

NATIONAL TECHNICAL UNIVERSITY OF ATHENS (NTUA)
SCHOOL OF MECHANICAL ENGINEERING
PARALLEL CFD & OPTIMIZATION UNIT (PCOpt/NTUA)



*Evolutionary Algorithms & Low-Cost Evolutionary
Optimization, with Applications*

Dr. Varvara Asouti, Senior Researcher NTUA

Athens, October 2024

Scope

Introduction to evolutionary algorithms and how to make them attractive for real-world applications


We will talk about:

- **Population-based stochastic-based optimization methods**
 - **“Generalized” Evolutionary Algorithm (EA)**
 - **Encoding, evolution operators**
- **Cost reduction techniques in EAs**
- **Industrial applications**

Constrained Optimization: Problem Statement

$$\min \vec{F}(\vec{b}) = \min\{f_1(\vec{b}), \dots, f_{M_o}(\vec{b})\} \quad \vec{b} \in \mathbb{R}^N$$

subject to $c_j(\vec{b}) \leq 0, j = 1, M_c$


N design (optimization) variables
M_o objectives
M_c constraints

In practice: To compute f_i or c_j , calls to the evaluation software (**eval.exe**) is needed. The computational cost of such an evaluation is the time-unit used to measure the cost of the optimization run as a whole.

$$\text{Optimization Cost} = \text{\# calls to the evaluation s/w} * \text{Cost per evaluation s/w (=time-unit)}$$

Part 1:
Population-based Stochastic-based Optimization
Methods for Beginners –
A Generalized Evolutionary Algorithm

Terminology - Minimization or maximization, etc.

Objective Function

Fitness Function (if maximization)

Cost Function (if minimization)

$M_o = 1$ → Single-Objective Optimization (SOO)

$M_o > 1$ → Multi-Objective Optimization (MOO)

max C_L
min C_D

min $-C_L$
min C_D

min $1/C_L$
min C_D

max C_L
max $-C_D$

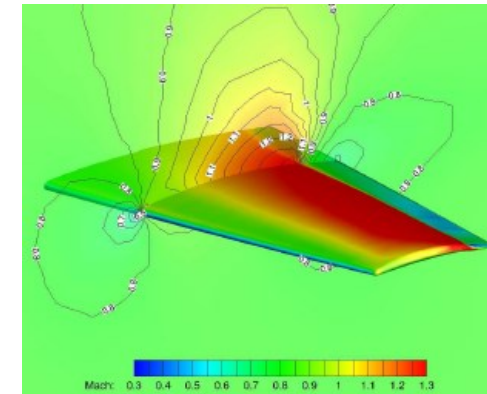
min $C_D + w/C_L$

min $C_D + 1/C_L$

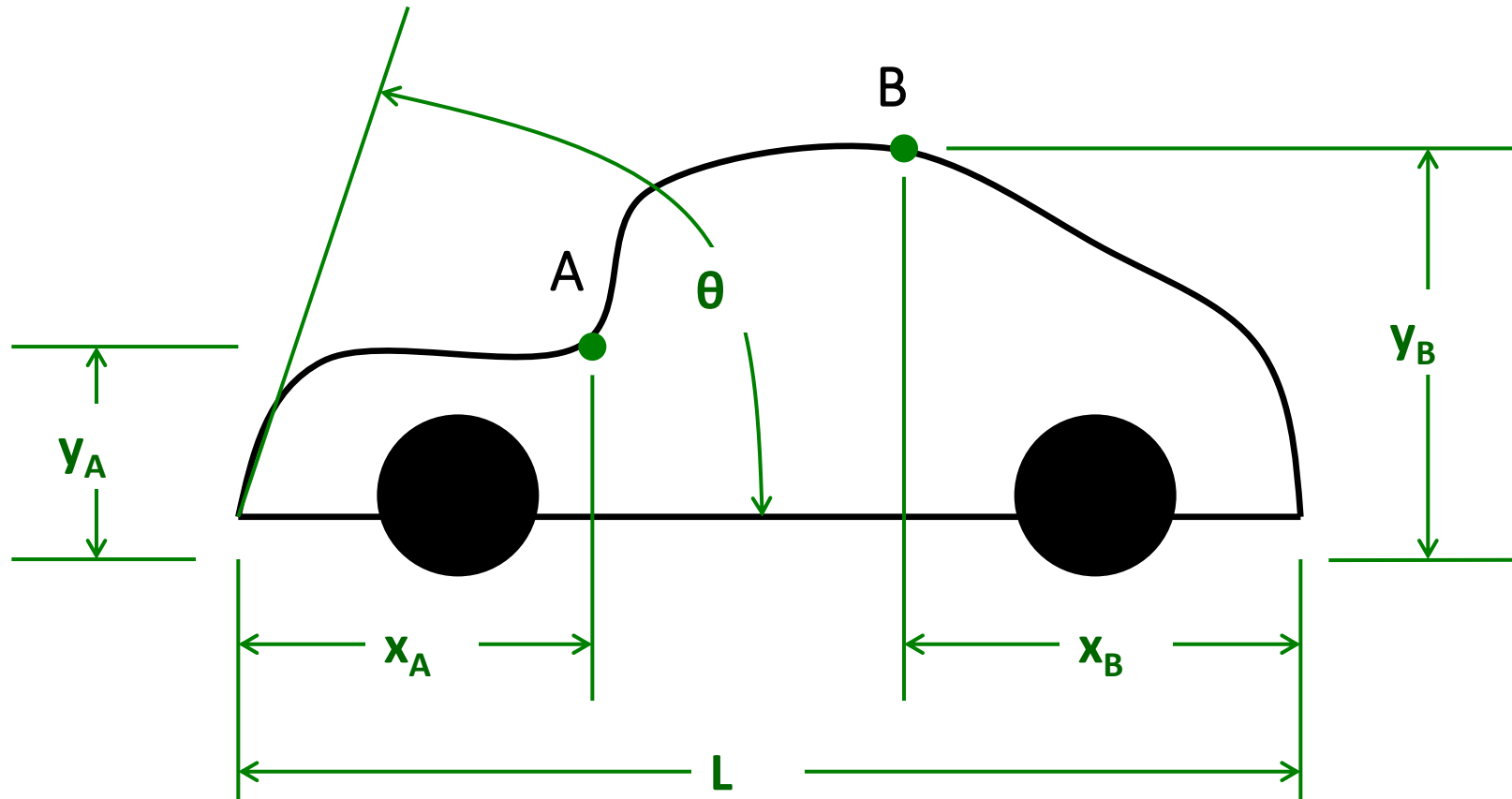
min $C_D + 10/C_L$

min C_D
constrained by: $C_L = 1.2$

min C_D
constrained by: $C_L > 1.2$



Very simple example of an EA with 6 parents & 6 offspring (SOO)

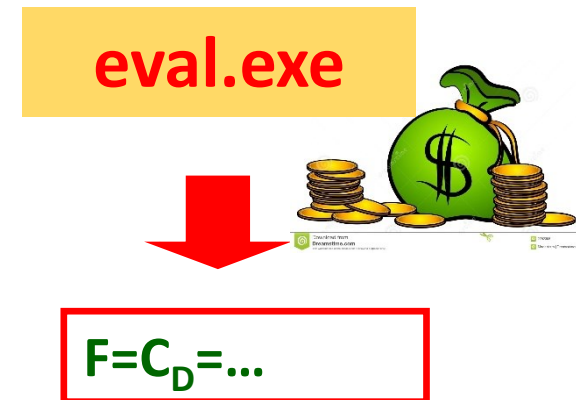
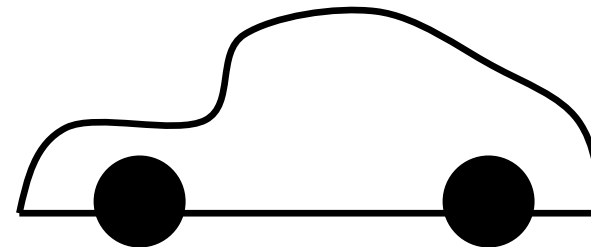
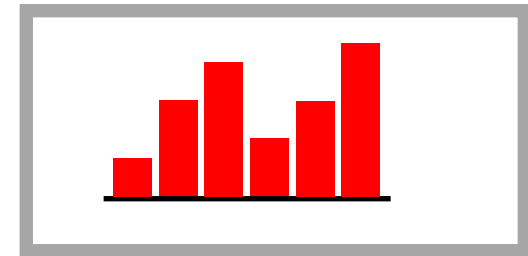
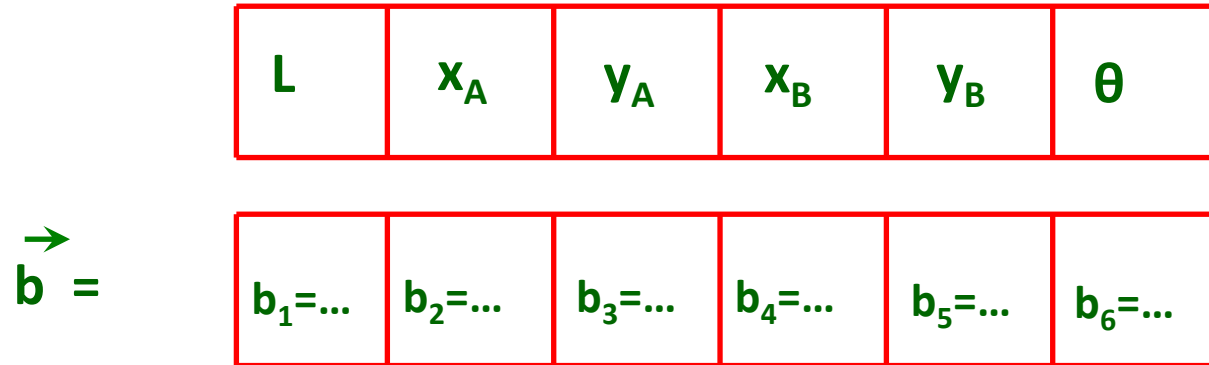


N=6 degrees of freedom (DOFs), design or optimization variables.

Objective (cost) function: min. (drag)

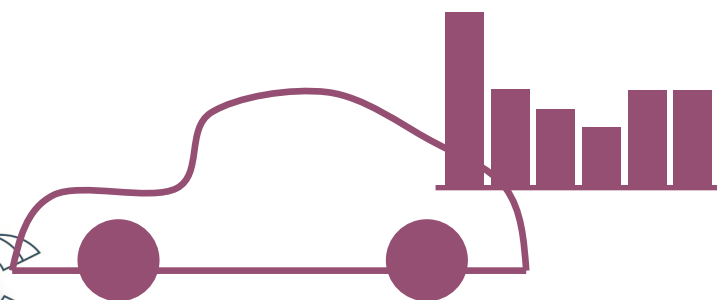
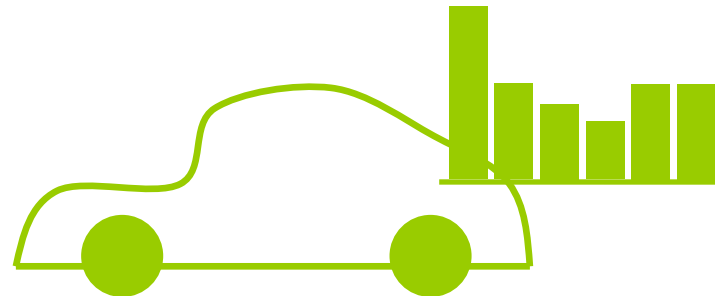
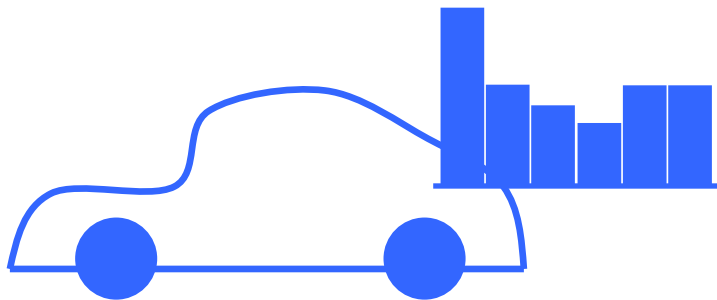
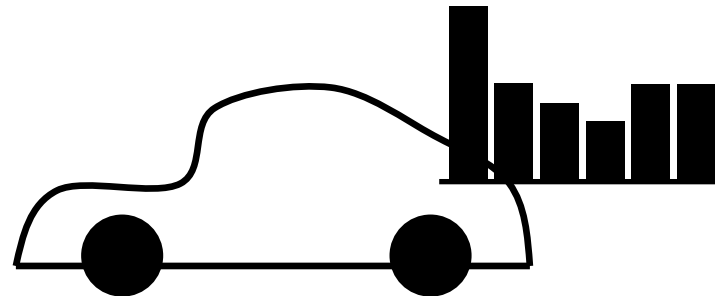
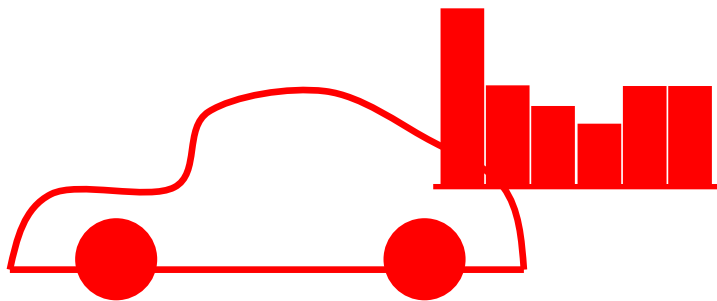
Evaluation s/w: a CFD solver

The evaluation S/W (eval.exe)



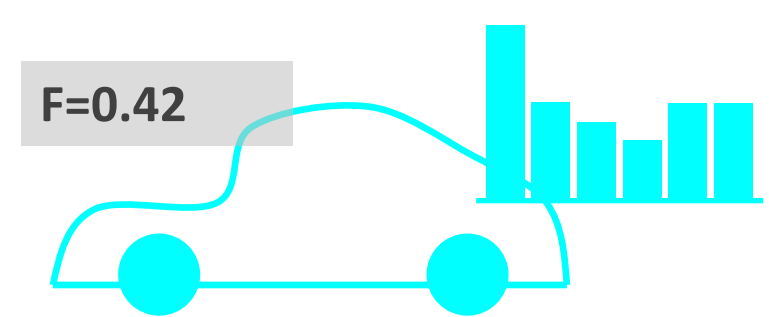
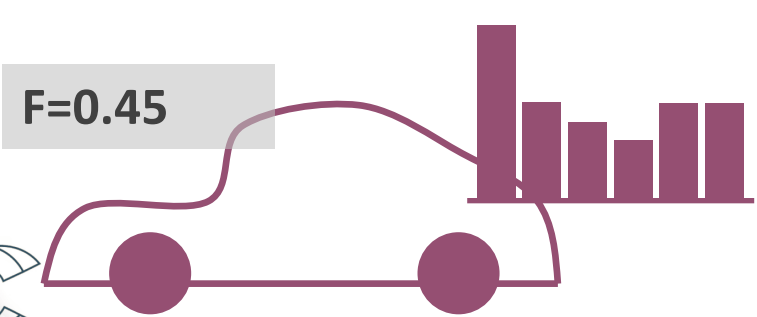
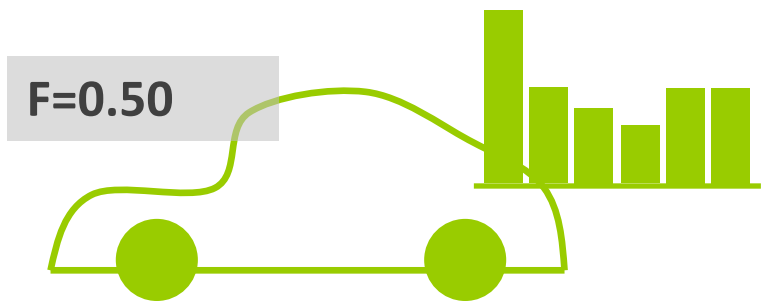
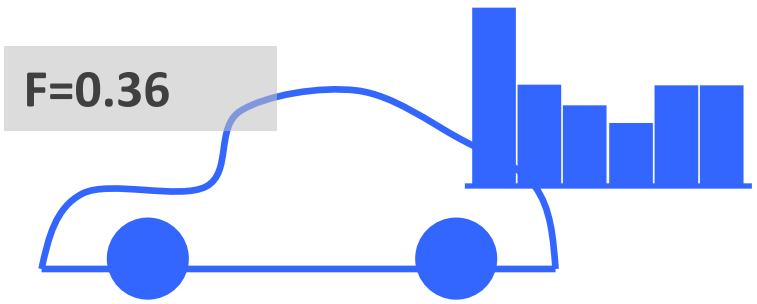
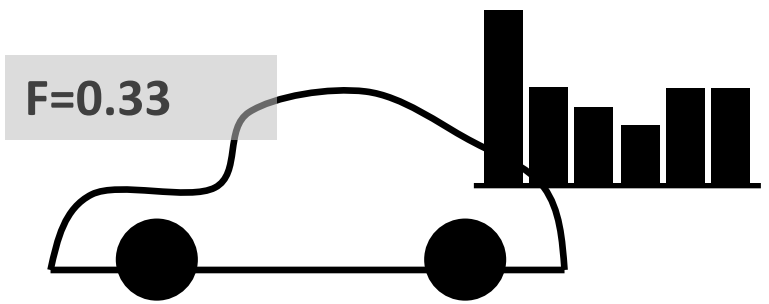
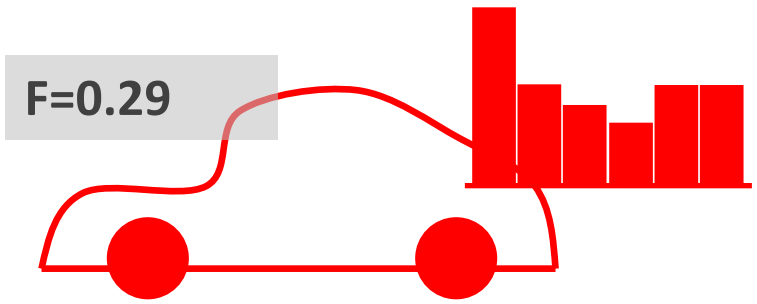
No access to the source code is needed!!!

Very simple example of a (μ, λ) EA with $\mu=6$ parents & $\lambda=6$ offspring (SOO)



Start by λ randomly selected offspring.
(All these b_n values have been selected at random between known the lower and upper bounds of each design variable)

Very simple example of a (μ, λ) EA with $\mu=6$ parents & $\lambda=6$ offspring (SOO)

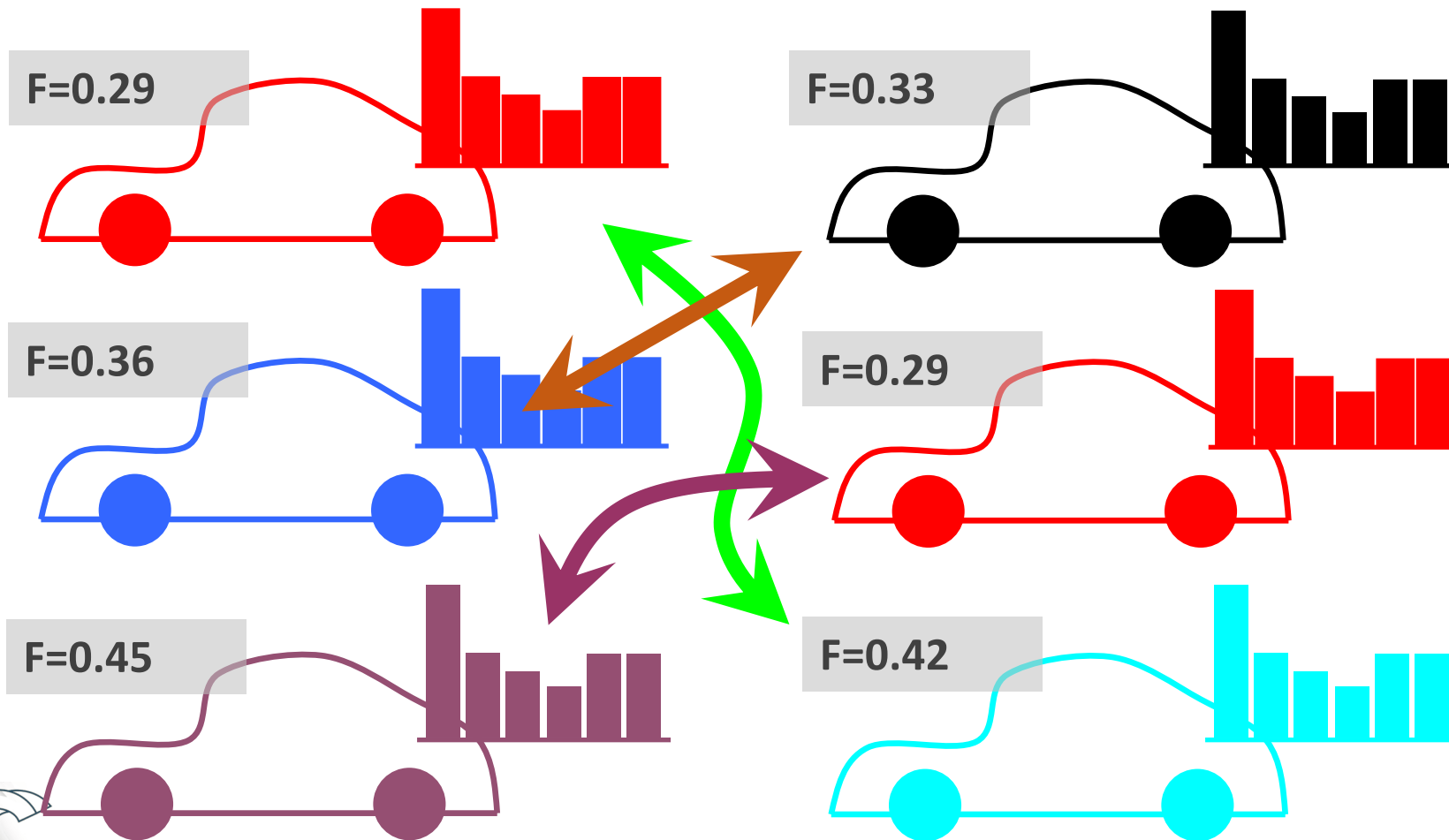


Evaluation of the current offspring population. Cost: $\lambda=6$ calls to eval.exe. $F=C_D$ (to be minimized)



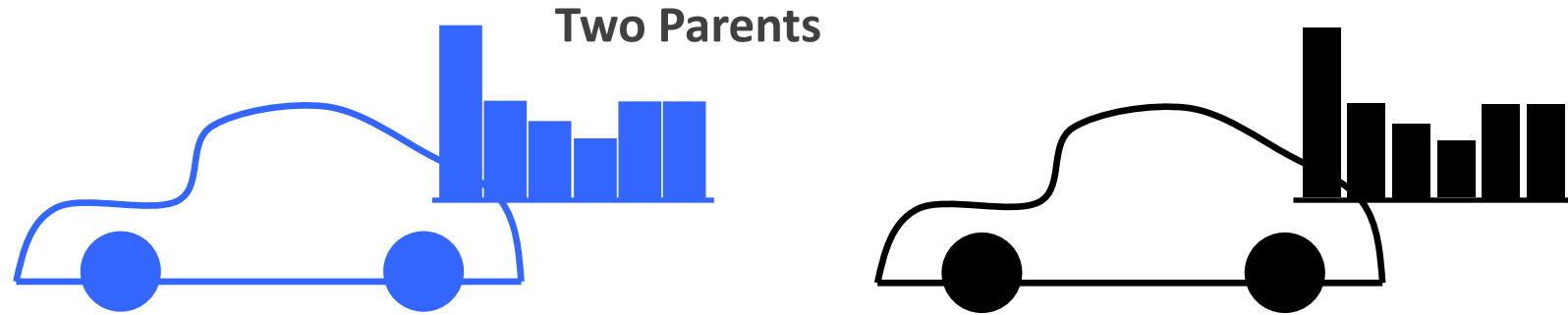
(Concurrent runs on a multi-processor system is possible)

Very simple example of a (μ, λ) EA with $\mu=6$ parents & $\lambda=6$ offspring (SOO)

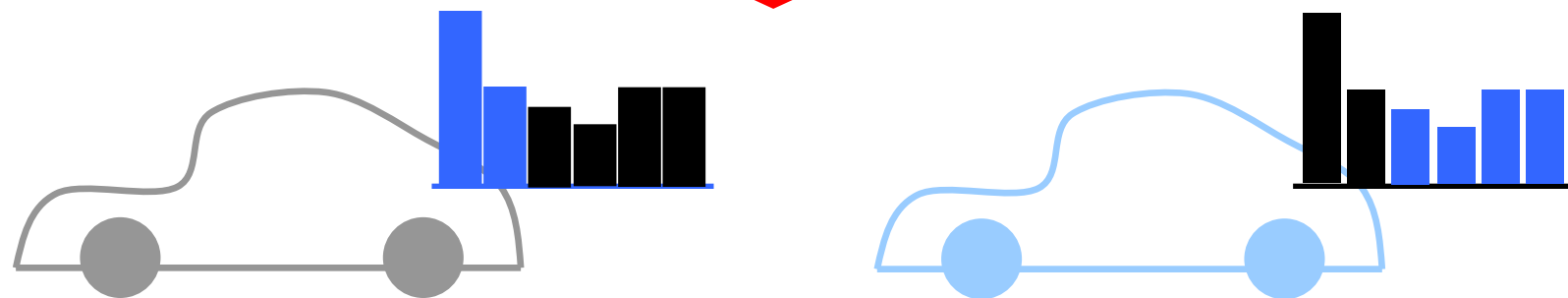
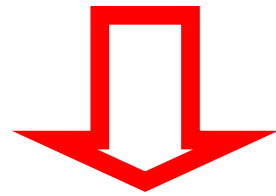


Parent selection (λ evaluated offspring $\rightarrow \mu$ parents).
 (This a absolutely simple and, thus, non-well performing, way of selecting parents)
 (It is just by chance that, here, $\mu=\lambda$)

Very simple example of a (μ, λ) EA with $\mu=6$ parents & $\lambda=6$ offspring (SOO)



Crossover (Recombination)



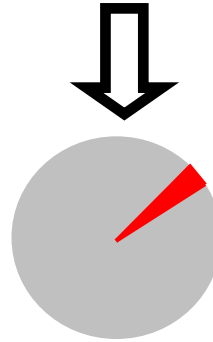
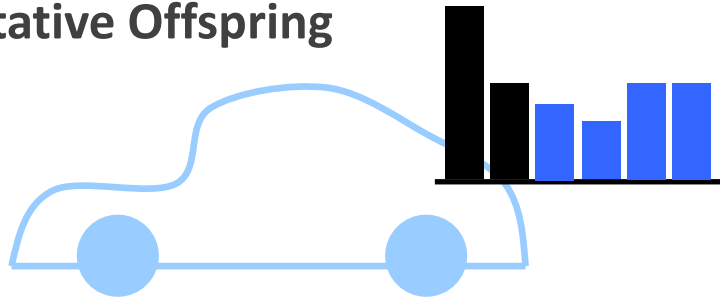
Application of evolution operators on the parent population.

(Based on a Random Number Generator or RNG)

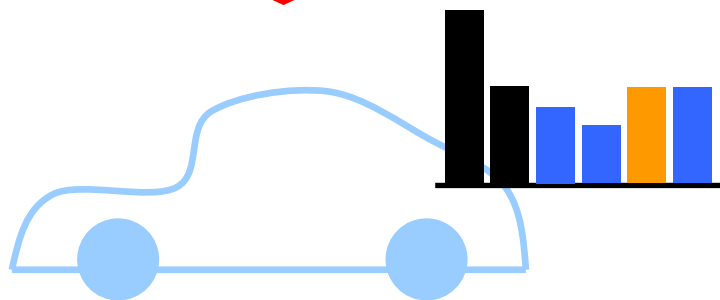
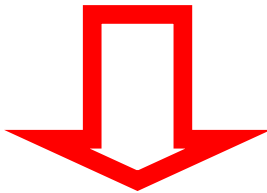
(Using 2 parents to create 2 offspring is not necessarily the best option)

Very simple example of a (μ, λ) EA with $\mu=6$ parents & $\lambda=6$ offspring (SOO)

Tentative Offspring



Mutation

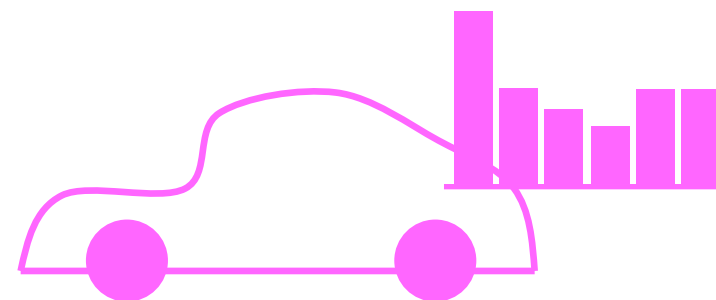
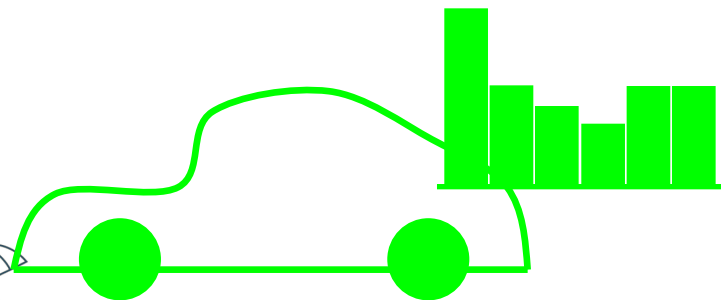
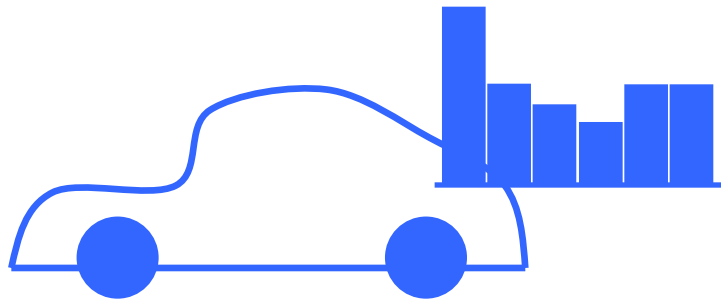
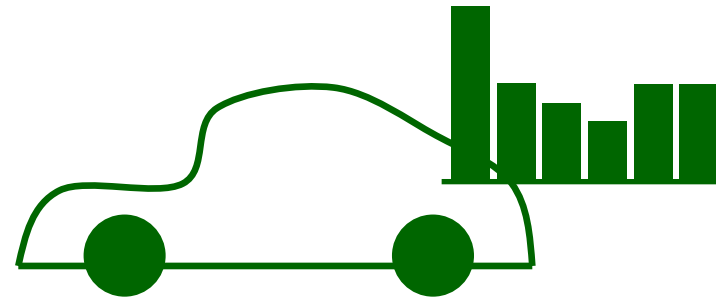
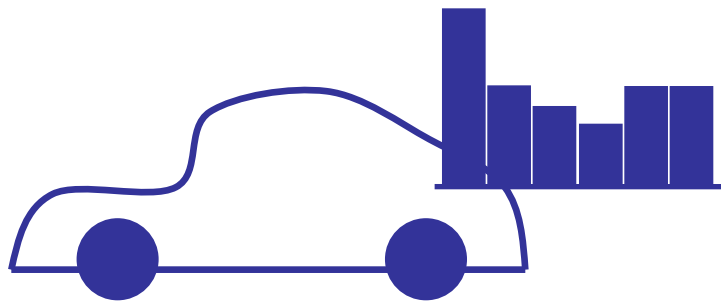


Offspring

Application of evolution operators on the parent population.

+ Elitism!

Very simple example of a (μ, λ) EA with $\mu=6$ parents & $\lambda=6$ offspring (SOO)



The new offspring population (λ individuals).

Converged or repeat the same steps?

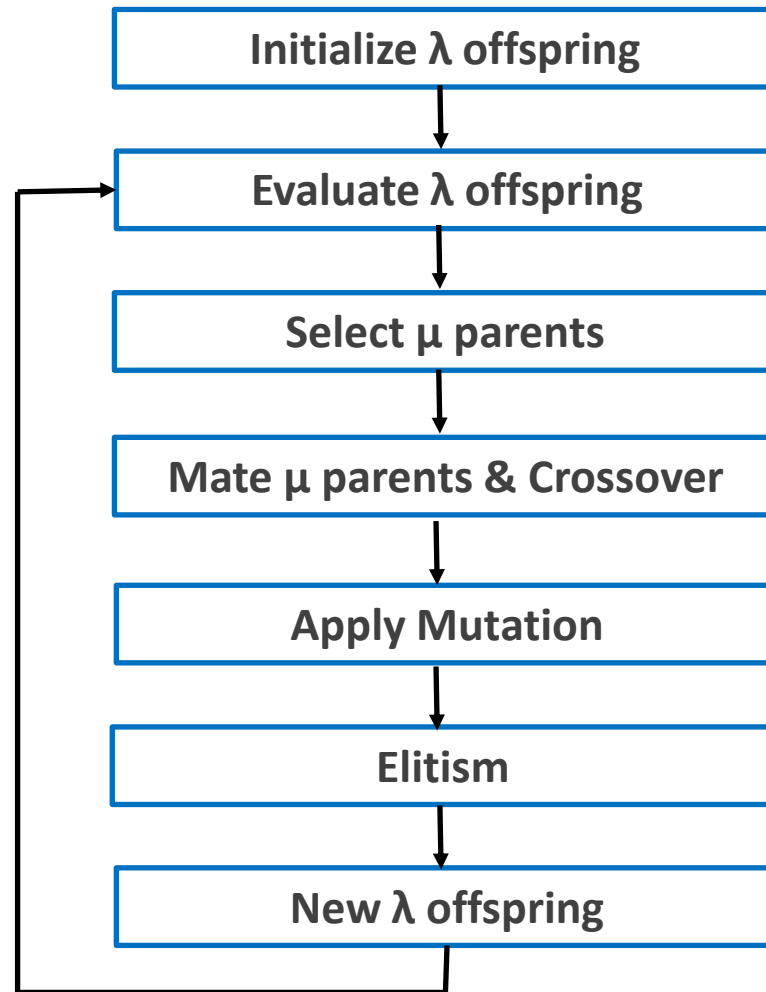
Other Stochastic Population-based Optimization Methods

- ◆ Genetic Algorithms (GA) & Evolution Strategies (ES)
- ◆ Particle Swarm Optimization (PSO)
- ◆ Ant-Colonies Optimization (ACO)
- ◆ Pity Beetle Algorithm
- ◆ Chicken Swarm Optimization (CSO)
- ◆ Harmony Search
- ◆ ...

This presentation is based on a **“Generalized” Evolutionary Algorithm (EA)**, among other bridging the gap between GA and ES.

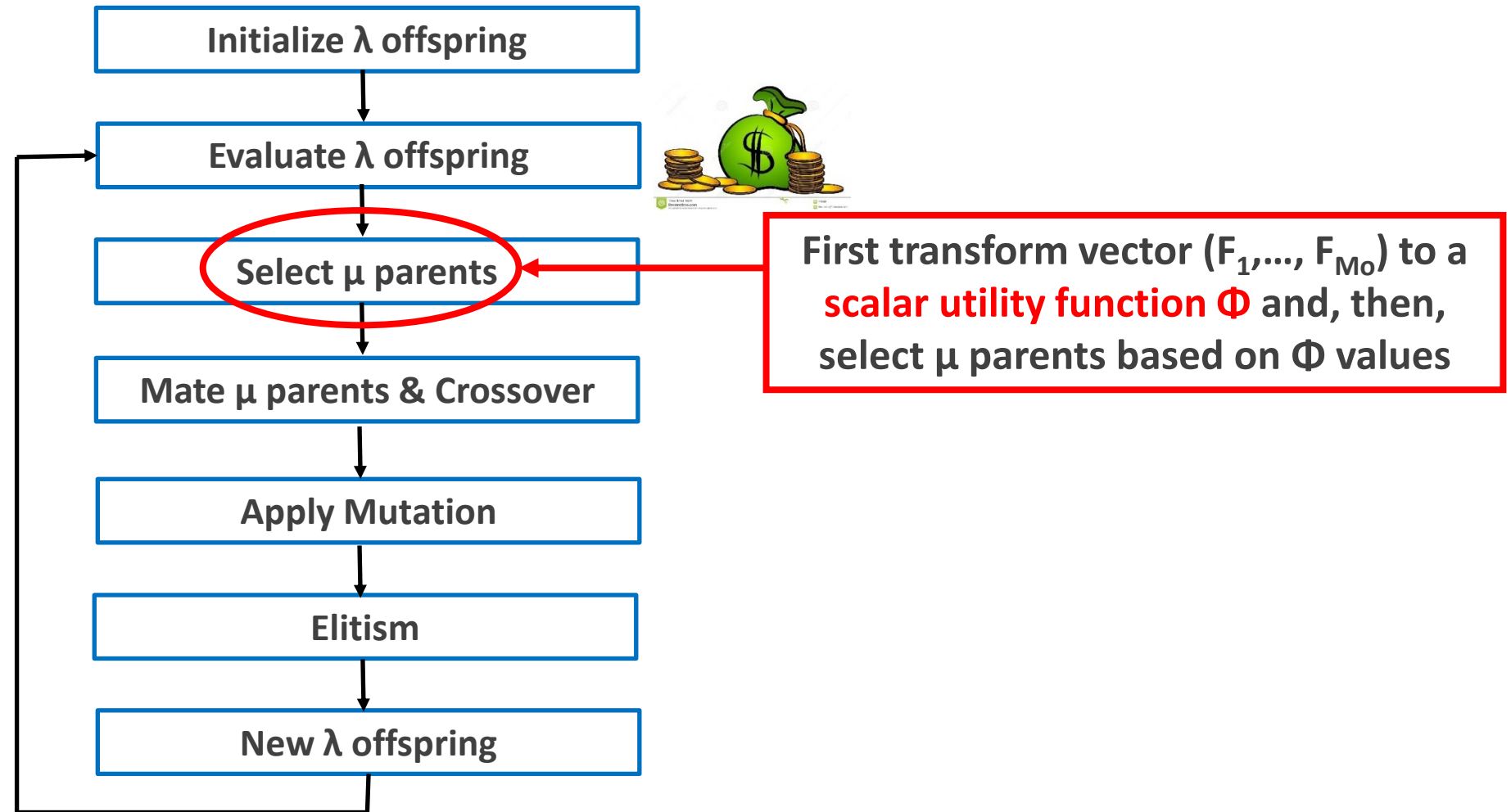
The **(μ, λ) EA**, with μ parents and λ offspring, supporting both binary and real coding of the design variables.

Flowchart of the (μ, λ) EA in a SOO Problem

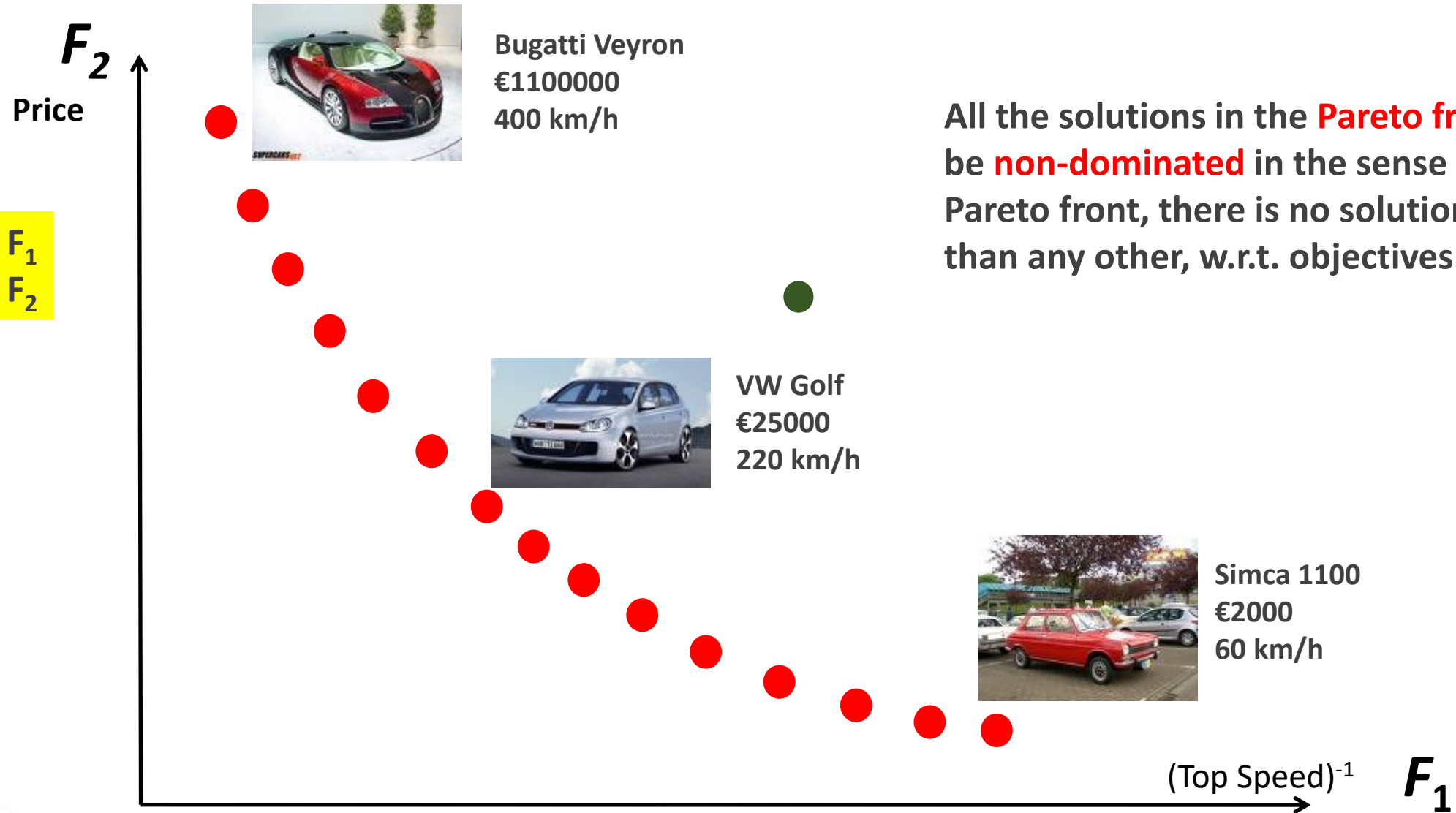


... λ calls to eval.exe,
amenable though to parallelization

Flowchart of the (μ, λ) EA in a MOO Problem



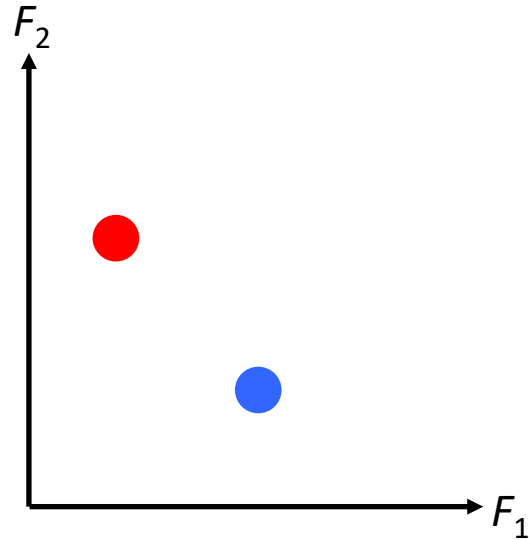
Pareto Front of Front of Non-Dominated Solutions



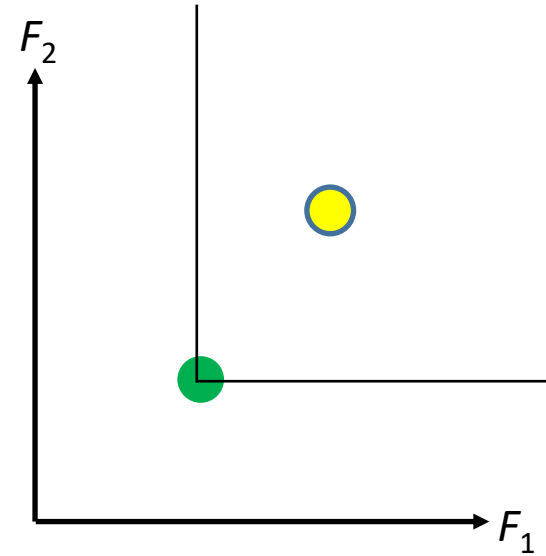
All the solutions in the **Pareto front** are said to be **non-dominated** in the sense that, in the Pareto front, there is no solution that is better than any other, w.r.t. objectives.

Pareto Front of Front of Non-Dominated Solutions - Compare

Min. F_1
Min. F_2

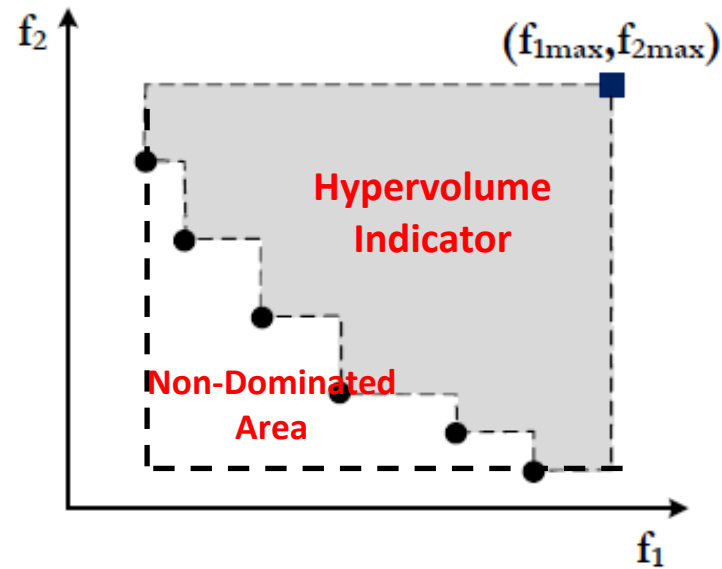


To improve one objective will imply to introduce a loss regarding the other ones.



The yellow solution is **dominated** by the green one.

A way to compare Fronts of Non-dominated Solutions in MOO



Definition of the **Hypervolume Indicator**
(for min. f_1 & min. f_2).

Constraint Handling in EAs

Exponential penalty if

$$0 < c_j(\vec{b}) < c_j^{relax}$$

$$f_m(\vec{b}) \leftarrow f_m(\vec{b}) + \prod_{j=1}^{M_c} \exp\left(a_j \frac{c_j}{c_j^{relax}}\right) \quad m \in [1, M_o]$$

Death penalty if

$$c_j(\vec{b}) > c_j^{relax}$$

(* minimization problem)

EAs: Exploration vs. Exploitation

In EAs, two main abilities maintained which are **Exploration** and **Exploitation**.

Exploration means that the algorithm searches for new solutions in new regions.

Exploitation means that the algorithm uses already existing solution(s) and makes refinement to it (them) so as to improve its (their) fitness.

Stochastic, Population-Based Optimization Methods. Pros & Cons

Pros:

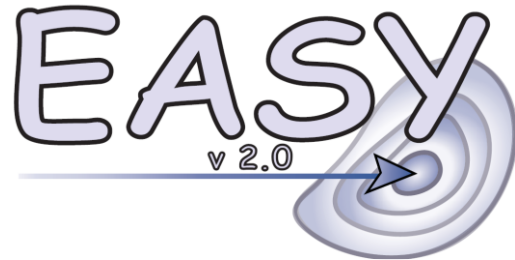
- Readily accommodate any analysis-evaluation software (as a black-box), to compute the cost or fitness function value(s) of candidate solutions.
- Gradient-free search.
- Compute (Pareto) front of non-dominated solutions, in many-objective optimization, via a single run.
- Handle constraints in the simplest possible way: through penalties.
- Are amenable to parallelization (simultaneous independent evaluations).

Cons:

- Require a great number of (costly/CFD) evaluations: many calls to eval.exe!

The PCOpt/NTUA EA: The EASY Platform

All theory and computations presented below are based on the s/w EASY developed by the PCOpt/NTUA, using a (μ, λ) EA as the background optimization method.



The Evolutionary Algorithm System

<http://velos0.ltt.mech.ntua.gr/EASY>

<http://147.102.55.162/EASY>

Part 2:
Population-based Stochastic-based Optimization
Methods for Beginners –
Evolution Operators

Notations – Key Terms

Notations:

μ = parent population size

λ = offspring population size

e = elite population size

g = generation index

ρ = number of parents to form an offspring

$$P_{\mu}^g, P_{\lambda}^g, P_e^g$$

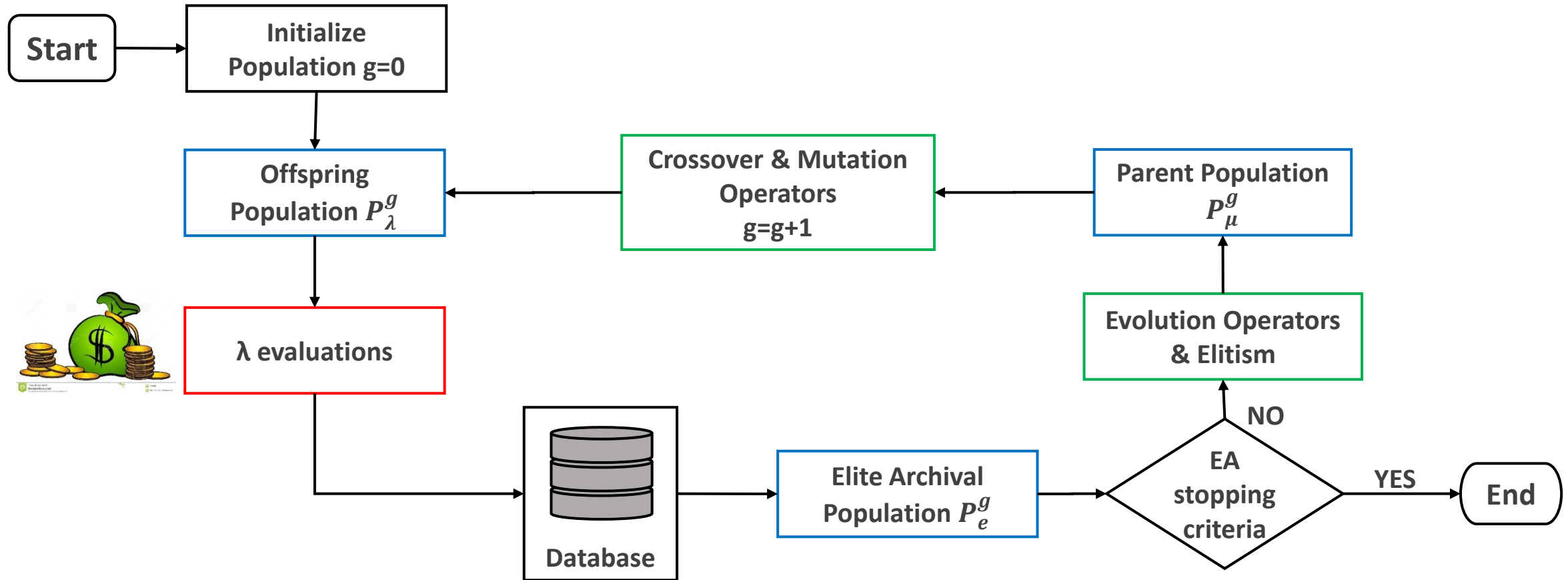
Key Terms:

Individual: Candidate solution

Genes: parameters (design variables) characterizing an individual

Chromosome (genotype): the set of genes of an individual

Flowchart of a standard Evolutionary Algorithm

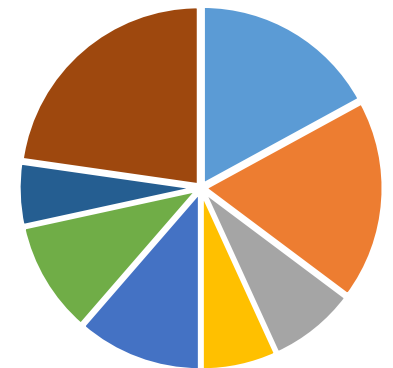
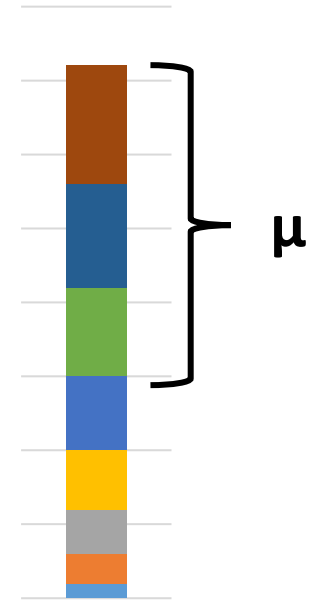


Parent Selection & Elitism (1/2)

$$P_{\mu}^g \leftarrow (P_{\mu}^{g-1} \cup P_{\lambda}^g)$$

Linear Ranking: $P_{\mu}^{g-1} \cup P_{\lambda}^g$ individuals are sorted based on their fitness value. The probability of selecting an individual is based on its rank in a linear manner, i.e. practically the μ best members are selected.

Proportional Selection: $P_{\mu}^{g-1} \cup P_{\lambda}^g$ individuals are associated with a probability based on their fitness value (smaller values correspond to higher selection probability). Form a **roulette wheel** where each individual is associated with a slot with angular width proportional to its selection probability, and turn it μ times.

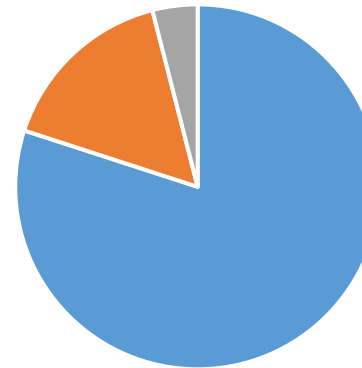


Parent Selection & Elitism (2/2)

Tournament Selection: Randomly select k individuals from $P_{\mu}^{g-1} \cup P_{\lambda}^g$ and sort them based on their fitness value. Select the best among them with a probability p , the second best with $p(1-p)$ and so on and so forth. Repeat this procedure μ times.

k : tournament size

p : tournament probability



$k=3$

$p=0.8$

80% for the blue, 16% for the orange and 4% for the grey to be selected as parent

Elitism:

Replace the worst members of P_{λ}^g with a few elite individuals from $P_e^g \rightarrow$

Practically increase the probability of an elite member to be selected as parent.

Chromosome representation

Real encoding: Chromosomes are strings of real values, e.g. [3.45 , 5.12 , -7.68 , 9.32 , 4.77]

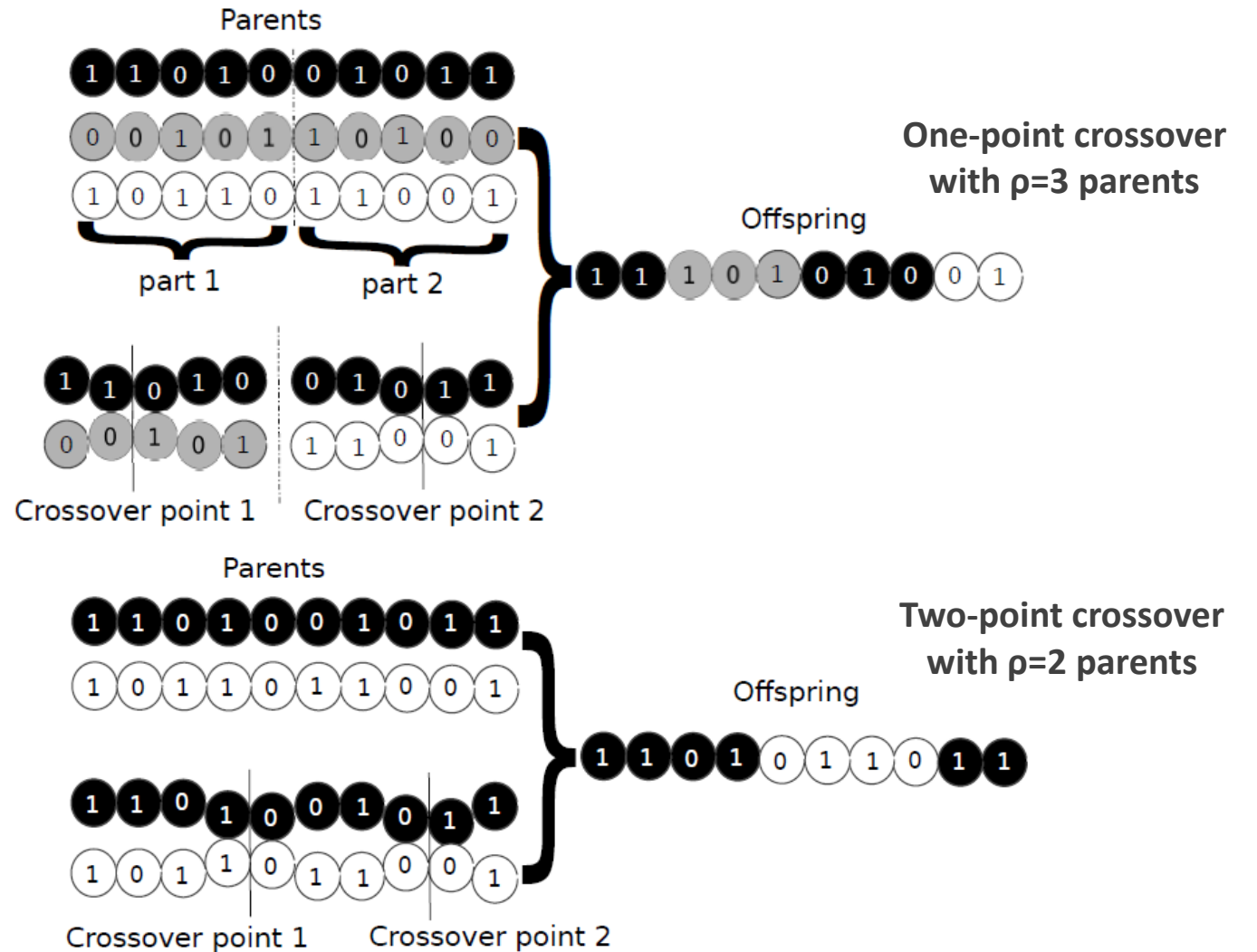
Binary encoding: Chromosomes are strings of 0s and 1s , e.g. [0110100111010100110]

Gray of reflected binary encoding: ordered binary encoding such that two successive integer values differ in only one bit/binary digit. Why?

<u>Decimal</u>	<u>Binary</u>	<u>Gray</u>
3	0011	0010
4	0100	0110

Binary Encoding Crossover

- One-point crossover
- Two-point crossover
- One- or two-point crossover per (design) variable
- etc.

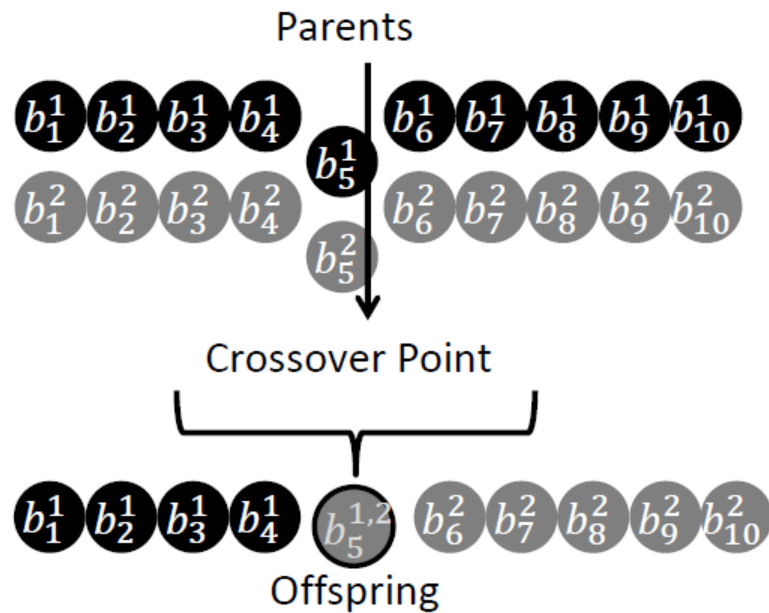


Real Encoding Crossover

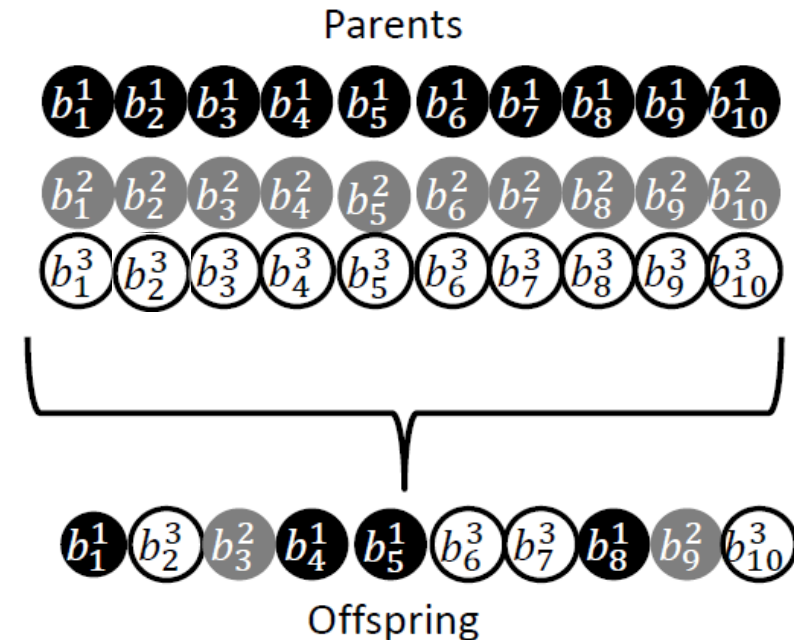
- One- or two-point crossover
- Discrete
- Intermediate
- Simulated binary
- etc.

$$\vec{b} = [(1 + \beta)\vec{b}^{P1} + (1 - \beta)\vec{b}^{P2}] \quad \beta = \begin{cases} (2u)^{1/(n+1)} & , u < 0.5 \\ \left(\frac{1}{2(1-u)}\right)^{1/(n+1)} & , u \geq 0.5 \end{cases}$$

Simulated Binary Crossover (SBX) $u \in [0, 1], n \in [1, N]$



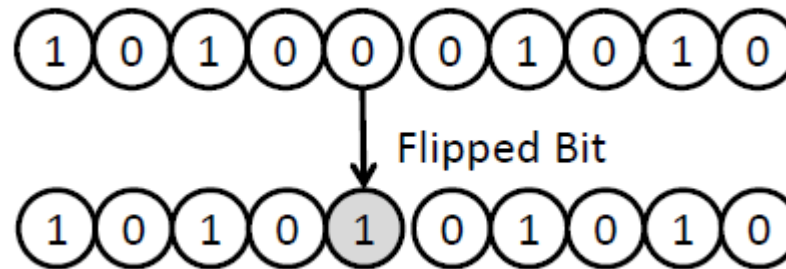
One-point crossover with $p=2$ parents



Discrete crossover with $p=3$ parents

Mutation

Binary/Gray encoding:

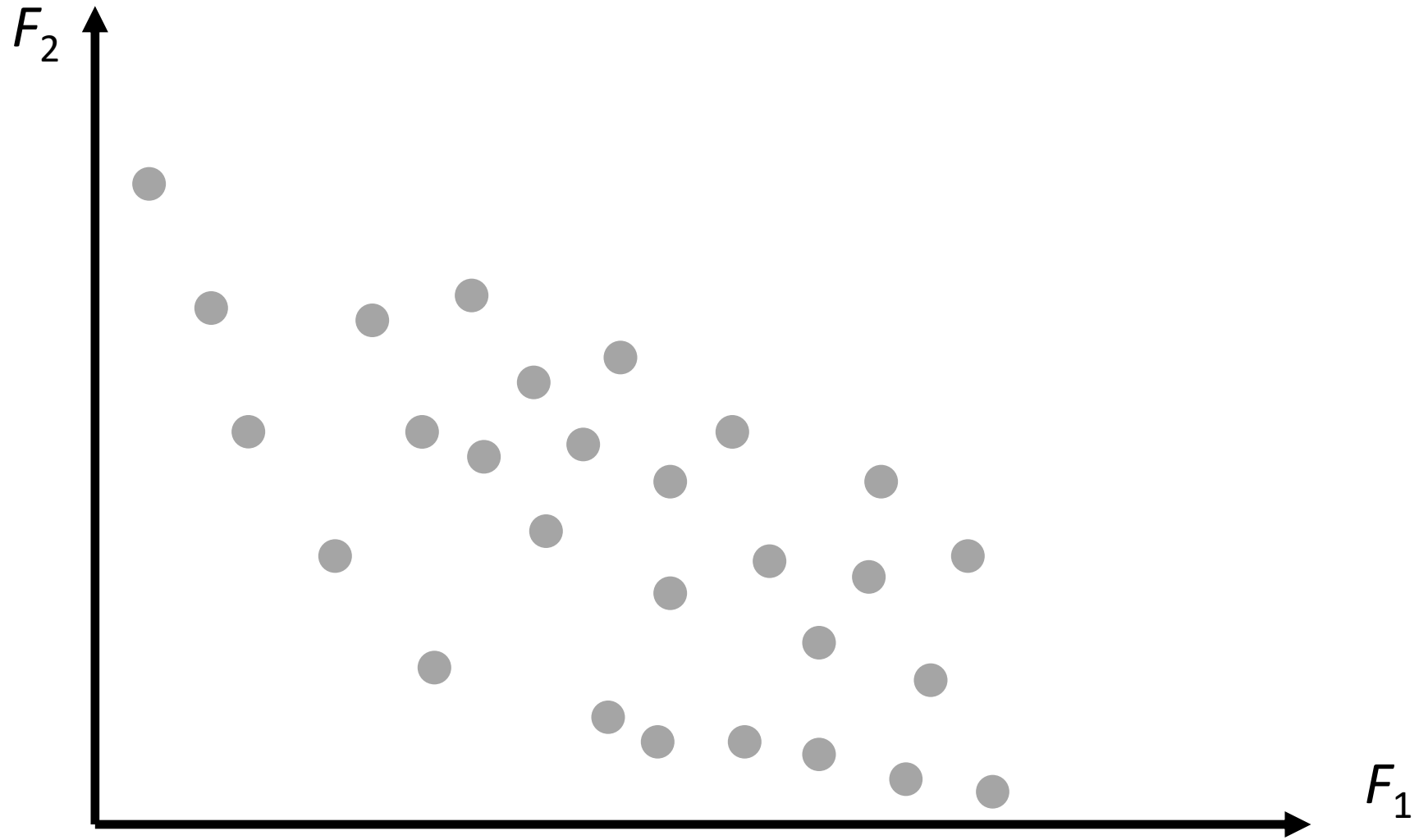


Real encoding:

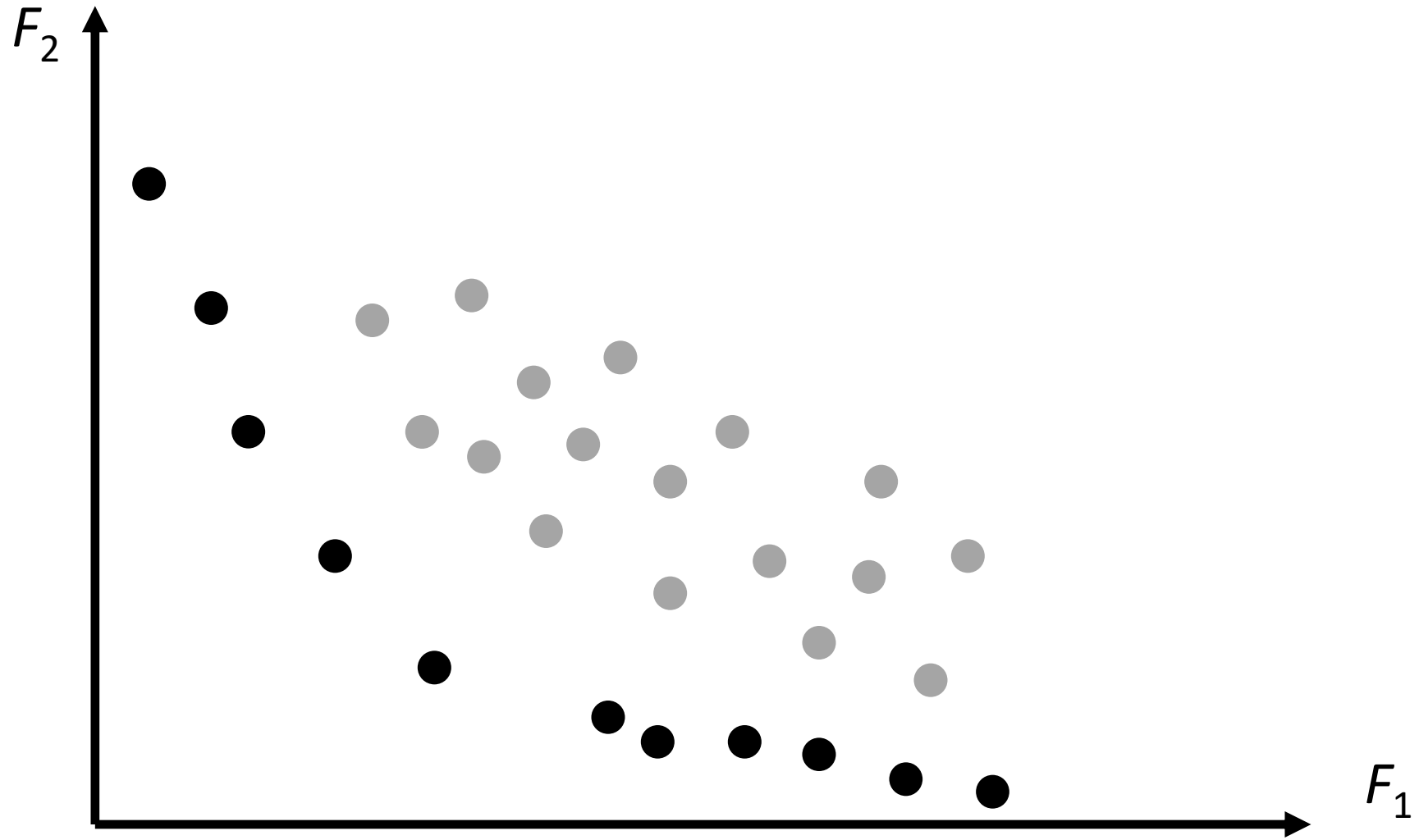
$$\vec{b} = \begin{cases} \vec{b} + D(g, \vec{b}^{max} - \vec{b}), & r > 0.5 \\ \vec{b} - D(g, \vec{b} - \vec{b}^{min}), & r \leq 0.5 \end{cases}$$

D : depends on the current generation and the maximum generations

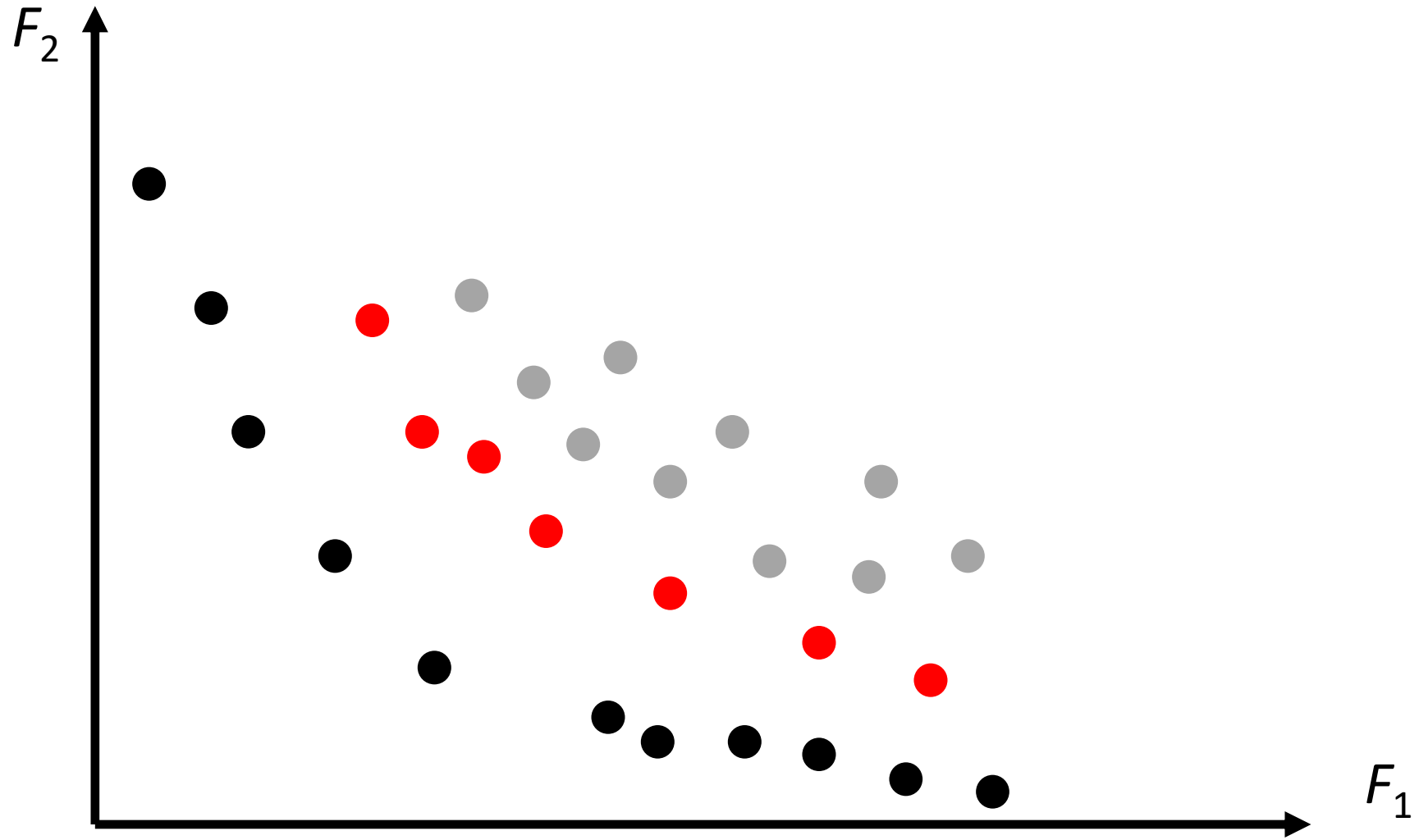
Computing Φ in MOO problems - Front Ranking (1/5)



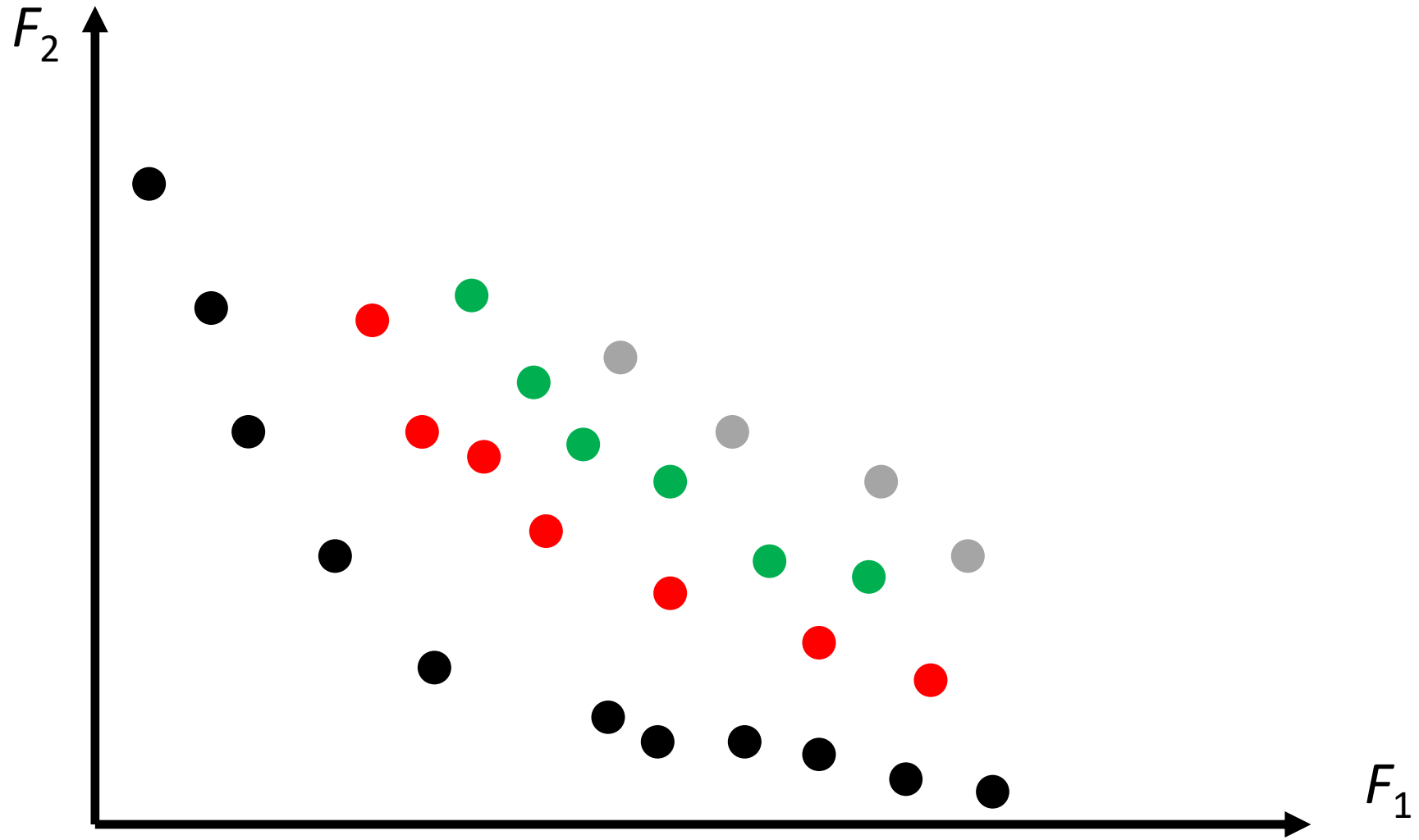
Computing Φ in MOO problems - Front Ranking (2/5)



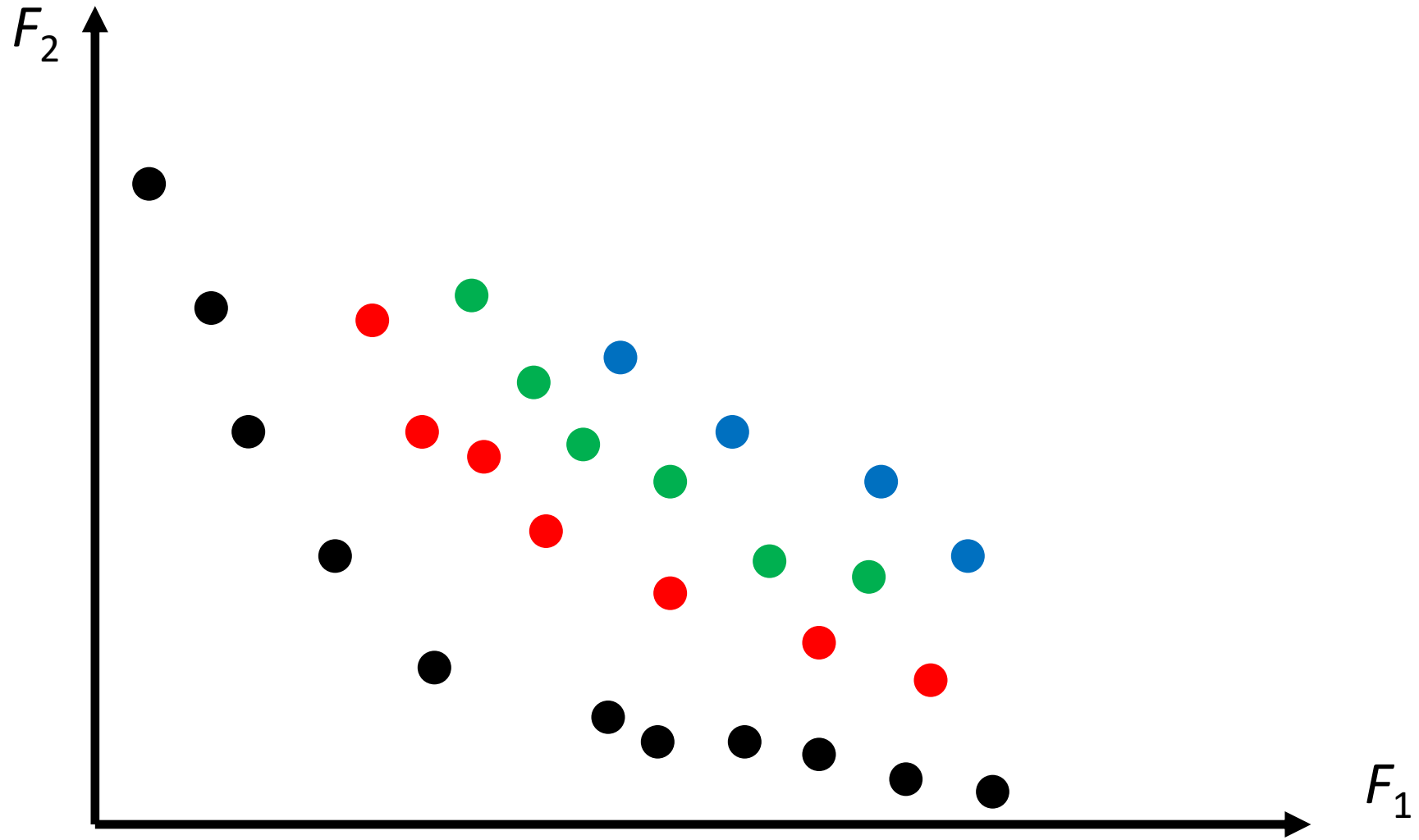
Computing Φ in MOO problems - Front Ranking (3/5)



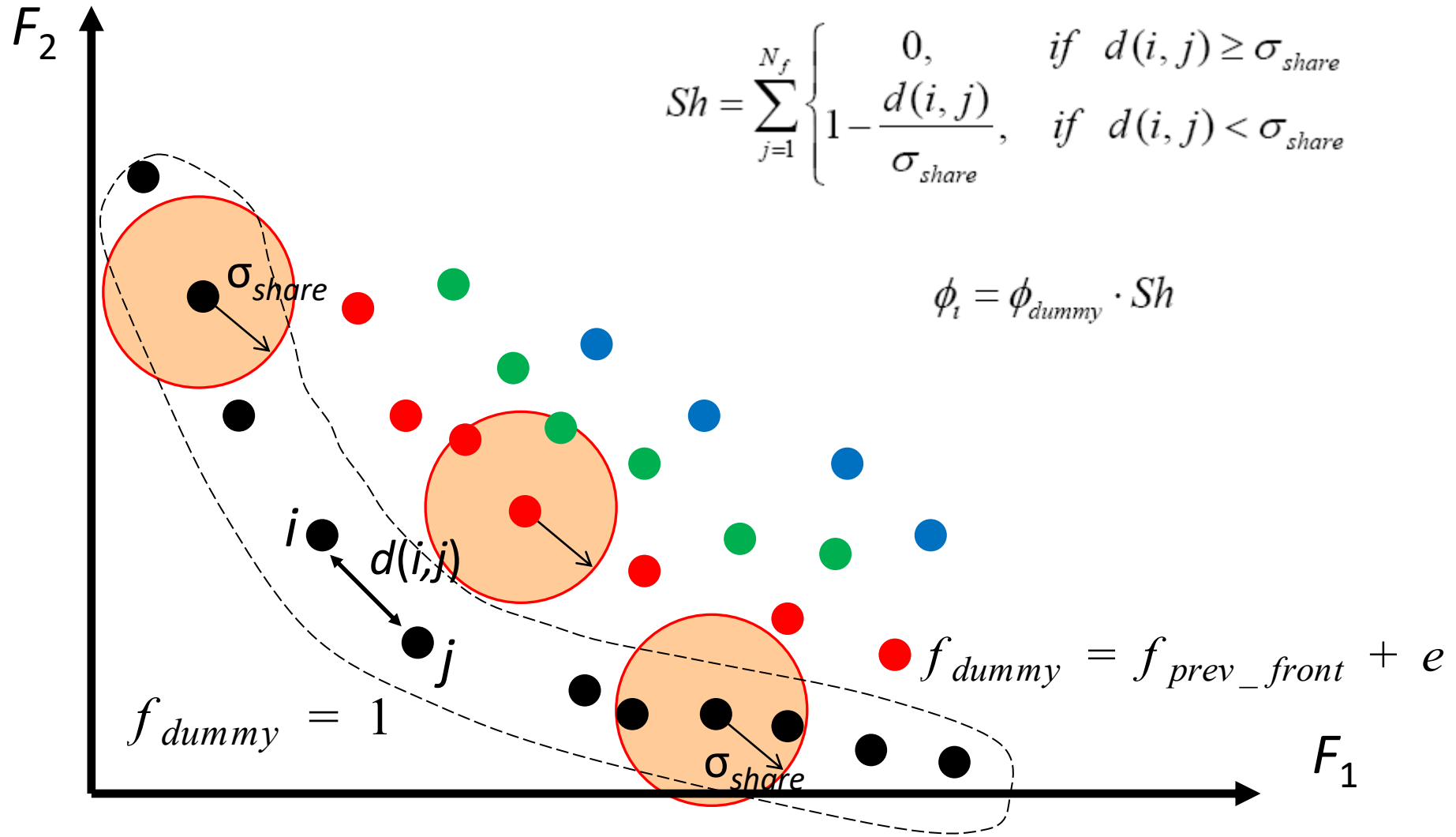
Computing Φ in MOO problems - Front Ranking (4/5)



Computing Φ in MOO problems - Front Ranking (5/5)



Computing Φ in MOO problems - NSGA



Part 3:
Cost Reduction in EAs

Possible ways to Reduce the Computational Cost of EAs

- ◆ Use **Metamodels** (or Surrogate Evaluation Models) to reduce the number of calls to the expensive **Problem Specific Model (PSM)** (CFD, MD, etc. s/w) → **Metamodel-Assisted EAs (MAEAs)**.
- ◆ Perform Distributed search → **Distributed EAs or DEAs or DMAEAs**.
- ◆ Consider Dimensionality Reduction to overcome the «curse of dimensionality» → **PCA-driven EAs or MAEAs**.
- ◆ Perform Hierarchical or Multilevel search → **Hierarchical EAs (HEAs or HMAEAs)**.
- ◆ Hybridize EAs with Gradient-based optimization → **Hybrid Optimization**.
- ◆ Overcome the generation synchronization barrier → **Asynchronous EAs**.

Metamodel-Assisted EAs (MAEAs) – The Concept

Having already evaluated a number of candidate solutions (individuals), build a **model-agnostic black-box** (metamodel or surrogate evaluation model) to approximate the objective or constraint function values of new individuals generated by the EA and avoid the use of the costly PSM tool, as much as possible.

Valid for both Single- & Multi-Objective Optimization (SOO & MOO)

Questions:

- When and how to train the metamodel(s)?
- Which metamodel (polynomial regression, neural networks, ...)?
- How to collect training samples for the metamodel?
- How to use the cost/fitness function approximation provided by the metamodel?

MAEAs: Ways to Implement Metamodels

MAEAs with Off-Line Trained Metamodels:

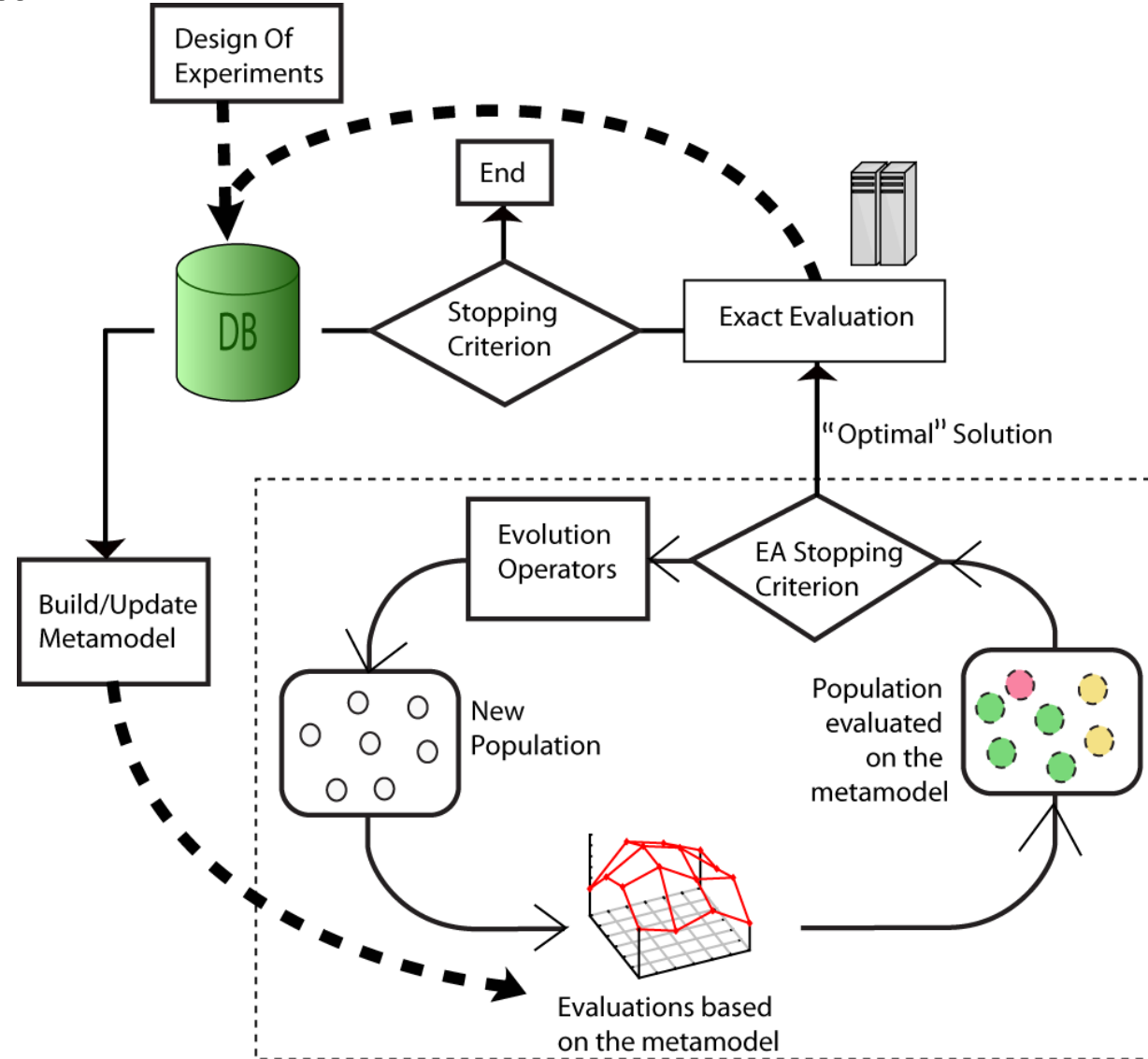
- A **Design of Experiment (DoE)** technique is used to sample the design space, collect training patterns, evaluate them on the PSM & store them into the **Database (DB)**.
- A global metamodel is trained and the EA search relies exclusively upon its use.
- “Optimal” solution(s) is/are re-evaluated on the PSM (e.g. the CFD code).
- If necessary, new samples are collected & evaluated on the PSM, DB is enriched, a new (hopefully, better) metamodel is trained.
- Iterate (successive EA-based searches) until convergence.

MAEAs with On-Line Trained Metamodels:

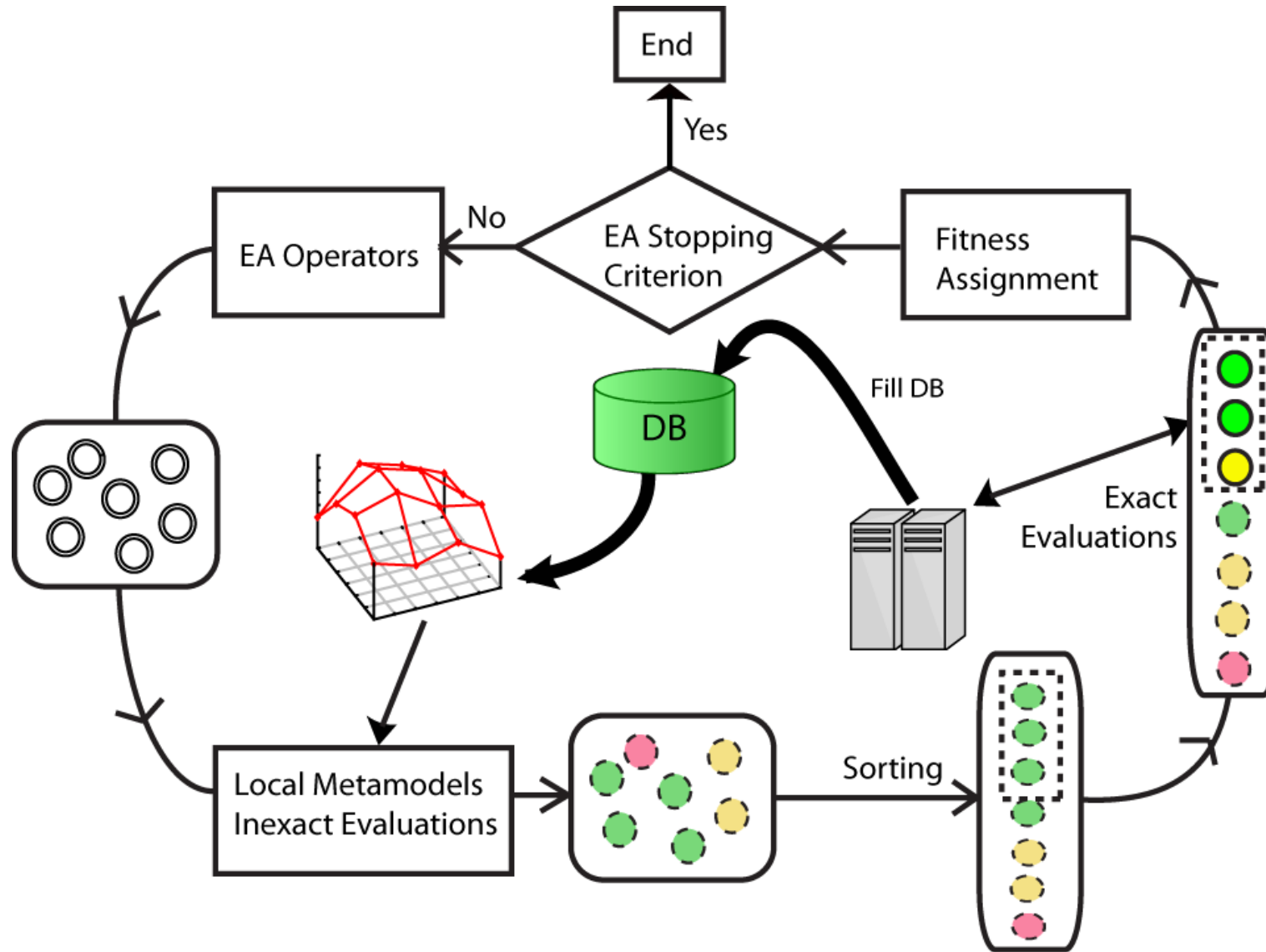
- Local metamodels are trained during the evolution by on-line collected training patterns. The **Low-Cost Pre-Evaluation (LCPE)** phase.
- Coordinated use of metamodels and the PSM.
- Considered to be the distinguishing feature of



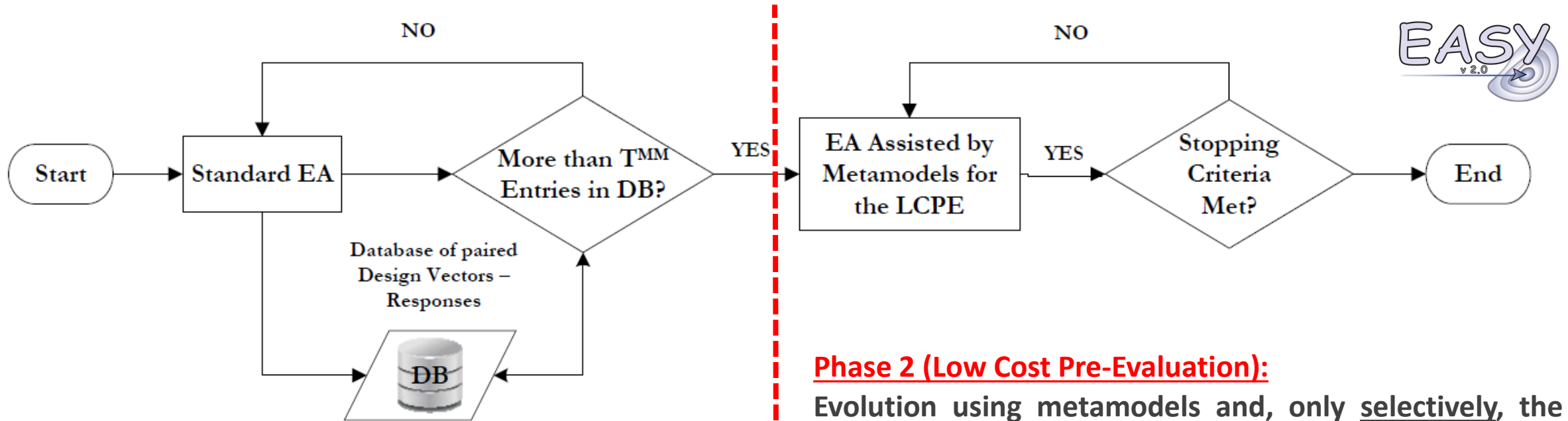
A MAEA with Off-Line Trained Metamodels



A MAEA with On-Line Trained Metamodels



A MAEA with On-Line Trained Metamodels



Phase 1:

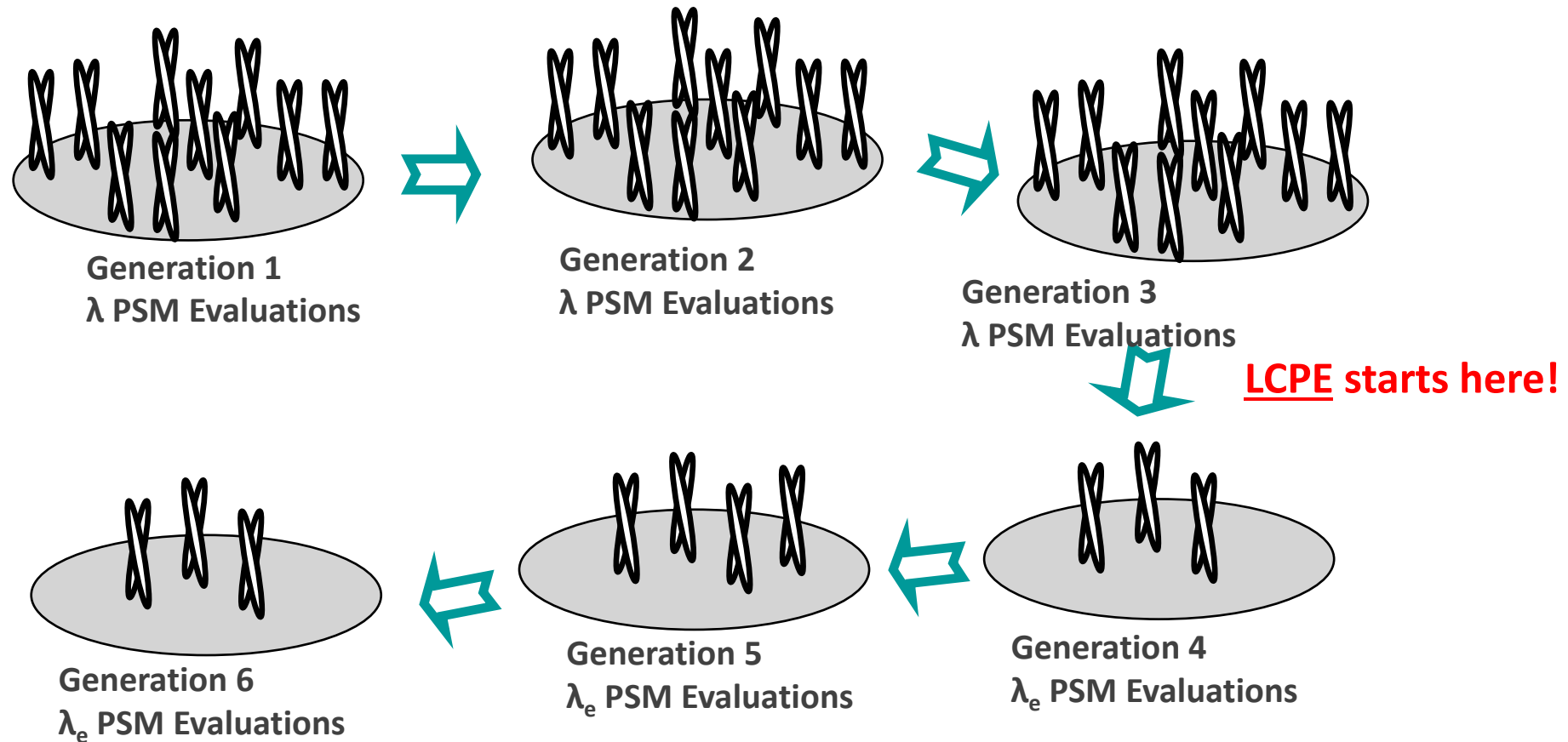
Exclusive use of the PSM; no use of metamodels
Terminates once T^{MM} evaluated individuals are stored in the DB.

Phase 2 (Low Cost Pre-Evaluation):

Evolution using metamodels and, only selectively, the PSM.

- All population members are evaluated on **local/ personalized metamodels** trained on neighboring data previously stored in the DB.
- The best λ_e (λ_e « λ , $\lambda_{e,\min} \leq \lambda_e \leq \lambda_{e,\max}$) of them are re-evaluated on the PSM & stored in the DB; this determines the computational cost of each generation.

A MAEA with On-Line Trained Metamodels – The LCPE Phase at a glance



Int. Review Journal Progress in Aerospace Sciences, 38:43-76, 2002.

Dr. Varvara G. Asouti, vasouti@mail.ntua.gr

On Metamodels (Radial Basis Function-RBF Networks)

Unless otherwise stated, used in all presented studies.

$$o(\vec{x}) = \sum_{k=1}^K w_k \mathcal{G}(\|\vec{x} - \vec{c}^{(k)}\|_2)$$

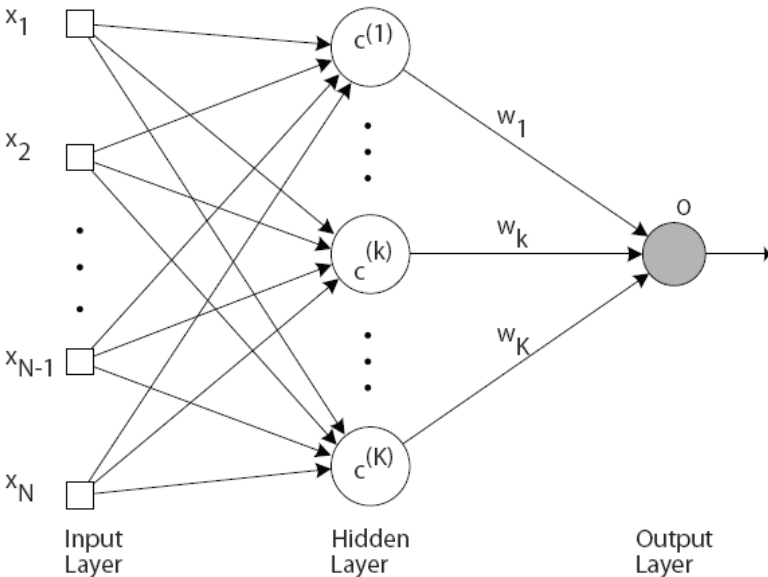
Possible activation functions:

$$\mathcal{G}(h) = \exp(-h^2/r^2) \quad \mathcal{G}(h) = (h^2 + r^2)^{1/2} \quad \mathcal{G}(h) = (h^2 + r^2)^{-1/2} \text{ etc.}$$

Training:

$$\begin{bmatrix} \mathcal{G}(\|\vec{x}^{(1)} - \vec{c}^{(1)}\|_2) & \dots & \mathcal{G}(\|\vec{x}^{(1)} - \vec{c}^{(T)}\|_2) \\ \vdots & \ddots & \vdots \\ \mathcal{G}(\|\vec{x}^{(T)} - \vec{c}^{(1)}\|_2) & \dots & \mathcal{G}(\|\vec{x}^{(T)} - \vec{c}^{(T)}\|_2) \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ \vdots \\ w_T \end{bmatrix} = \begin{bmatrix} \zeta_1 \\ \vdots \\ \zeta_T \end{bmatrix}$$

Solution of a linear/symmetric system. Interpolation (**K=T**) or Approximation (**K<T**).



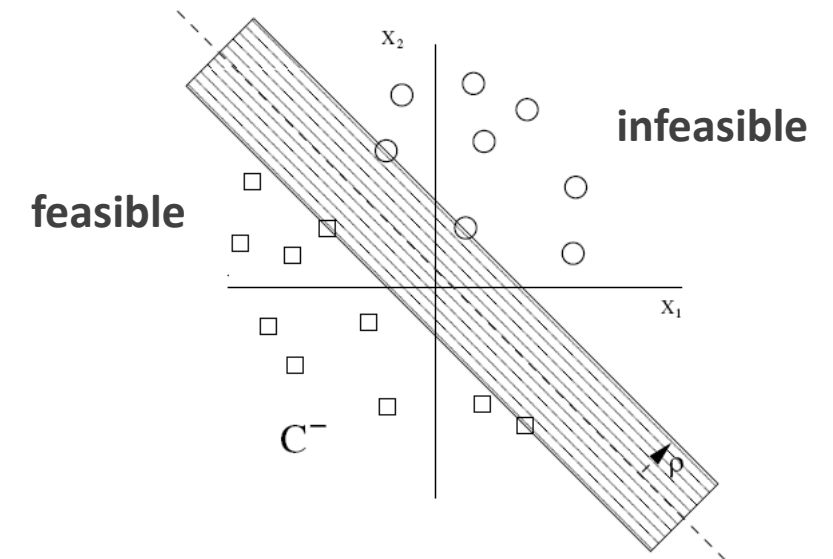
RBF network with N inputs, **K hidden units** (RBF centers) and a single output ($M_o=1$). To be trained with **T training patterns**.

Constrained Optimization in MAEA – Support Vector Machines (SVM)

Infeasible population members, penalized with an (+/-) infinite F value, should not be used as training patterns for the metamodel, but, on the other hand, the MAEA should take them into account. A special treatment is needed.

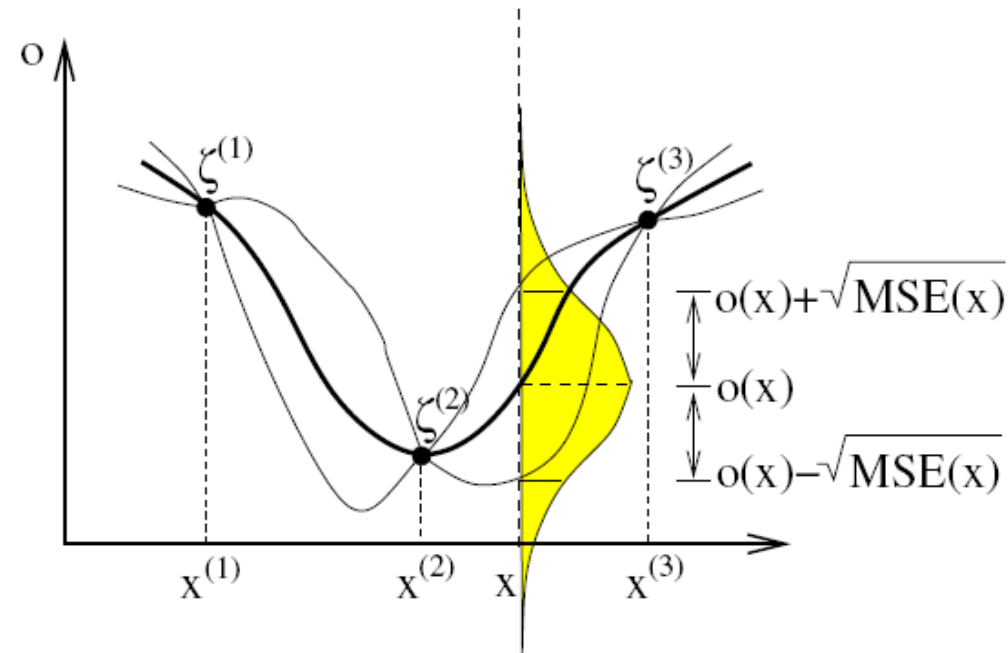
Proposed Treatment: A trained SVM classifies new individuals as “feasible” or “infeasible”:

- RBF is used only for individuals marked as “feasible”.
- Individuals marked as “infeasible” are penalized without resorting to the metamodel.



Other Metamodels - Kriging

Gaussian processes (Kriging) estimate and use (as an extra criterion) the Confidence Interval, pertinent to the guessed value of the objective or constraint function.



IEEE Transactions on Evolutionary Computation, 10:421-439, 2006.

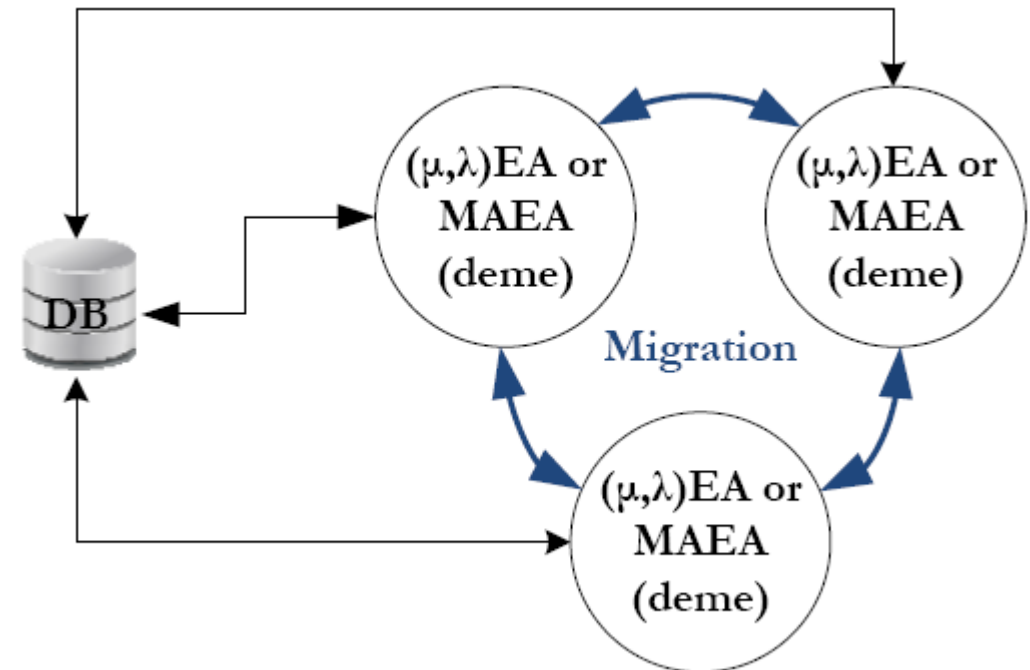
Dr. Varvara G. Asouti, vasouti@mail.ntua.gr

Distributed EAs and MAEAs

Run more than one EAs, with different populations evolving in semi-isolation: by regularly exchanging promising etc individuals.

User-Defined Parameters:

- Number of demes or islands
- Communication topology
- Communication frequency
- Migration policy
- EA set-up per deme; exploration/exploitation oriented demes!



Common DB for all demes.

 *International Journal for Numerical Methods in Fluids*, 53:455-469, 2007.

Dr. Varvara G. Asouti, vasouti@mail.ntua.gr

Demo Case A (SOO)

Shape optimization (ShpO) of an isolated airfoil.

Target: max. Lift (C_L).

Inviscid flow: $M_{inf}=0.40$, $\alpha_{inf}=5^\circ$.

Constraints: Lower and upper bounds of the design variables.

Unstructured grid, $\sim 20K$ nodes.

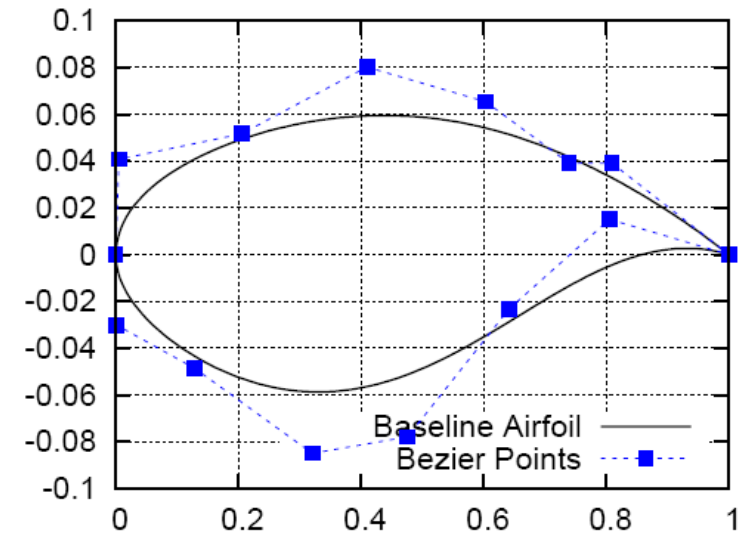
Parameterization: 2 Bezier curves (8 Control Points each).

N=12 DoFs (internal control points moving in the y-direction).

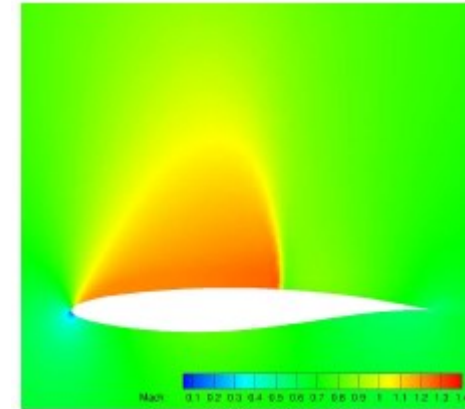
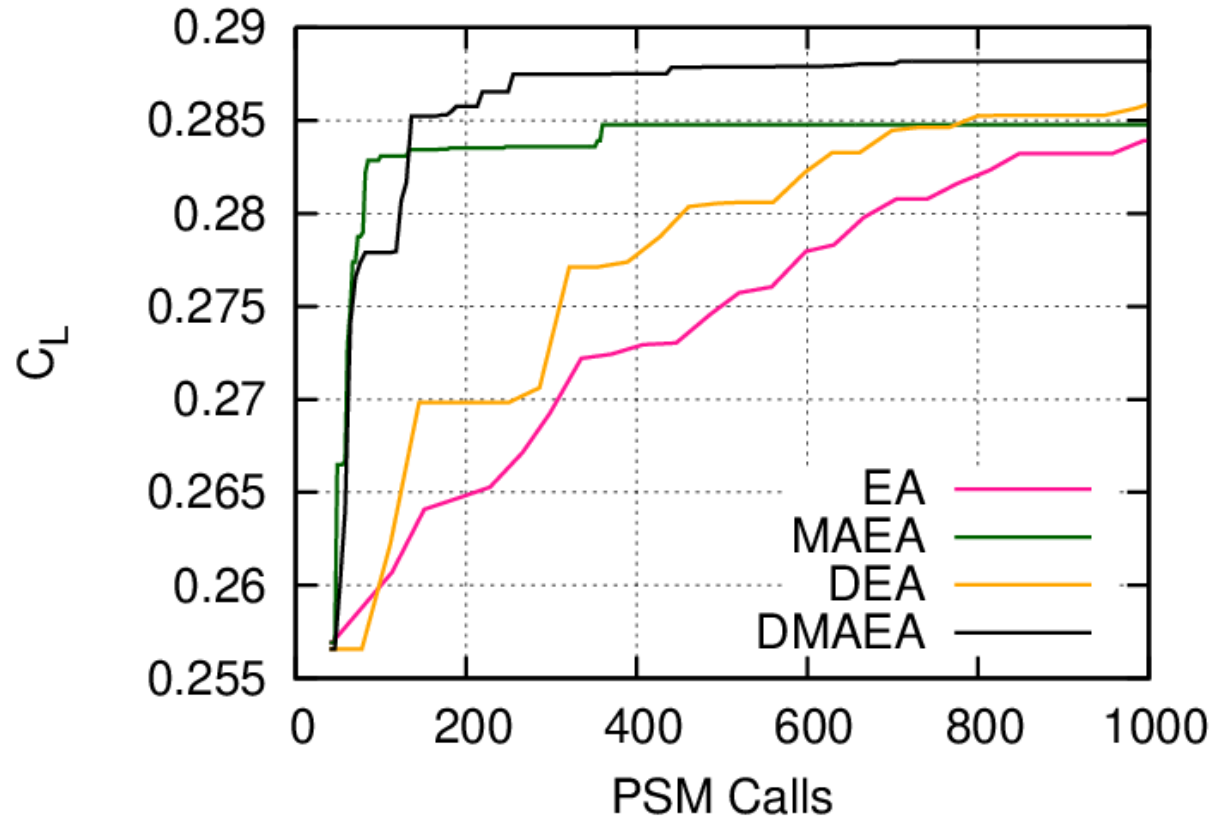
PSM: The PCOpt/NTUA GPU-enabled flow solver (PUMA).

Cost per evaluation: ~ 5 sec. on one NVIDIA K20 GPU.

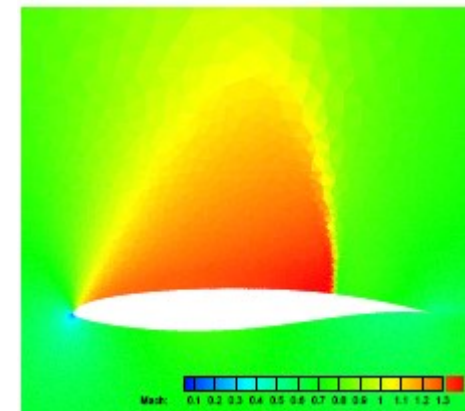
Basis of Comparison: a (20,40)EA.



Demo Case A (SOO)



Starting



Optimal

(20,40) EA
 (20,40) MAEA, $T^{MM}=40$, $\lambda_e=3$
 DEA or DMAEA: two demes (10,20)

Demo Case B (MOO)

Shape optimization of an isolated wing (two objectives)

Target: max. Lift (L) and min. Drag (D)

Inviscid flow: $M_{inf}=0.8394$, $\alpha_{pitch,inf}=3.06^\circ$, $\alpha_{yaw,inf}=0^\circ$.

Constraints: Lower and upper bounds of the design variables.

Unstructured grid, ~ 1.33 Mi nodes.

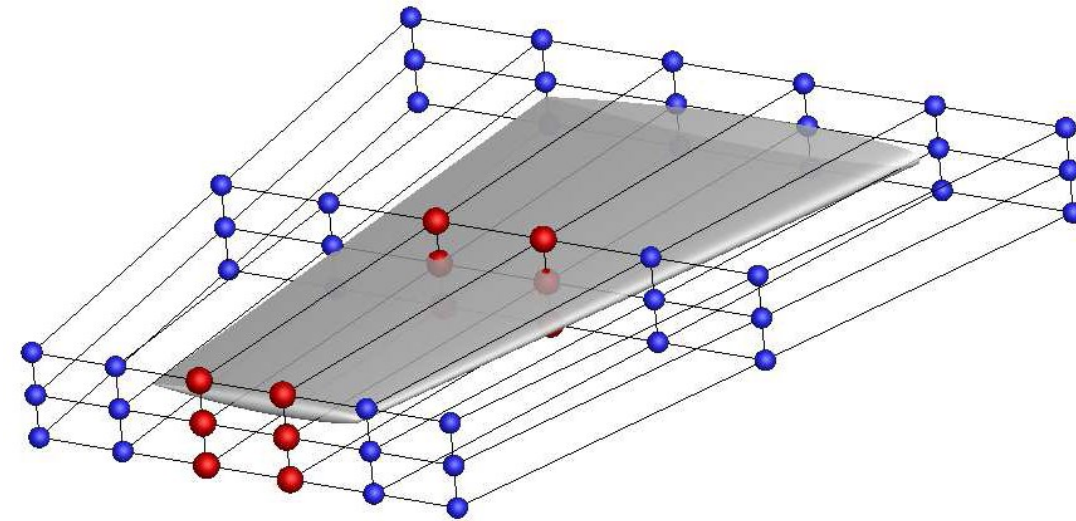
Parameterization: 6x3x3 Volumetric NURBS Control Grid.

N=24 DoFs (internal control points moving normal to the planform).

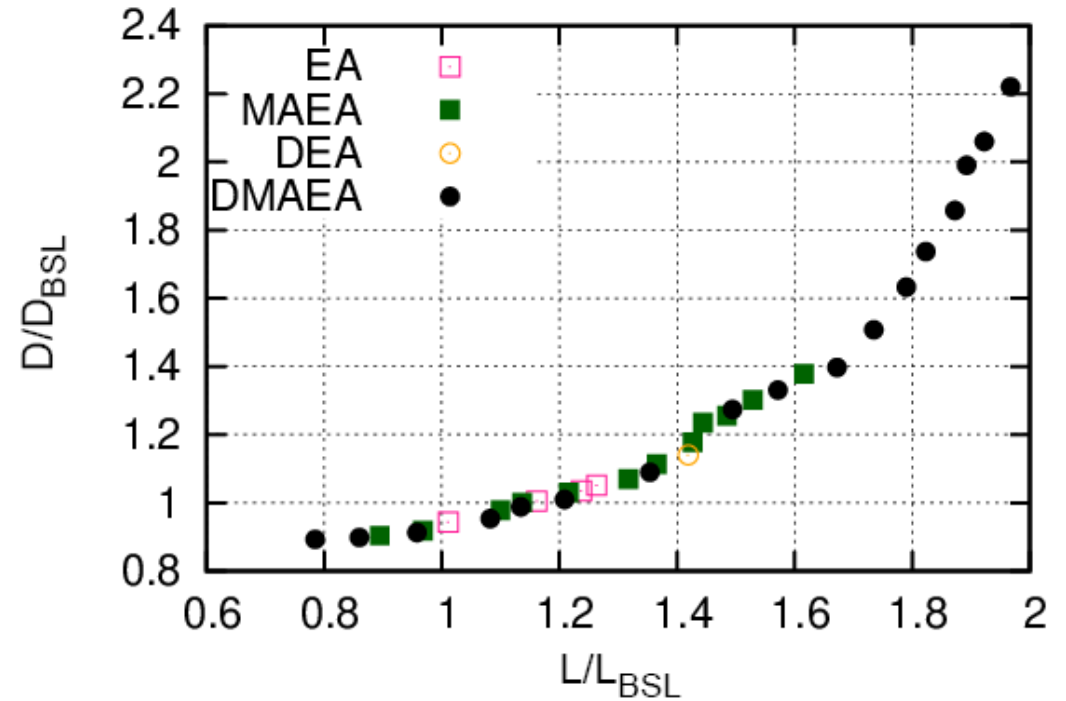
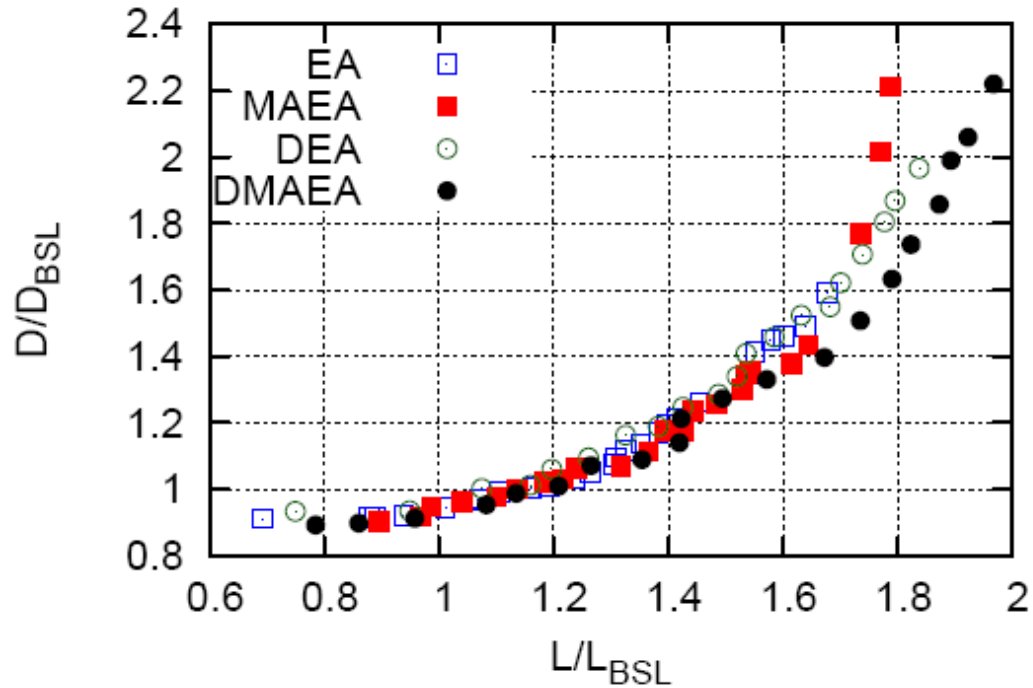
PSM: The PCOpt/NTUA GPU-enabled flow solver (PUMA).

Cost per evaluation: ~ 2 min. on one NVIDIA P100 GPU.

Basis of Comparison: a (10,20)EA.

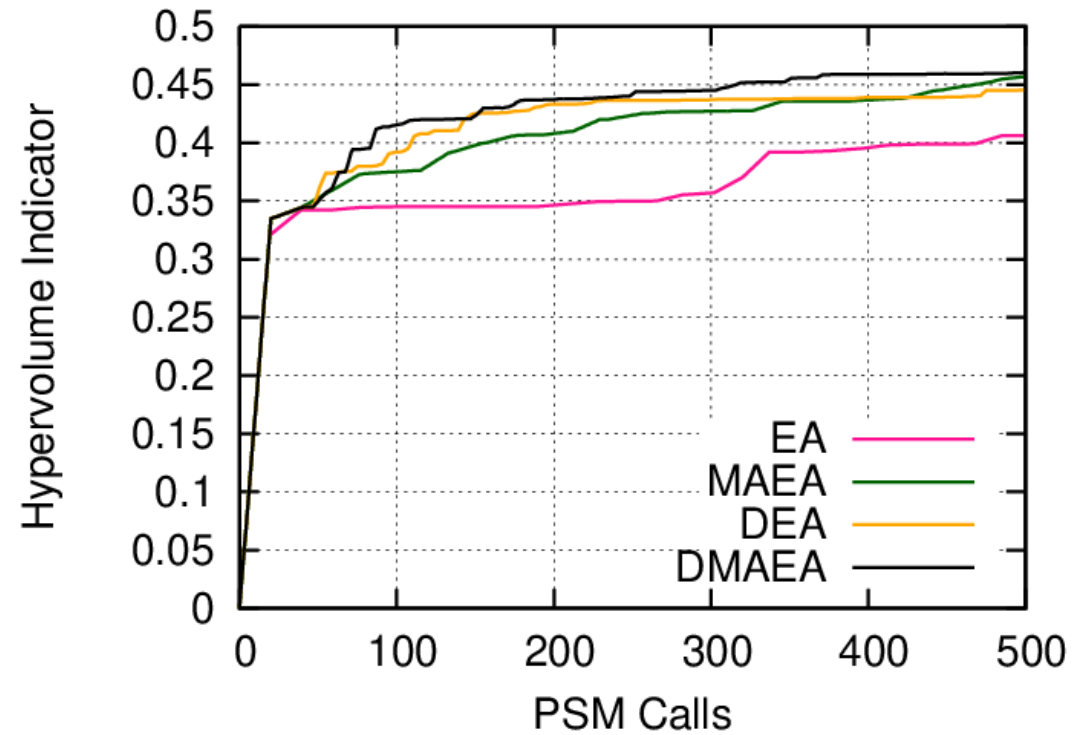


Demo Case B (MOO)



$T^{MM} = 30$ evaluations on the PSM, before starting the LCPE phase.

Demo Case B (MOO)



(MA)EAs & the Curse of Dimensionality

- The evolution slows down in the presence of **many design variables**.
- In MAEAs building a dependable metamodel is difficult. Need for many training patterns. Consequently:
 - The cost for training metamodel(s) increases a lot.
 - The start of the LCPE phase must be delayed.
 - The quality of metamodel-based predictions is not as good as it should.

Remedy (in each and every generation of the MAEA):

- Perform the **Principal Component Analysis (Kernel PCA)** of the λ members of the current offspring population, and compute the eigenvectors (which are the principal components defining the **feature space**) and the eigenvalues (their variances).
- Perform the evolution in the so-computed **feature space**, and enjoy the benefits of running a “more separable” optimization problem.
- Train metamodels using a small number of the most important principal components (less input units) using **truncation**.

D/MA/EA-PCA: Demo Case B (Revisited)

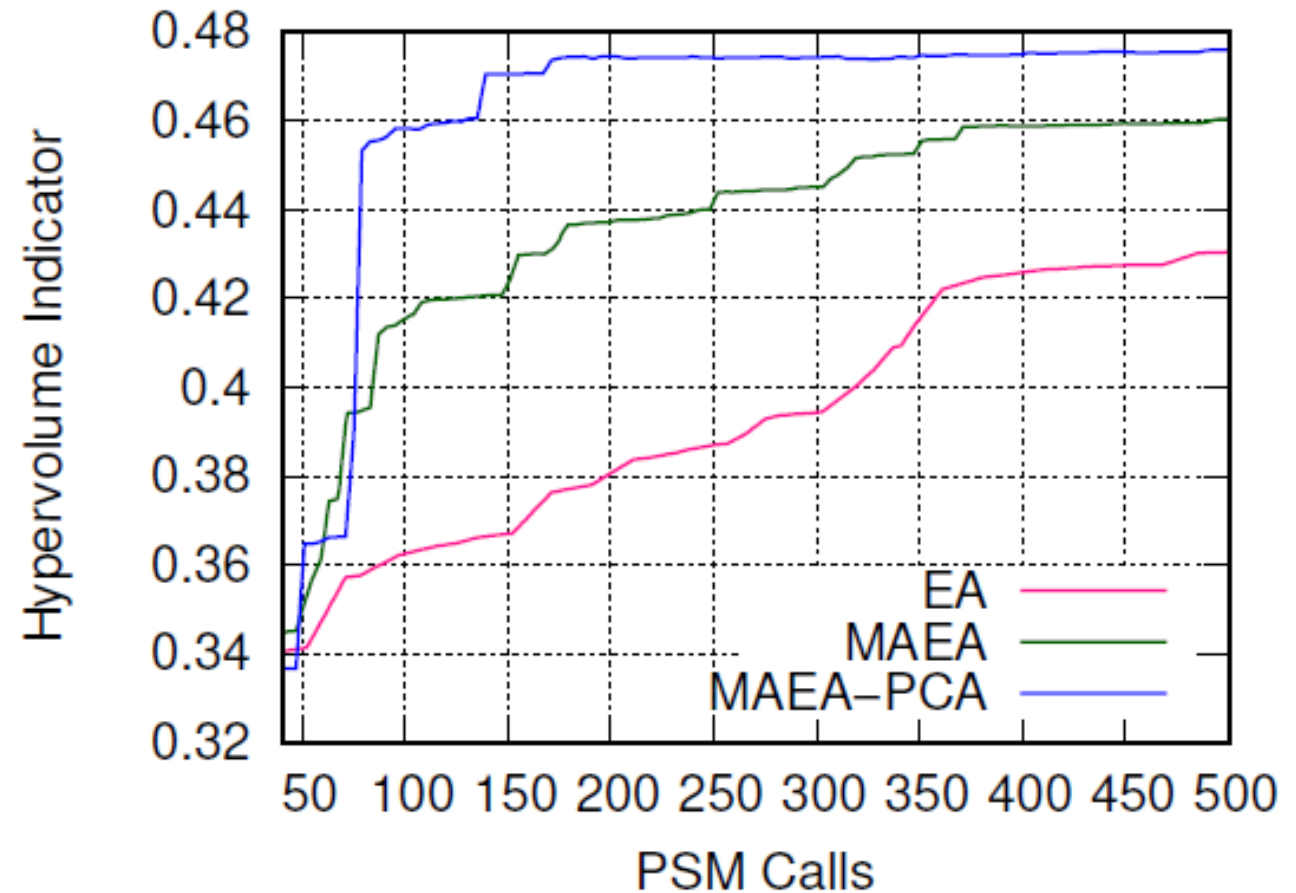
ShpO of an isolated wing

Target: max. Lift (L) and min. Drag (D)

New conditions: $M_{inf}=0.8395$,
 $\alpha_{pitch,inf}=3.06^\circ$, $\alpha_{yaw,inf}=0^\circ$.

N=24 DoFs

Basis of Comparison: a (10,20)EA.



Demo Case C (SOO)

SOO & MOO of the NACA4318 isolated airfoil.

$M_\infty=0.13$, $\alpha_\infty=2.2^\circ$, $Re=3.8 \times 10^6$.

Controlled by the 8×5 control box. **N=32 DoFs**.

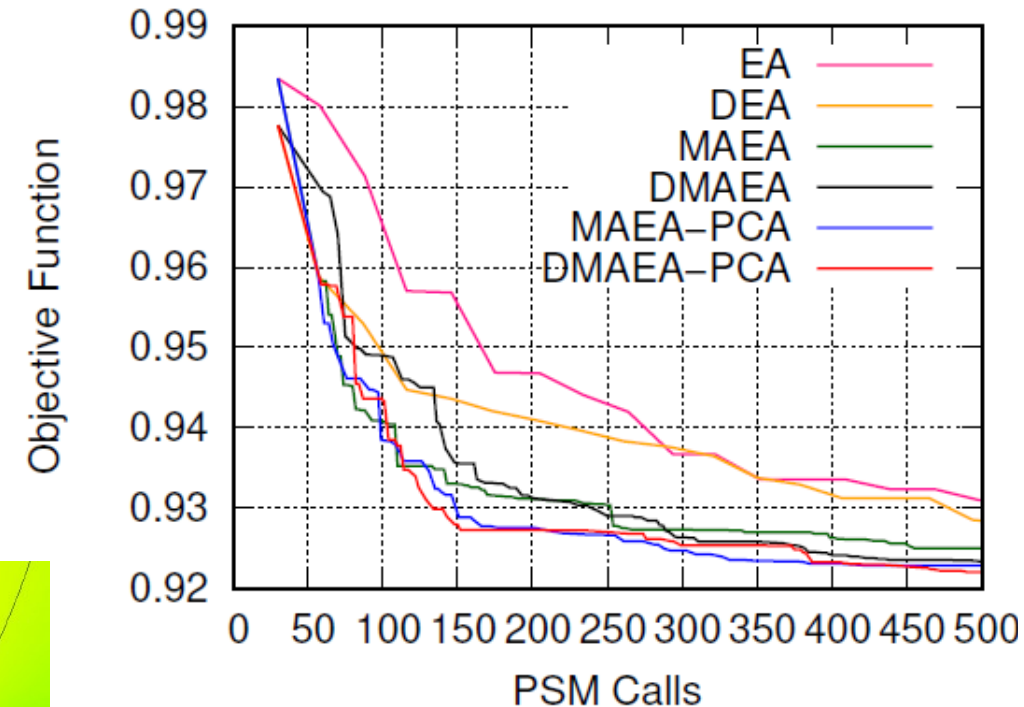
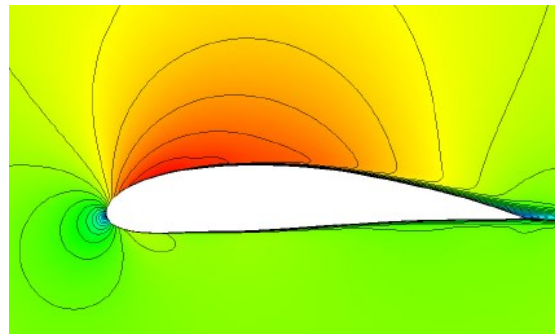
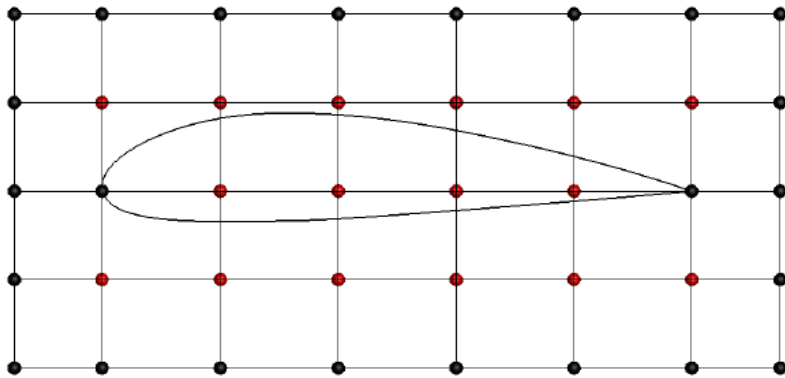
Unstructured grid with $\sim 30K$ nodes.

Targets: C_L & C_D . Weighted sum in the SOO.

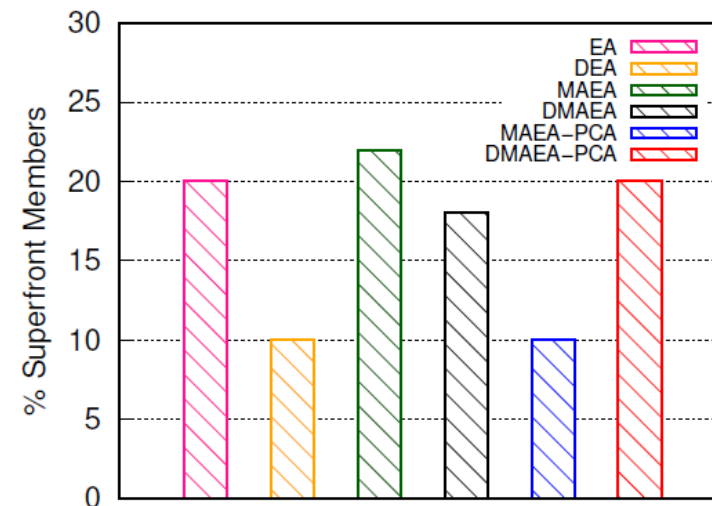
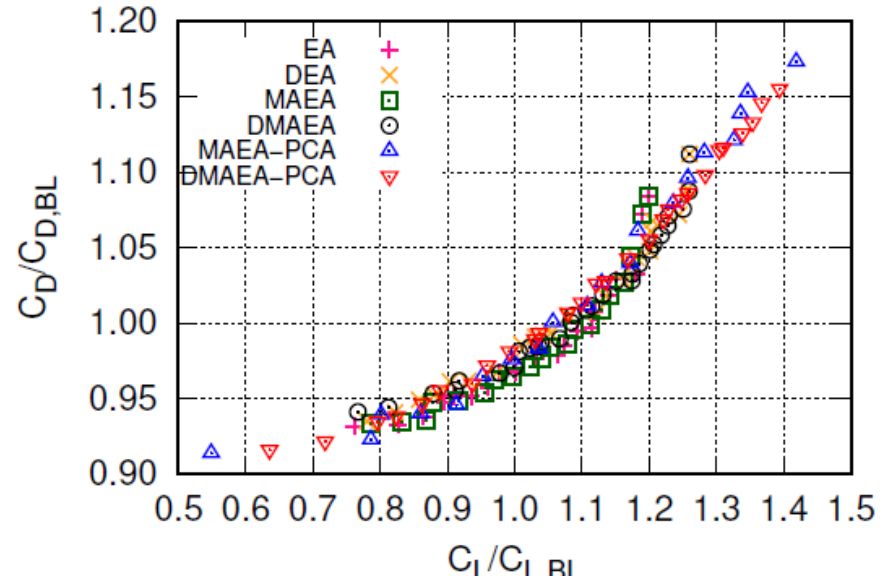
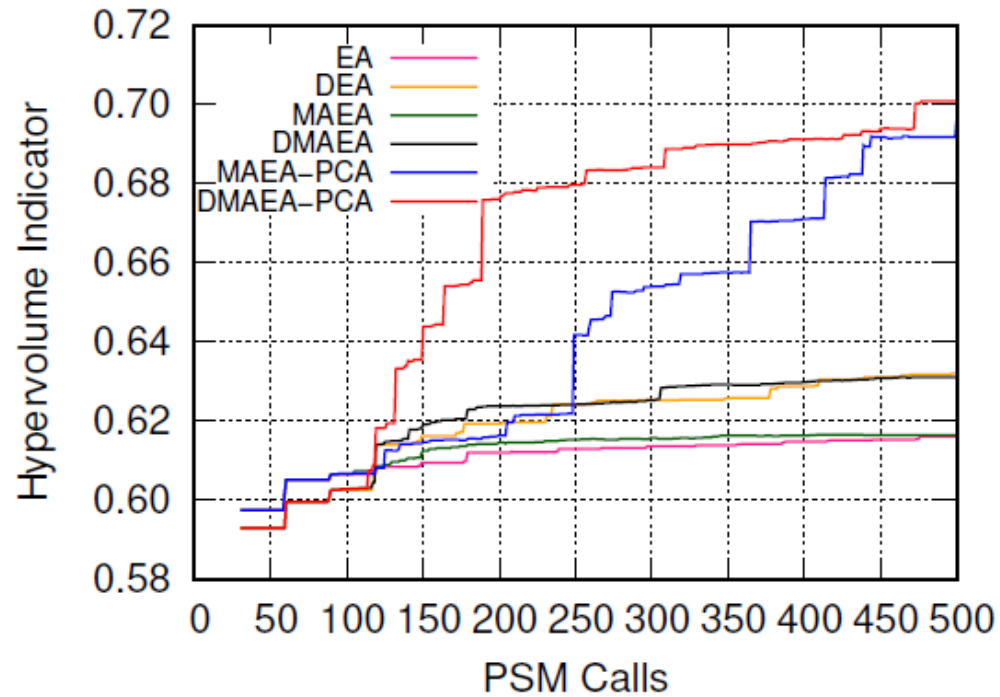
PSM: The PCOpt/NTUA GPU-enabled flow solver (PUMA).

Basis of Comparison: a (10,30)EA.

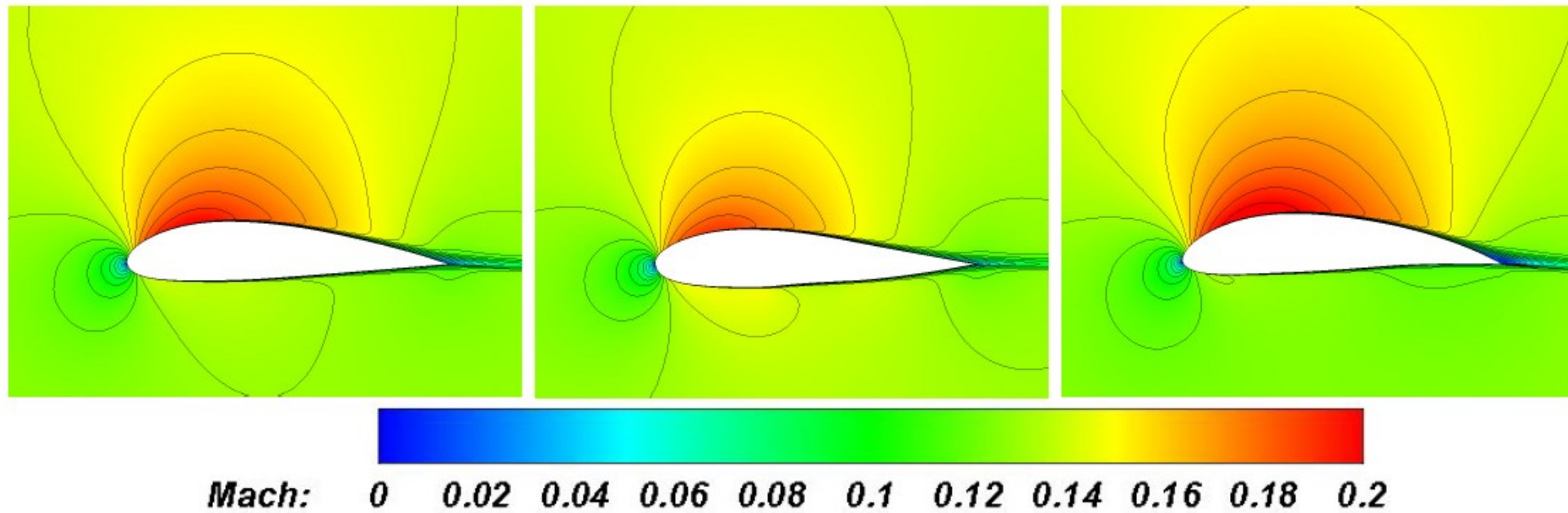
Cost per evaluation: ~ 1 min. on one NVIDIA A100 GPU.



Demo Case C (SOO)



Demo Case C (SOO)



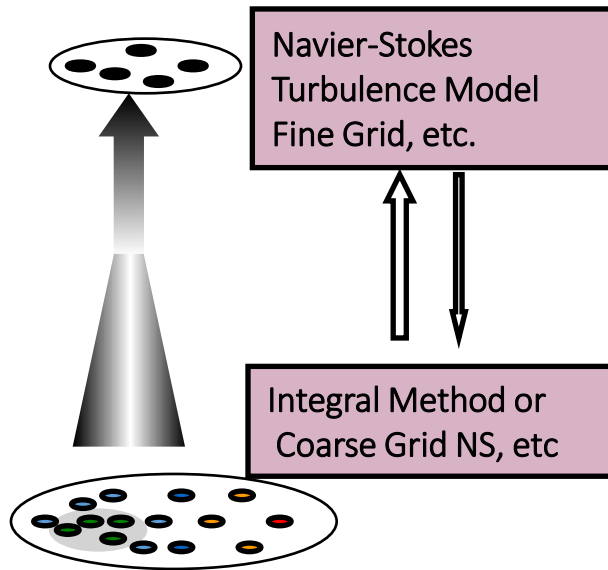
Reference shape

Optimized for min. C_D

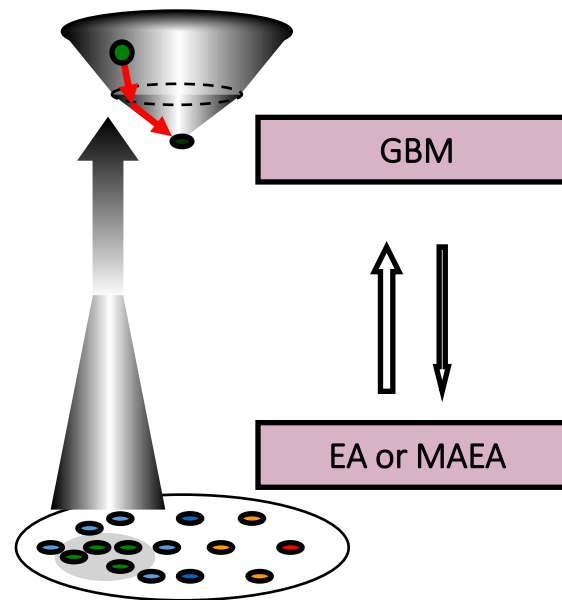
Optimized for max. C_L

Hierarchical/Multilevel Schemes

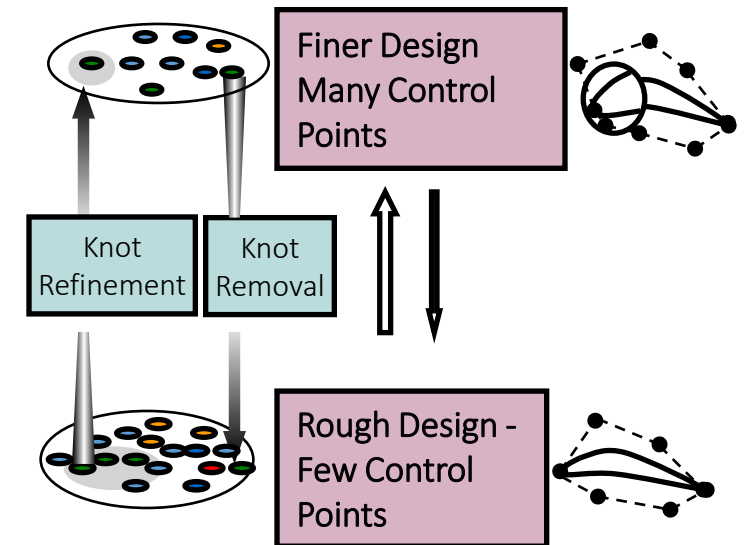
Hierarchical Evaluation



Hierarchical Search



Hierarchical Parameterization



Asynchronous EAs and MAEAs

Main Concept:

- Maximum exploitation of all available computational resources (processors)
- **Remove the generation barrier**
- Once a processor becomes idle, a new offspring is generated on-the-fly and its evaluation is assigned to this processor.

What about metamodels and Asynchronous EAs?

- Instead of generating a single new member to be evaluated, a few trial members are generated. For each trial member, a local metamodel is trained. The most promising new individual among the trial ones, according to the metamodel (used separately for all of them), is send to the idle processor.

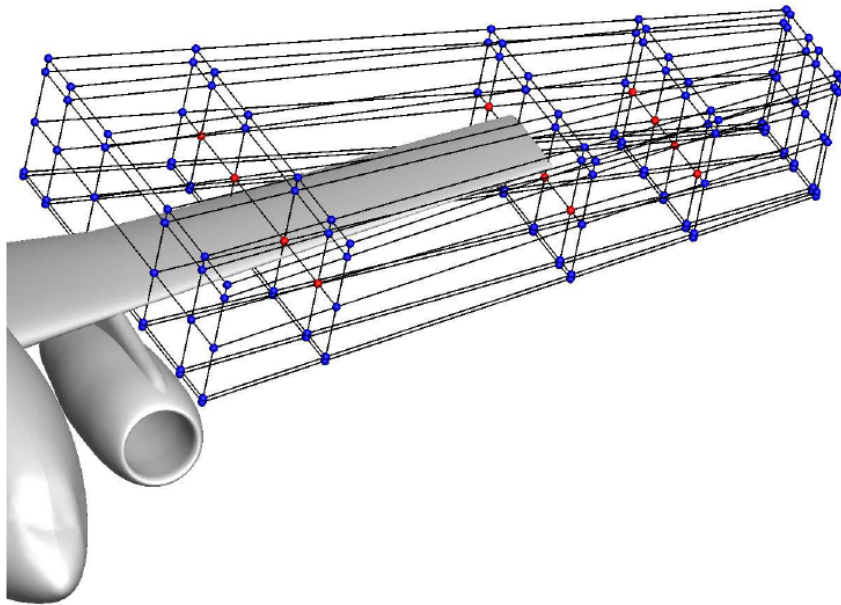
 *Engineering Optimization, 41:241-257, 2009.*

 *Genetic Programming and Evolvable Machines, 10:373:389, 2009.*

Dr. Varvara G. Asouti, vasouti@mail.ntua.gr

Part 4: Industrial Applications

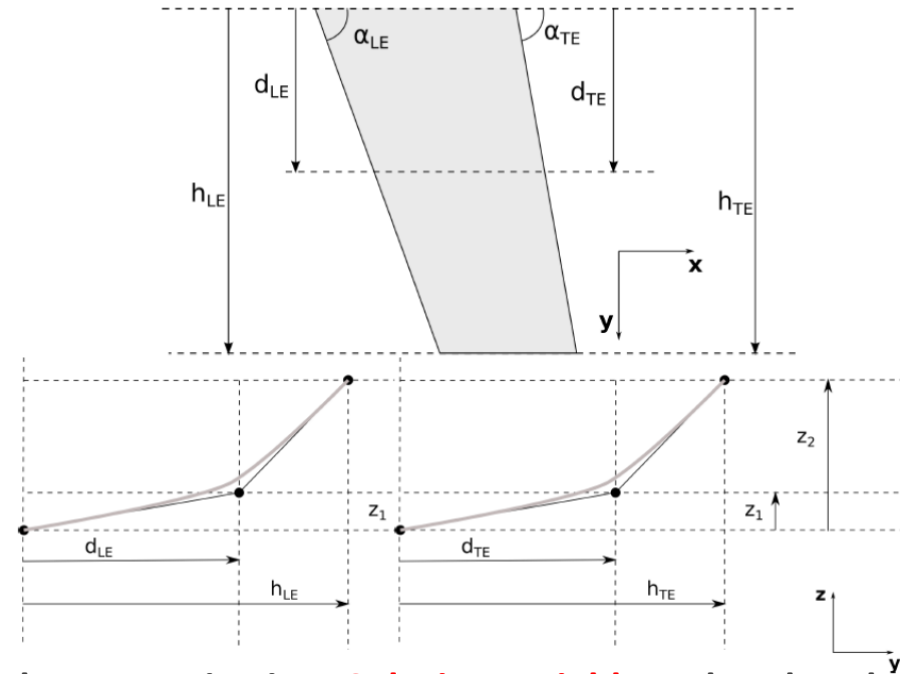
Optimization of an Aircraft Wing-Body Configuration



Re-design of the wing of an aircraft wing-body configuration, for max. C_L and min. C_D .

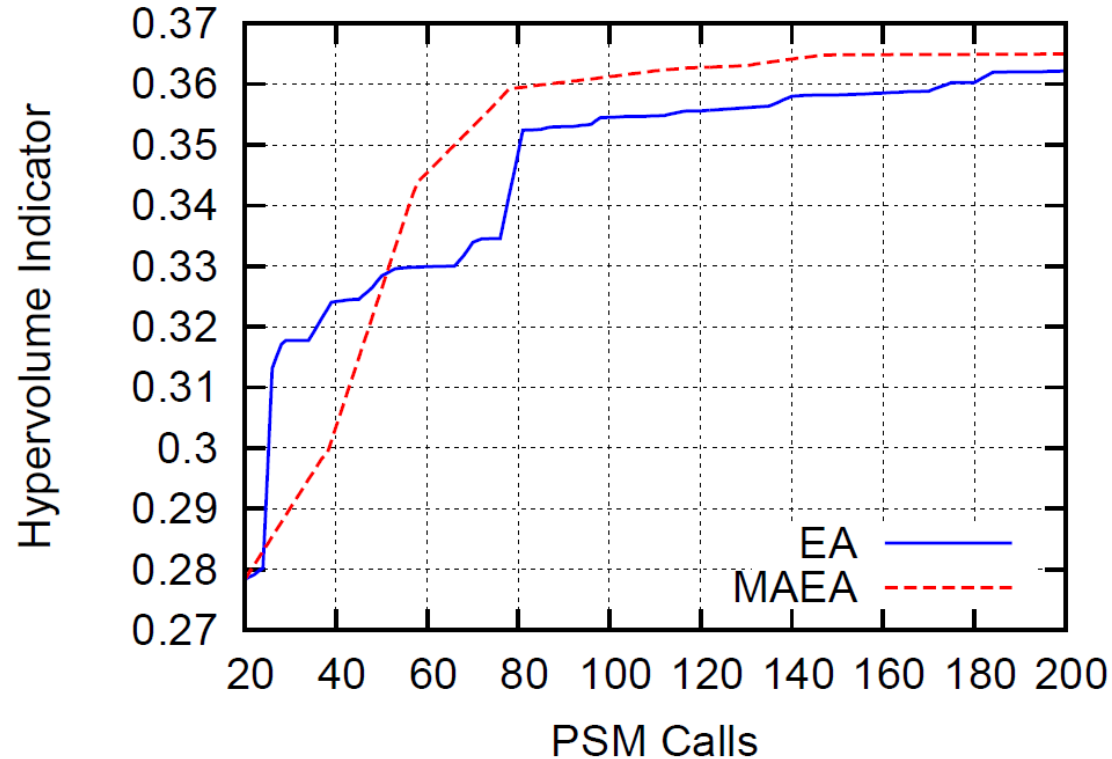
Turbulent flow: $Re_c=10^6$, $M_{inf}=0.75$, $\alpha_{inf}=0^\circ$.

These 8 design variables affect, in turn, the coordinates of some of the nodes of a $3 \times 5 \times 4$ NURBS control grid which undertakes the internal grid adaptation.



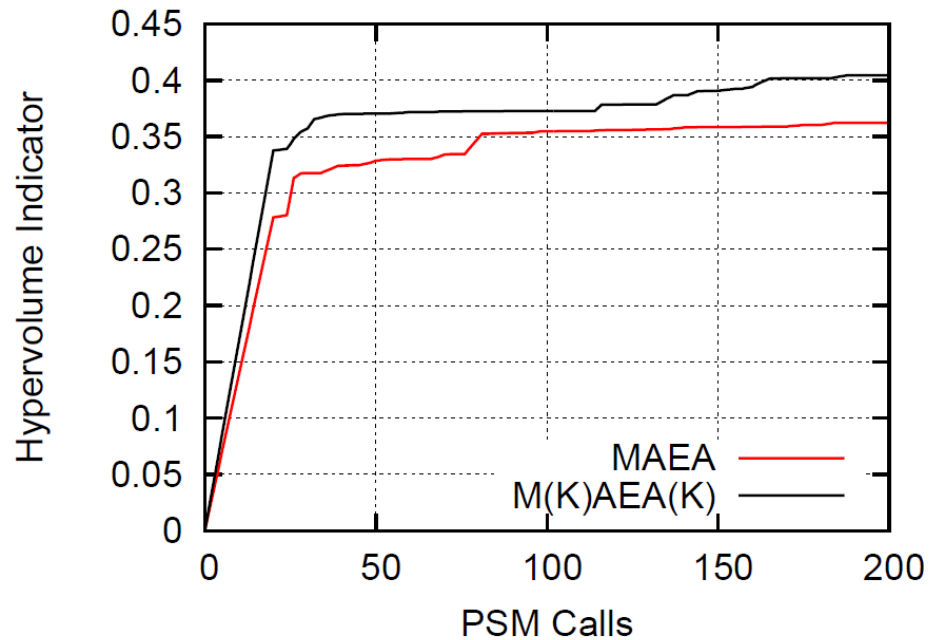
Customized parametrization: **8 design variables** related to the wing planform (top) & the wing dihedral angles at the leading & training edges (bottom).

Optimization of an Aircraft Wing-Body Configuration

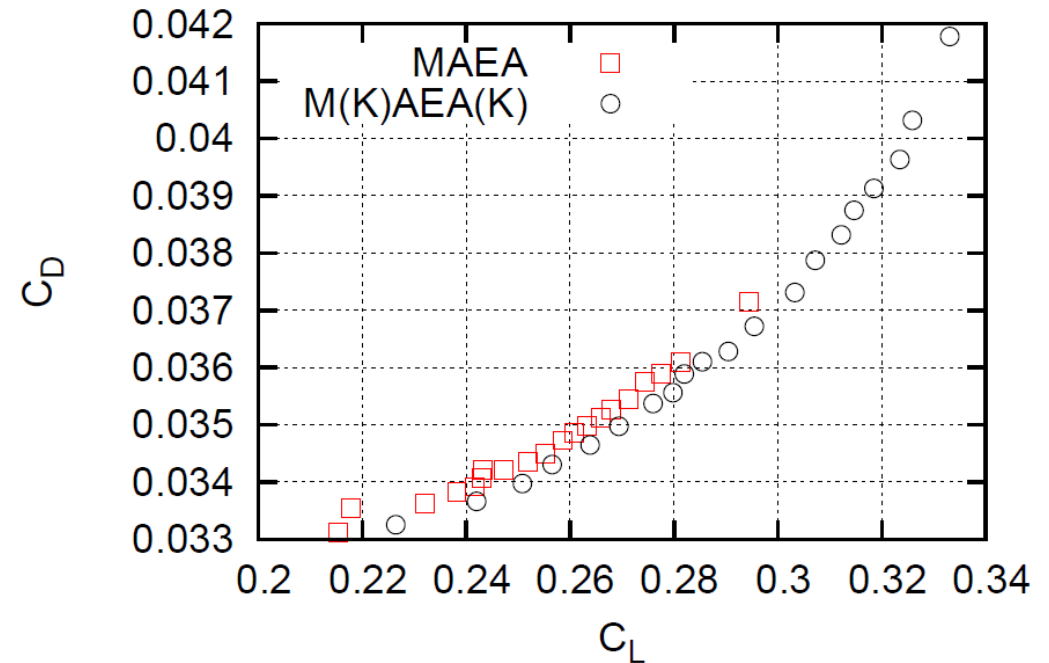


Comparison of the averaged convergence histories for **three RNG seeds** of the (5,10) EA and MAEA, in terms of the number of CFD evaluations (PSM Calls). The LCPE phase starts after the first $T^{MM}=20$ calls to the PSM and the $\lambda_e=4$ most "promising" individuals are re-evaluated in each generation. Stopping criterion = 200 CFD evaluations.

Optimization of an Aircraft Wing-Body Configuration

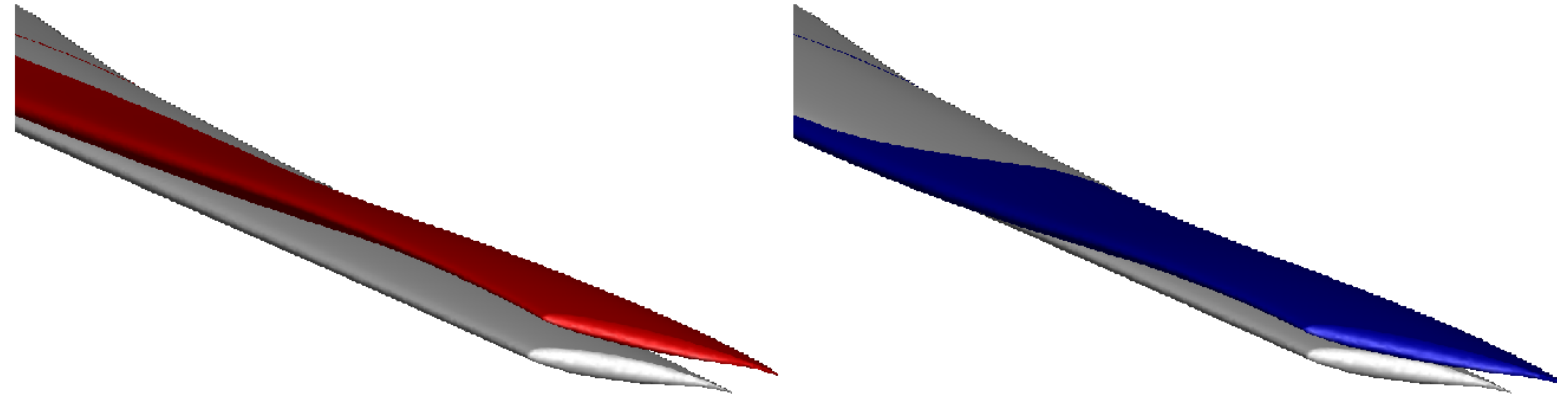


Comparison of the convergence histories (expressed by the hypervolume indicator) of the MAEA and M(K)AEA(K), in terms of the number of CFD evaluations (PSM Calls).

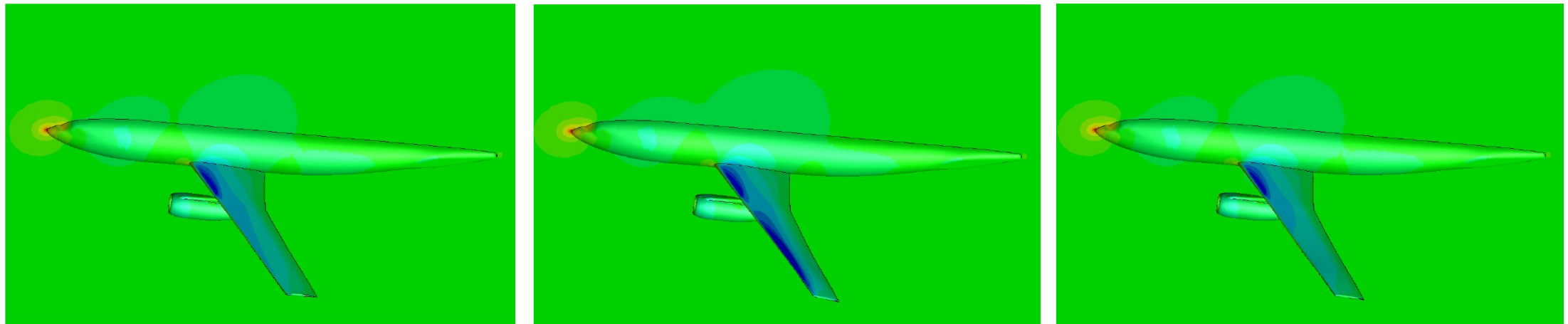


Comparison of the fronts of non-dominated solutions computed by the MAEA and M(K)AEA(K) (right). The M(K)AEA(K) computed an improved front, due to the wide spreading of the optimal solutions.

Optimization of an Aircraft Wing-Body Configuration



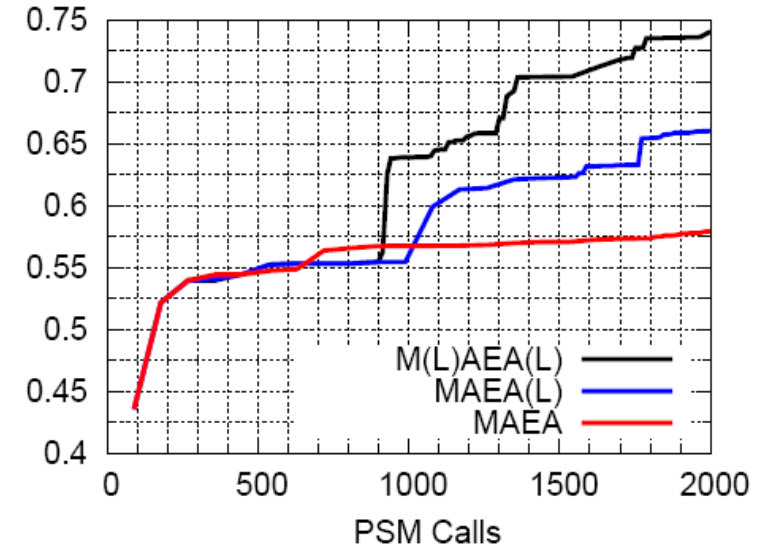
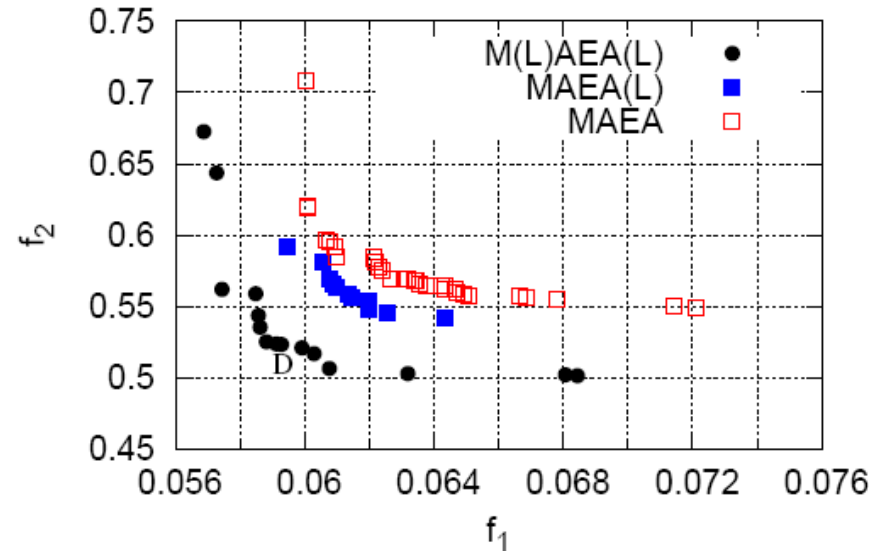
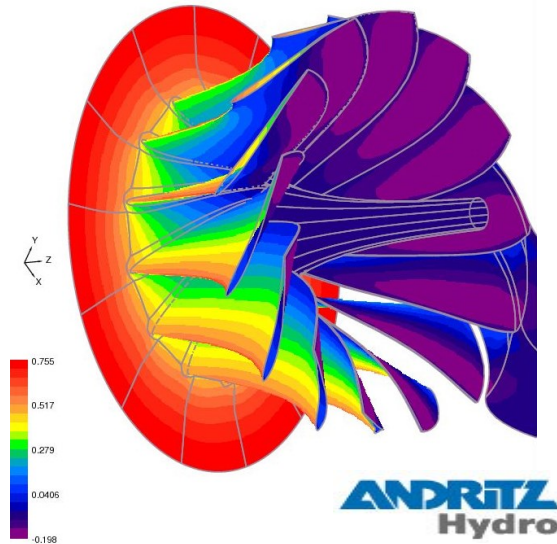
Comparison of the baseline configuration (in grey) and the max. C_L (in red) and min. C_D (in blue) ones.



Comparison of pressure distributions for different wing shapes: baseline configuration (left), max. C_L configuration (center) and min. C_D configuration (right).

Dr. Varvara G. Asouti, vasouti@mail.ntua.gr

Shape Optimization of a Francis Runner

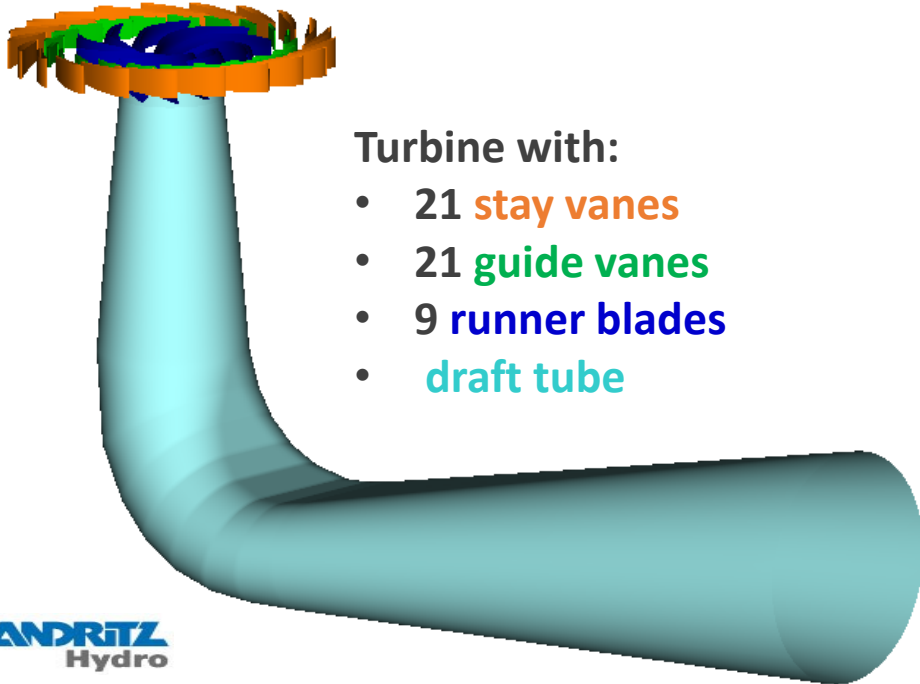


- Design of the Francis runner, at 3 operating points with two objectives: (a) exit velocity profiles' quality and (b) uniformity of the blade loading. Two constraints (head and cavitation). There are **372 DoFs**, in total!
- Comparison of fronts of non-dominated solutions obtained at the same number of evaluations on the PSM (same CPU cost).
- Due to the extremely high problem dimension, the use of M(L)AEA(L) becomes absolutely necessary!

Shape Optimization of a Hydraulic Turbine

Turbine with:

- 21 stay vanes
- 21 guide vanes
- 9 runner blades
- draft tube



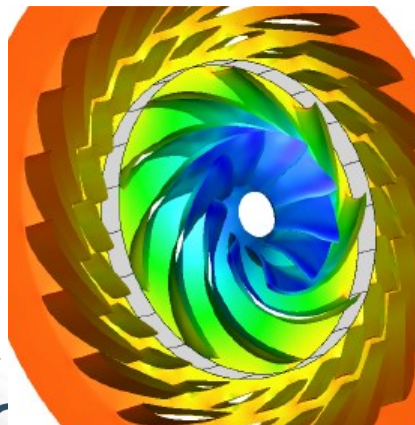
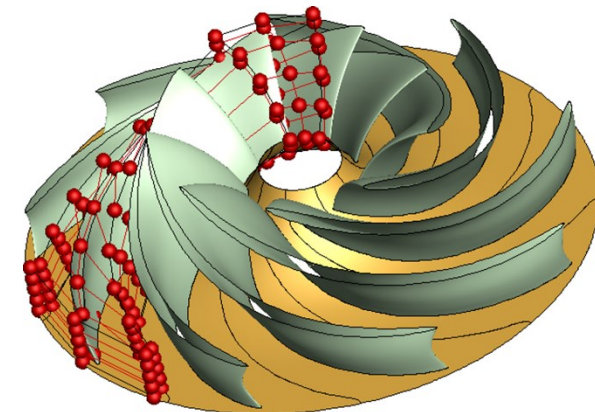
➤ Objective: $f_M = p_{max} - p_{min}$

the amplitude of the pressure field computed along a circumference located at the area between the guide vane and the runner)

➤ Constraints:

- Head (deviation from rated head less than 1.5%)
- Efficiency (greater than that of the baseline)
- Cavitation free (constraint on the min. pressure on the runner surface)

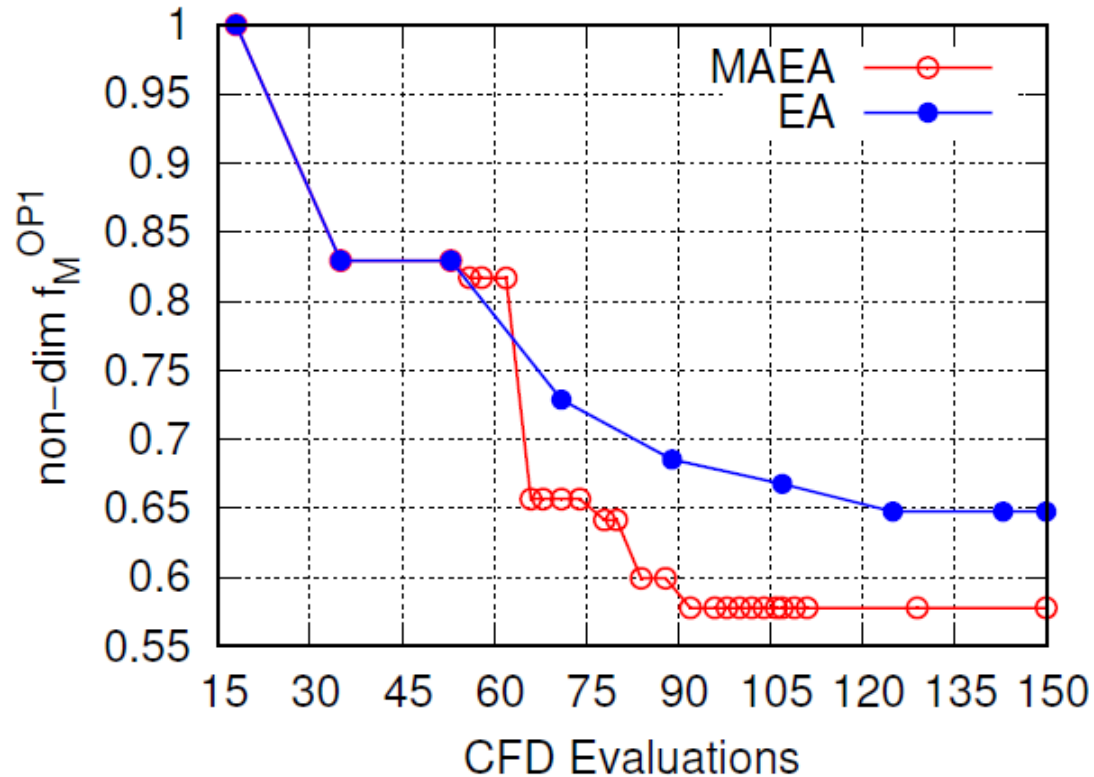
➤ Only the runner blade is altered during the optimization. This is controlled by a 11×3×5 volumetric NURBS lattice, with **180 design variables**.



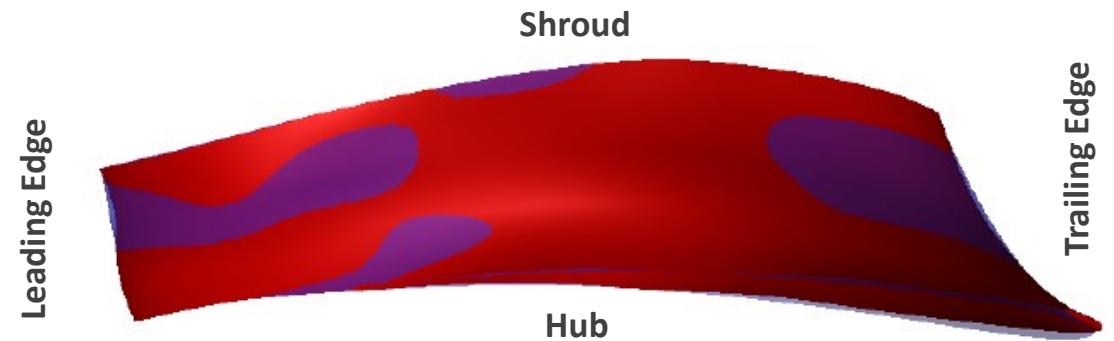
Shape Optimization of a Hydraulic Turbine

- $(\mu, \lambda) = (10, 18)$ MAEA:
 - $T^{MM} = 50$ entries in the DB before activating the LCPE.
 - $\lambda_e \in [2, 4]$ re-evaluated individuals.
 - PCA activated after the 2nd generation.
 - Metamodels use 40 inputs (40 first principal components, truncated after PCA).
- Computational budget: 150 CFD evaluations.
- PSM: The PCOpt/NTUA GPU-enabled flow solver (PUMA).
- Computational platform: A single node with 4 x A100 NVIDIA GPUs.

Shape Optimization of a Hydraulic Turbine



- ~43% reduction in f_M
- Constraints satisfied



View of the **baseline** and **optimized** runner blade.

Comparison of the convergence histories of the EA and the PCA-driven MAEA. The objective function value is non-dimensionalized with that of the baseline geometry.

Overview – Comments

- **The CFD-based constrained optimization of real-world industrial problems is computationally demanding by itself.**
- **Optimization turnaround time can be reduced thanks to the use of surrogate evaluation models (metamodels) and dimensionality reduction (such as through the PCA).**
- **Metamodels can also be used in distributed search, hierarchical schemes, asynchronous algorithms etc. The gain offered by using metamodels is superimposed to that of the other techniques.**