Derivative Free Optimization

joint course between
Optimization Master Paris Saclay - AMS Master
2024/2025

Anne Auger - Dimo Brockhoff
RandOpt team
Inria and CMAP, Ecole Polytechnique, IP Paris
anne.auger@inria.fr

Organization of the class

When: Friday afternoon - 2pm - 5:15pm at ENSTA

29/11/2024	room 1314
06/12/2024	room 1314
13/12/2024	room 1314
20/12/2024	room 1213
10/01/2025	room 1213
17/01/2025	room 1314
24/01/2025	room 1314
31/01/2025	room 1314
07/02/2025	room 1314
14/02/2025 [EXAM]	TBA

Evaluation

Written exam on 14/02/2025) 60%

Project (in group) around benchmarking of algorithms) 40%

- oral presentation to the class

Syllabus

Topics covered

Derivative Free Optimization / Black-box optimization

- Single-objective optimization
  - what makes a problem difficult
  - algorithm to solve those difficulties (mostly stochastic)
- Multi-objective optimization [taught D. Brockhoff]
- Benchmarking (partly taught by D. Brockhoff)

Practical Exercices

practical exercices: implement/manipulate algorithms

Python / Matlab / ...

ultimate goal: optimize a (real) black-box problem on your own

- understand and visualize convergence / adaptation / invariance
- experience numerics numerical errors, finite machine precision

Derivative-Free / Black-box Optimization

Task: minimize a numerical **objective** function (also called *fitness* function or *loss* function)

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

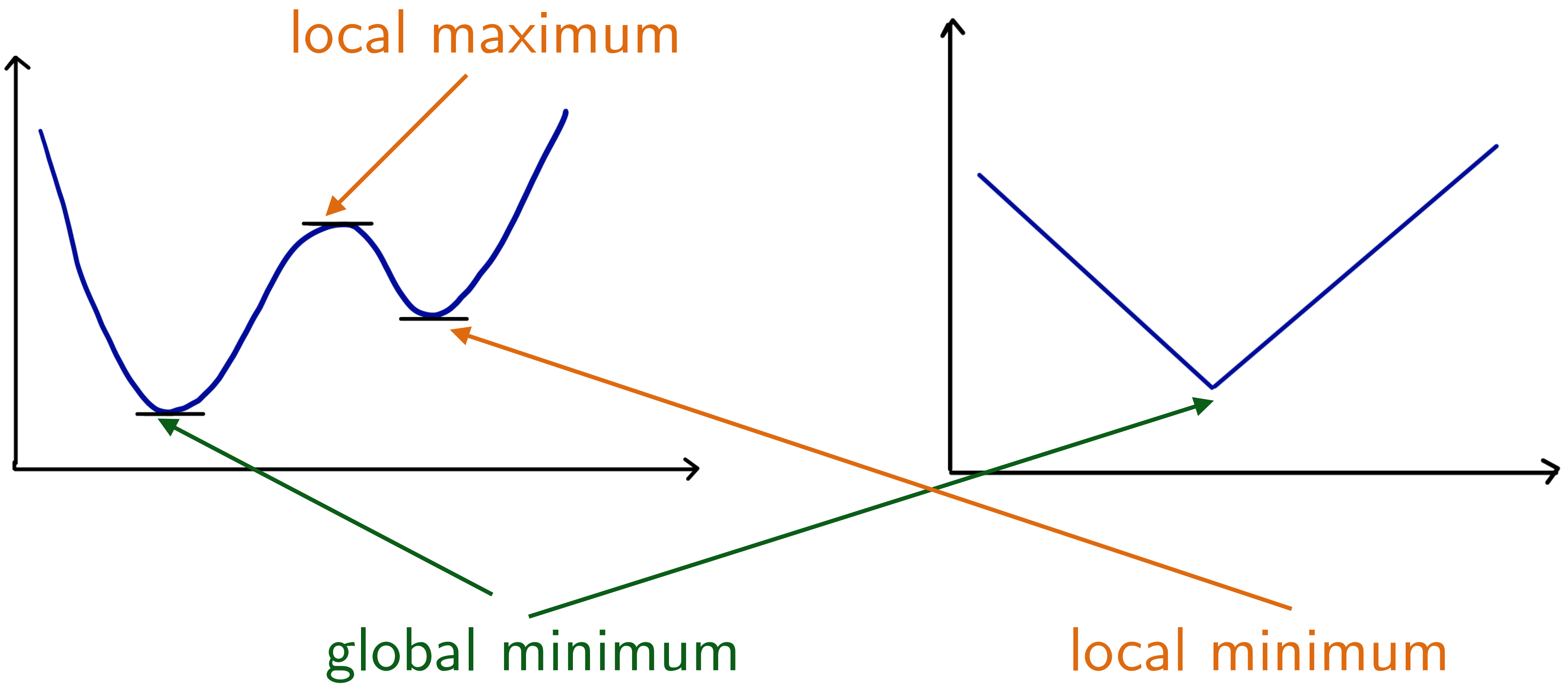
without derivatives (gradient). Ω: search space, n :dimension of the search space

Also called **zero-order black-box** optimization

The function is seen by the algorithm as a zero-order **oracle** [a first order oracle would also return gradients] that can be queried at points and the oracle returns an answer

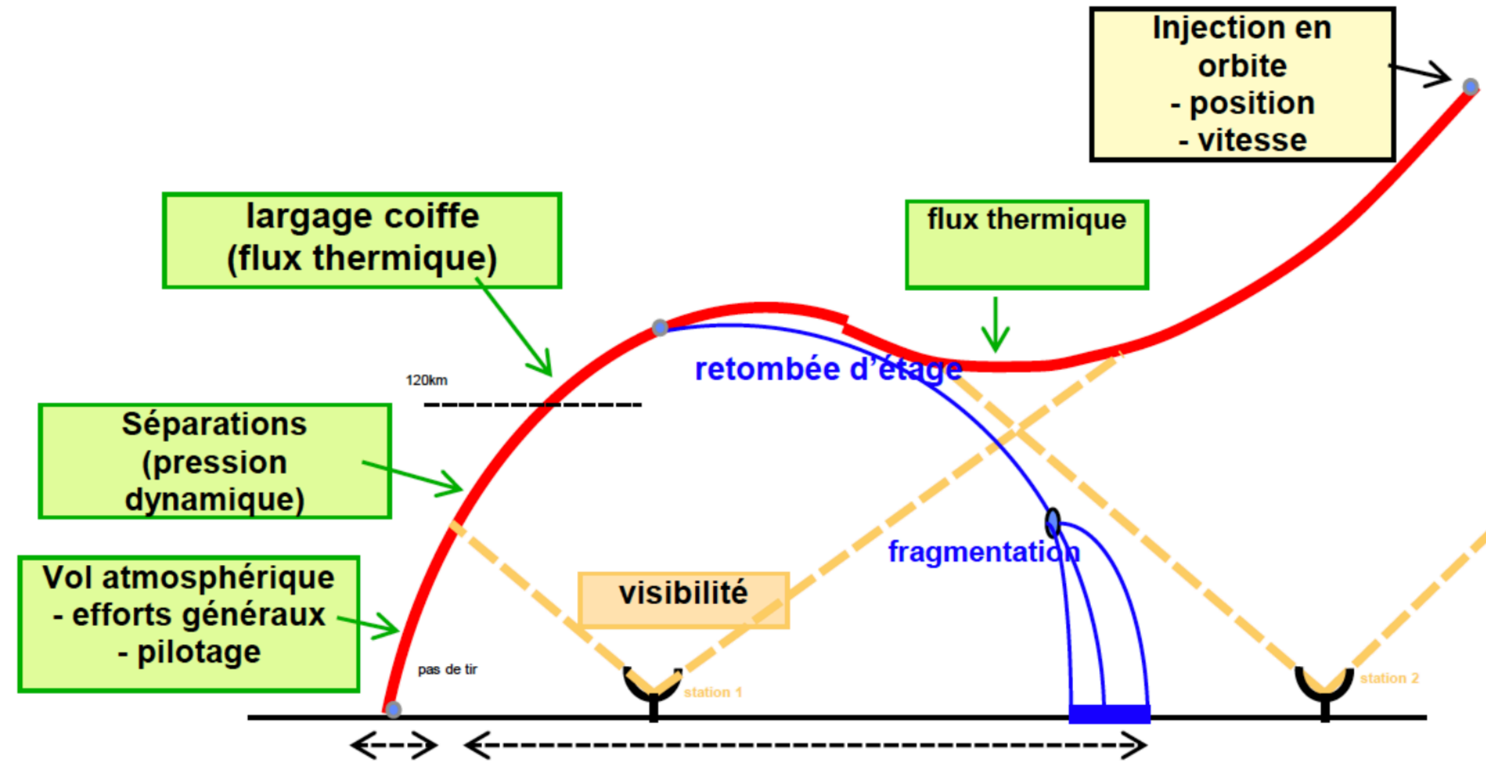
Reminder: Local versus Global Optimum

n=1



Examples: Optimization of the Design of a Launcher

Poppy



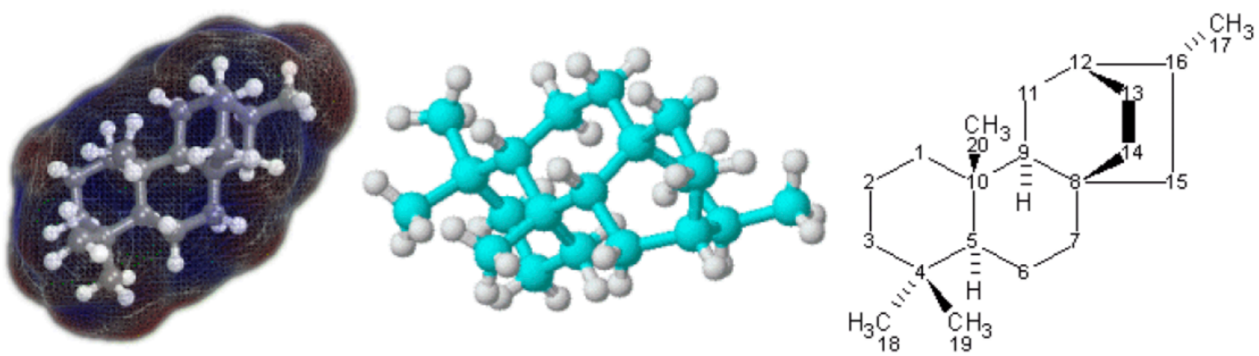
- Scenario: multi-stage launcher brings a satellite into orbit
- Minimize the overall cost of a launch
- Parameters: propellant mass of each stage / diameter of each stage / flux of each engine / parameters of the command law

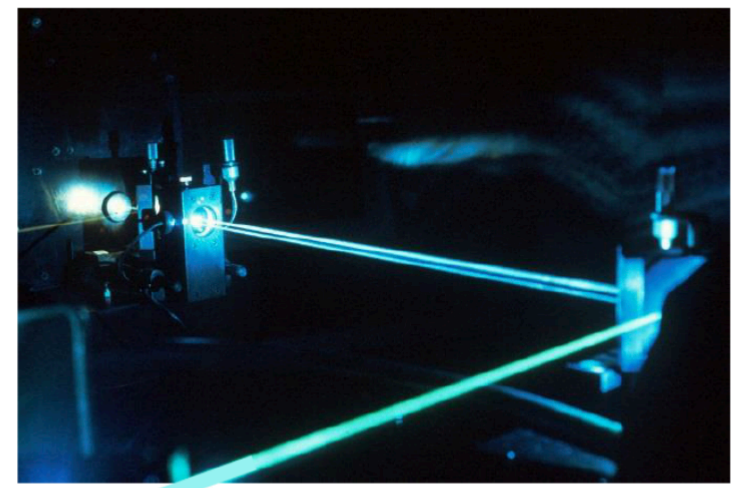
23 continuous parameters to optimize

+ constraints

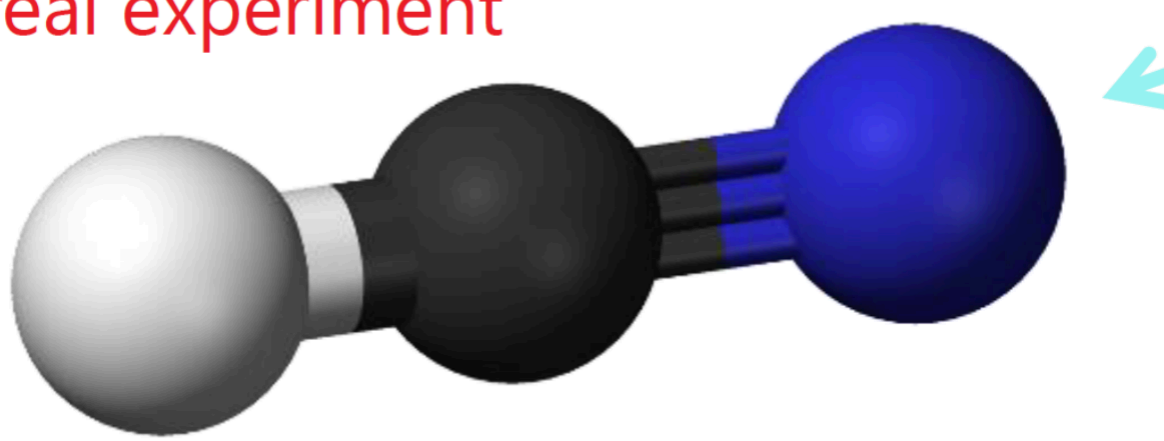
Control of the Alignement of Molecules

application domain: quantum physics or chemistry





Objective function:
via numerical simulation
or a real experiment



possible application in drug design

In the case of a real lab experiment: the objective function is a real black-box

Coffee Tasting Problem (A real Black-box)

Coffee Tasting Problem

$x_i \geq 0$
$\sum x_i = 1$

- Find a mixture of coffee in order to keep the coffee taste from one year to another

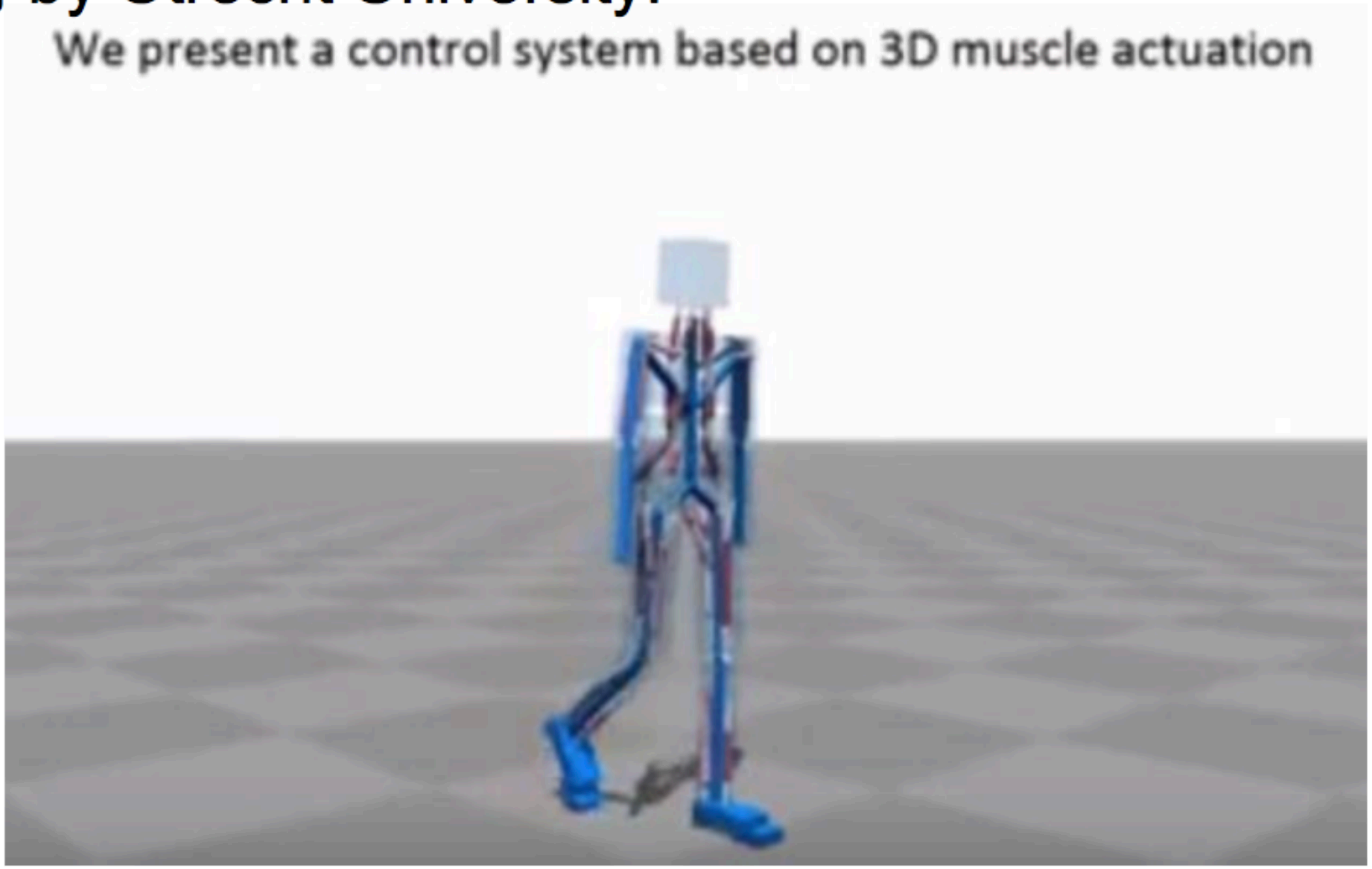
$(x_1, x_2, x_3, x_4) \longrightarrow$ Taste

- Objective function = opinion of one expert

M. Herdy: "Evolution Strategies with subjective selection", 1996

A last Application

Computer simulation teaches itself to walk upright (virtual robots (of different shapes) learning to walk, through stochastic optimization (CMA-ES)), by Utrecht University:



https://www.youtube.com/watch?v=yci5FuI1ovk

T. Geitjtenbeek, M. Van de Panne, F. Van der Stappen: "Flexible Muscle-Based Locomotion for Bipedal Creatures", SIGGRAPH Asia, 2013.

What is the Goal?

- We want to find $x^\star$ such that $f(x^\star) \leq f(x)$ for all x

$$x^\star \in \operatorname{argmin}_x f(x)$$

- In general we will never find $x^\star$

why?

What is the Goal?

- We want to find $x^\star$ such that $f(x^\star) \leq f(x)$ for all x

$$x^\star \in \operatorname{argmin}_x f(x)$$

- In general we will never find $x^\star$

- Because of the numerical/continuous nature of the search space we typically never hit exactly $x^\star$, we instead converge to a solution:

  we want to find $x_t \in \mathbb{R}^n$ such that $\lim_{t\to\infty} f(x_t) = \min f$

  of course we want ***fast*** convergence

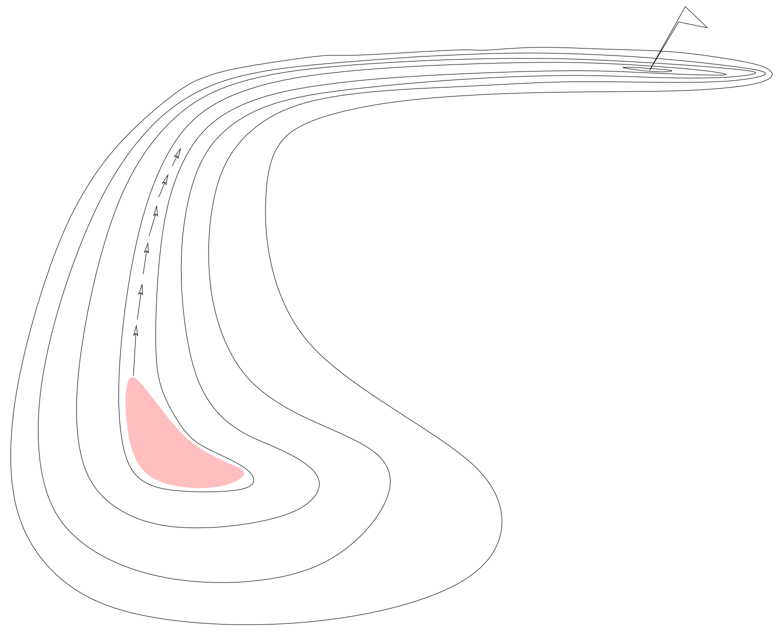
Level Sets of a Function

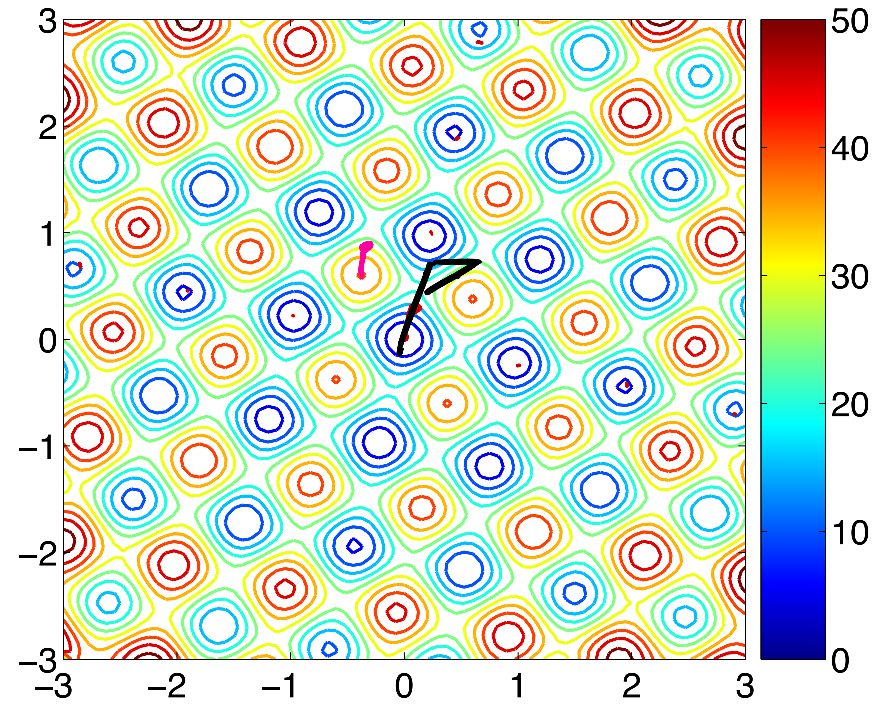
Level Sets: Visualization of a Function

One-dimensional (1-D) representations are often misleading (as 1-D optimization is "trivial", see slides related to curse of dimensionality), we therefore often represent level-sets of functions

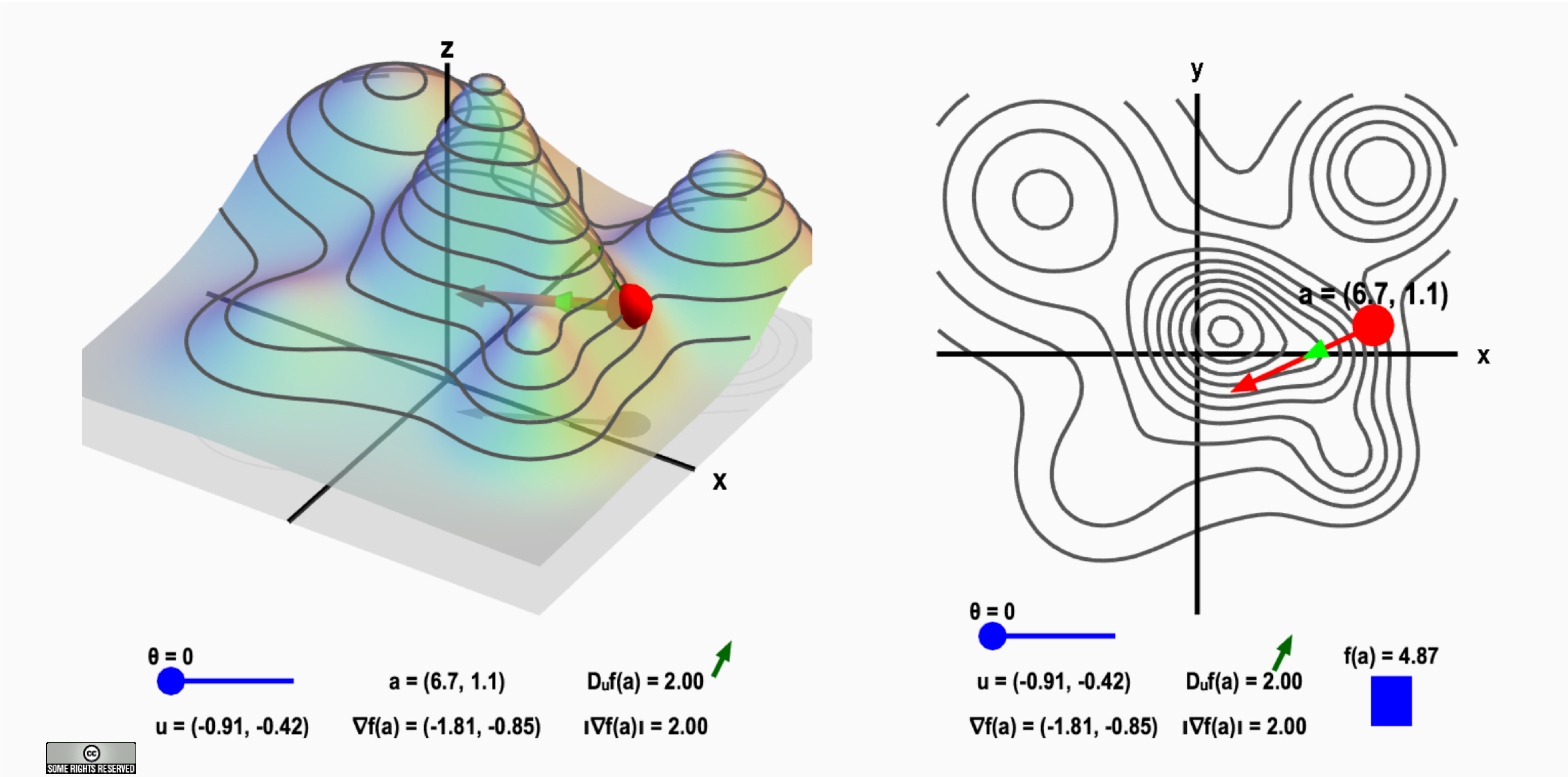
$$\mathcal{L}_c = \{x \in \mathbb{R}^n \,|\, f(x) = c\}, c \in \mathbb{R}$$

Examples of level sets in 2D





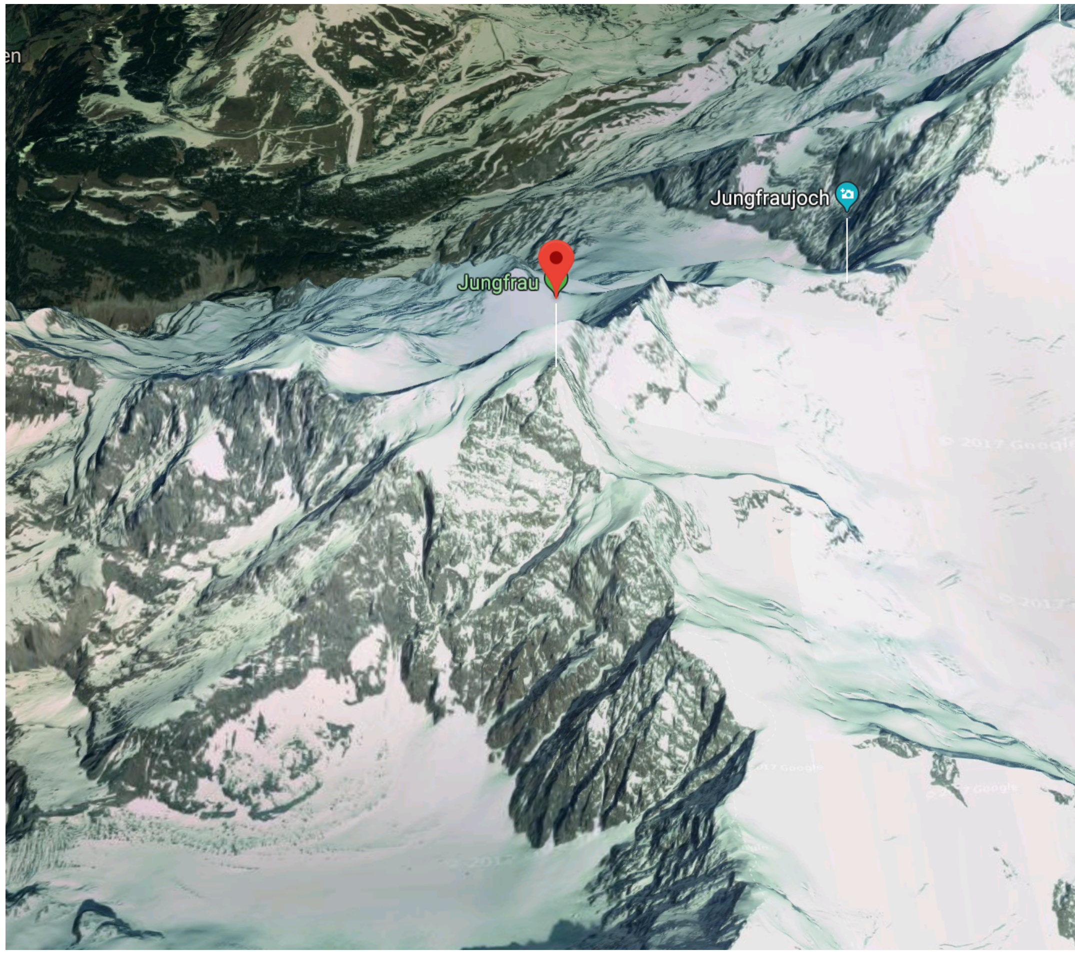
Level Sets: Visualization of a Function



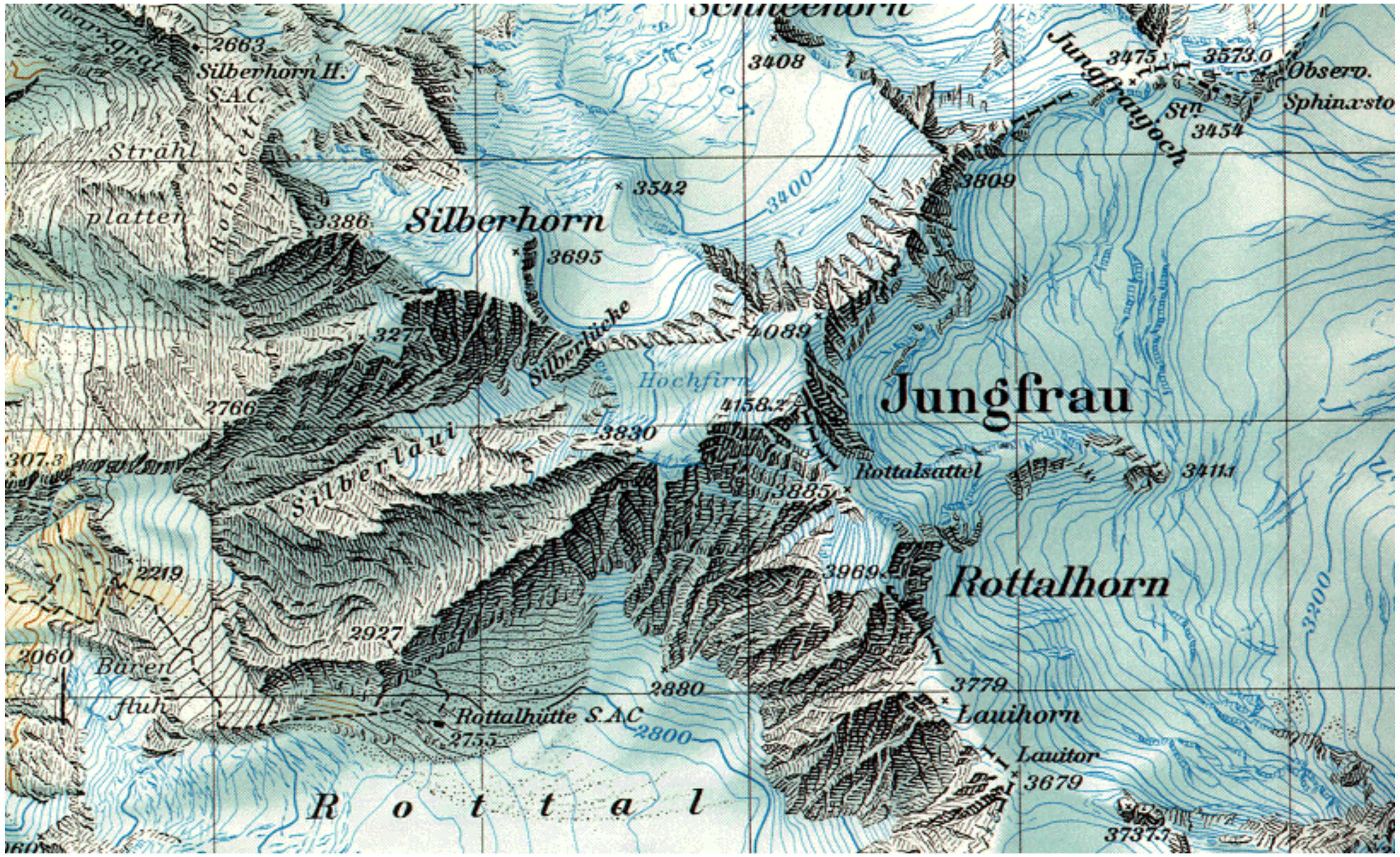
Source: Nykamp DQ, "Directional derivative on a mountain." From *Math Insight*. http://mathinsight.org/applet/directional_derivative_mountain

Level Sets: Topographic Map

The function is the altitude



3-D picture



Topographic map

Level Set: Exercice

Consider a strictly convex-quadratic function

$$f(x) = \frac{1}{2}(x - x^\star)^\top H (x - x^\star) = \frac{1}{2}\sum_i h_{ii}(x_i - x_i^\star)^2 + \frac{1}{2}\sum_{i \neq j} h_{ij}(x_i - x_i^\star)(x_j - x_j^\star)$$

with H a symmetric, positive, definite matrix ($H \succ 0$).

1. What is/are the optima of f ? What does H represent for the function ?

2. Assume n=2, $H = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ plot the level sets of f

3. Same question with $H = \begin{bmatrix} 1 & 0 \\ 0 & 9 \end{bmatrix}$

4. Same question with $H = P \begin{bmatrix} 1 & 0 \\ 0 & 9 \end{bmatrix} P^T$ with $P \in \mathbb{R}^{2\times 2}$

   P orthogonal

$f(x) = \frac{1}{2}(x - x^*)^T H (x - x^*)$ $\quad H \succ 0$

strictly convex.

$f(x) \geq 0$ because $H \succ 0$

$f(x) = 0 \iff x - x^* = 0 \iff x = x^*$

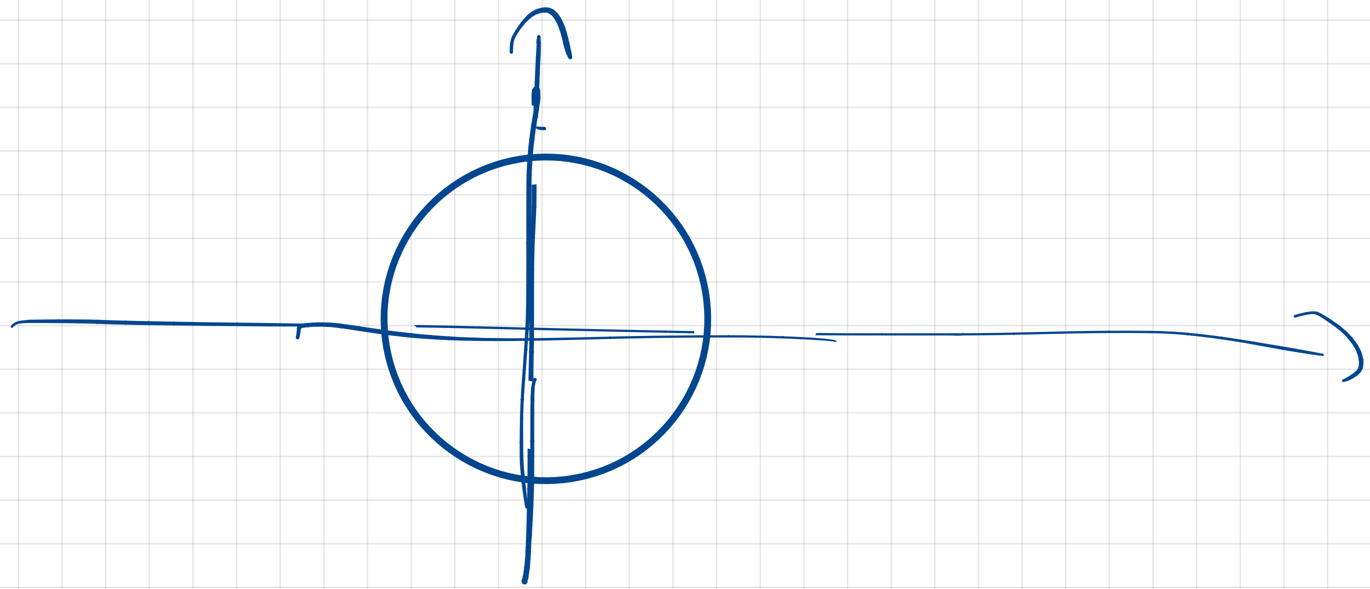
$H = \nabla^2 f(x)$ Hessian matrix

WLG $x^* = 0$

2) $H = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ $\quad f(x) = \frac{1}{2}(x_1^2 + x_2^2)$

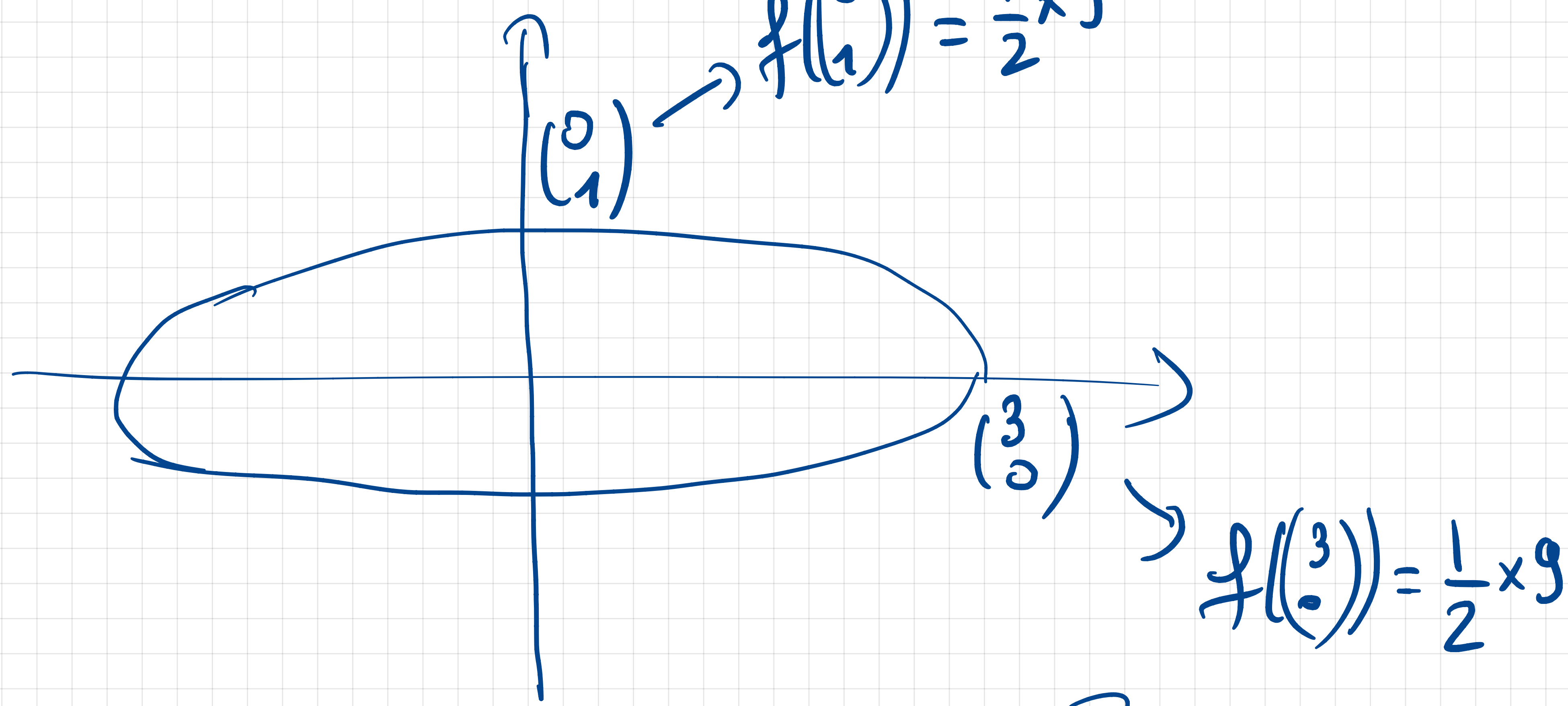
$L_c = \{ x \mid \frac{1}{2}(x_1^2 + x_2^2) = c \}$

$c > 0$



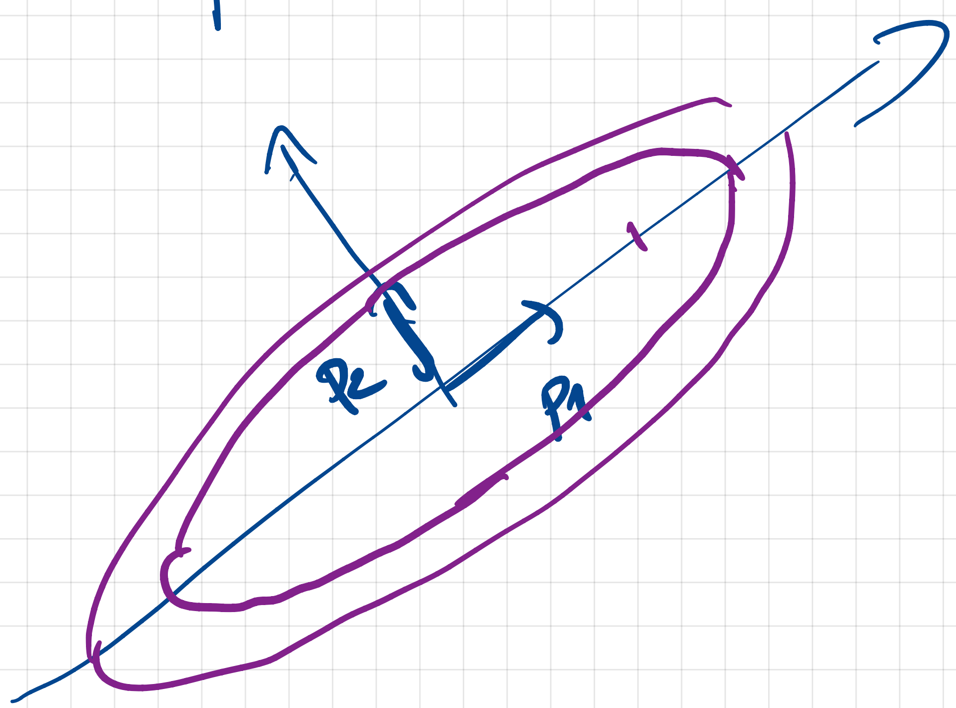
$$H = \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix} \qquad L_c = \left\{ x = (x_1, x_2) \;\middle|\; \frac{1}{2}(x_1^2 + 9x_2^2) = c \right\}$$

$c > 0$, ellipsoid.



$$H = P^T \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix} P$$

$P = (p_1, p_2)$ eigenvectors of H



What Makes an Optimization Problem Difficult?

What Makes a Function Difficult to Solve?

Why stochastic search?

- non-linear, non-quadratic, non-convex

  on linear and quadratic functions much better search policies are available

- ruggedness

  non-smooth, discontinuous, multimodal, and/or noisy function

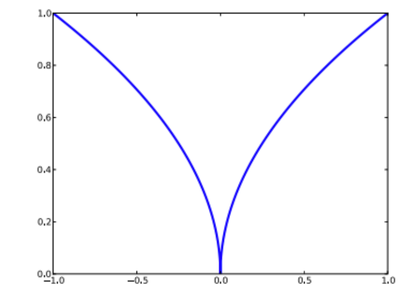
- dimensionality (size of search space)

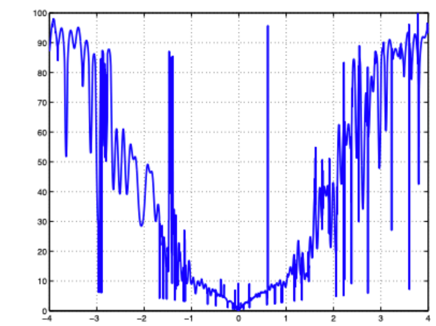
  (considerably) larger than three

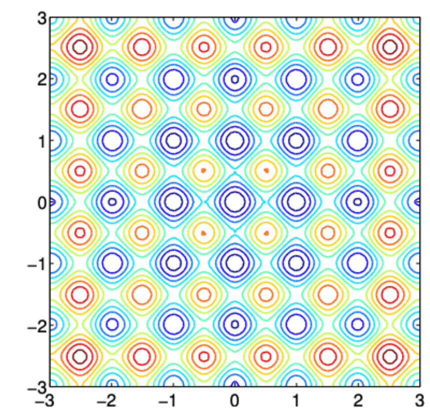
- non-separability

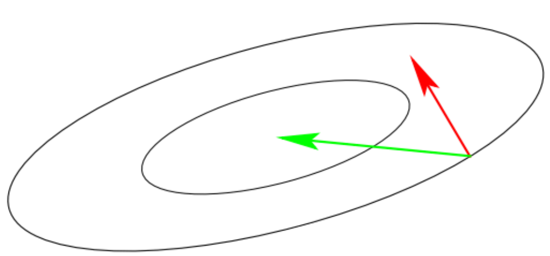
  dependencies between the objective variables

- ill-conditioning



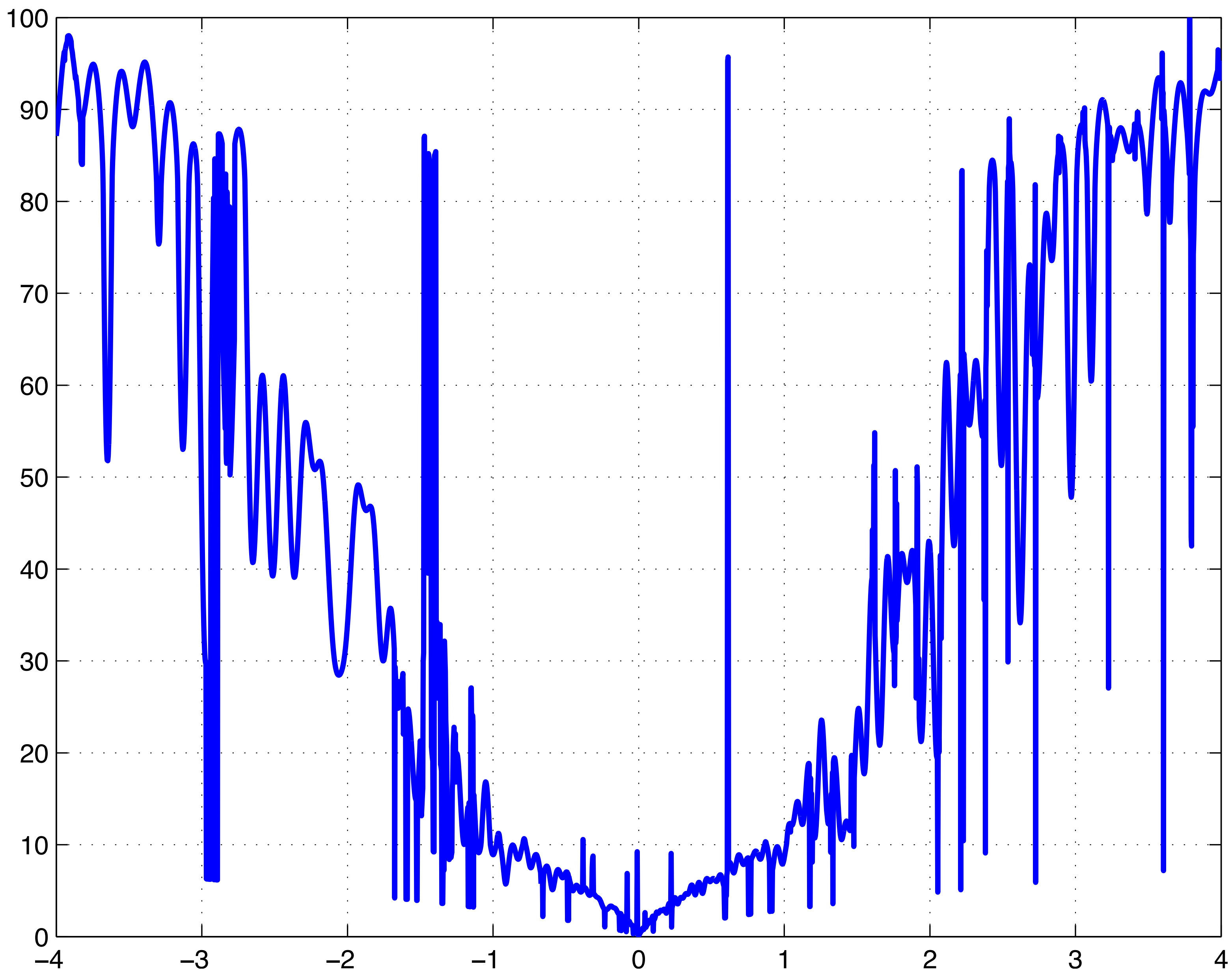






gradient direction Newton directio

Ruggedness



$f: x \in \mathbb{R}^4 \longrightarrow \mathbb{R}$

$d \in \mathbb{R}^4$

$t \in \mathbb{R} \longrightarrow f(td)$

A cut of a 4-D function that can easily be solved with the CMA-ES algorithm

Why is Optimization a non-trivial Problem?

Curse of dimensionality

if n=1, which simple approach could you use to minimize:

$$f : [0, 1] \to \mathbb{R} \quad ?$$

Why is Optimization a non-trivial Problem?

Curse of dimensionality

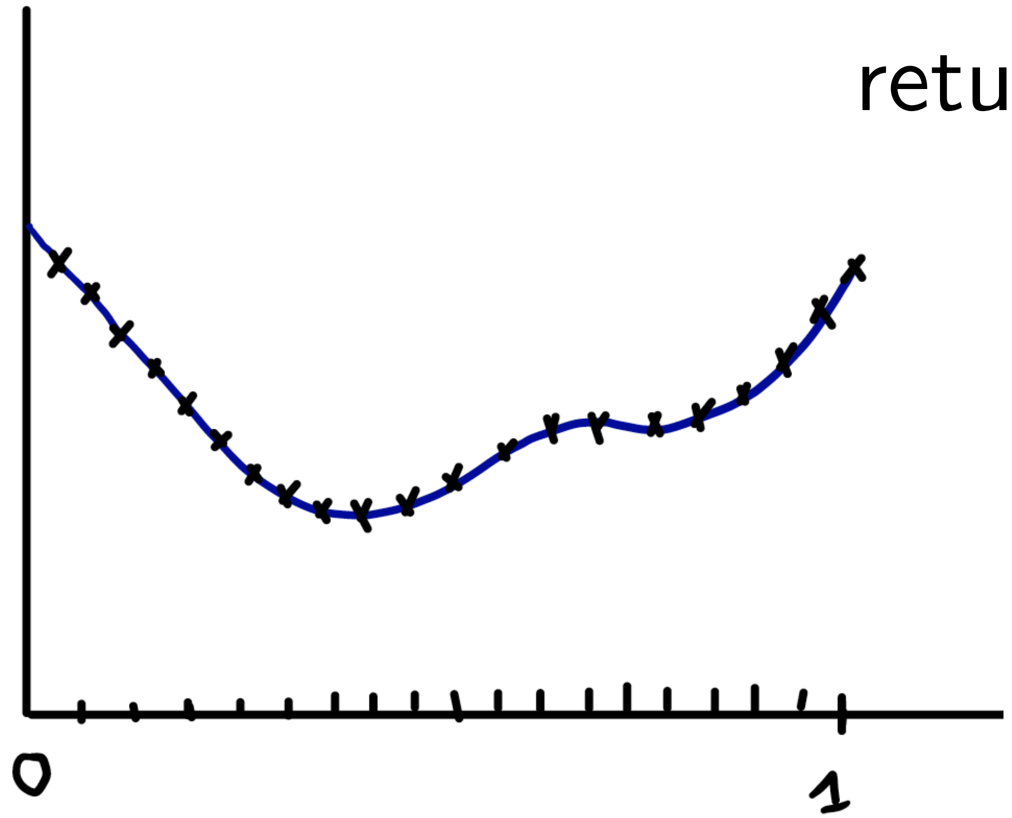
if n=1, which simple approach could you use to minimize:

$$f : [0, 1] \to \mathbb{R} \quad ?$$

set a regular grid on [0,1]

evaluate on f all the points of the grid

return the lowest function value



Why is Optimization a non-trivial Problem?

Curse of dimensionality

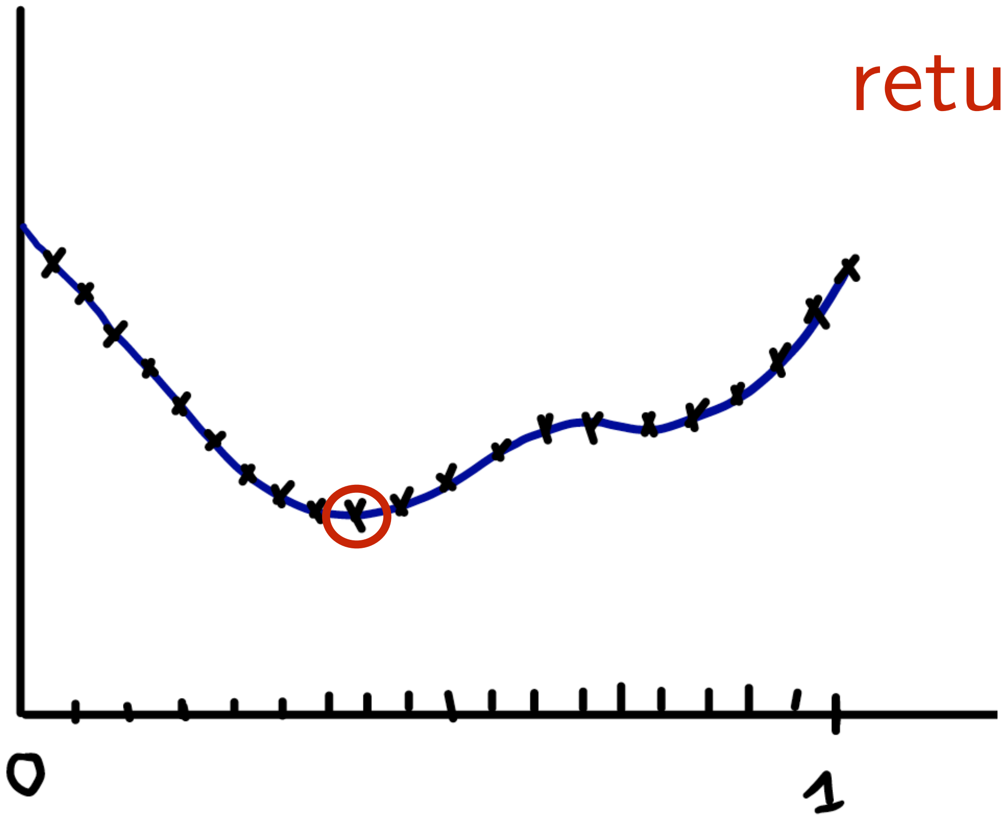
if n=1, which simple approach could you use to minimize:

$$f : [0,1] \to \mathbb{R} \quad ?$$

set a regular grid on [0,1]

evaluate on f all the points of the grid

return the lowest function value



Why is Optimization a non-trivial Problem?

Curse of dimensionality

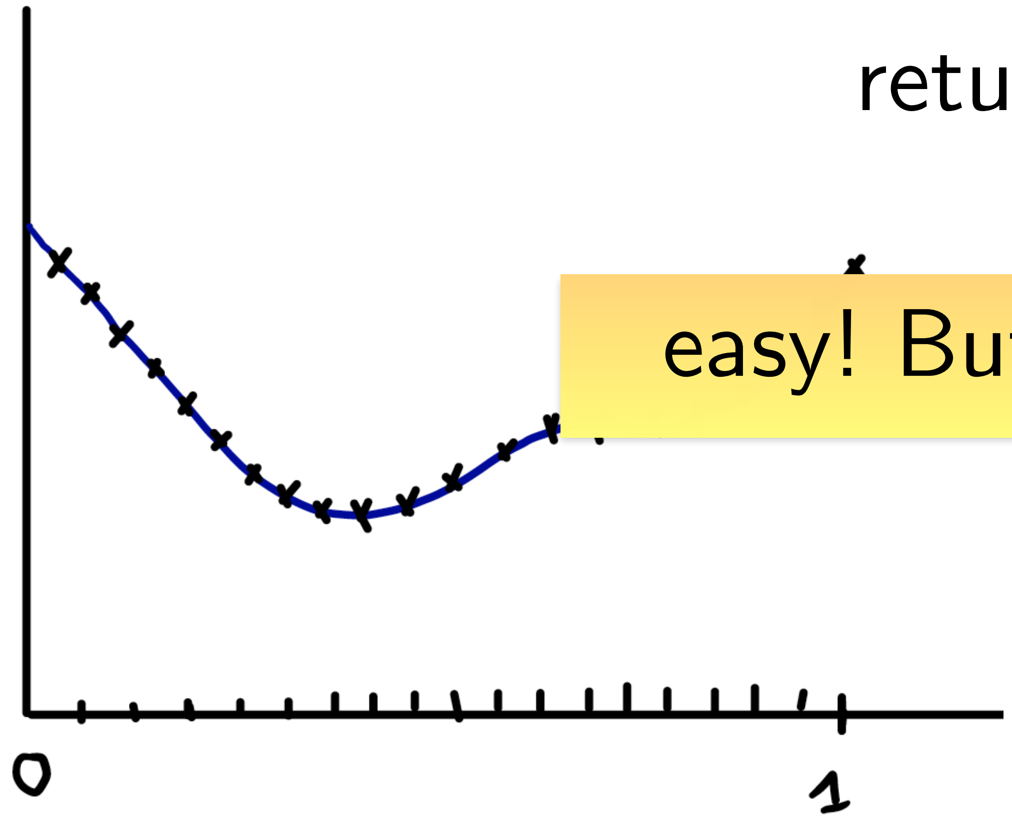
if n=1, which simple approach could you use to minimize:

$$f : [0, 1] \to \mathbb{R} \quad ?$$

set a regular grid on [0,1]

evaluate on f all the points of the grid

return the lowest function value



easy! But how does it scale when n increases?

1-D optimization is trivial

Curse of Dimensionality

The term curse of dimensionality (Richard Bellman) refers to problems caused by the rapid increase in volume associated with adding extra dimensions to a (mathematical) space.

Example: Consider placing 100 points onto a real interval, say [0,1].

How many points would you need to get a similar coverage (in terms of distance between adjacent points) in dimension 10?

Curse of Dimensionality

The term curse of dimensionality (Richard Bellman) refers to problems caused by the rapid increase in volume associated with adding extra dimensions to a (mathematical) space.

Example: Consider placing 100 points onto a real interval, say $[0,1]$. To get similar coverage, in terms of distance between adjacent points, of the 10-dimensional space $[0,1]^{10}$ would require $100^{10} = 10^{20}$ points. A 100 points appear now as isolated points in a vast empty space.

grid search

Consequence: a search policy (e.g. exhaustive search) that is valuable in small dimensions might be useless in moderate or large dimensional search spaces.

Curse of Dimensionality

How long would it take to evaluate 10^{20} points?

Curse of Dimensionality

How long would it take to evaluate 10^{20} points?

```
import timeit
timeit.timeit('import numpy as np ;
np.sum(np.ones(10)*np.ones(10))', number=1000000)
> 7.0521080493927
```

7 seconds for 10^6 evaluations of $f(x) = \sum_{i=1}^{10} x_i^2$

We would need more than 10^8 days for evaluating 10^{20} points

[As a reference: origin of human species: roughly 6 x 10^8 days]

Separability

Given $x = (x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots x_n)$ denote

$$x^{\neg i} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) \in \mathbb{R}^{n-1}$$

$$f_{x^{\neg i}}(y) = f(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_n)$$

The function $f_{x^{\neg i}}(y)$ is a 1-D function which is a cut of f along the coordinate i.

Definition: A function f is **separable** if for all i, for all $x, \bar{x}$

$$\mathrm{argmin}_y f_{x^{\neg i}}(y) = \mathrm{argmin}_y f_{\bar{x}^{\neg i}}(y)$$

$\rightarrow$ *the optimum along the coordinate i, does not depend on how the other coordinates are fixed.*

a weak definition of separability

Lemma: Given $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathrm{Im}(f) \to \mathbb{R}$ strictly increasing. If f is **separable** then $g \circ f$ is separable.

Proof: $\underset{\mathbb{R}}{y} \xmapsto{h} \overset{\mathbb{R}}{h(y)}$ Let $g : \mathrm{Im}(h) \to \mathbb{R}$ strict increasing

$$\boxed{\operatorname*{argmin}_y h(y) = \operatorname*{argmin}_y g \circ f(y)}$$

Let $\bar{x} \in \operatorname*{argmin}_y h(y)$ $h(\bar{x}) \leq h(y) \quad \forall y$

Since g strict increasing

$g \circ h(\bar{x}) \leq g \circ h(y) \quad \forall y$

$\Rightarrow \bar{x} \in \operatorname{argmin} g \circ h$

Let $\bar{x} \in \underset{y}{\arg\min}\ g\circ h(y)$

$$g\circ h(\bar{x}) \le g\circ h(y) \quad \forall y$$

generalized inverse

$$\overset{g^{-1}}{\Longrightarrow} \quad g^{-1}\big(g\circ h(\bar{x})\big) \le g^{-1}\big(g\circ h(y)\big) \quad \forall y$$

$$\Rightarrow \quad h(\bar{x}) \le h(y) \quad \forall y$$

$$\Rightarrow \quad \bar{x} \in \arg\min h$$

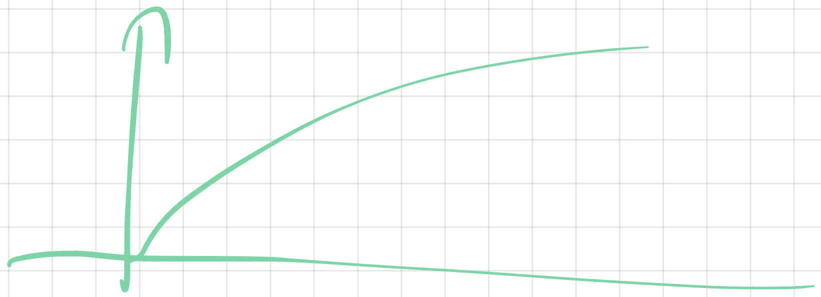
Since the argmin is preserved when composing with g strict increasing to the left, then if f is separable, $g\circ f$ is separable.

Example

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

$$\tilde{f}(x) = \left(\sum_{i=1}^{m} x_i^2\right)^{1/4} = g\circ f(x)$$

$$g(x) = \begin{cases} \mathrm{Im} f = \mathbb{R}_{\geq 0} \longrightarrow \mathbb{R} \\ x \longrightarrow x^{1/4} \end{cases}$$



$g \curvearrowright \qquad \operatorname{argmin} h = \operatorname{argmax} g \circ f$

Proposition: Let f be a **separable** then for all x

$$\operatorname{argmin} f(x_1, \ldots, x_n) = \left(\operatorname{argmin}_y f_{x\neg 1}(y), \ldots, \operatorname{argmin}_y f^n_{x\neg n}(y)\right)$$

and f can be optimized using n minimization along the coordinates.

Exercice: prove the proposition

Let us prove that $(\operatorname{argmin}_y f_{x\neg n}(y), \ldots, \operatorname{argmin}_y f_{x\neg n}(y)) \subset \operatorname{argmin} f$

$$\alpha_i \in \operatorname{argmin}_\alpha f(x_1, \ldots, x_{i-1}, \alpha, x_{i+1}, \ldots, x_n) \quad i = 1, \ldots, n$$

$$f(x) = f(x_1, \ldots, x_n) \underset{\substack{\uparrow \\ \text{by def of } \alpha_1}}{\geq} f(\alpha_1, x_2, \ldots, x_n) \underset{\substack{\uparrow \\ \text{by def of } \alpha_2}}{\geq} f(\alpha_1, \alpha_2, x_3, \ldots, x_n)$$

$$f(x) \geq \cdots \geq f(\alpha_1, \ldots, \alpha_n) \quad \forall x$$

$$(\alpha_1, \ldots, \alpha_n) \in \operatorname*{argmin}_{x \in \mathbb{R}^n} f .$$

The other inclusion is immediate:

$$\operatorname*{argmin}_{x} f \subset \left(\operatorname*{argmin}_{x_1} f(y), \ldots, \right)$$

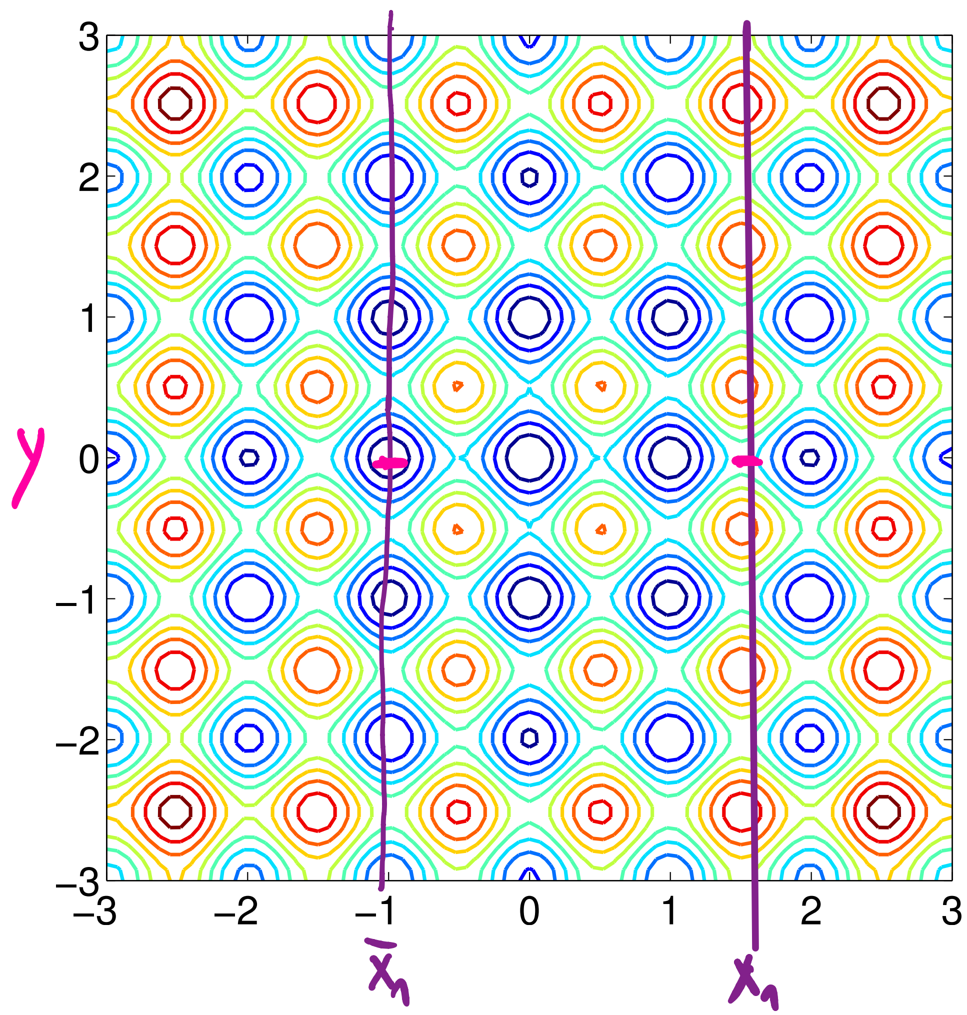
Example: Additively Decomposable Functions

Lemma: Let $f(x_1, \ldots, x_n) = \sum_{i=1}^{n} h_i(x_i)$ for h_i having a unique argmin.

Then f is separable. We say in this case that f is additively decomposable.

Example: Rastrigin function

$$f(x) = 10n + \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i))$$



Consequence

Consider $f(x) = \prod_{i=1}^{n} h_i(x_i)$ with $h_i(x_i) > 0$. Then it is separable.

Proof:

$$f(x) = \exp\left(\ln \prod_{i=1}^{n} h_i(x_i)\right)$$

$$= \exp\left(\sum_{i=1}^{n} \ln h_i(x_i)\right)$$

$$= g \circ \text{additively decomposable}$$

$g(x) = \exp(x)$ strict inc

$\tilde{f}(x) = \sum_{i=1}^{n} \ln h_i(x_i)$: additively decomposable

$\hookrightarrow$ separable.

Non-separable Problems

Separable problems are typically easy to optimize. Yet **difficult real-word problems are non-separable**.

One needs to be careful when evaluating optimization algorithms that not too many test functions are separable and if so that the *algorithms do not exploit separability*.

> ***Otherwise:*** *good performance on test problems will not reflect good performance of the algorithm to solve difficult problems*

Algorithms known to exploit separability:

> Many Genetic Algorithms (GA), Most Particle Swarm Optimization (PSO)

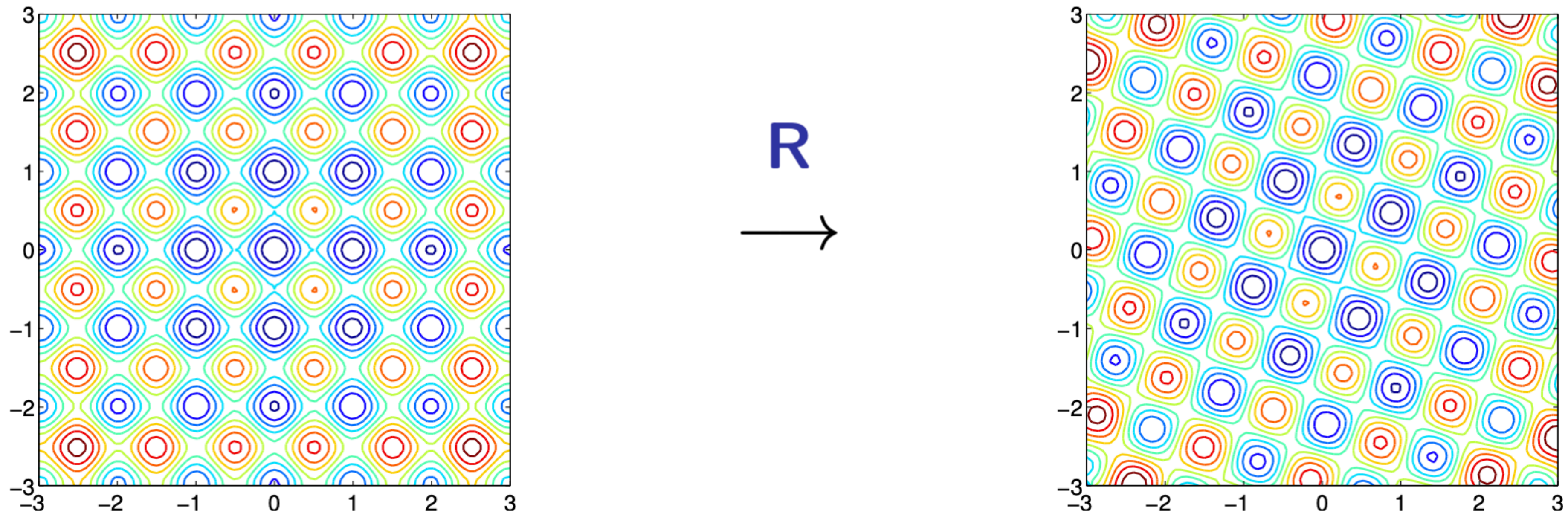
Non-separable Problems

Building a non-separable problem from a separable one

Rotating the coordinate system

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

R rotation matrix



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

2 Salomon (1996). "Reevaluating Genetic Algorithm Performance under Coordinate Rotation of Benchmark Functions; A survey of some theoretical and practical aspects of genetic algorithms." BioSystems, 39(3):263-278

Ill-conditioned Problems - Case of Convex-quadratic functions

Consider a strictly convex-quadratic function
$f(x) = \frac{1}{2}(x - x^\star)^\top H(x - x^\star)$ for $x = (x_1, \ldots, x_n)^\top \in \mathbb{R}^n$ and $x^\star \in \mathbb{R}^n$ with H a symmetric, positive, definite (SPD) matrix.

Remember that $H = \nabla^2 f(x)$.

The condition number of the matrix H (with respect to the Euclidean norm) is defined as

$$\mathrm{cond}(H) = \frac{\lambda_{\max}(H)}{\lambda_{\min}(H)}$$

with $\lambda_{\max}()$ and $\lambda_{\min}()$ being respectively the largest and smallest eigenvalues.

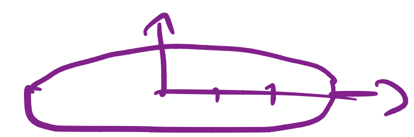
Ill-conditioned means a high condition number of the Hessian matrix H.

Consider now the specific case of the function $f(x) = \dfrac{1}{2}(x_1^2 + 9x_2^2)$

1. Compute its Hessian matrix, its condition number $H = \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix}$

$\mathrm{cond}(H) = 9$

2. Plots the level sets of f, relate the condition number to the axis ratio of the level sets of f



3. Generalize to a general convex-quadratic function

Real-world problems are often ill-conditioned.

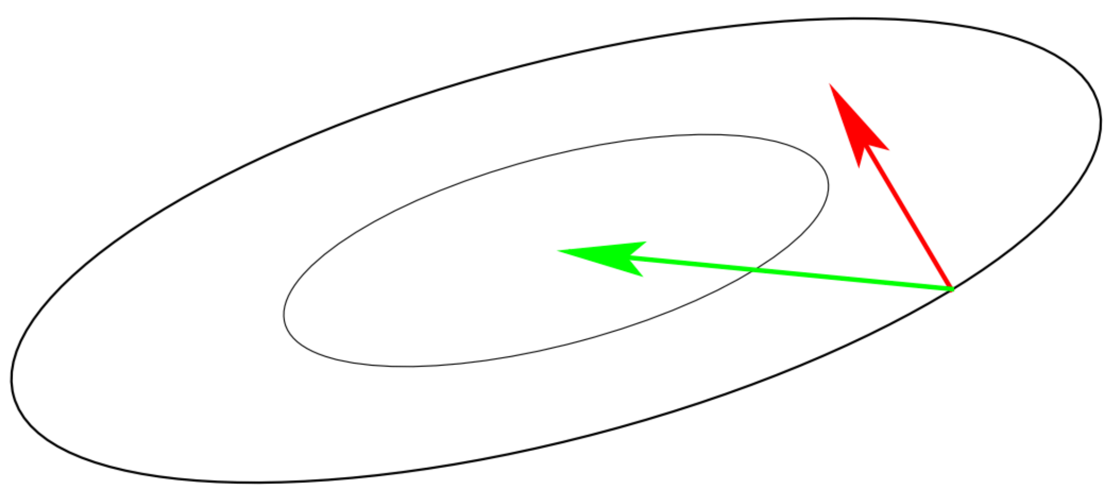
4. Why do you think it is the case? → physical variables optimized can live on different scales.

5. why are ill-conditioned problems difficult?

Ill-conditioned Problems

consider the curvature of the level sets of a function

ill-conditioned means "squeezed" lines of equal function value (high curvatures)



gradient direction $-f'(\boldsymbol{x})^{\mathrm{T}}$

Newton direction
$-\boldsymbol{H}^{-1}f'(\boldsymbol{x})^{\mathrm{T}}$

Condition number equals nine here. Condition numbers up to 10^{10} 10^6 are not unusual in real world problems.

DERIVATIVE FREE OPTIMIZATION 2024/2025.

CLASS 2.

Pure Random search (PRS)

Assume $f: \begin{cases} [-1,1]^n \longrightarrow \mathbb{R} \\ x=(x_1,\dots,x_n) \longrightarrow f(x) \end{cases}$

PRS:

Initialize $\quad xbest = \text{Unif}([-1,1]^n)$ (uniform)

WHILE NOT HAPPY (while stop criterion not met)

$\quad$ Sample $x \sim \text{Unif}([-1,1]^n)$

$\quad$ If $f(x) \leq f(xbest)$

$\quad\quad xbest \leftarrow x$

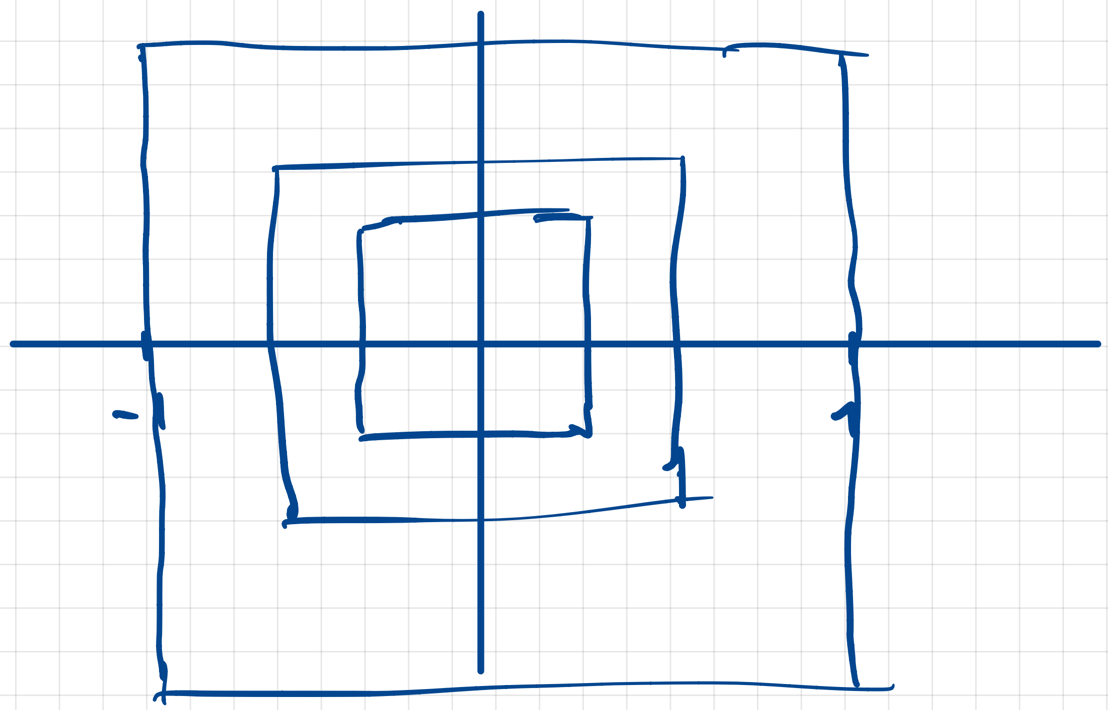
(see also Exercise 1)

Does this algorithm converge: Yes under mild assumptions on f
(need to have "volume" in a neighborhood of global optimum)

Simplified proof setting

$$f(x) = \|x\|_\infty = \max(|x_1|, \dots, |x_n|)$$

Level sets : $n = 2$



In exercice 1 : uniform samples $(v_0, v_1, \dots, v_t, \dots)$

$$v_i \sim \mathrm{Unif}([-1, 1]^n)$$

X_t : best solution at iteration t

$$f(X_t) = \min\{f(v_1), \dots, f(v_t)\}$$

By induction.

Prove that $\forall \varepsilon > 0 \quad \lim_{t \to +\infty} \mathbb{P}r(\|X_t\|_\infty \geq \varepsilon) = 0$

↳ give cv in probability.

$$\|X_t\|_\infty = \min \{ \|U_1\|_\infty, \dots, \|U_t\|_\infty \}$$

$$\{ \|X_t\|_\infty \geq \varepsilon \} = \{ \forall k = 1, \dots, t \ \|U_k\|_\infty \geq \varepsilon \} = \bigcap_{k=1}^{t} \{ \|U_k\|_\infty \geq \varepsilon \}$$

$$\mathbb{P}r(\{ \|X_t\|_\infty \geq \varepsilon \}) = \mathbb{P}r\left(\bigcap_{k=1}^{t} \{ \|U_k\|_\infty \geq \varepsilon \} \right)$$

$$= \prod_{k=1}^{t} \mathbb{P}r(\|U_k\|_\infty \geq \varepsilon) = \underbrace{\mathbb{P}r(\|U_1\|_\infty \geq \varepsilon)^t}_{= (1-\varepsilon^n)^t}$$

by ind of $\{U_{k'}, k' = 1, \dots k\}$

because U_k are identically distributed.

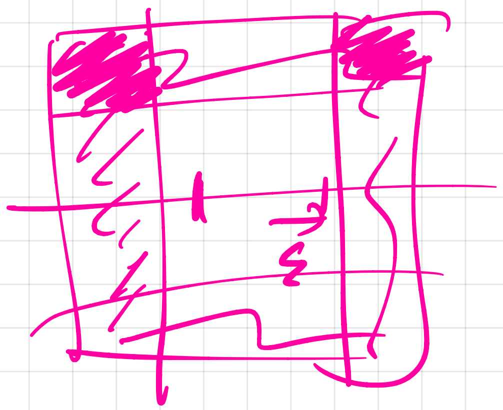
$$\mathbb{P}r\left(\|U_1\|_\infty > \varepsilon\right) \overset{?}{=} 1 - \mathbb{P}r\left(\|U_1\|_\infty \leq \varepsilon\right)$$

$$= 1 - \frac{\mathrm{Vol}\left(\{x \mid \|x\|_\infty \leq \varepsilon\}\right)}{\mathrm{Vol}\left(\{x \mid \|x\|_\infty \leq 1\}\right)}$$

$\not\parallel$

$\prod_{\text{coord}} \left(\text{each coordinate} > \varepsilon \right)$

$$= 1 - \frac{(2\varepsilon)^n}{2^n} = 1 - \varepsilon^n$$



$$\mathbb{P}r\left(\|X_t\|_\infty > \varepsilon\right) = \left(1 - \varepsilon^n\right)^t \xrightarrow[t \to +\infty]{} 0$$

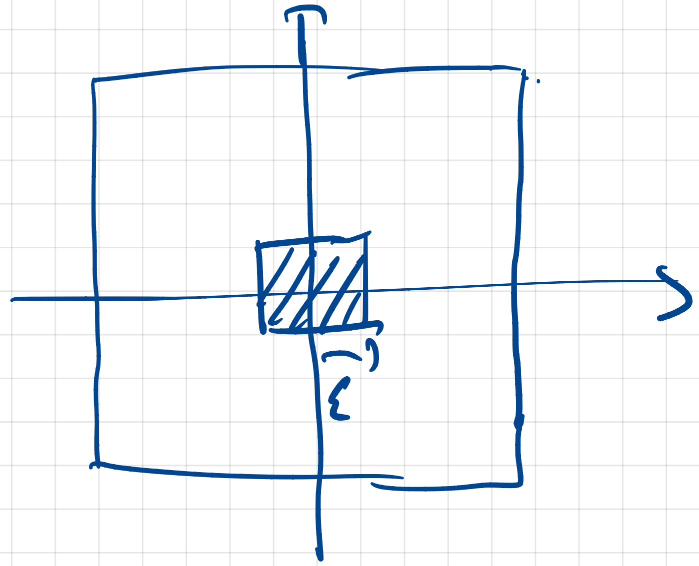
Hence PRS converges in probability to the optimum of f.

Let's look at how fast it converges.

$$T_\varepsilon = \inf\left\{ t \mid X_t \in B(x^*, \varepsilon) \right\}$$

(Ball for infinity norm)

$$\{x \mid \|x\|_\infty < \varepsilon\}$$



PRS is similar to game:

Sample U_t, win if $f(U_t) \leq \varepsilon$

$\iff U_t \in B(0, \varepsilon)$

Lose otherwise.

T_ε : time it takes to win this game.

Given a game with 2 outcomes win with proba p and lose with proba $(1-p)$, where the outcome is sampled randomly and independently [example: flip a coin], the time it takes to win a game is distributed according to a geometric distribution.

$$\mathbb{E}[T_\varepsilon] = \frac{1}{p}$$

Back to PRS. $p = \mathbb{P}r(\text{"win"}) = \mathbb{P}r(\|U_t\| \leq \varepsilon) = \varepsilon^n$

$$\mathbb{E}(T_\varepsilon) = \frac{1}{\varepsilon^n}$$

Is it fast ? No

To compare : linear cv $T_\varepsilon \sim n \log\left(\frac{1}{\varepsilon}\right)$ [gradient desc on L-smooth strongly convex]

The algorithm is "blind", does not take into account the information gathered on f, through the sampling of points to sample "better" solutions.

Part II: Algorithms

Landscape of Derivative Free Optimization Algorithms

Deterministic Algorithms

- Quasi-Newton with estimation of gradient (BFGS) [Broyden et al. 1970]
- Simplex downhill [Nelder and Mead 1965]
- Pattern search, Direct Search [Hooke and Jeeves 1961]
- Trust-region/Model Based methods (NEWUOA, BOBYQA) [Powell, 06,09]

Stochastic (randomized) search methods

- Evolutionary Algorithms (continuous domain)
  - Differential Evolution [Storn, Price 1997]
  - Particle Swarm Optimization [Kennedy and Eberhart 1995]
  - **Evolution Strategies, CMA-ES** [Rechenberg 1965, Hansen, Ostermeier 2001]
  - Estimation of Distribution Algorithms (EDAs) [Larrañaga, Lozano, 2002]
  - Cross Entropy Method (same as EDAs) [Rubinstein, Kroese, 2004]
  - Genetic Algorithms [Holland 1975, Goldberg 1989]
- Simulated Annealing [Kirkpatrick et al. 1983]

A Generic Template for Stochastic Search

Define $\{P_\theta : \theta \in \Theta\}$, a family of probability distributions on $\mathbb{R}^n$

Generic template to optimize $f : \mathbb{R}^n \to \mathbb{R}$

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

While not terminate

1. Sample $\underset{\in \mathbb{R}^n}{x_1}, \ldots, \underset{\in \mathbb{R}^n}{x_\lambda}$ according to P_θ
2. Evaluate $x_1, \ldots, x_\lambda$ on f
3. Update parameters $\theta \leftarrow F(\theta, x_1, \ldots, x_\lambda, f(x_1), \ldots, f(x_\lambda))$

the update of θ should drive P_θ to concentrate on the optima of f

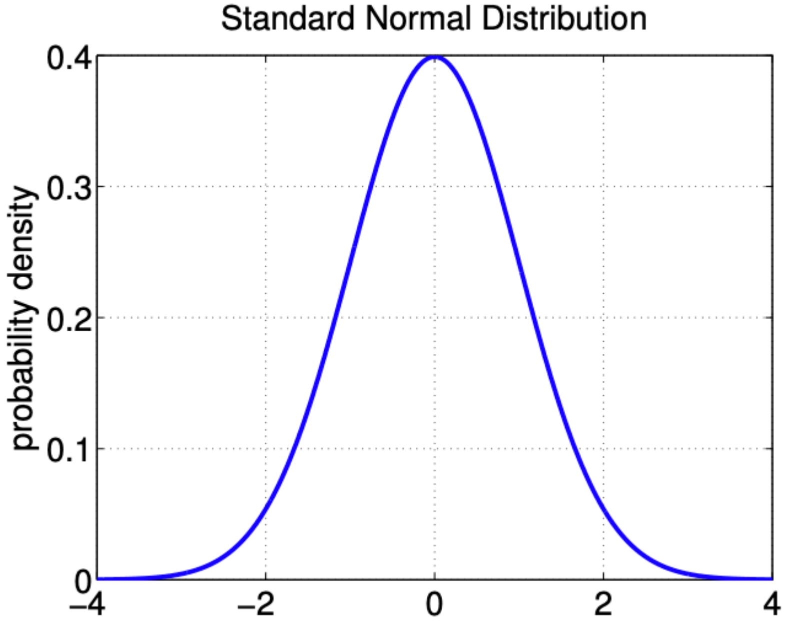
To obtain an optimization algorithm we need:

❶ to define $\{P_\theta, \theta \in \Theta\}$

❷ to define F the update function of θ

Which probability distribution to sample candidate solutions?

Normal distribution - 1D case



probability density of the 1-D standard normal distribution $\mathcal{N}(0,1)$

(expected (mean) value, variance) = (0,1)

$$p(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$$

General case

- Normal distribution $\mathcal{N}(m, \sigma^2)$

  (expected value, variance) $= (m, \sigma^2)$

  density: $p_{m,\sigma}(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right)$

- A normal distribution is entirely determined by its mean value and variance
- The family of normal distributions is closed under linear transformations: if X is normally distributed then a linear transformation $aX + b$ is also normally distributed
- **Exercice:** Show that $m + \sigma\mathcal{N}(0,1) = \mathcal{N}(m, \sigma^2)$

$m + \sigma \mathcal{N}(0,1)$ is normally distributed

We only need to identify its mean and variance:

$$\mathbb{E}\left(m + \sigma \mathcal{N}(0,1)\right) \underset{\substack{\text{by linearity}\\ \text{on } \mathbb{E}}}{=} m + \sigma \underbrace{\mathbb{E}(\mathcal{N}(0,1))}_{=0} = m$$

$$\begin{aligned}
&\mathrm{Var}\left(m + \sigma \mathcal{N}(0,1)\right) \\
&= \mathbb{E}\left(\left(m + \sigma \mathcal{N}(0,1) - m\right)^2\right) \\
&= \mathbb{E}\left(\sigma^2 \mathcal{N}(0,1)^2\right) \\
&= \sigma^2 \underbrace{\mathbb{E}(\mathcal{N}(0,1)^2)}_{=1} = \sigma^2
\end{aligned}$$

$$\Rightarrow \quad m + \sigma \mathcal{N}(0,1) \sim \mathcal{N}(m, \sigma^2)$$

$$\mathbb{P}\left(m + \sigma \mathcal{N}(0,1) \leq t \right) = \mathbb{P}r\left(\mathcal{N}(0,1) \leq \frac{t-m}{\sigma} \right)$$

$$= \int_{-\infty}^{\frac{t-m}{\sigma}} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx$$

$dy = \sigma dx$

$y = \sigma x + m$

$$= \int_{-\infty}^{t} \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\left(\frac{y-m}{2\sigma}\right)^2\right) dy$$

$x = \frac{t-m}{\sigma} \quad y = t$

Generalization to n Variables: Independent Case

Assume X1 ~ $\mathcal{N}(\mu_1, \sigma_1^2)$ denote its density $p(x_1) = \frac{1}{Z_1}\exp\left(-\frac{1}{2\sigma_1^2}(x_1-\mu_1)^2\right)$

Assume X2~ $\mathcal{N}(\mu_2, \sigma_2^2)$ denote its density $p(x_2) = \frac{1}{Z_2}\exp\left(-\frac{1}{2\sigma_2^2}(x_2-\mu_2)^2\right)$

Assume X1 and X2 are **independent**, then (X1,X2) is a Gaussian vector with

$$p(x_1, x_2) =$$

Generalization to n Variables: Independent Case

Assume X1 ~ $\mathcal{N}(\mu_1, \sigma_1^2)$ denote its density $p(x_1) = \frac{1}{Z_1}\exp\Big(-\frac{1}{2\sigma_1^2}(x_1-\mu_1)^2\Big)$

Assume X2~ $\mathcal{N}(\mu_2, \sigma_2^2)$ denote its density $p(x_2) = \frac{1}{Z_1}\exp\Big(-\frac{1}{2\sigma_2^2}(x_2-\mu_2)^2\Big)$

Assume X1 and X2 are **independent**, then (X1,X2) is a Gaussian vector with

$$p(x_1, x_2) = p(x_1)p(x_2) = \frac{1}{Z_1Z_2}\exp\Big(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\Big)$$

with $x = (x_1, x_2)^T$ $\mu = (\mu_1, \mu_2)^T$ $\Sigma = \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix}$

Generalization to n Variables: Independent Case

Assume X1 $\sim \mathcal{N}(\mu_1, \sigma_1^2)$ denote its density $p(x_1) = \frac{1}{Z_1} \exp\left(-\frac{1}{2\sigma_1^2}(x_1 - \mu_1)^2 \right)$

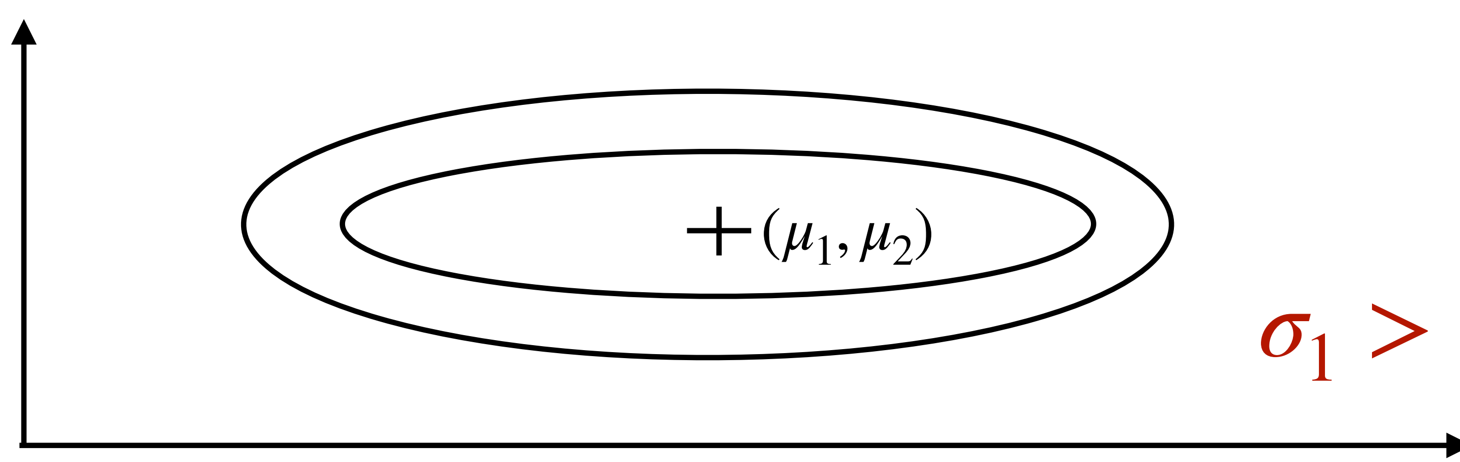
Assume X2$\sim \mathcal{N}(\mu_2, \sigma_2^2)$ denote its density $p(x_2) = \frac{1}{Z_1} \exp\left(-\frac{1}{2\sigma_2^2}(x_2 - \mu_2)^2 \right)$

Assume X1 and X2 are **independent**, then (X1,X2) is a Gaussian vector with

$$p(x_1, x_2) = p(x_1)p(x_2) = \frac{1}{Z_1 Z_2} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu) \right)$$

$\rightarrow -\frac{1}{2}\left[\frac{(x_1-\mu_1)^2}{\sigma_1^2} + \frac{(x_2-\mu_2)^2}{\sigma_2^2} \right]$

with $x = (x_1, x_2)^T$ $\mu = (\mu_1, \mu_2)^T$ $\Sigma = \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix}$



$\sigma_1 > \sigma_2$

Generalization to n Variables: General Case

Gaussian Vector - Multivariate Normal Distribution

A random vector $X = (X_1, \ldots, X_n) \in \mathbb{R}^n$ is a Gaussian vector (or multivariate normal) if and only if for all real numbers $a_1, \ldots, a_n$, the random variable $a_1 X_1 + \ldots + a_n X_n$ has a normal distribution.

Gaussian Vector - Multivariate Normal Distribution

A random variable following a 1-D normal distribution is determined by its mean value $\boldsymbol{m}$ and variance σ^2.

In the n-dimensional case it is determined by its mean vector and covariance matrix

Covariance Matrix

If the entries in a vector $\boldsymbol{X} = (X_1, \ldots, X_n)^T$ are random variables, each with finite variance, then the covariance matrix Σ is the matrix whose (i, j) entries are the covariance of (X_i, X_j)

$$\Sigma_{ij} = \mathrm{cov}(X_i, X_j) = \mathrm{E}\left[(X_i - \mu_i)(X_j - \mu_j)\right]$$

where $\mu_i = \mathrm{E}(X_i)$. Considering the expectation of a matrix as the expectation of each entry, we have

$\Sigma_{ii} = Var(x_i)$

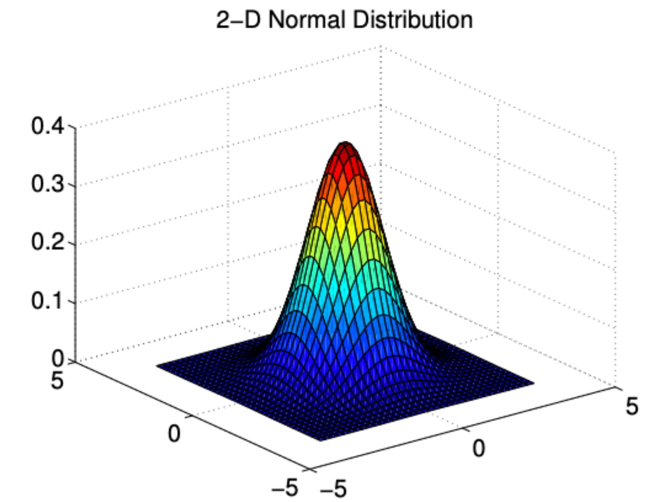
$$\Sigma = \mathrm{E}[(X - \mu)(X - \mu)^T]$$

Σ is symmetric, positive definite

Density of a n-dimensional Gaussian vector $\mathcal{N}(m, C)$:

$$p_{\mathcal{N}(m.C)}(x) = \frac{1}{(2\pi)^{n/2}|C|^{1/2}} \exp\left(-\frac{1}{2}(x-m)^\top C^{-1}(x-m)\right)$$

$|C| = \det(C)$



The mean vector m:

determines the displacement

is the value with the largest density

the distribution is symmetric around the mean

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

The covariance matrix:

determines the geometrical shape (see next slides)

Geometry of a Gaussian Vector

Consider a Gaussian vector $\mathcal{N}(m, C)$, remind that lines of equal densities are given by:

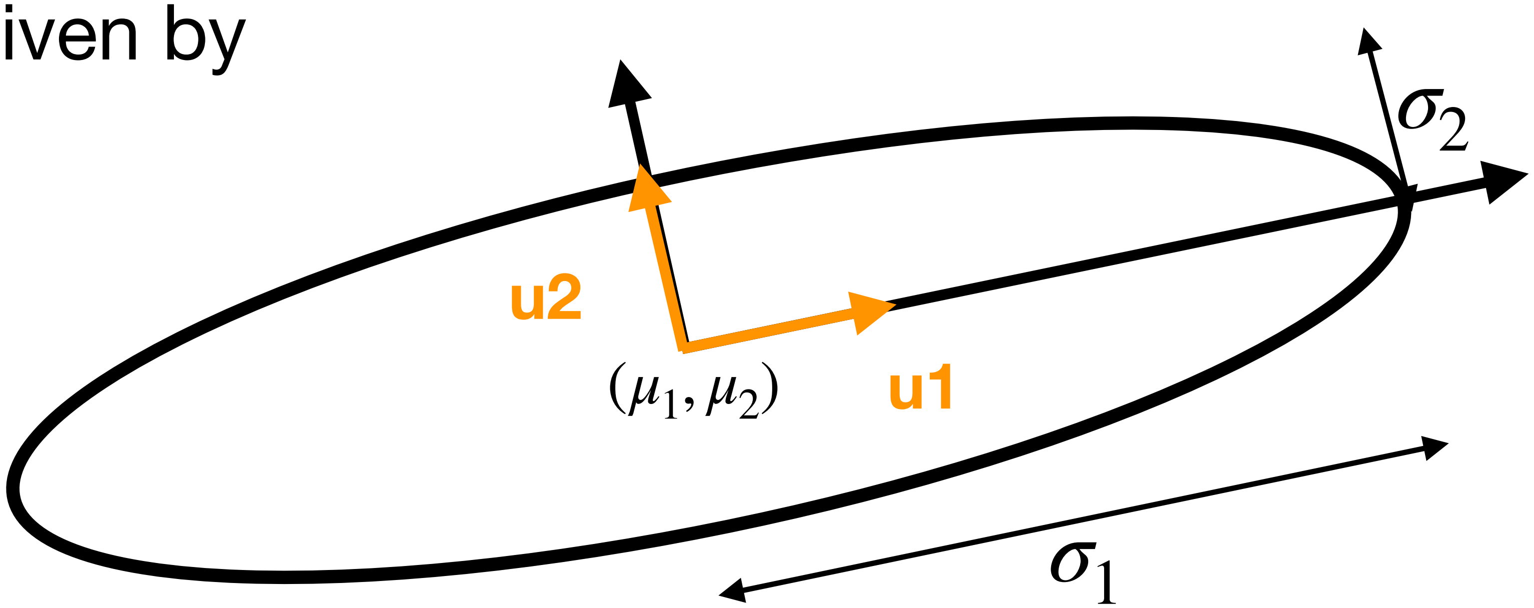
$$\{x \,|\, \Delta^2 = (x-m)^T C^{-1}(x-m) = \text{cst}\}$$

Decompose $C = U \Lambda U^\top$ with U orthogonal, i.e.

$$C = \begin{pmatrix} u_1 & u_2 \\ | & | \end{pmatrix} \begin{pmatrix} \sigma_1^2 & 0 \\ 0 | & \sigma_2^2 \end{pmatrix} \begin{pmatrix} u_1 & - \\ u_2 & - \end{pmatrix}$$

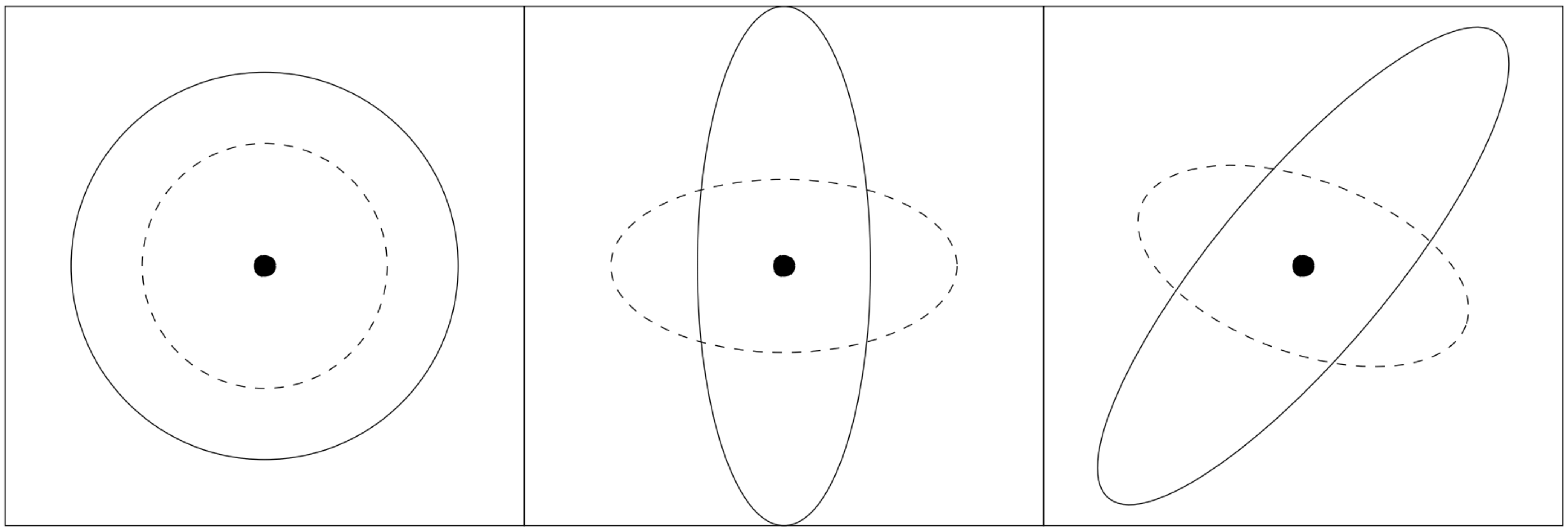
Let $Y = U^\top (x-m)$, then in the coordinate system, (u1,u2), the lines of equal densities are given by

$$\{x \,|\, \Delta^2 = \frac{Y_1^2}{\sigma_1^2} + \frac{Y_2^2}{\sigma_2^2} = \text{cst}\}$$



...any covariance matrix can be uniquely identified with the iso-density ellipsoid $\{\boldsymbol{x} \in \mathbb{R}^n \,|\, (\boldsymbol{x} - \boldsymbol{m})^{\mathrm{T}} \mathbf{C}^{-1} (\boldsymbol{x} - \boldsymbol{m}) = 1\}$

Lines of Equal Density



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

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

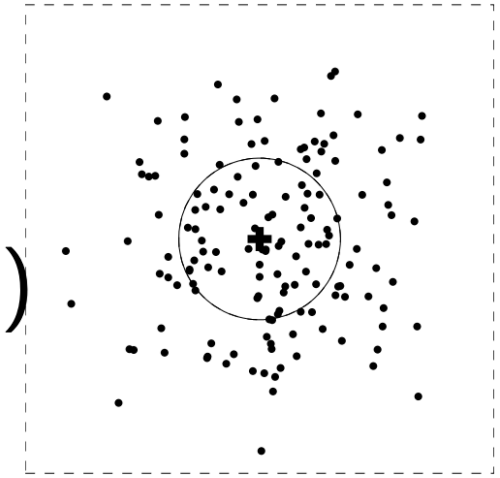
$\mathcal{N}(\boldsymbol{m}, \mathbf{C}) \sim \boldsymbol{m} + \mathbf{C}^{\frac{1}{2}} \mathcal{N}(\mathbf{0}, \mathbf{I})$
$(n^2 + n)/2$ degrees of freedom
components are correlated

where $\mathbf{I}$ is the identity matrix (isotropic case) and $\mathbf{D}$ is a diagonal matrix (reasonable for separable problems) and $\mathbf{A} \times \mathcal{N}(\mathbf{0}, \mathbf{I}) \sim \mathcal{N}(\mathbf{0}, \mathbf{A}\mathbf{A}^{\mathrm{T}})$ holds for all $\mathbf{A}$.

Evolution Strategies

New search points are sampled normally distributed

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



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

$x_i \sim \mathcal{N}(m, \sigma^2 C)$

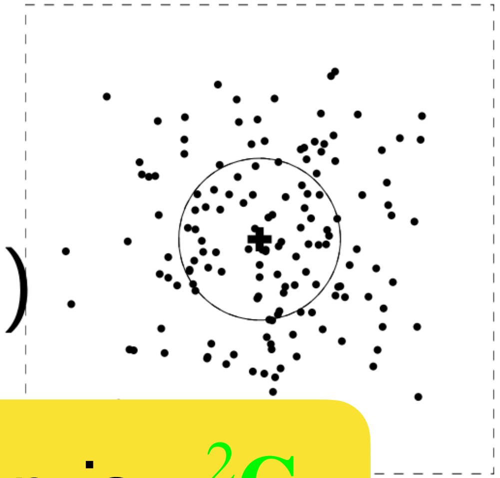
where

- 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

here, all new points are sampled with the same parameters

Evolution Strategies

New search points are sampled normally distributed



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

In fact, the covariance matrix of the sampling distribution is $\sigma^2\mathbf{C}$ but it is convenient to refer to $\mathbf{C}$ as the covariance matrix (it is a covariance matrix but not of the sampling distribution)

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

here, all new points are sampled with the same parameters

How to update the different parameters $m, \sigma, \mathbf{C}$?

1. **Adapting the mean m**
2. Adapting the step-size σ
3. Adapting the covariance matrix C

Update the Mean: a Simple Algorithm the (1+1)-ES

Notation and Terminology:

one solution kept from one iteration to the next

(1+1)-ES

one new solution (offspring) sampled at each iteration

The + means that we keep the best between current solution and new solution, we talk about *elitist* selection

(1+1)-ES algorithm (update of the mean)

sample one candidate solution from the mean $\mathbf{m}$

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

if $\mathbf{x}$ is better than $\mathbf{m}$ (i.e. if $f(\mathbf{x}) \leq f(\mathbf{m})$), select $\mathbf{m}$

$$\mathbf{m} \leftarrow \mathbf{x}$$

The (1+1)-ES algorithm is a simple algorithm, yet:

- the elitist selection is not robust to outliers

we cannot loose solutions accepted by "chance", for instance that look good because the noise gave it a low function value

- there is no population (just a single solution is sampled) which makes it less robust

In practice, one should rather use a:

$$(\mu/\mu, \lambda)\text{-ES}$$

The μ best solutions are selected and recombined (to form the new mean)

λ solutions are sampled at each iteration

The $(\mu/\mu, \lambda)$-ES - Update of the Mean Vector

Given the i-th solution point $\boldsymbol{x}_i = \boldsymbol{m} + \sigma \underbrace{\boldsymbol{y}_i}_{\sim\mathcal{N}(\mathbf{0},\mathbf{C})}$ $i=1,\dots,\lambda$

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

Notation: we denote $\boldsymbol{y}_{i:\lambda}$ the vector such that $\boldsymbol{x}_{i:\lambda} = \boldsymbol{m} + \sigma \boldsymbol{y}_{i:\lambda}$

Exercice: realize that $\boldsymbol{y}_{i:\lambda}$ is generally not distributed as $\mathcal{N}(\mathbf{0}, \mathbf{C})$

The new mean reads

$$\boldsymbol{m} \leftarrow \sum_{i=1}^{\mu} w_i \, \boldsymbol{x}_{i:\lambda}$$

where

$$w_1 \geq \cdots \geq w_\mu > 0, \quad \sum_{i=1}^{\mu} w_i = 1, \quad \frac{1}{\sum_{i=1}^{\mu} {w_i}^2} =: \mu_w \approx \frac{\lambda}{4}$$

(typically $\mu = \frac{\lambda}{2}$)

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

What changes in the previous slide if instead of optimizing f, we optimize $g \circ f$ where $g : \mathrm{Im}(f) \to \mathbb{R}$ is strictly increasing?

$f(x) = x^2$

$g \circ f(x) = (x^2)^{1/4}$

$= \sqrt{|x|}$

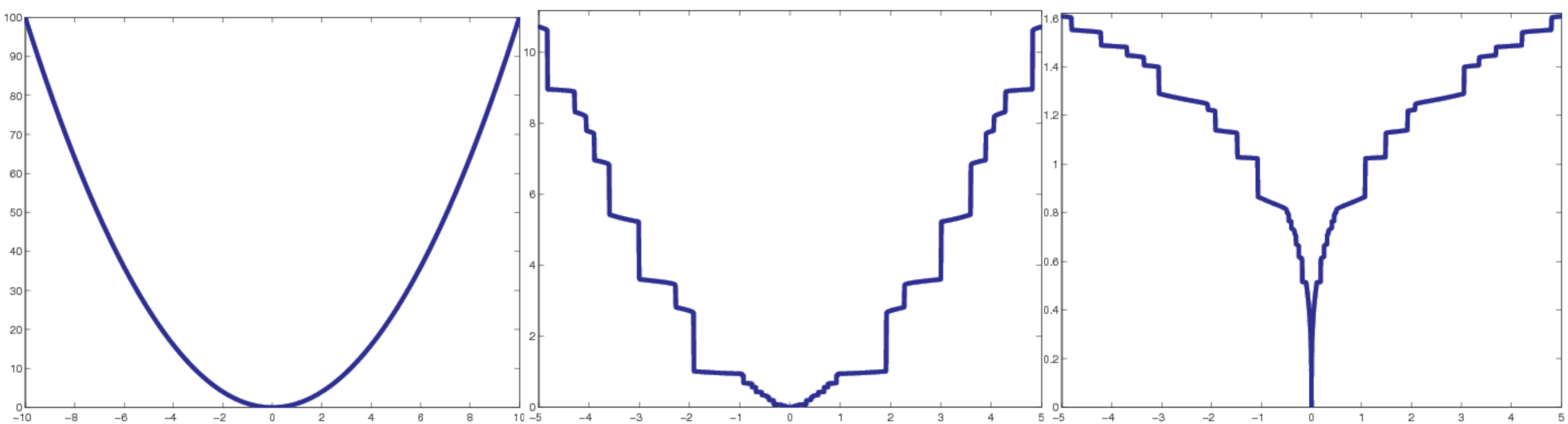
$g(x) = x \in \mathbb{R}^+ \to x^{1/4}$

Invariance Under Monotonically Increasing Functions

Comparison-based/ranking-based algorithms:

Update of all parameters uses only the ranking:

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



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

for all $g : \mathrm{Im}(f) \to \mathbb{R}$ strictly increasing

A Template for Comparison-based Stochastic Search

Define $\{P_\theta : \theta \in \Theta\}$, a family of probability distributions on $\mathbb{R}^n$

Generic template to optimize $f : \mathbb{R}^n \to \mathbb{R}$

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

While not terminate

1. Sample $x_1, \ldots, x_\lambda$ according to P_θ
2. Evaluate $x_1, \ldots, x_\lambda$ on f
3. Rank the solutions and find π the permutation such

$$f(x_{\pi(1)}) \leq f(x_{\pi(2)}) \leq \ldots \leq f(x_{\pi(\lambda)})$$

4. Update parameters $\theta \leftarrow F(\theta, x_1, \ldots, x_\lambda, \pi)$

How to update the different parameters $m, \sigma, \mathbf{C}$?

1. Adapting the mean m
2. **Adapting the step-size σ**
3. Adapting the covariance matrix C

Why Step-size Adaptation?

Assume a (1+1)-ES algorithm with fixed step-size σ (and $C = I_d$) optimizing the function $f(x) = \sum_{i=1}^{n} x_i^2 = \|x\|^2$.

Initialize $\mathbf{m}, \sigma$

While *(stopping criterion not met)*

sample new solution:

$$\mathbf{x} \leftarrow \mathbf{m} + \sigma \mathcal{N}(0, I_d)$$

if $f(\mathbf{x}) \leq f(\mathbf{m})$

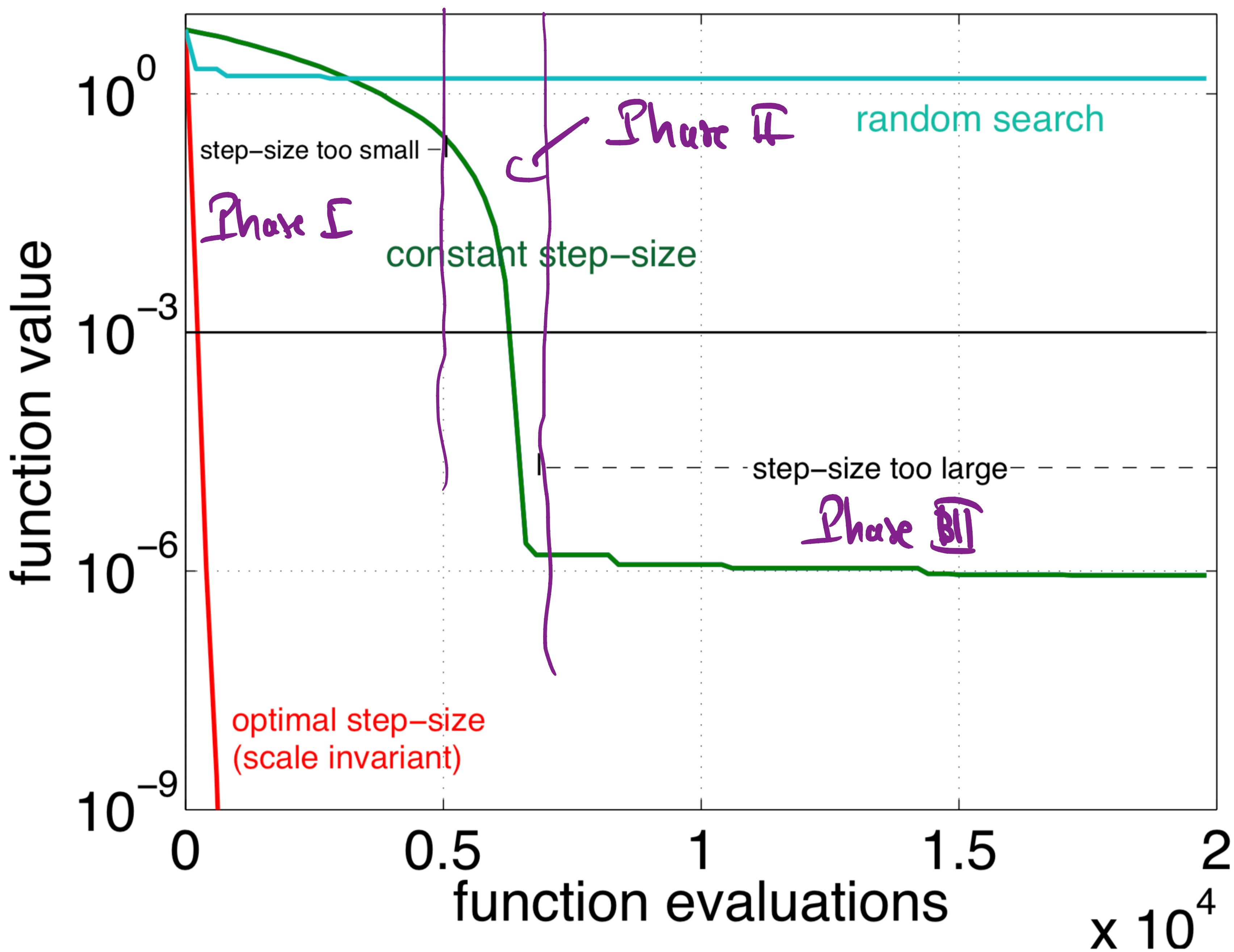
$$\mathbf{m} \leftarrow \mathbf{x}$$

$\sigma \leftarrow \sigma \times 1{,}5$

else $\sigma \leftarrow \sigma \times (1{,}5)^{-1/4}$ → $\frac{1}{5}$ success-rule

What will happen if you look at the convergence of $f(m)$?

Why Step-size Adaptation?



(1+1)-ES
(red & green)

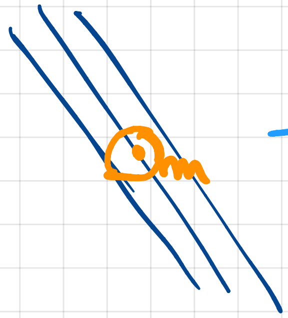
$$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2$$

in $[-2.2, 0.8]^n$
for $n = 10$

red curve: (1+1)-ES with optimal step-size (see later)

green curve: (1+1)-ES with constant step-size ($\sigma = 10^{-3}$)

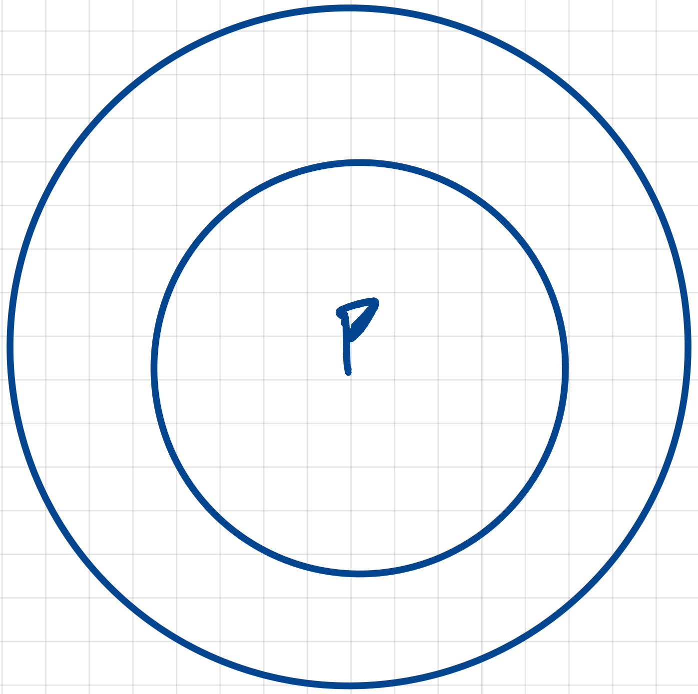
Phase I



→ sample very close to $m_t = \begin{pmatrix} \vdots \\ \vdots \end{pmatrix}$ because step-size very small

I progress but slowly because step-size is small.

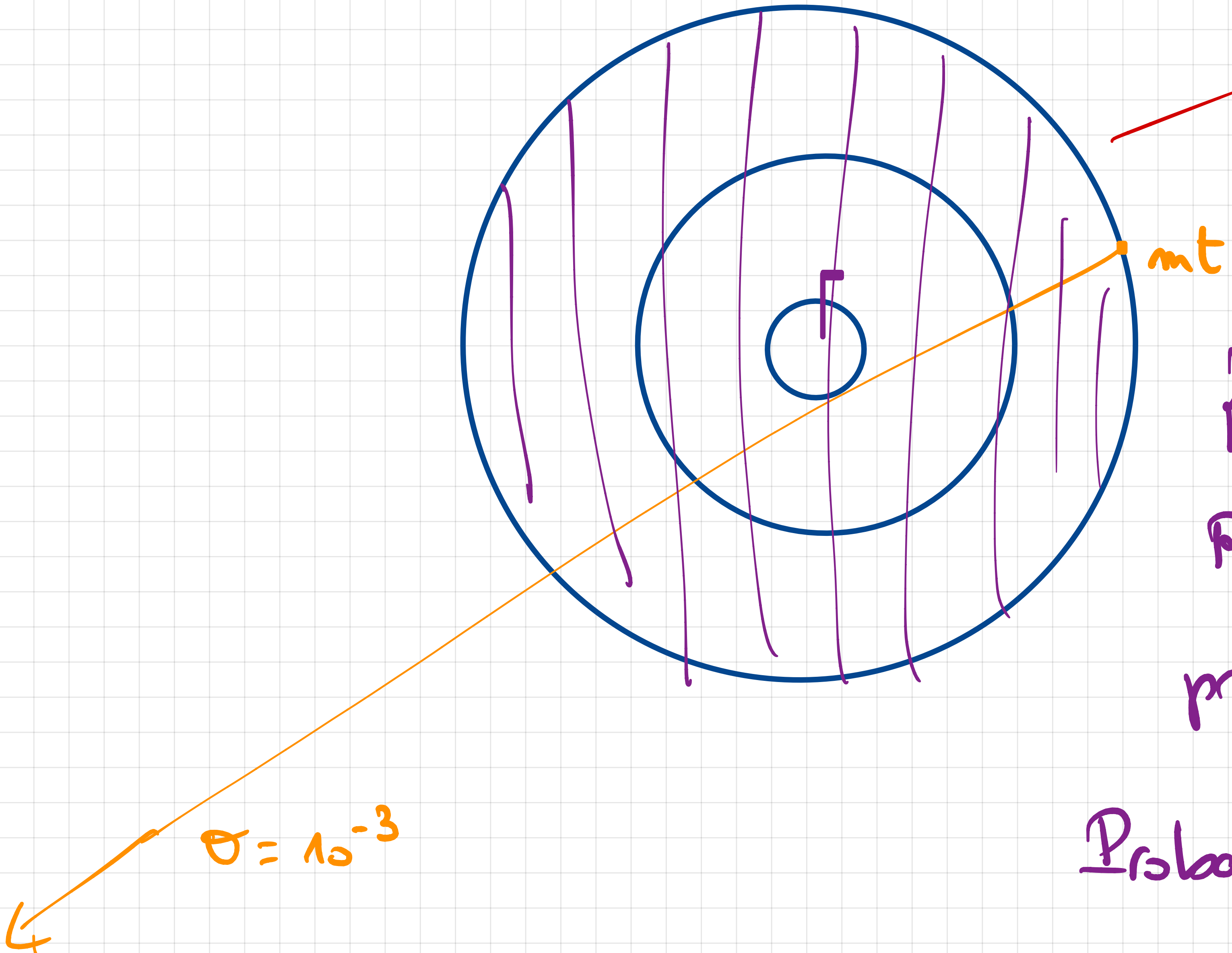
$\mathbb{P}$(sample better solution) $\approx \frac{1}{2}$



↳ Step-size too small

with bigger, faster progress

Phase III



I would like to increase the step-size

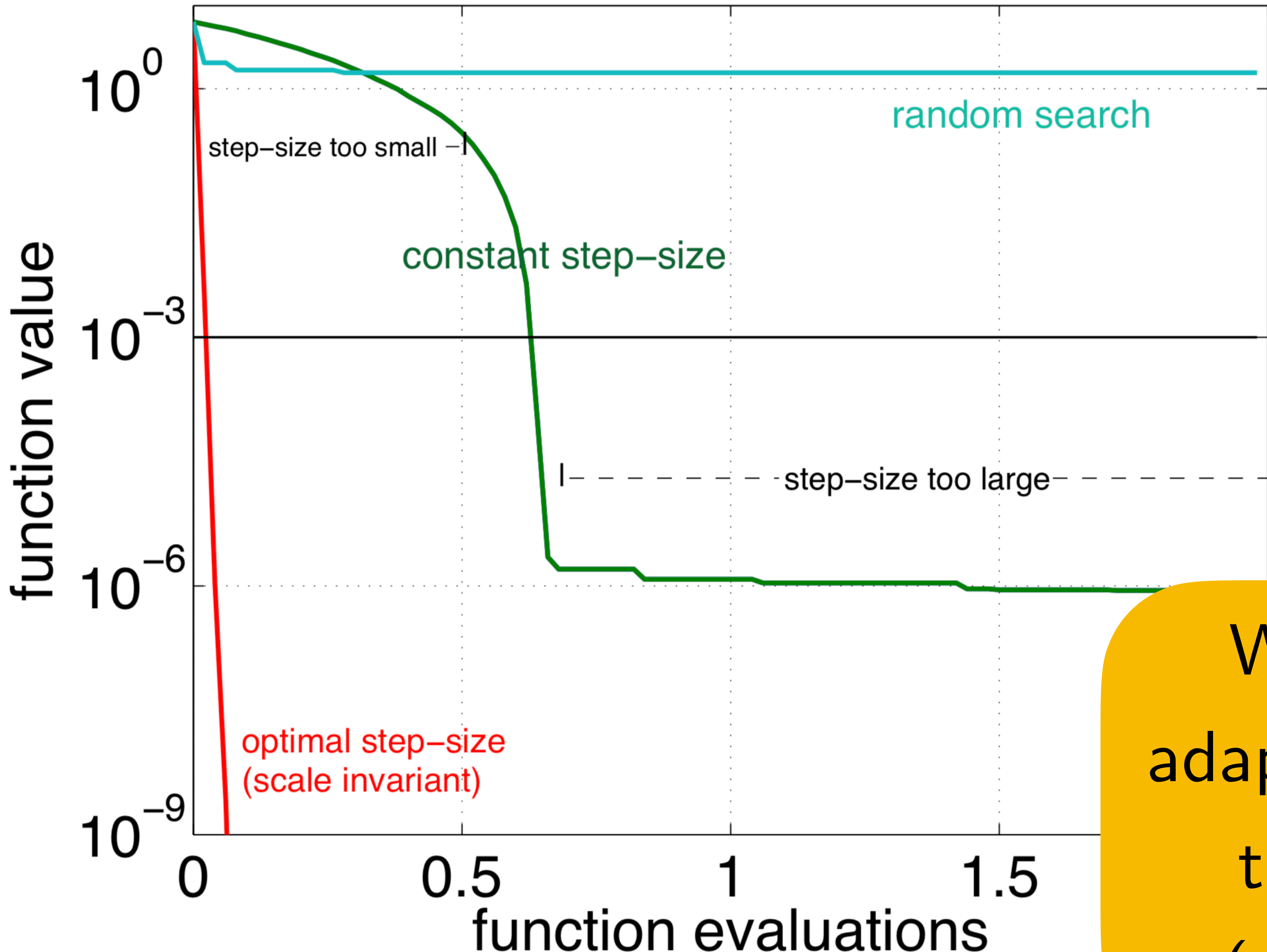
$\{x \mid f(x) \leq f(mt)\}$

Region where we need to sample to progress.

Probability to sample in this region is very small, because $\sigma >> \|mt - x^*\|$

Phase II : The step size has the right order of magnitude compared to $\|m_t - x^*\|$, progress is close to optimal.

Why Step-size Adaptation?



(1+1)-ES
(red & green)

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

We need step-size adaptation to approach the optimum fast (converge linearly)

red curve: (1+1)-ES with optimal step-size (see later)

green curve: (1+1)-ES with constant step-size ($\sigma = 10^{-3}$)

Methods for Step-size Adaptation

1/5th success rule, typically applied with "+" selection

[Rechenberg, 73][Schumer and Steiglitz, 78][Devroye, 72]

σ-self adaptation, applied with "," selection [Schwefel, 81]

random variation is applied to the step-size and the better one, according to the objective function value, is selected

path-length control or Cumulative step-size adaptation (CSA), applied with "," selection

[Ostermeier et al. 84][Hansen, Ostermeier, 2001]

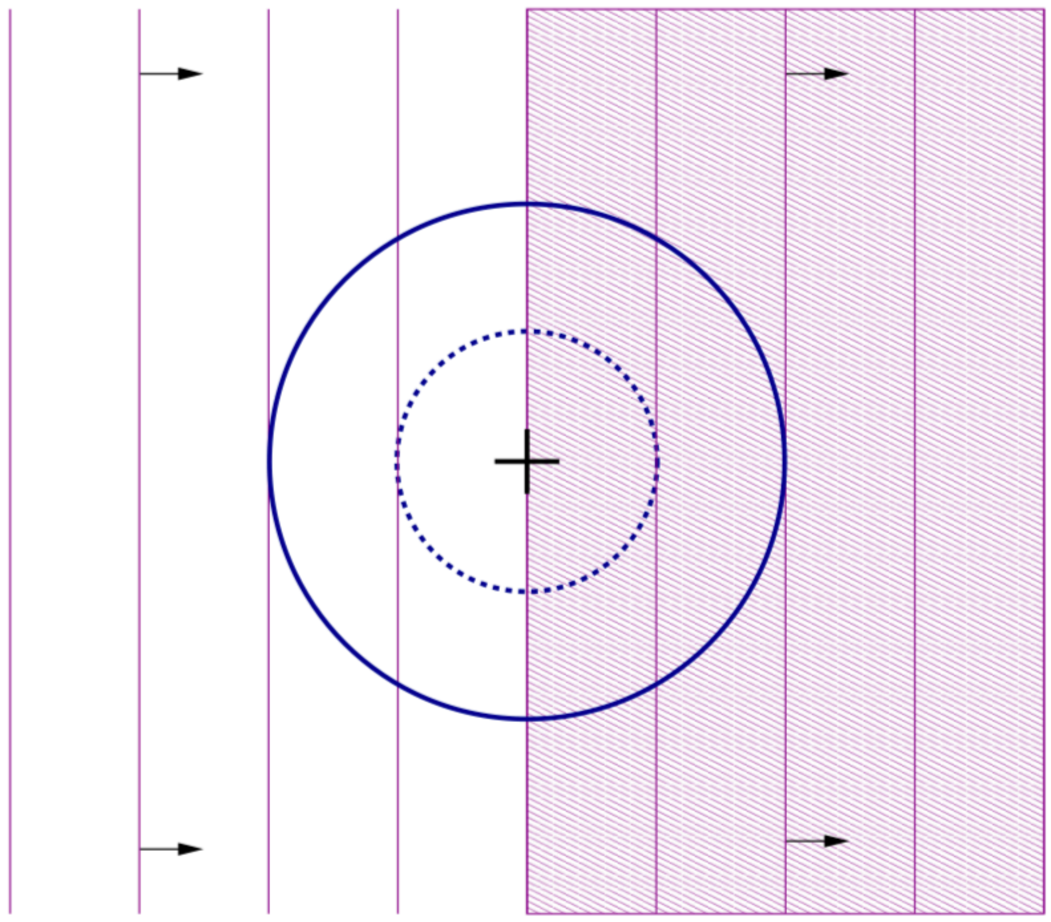
two-point adaptation (TPA), applied with "," selection [Hansen 2008]

test two solutions in the direction of the mean shift, increase or decrease accordingly the step-size

Step-size control: 1/5th Success Rule

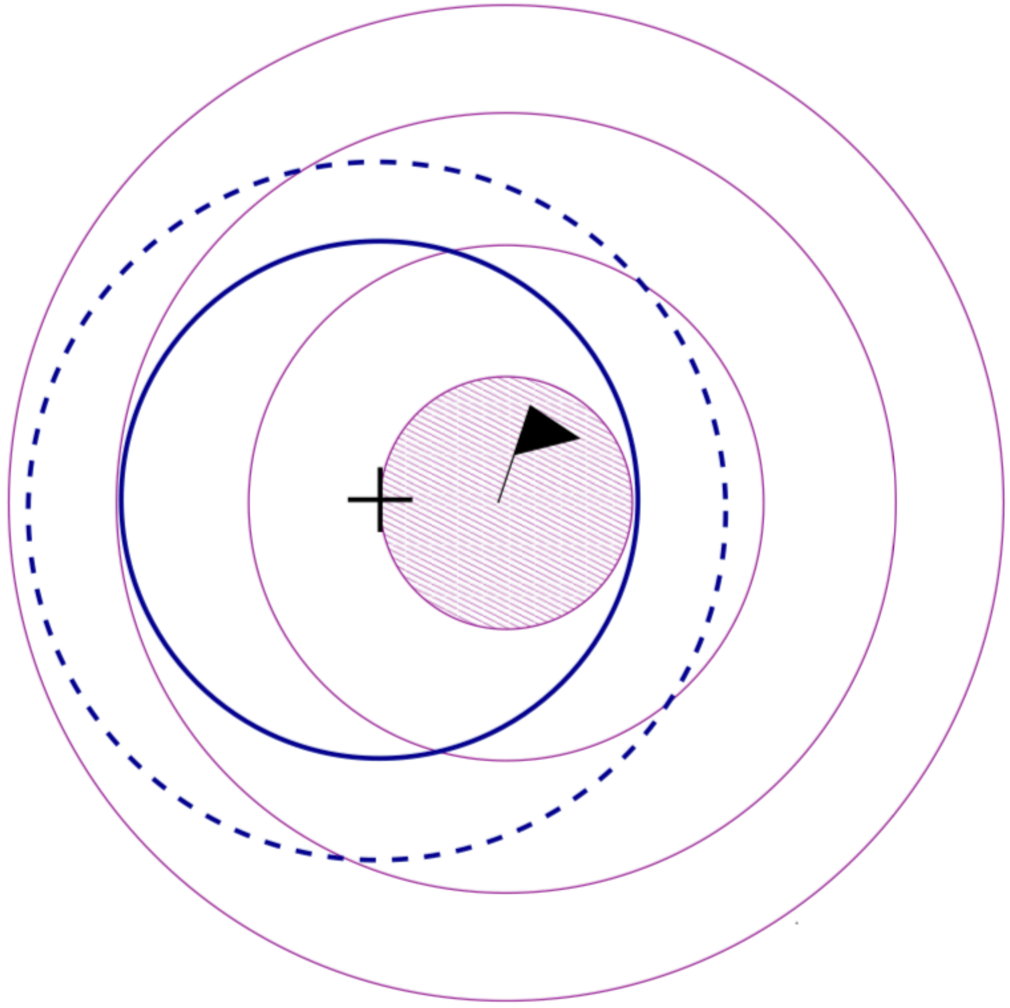
$x = (x_1, x_2)$

$f(x) = -x_1$



$\downarrow$

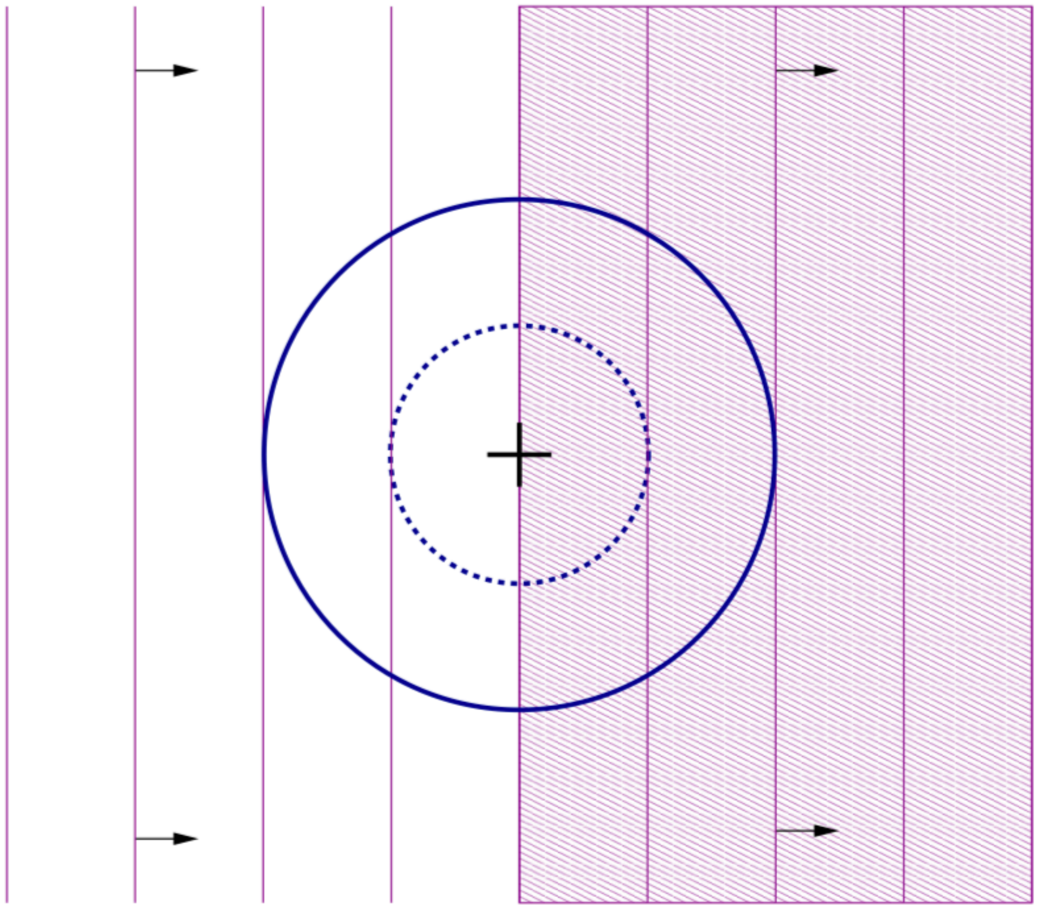
increase σ

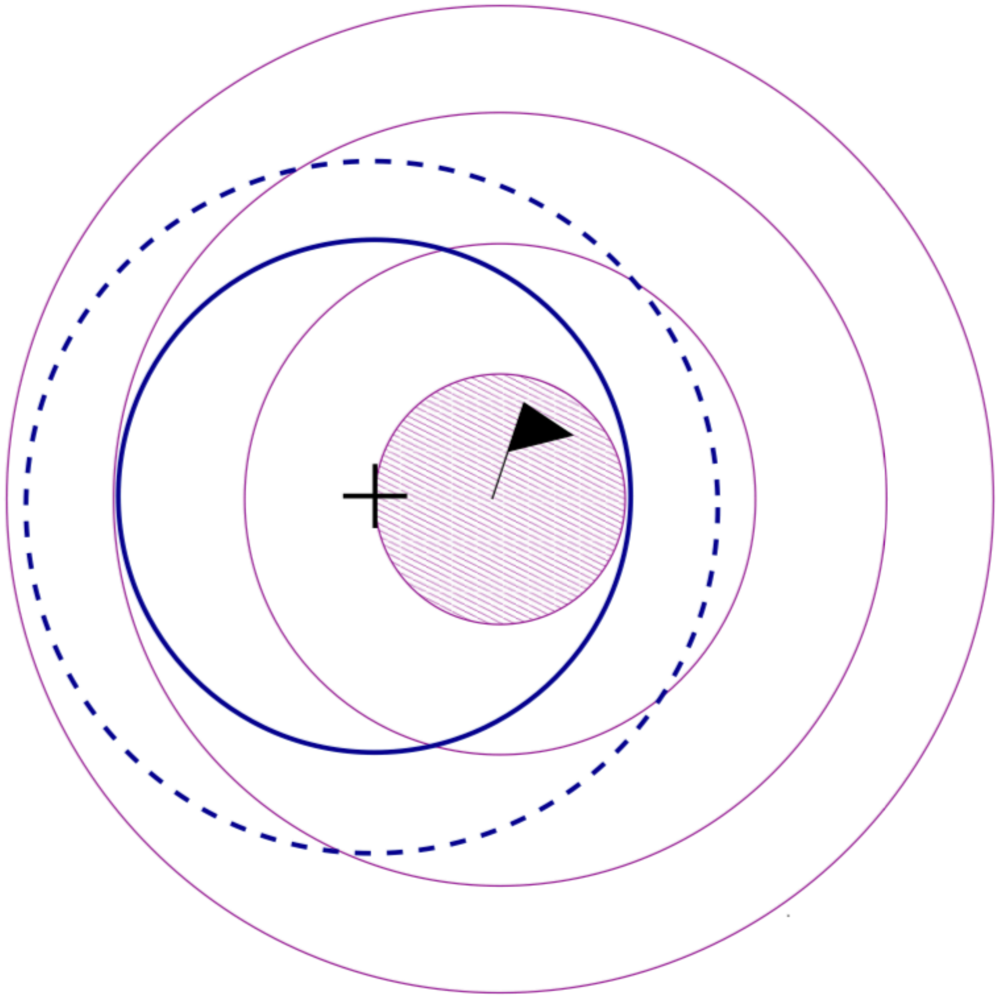


$\downarrow$

decrease σ

Step-size control: 1/5th Success Rule





Probability of success (p_s) Probability of success (p_s)

1/2 1/5 "too small"

Step-size control: 1/5th Success Rule

probability of success per iteration:

$$\mathrm{ps} = \frac{\#\text{candidate solutions better than } m}{\#\text{candidate solutions}}$$

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

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

(1 + 1)-ES

$p_{target} = 1/5$

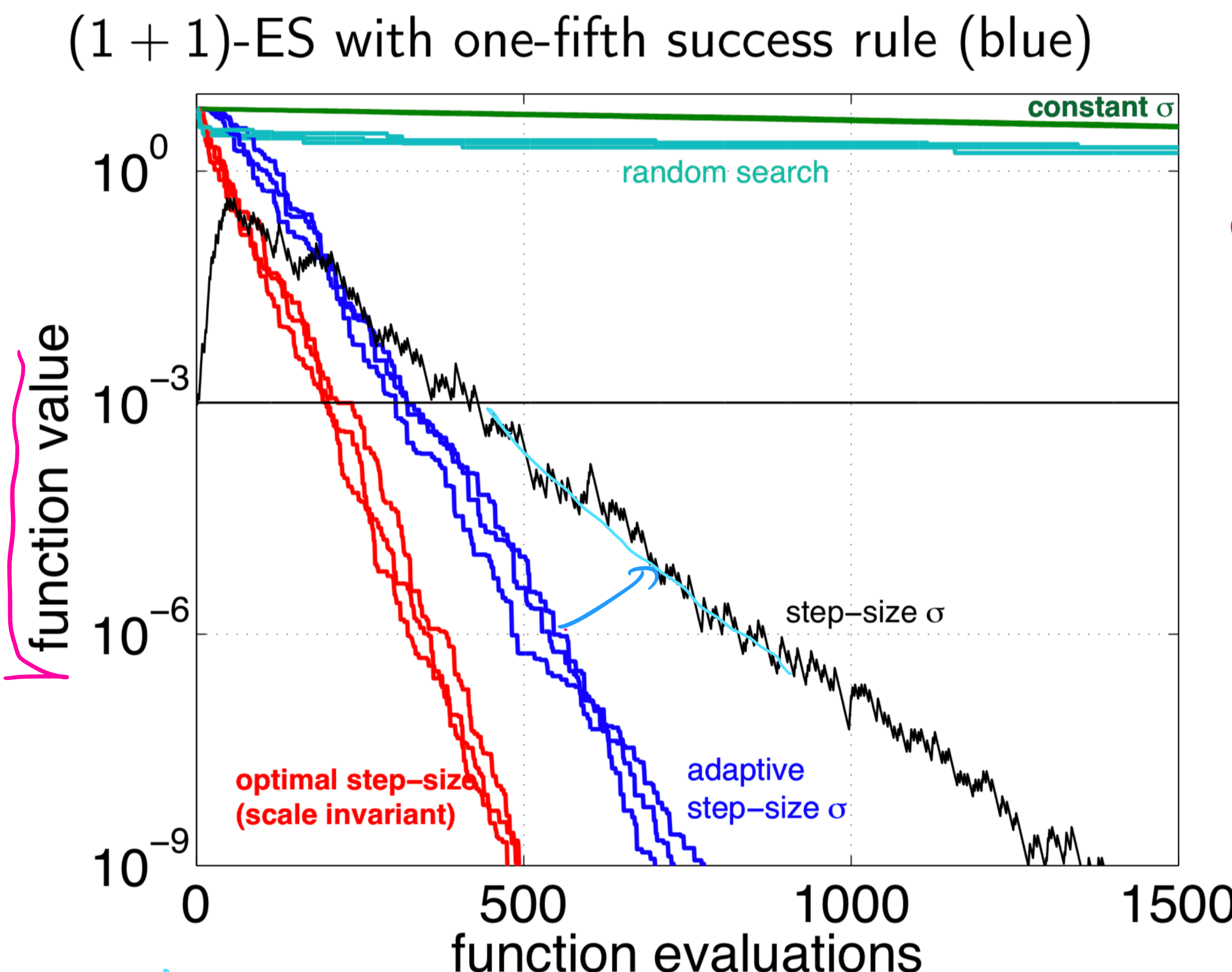
IF *offspring better parent* $[f(\mathbf{x}) \leq f(\mathbf{m})]$

$$p_s = 1,\ \sigma \leftarrow \sigma \times \underbrace{\exp(1/3)}_{1{,}5\ \text{[in exercice]}}$$

ELSE

$$p_s = 0,\ \sigma \leftarrow \sigma / \underbrace{\exp(1/3)^{1/4}}_{1{,}5^{1/4}}$$

(1+1)-ES with One-fifth Success Rule - Convergence



If we display $\sqrt{f(x)} = \|x\|$ instead of $\|x\|^2$, then the blue convergence graph would have the same slope as the step-size slope (black).

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

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

Linear convergence

$(m_t, \sigma_t)_{t \in \mathbb{N}}$.

$\log \|m_t\|^2$

$\log \|m_t\|^2$

The convergence observed on the previous slide is called linear convergence. We formalize it as

$\forall m_0$
$\forall \sigma_0$

$$\frac{1}{t} \ln \|m_t - x^*\| \xrightarrow[t \to +\infty]{} -c$$

convergence rate

$c > 0$

(almost surely).

$-c$: slope of the black graph
slope of blue graph if we display $\sqrt{f}$

Path Length Control - Cumulative Step-size Adaptation (CSA)

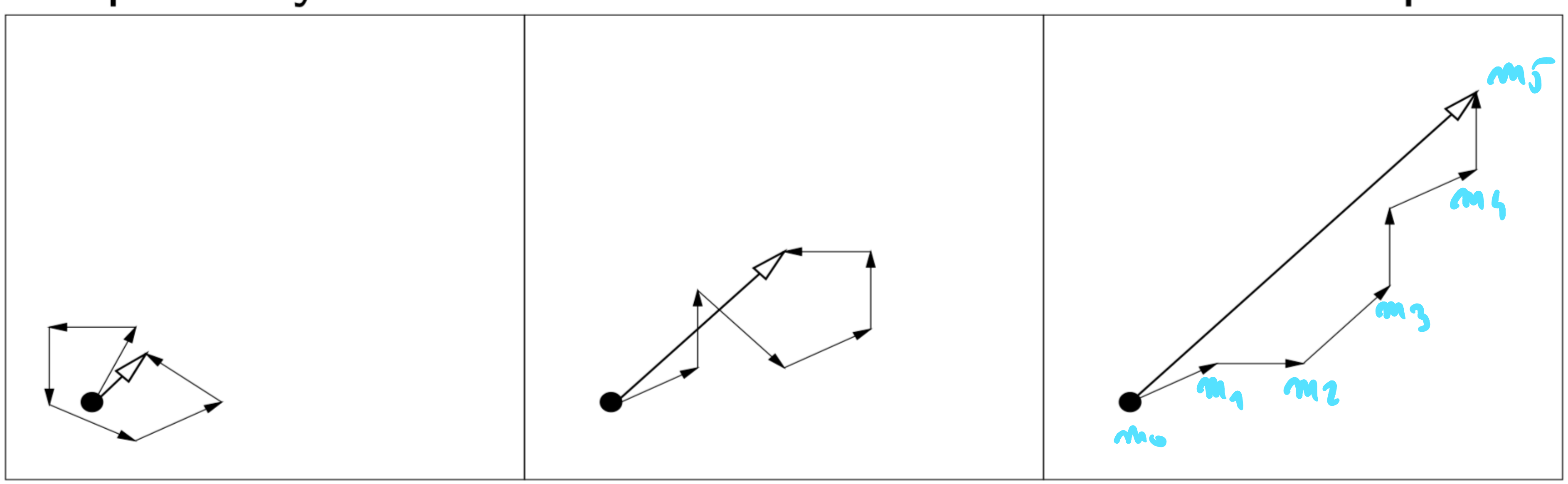
step-size adaptation used in the $(\mu/\mu_w, \lambda)$-ES algorithm framework (in CMA-ES in particular)

Main Idea:

$$\begin{aligned} \boldsymbol{x}_i &= \boldsymbol{m} + \sigma\, \boldsymbol{y}_i \\ \boldsymbol{m} &\leftarrow \boldsymbol{m} + \sigma\, \boldsymbol{y}_w \end{aligned}$$

Measure the length of the *evolution path*

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



$\Downarrow$ decrease σ

$\Downarrow$ increase σ

Sampling of solutions, notations as on slide "The $(\mu/\mu, \lambda)$-ES - Update of the mean vector" with $\mathbf{C}$ equal to the identity.

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

$\sim \mathcal{N}(0, I_d)$

$x_i = m + \sigma y_i$

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

$x_{i:\lambda} = m + \sigma y_{i:\lambda}$

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

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

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

The CSA update with time index notation writes:

$X_i = m_t + \sigma_t y^i_{t+1}$

$\{ y^i_{t+1}, i = \dots, \lambda \} \overset{iid}{\sim} \mathcal{N}(0, I_d)$

$$m_{t+1} = m_t + \sigma_t \sum_{i=1}^{\mu} w_i y_{t+1}^{i:\lambda}$$

$$p^\sigma_{t+1} = (1 - c_\sigma) p^\sigma_t + \sqrt{1 - (1 - c_\sigma)^2} \sqrt{\mu_w} \, y_w^{t+1}$$

$y_w^{t+1} = \sum_{i=1}^{\mu} w_i y^{i:\lambda}_{t+1}$

$$\sigma_{t+1} = \sigma_t \exp\left(\frac{c_\sigma}{d_\sigma} \left(\frac{\| p^\sigma_{t+1} \|}{\mathbb{E}(\| \mathcal{N}(0, I_d) \|)} - 1 \right) \right)$$

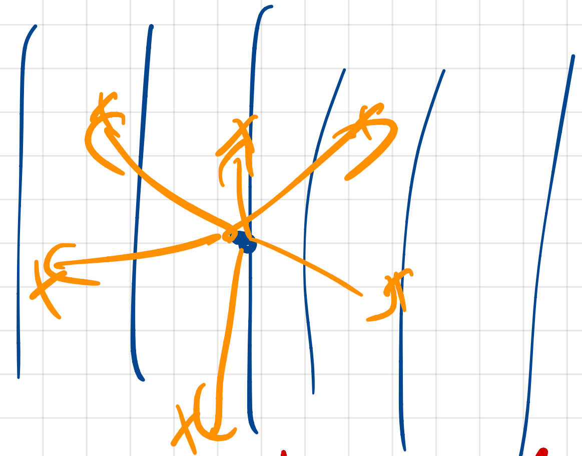
$\mu_w = \dfrac{1}{\sum w_i^2}$

When $f(x)$ = random [each time we evaluate a point on f, it returns a random numer independant from the previous ones and identically distributed]

then $\{ y^{i:\lambda}_{t+1}, i = 1, \dots, \lambda \}$ are iid $\sim \mathcal{N}(0, I_d)$

$y^i_{t+1} \sim \mathcal{N}(0, I_d)$

is not true on other functions. The selection



on f, generally changes the distributi°

If $y_{t+1}^{i:\lambda} \sim \mathcal{N}(0, I_d)$, then $\sqrt{\mu_w}\, y_w^{t+1} = \frac{1}{\sqrt{\sum w_i^2}} \sum_{i=1}^{\mu} w_i\, y_{t+1}^{i:\lambda} \sim \mathcal{N}(0, I_d)$

(HOME PROOF)

under random selection

If $p_\sigma^t \sim \mathcal{N}(0, I_d)$, $\checkmark$ since $\sqrt{\mu_w}\, y_w^{t+1} \sim \mathcal{N}(0, I_d)$

then

$$p_\sigma^{t+1} = (1 - c_\sigma)\, p_\sigma^t + \sqrt{1 - (1 - c_\sigma)^2}\, \sqrt{\mu_w}\, y_w^{t+1} \sim \mathcal{N}(0, I_d)$$

(HOME PROOF)

<u>Then</u> : under random selection :

$$\sigma_{t+1} = \sigma_t \exp\left(\frac{c_\sigma}{d_\sigma} \left(\frac{\| p_{t+1}^{\sigma} \|}{\mathbb{E}(\| \mathcal{N}(0, I_d) \|)} - 1 \right) \right)$$

$$\log \sigma_{t+1} = \log \sigma_t + \frac{c_\sigma}{d_\sigma}\left(\frac{\|p^\sigma_{t+1}\|}{\mathbb{E}\|\mathcal{N}(0,I_d)\|} - 1\right)$$

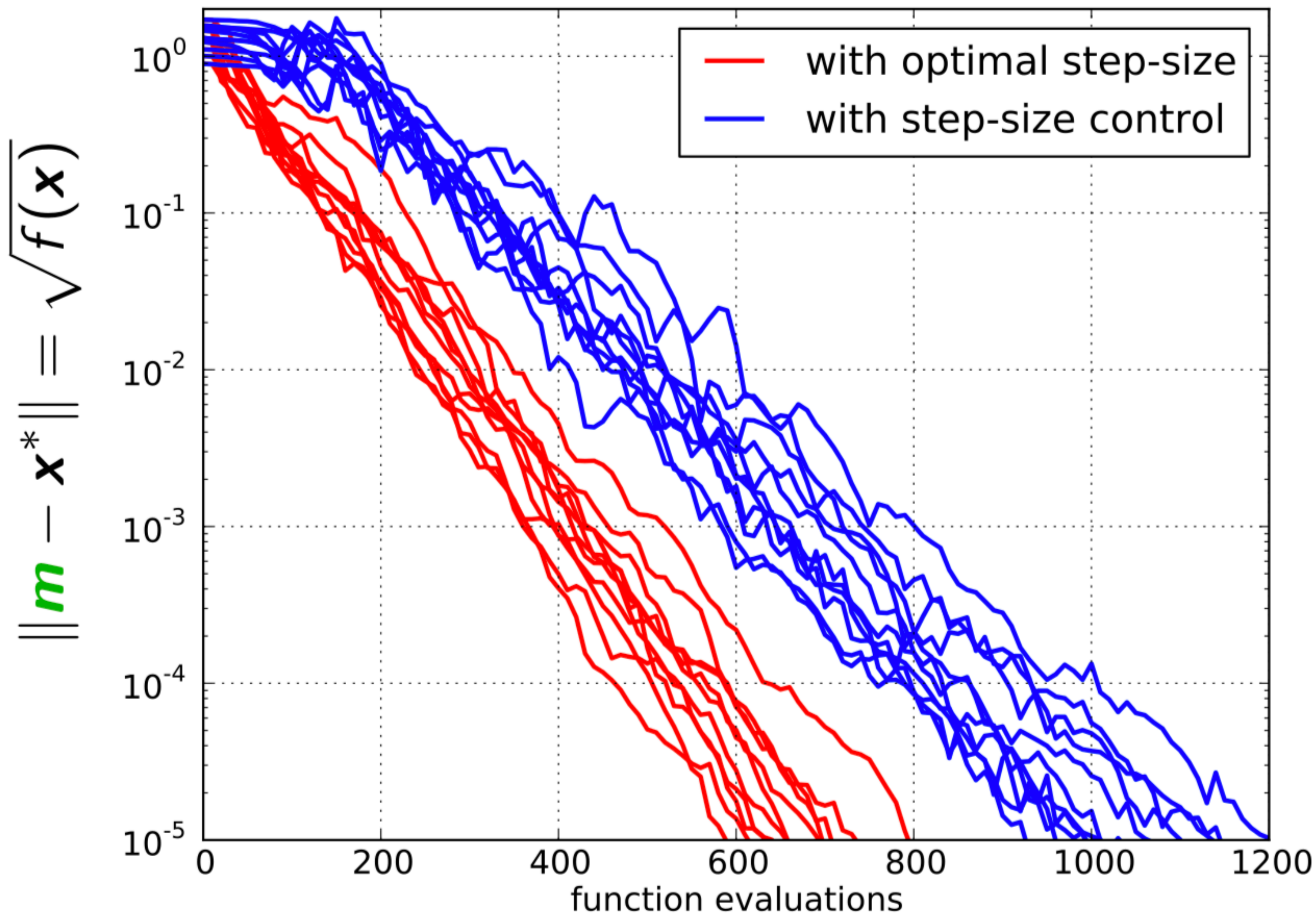
$$\mathbb{E}\left[\log \sigma_{t+1} \mid m_t, \sigma_t\right] = \log \sigma_t + \underbrace{\frac{c_\sigma}{d_\sigma}\left(\frac{\mathbb{E}(\overbrace{\|p^\sigma_{t+1}\|}^{\|\mathcal{N}(0,I_d)\|})}{\mathbb{E}(\|\mathcal{N}(0,I_d)\|)} - 1\right)}_{=0}$$

$$\mathbb{E}\left[\log(\sigma_{t+1}) \mid m_t, \sigma_t\right] = \log \sigma_t$$

$\hookrightarrow$ Formal explanation of log step-size constant under random selection.

Convergence of $(\mu/\mu_w, \lambda)$-CSA-ES

2x11 runs

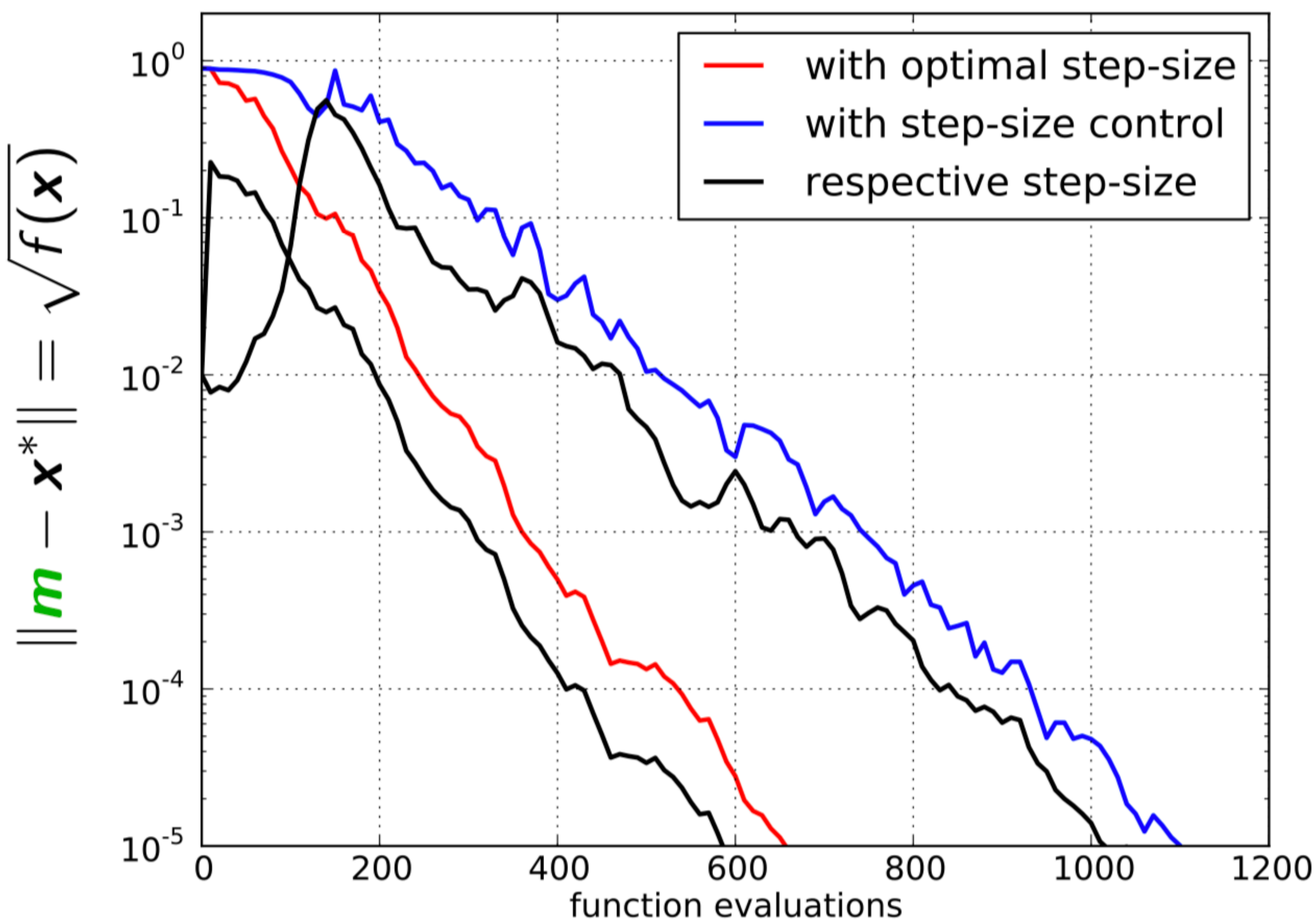


$$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2$$

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

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

Convergence of $(\mu/\mu_w, \lambda)$-CSA-ES



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

for $n = 10$

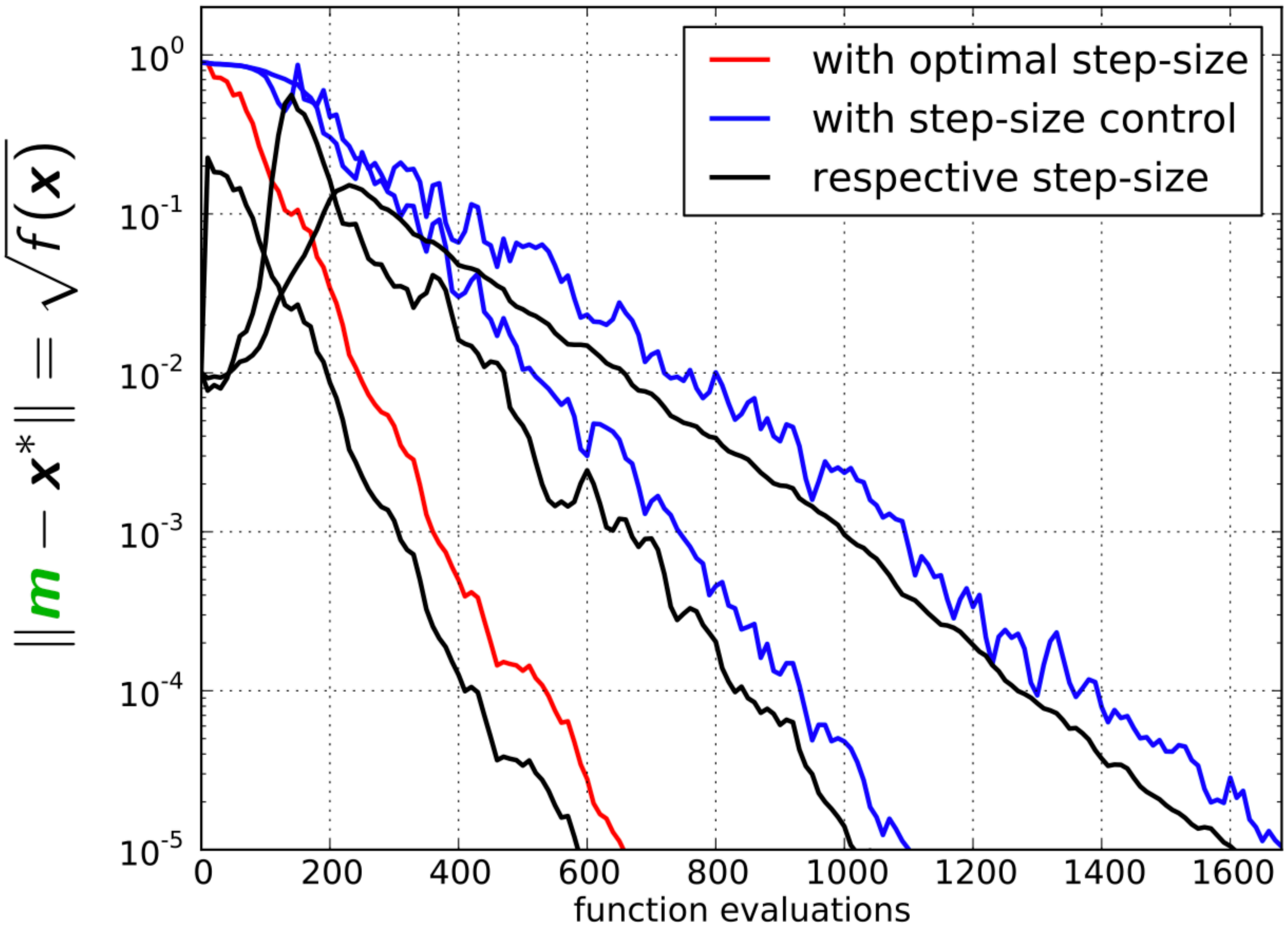
and

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

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

Note: initial step-size taken too small ($\sigma_0 = 10^{-2}$) to illustrate the step-size adaptation

Convergence of $(\mu/\mu_w, \lambda)$-CSA-ES



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

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

comparing optimal versus default damping parameter d_σ:
$\frac{1700}{1100} \approx 1.5$

Optimal Step-size - Lower-bound for Convergence Rates

In the previous slides we have displayed some runs with "optimal" step-size.

Optimal step-size relates to step-size proportional to the distance to the optimum: $\sigma_t = \sigma\|x - x^\star\|$ where $x^\star$ is the optimum of the optimized function (with σ properly chosen).

The associated algorithm is not a real algorithm (as it needs to know the distance to the optimum) but it gives bounds on convergence rates and allows to compute many important quantities.

The goal for a step-size adaptive algorithm is to achieve convergence rates close to the one with optimal step-size

We will formalize this in the context of the (1+1)-ES. Similar results can be obtained for other algorithm frameworks.

Optimal Step-size - Bound on Convergence Rate - (1+1)-ES

Consider a (1+1)-ES algorithm with any step-size adaptation mechanism:

$$X_{t+1} = \begin{cases} X_t + \sigma_t \mathcal{N}_{t+1} & \text{if } f(X_t + \sigma_t \mathcal{N}_{t+1}) \leq f(X_t) \\ X_t & \text{otherwise} \end{cases}$$

with $\{\mathcal{N}_t, t \geq 1\}$ i.i.d. $\sim \mathcal{N}(0, I_d)$

equivalent writing:

$$X_{t+1} = X_t + \sigma_t \mathcal{N}_{t+1} 1_{\{f(X_t + \sigma_t \mathcal{N}_{t+1}) \leq f(X_t)\}}$$

Bound on Convergence Rate - (1+1)-ES

Theorem: For any objective function $f : \mathbb{R}^n \to \mathbb{R}$, for any $y^\star \in \mathbb{R}^n$

$$E[\|X_{t+1} - y^\star\|] \geq E[\|X_t - y^\star\|] - \tau$$

lower bound

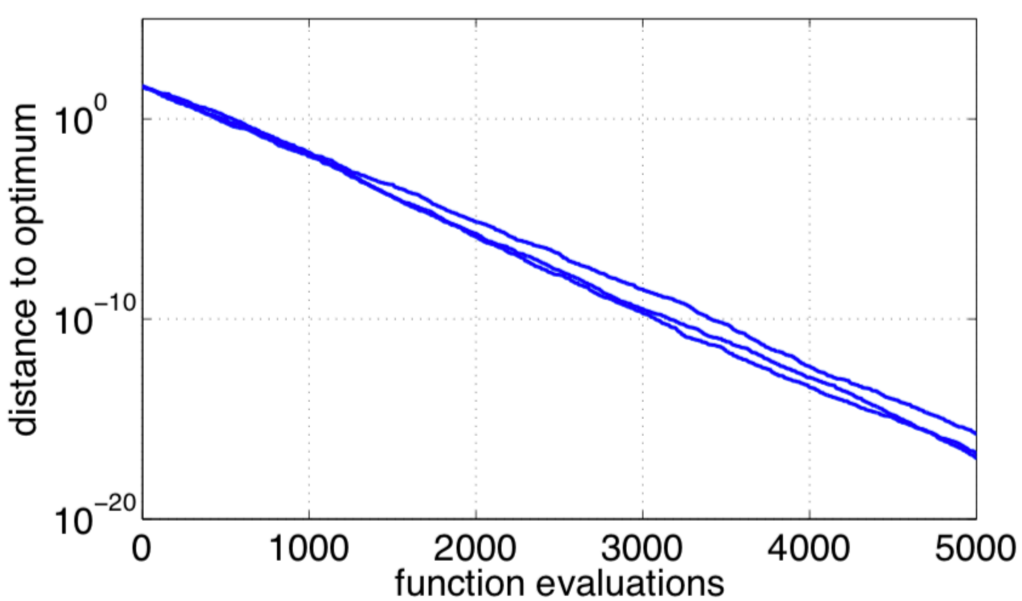
where $\tau = \max_{\sigma \in \mathbb{R}>} \underbrace{E[\ln^- \|e_1 + \sigma \mathcal{N}\|]}_{=:\varphi(\sigma)}$ with $e_1 = (1,0,\ldots,0)$

Theorem: The convergence rate lower-bound is reached on spherical functions $f(x) = g(\|x - x^\star\|)$ (with $g : \mathbb{R}_{\geq 0} \to \mathbb{R}$ strictly increasing) and step-size proportional to the distance to the optimum $\sigma_t = \sigma_{\text{opt}}\|x - x^\star\|$ with σ_{opt} such that $\varphi(\sigma_{\text{opt}}) = \tau$.

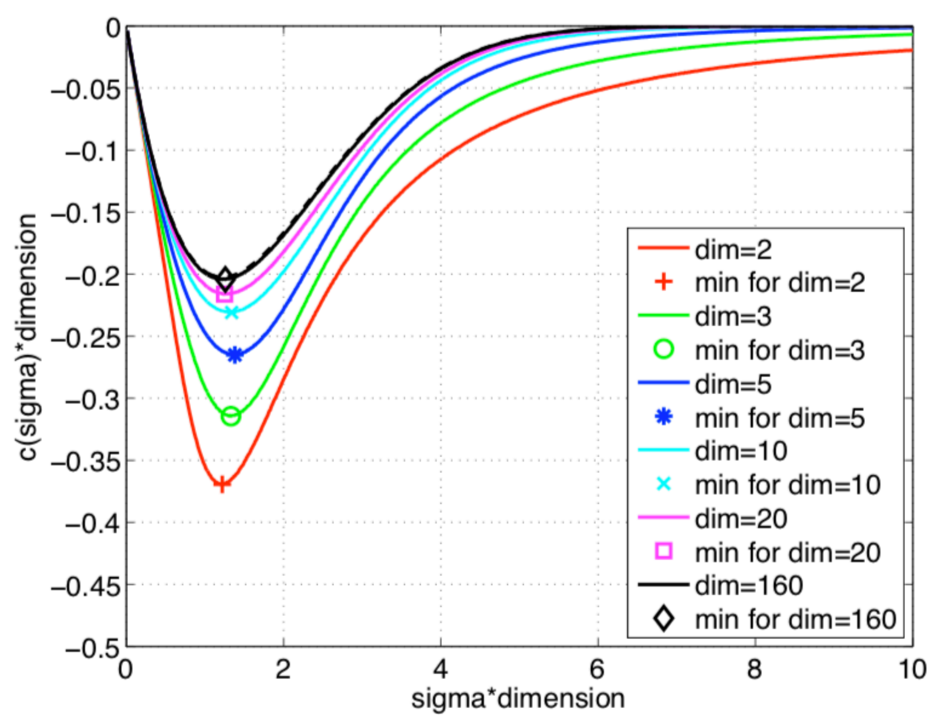
Log-Linear Convergence of scale-invariance step-size ES

Theorem: The (1+1)-ES with step-size proportional to the distance to the optimum $\sigma_t = \sigma \|x\|$ converges (log)-linearly on the sphere function $f(x) = g(\|x\|)$ almost surely:

$$\frac{1}{t} \ln \frac{\|X_t\|}{\|X_0\|} \xrightarrow[t\to\infty]{} -\varphi(\sigma) =: \mathrm{CR}_{(1+1)}(\sigma)$$



$n = 20$ and $\sigma = 0.6/n$



Asymptotic Results ($n \to \infty$)

Theorem

Let $\sigma > 0$, the convergence rate of the (1+1)-ES with scale-invariant step-size on spherical functions satisfies at the limit

$$\lim_{n\to\infty} n \times \mathrm{CR}_{(1+1)}\left(\frac{\sigma}{n}\right) = \frac{-\sigma}{\sqrt{2\pi}} \exp\left(-\frac{\sigma^2}{8}\right) + \frac{\sigma^2}{2}\Phi\left(-\frac{\sigma}{2}\right)$$

where Φ is the cumulative distribution of a normal distribution.

optimal convergence rate decreases to zero like $\frac{1}{n}$