



# Evolutionary Multiobjective Optimization

Bhupinder Singh Saini

Postdoctoral Researcher

Faculty of Information Technology

[bhupinder.s.saini@jyu.fi](mailto:bhupinder.s.saini@jyu.fi)



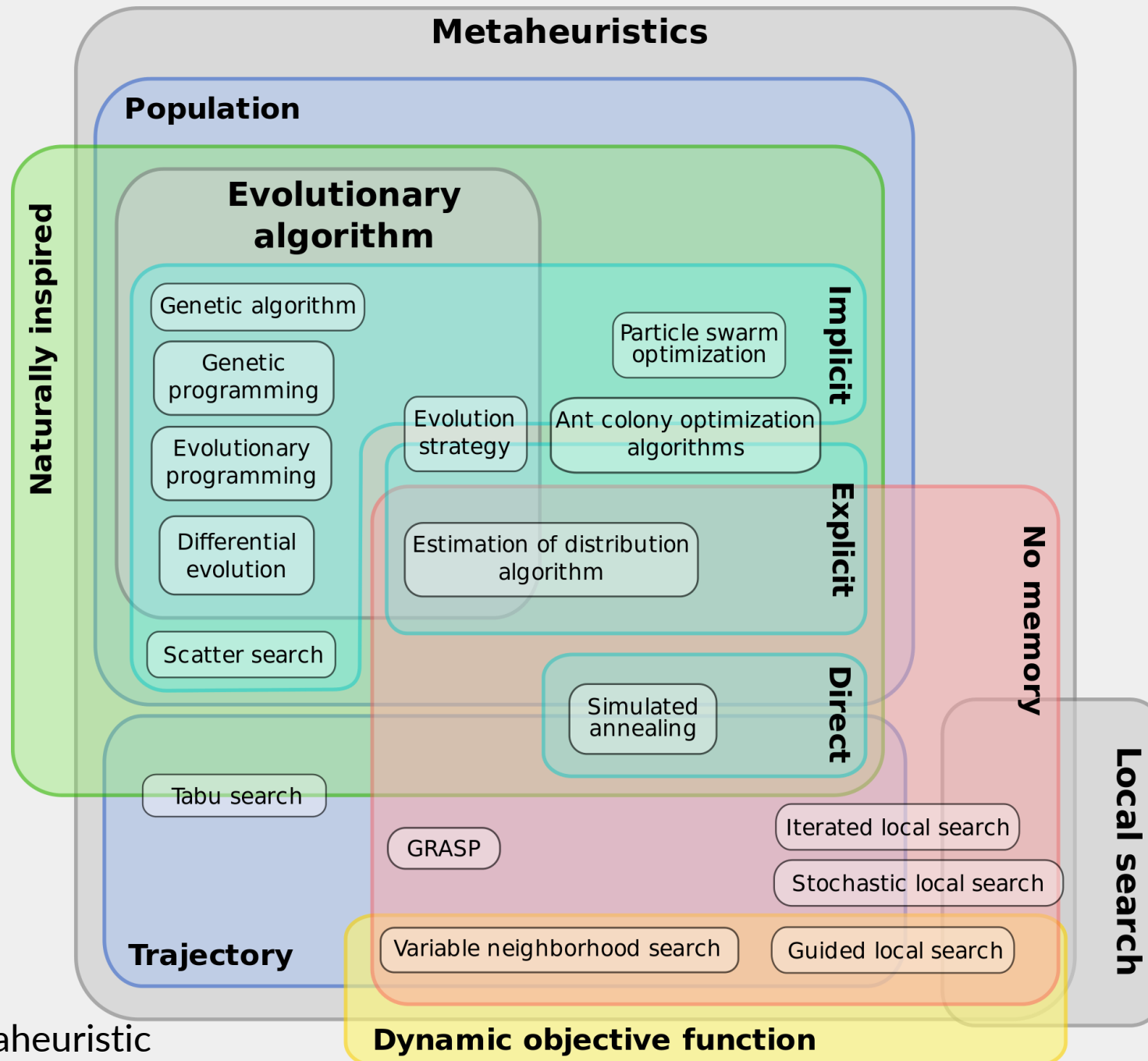
# Contents

- Brief introduction to Metaheuristics
- Introduction to Evolutionary Algorithms
- Evolutionary Multiobjective Optimization
- Interactive Evolutionary Multiobjective Optimization

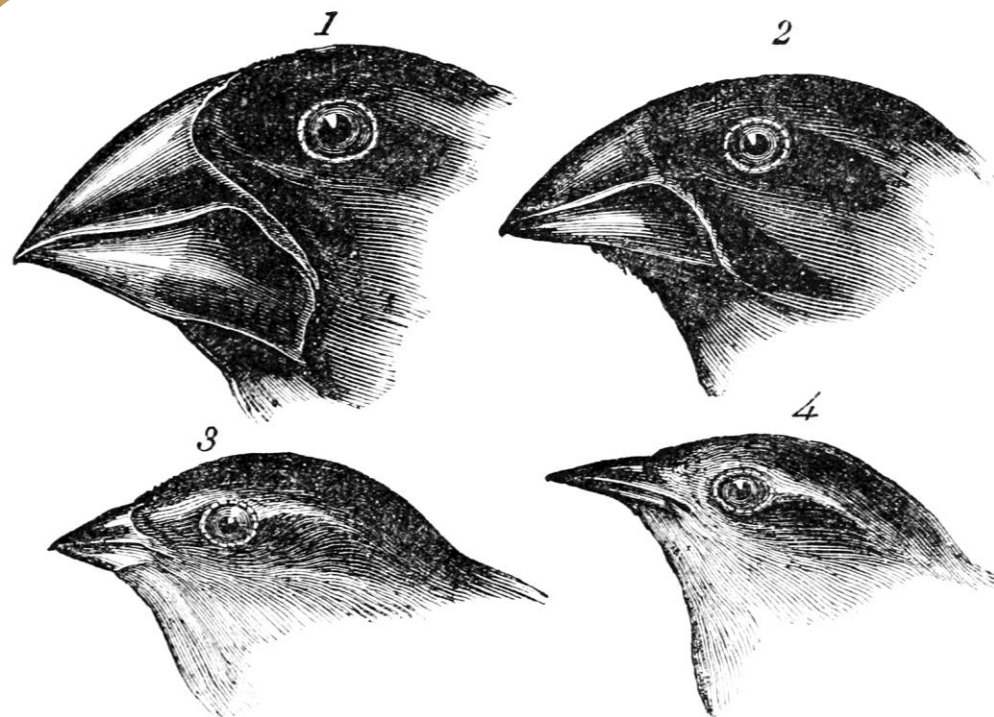


# Metaheuristics

- What's a heuristic?
  - A fast and approximate algorithm to solve a complex optimization/search problem
  - Optimality is not guaranteed
  - Used when classical algorithms are too inefficient for a given problem
- Metaheuristics:
  - A guiding strategy to generate heuristics
  - No assumptions about the specifics of the problem



# Evolutionary Algorithms



1. *Geospiza magnirostris*.  
3. *Geospiza parvula*.

2. *Geospiza fortis*.  
4. *Certhidea olivacea*.



# A basic EA procedure

- Create initial population (randomly)
- Repeat until termination criteria is met:
  - Recombination: Create new offsprings
  - Evaluation: Calculate fitness of all individuals
  - Selection: Kill unfit individuals

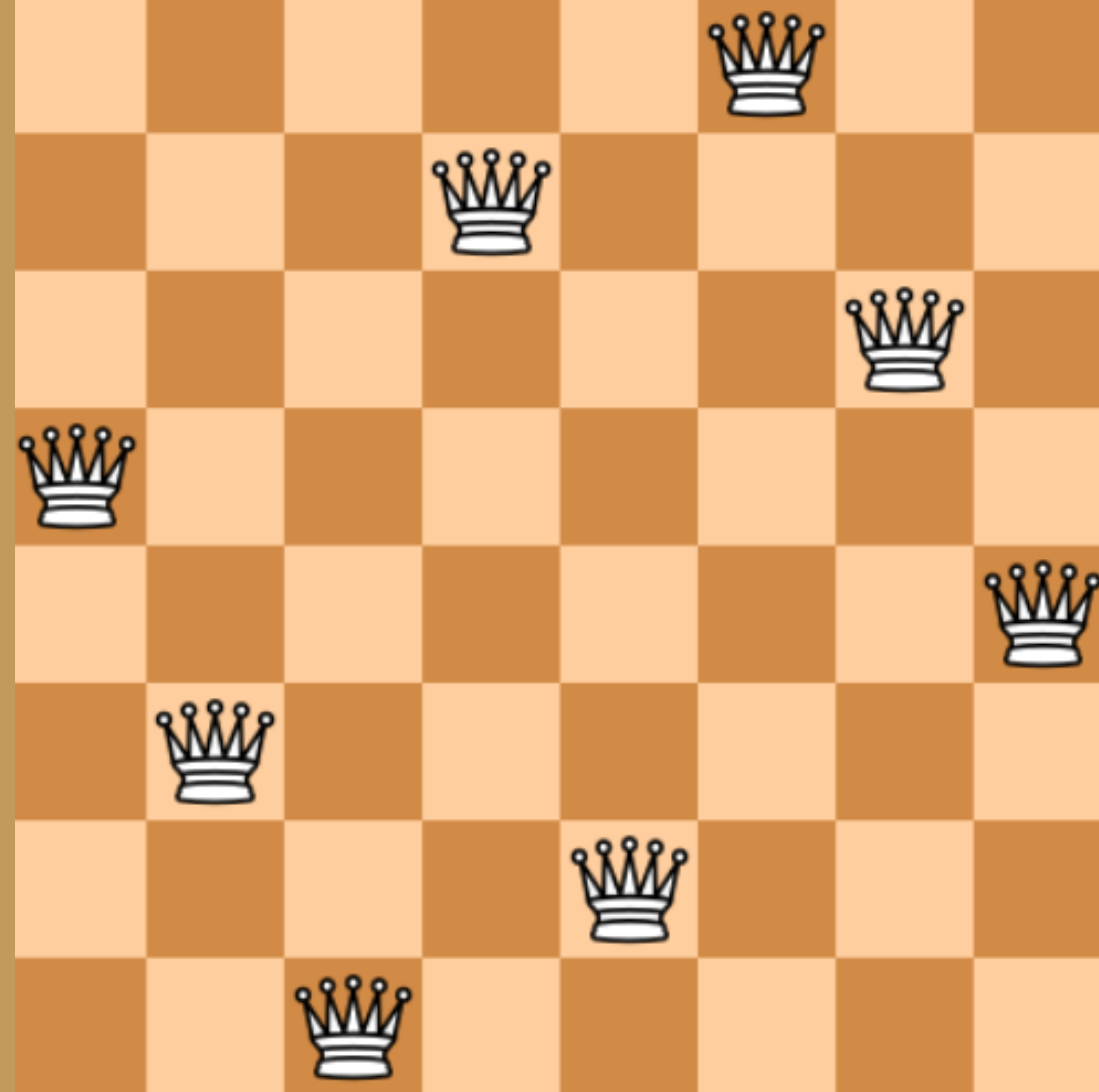
```
1  def template(...) -> Result:
2      solutions, outputs = generate()
3      while not terminate():
4          offspring = crossover(population=solutions)
5          offspring = mutate(offspring, solutions)
6          offspring_outputs = evaluate(offspring)
7          solutions, outputs = select(
8              parents=(solutions, outputs),
9              offsprings=(offspring, offspring_outputs)
10         )
11     return Result(solutions=solutions, outputs=outputs)
```



## Related terms

- Individuals: Solutions, Candidates
- Chromosomes: Genotype, decision vector
- Fitness: Phenotype, objective vectors (not always accurate)
- Recombination: How parents create offspring
  - Crossover operator: “Mix” the parents together
  - Mutation operator: Add random variations
- Selection operator: How the next generation is chosen. ``Survival of the fittest´´.

# An Example: The 8 Queens puzzle







# Let's define the problem

- Objective: Minimize the number of Queens that “see” each other.
- Optimal solution: Objective function must reach a value of zero.
- How to represent a solution (decision space)? -> “Encoding”
  - Chess notation?
  - List of queen coordinates? (a5, d7, b3...)
  - Sparse 8x8 matrix (each matrix element represents a square on the board, 0 represents empty squares, 1 represents squares with a queen)
- Why is this important?
  - Different representations of the same problem may be easier or more difficult to solve.
  - Crossover and mutation operators are strongly linked with how you represent a solution.



# Let's define the problem

- Objective: Minimize the number of Queens that “see” each other.
- Optimal solution: Objective function must reach a value of zero.
- How to represent a solution (decision space)? -> “Encoding”
  - Chess notation?
  - List of queen coordinates? (a5, d7, b3...)
  - Sparse 8x8 matrix (each matrix element represents a square on the board, 0 represents empty squares, 1 represents squares with a queen)
- Why is this important?
  - Different representations of the same problem may be easier or more difficult to solve.
  - Crossover and mutation operators are strongly linked with how you represent a solution.
- Let's represent a solution as a list of 8 ranks (e.g., [5, 3, 1, 7, 2, 8, 6, 4]), where each element corresponds to a unique file ([a, b, ..., h]) in order.



# Some crossover operators

- One point crossover: Take two solutions, choose an index (randomly). Swap all decision variable values after that index.

A	B	C	D	E	F	G	H
6	5	5	2	6	3	1	6
5	2	7	4	3	1	3	2

A	B	C	D	E	F	G	H
6	5	5	4	3	1	3	2
5	2	7	2	6	3	1	6



# Some crossover operators

- One-point crossover: Take two solutions, choose an index randomly. Swap all decision variable values after that index.
- Two-point crossover: Take two solutions, choose two indices randomly. Swap all decision variable values in between the indices.

A	B	C	D	E	F	G	H
6	5	5	2	6	3	1	6
5	2	7	4	3	1	3	2

A	B	C	D	E	F	G	H
6	5	5	4	3	1	1	6
5	2	7	2	6	3	3	2



# Some crossover operators

- One-point crossover: Take two solutions, choose an index randomly. Swap all decision variable values after that index.
- Two-point crossover: Take two solutions, choose two indices randomly. Swap all decision variable values in between the indices.
- Uniform crossover: Take two solutions, generate a random mask vector of the same number of dimensions. Swap decision variables according to the mask

A	B	C	D	E	F	G	H
6	5	5	2	6	3	1	6
5	2	7	4	3	1	3	2

Mask = [1, 0, 0, 1, 0, 1, 0, 0]

A	B	C	D	E	F	G	H
5	5	5	4	6	1	1	6
6	2	7	2	3	3	3	2



# Some crossover operators

- One-point crossover: Take two solutions, choose an index randomly. Swap all decision variable values after that index.
- Two-point crossover: Take two solutions, choose two indices randomly. Swap all decision variable values in between the indices.
- Uniform crossover: Take two solutions, generate a random mask vector of the same number of dimensions. Swap decision variables according to the mask
- Some other types: Linear crossover, blend crossover, simulated binary crossover



# Simulated Binary crossover (for real-coded problems)

- Simulates the behaviour of one-point crossover, but for real-valued decision variables.
- Able to deal with multi-modal problems well.
- The distribution of the offsprings around the parents can be controlled.

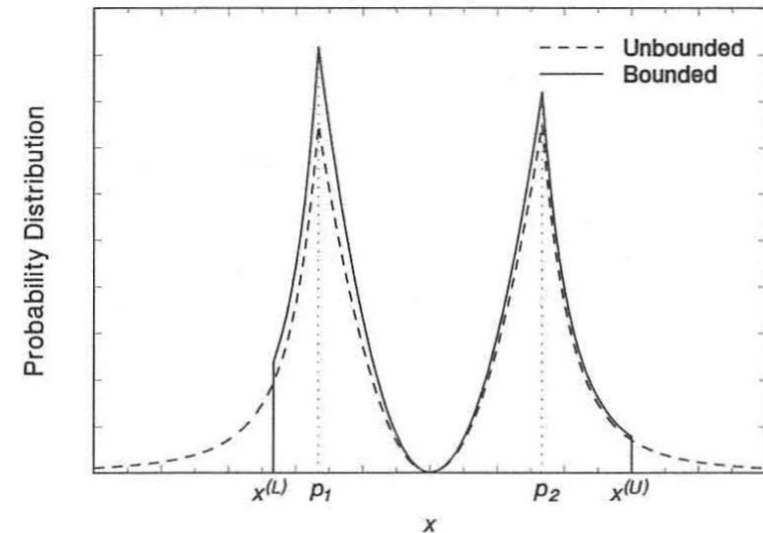


Figure 2: Probability distributions are shown for bounded and unbounded cases.

Deb, K. and Kumar, A. (1995). Real-coded genetic algorithms with simulated binary crossover: Studies on multi-modal and multi-objective problems. *Complex Systems*, 9(6), 431–454.



# Mutation

- Add random changes to a solution
- Helps in increasing diversity in the population
- E.g. Random mutation, one-point mutation, polynomial mutation (real coded)

A	B	C	D	E	F	G	H
6	6	6	8	8	5	1	1

A	B	C	D	E	F	G	H
6	6	6	8	8	5	3	1

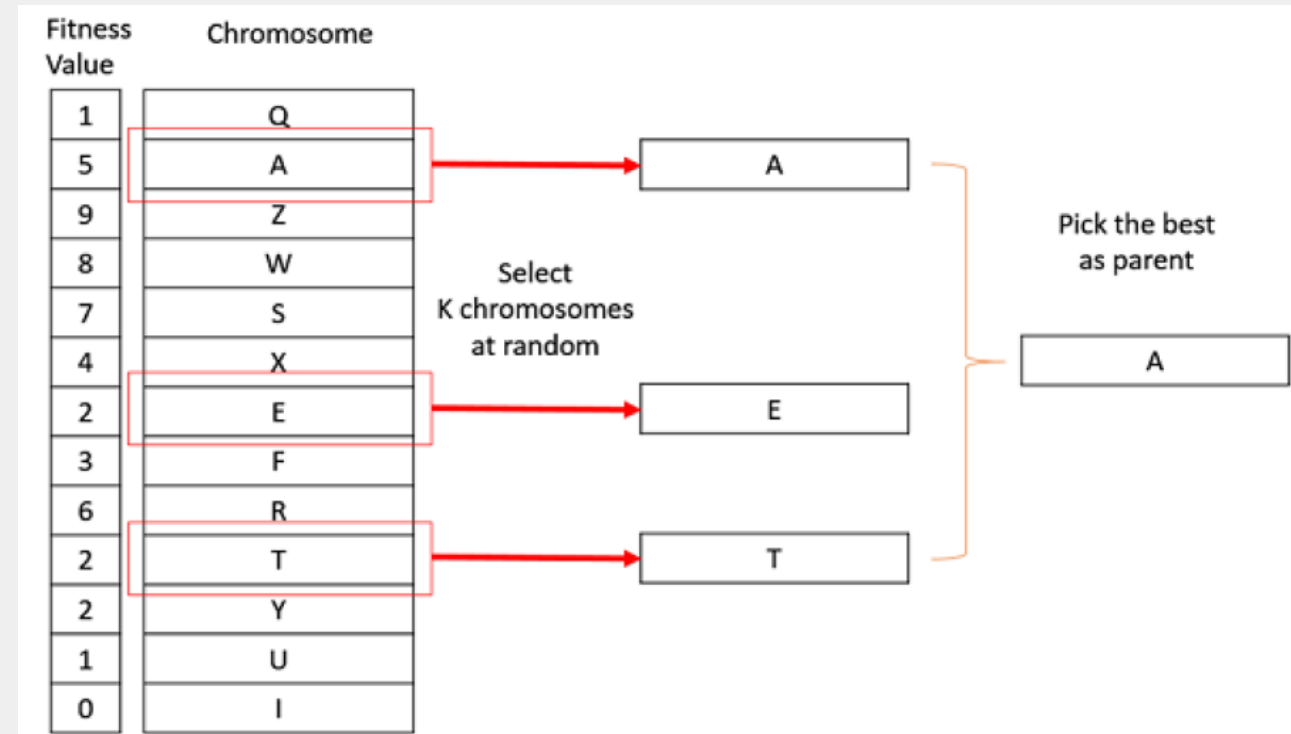
A	B	C	D	E	F	G	H
6	6	6	8	1	5	8	1





# Selection

- Apply “selection pressure” to make the population better over generations.
- E.g. Tournament selection:
  - “K” individuals from the population are selected randomly.
  - The best one out of these is selected for the next generation.
  - Repeat until appropriate number of individuals have been chosen.
- Other examples: Roulette wheel selection, steady state selection, etc

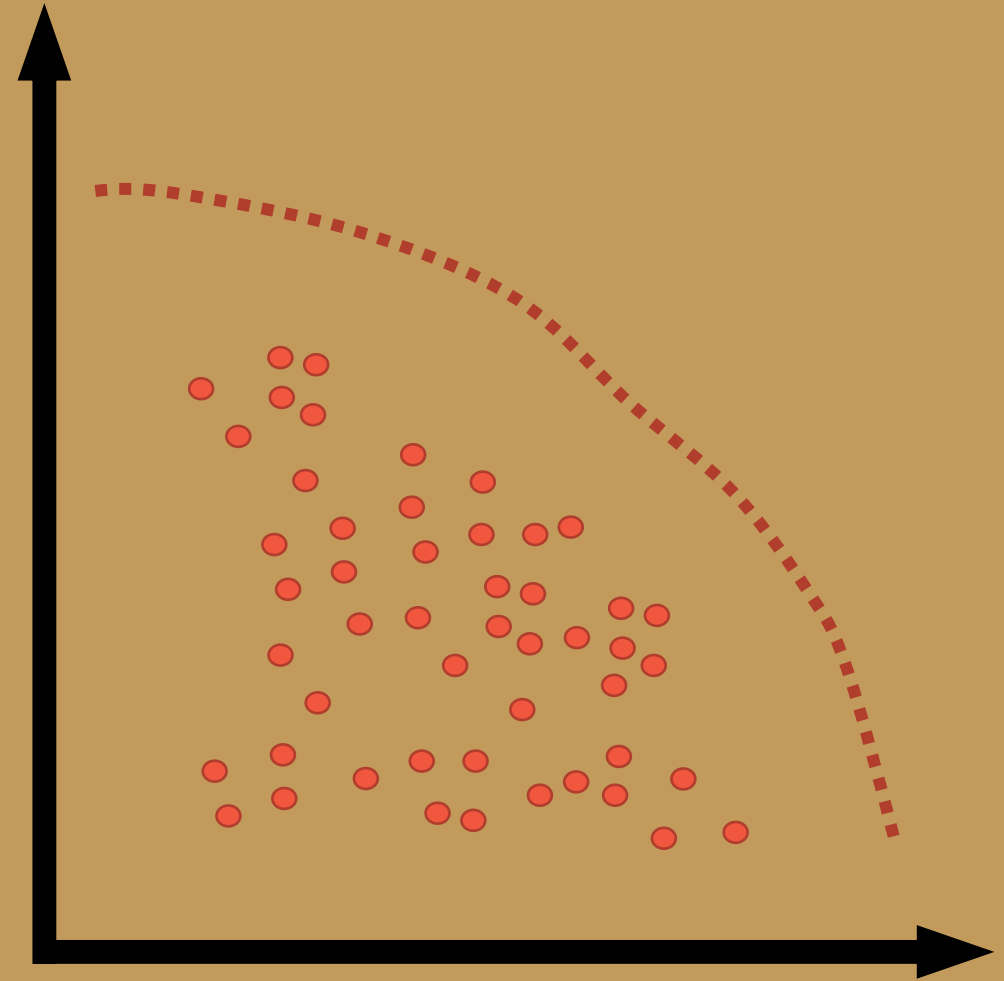




# Issues with EAs

- Optimality is not guaranteed:
  - Diversity: An EA needs to maintain a diverse population (“gene pool”) to have a good chance at reaching the global optima.
  - Convergence: The ability of the EA to reach the global optima quickly.
  - Balancing diversity (by keeping potentially worse solutions alive) and convergence (by applying selection pressure) is crucial!
- How to tackle multiobjective optimization problems??

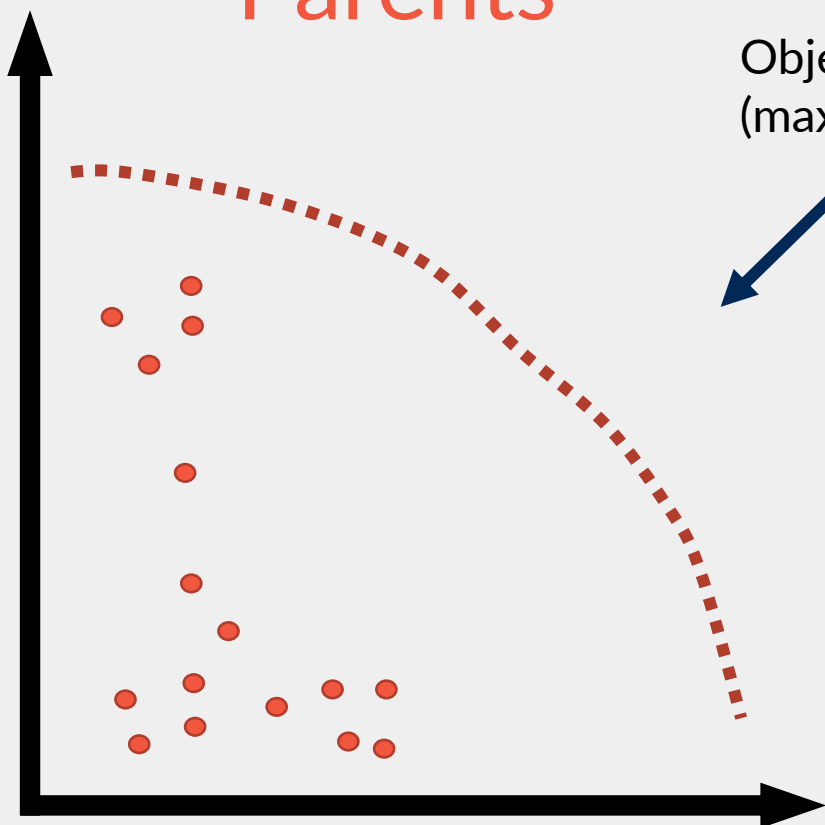
# Multiobjective Evolutionary Algorithms (MOEAs)



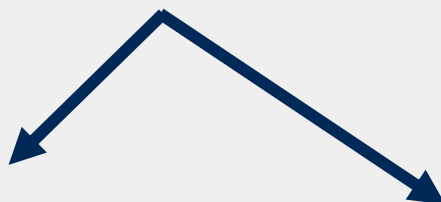


# MOEA

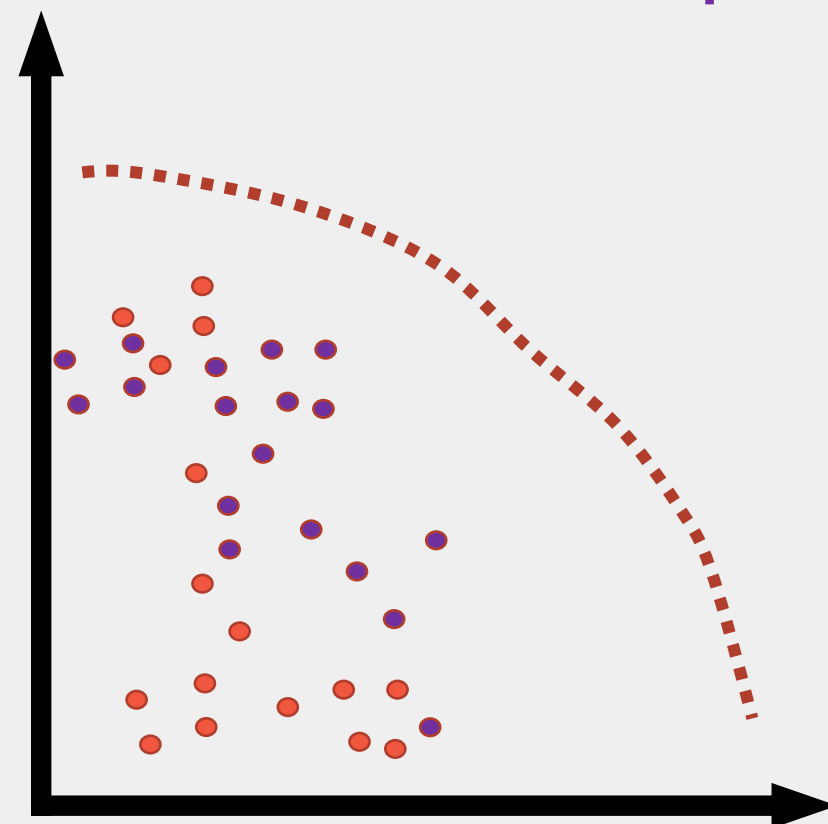
Parents



Objective space  
(maximization)



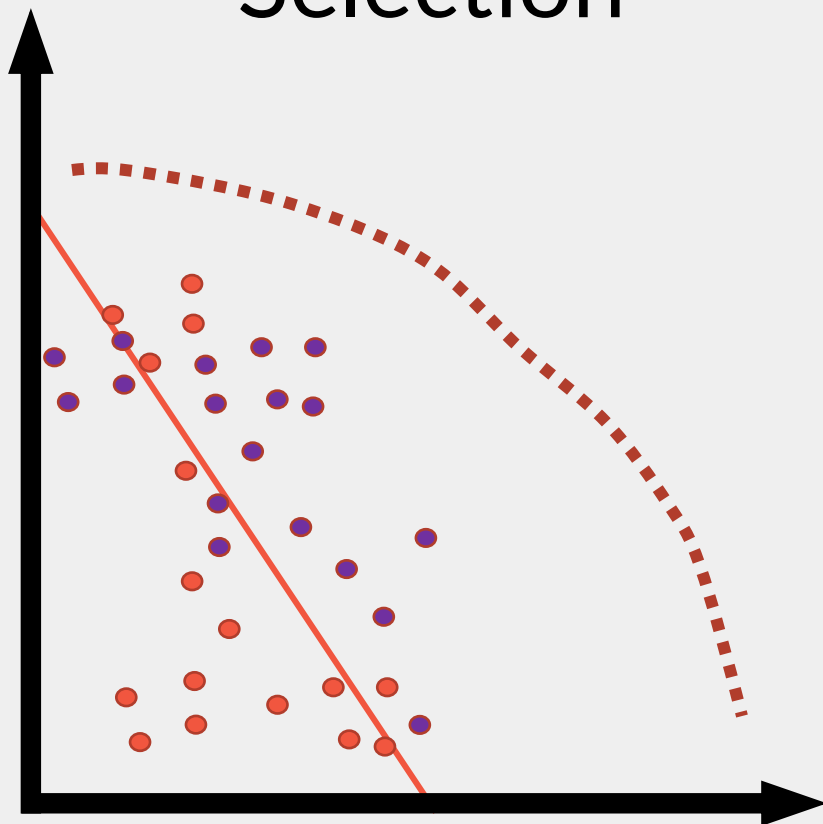
Parents + Offspring



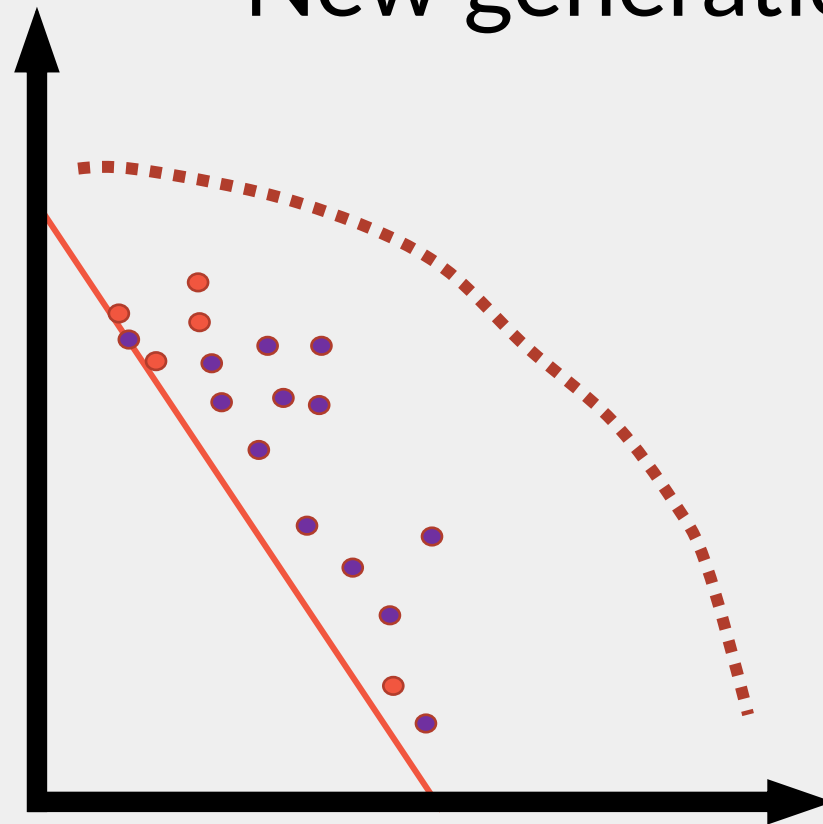


# MOEA

## Selection



## New generation



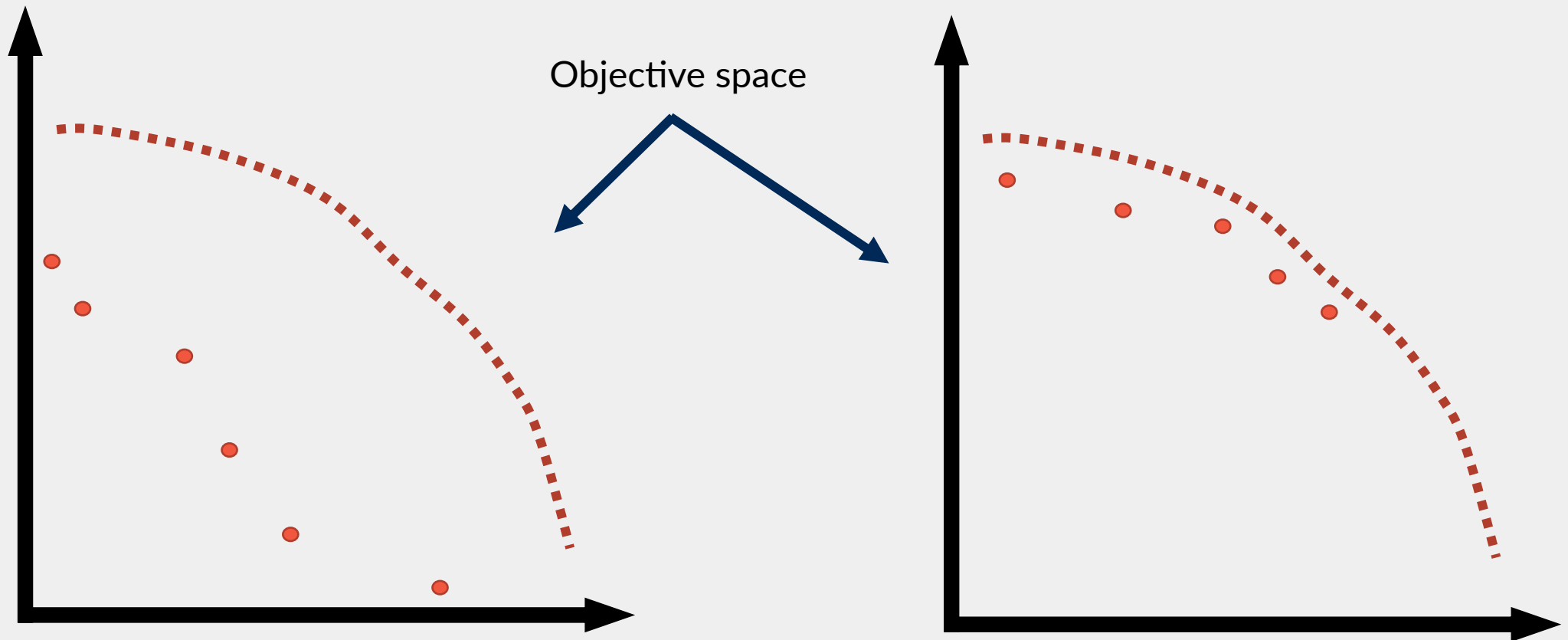


# MOEA operators

- Crossover and mutation
  - These operators are applied in the decision space
  - We can use the same operators as single objective EAs
  - Depends on the encoding used for the decision vectors
  - Popular crossover operator for MOEAs: Simulated Binary Crossover (SBX)
  - Popular mutation operator for MOEAs: Bounded Polynomial Mutation
- Selection:
  - New operators are needed.

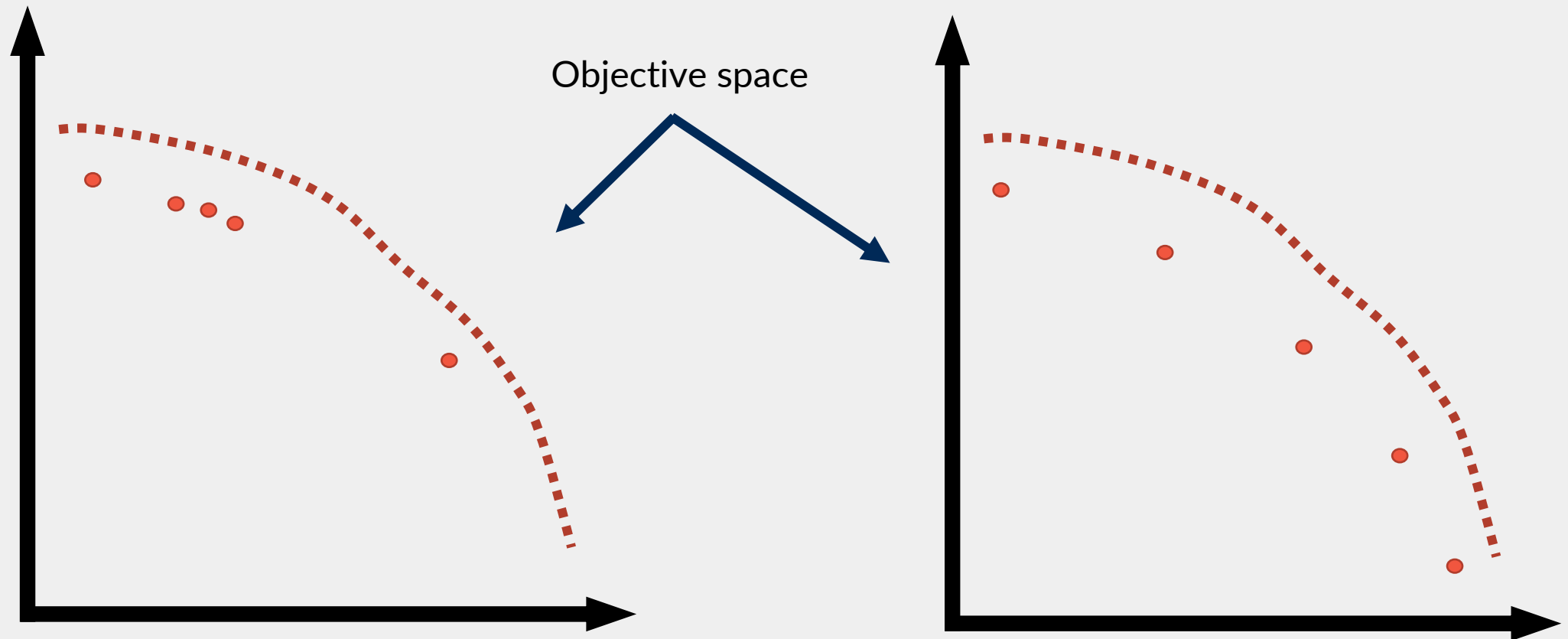


# Goals: Convergence





# Goals: Diversity (uniformity and spread)







## Goals: Quality indicators

- Generally used for evaluating performance of MOEAs on benchmark problems with known Pareto fronts.
- Convergence: Distance from a “reference set” (e.g., known Pareto optimal solutions)
- Diversity: Uniformity and spread of solutions
- Combination of convergence and diversity: Hypervolume, Inverted generational distance
- Preference satisfaction: R-metrics



# Types of MOEAs

- Based on the selection strategy, MOEAs can be divided into three classes:
  - Dominance-based MOEAs: Selection is based on Pareto dominance
  - Indicator-based MOEAs: Selection is based on improving indicator values
  - Decomposition-based MOEAs: Problem is decomposed into multiple simpler (e.g., single objective) problems and solved simultaneously.
- Not all MOEAs fit neatly into these categories and can employ multiple strategies simultaneously!

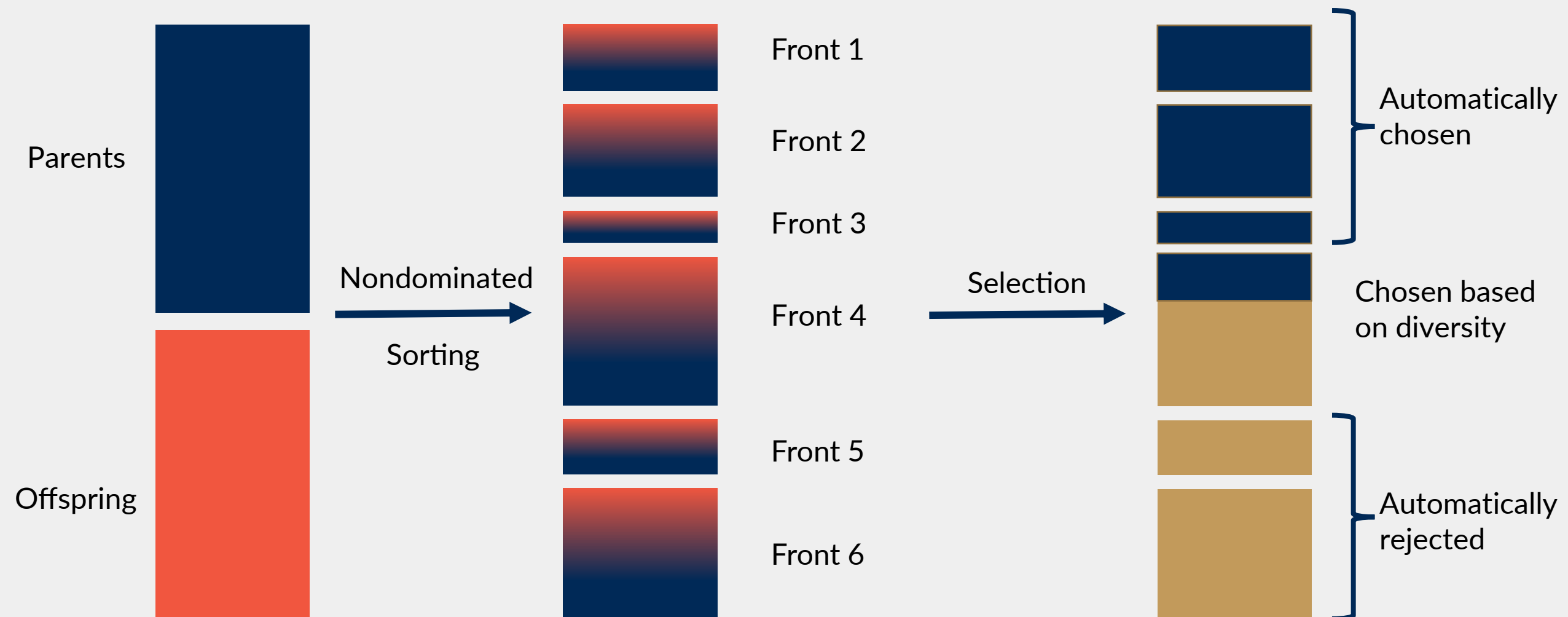


# Dominance-based MOEAs

- Fitness is based on the concept of Pareto dominance.
- First proposed by Goldberg in 1989.
- Ways to define “fitness”:
  - Dominance rank: Number of individuals that dominate the current individual (MOGA, SPEA2)
  - Dominance count: Number of individuals dominated by the current individual (SPEA2)
  - Dominance depth: NSGA-II



# NSGA-II selection





# Issues with dominance-based EAs

- Dominance comparisons (and especially non-dominated sorting) is a very expensive process.
- In problems with an increasing number of objectives, most solutions become mutually non-dominating.
  - Without the selection pressure from Pareto dominance, the MOEA fails to converge.

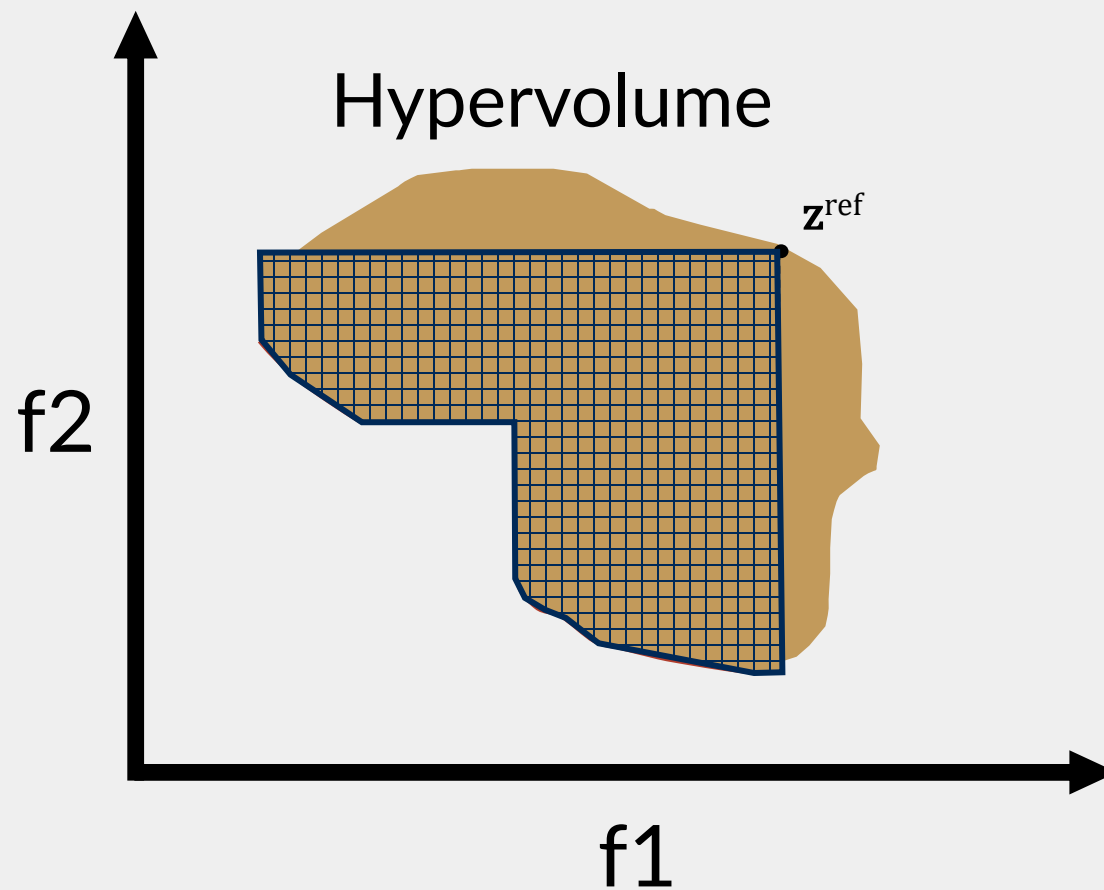
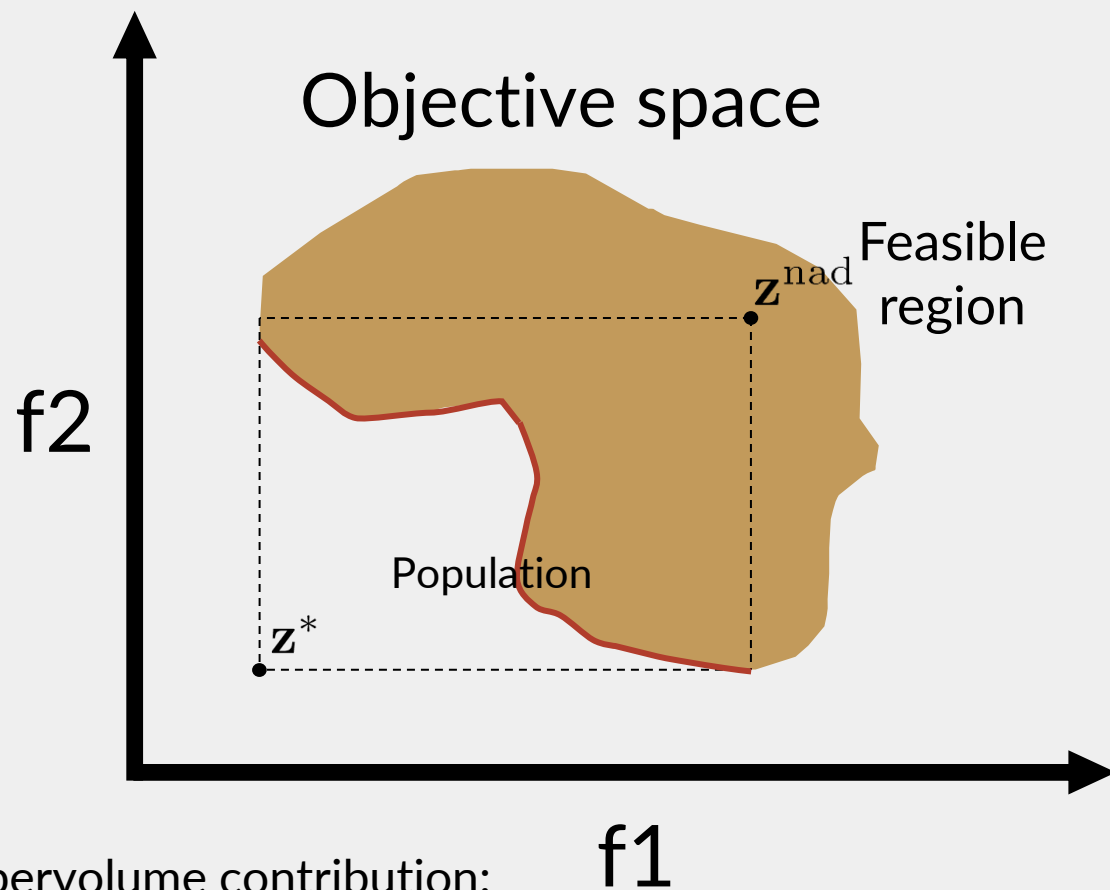


# Indicator-based MOEAs

- Why not use an indicator value as the fitness to be optimized?
- Some indicators, such as hypervolume, do not require a “reference set”.
- Most indicators judge the “goodness” of a population rather than an indicator.
- Thus, we need to calculate each individual’s contributions to the overall indicator value.
- Example: IBEA, SMS-EMOA



# Hypervolume: Area (volume) dominated by the population up until a reference point



Hypervolume contribution:

f1

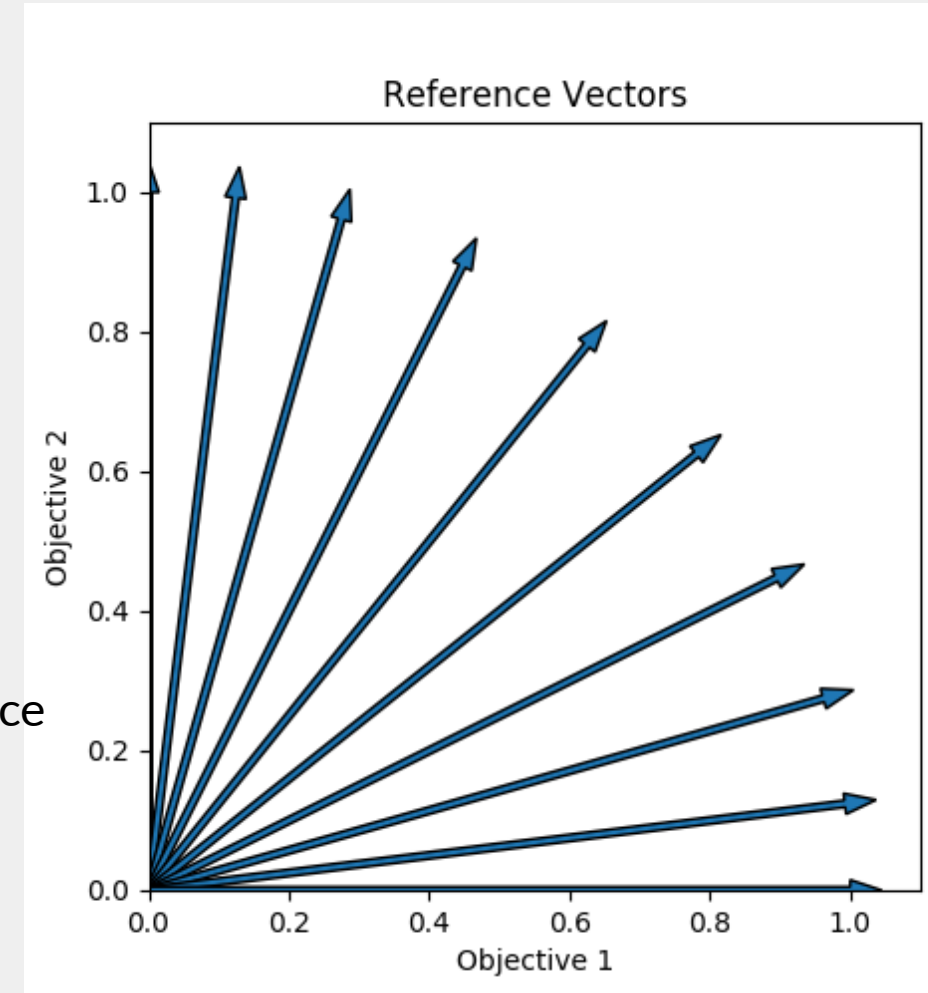
- What would the hypervolume be if the current individual was not a part of the population?
- Subtract total hypervolume by the value calculated above.

Thiele, L., Miettinen, K., Korhonen, P. J., & Molina, J. (2009). A preference-based evolutionary algorithm for multi-objective optimization. *Evolutionary computation*, 17(3), 411-436.



# Decomposition-based MOEAs

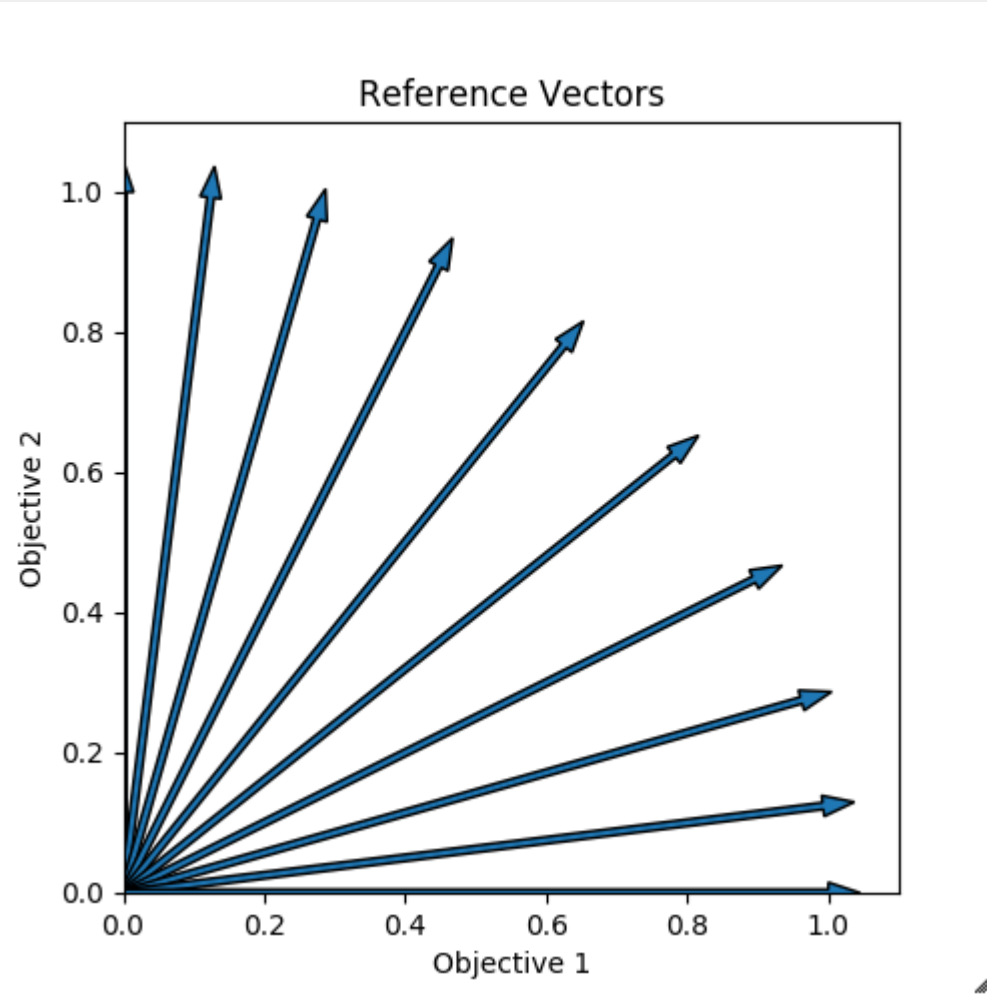
- Decomposition:
  - Split the multiobjective optimization problem into multiple simpler (perhaps even single objective) problems
  - with the use of directional vectors (and scalarization functions).
  - Divide the population into neighbourhoods.
  - Conduct selection only inside neighbourhoods.
  - Each neighbourhood converges along the vector.
  - Diversity is maintained by uniformly generating the reference vectors.
- Synonyms: Reference vectors, reference directions, reference points.
- MOEA/D, RVEA, NSGA-III







# Decomposition based MOEAs (RVEA)



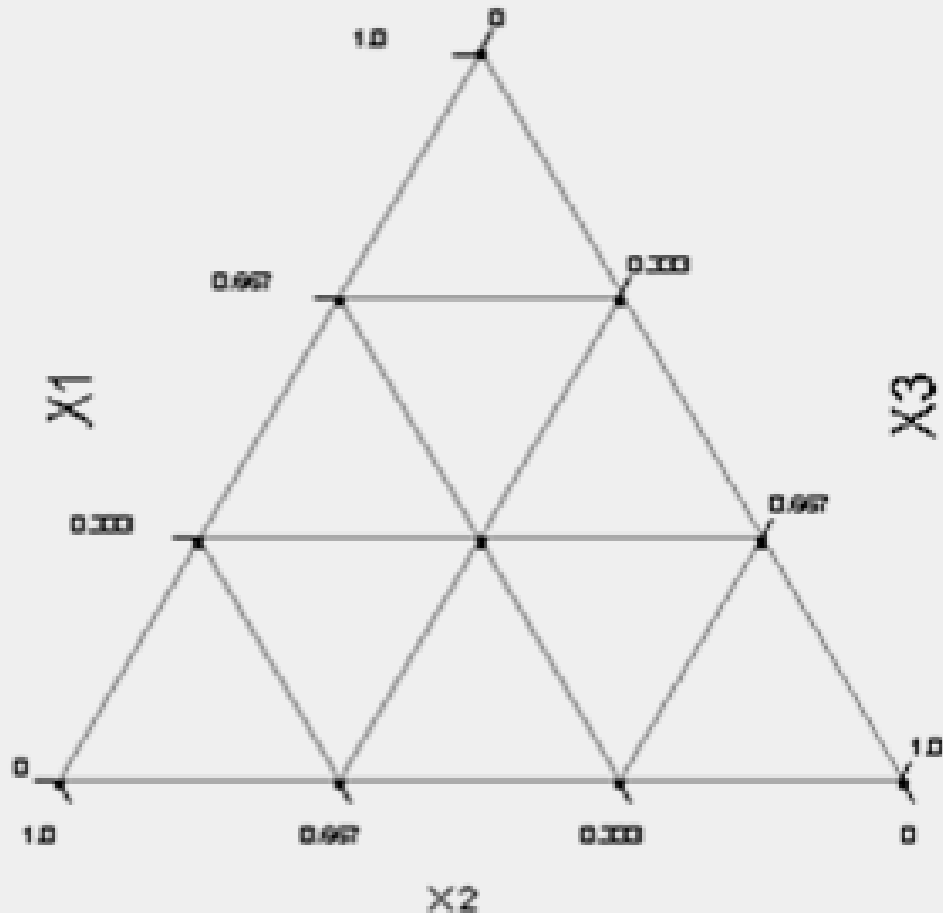
$$d_{t,i,j} = (1 + P(\theta_{t,i,j})) \cdot \|\mathbf{f}'_{t,i}\|$$

$$P(\theta_{t,i,j}) = M \cdot \left(\frac{t}{t_{max}}\right)^\alpha \cdot \frac{\theta_{t,i,j}}{\gamma_{\mathbf{v}_{t,j}}}$$

Cheng, R., Jin, Y., Olhofer, M., & Sendhoff, B. (2016). A reference vector guided evolutionary algorithm for many-objective optimization. *IEEE transactions on evolutionary computation*, 20(5), 773-791.



# Decomposition based MOEAs (RVEA)



Some ways to generate reference vectors uniformly:

- Simplex lattice design
- Das-Dennis approach



# Constraint handling in EAs

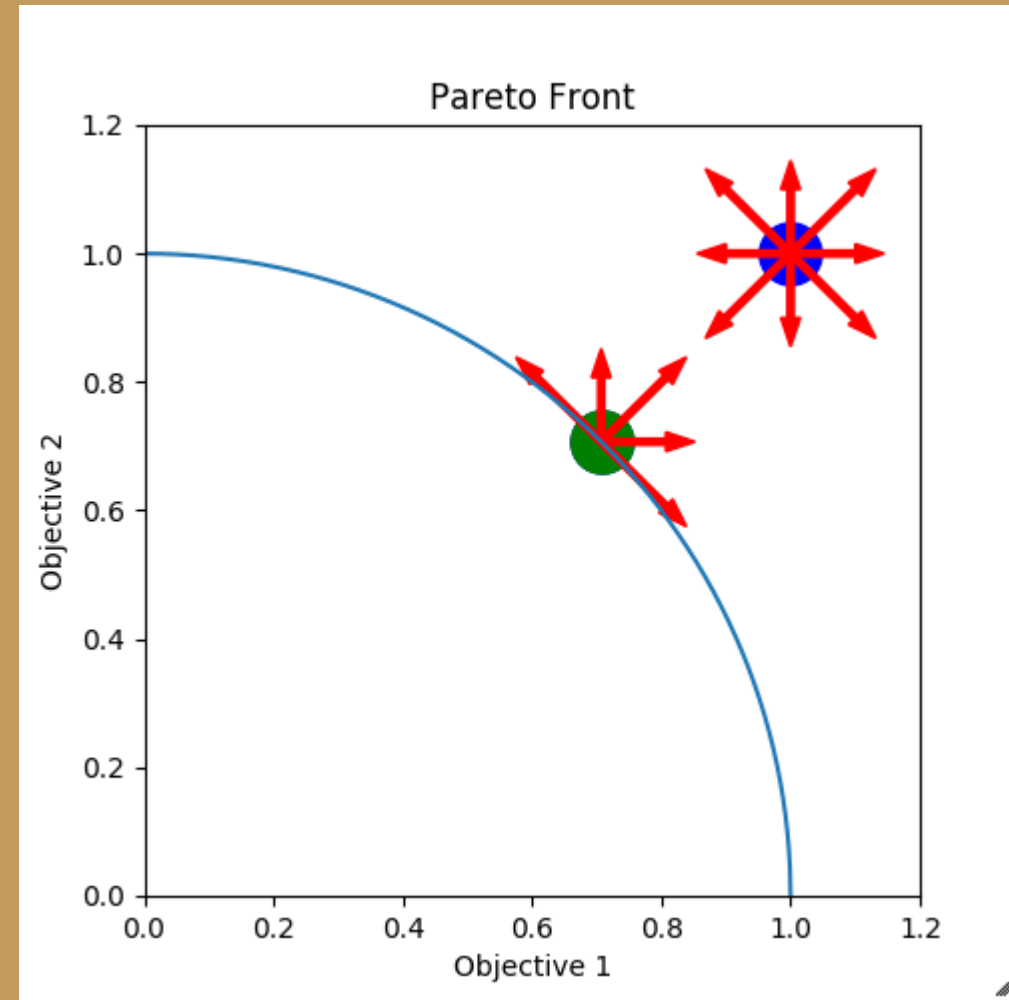
- Different ways to handle constraints.
- Use constraints as additional objectives.
  - Useful when the feasible region is very narrow.
- Penalty function approach:
  - Calculate the sum total of constraint violations for each solution
  - Add this value to each of the objectives (in case of maximization), thus making infeasible solutions worse.
- Box constraints:
  - Decision variables have upper and lower bounds
  - Make sure that the crossover and mutation operators are bounded
  - If they are not bounded “repair” the individuals by resetting the decision variable values to be inside the bounds



# Constraint handling in EAs

- Deb's constraint domination strategy (penalty parameter-less approach). How to compare two individuals?
  - If only one individual is feasible, choose it
  - If both individuals are infeasible, choose the one with a lower total constraint violation value
  - If both individuals are feasible, apply the usual selection strategy (dominance/indicator/decomposition).

# Interactive MOEAs





## Simple case

- Scalarization function + single objective evolutionary algorithm.
- The implementation of NIMBUS in DESDEO uses differential evolution.
- While this is an EA solving an MOP interactively, this technically isn't an interactive MOEA.



# Interactive MOEAs

- Generally, modifications of non-interactive MOEAs.
- Dominance-based MOEAs: Modify the dominance definition
- Indicator-based MOEAs: Modify the indicator to incorporate preference information
- Decomposition-based MOEAs: Reposition the reference vectors

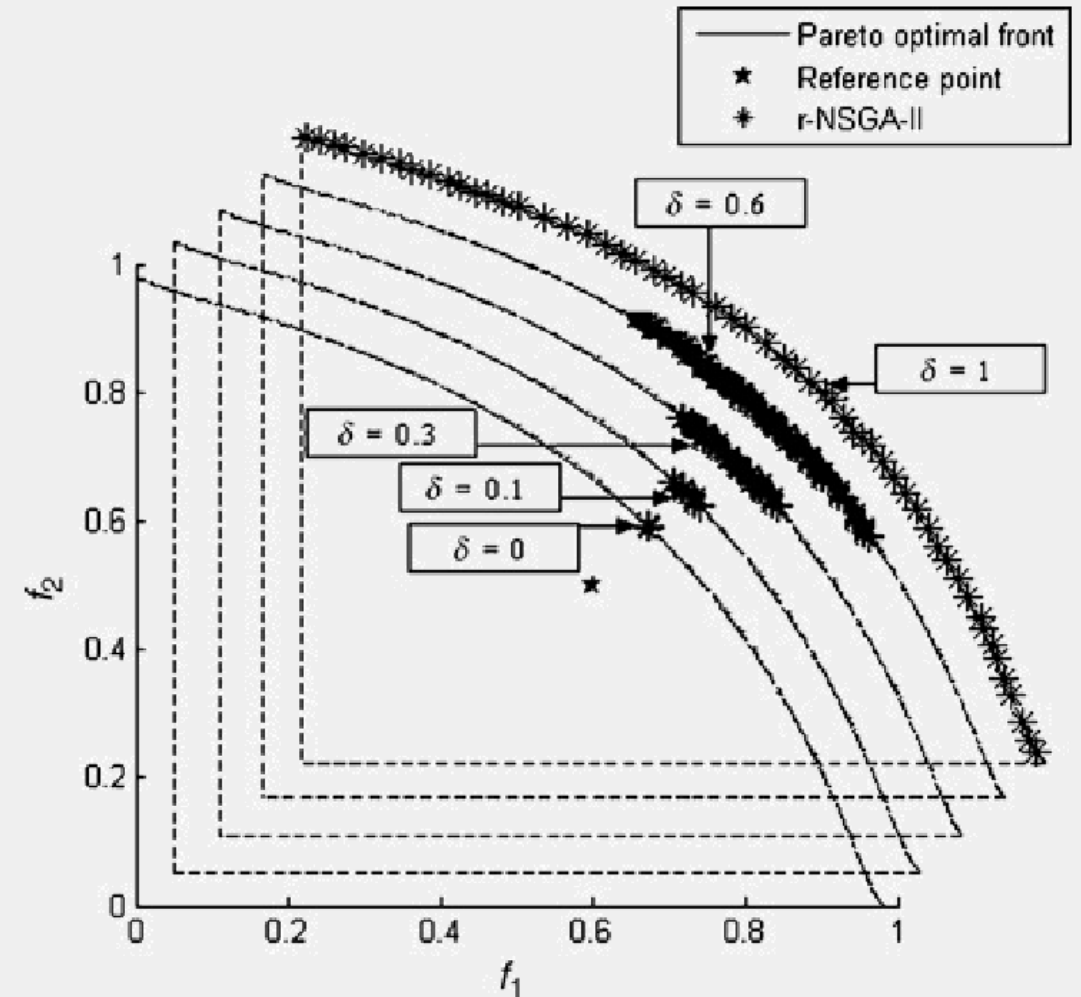
Xin, Bin, et al. "Interactive multiobjective optimization: A review of the state-of-the-art." *IEEE Access* 6 (2018): 41256-41279.



# Dominance-based MOEA (r-NSGA-II)

- New dominance definition (r-dominance)
- If  $x$  dominates  $y$  .....  $x$  also r-dominates  $y$
- If  $x$  and  $y$  are mutually non-dominating .....  $x$  r-dominates  $y$  if:

$$nDist(x, rp) - nDist(y, rp) < -\delta$$







# Indicator-based MOEAs (IBEA -> PBEA)

$$I_{\varepsilon}(\mathbf{x}, \mathbf{y}) = \min_{\varepsilon} \left\{ f_i(\mathbf{x}) - \varepsilon \leq f_i(\mathbf{y}) \right. \\ \left. \text{for } i = 1, \dots, k \right\} \longrightarrow I_p(\mathbf{y}, \mathbf{x}) = I_{\varepsilon}(\mathbf{y}, \mathbf{x}) / s(\mathbf{g}, \mathbf{f}(\mathbf{x}), \delta)$$

Achievement scalarizing function scaled between:  $[\delta, \infty)$

Small delta values lead to a smaller “region of interest”.



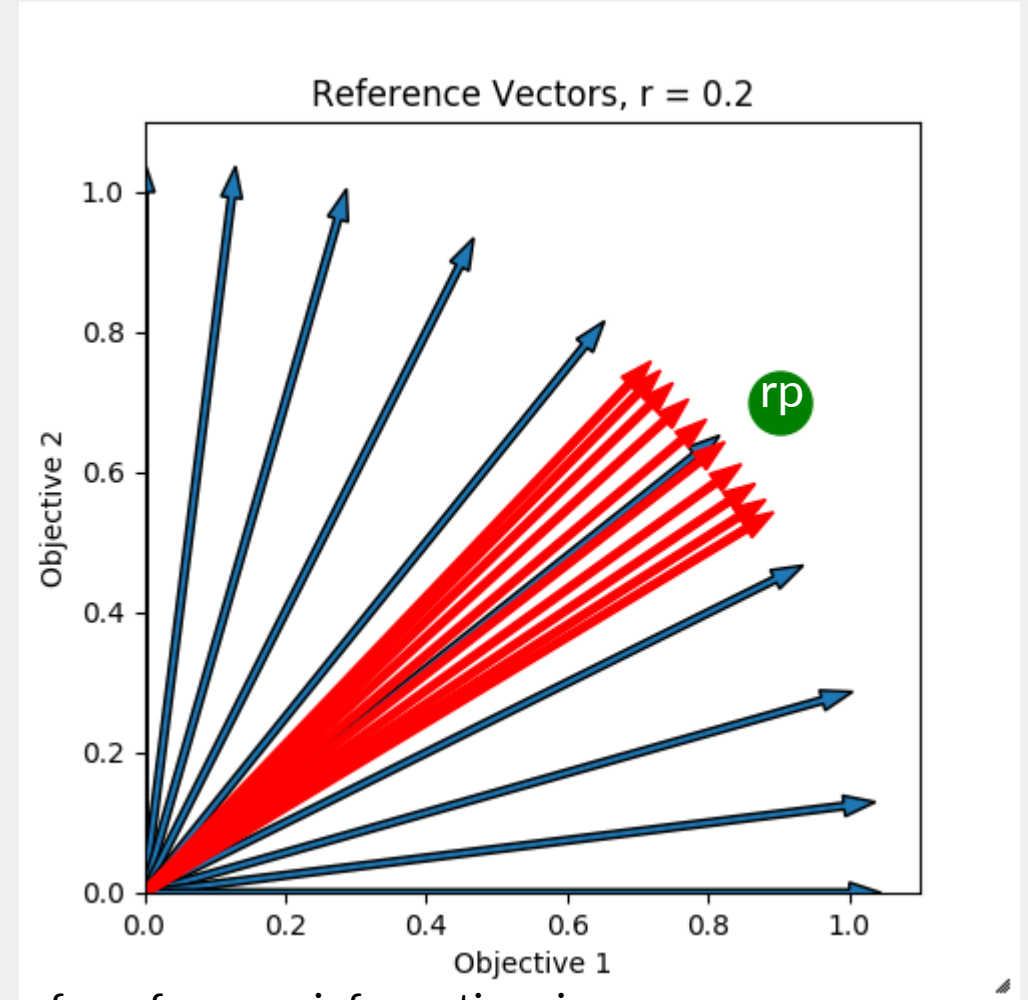
# Interactive RVEA

## Focused reference vectors

$$v' = \frac{r \cdot v + (1-r) \cdot rp}{\|r \cdot v + (1-r) \cdot rp\|}$$

$r \in (0, 1)$

controls the spread of reference vectors around Decision Maker's preference



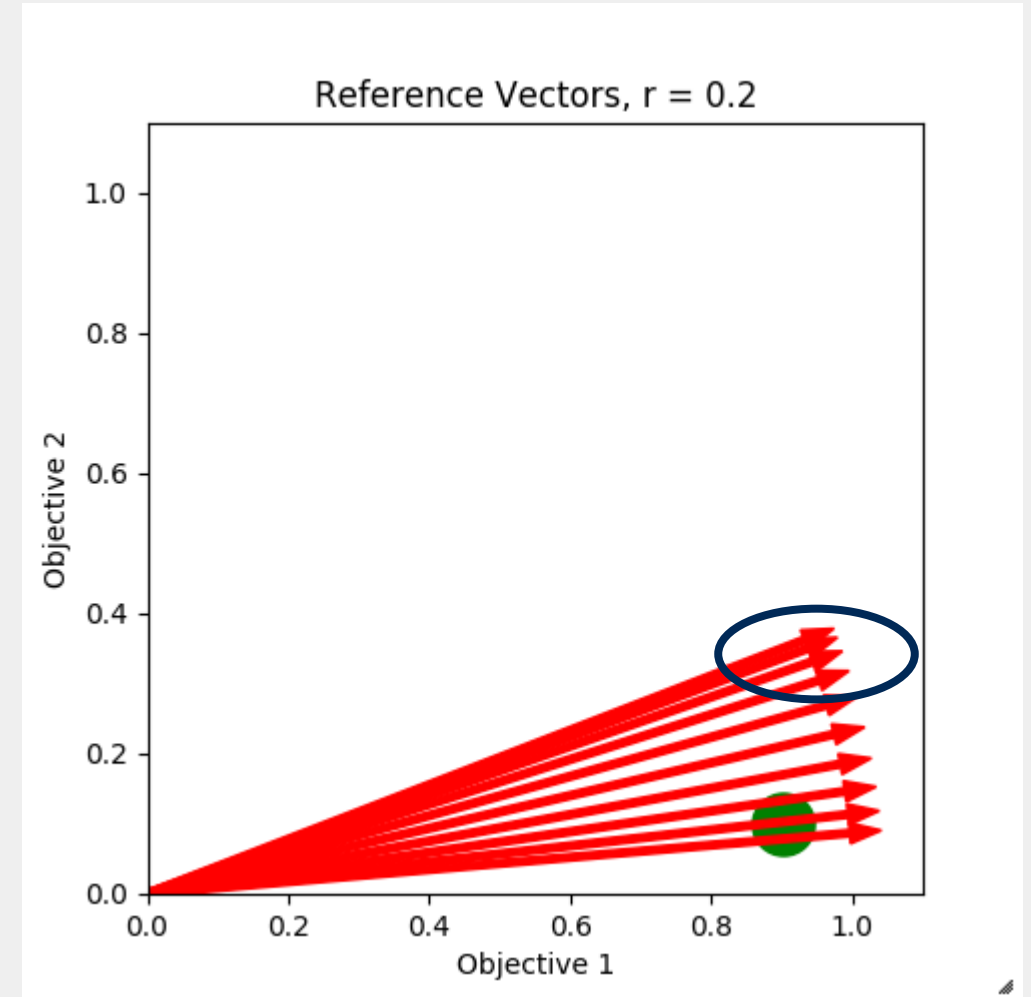
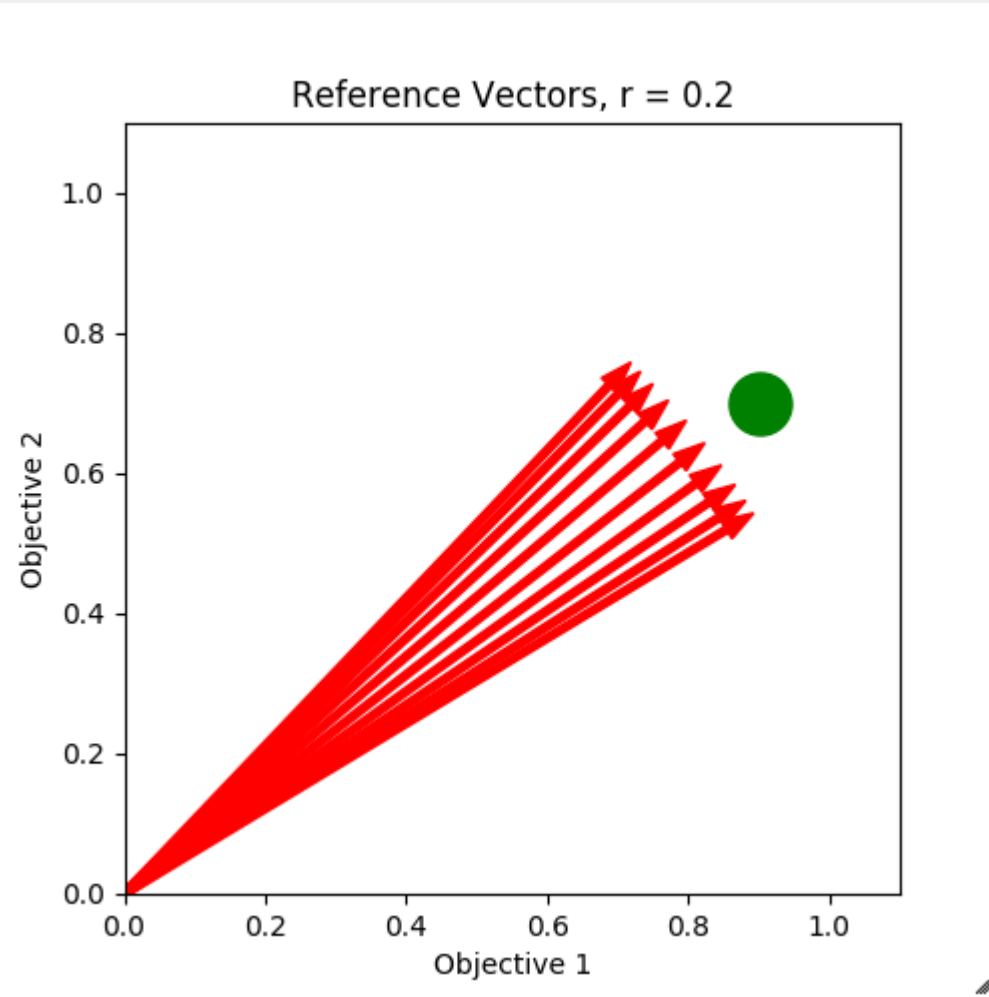
Hakanen, et al. "Connections of reference vectors and different types of preference information in interactive multiobjective evolutionary algorithms." 2016 *IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2016.

# Challenges





# Example: interactive RVEA





# Example: interactive RVEA

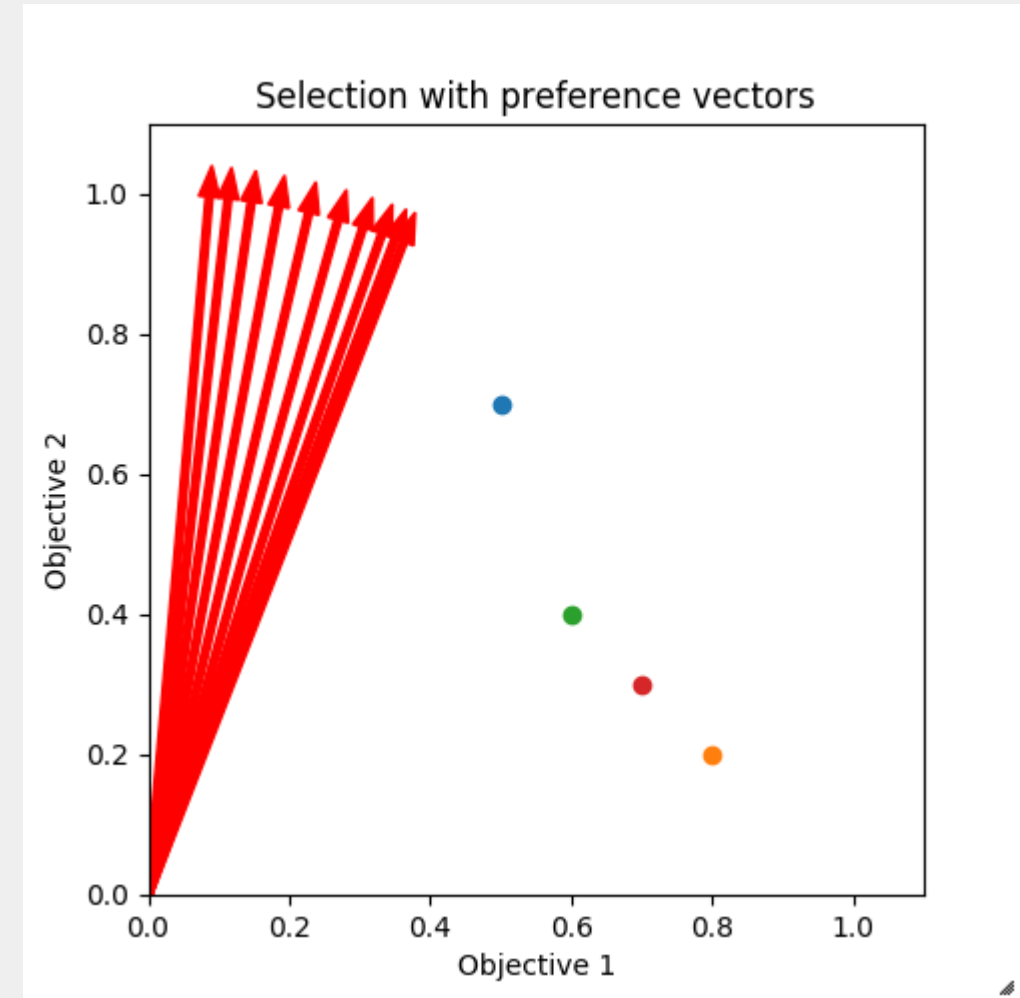
Population diversity is difficult to maintain.

Population diversity is always lost if preferences are changed significantly.

The loss occurs whenever the reference vectors and the solutions are distant.






The reference vectors can change drastically when:

- Ideal point of the population shifts
- Preferences of the DM changes drastically





# General desirable properties

- The method captures the preferences of the DM. 
- The method sets as low cognitive burden on the DM as possible. 
- A user interface supports the DM in problem solving. 
- The DM feels being in control while interacting with the method. 
- The method prevents premature termination of the overall solution process. 



# Desirable properties (learning phase)

- The method helps the DM avoid anchoring. ⚠️
- The method allows exploring any part of the Pareto optimal set. ✅
- The method easily changes the area explored as a response to a change in the preference information given by the DM. ⚠️
- The method allows the DM to learn about the conflict degree and trade-offs among the objectives in each part of the Pareto optimal set explored. ✅
- The method properly handles uncertainty of the information provided by the DM. ❌
- The method allows the DM to find one's region of interest at the end of the learning phase. ⚠️

Afsar, B., Miettinen, K., Ruiz, F., *Assessing the Performance of Interactive Multiobjective Optimization Methods: A Survey*, ACM Computing Surveys, 54(4), article 85, 2021.



# Desirable properties (decision phase)

- The method allows the DM to be fully convinced that (s)he has reached the best possible solution at the end of the solution process. ⚠️
- The method reaches the DM's most preferred solution. ⚠️ ✅
- The method allows the DM to fine-tune solutions in a reasonable number of iterations and/or reasonable waiting time. ✅
- The method does not miss any Pareto optimal solution that is more preferred (with a given tolerance) for the DM than the one chosen. ⚠️





# Open questions

- How to measure the desirable properties quantitatively?
- How to compare different interactive MOEAs?
- How to choose a way to get preferences from the DM?
- How to incorporate those preferences into the MOEA?
- How to choose which MOEA to use for optimization?
- ... and many more.



# References

- K. Deb. Multi-Objective Optimization using Evolutionary Algorithms. Wiley, Chichester, 2001.
- K. Miettinen. Nonlinear Multiobjective Optimization. Kluwer, Boston, 1999.
- El-Ghazali Talbi. Metaheuristics: From Design to Implementation. Wiley Publishing, 2009.
- Jin, Y., Wang, H., & Sun, C. Data-driven evolutionary optimization. Springer International Publishing, 2021.
- Branke et al. (Eds): Multiobjective Optimization: Interactive and Evolutionary Approaches, 2008.