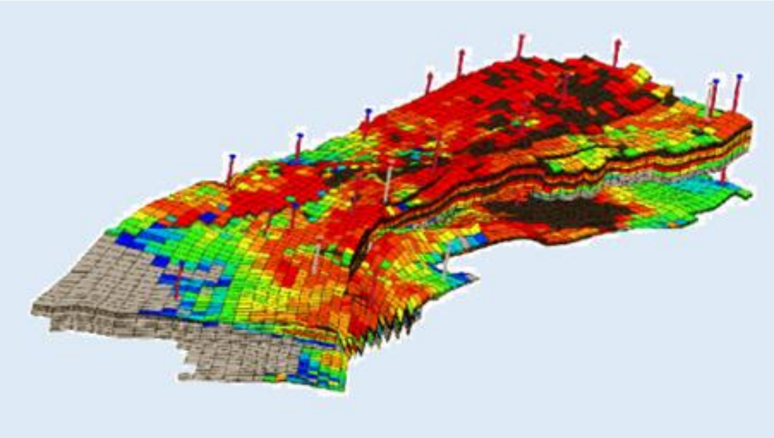
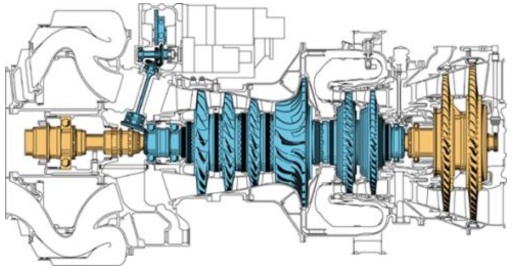
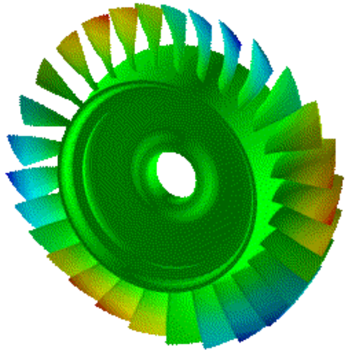
DERIVATIVE FREE OPTIMIZATION AND APPLICATIONS

Delphine Sinoquet (IFPEN)

COURSE 1: MAIN DFO METHODS







https://www.ifpenergiesnouvelles.fr/page/delphine-sinoquet

DERIVATIVE FREE OPTIMIZATION AND APPLICATIONS

- **Course 1: main DFO methods**
- Course 2: various applications of DFO
- Course 3: some challenges in DFO

REFERENCES

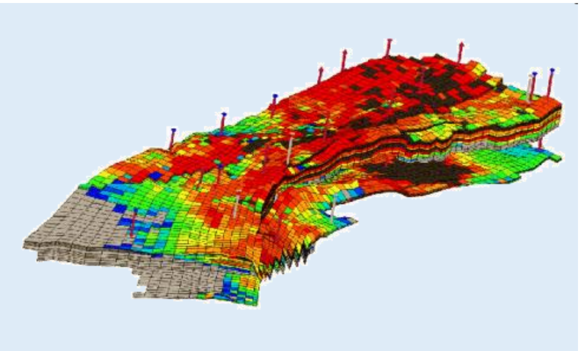
- Audet, C. and Hare W., Derivative-Free and Blackbox Optimization, Springer Series in Operations Research and Financial Engineering (2017)

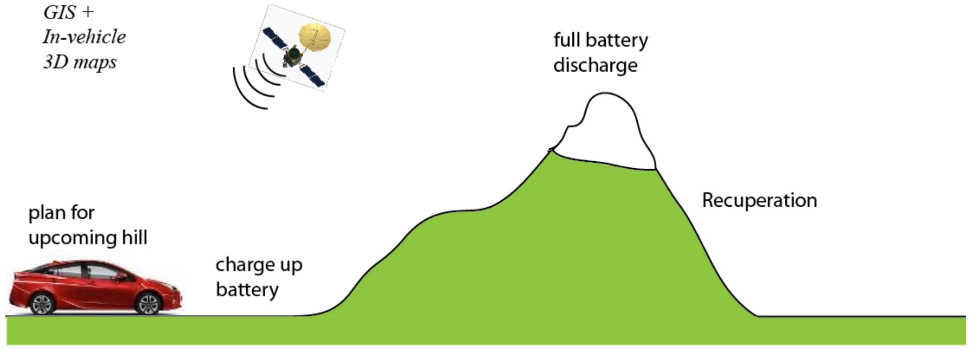
- Conn, A.R., Scheinberg, K., Vicente, L.N., Introduction to derivative-free optimization. SIAM, Philadelphia (2009)

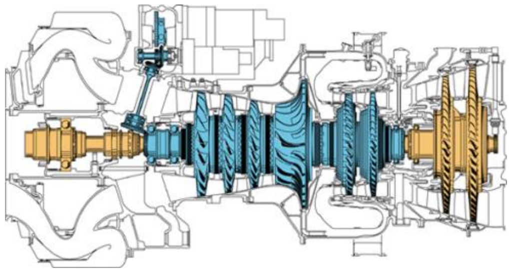
- Rios, L.M. & Sahinidis, N.V., Derivative-free optimization: a review of algorithms and comparison of software implementations, J Glob Optim (2013), Vol. 56, Issue 3, pp 1247–1293

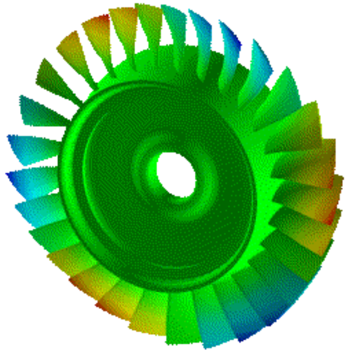
- Cartis, C., Lectures on global and derivative-free optimization, University of Oxford (2018)

WHY DERIVATIVE FREE OPTIMIZATION ?

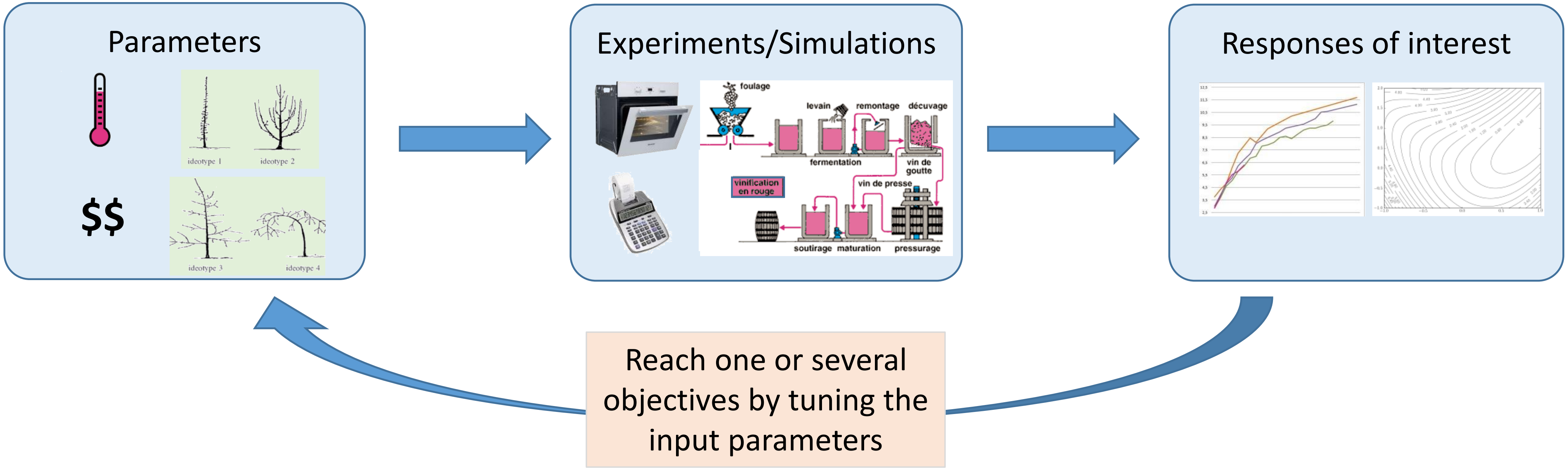








STUDY OF COMPLEX SYSTEMS



Parameters

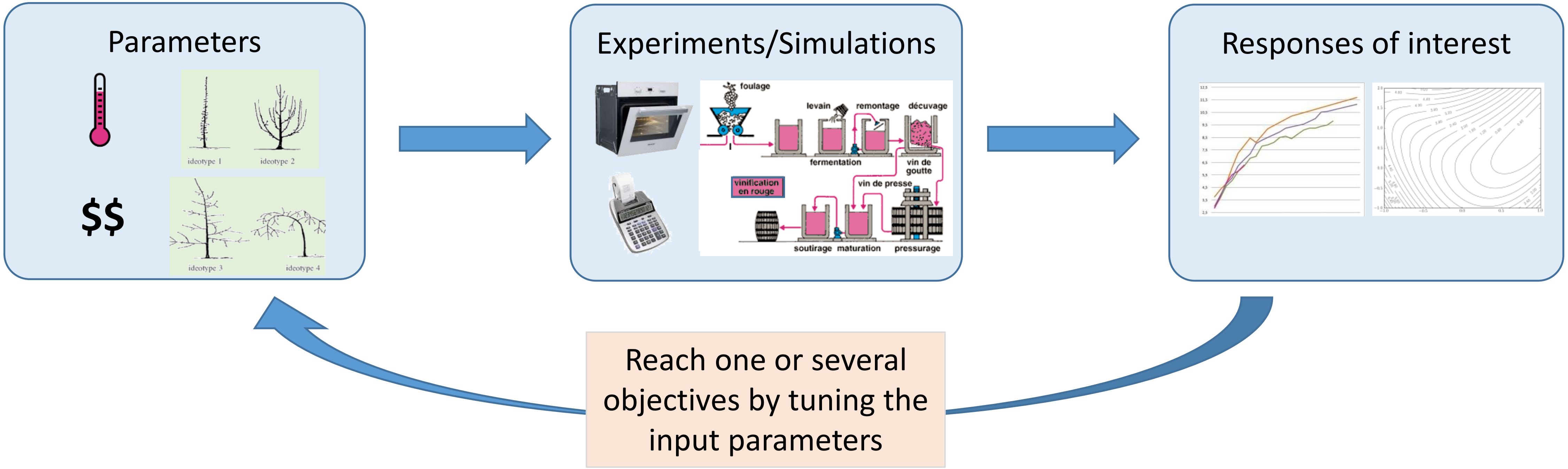
$$

Experiments/Simulations

Responses of interest

Reach one or several objectives by tuning the input parameters

STUDY OF COMPLEX SYSTEMS



Parameters

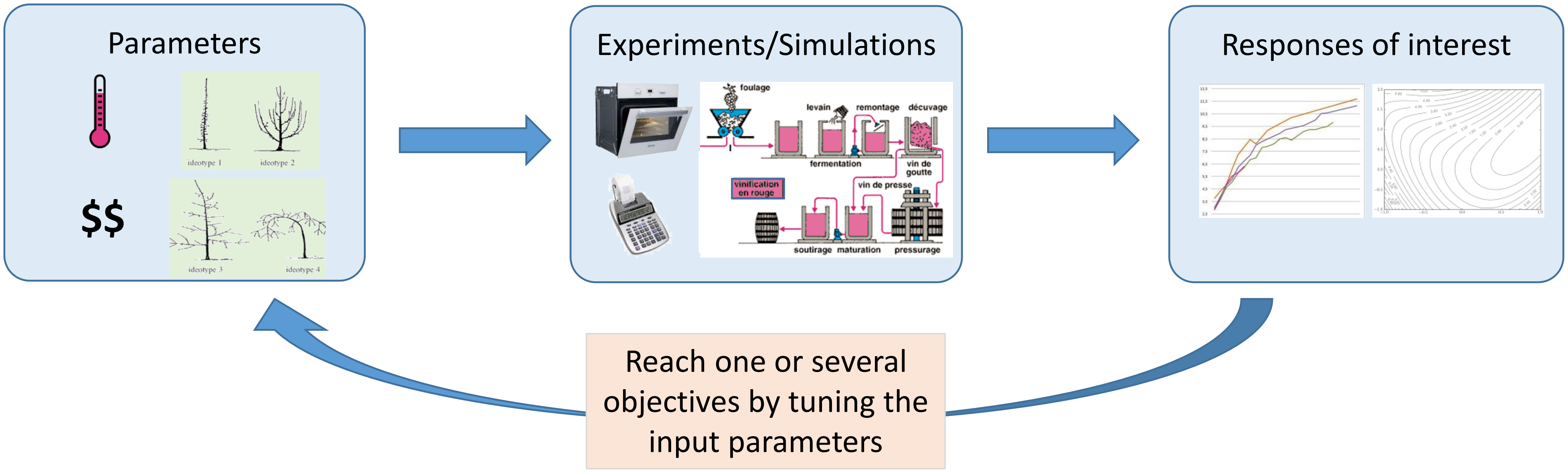
Experiments/Simulations

Responses of interest

Reach one or several objectives by tuning the input parameters

- ➢ Manual optimization (trial & error): when the expert knows very well and can control the system, and when the number of parameters is small

STUDY OF COMPLEX SYSTEMS



Parameters

$$

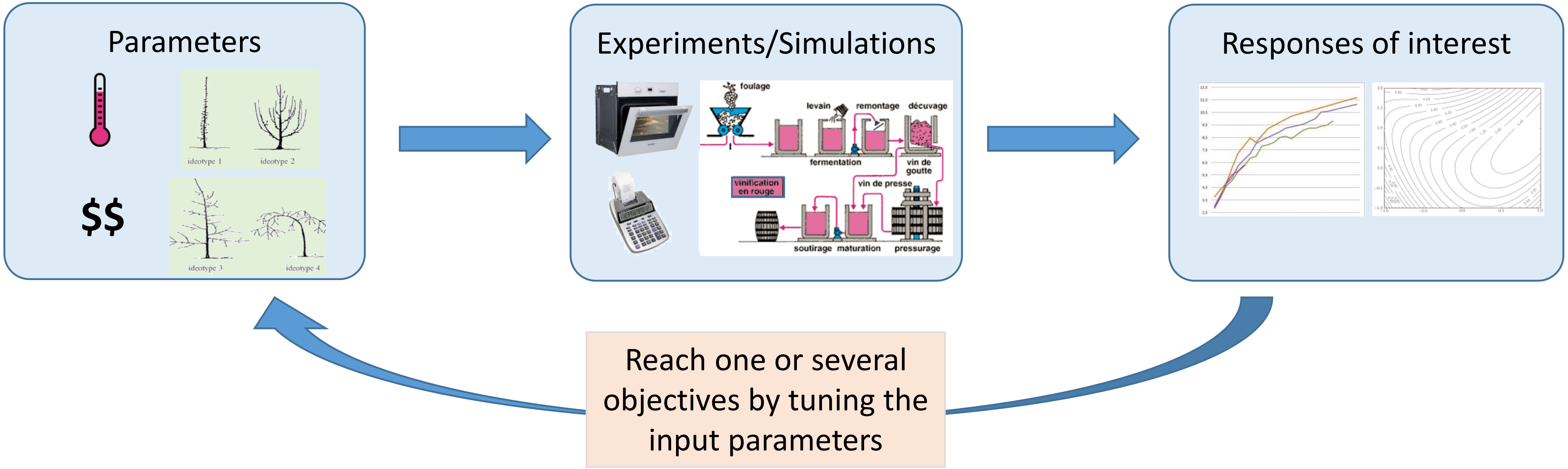
Experiments/Simulations

Responses of interest

Reach one or several objectives by tuning the input parameters

- ➢ Manual optimization (trial & error): when the expert knows very well and can control the system, and when the number of parameters is small
- ➢ Random exploration: How many simulations should we do ? How do we know that the current set of values is close to a solution ?

STUDY OF COMPLEX SYSTEMS



Parameters

$$

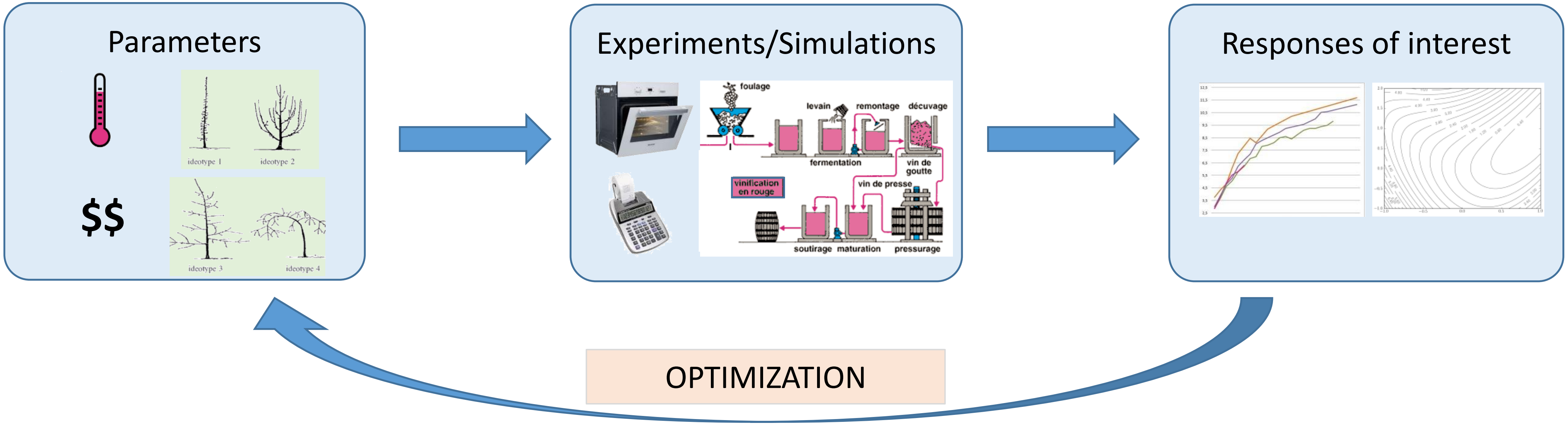
Experiments/Simulations

Responses of interest

Reach one or several objectives by tuning the input parameters

- Manual optimization (trial & error): when the expert knows very well and can control the system, and when the number of parameters is small
- Random exploration: How many simulations should we do ? How do we know that the current set of values is close to a solution ?
- Discretisation of the parameter space on a regular grid: 3^n simulations if we consider 3 points per dimension and n parameters !!!

STUDY OF COMPLEX SYSTEMS



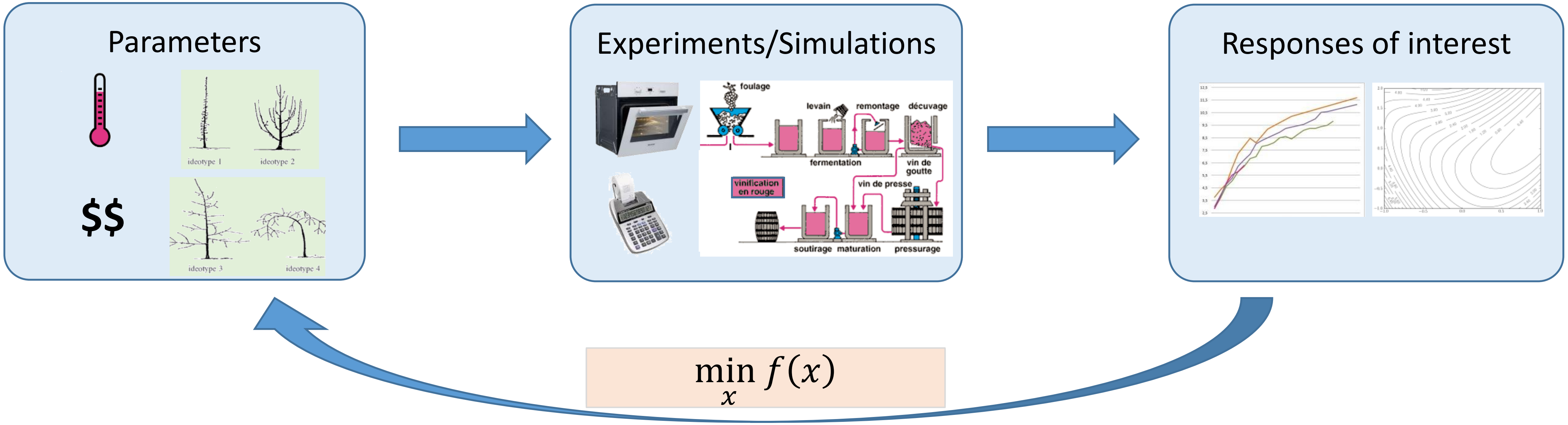
Parameters

Experiments/Simulations

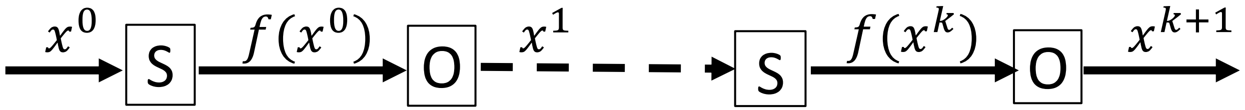
Responses of interest

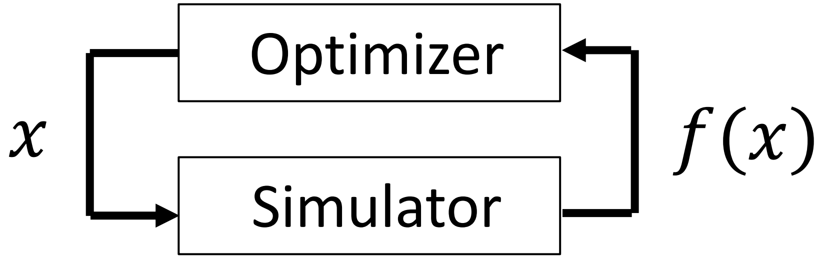
OPTIMIZATION

STUDY OF COMPLEX SYSTEMS

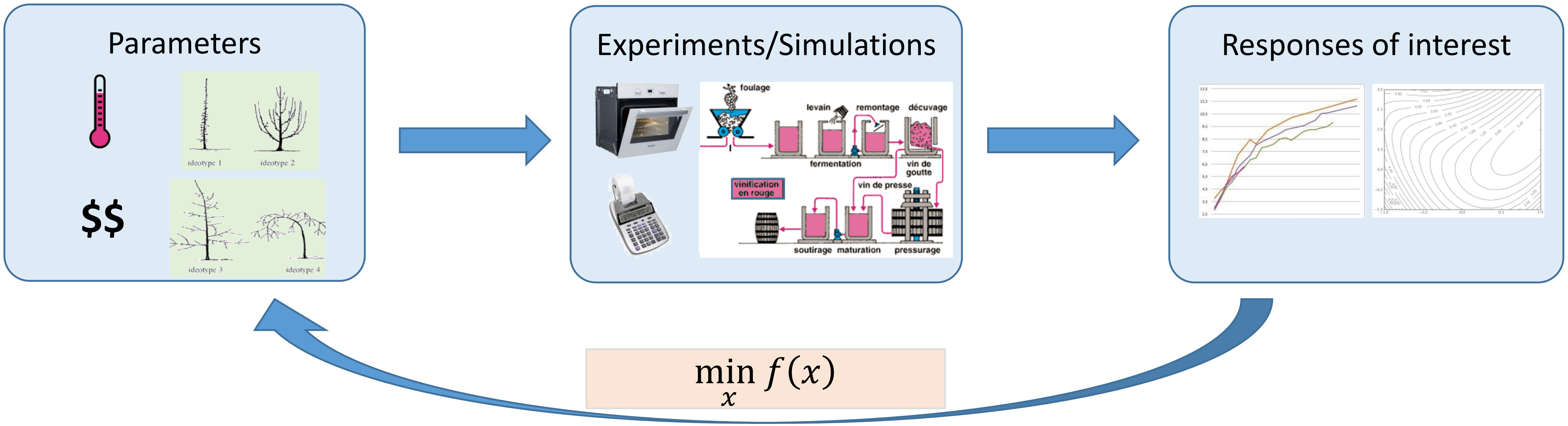


- An optimizer *O* is an algorithm which proposes iteratively a new x based on the information from previous trials in order to approximate the solution of the problem $x = \operatorname{argmin}(f(x))$

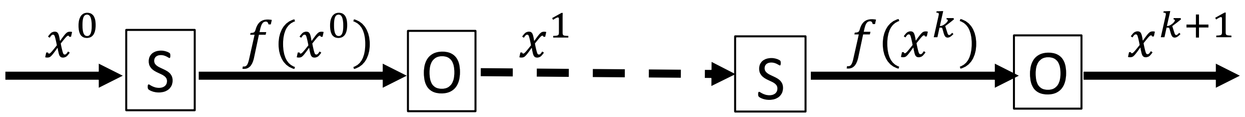


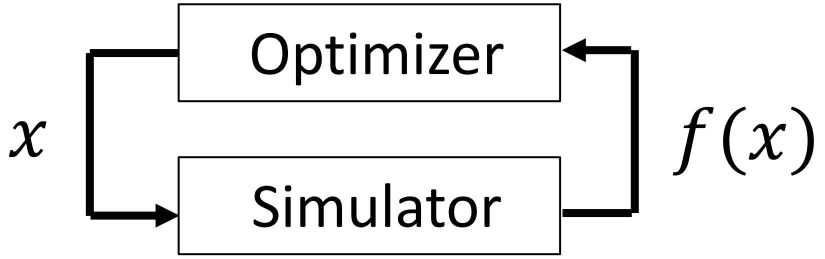


STUDY OF COMPLEX SYSTEMS



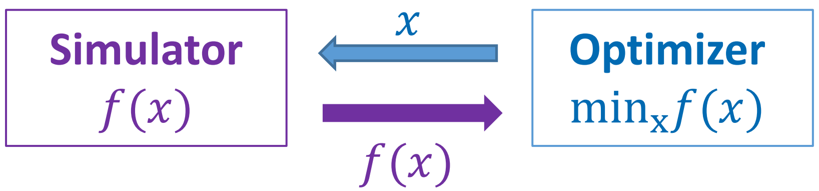
- An optimizer O is an algorithm which proposes iteratively a new x based on the information from previous trials in order to approximate the solution of the problem $x = \text{argmin}(f(x))$
- The cost of the optimization is linked with the number of calls to the simulator S which evaluates f





WHY DERIVATIVE FREE OPTIMIZATION ?

$$\min_{x \in \mathbb{R}^n} f(x)$$



Simulator
$f(x)$

x

$f(x)$

Optimizer
$\min_x f(x)$

- More and more complex simulators
  - black-box simulator (proprietary code or a simulation package) derivatives of objective function are not available
  - numerical approximation of $\nabla f(x)$ is expensive: finite-differences when computing $f(x)$ is expensive or for a high number of optimization variables x
  - computing $f(x)$ is expensive: time consuming numerical simulations or experiments

➢ Need for optimization methods adapted to derivative free problems

DFO METHODS

- *(Standard derivative-based methods with approximate gradients)*
- Direct Search methods
  - Nelder Mead Simplex
  - Pattern Search
- Surrogate optimization / model-based DFO methods
  - Local model of the objective function
  - Global model of the objective function
- Stochastic DFO methods
  - Evolutionary strategies
  - Simulated annealing

APPROXIMATE GRADIENTS

- by finite differences (F.D.)
  n+1 simulations per iteration

$$\nabla_{x_i} f(x^k) \approx \frac{f(x^k + he_i) - f(x^k)}{h}$$

- by generalized finite derivatives (E.F.D.)

$$\nabla f(x^k) \approx (\Delta x)^{-1}\left(f(x^k + \Delta x) - f(x^k)\right)$$

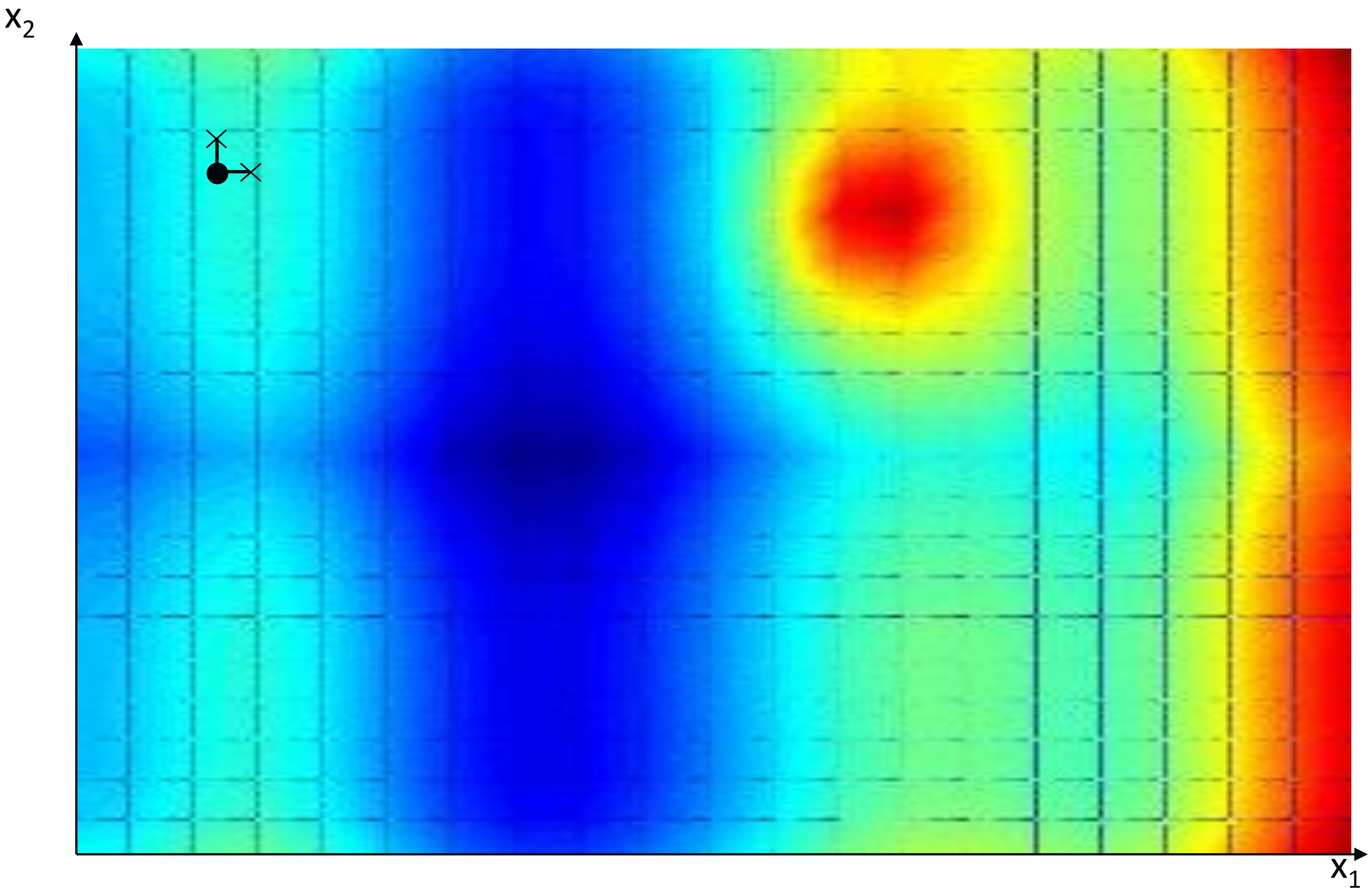
Δx is the perturbation matrix (for F.D. $\Delta x = I_n$)

- by an approximation model of the objective function
  simulations from previous iterations + m (≤ n) new simulations

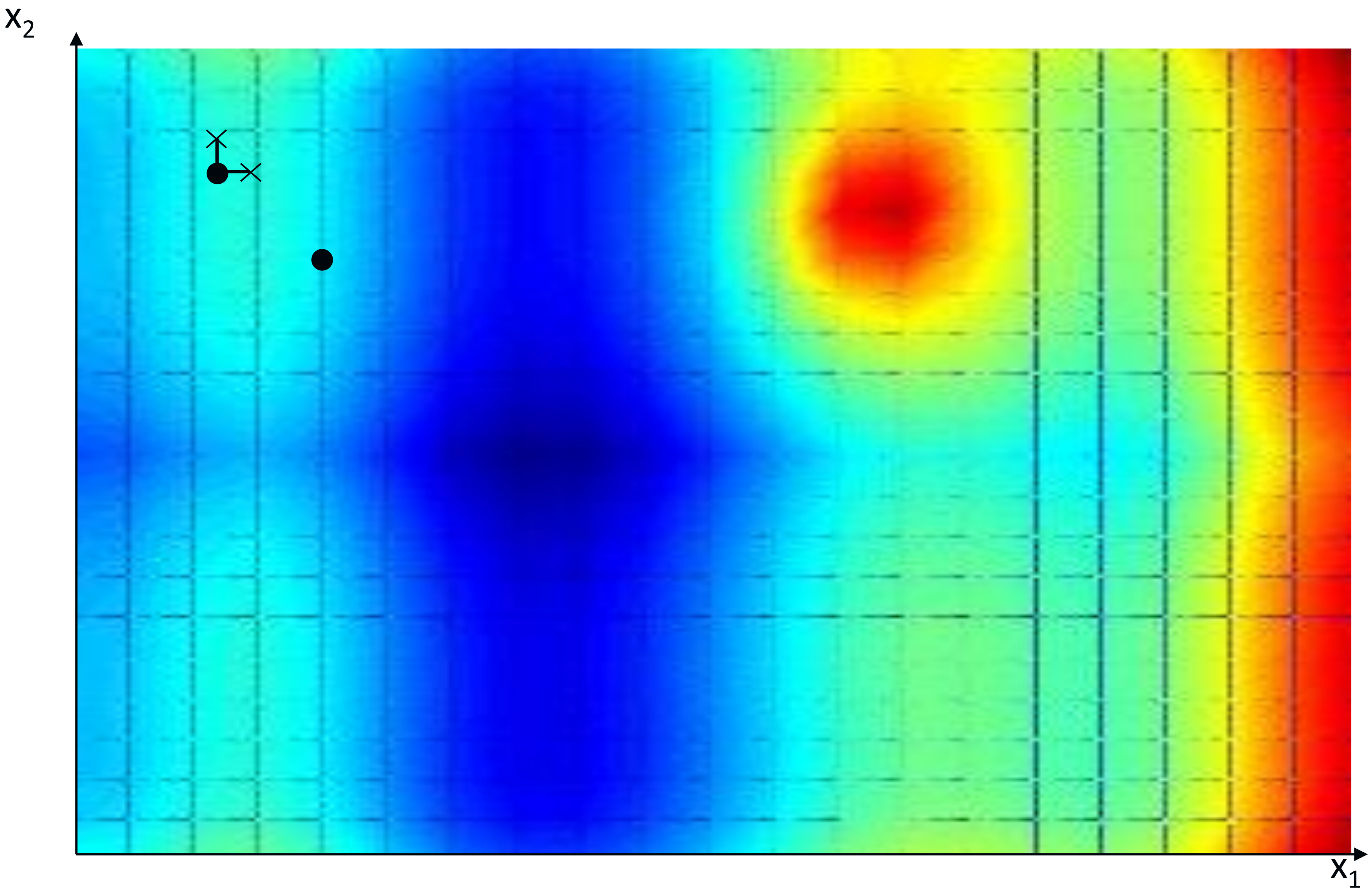
$$\tilde{F}(x^k + s) = f(x^k) + \tilde{g}^k s \quad \text{ou} \quad \tilde{F}(x^k + s) = f(x^k) + \tilde{g}^k s + s^T \widetilde{H}^k s$$

$$\nabla f(x^k) \approx \tilde{g}^k$$

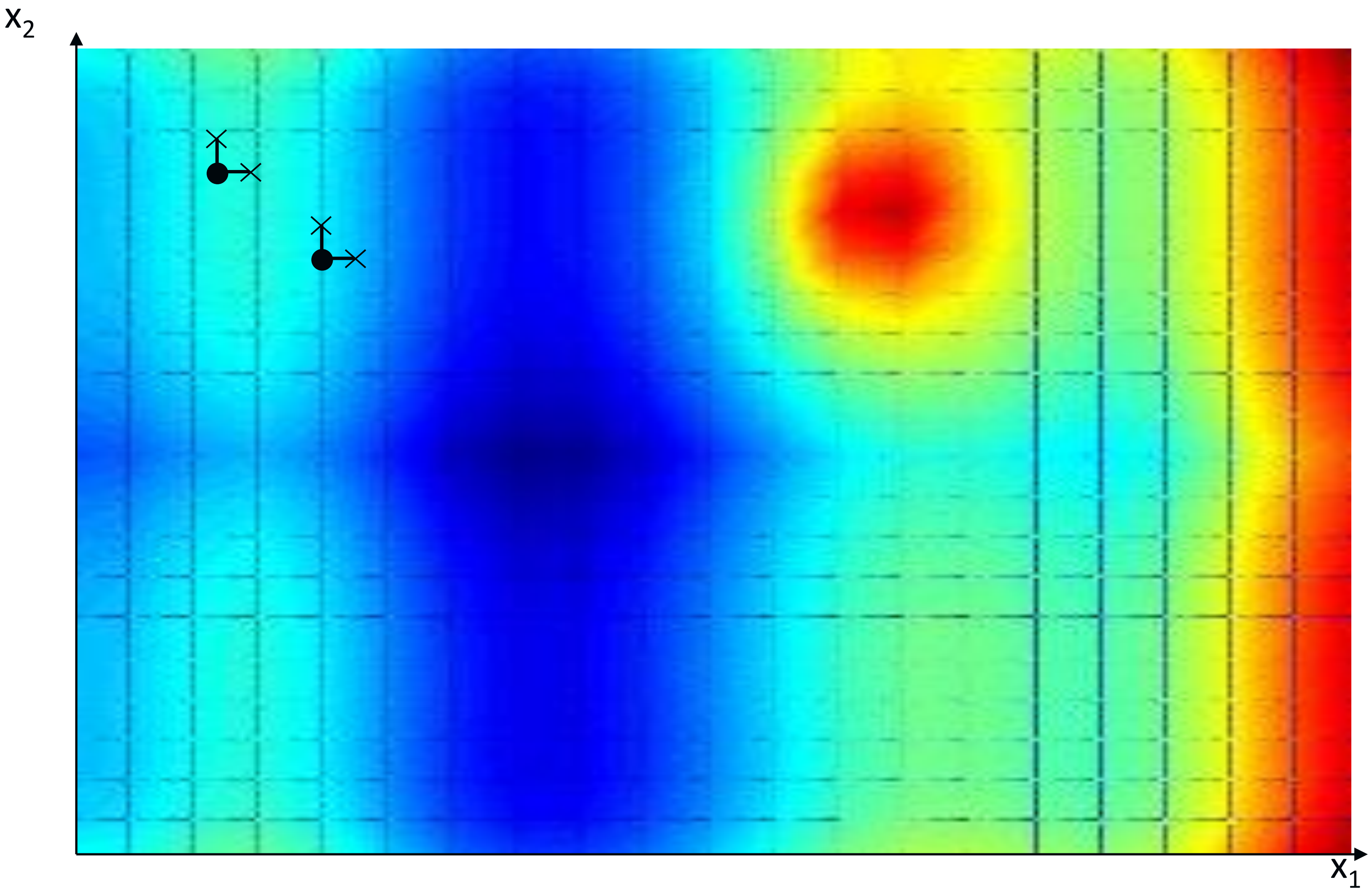
APPROXIMATE GRADIENTS: FINITE DIFFERENCES



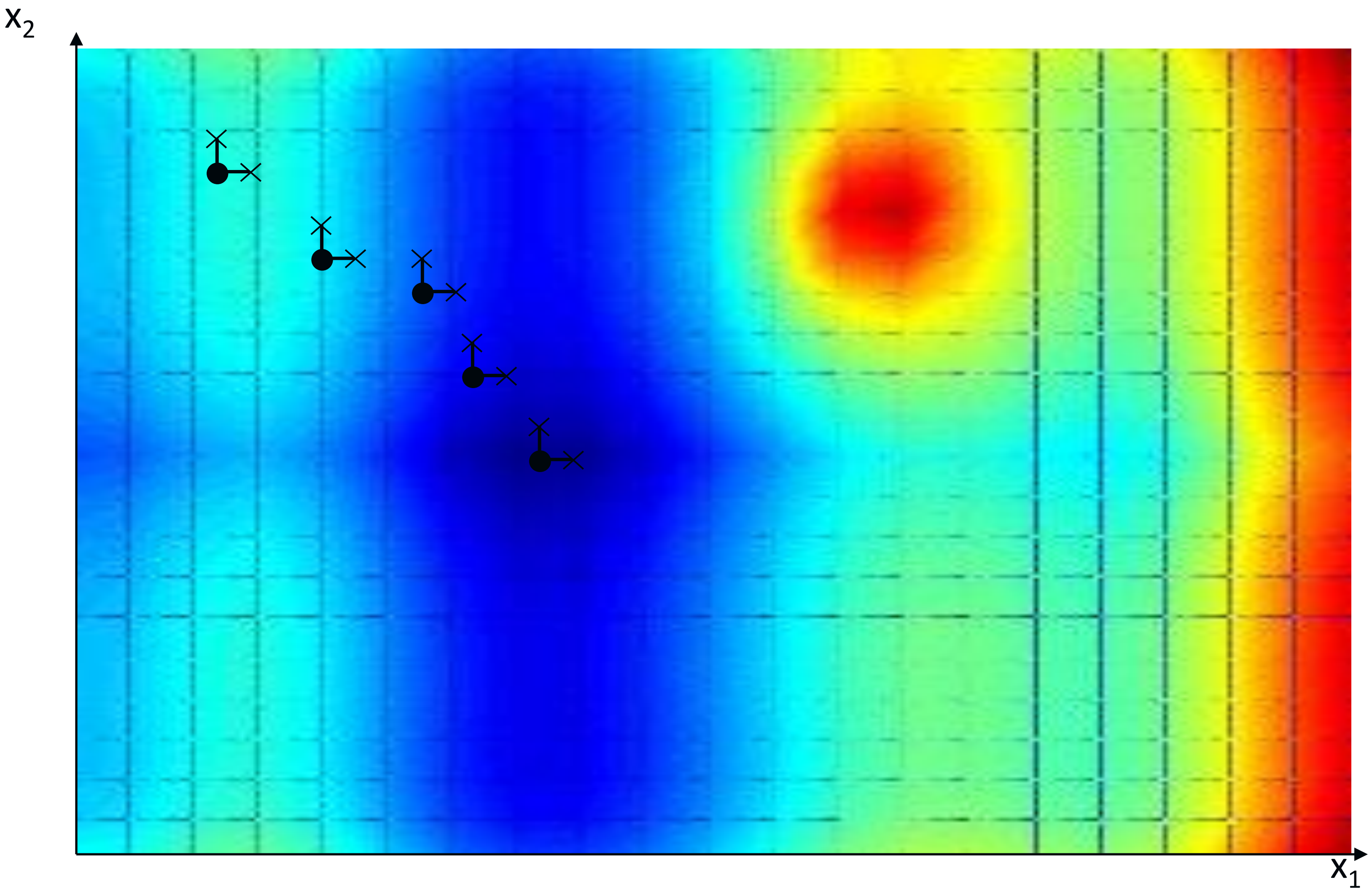
APPROXIMATE GRADIENTS: FINITE DIFFERENCES



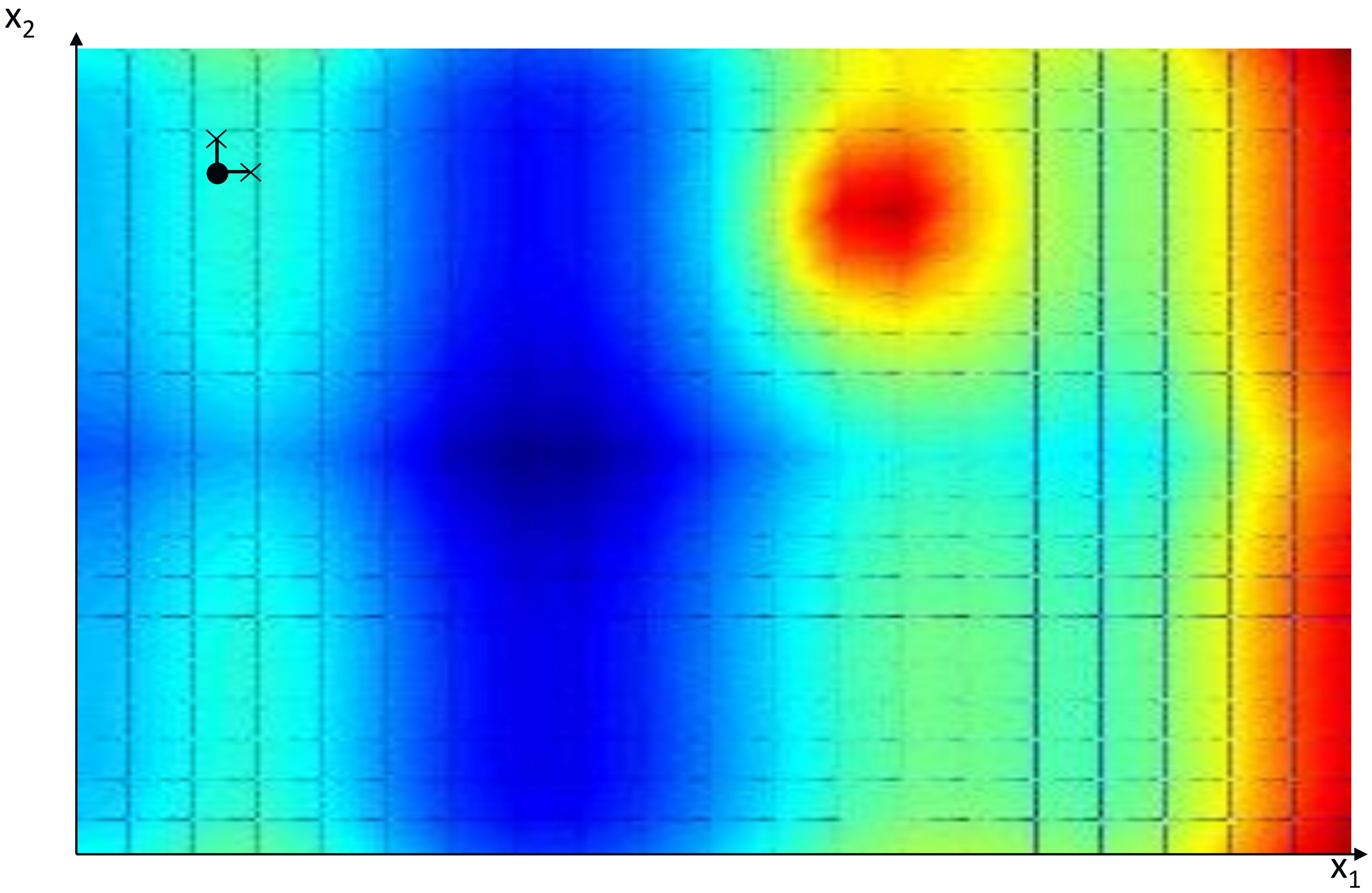
APPROXIMATE GRADIENTS: FINITE DIFFERENCES



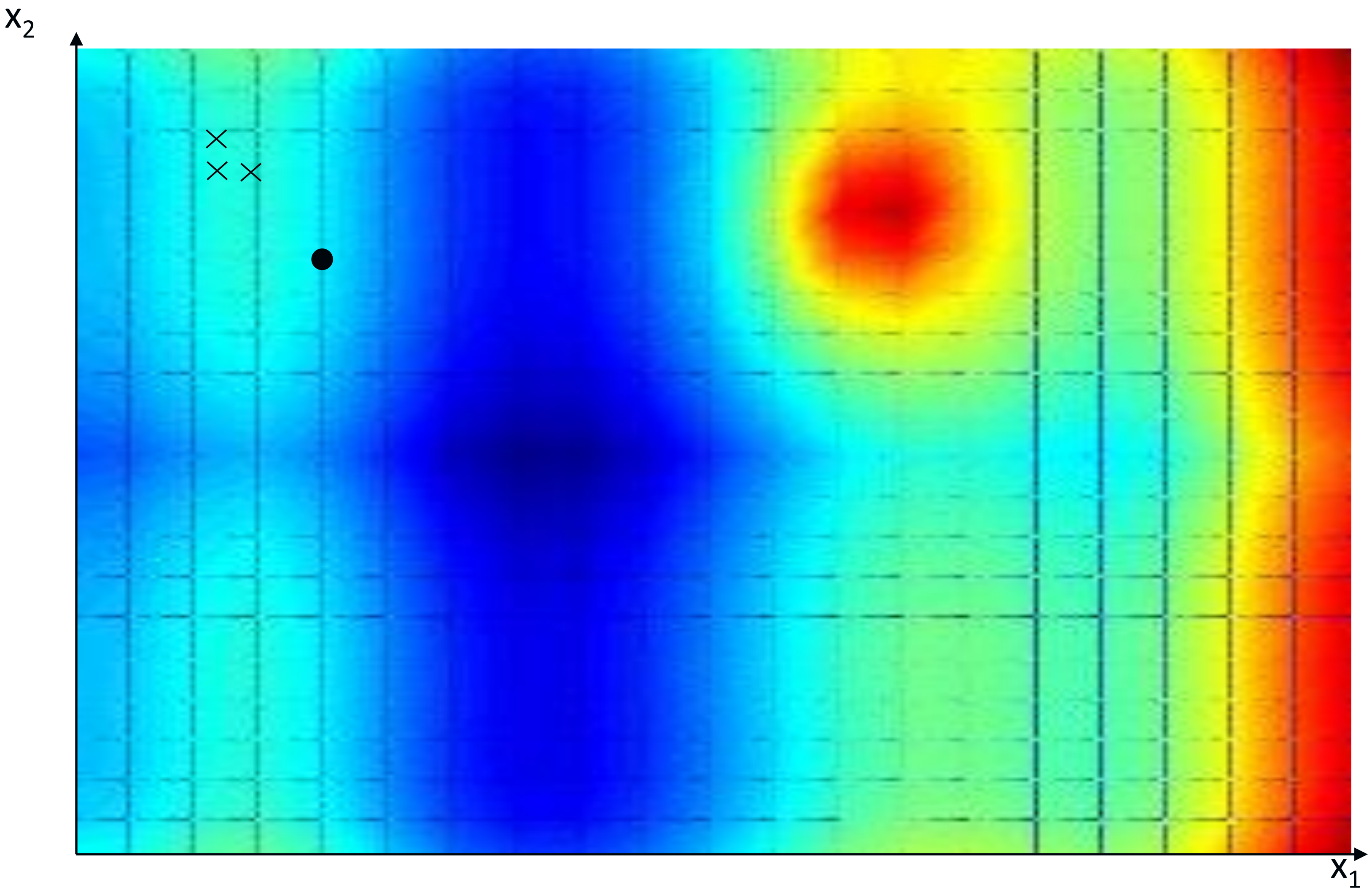
APPROXIMATE GRADIENTS: FINITE DIFFERENCES



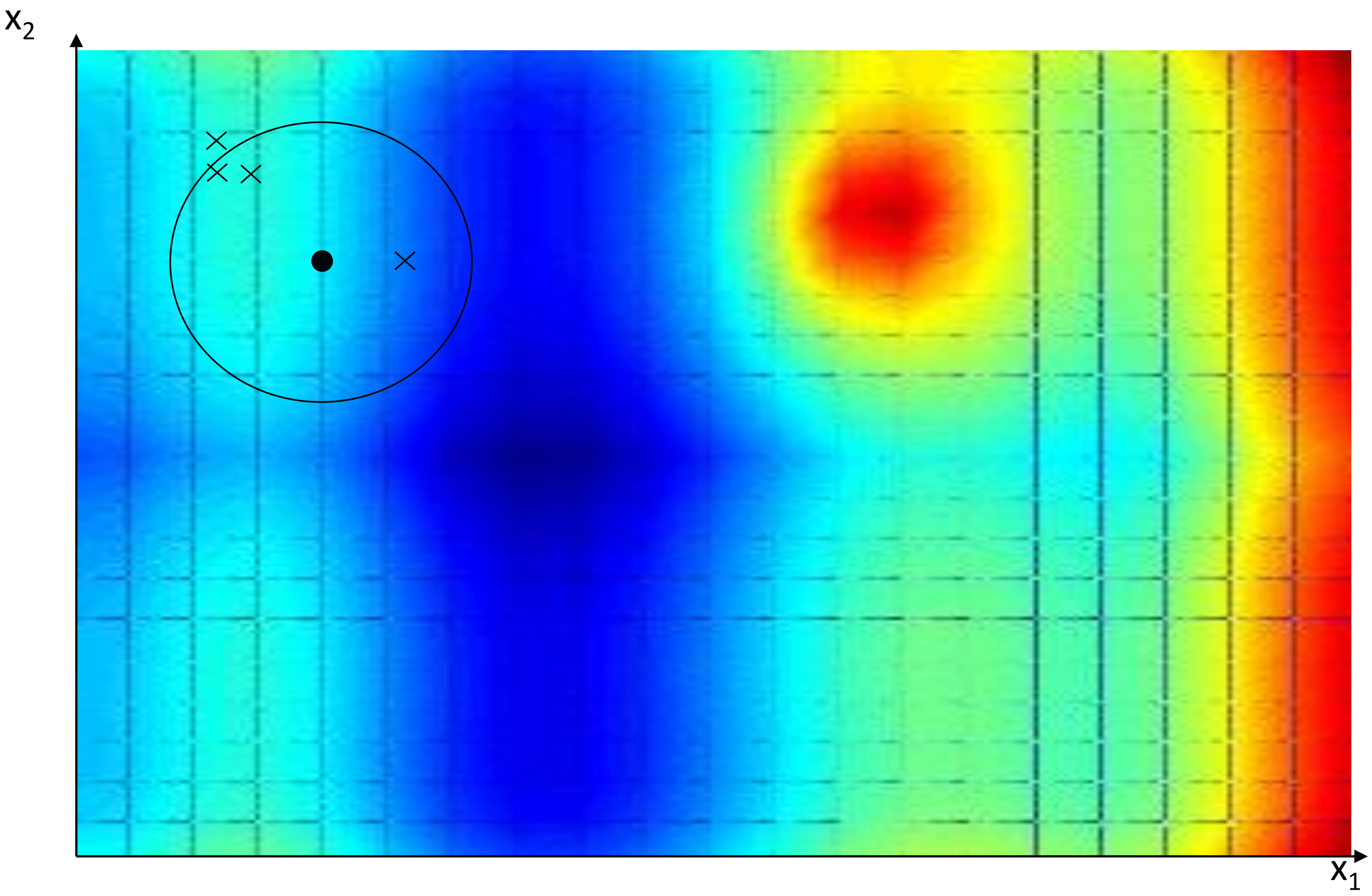
APPROXIMATE GRADIENTS: MODEL OF THE OBJECTIVE FUNCTION



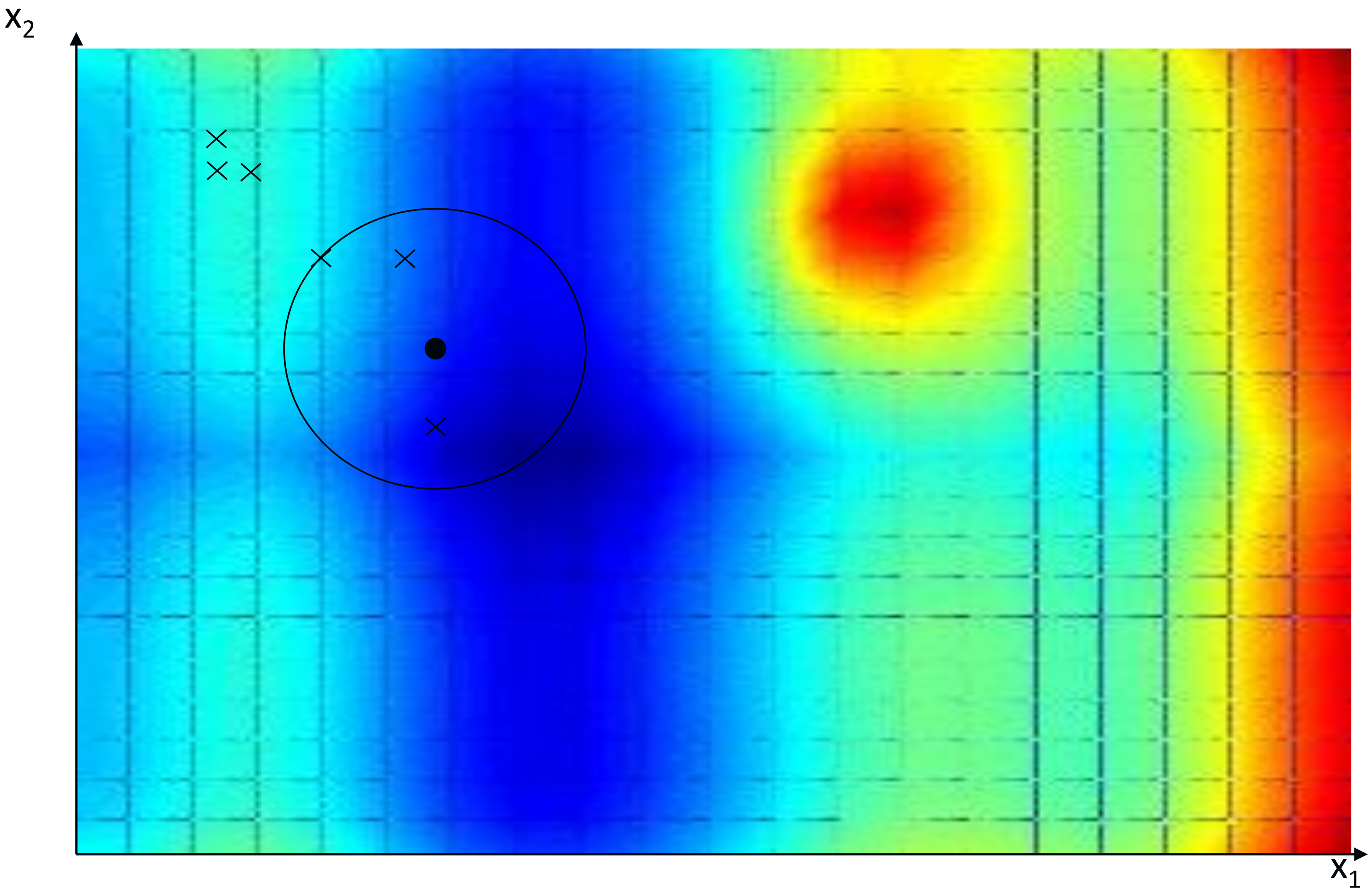
APPROXIMATE GRADIENTS: MODEL OF THE OBJECTIVE FUNCTION



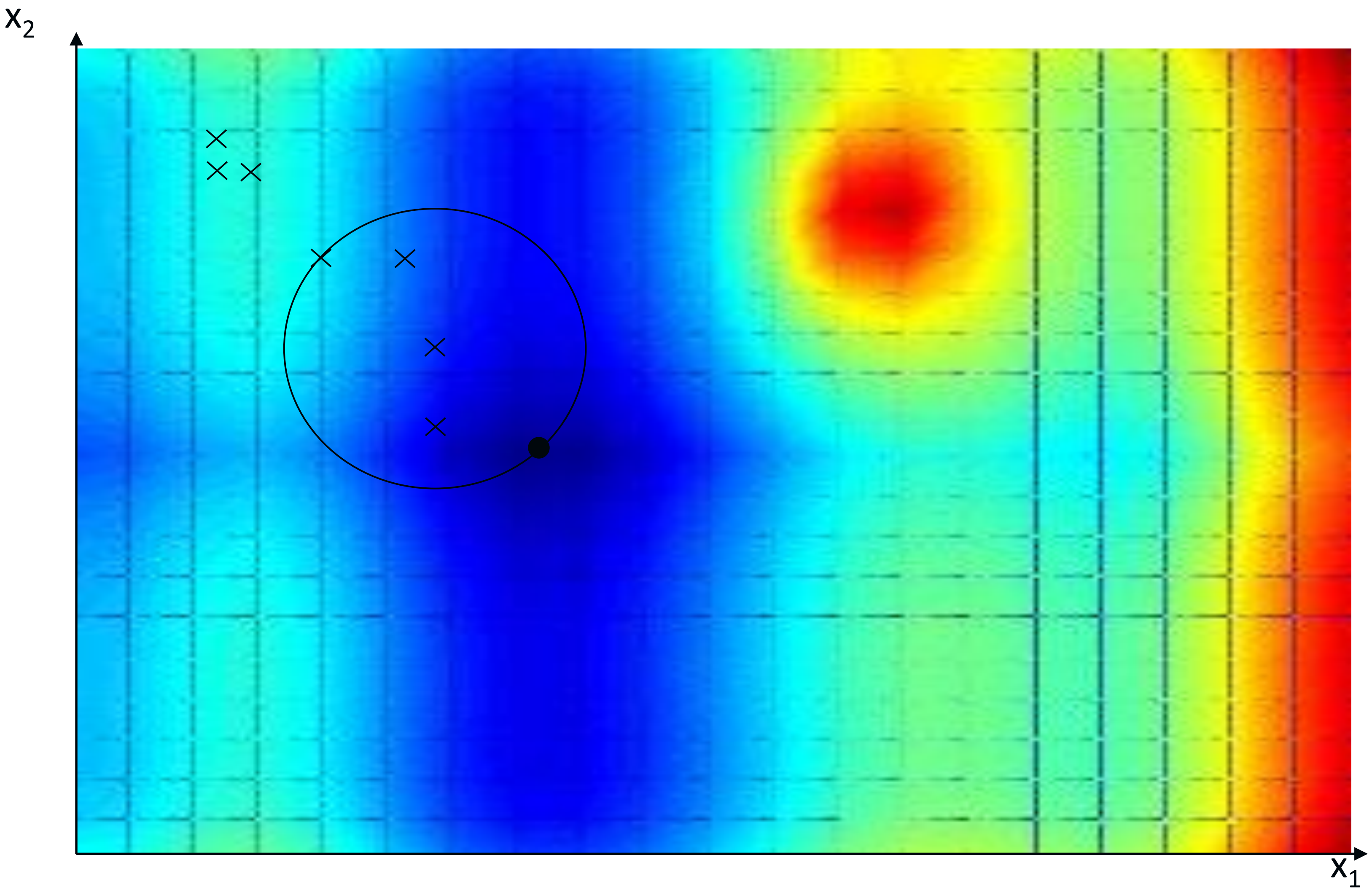
APPROXIMATE GRADIENTS: MODEL OF THE OBJECTIVE FUNCTION



APPROXIMATE GRADIENTS: MODEL OF THE OBJECTIVE FUNCTION

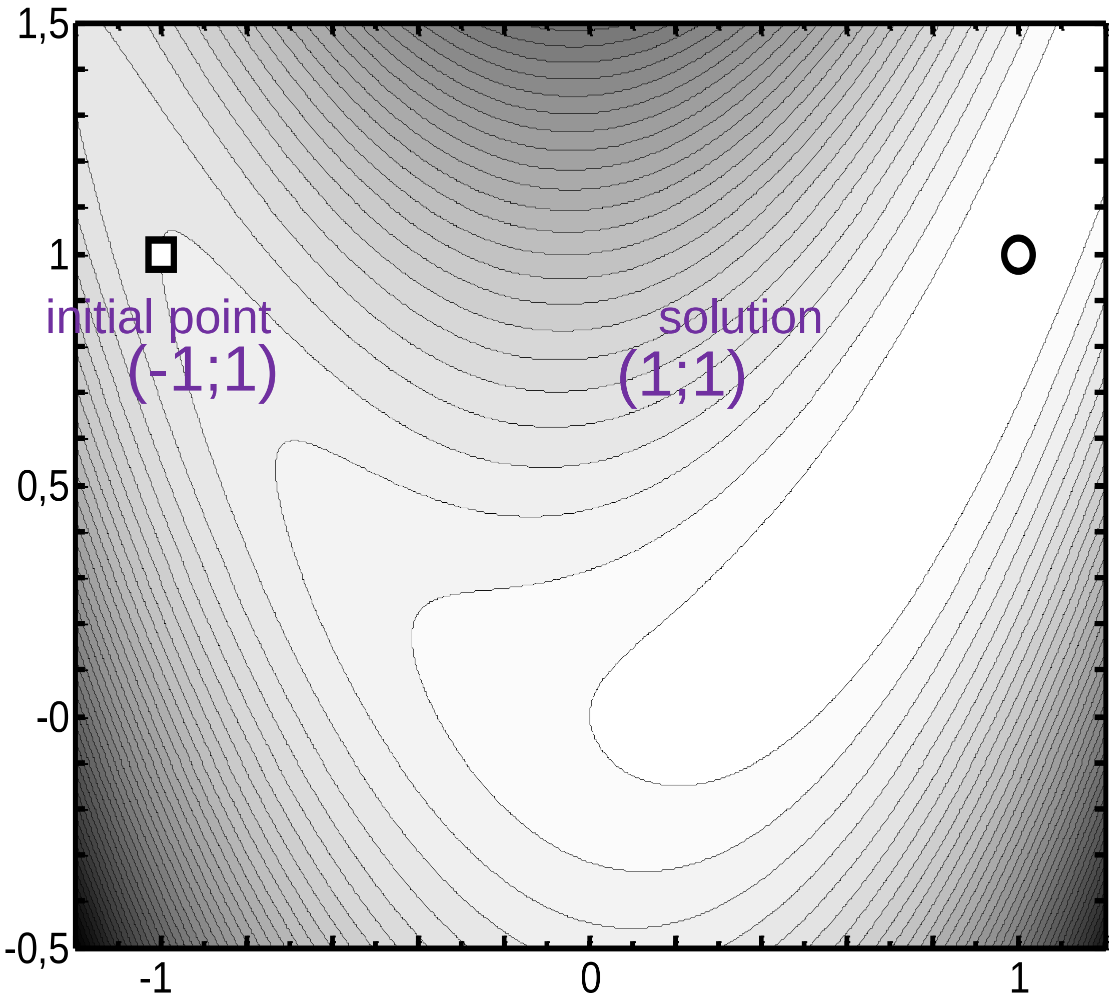


APPROXIMATE GRADIENTS: MODEL OF THE OBJECTIVE FUNCTION



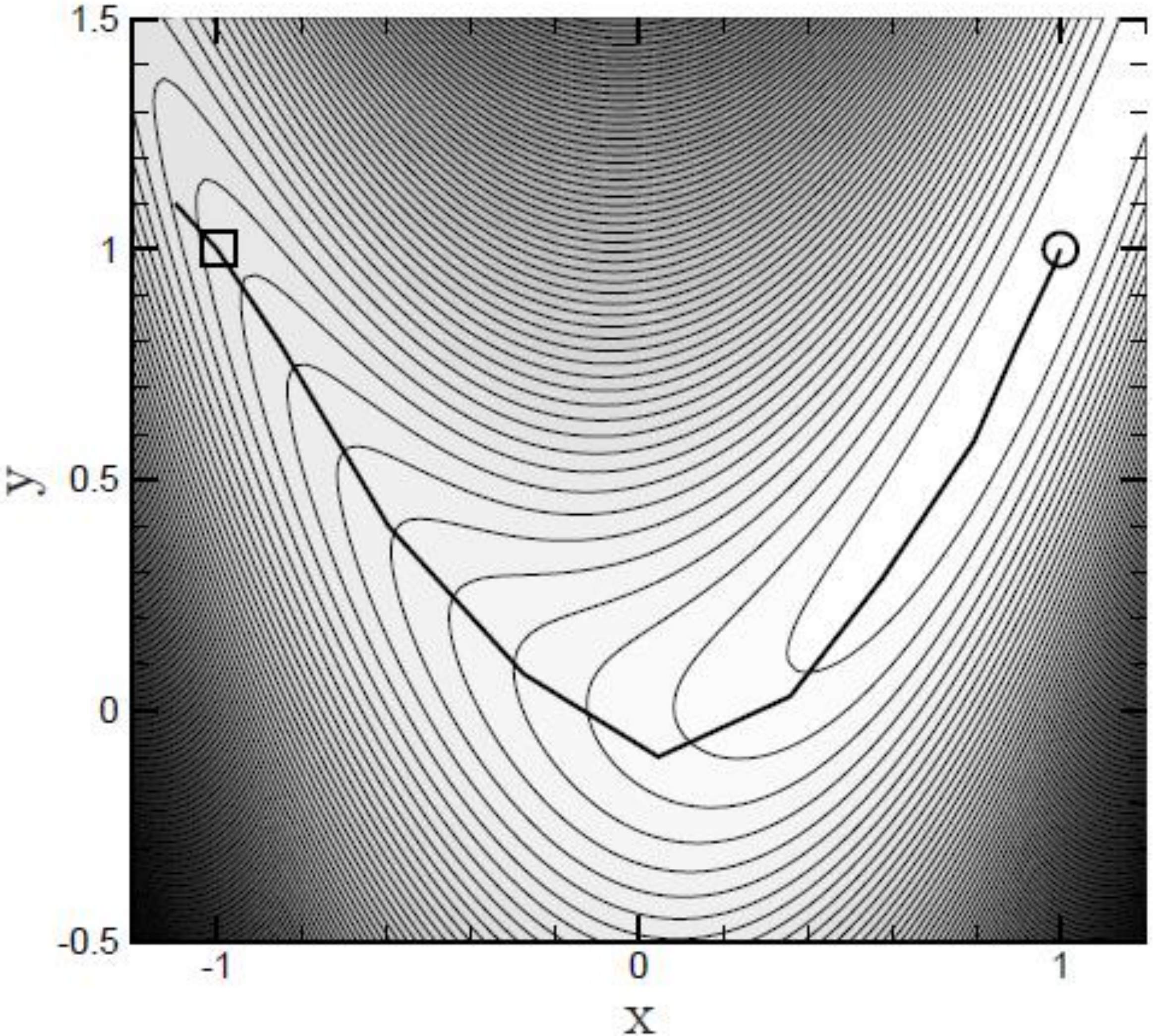
A CLASSICAL EXAMPLE FOR NL OPTIMIZATION

Rosenbrock function : $\min_{(x,y)}\left((1-x)^2+10(x^2-y)^2\right)$



DERIVATIVE-BASED METHOD

Solution obtained with Newton method

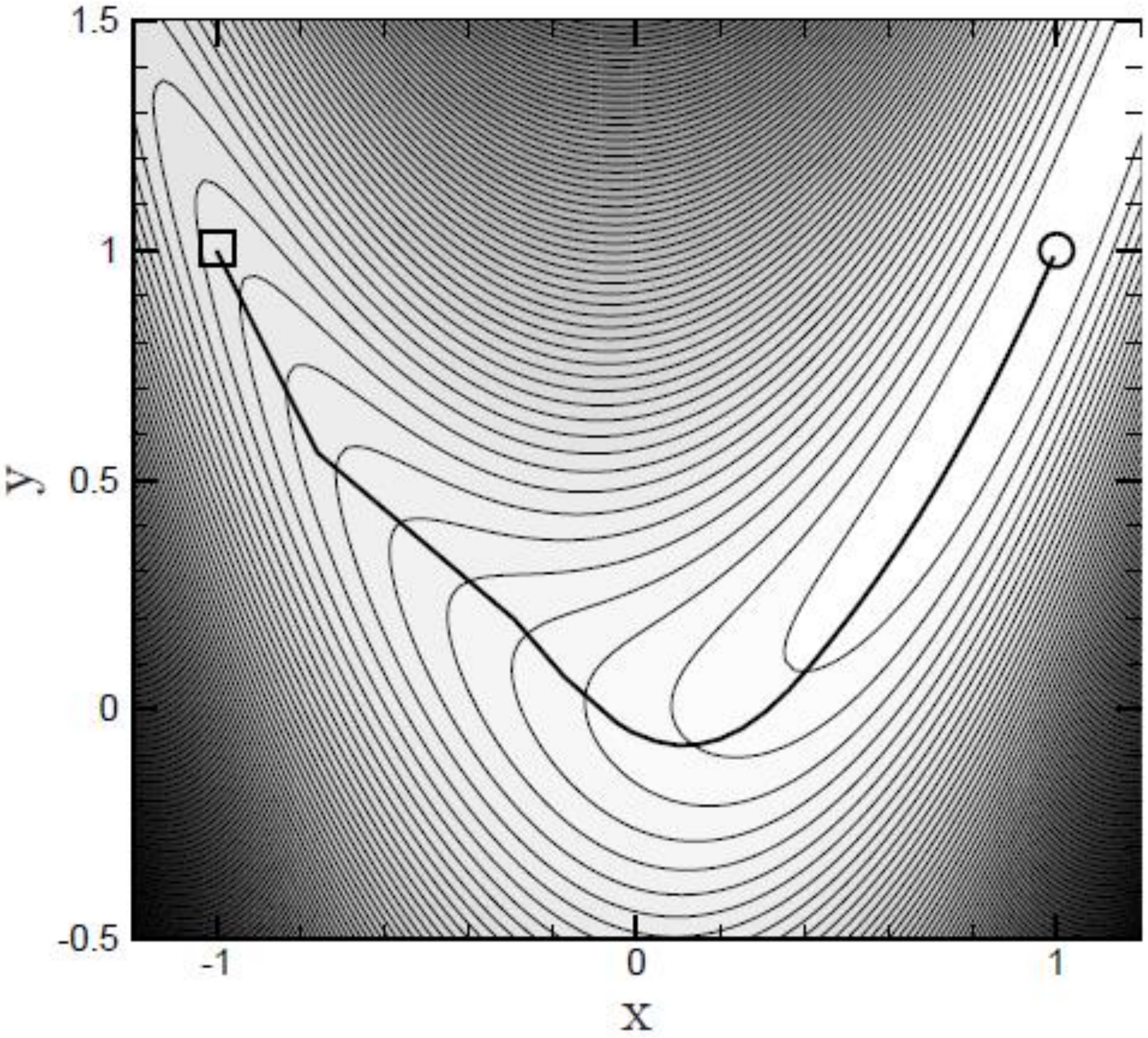


14 iterations

$(x^*, y^*) =$
$(1,000; 0,999)$

DERIVATIVE-BASED METHOD

Solution obtained with BFGS method

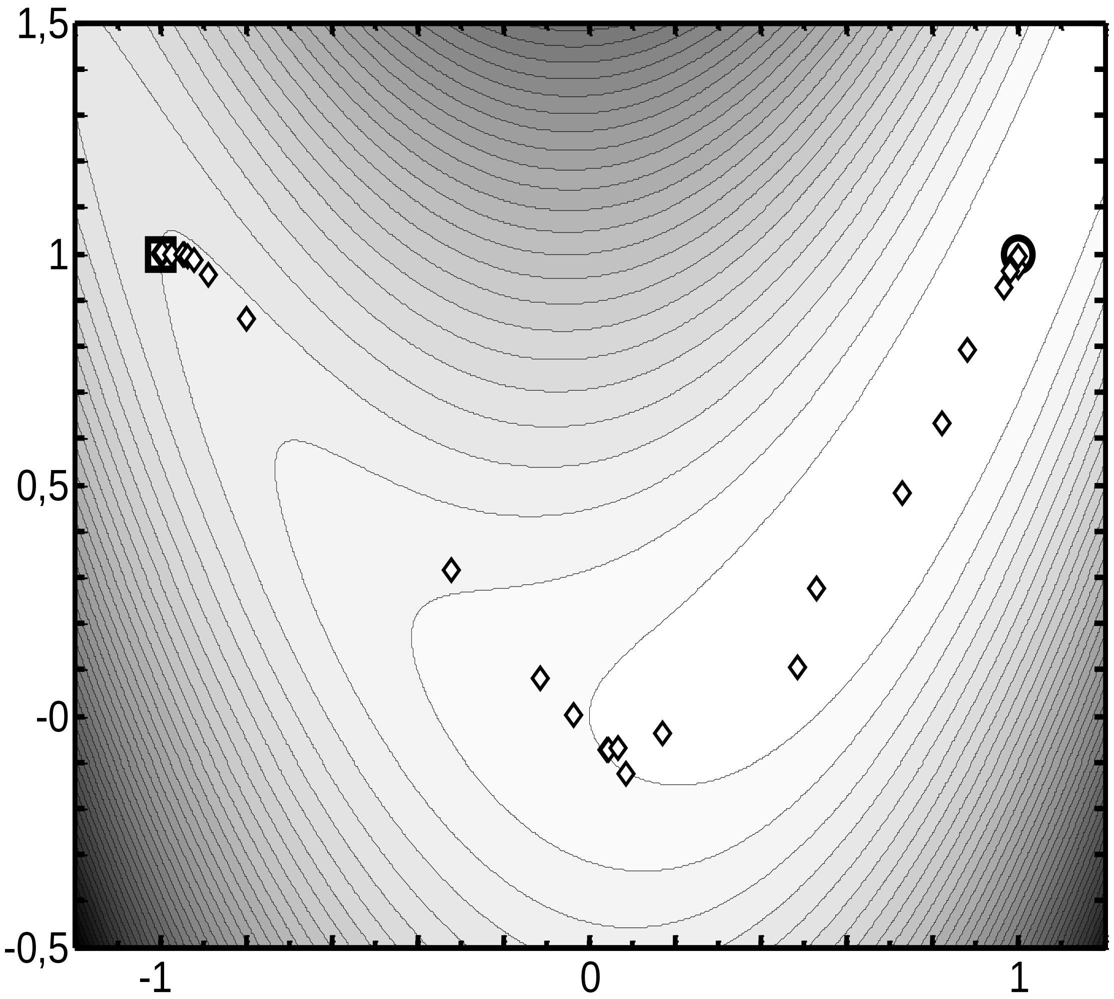


53 iterations

$(x^*, y^*) =$
$(0,995; 0,990)$

APPROXIMATE GRADIENTS

Solution obtained with BFGS method



92 simulations

$(x^*, y^*) =$

$(0,997; 0,994)$

$h = 10^{-3}$

APPROXIMATE GRADIENTS

- Function-evaluation cost: $(n + 1)$ evaluations at each iteration
- Difficulty to choose the finite difference step h
- If noisy function → meaningless approximate gradients

➢ Convergence issues

DFO METHODS

- Use only objective function values
- No gradient approximation
- Sample of points $\{x_i\}_{i=1,\dots,p}$ ➔ simulations $\{x_i\}_{i=1,\dots,p}$ ➔ new iterate x^k

DFO METHODS

- *(Standard derivative-based methods with approximate gradients)*
- Direct Search methods
  - Nelder Mead Simplex
  - Pattern Search
- Surrogate optimization / model-based DFO methods
  - Local model of the objective function
  - Global model of the objective function
- Stochastic DFO methods
  - Evolutionary strategies
  - Simulated annealing

DIRECT SEARCH METHODS

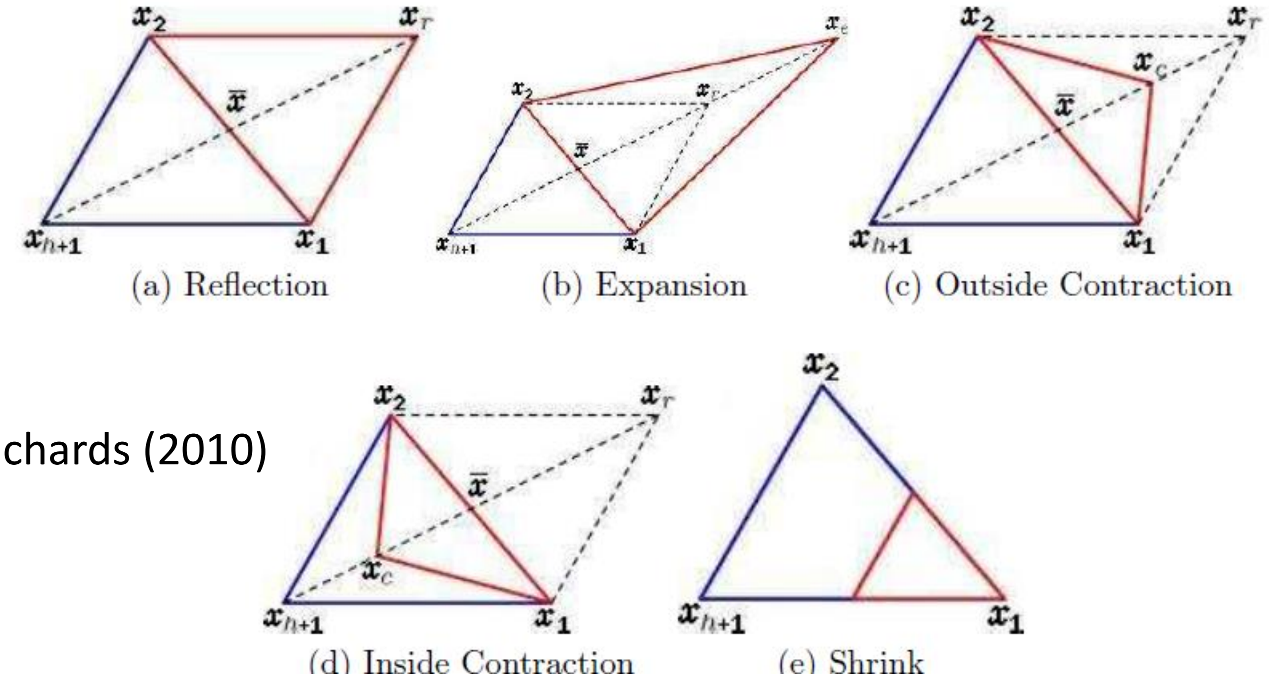
Nelder-Mead simplex algorithm

- based on the comparison of objective function values on a $(n + 1)$ simplex:

$$f(x_1) \leq f(x_2) \ldots \leq f(x_{n+1})$$

- Attempt to improve the worst objective function value $f(x_{n+1})$:
  x_{n+1} is replaced by a point belonging to the line $(\bar{x}, x_{n+1})$
  with $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, centroid of the best n points

→ expansion, reflection or contraction of the simplex at each iteration



(a) Reflection

(b) Expansion

(c) Outside Contraction

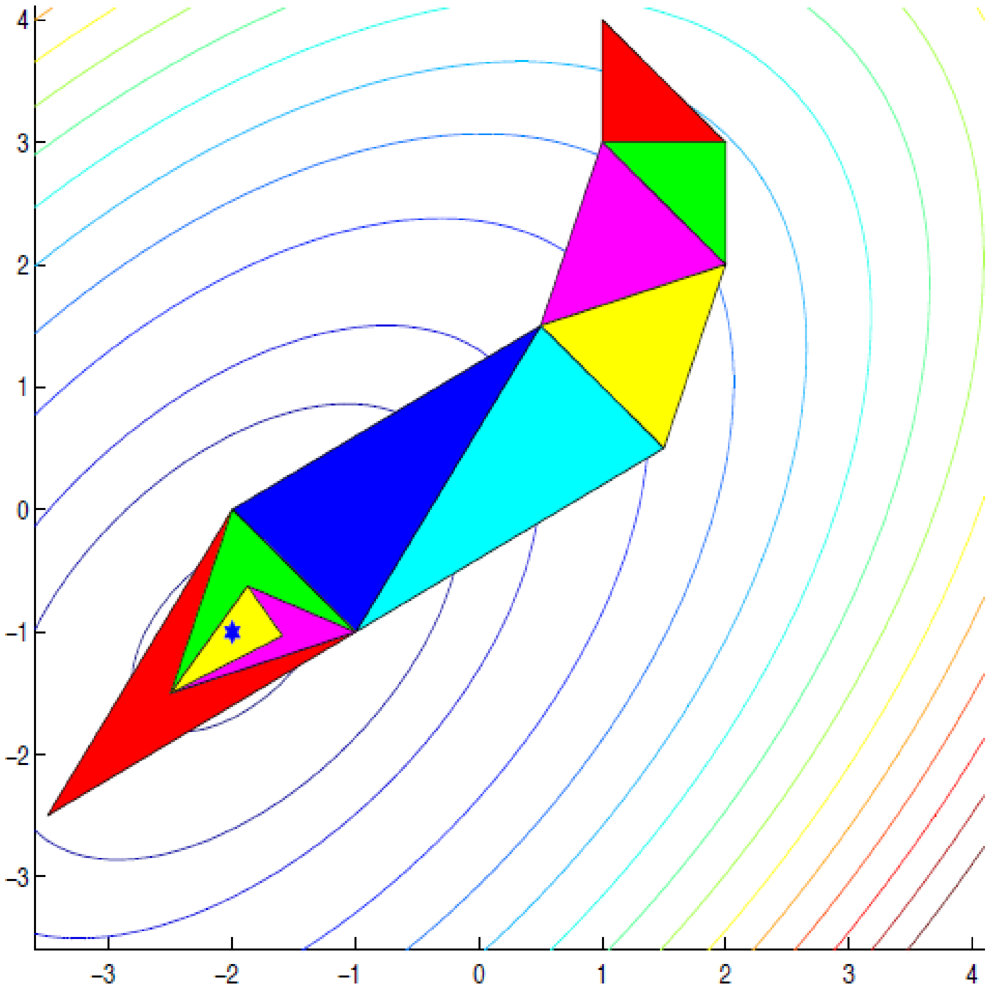
(d) Inside Contraction

(e) Shrink

Source : Richards (2010)

DIRECT SEARCH METHODS

Nelder-Mead simplex algorithm



Source : Wright (2013)

Nelder-Mead simplex algorithm

- Termination conditions:
  - function values at vertices are close to each other
  - or simplex becomes too small

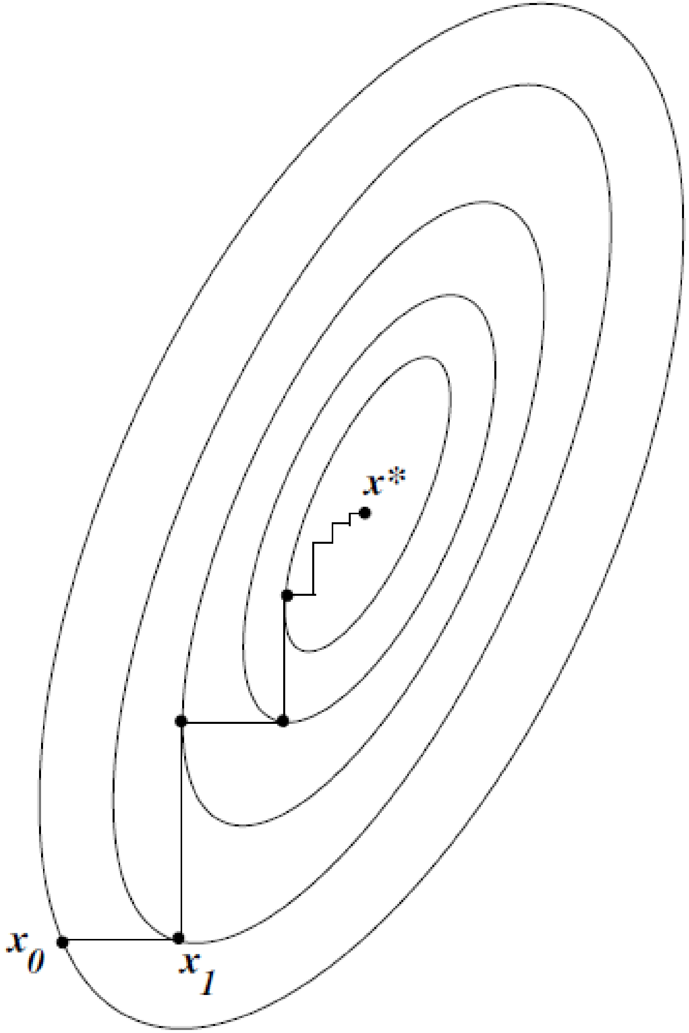
- Simulation cost:
  - k=0 and for any shrinkage step: $(n+1)$ evaluations
  - 1 or 2 evaluations for all other steps

- Limited convergence results (only for $n = 1$ or $n = 2$)
  see Torczon (1991) for other simplex methods with better convergence results
- Lot of failure examples

DIRECT SEARCH METHODS

Linesearch derivative free methods: *e.g.* coordinate search method

$$
\begin{aligned}
&x^1 = x^0 + \alpha^0 e^1\\
&x^2 = x^1 + \alpha^1 e^2\\
&\vdots\\
&x^n = x^{n-1} + \alpha^{n-1} e^{n-1}\\
&x^{n+1} = x^n + \alpha^n e^1\\
&\qquad\qquad\vdots
\end{aligned}
$$



α^k is chosen to produce a sufficient decrease

$$f\left(x^k + \alpha^k e^{k-1}\right) < f\left(x^k\right) - \rho\left(\alpha^k\right)$$

with $\rho(t) \geq 0$ increasing function of t, $\rho(t)/t \underset{t\to 0}{\longrightarrow} 0$

→**Inefficient:** coordinate direction (almost) $\perp \nabla f\left(x^k\right)$

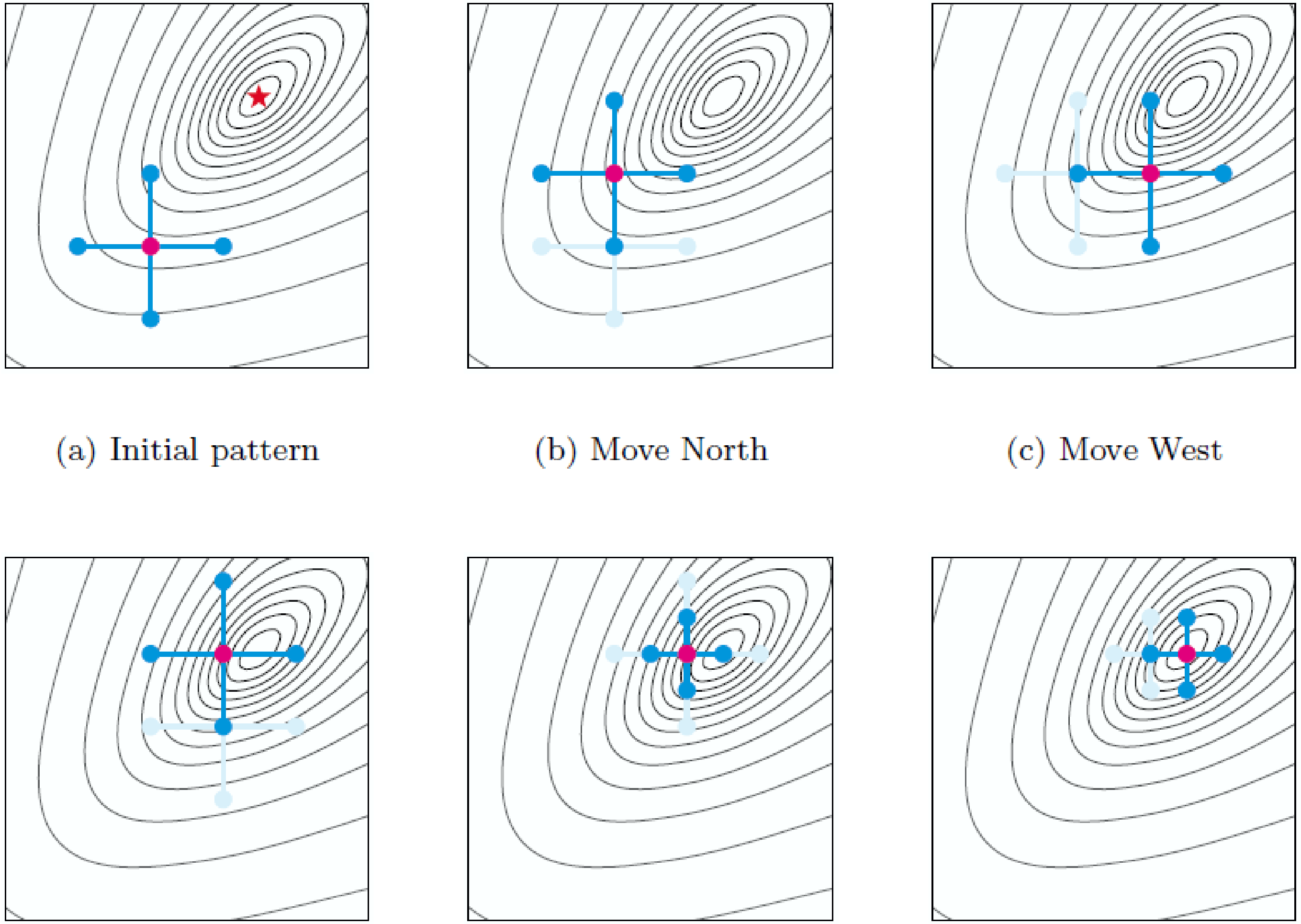
→Efficient when the variables are essentially uncoupled

Pattern search methods

- Motivation: parallelisation of function evaluations
  Instead of one search direction $s^l (= e^l)$ in linesearch, explore a set of directions D^k
  e.g. $D^k = \{e^1, e^2, \dots, e^n, -e^1, -e^2, \dots, -e^n\}$
- At each iteration, for a given mesh step α^k:
  - **Search step** (OPTIONNAL): evaluate the objective function on a finite number of points with any method: along a given direction, on a simplex, …
  - **Poll Step:** if no better point if found in optionnal search step, search for a better point in the D^k directions: $x^k + \alpha^k s^i \ , s^i \in D^k$
    - If no better point is found (no smaller function value):
      $x^{k+1} = x^k$ and decrease the mesh size α^k
    - else $x^{k+1} = x^k + \alpha^k s^i$ and increase the mesh size

DIRECT SEARCH METHODS

Pattern search methods



(a) Initial pattern

(b) Move North

(c) Move West

(d) Move North

(e) Contract

(f) Move West

Source : Kolda et al. (2003)

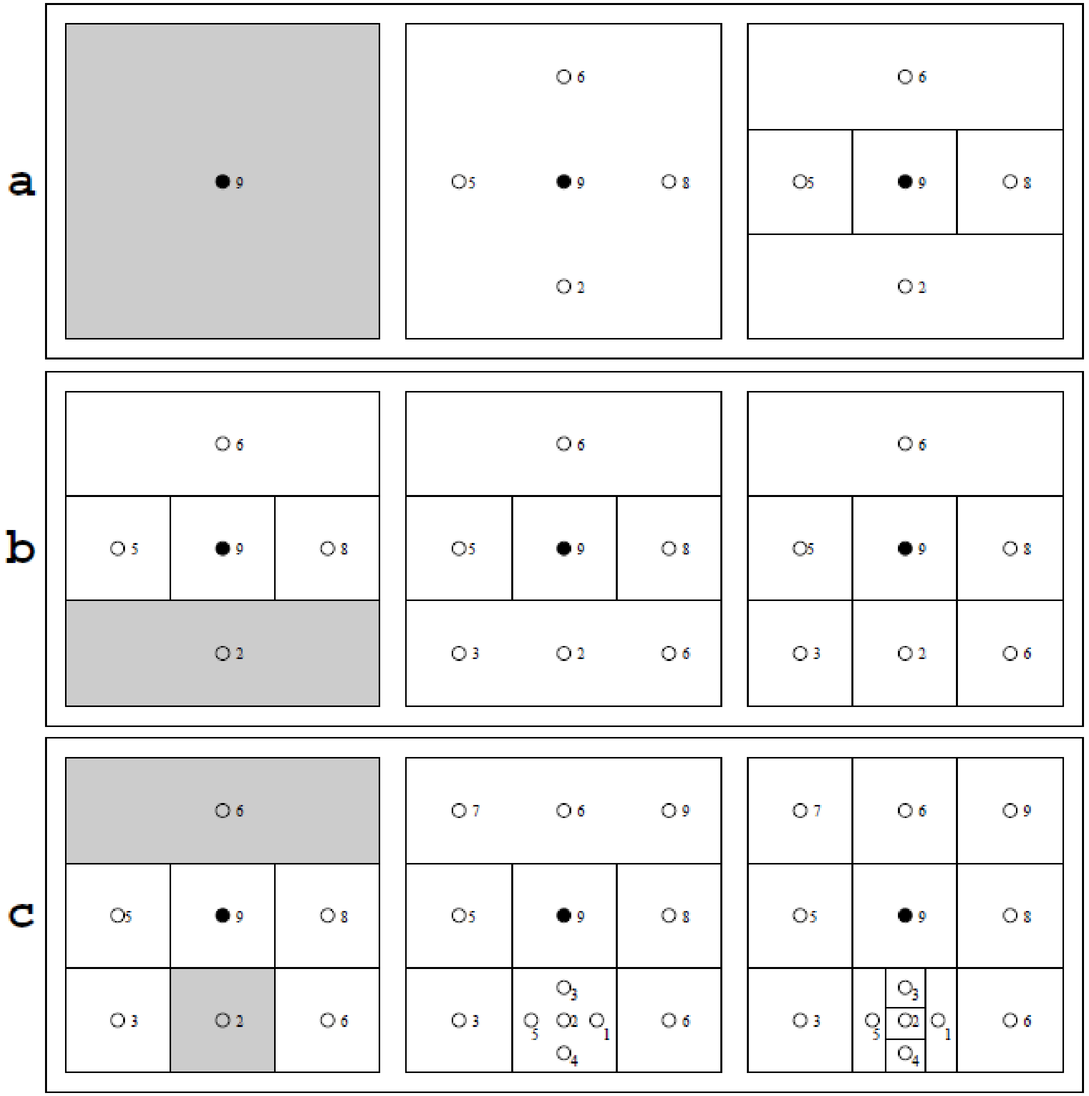
DIRECT SEARCH METHODS

Pattern search methods: DIRECT

- = Dividing RECTangles

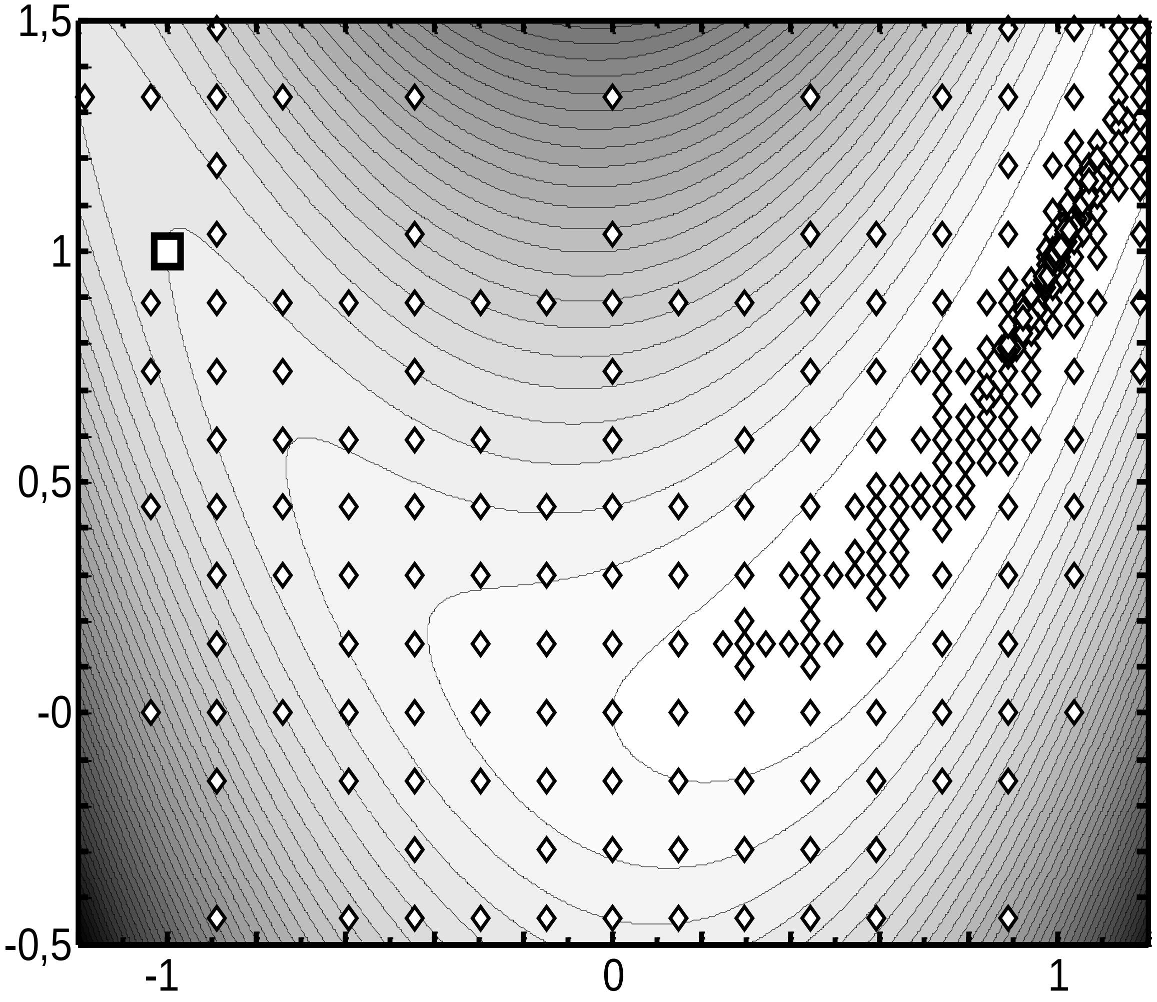
- Divide each side of the « rectangle(s) » associated with the smallest function values into 3 in order to define sub-rectangles
- Evaluate the center of the new rectangles
- Stopping criteria: minimal size of the rectangles

- Global convergence for continuous functions
- High evaluation cost



Source : Perttunen et al. (1993)

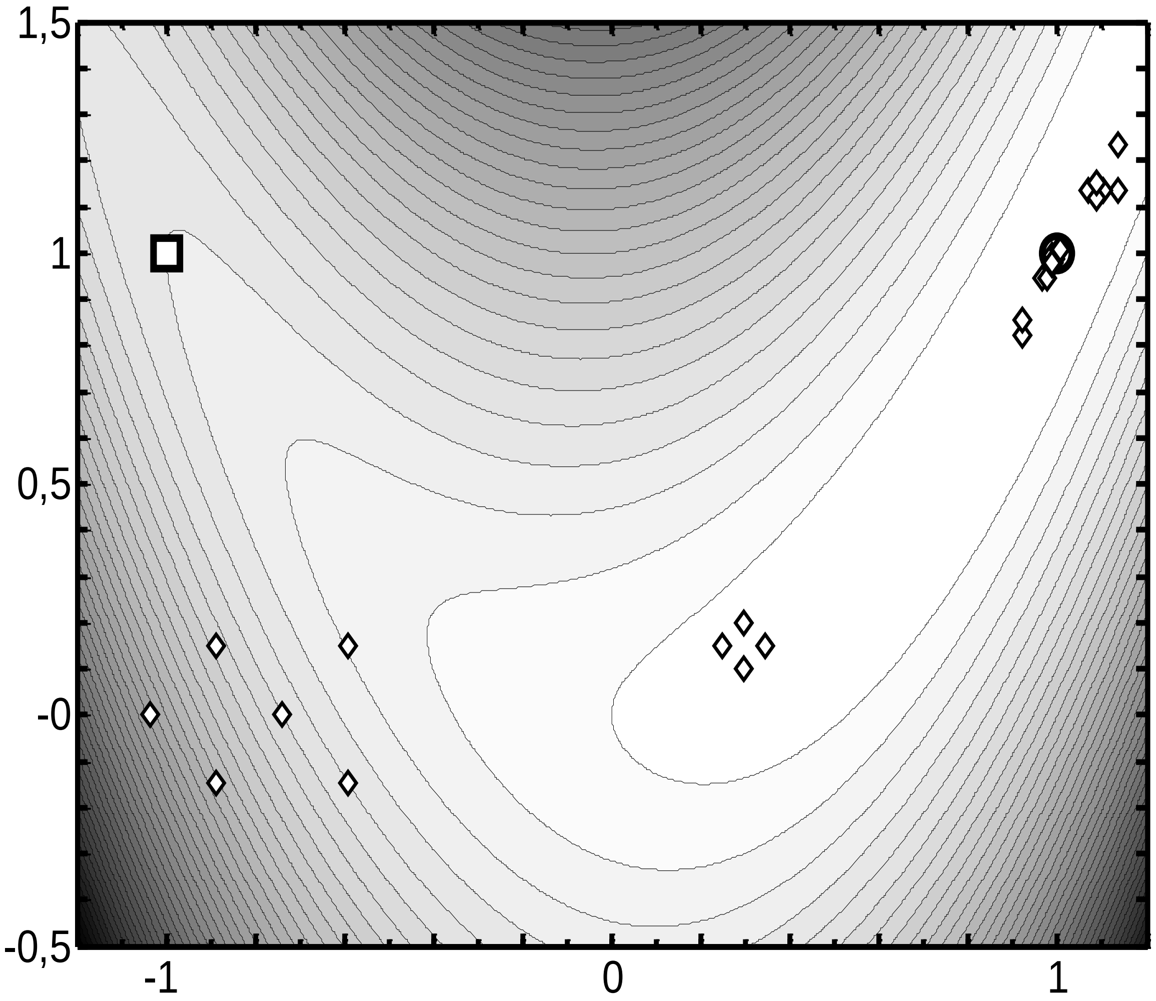
DIRECT ALGORITHM



517 simulations

$(x^*, y^*) = (1,004; 1,008)$

DIRECT ALGORITHM



Last iterations

DIRECT SEARCH METHODS

Pattern search methods

- Popular methods
  - Easy to implement
  - Easy to parallelize

- But expensive in terms of simulations

- often coupled with a surrogate model in the search step

DFO METHODS

- *(Standard derivative-based methods with approximate gradients)*
- Direct Search methods
  - Nelder Mead Simplex
  - Pattern Search
- Surrogate optimization / model-based DFO methods
  - Local model of the objective function
  - Global model of the objective function
- Stochastic DFO methods
  - Evolutionary strategies
  - Simulated annealing

SURROGATE OPTIMIZATION METHODS

Optimization methods based on a surrogate model of the objective function

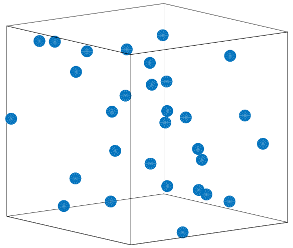
- to limit the number of evaluations of the objective functions
- the model is updated during the iterations based on new simulations
- The model is either global or local

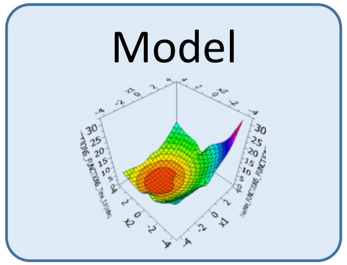
SURROGATE OPTIMIZATION METHODS

Global models

- **Design of experiment technique**
  choose evaluation points to be used to compute the initial model

  space filling design (maximin criterion)

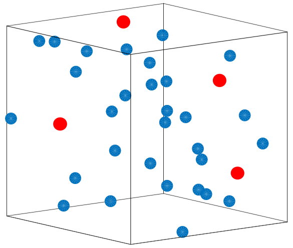




- **Regression**
  choose a model type

  Gaussian process or kriging, Radial Basis Function (RBF), Neuronal Networks

- **Sampling criterion**
  choose new point(s) to evaluate for the update of the model
  minimum of the current model, maximum of the error prediction ...



Global models: Gaussian process (kriging)

- **Assumption:** the objective function is assumed to be a realization of a Gaussian random process (GP) with parametric mean function and stationary covariance function

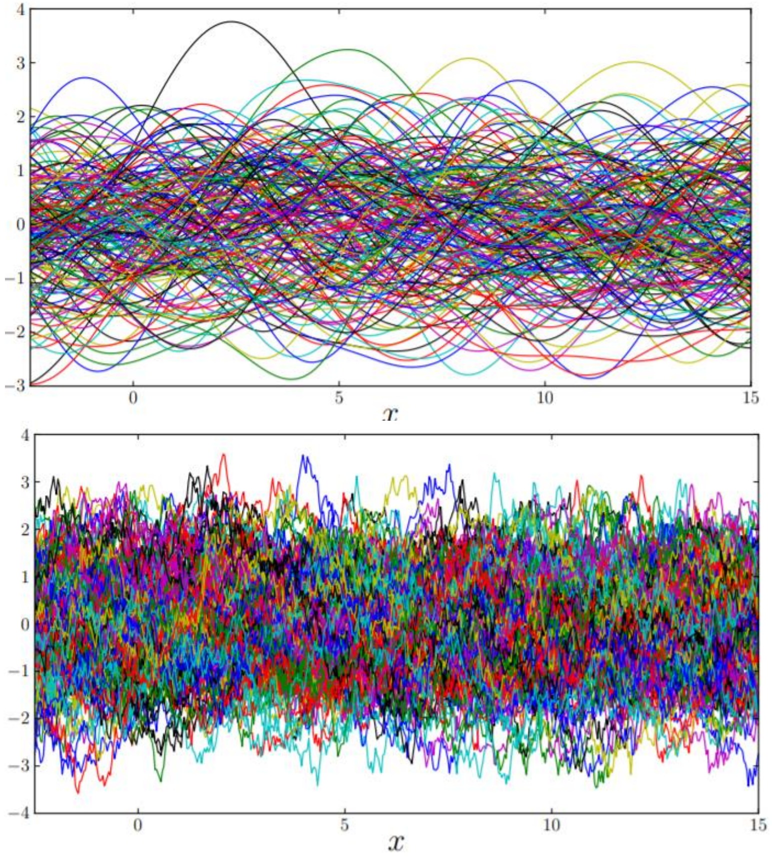
$$F(x) = \beta^T r(x) + Z(x)$$

Regression
(*e.g.* polynomial d=1)

stochastic part
zero-mean, stationary covariance

$$\mathrm{Cov}_Z(x, x') = \sigma^2 \rho(\|x - x'\|)$$

$$\text{e.g. } \mathrm{Cov}_Z(x, x') = \sigma^2 \mathrm{e}^{-\theta(x-x')^2}$$



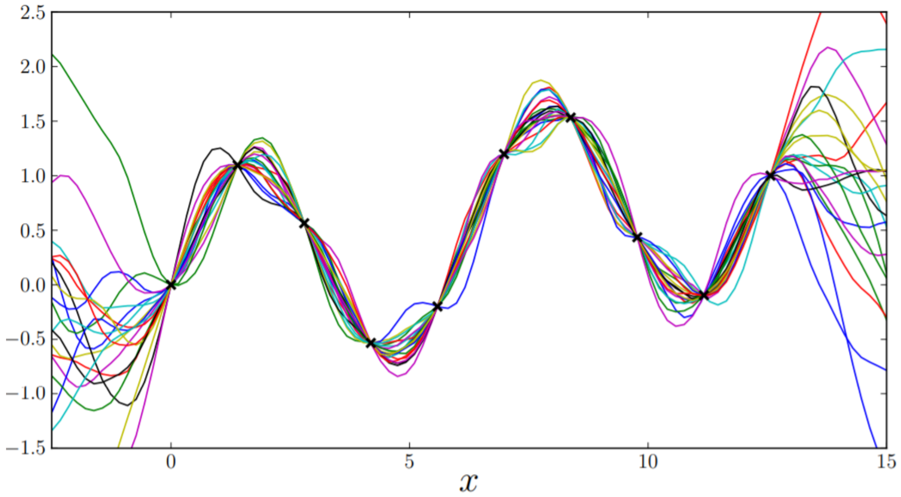
Global models: Gaussian process (kriging)

- **Assumption:** the objective function is assumed to be a realization of a Gaussian random process (GP) with parametric mean function and stationary covariance function

$$F(x) = \beta^T r(x) + Z(x)$$

- **The surrogate model** is the conditional expectation of the GP

$$\hat{F}(x) = E\left(F(x) \middle| \left(x_i, f(x_i)\right)_{i=1,\dots,p}\right)$$
$$= \beta^T\, r(x) + k^T(x) K^{-1}\left(Y_p - R\beta\right)$$



avec $R = \left(r_j(x_i)\right)_{i,j}$, $K = \left(\rho(x_i, x_j)\right)_{i,j}$, $k(x) = (\rho(x, x_1), \dots, \rho(x, x_p))$

Global models: Gaussian process (kriging)

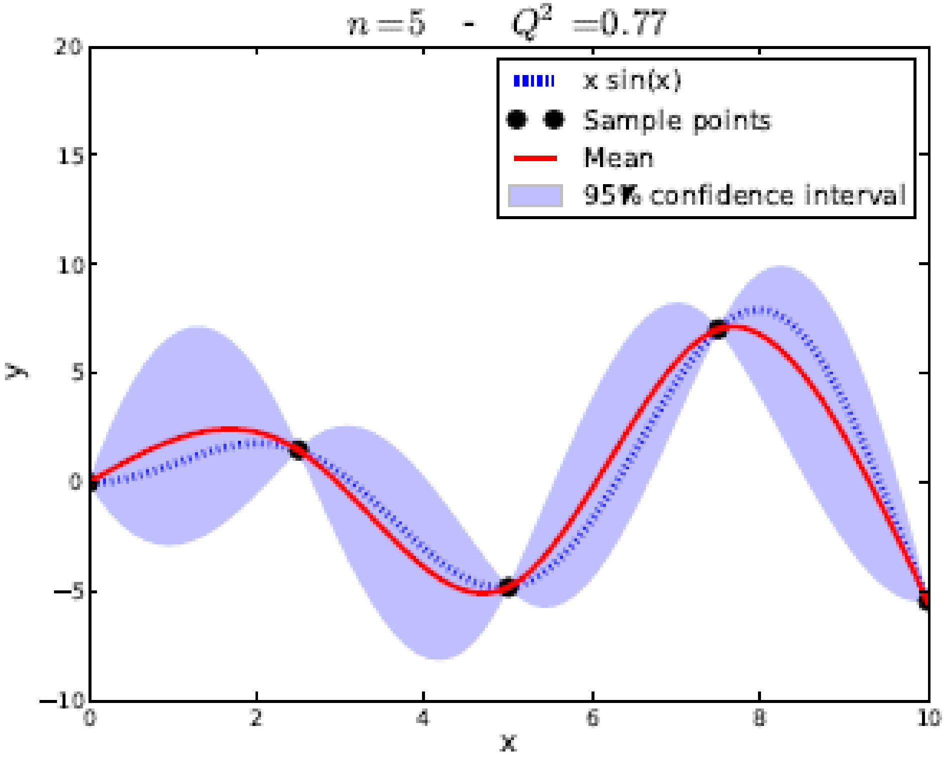
- **Assumption:** the objective function is assumed to be a realization of a Gaussian random process (GP) with parametric mean function and stationary covariance function

$$F(x) = \beta^T r(x) + Z(x)$$

- **The surrogate model** is the conditional expectation of the GP

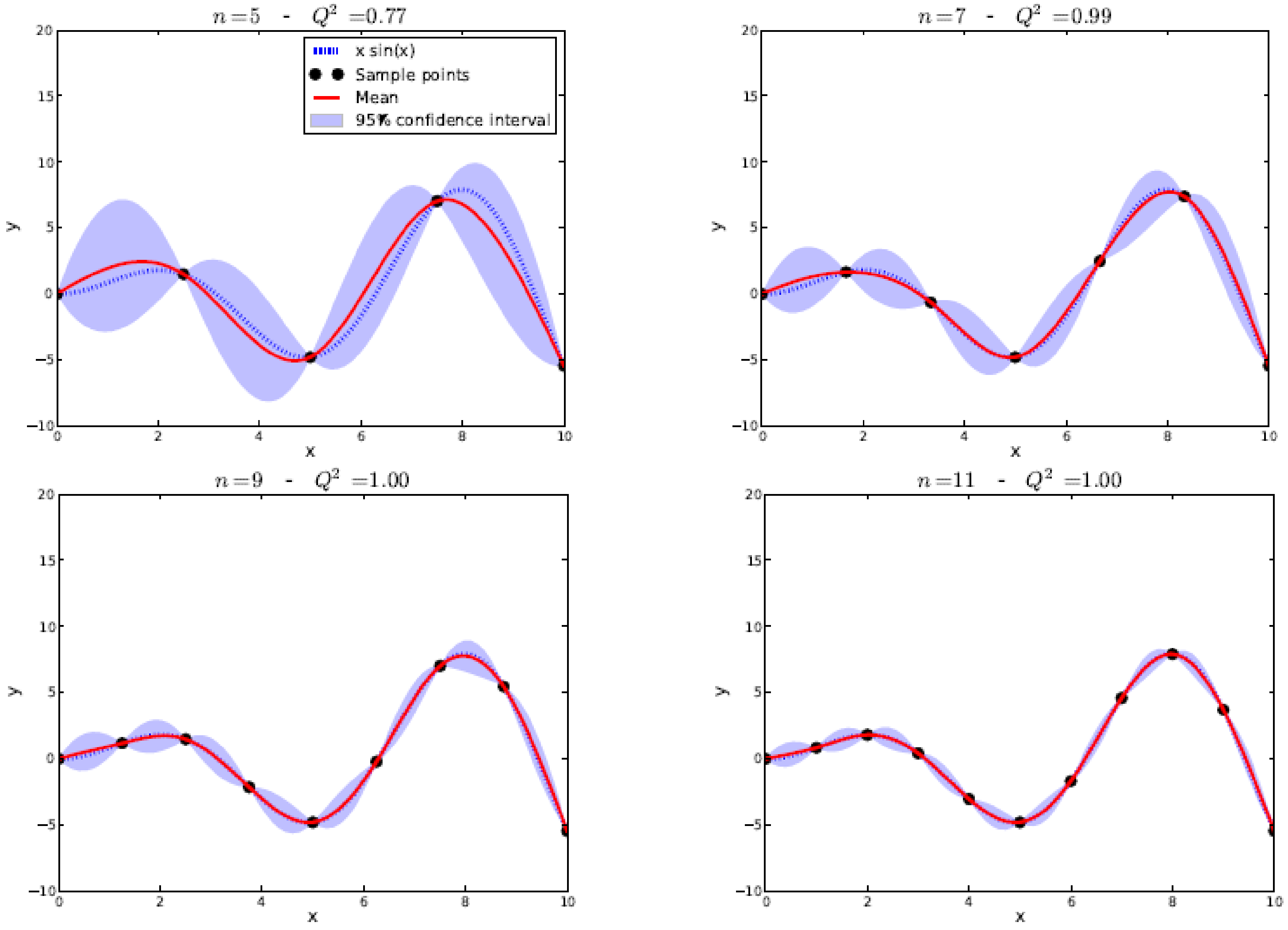
$$\hat{F}(x) = E\left(F(x)\middle|\left(x_i, f(x_i)\right)_{i=1,\dots,p}\right)$$
$$= \beta^T\, r(x) + k^T(x)K^{-1}\left(Y_p - R\beta\right)$$

- The variance of GP are used as **error indicators**
$\sigma^2(x) = \sigma^2 - k^T(x)K^{-1}k^T(x)$



avec $R = \left(r_j(x_i)\right)_{i,j}$, $K = \left(\rho(x_i, x_j)\right)_{i,j}$, $k(x) = \left(\rho(x, x_1), \dots, \rho(x, x_p)\right)$

Global models: Gaussian process (kriging)



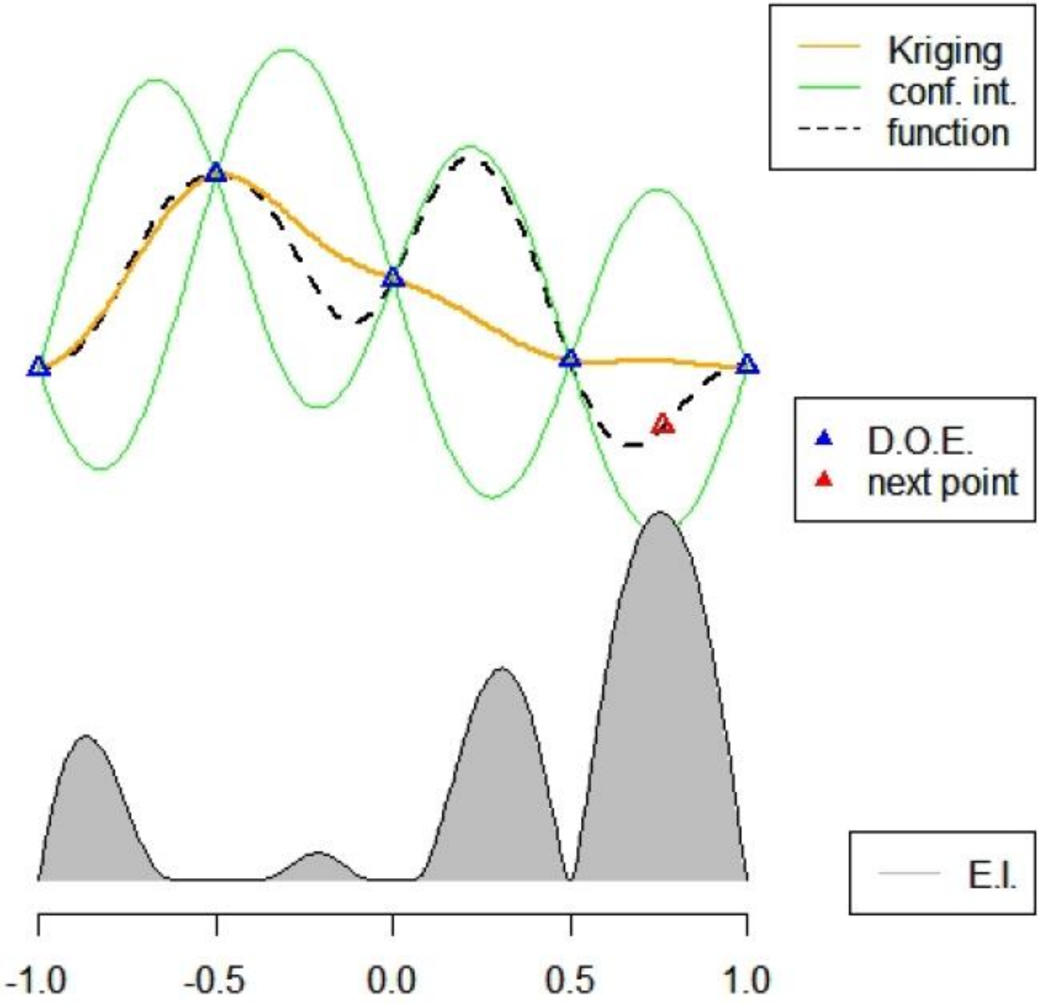
SURROGATE OPTIMIZATION METHODS

Sampling criterion based on Expected Improvement (EI)

a balance between exploration and minimization

$$\mathrm{argmax}\big(EI(x)\big) = E\big(I(x)\big)$$
$$= E\big(\max(0, f_{min} - \hat{F}(x))\big)$$

- EGO (Efficient Global Optimization) or Bayesian Optimization



SURROGATE OPTIMIZATION METHODS

Local models

- **Quadratic interpolation models** built from a set of appropriately chosen sample points

$$\hat{F}_k(s) = c_k + s^T g_k + \frac{1}{2} s^T H_k s,\ \ s \in \mathcal{B}\left(x^k, \Delta\right)\ \text{(trust region)} \quad \text{(TR)}$$

  with $c_k \in \mathbb{R}, g_k \in \mathbb{R}^n$ and $H_k \in \mathbb{R}^{n\times n}$ (symmetric) that satisfy interpolation conditions:

$$\hat{F}_k\left(x_i - x^k\right) = f(x_i)$$

  The matrix of linear system must be non-singular and well conditioned

- **Minimization of the quadratic model in the trust region** $\min\limits_{\|s\|\leq\Delta_k} \hat{F}_k(s)$
- **Update the model** with new evaluations
- **Improve the geometry of the interpolation set** to help with the model interpolation step
  One point is replaced by another one that improves the conditioning of the interpolation matrix

SURROGATE OPTIMIZATION METHODS

Local models

Let s_k be the solution of the (TR) minimization problem

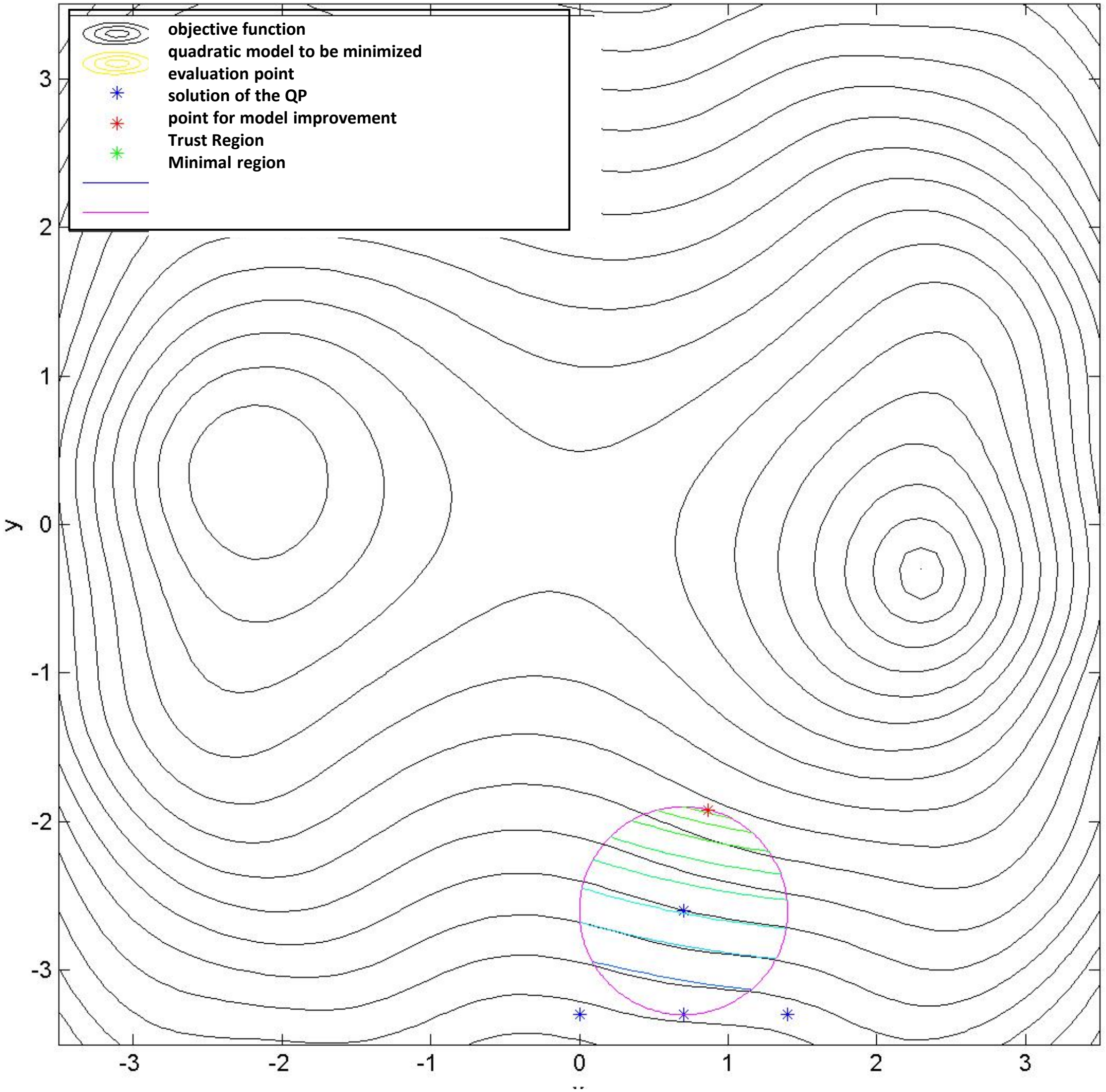
- Predicted model decrease $\hat{F}_k(0) - \hat{F}_k(s^k) = f(x^k) - \hat{F}_k(s^k)$
- Actual function decrease $f(x^k) - f(x^k + s^k)$

The trust region is updated according to the value of $\rho_k = \frac{f(x^k) - f(x^k + s^k)}{f(x^k) - \hat{F}_k(s^k)}$

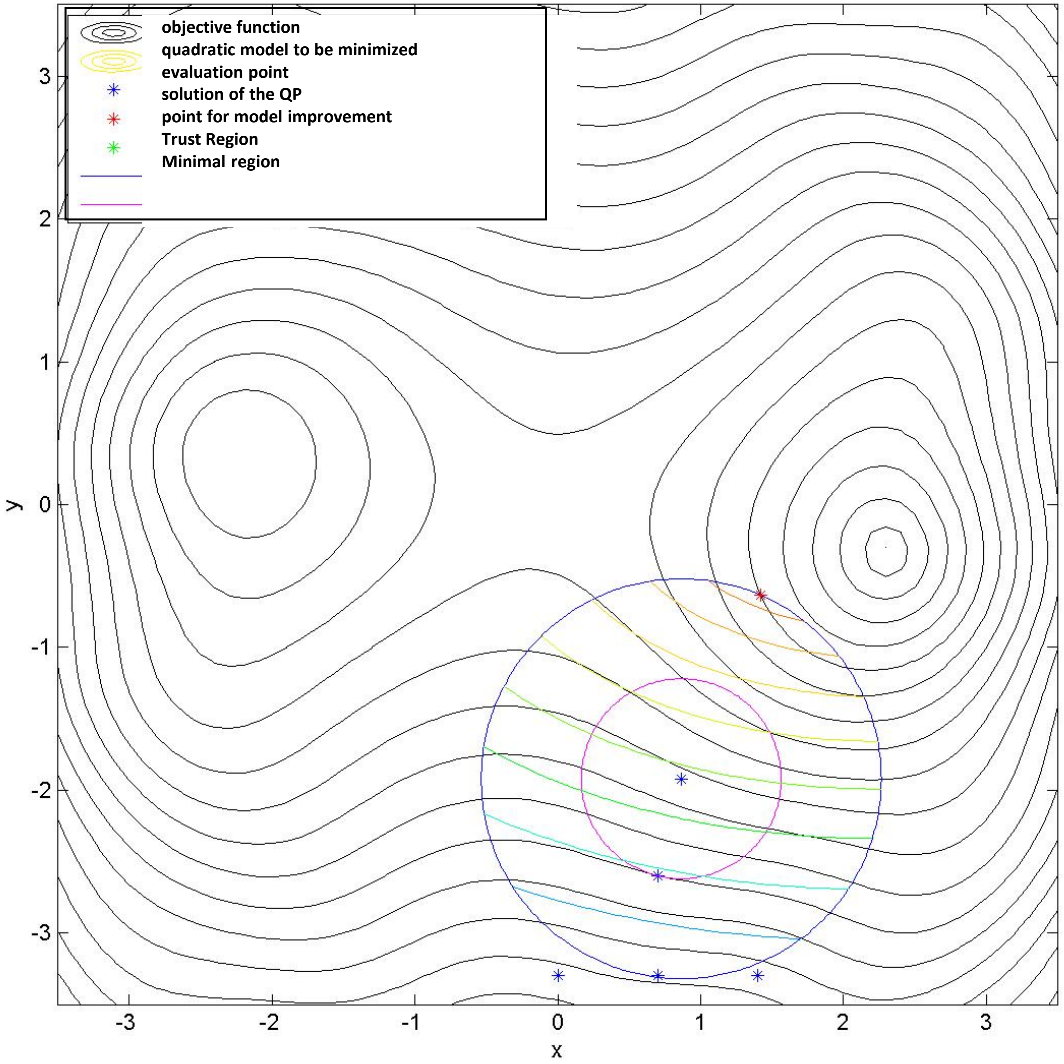
- If $\rho_k \geq \eta$ (successful step): $x^{k+1} = x^k + s^k$, $\Delta^{k+1} \geq \Delta^k$
- If $\rho_k < \eta$ (unsuccessful step): $x^{k+1} = x^k$, Δ^k is reduced or the interpolation set is improved

SURROGATE OPTIMIZATION METHODS

Local models

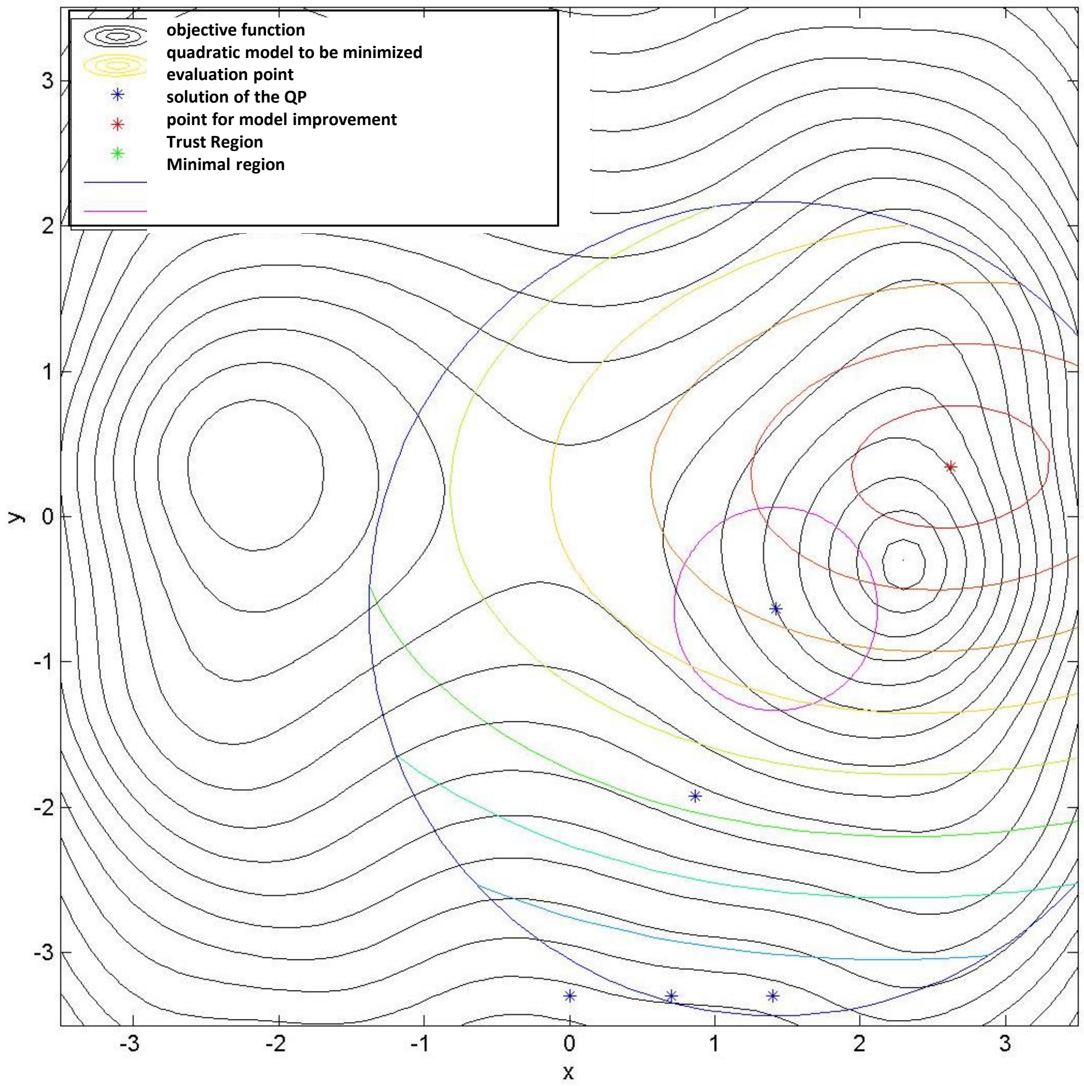


Local models

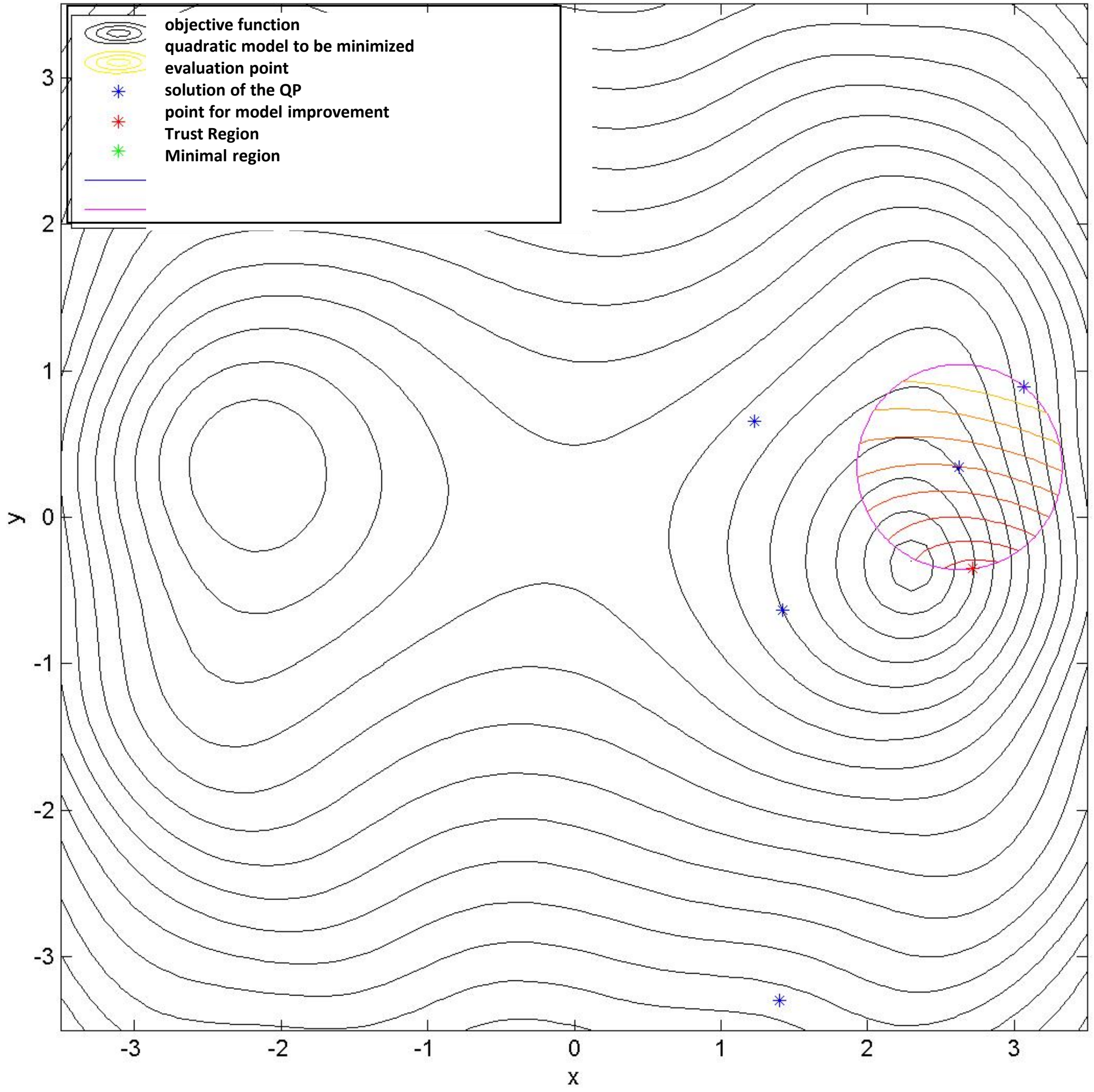


SURROGATE OPTIMIZATION METHODS

Local models

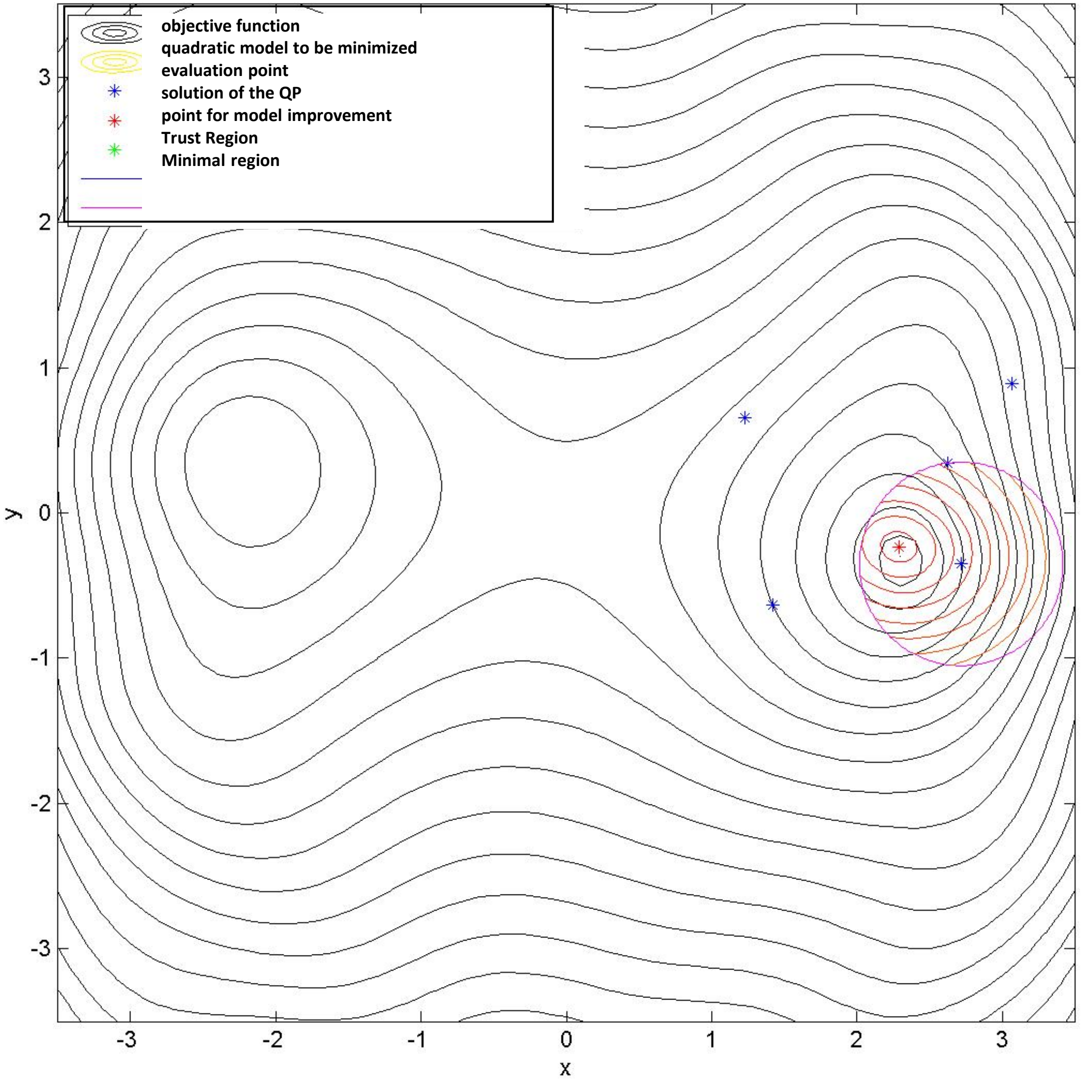


Local models



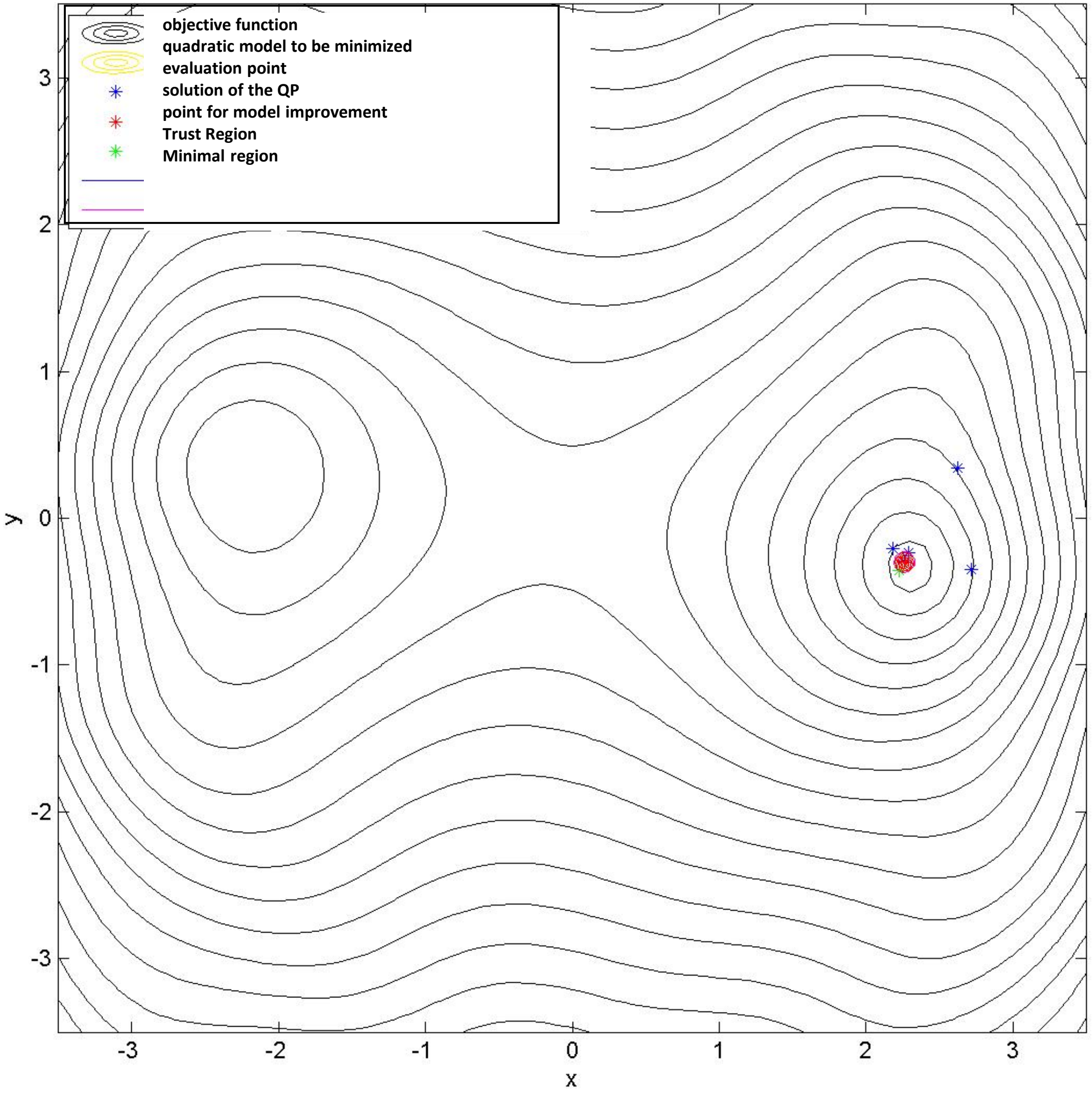
SURROGATE OPTIMIZATION METHODS

Local models

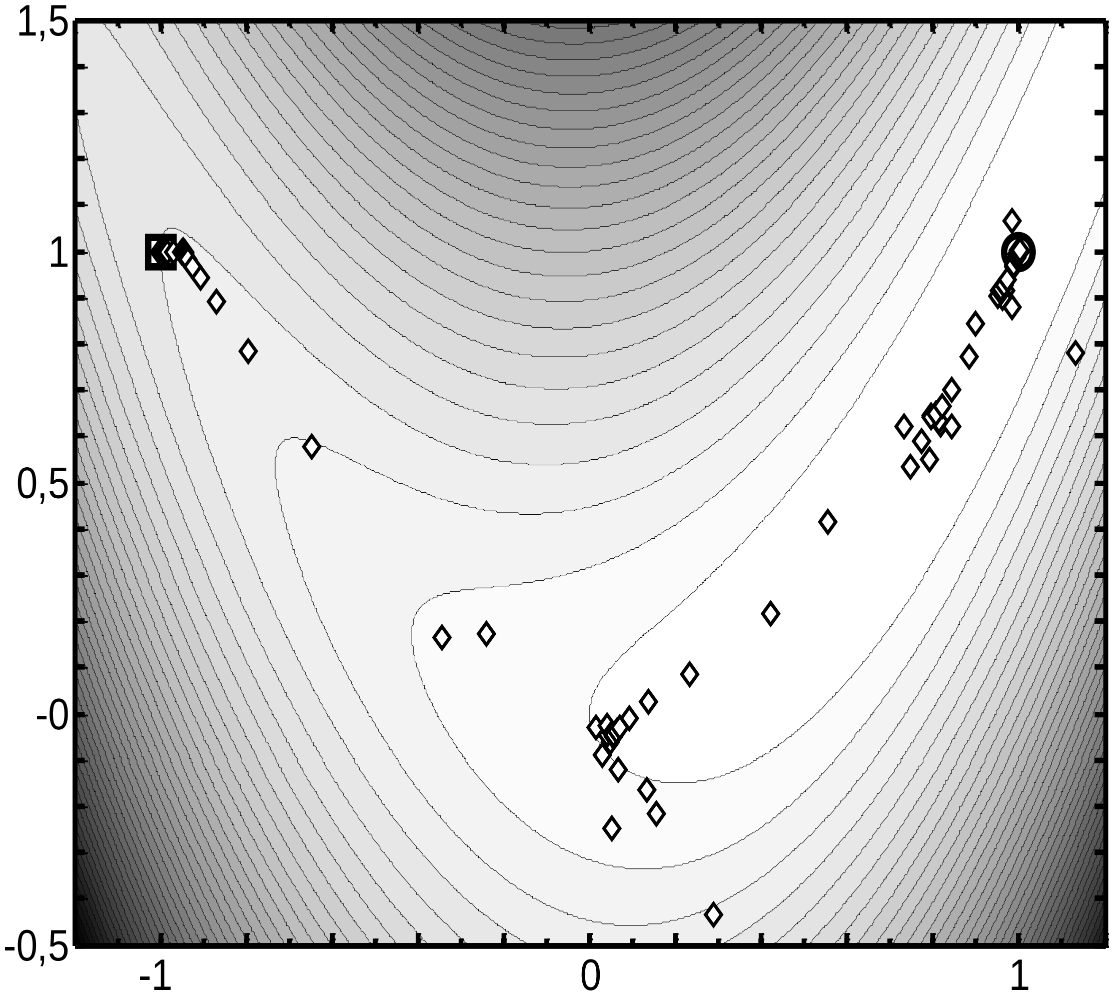


SURROGATE OPTIMIZATION METHODS

Local models



Local models (TR DFO)

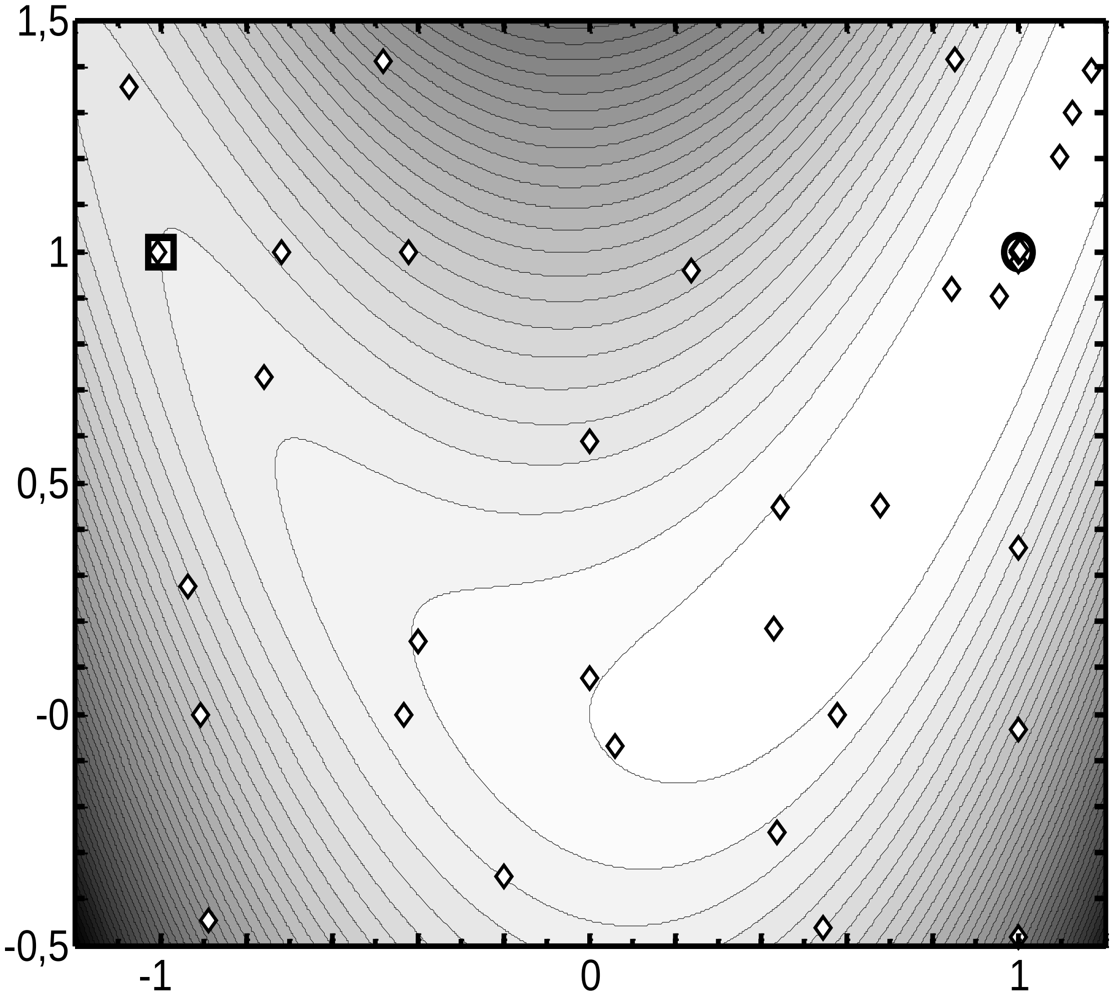


86 simulations

$(x^*, y^*) =$
$(1,001; 1,002)$

SURROGATE OPTIMIZATION METHODS

Global models (Gaussian Process)



29 simulations

$$(x^*, y^*) = (0,999; 0,999)$$

DFO METHODS

- *(Standard derivative-based methods with approximate gradients)*
- Direct Search methods
    - Nelder Mead Simplex
    - Pattern Search
- Surrogate optimization / model-based DFO methods
    - Local model of the objective function
    - Global model of the objective function
- Stochastic DFO methods
    - Evolutionary strategies
    - Simulated annealing

Evolution strategies

- Global optimization
- Very few assumptions on function regularity
- Main principle
  1. **Random generation** of initial population
  2. **Evaluation** of each individual of current generation
  3. **Reproduction:** selection of the best individuals
  4. **Diversification:** cross-over and mutation
  5. **Replacement :** survival of the best individuals
  6. **Repeat step 2** until satisfying solution is obtained

Evolution strategies

- An example of a **cross-over operator** for continuous variables
  Pair of individuals (x, y) selected randomly

$$(x, y) \rightarrow \alpha x + (1 - \alpha) y, \qquad \alpha \sim U([0; 1])$$

- An example of a **mutation operator** for continuous variables
  addition of a Gaussian noise

$$x_i \;\coloneqq x_i + u_i, \qquad u_i \sim N\left(0, \sigma_i^2\right), i = 1,2, \dots, n$$

σ_i is a critical parameter to tune

Evolution strategies

- 1/5-rule (Rechenberg)
  - If $\tau < 0.2$, then σ increases
  - If $\tau > 0.2$, then σ decreases
    (where τ = % of successful mutations over T generations)
  - ➢ Can adapt one general step-size
  - ➢ But no individual step size

- Mutative step-size control
  Strategy parameters (step-sizes) treated similarly to optimized parameters
  - ➢ Facilitates adaptation of individual step-sizes

Evolution strategies

- Mutation operator $x_{1\leq l\leq \lambda}^{(g+1)} \sim \mathcal{N}(m^{(g)}, [\sigma^{(g)}]^2 C^{(g)})$

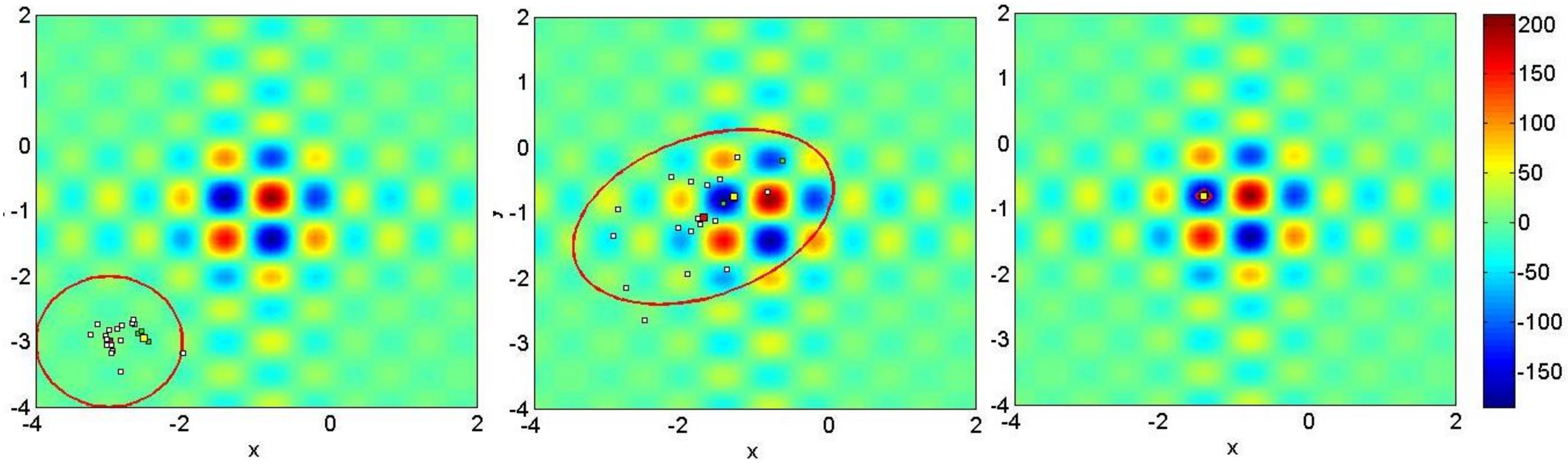
  Covariance matrix ~ inverse of a “global” Hessian matrix

- Select the μ best individuals to compute the mean
  - update of
    - the covariance matrix $C^{(g)}$
    - the global standard deviation $\sigma^{(g)}$

Evolution strategies vs. genetic algorithms

- Main difference : in evolutionary strategies, only the best individuals are allowed to reproduce (elitist selection)
- The parents can be included in the next generation
- Similar operators: mutation, cross-over

Evolution strategies: CMAES method (Hansen)



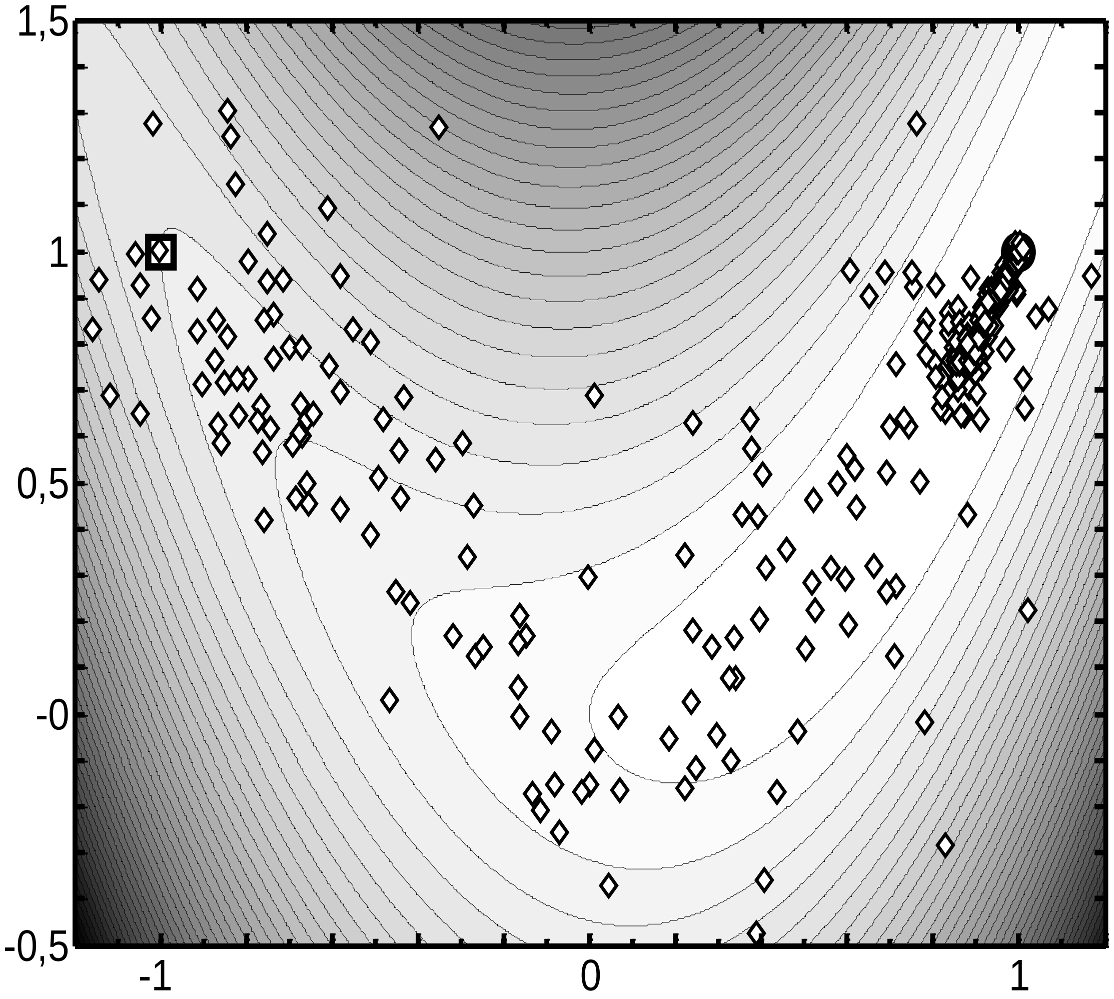
Generation 1

Generation 20

Generation 30

SURROGATE OPTIMIZATION METHODS

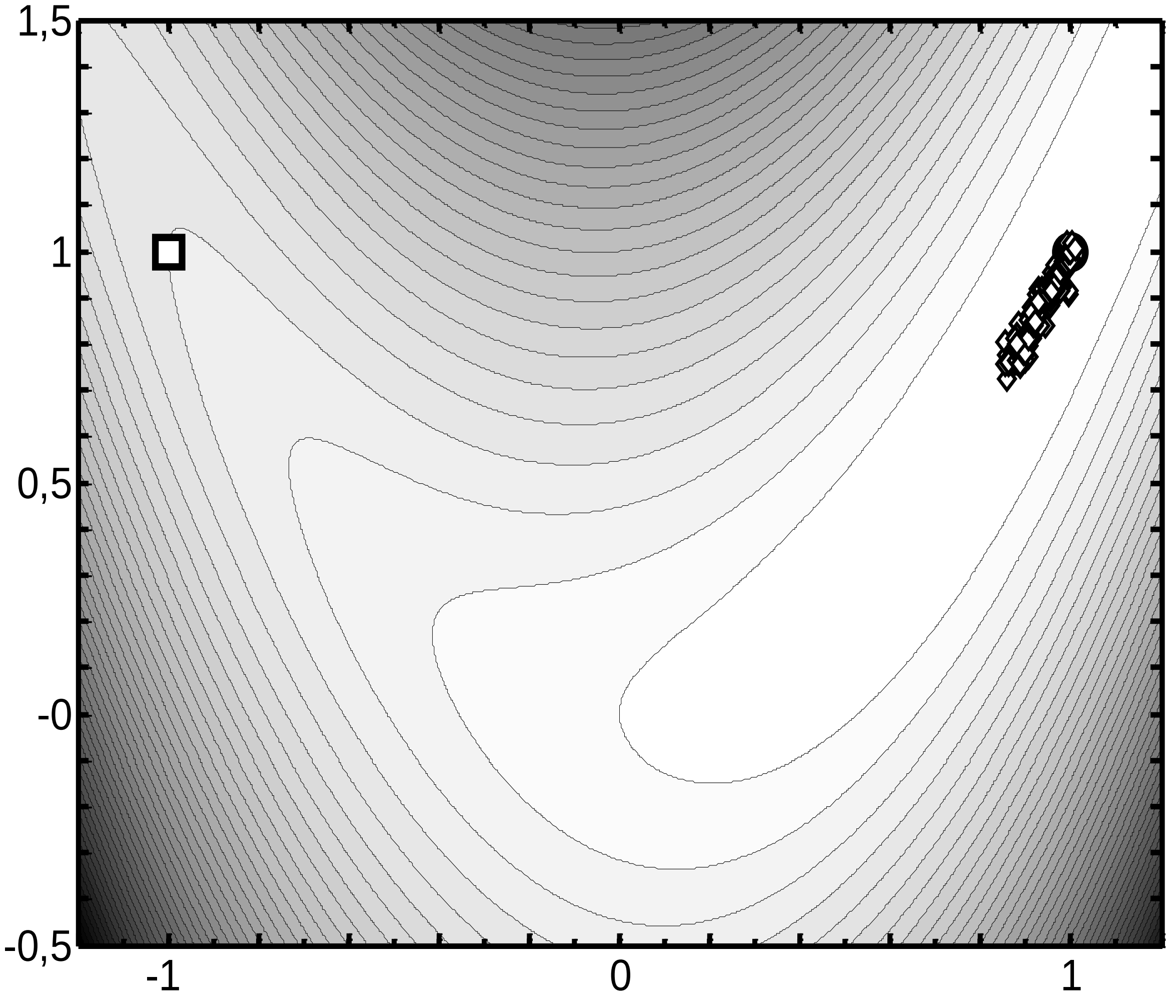
Evolution strategy; CMAES



370 simulations

$(x^*, y^*) =$
$(1,008; 1,007)$

Evolution strategy; CMAES



170 last simulations

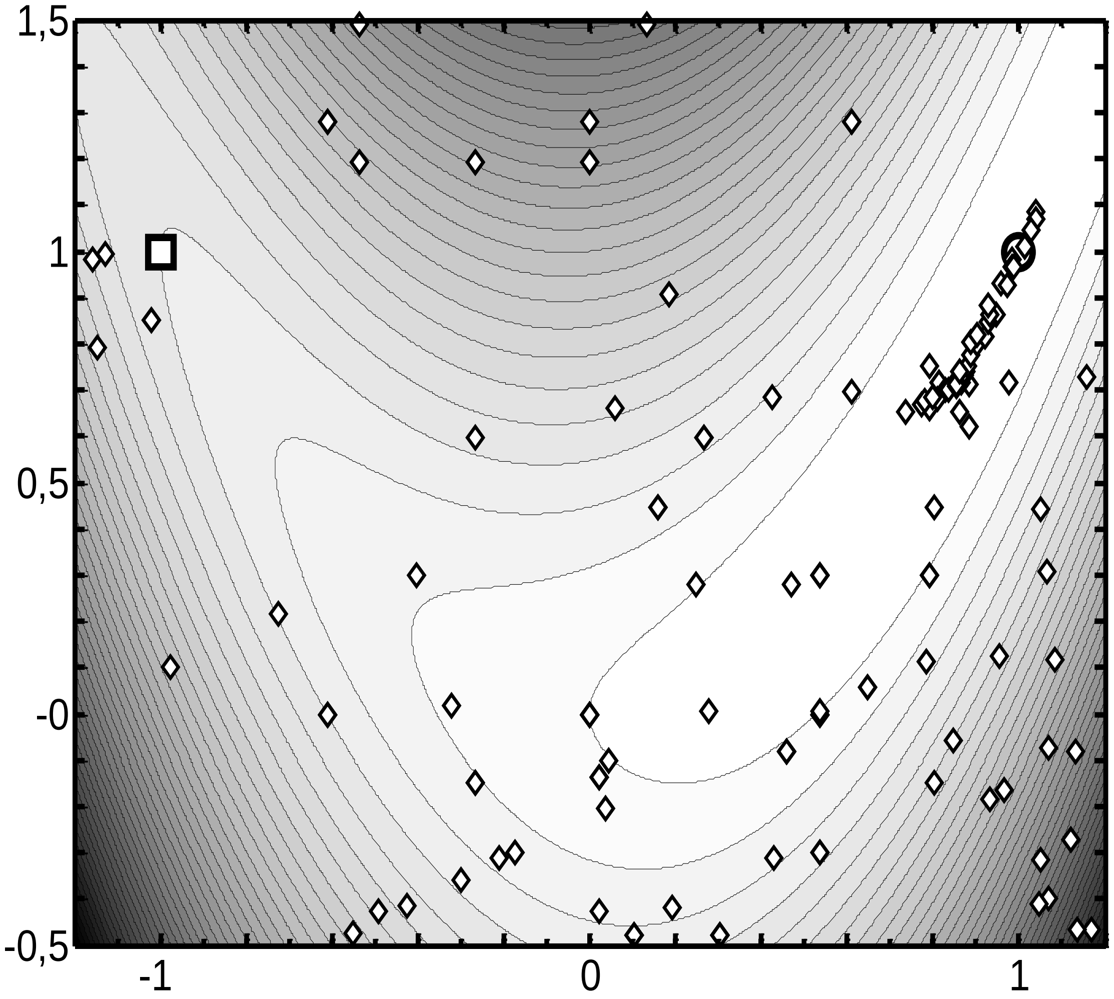
Simulated annealing

- **Principle:** emulate the physical system of the cooling of a solid so that the frozen state is frozen for a minimum energy configuration
- At a given iteration, a new solution (state) x_{k+1} is determined from the previous solution (state) x_k
  - **randomly choose a neighbour of $\boldsymbol{x_k}$: $\boldsymbol{y_k}$**
  - then, **the new state** is

$$x_{k+1} = \begin{cases} y_k & \text{if } f(y_k) \leq f(x_k) \\ y_k & \text{if } f(y_k) > f(x_k) \text{ with the probability } e^{\frac{f(x_k)-f(y_k)}{t_k}} \\ x_k & \qquad \text{otherwise} \end{cases}$$

  - **$\boldsymbol{t_k}$ (temperature of the system) is a decreasing sequence** in order to decrease the probability to accept a bad solution (increasing f) at last iterations

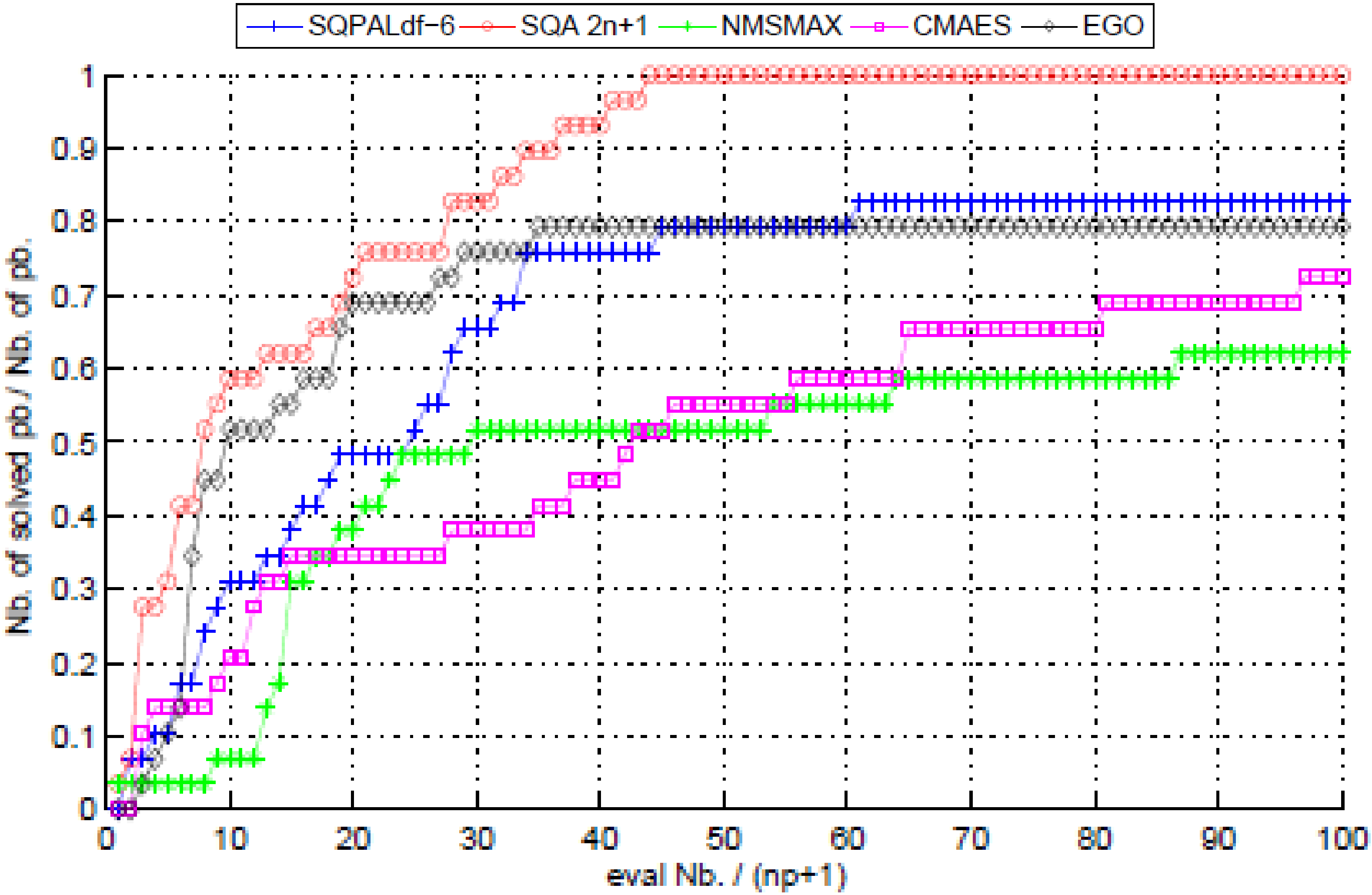
Simulated annealing: SIMPSA



280 last simulations

$(x^*, y^*) =$
$(0.989; 0.968)$

Benchmark Moré & Wild



SQPAL = SQP BFGS – FD
SQA = local surrogate optim
EGO = global surrogate optim (kriging)
NMSMAX = Nelder-Mead simplex
CMAES = Evolutionary Strategy

53 problems
$2 \leq n \leq 12$

Summary (I)

- Classical methods with approximate gradients
  - *popular (do not change optimizer)*
  - *but not adapted for large scale problem*
  - *step size tuning is cumbersome*
- Direct Search methods
  - *revival of these methods with parallelization*
  - *hybrid implementation (coupled with surrogate models)*
  - *not adapted for large scale problems*

Summary (II)

- Surrogate optimization
  - Local interpolation model with TR
    - *can handle constraints (coupled with SQP)*
    - *good performances in terms of number of function evaluations*
  - Global models
    - *global methods*
    - *not adapted for large scale problems:*
      needs a lot of evaluation to obtain a good accuracy

DFO METHODS

Summary (III)

- Evolutionary **strategies** / Simulated annealing
  - *no assumption on function regularities*
  - *discrete optimization is possible*
  - *global methods*
  - *but expensive in terms of function evaluations*
  - *difficulties to handle constraints*

Global versus Local

- Multi-start optimization
  - run a local method from several initial points
  - *well adapted for functions with a small number of local minima*
  - *handles constraints*

- Global surrogate optimization (kriging, RBF, NN) / Evolutionary algorithms / Simulated annealing
  - *not adapted for large scale problems:*
    needs a lot of evaluations to obtain a good accuracy
  - *difficulties for handling constraints*

REFERENCES

- Audet, C. and Hare W., Derivative-Free and Blackbox Optimization, Springer Series in Operations Research and Financial Engineering (2017)
- Conn, A.R., Scheinberg, K., Vicente, L.N., Introduction to derivative-free optimization. SIAM, Philadelphia (2009)
- Rios, L.M. & Sahinidis, N.V., Derivative-free optimization: a review of algorithms and comparison of software implementations, J Glob Optim (2013), Vol. 56, Issue 3, pp 1247–1293
- Cartis, C., Lectures on global and derivative-free optimization, University of Oxford (2018)

DERIVATIVE FREE OPTIMIZATION AND APPLICATIONS

- Course 1: main DFO methods
- Course 2: various applications of DFO
- Course 3: some challenges in DFO