# Optimisation

# Heuristic methods

# Optimisation methods : iterative

- Direct search
  - Exploring the search space from a location to find the lowest point
    - Easy to use and robust
    - High computing time
- Indirect search
  - Mathematical condition of the optimality
    - More efficient
    - More complex (derivatives)
- Heuristic methods
  - Explore the search space based on a property of the system
    - e.g. genetic algorithms
    - Global optimum
    - Very high computing time
    - Highly dependent on the quality of the model

# Black box method

Optimisation : min $OBJ(X^*_{decision})$

Subject to $G_{inequality}(X^*_{decision}) \geq 0$

$X^*_{decision}$

$BOJ(X^*_{decision})$

$G(X^*_{decision})$ inequality

Status

Model : Solve

$$\left.\begin{array}{c} F(X_{dependent}, X_{specification}, X_{decision})=0 \\ S(X_{dependent}, X_{specification}, X_{decision})=0 \\ X_{decision} - X^*_{decision} = 0 \end{array}\right\} \Rightarrow X(X^*_{decision})$$

then calculate $OBJ(X(X^*_{decision}))$

$G(X(X^*_{decision}))$

# Heuristic methods

- Applies only on black box strategy
- Exploring the search domain
  - systematically
  - based on some analogy
- Simulated annealing
  - based on the analogy with metallurgy
    - heating/cooling of metal to minimize the energy content
- Evolutionary algorithm
  - genetic algorithms
    - based on the analogy of the evolution
      - Best fitted individuals have a higher probability to survive and reproduce
      - Reproduction based on sharing gene info
- Particle swarm
  - initial speed + communication between agents
- Ants colony

# Heuristic methods

- Simulated annealing

$$
\begin{array}{rl}
\text{set} & k = 0 \\
\text{Choose} & T = T_0, \sigma \\
\text{Choose} & x_{ref} = x_0 \\
\text{Repeat if} & k < k_{max} \\
 & \text{Choose } x \in V(x_{ref}) \\
 & \delta = F(x) - F(x_{ref}) \\
if & \delta < 0 \\
then & x_{ref} = x \\
else & \text{select randomly } p \in \{0..1\} \\
 & if(p < exp(\frac{-\delta}{T}))then \quad x_{ref} = x \\
 & T = \frac{T}{1+\frac{T ln(1+\delta)}{3\sigma}} \\
 & k = k + 1 \\
\text{Repeat} &
\end{array}
$$

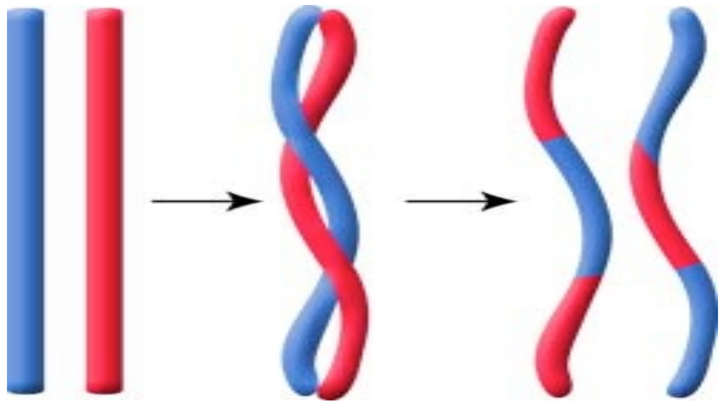
<- x in the vicinity of $x_{ref}$

Selecting the new point in the vicinity of the best point is the trick and depends on the problem to be solved, can be permutations or other vectorial moves near the optimal point

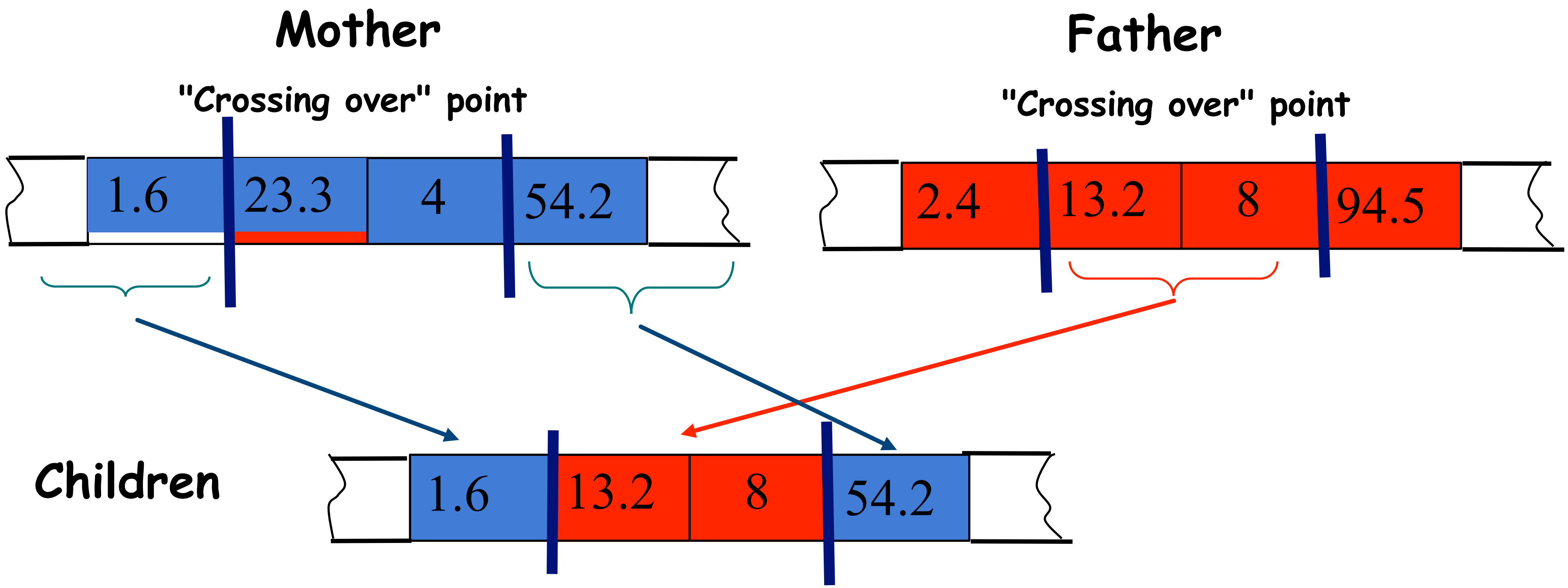
# Evolutionary algorithms

- **Characteristics**
  - Population (X)
  - Objective Function(s) : Performances Y=F(X)
  - No direction ( No derivatives - No "iteration")
  - Heavy duty : Computing time !
  - Problem definition is free
  - Random nature : explore the search space
  - Inequality constraints ?
- **Principle**
  - Initialization (random population generation (e.g. 100 sets))
  - Reproduction => select parents & reproduce
  - New individual
    - Cross-over (random)
    - Mutation
  - Update population (maintain population)
    - eliminate the worst individuals
    - re-group by types to preserve diversity

# reproduction by "Crossing over"



Random selection of parents in the population
Random selection of the genes to share

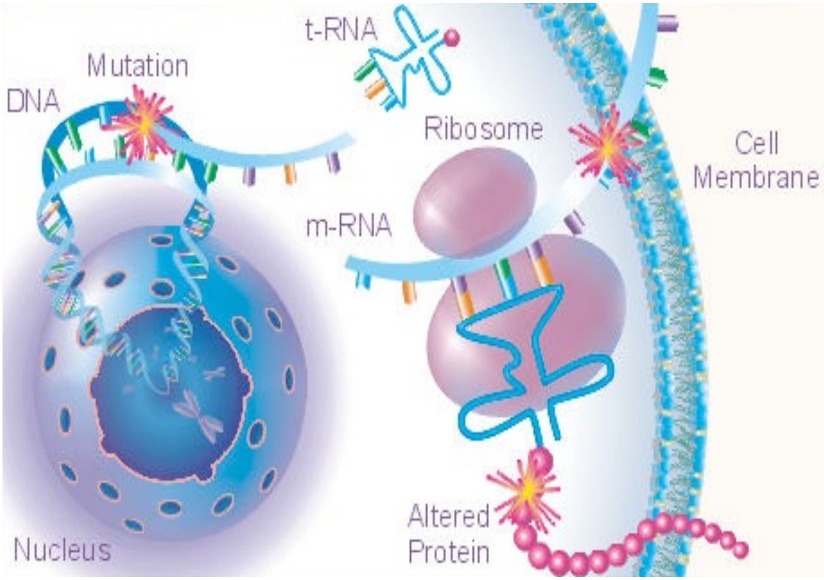


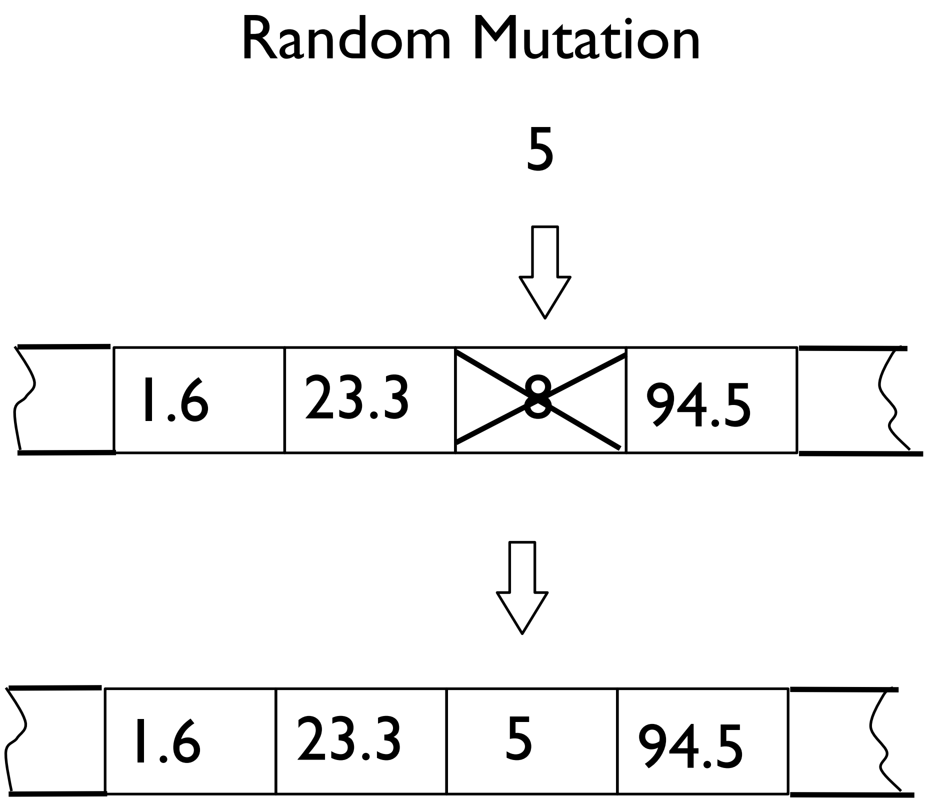
# Cross over

- **Cross-over can use interpolation techniques**
  - e.g. quadratic approximation based on a subset of the population
  - select randomly and/or take the bests
- **Preserve the random nature !**
  - e.g. random relaxation

# Mutation

Mutation allows to ensure that the system will not be trapped in a local optimum and that the whole space will be observed



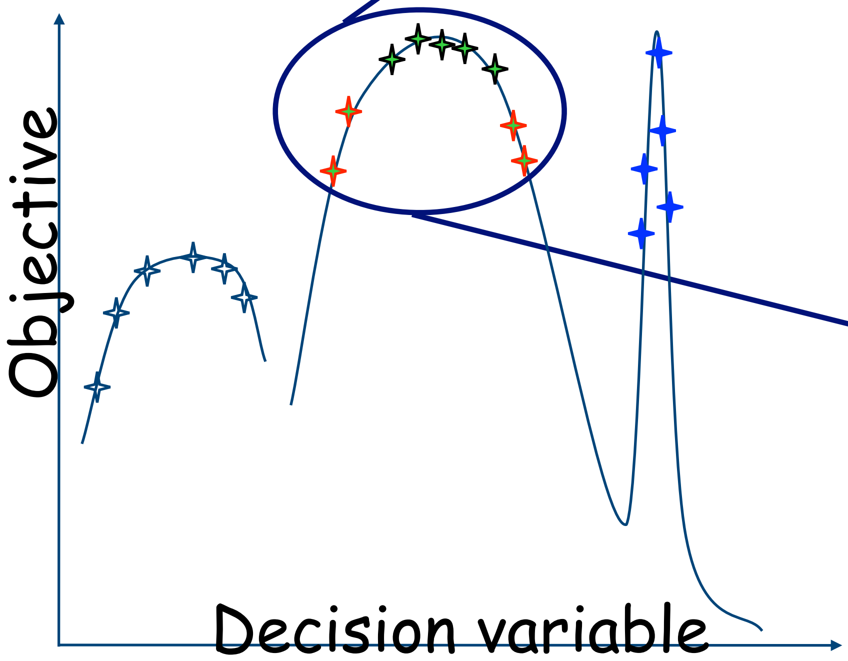



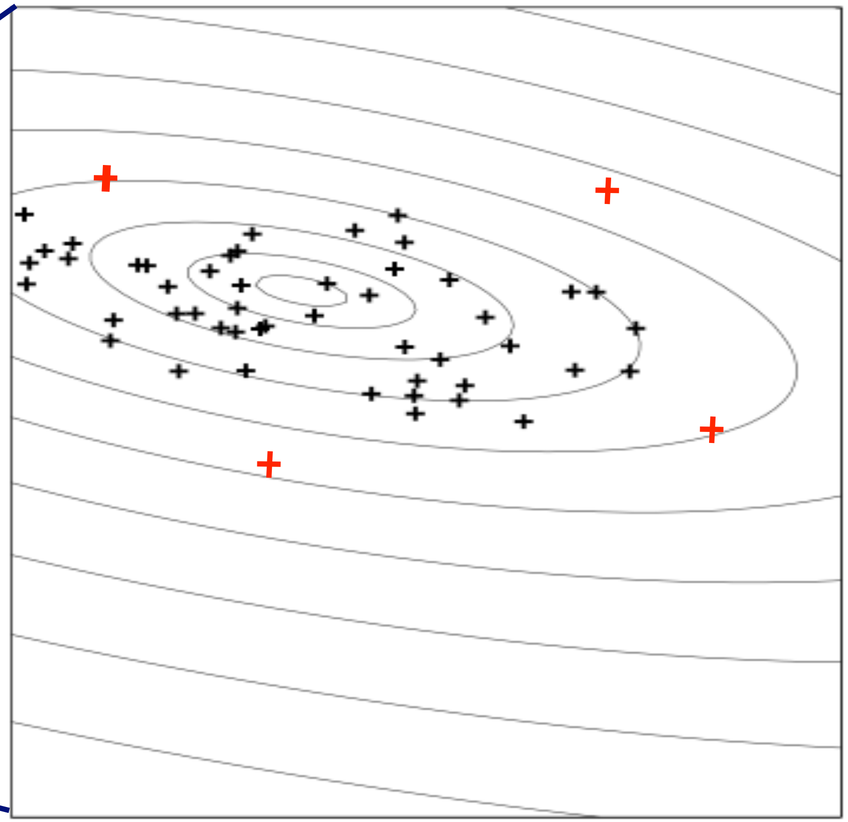

before mutation

after mutation

# Elimination

## Elitism : preserve the best candidates



Elitism is defining the population in which the good candidates are selected : this allows multi-objective optimisation and the search for local and global optimum.

# Evolutionary algorithms : conditions

- Black box approach
  - Equality constraints are solved explicitly
  - Inequality constraints transformed into decision variables bounds
- Fast F(X) calculation
  - Search space exploration => huge number of evaluation
- Robust F(X) calculation
  - search space response
- No efficient mathematical programming methods
  - Non differentiable problems
  - MINLP
- Limited number of degrees of freedom

# Evolutionary algorithm : advantages

- **Global optimisation**
  - Exploration of the search space
- **Black Box**
  - Accepts different type of objective function
    - incl. observations
- **Non differentiable problems**
  - The objective function can have jumps or steps
- **Easy to parallelise**
- **Freedom in the choice of decision variables**
  - x1*x2*x3 is not a problem
- **Multi-objective problem**
  - Efficient use of the computing time
  - Dominancy criteria

# Evolutionary algorithm : drawbacks

- **Speed of resolution of F(X)**
  - Requires a large number of F(X) evaluation
  - Use of surrogate models
- **Number of decision variables**
  - Convergence properties is a combinatorial function of the number of variables
- **Limited Feasible domain**
  - Probability of finding feasible F(X) is low
  - Choice of the decision variables
- **Constraints handling**
  - Equality or inequality

# Evolutionary algorithm

- Handling inequality constraints

$$\min_{X} OBJ(X)$$
$$st.$$
$$F(X) = 0$$
$$G(X) \leq 0$$

$$\min_{X^d} OBJ(X^d, Y(X^d)) + P(X^d)$$
$$st.$$
$$Y(X^d) = F(X^d)$$
$$P(X^d) = \sum (max(G(X^d, Y(X^d)), 0))^2$$
$$X^d_{max} \leq X^d \leq X^d_{max}$$

# Evolutionary algorithm

- Choosing the appropriate decision variables

$$\min_{x_1,x_2} f(x_1,x_2)$$

$$st.$$

$$x_1 \leq x_2$$

$$x_1^{min} \leq x_1 \leq x_1^{max}$$

$$x_2^{min} \leq x_2 \leq x_2^{max}$$

**Becomes**

$$\min_{x_1^*,x_2} f(x_1 = x_1^{min} + (min(x_2, x_1^{max}) - (min(x_2, x_1^{max}) - x_1^{min})) \cdot x_1^*, x_2)$$

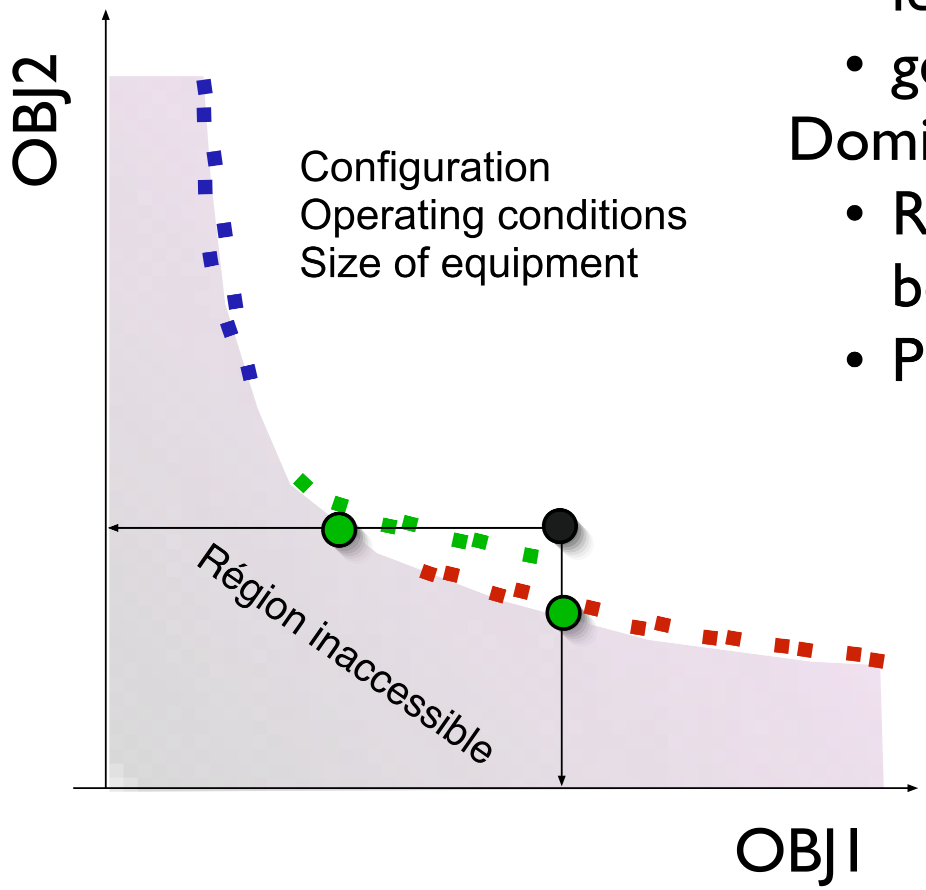
$$st.$$

$$0 \leq x_1^* \leq 1$$

$$x_2^{min} \leq x_2 \leq x_2^{max}$$

**Works well for Evolutionary algorithm**
**do not use for mathematical programming**

# Multi-objective optimisation




Clustering techniques (big data)

- identify decision variable sub-spaces
- generate multiple Pareto curves

Dominancy

- Remains in the population if at least better than others for 1 objective
- Preserve sub-optimal population

The goal is indeed to take decisions : being informed about the collection of good solutions allows to have a better knowledge of what is building a solution and for which reason the filan solution will be selected

# Evolutionary solving strategies

- Hybrid methods
  - Use Evolutionary algorithm to find initial point for mathematical programming
  - Global optimization (find min of min )
  - Limited number of NLP
  - Do it in 2 directions
    - min obj1
    - min obj2