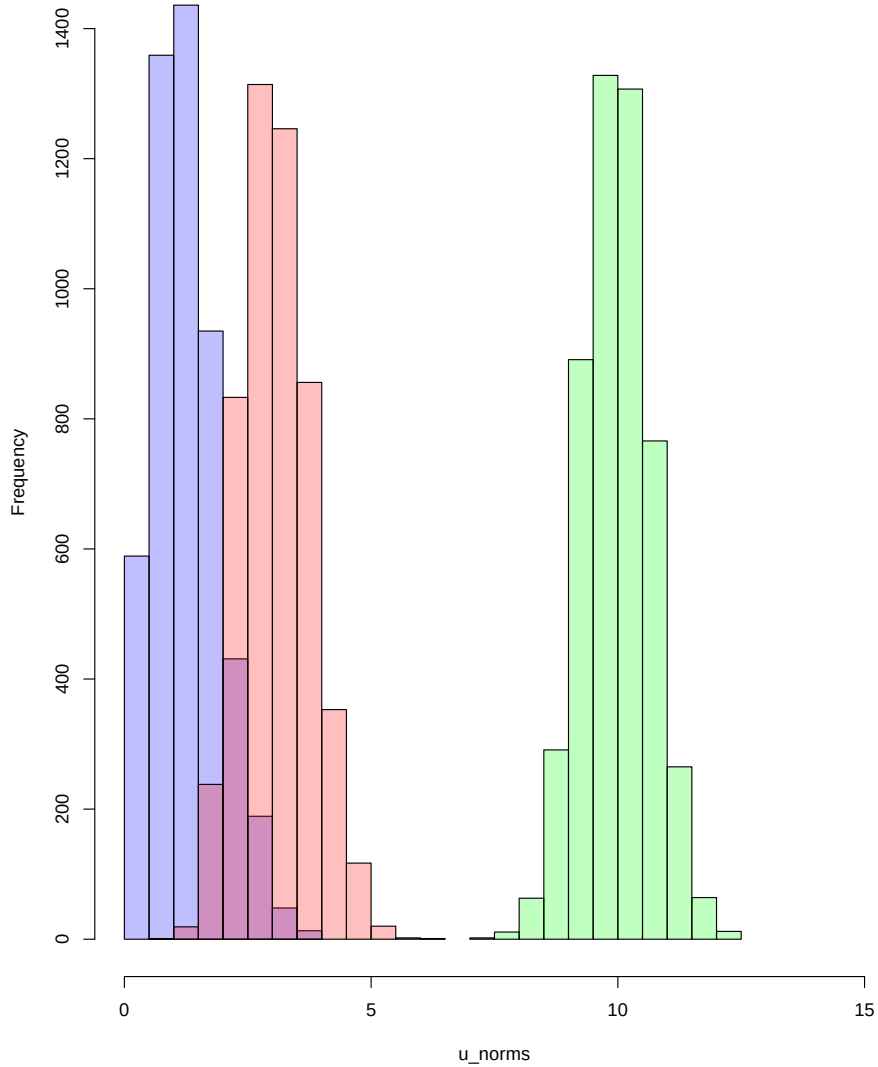
Multivariate normal facts

In the simple ES-(1+1), the perturbation seen by the current mean $\mathbf{m}$ is $\|\mathbf{x} - \mathbf{m}\|^2 = \sigma^2 \sum_{i=1}^{d} u_i^2$ where $u_i \sim \mathcal{N}(0,1)$ iid,

$\|\mathbf{x} - \mathbf{m}\|^2 \sim \sigma^2 \chi_d^2$

As $d \nearrow$, $\chi_d^2 \to \mathcal{N}(d, 2d)$ and $\sqrt{\chi_d^2} \to \mathcal{N}(\sqrt{d}, 1/2)$ (see graph, $d = 2, 10, 100$)

$\Rightarrow$ as $d \nearrow$, the points are concentrated in a sphere of radius $\sigma\sqrt{d}$ centered at $\mathbf{m}$.



Optimal step size of ES-(1+1) with the sphere function (1)

The improvement ito f is
$I_2 = \|\mathbf{m}\|^2 - \|\mathbf{m} + \sigma\mathbf{U}\|^2$
$= \mathbf{m}^\top\mathbf{m} - (\mathbf{m} + \sigma\mathbf{U})^\top(\mathbf{m} + \sigma\mathbf{U})$
$= -\sigma^2\mathbf{U}^\top\mathbf{U} - 2\sigma\mathbf{m}^\top\mathbf{U}$

$\mathbf{U}^\top\mathbf{U} \xrightarrow{d \nearrow} \mathcal{N}(d, 2d) \approx d$
$I_2 \approx -\sigma^2 d - 2\sigma\mathbf{m}^\top\mathbf{U}$

$I_2 \sim \mathcal{N}(-\sigma^2 d, 4\sigma^2\|\mathbf{m}\|^2)$

The expected improvement of ES-(1+1) is

$\mathbb{E}[\max(0, I_2)] = \int_0^\infty I_2 p(I_2) dI_2 =$
$\int_{\sigma d/(2\|\mathbf{m}\|)}^{+\infty} (2\sigma\|m\| i - \sigma^2 d)\phi(i) di$

with the normalization $i = (I_2 + \sigma^2 d)/(2\sigma\|m\|)$ and $\phi()$ pdf of $i \sim \mathcal{N}(0, 1)$

Optimal step size of ES-(1+1) with the sphere function (2)

$$\mathbb{E}\left[\max(0, I_2)\right] = -\sigma^2 d\left[1 - \Phi\left(\frac{\sigma d}{2\|\mathbf{m}\|}\right)\right] + 2\sigma\|\mathbf{m}\|\phi\left(\frac{\sigma d}{2\|\mathbf{m}\|}\right)$$

where $\Phi()$ denotes the cdf of $\mathcal{N}(0,1)$. Multiplying by $\frac{d}{\|\mathbf{m}\|^2}$ and introducing the normalized step $\overline{\sigma} = \frac{\sigma d}{\|\mathbf{m}\|}$, the normalized improvement is

$$\boxed{\frac{d}{\|\mathbf{m}\|^2}\mathbb{E}\left[\max(0, I_2)\right] \;=\; \overline{\sigma}^2\left[\Phi(\frac{\overline{\sigma}}{2})\right] + 2\overline{\sigma}\phi(\frac{\overline{\sigma}}{2})}$$

which is maximized at $\overline{\sigma^\star} \approx 1.22 \Rightarrow \sigma^\star \approx 1.22\frac{\|\mathbf{m}\|}{d}$

Optimal step size of ES-(1+1) with a parabolic function

Let optimize $f(x) = \frac{1}{2}\mathbf{x}^\top \mathbf{H}\mathbf{x}$ with an ES-(1+1) whose steps follow $\mathbf{x} - \mathbf{m} \sim \mathcal{N}(0, \sigma^2 \overline{C})$.
$\overline{C}$ can be decomposed into the product of its eigenvectors $\mathbf{B}$ and eigenvalues $\mathbf{D}^2$, $\overline{C} = \mathbf{B}\mathbf{D}^2\mathbf{B}^\top$.
We choose $\overline{C} = \mathbf{H}^{-1}$ and perform the change of variables $\mathbf{y} = \mathbf{D}^{-1}\mathbf{B}^\top\mathbf{x}$. Then, $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top\mathbf{B}\mathbf{D}^{-1}\mathbf{D}^{-1}\mathbf{B}^\top\mathbf{x} = \frac{1}{2}\mathbf{y}^\top\mathbf{y} = f(\mathbf{y}) \Rightarrow$ in the y-space f is a sphere.
The covariance of Y is $\mathbf{C}_\mathbf{y} = \sigma^2\mathbf{D}^{-1}\mathbf{B}^\top\mathbf{B}\mathbf{D}^2\mathbf{B}^\top\mathbf{B}\mathbf{D}^{-1} = \sigma^2\mathbf{I}$
$\Rightarrow$ sphere function and spherical perturbations, we can use the result
$\sigma^\star = 1.22\frac{\|\mathbf{y}\|}{d} = \frac{1.22}{d}\sqrt{\mathbf{x}^\top\mathbf{B}\mathbf{D}^{-2}\mathbf{B}^\top\mathbf{x}} = \frac{1.22}{d}\sqrt{\mathbf{x}^\top\mathbf{H}\mathbf{x}}$

C proportional to the inverse of the Hessian allows to use the sphere optimality results and is the best choice when there is only local information $(\nabla f(\mathbf{m}), \nabla^2 f(\mathbf{m}))$.

This is what CMA-ES learns.

ES-(μ, λ) flow chart

CMA-ES is an evolution strategy ES-(μ, λ): λ points sampled, the μ best are used for updating $\mathbf{m}$ and $\mathbf{C}$.

Algorithm 2 Generic ES-(μ, λ)

Require: $\mathbf{m} \in \mathcal{S}$, $\mathbf{C}$, $t^{\max}$

1: calculate $f(\mathbf{m})$, $t \leftarrow 1$, $g \leftarrow 1$
2: **while** $t < t^{\max}$ **do**
3: sample: $\mathbf{x}^1, \ldots, \mathbf{x}^\lambda \sim \mathcal{N}(\mathbf{m}^{(g)}, \mathbf{C}^{(g)})$
4: calculate $f(\mathbf{x}^1), \ldots, f(\mathbf{x}^\lambda)$, $t \leftarrow t + \lambda$
5: rank: $f(\mathbf{x}^{1:\lambda}) \leq \ldots \leq f(\mathbf{x}^{\lambda:\lambda})$
6: estimate $\mathbf{m}^{(g+1)}$ and $\mathbf{C}^{(g+1)}$ with the μ bests, $\mathbf{x}^{1:\lambda}, \mathbf{x}^{2:\lambda}, \ldots, \mathbf{x}^{\lambda:\lambda}$.
7: $g \leftarrow g + 1$
8: **end while**

Default semi-empirical values for CMA-ES (N. Hansen):
$\lambda = 4 + \lfloor 3 \ln(d) \rfloor$, $\mu = \lfloor \frac{\lambda}{2} \rfloor$.

A simplified CMA-ES

We now present a simplification of the CMA-ES algorithm that is easier to understand while keeping some of the key ideas. It will not compare in performance with the true CMA-ES. Disregarded ingredients are

- weighting of the sampled points
- separation of the adaptation of the step size and covariance "shape", $\mathbf{C} = \sigma^2 \overline{\mathbf{C}}$
- simultaneous rank 1 and rank μ updates of the covariance matrix

For competitive implementations, refer to the official tutorial [3].

Points versus steps (1)

Points are generated from sampling, at iteration g, $\mathcal{N}(\mathbf{m}^{(g)}, \mathbf{C}^{(g)})$,

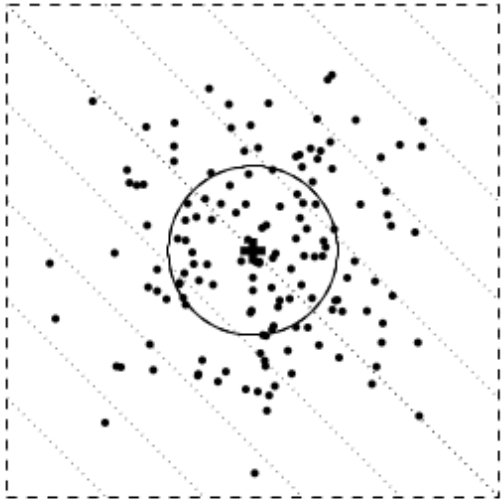
$$\mathbf{x}^i \;=\; \mathbf{m}^{(g)} + \mathcal{N}_i(0, \mathbf{C}^{(g)}) \quad , \quad i = 1, \lambda$$

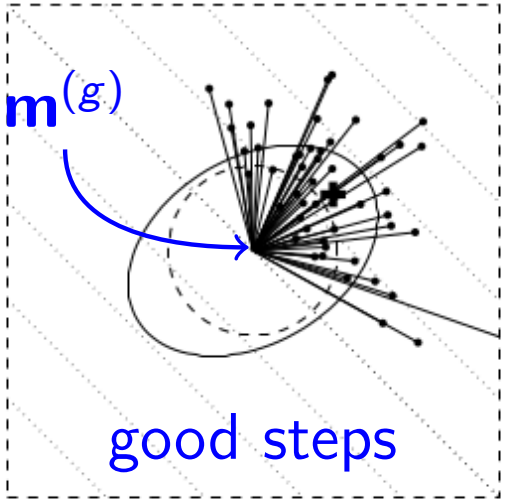
The associated **step** is

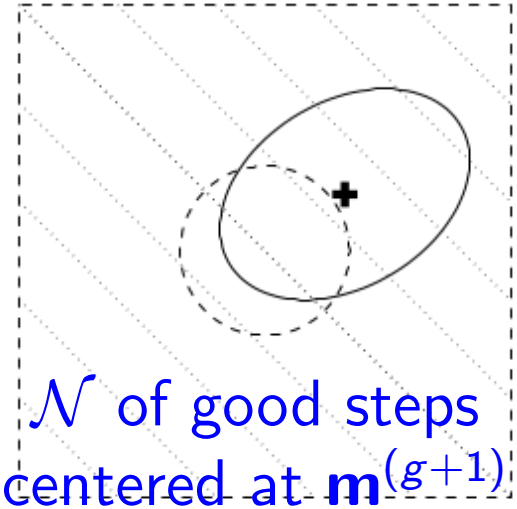
$$\mathbf{y}^i \;=\; \mathbf{x}^i - \mathbf{m}^{(g)} = \mathcal{N}_i(0, \mathbf{C}^{(g)})$$

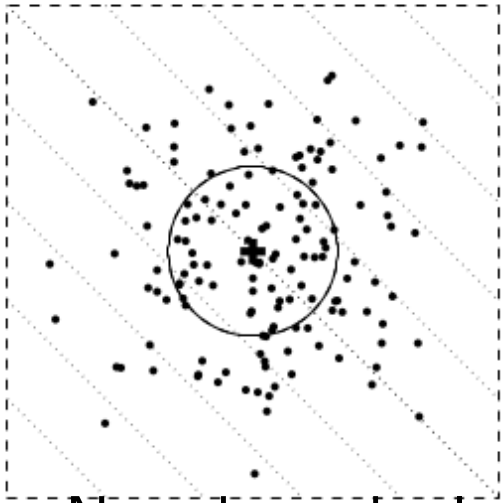
CMA-ES learns steps, as opposed to points. Points are static, steps are dynamic. Efficient optimizers (e.g., Nelder Mead, BFGS, ...) make steps.

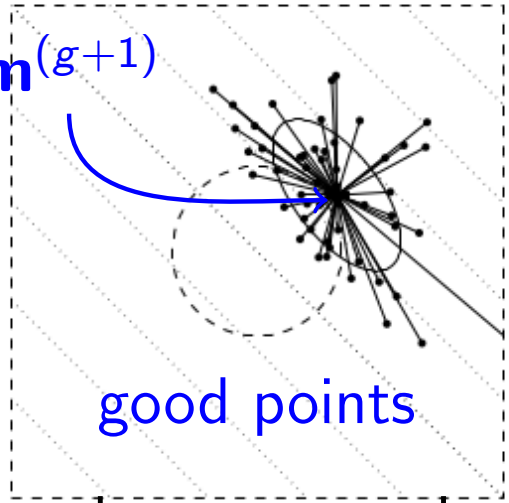
Points versus steps (2)

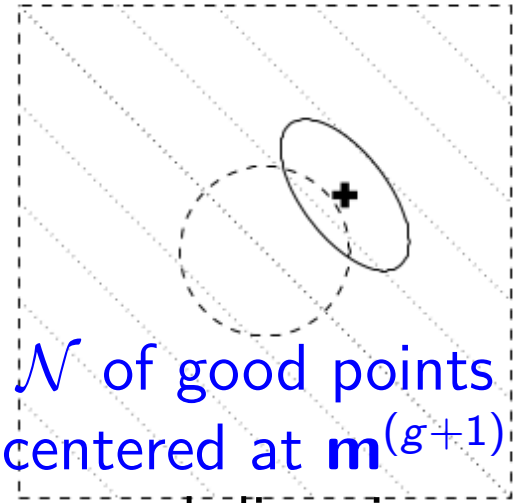












Note how the law of good steps stretches in the good direction as opposed to the good points (plot from [3])

Update of the mean

Remember the order statistics notation for the new sampled points

$$f(\mathbf{x}^{1:\lambda}) \leq \ldots \leq f(\mathbf{x}^{\lambda:\lambda})$$

The new mean is the average of the μ best points,

$$\boxed{\mathbf{m}^{(g+1)} = \frac{1}{\mu} \sum_{i=1}^{\mu} \mathbf{x}^{i:\lambda}}$$

Covariance matrix estimation

Empirical estimate of the good steps,

$$\mathbf{C}^{(g+1)} = \frac{1}{\mu}\sum_{i=1}^{\mu}(\mathbf{x}^{i:\lambda} - \mathbf{m}^{(g)})(\mathbf{x}^{i:\lambda} - \mathbf{m}^{(g)})^\top \tag{4}$$

$$= \frac{1}{\mu}\sum_{i=1}^{\mu}\mathbf{y}^{i:\lambda}{\mathbf{y}^{i:\lambda}}^\top$$

Empirical estimate of the good points (don't use),

$$\mathbf{C}_{\text{EDA}}^{(g+1)} = \frac{1}{\mu-1}\sum_{i=1}^{\mu}(\mathbf{x}^{i:\lambda} - \mathbf{m}^{(g+1)})(\mathbf{x}^{i:\lambda} - \mathbf{m}^{(g+1)})^\top$$

where EDA means Estimation of Density Algorithm (aka cross-entropy method, EMNA – Estimation of Multivariate Normal Algorithm).

Both estimators are unbiased. They require $\mu \geq d$ linearly independent $\mathbf{y}$'s to have full rank and $\lambda \geq 20d$ to be reliable for well-conditioned ($\text{cond}(\mathbf{C}) < 10$) problems [3].

Rank 1 and time averaging for covariance matrix estimation (1)

If f is ill-conditioned (large ratio of largest to smallest curvatures), the estimator (4) requires large μ (hence λ) which makes the optimizer costly.
Solution: account for information coming from past iterations by time averaging, which allows few samples and rank 1 updates.

Use the average good step,

$$\overline{\mathbf{y}} = \frac{1}{\mu}\sum_{i=1}^{\mu}\left(\mathbf{x}^{(i:\lambda)} - \mathbf{m}^{(g)}\right) = \mathbf{m}^{(g+1)} - \mathbf{m}^{(g)} \tag{5}$$

Rank 1 and time averaging for covariance matrix estimation (2)

and time averaging of the covariance estimates,

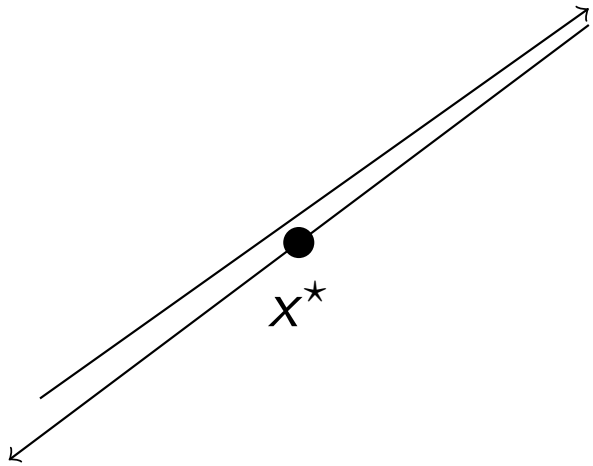
$$\boxed{\mathbf{C}^{(g+1)} = (1-c_1)\mathbf{C}^{(g)} + c_1 \overline{\mathbf{y}}\overline{\mathbf{y}}^\top} \quad (6)$$

where $0 \leq c_1 \leq 1$.
If $c_1 = 1$, no memory and $\mathbf{C}^{(g+1)}$ has rank 1 (degenerated gaussian along $\overline{\mathbf{y}}$).
Default[1] $c_1 = 2/((d+1.3)^2 + \mu)$.
$1/c_1$ is the backward time horizon that contributes to about 63% of the information [3].

[1] Default for the complete CMA-ES, not sure it applies to this simplified version. Idem with c_c later.

Steps cumulation

The estimation (6) has a flaw: it is not sensitive to the sign of the step, $\overline{\mathbf{y}}\overline{\mathbf{y}}^\top = (-\overline{\mathbf{y}})(-\overline{\mathbf{y}})^\top$. It will not detect oscillations and cannot converge. Worse, it will increase step size around $\mathbf{x}^\star$.



$\Rightarrow$ use cumulated steps (aka evolution path) instead of $\overline{\mathbf{y}}$ in (6),

$$\mathbf{p}^{(g+1)} = (1-c_c)\mathbf{p}^{(g)} + \sqrt{c_c(2-c_c)\mu}\ \overline{\mathbf{y}} \tag{7}$$

Default $c_c = \frac{4+\mu/d}{d+4+2\mu/d}$.

The weighting in (7) is such that if $\mathbf{y}^{i:\lambda} \sim \mathcal{N}(0, \mathbf{C})$ and $\mathbf{p}^{(g)} \sim \mathcal{N}(0, \mathbf{C})$, then $\mathbf{p}^{(g+1)} \sim \mathcal{N}(0, \mathbf{C})$.

Proof: $\overline{\mathbf{y}} = \frac{1}{\mu}\sum_{i=1}^{\mu} \mathbf{y}^{i:\lambda} \sim \mathcal{N}(0, \frac{\mu}{\mu^2}\mathbf{C})$,

$\mathbf{p}^{(g+1)} \sim \mathcal{N}(0, (1-c_c)^2\mathbf{C}) + \mathcal{N}(0, \frac{c_c(2-c_c)}{\mu}\mathbf{C}) \sim \mathcal{N}(0, \mathbf{C})$. $\square$

Practicalities

- Accounting for the variables bounds:

  repeat
  sample: $\mathbf{x} \sim \mathcal{N}(\mathbf{m}, \mathbf{C})$
  until $\mathbf{x}^{\text{LB}} \leq \mathbf{x} \leq \mathbf{x}^{\text{UB}}$

- Another stopping criterion: a maximum number of f calculations without progress (to avoid stagnation).

Concluding remarks I

- Both algorithms are likely to miss the global optimum: practically, they are local (although CMA-ES is asymptotically convergent to the global optimum and Nelder-Mead may go over regions of attraction to local optima).
- Restart Nelder-Mead and CMA-ES at convergence (possibly force a few early convergences for restarts). Competitive versions of these algorithms use restarts (e.g., [9, 11]).

Concluding remarks II

- A tempting idea when $f()$ is costly:
  1. Make a design of experiments $(\mathbb{X},\mathbb{F})$ and build a metamodel $\widehat{f}$
  2. optimize the (almost free) $\widehat{f}$ with your favorite optimizer, $\mathbf{x}^{t+1} = \arg\min_{\mathbf{x}\in\mathcal{S}} \widehat{f}(\mathbf{x})$
  3. Calculate $f(\mathbf{x}^{t+1})$, stop or back to 1 with $(\mathbb{X},\mathbb{F}) \cup (\mathbf{x}^{t+1}, f(\mathbf{x}^{t+1}))$.

  but convergence may be harmed if $\widehat{f}(\mathbf{x}^{t+1}) \approx f(\mathbf{x}^{t+1})$ and $\mathbf{x}^{t+1}$ far from $\mathbf{x}^\star \Rightarrow$ use a well-designed optimizer for this, e.g., EGO [7], saACM-ES [10] or an EGO-CMA mix [13].

Acknowledgments: I would like to thank Victor Picheny, Stéphanie Mahévas and Robert Faivre for their invitation at this MEXICO network school.

References I

[1] Anne Auger and Nikolaus Hansen.
Linear convergence of comparison-based step-size adaptive randomized search via stability of markov chains.
SIAM Journal on Optimization, 26(3):1589–1624, 2016.

[2] Andrew R. Conn, Katya Scheinberg, and Luis N. Vicente.
Introduction to Derivative-Free Optimization.
Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2009.

[3] Nikolaus Hansen.
The CMA Evolution Strategy: A Tutorial.
ArXiv e-prints, arXiv:1604.00772, 2016, 2005.

[4] Nikolaus Hansen and Andreas Ostermeier.
Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation.
pages 312–317. Morgan Kaufmann, 1996.

[5] Nikolaus Hansen and Andreas Ostermeier.
Completely derandomized self-adaptation in evolution strategies.
Evol. Comput., 9(2):159–195, June 2001.

[6] Steven G. Johnson.
The NLopt nonlinear-optimization package, 2011.

References II

[7] Donald R. Jones, Matthias Schonlau, and William J. Welch.
Efficient global optimization of expensive black-box functions.
Journal of Global optimization, 13(4):455–492, 1998.

[8] Jeffrey C. Lagarias, James A. Reeds, Margaret H. Wright, and Paul E. Wright.
Convergence properties of the Nelder-Mead simplex method in low dimensions.
SIAM Journal on Optimization, 9(1):112–147, 1998.

[9] Ilya Loshchilov.
Cma-es with restarts for solving cec 2013 benchmark problems.
In *2013 IEEE Congress on Evolutionary Computation*, pages 369–376, June 2013.

[10] Ilya Loshchilov, Marc Schoenauer, and Michèle Sebag.
Intensive Surrogate Model Exploitation in Self-adaptive Surrogate-assisted CMA-ES (saACM-ES).
In Christian Blum and Enrique Alba, editors, *Genetic and Evolutionary Computation Conference (GECCO 2013)*, pages 439–446, Amsterdam, Netherlands, July 2013.

[11] Marco A. Luersen and Rodolphe Le Riche.
Globalized nelder–mead method for engineering optimization.
Computers & Structures, 82(23):2251 – 2260, 2004.

References III

[12] Ken IM McKinnon.
Convergence of the nelder–mead simplex method to a nonstationary point.
SIAM Journal on Optimization, 9(1):148–158, 1998.

[13] Hossein Mohammadi, Rodolphe Le Riche, and Eric Touboul.
Making EGO and CMA-ES Complementary for Global Optimization.
In *Learning and Intelligent Optimization*, volume Volume 8994 of *9th International Conference, LION 9, Lille, France, January 12-15, 2015. Revised Selected Papers*, pages pp 287–292. May 2015.

[14] John A. Nelder and Roger Mead.
A simplex method for function minimization.
The Computer Journal, 7(4):308–313, 1965.

[15] Yann Ollivier, Ludovic Arnold, Anne Auger, and Nikolaus Hansen.
Information-geometric optimization algorithms: A unifying picture via invariance principles.
Journal of Machine Learning Research, 2017.
accepted.

[16] P. Tseng.
Fortified-descent simplicial search method: A general approach.
SIAM J. Optim., 10:269–288, 1999.