Interactive Multiobjective Optimization: Applications and Tools to Support Decision Making

Day 2: Scalarization-based methods

Giovanni Misitano[1]

[1]University of Jyväskylä (Finland), The Multiobjective Optimization Group

August 12, 2025

JYVÄSKYLÄN YLIOPISTO
UNIVERSITY OF JYVÄSKYLÄ

Intro I

- In this lecture, we will start by defining basic concepts and ideas in multiobjective optimization.
- We will then focus especially on scalarization and scalarization-based interactive multiobjective optimization methods.
- After this lecture, you should be familiar with the most common terms used in the field of multiobjective optimization, and have a general idea how an interactive multiobjective optimization method works.

Intro II

- While many of the ideas discussed during this lecture have a solid mathematical background, we will not try to prove or derive these. Instead, we will state important results and, when applicable, try to understand intuitively the ideas.
- We will mainly follow the book **Nonlinear Multiobjective Optimization** [1] by Kaisa Miettinen, and occasionally the book **Theory of Multiobjective Optimization** by Yoshikazu Sawaragi et. al. [2].
  - For anyone interested in pursuing research in the field of multiobjective optimization, these two books offer a very solid foundation to understand the theory and concepts in multiobjective optimization.
  - I personally recommend starting by reading Miettinen's book as it is easier to follow, it also focuses heavily on interactive multiobjective optimization methods in its second part.
  - To strengthen one's understanding of the mathematical theory behind the concepts in multiobjective optimization, Sawaragi's book is a solid read.

Overview

1. Optimality: what is optimal?
2. Pareto optimal fronts
3. Scalarization and scalarization functions
4. Scalarization-based methods

Optimality: what is optimal?

Definition of a multiobjective optimization problem I

- We can define a multiobjective optimization problem with k **objective functions** $f_i : \mathbb{R}^n \to \mathbb{R}, i \in [1, k]$, and n **decision variables** in a **decision vector** $\mathbf{x} \ni x_j, j \in [1, n]$ as

Multiobjective optimization problem

$$\begin{aligned} \min F(\mathbf{x}) &= (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})) \\ \text{s.t. } \mathbf{x} &= (x_1, x_2, \ldots, x_j) \in S \subseteq \mathbb{R}^n. \end{aligned} \tag{1}$$

- In (1), S is the **feasible set** of solutions. The feasible set is usually defined by constraints.
- $F(\mathbf{x})$, when computed, is called an **objective vector**, often denoted as $\mathbf{z} = F(\mathbf{x})$.
- In this lecture, we will assume that objective functions are always minimized.

Definition of a multiobjective optimization problem II

Think for a moment

If instead of minimizing an objective function we wish to maximize it, what should we do?

Constraints I

Generally, we have three types of constraints: **box-**, **equality** and **inequality** constraints.

Box-constraint

$$x_i^{\text{low}} \leq x_i \leq x_i^{\text{high}},\ x_i \in \mathbf{x} \tag{2}$$

Equality constraint

$$h(\mathbf{x}) - \delta_h = 0$$
$$\text{where } h : \mathbb{R}^n \to \mathbb{R} \text{ and } \delta_h \in \mathbb{R} \tag{3}$$

Constraints II

Inequality constraint

$$g(\mathbf{x}) - \delta_g > 0 \quad \text{where } g : \mathbb{R}^n \to \mathbb{R} \text{ and } \delta_g \in \mathbb{R} \tag{4}$$

- A multiobjective optimization problem may have multiple constraints or sometimes none!
- When a decision vector $\mathbf{x}$ is congruent with all existing constraints, the decision vector in question belongs to the feasible set S.

Constraints III

- Practical tip: when computing constraints using floating point precision, equality constraints (3) are best converted to inequality constraints utilizing some small epsilon $\varepsilon \in \mathbb{R}, \varepsilon > 0$.

Floating point compatible equality constraint

$$|\varepsilon - (h(\mathbf{x}) - \delta_h)| > 0. \tag{5}$$

Constraints IV

Think for a moment

If a multiobjective optimization problem has no constraints, what is the feasible set S?

Optimality I

- For a single-objective optimization problem, Karush-Kuhn-Tucker (KKT) conditions (also known as the saddle-point theorem) can be used to determine if a solution is optimal or not.
- We will not delve deeply in the theory behind KKT conditions in this lecture, but will state some important properties of the theorem.
  1. KKT conditions are necessary and sufficient for optimality in linear and convex optimization (if and only if KKTs are true, then the solution is optimal).
  2. KKT conditions are necessary for optimality in non-convex optimization. I.e., a solution can be optimal only if KKTs are true, but KKTs being true does not imply that the solution is optimal. Non-linear optimization can sometimes be convex, in which case KKTs are necessary and sufficient for a solution to be optimal.
  3. The KKT conditions can be generalized to determine the optimality of multiobjective optimization problems as well.

Optimality II

There are two important definitions for optimality in multiobjective optimization: **Pareto optimality** and **weak Pareto optimality**.

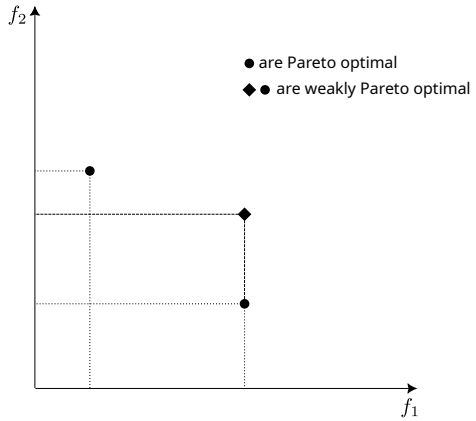
Pareto optimality

A solution $\mathbf{x}^*$ is said to be Pareto optimal if, and only if, $\mathbf{x}^*$ is feasible and there exists **no other** feasible solution $\mathbf{x}$ such that $f_i(\mathbf{x}) \leq f_i(\mathbf{x}^*)$ for all $i \in [1, k]$ and $f_i(\mathbf{x}) < f_i(\mathbf{x}^*)\ \exists i \in [1, k]$.

Weak Pareto optimality

A solution $\mathbf{x}^*$ is said to be weakly Pareto optimal if, and only if, $\mathbf{x}^*$ is feasible and there exists **no other** feasible solution $\mathbf{x}$ such that $f_i(\mathbf{x}) < f_i(\mathbf{x}^*)$ for all $i \in [1, k]$.

Optimality III



- The circles and square are objective vectors corresponding to some feasible solutions to a multiobjective optimization problem (1) with $k = 2$ objectives.

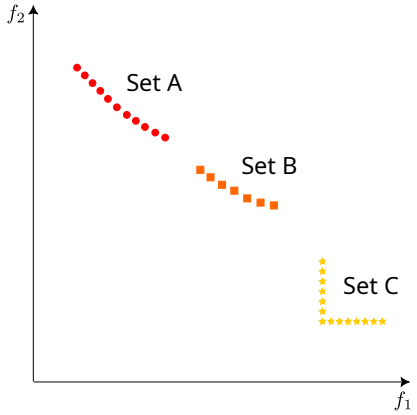
Optimality IV

- In the figure on the previous slide, the square is not Pareto optimal because there exists another solution which is less than or equal in all objective values compared to the square.
- While not a former proof, the figure also illustrates the fact that Pareto optimal solutions are a subset of weakly Pareto optimal solutions.

Optimality V

Think for a moment

What can you say about the Pareto optimality of the sets A, B, and C?



Some properties of Pareto optimal solutions I

There are a few more useful concepts to keep in mind when it comes to the optimality of a solution to a multiobjective optimization problem (1).

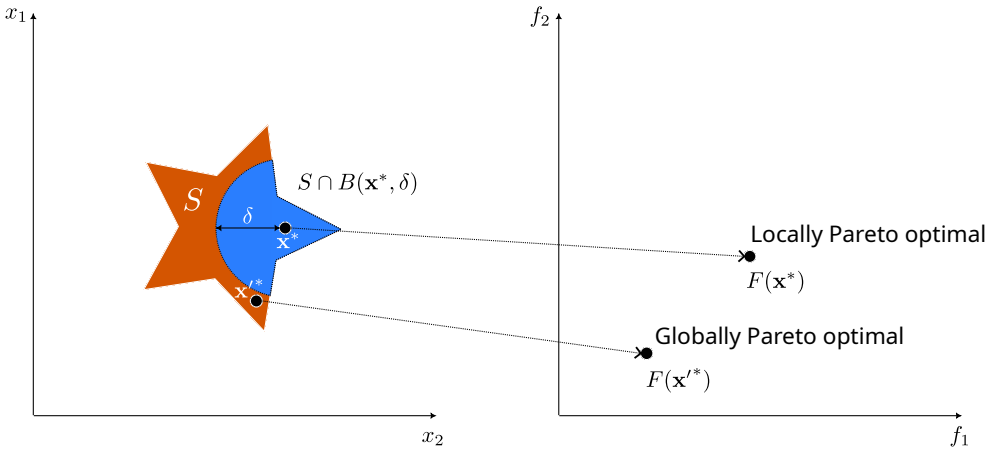
Local Pareto optimality

A solution $\mathbf{x}^*$ is locally Pareto optimal if it is Pareto optimal in some non-vanishing δ-neighborhood centered on $\mathbf{x}^*$ in S. In other words, there exists some $\delta > 0$ such that $\mathbf{x}^*$ is Pareto optimal in $S \cap B(\mathbf{x}^*, \delta)$, where $B(.)$ is an open ball centered on $\mathbf{x}^*$ and of radius δ.

Global Pareto optimality

We may define global Pareto optimality using the above concept of an open ball. A solution $\mathbf{x}^*$ is globally Pareto optimal if there exists some $\delta > 0$ large enough such that $S \subseteq B(\mathbf{x}^*, \delta)$ and $\mathbf{x}^*$ is Pareto optimal in $S \cap B(\mathbf{x}^*, \delta)$. I.e., $\mathbf{x}^*$ is Pareto optimal in a δ-neighborhood in S which engulfs the whole feasible set S.

Some properties of Pareto optimal solutions II



Some properties of Pareto optimal solutions III

- But how do we even know that a (Pareto optimal) solution exists for a multiobjective optimization problem?
- We will present the highlights of these conditions, but before that, we need to define the concepts of domination.

Domination

A solution $\mathbf{x}^*$ is said to **dominate** another solution $\mathbf{x}$ if, and only if, $f_i(\mathbf{x}^*) \leq f_i(\mathbf{x})\ \forall i \in [1, k]$ and $f_i(\mathbf{x}^*) < f_i(\mathbf{x})\ \exists i \in [1, k]$. When a solution $\mathbf{x}^*$ dominates another solution $\mathbf{x}$, it is notated as $F(\mathbf{x}^*) \prec F(\mathbf{x})$. I.e., when we talk about domination relations, we always compare the objective function values.

Some properties of Pareto optimal solutions IV

- The first condition for solutions to exist for a multiobjective optimization problem is the trivial requirement for the feasible set to be non-empty: $S \neq \emptyset$.
- The second condition is that the domination structure of the solutions must be **acyclic**. In other words, there must be no cycles in the dominance relation of the optimal solutions.
  - Cyclic dominance relation: $F(\mathbf{x}^1) \prec F(\mathbf{x}^2) \prec F(\mathbf{x}^3) \prec F(\mathbf{x}^1)$.
  - Acyclic dominance relation: $F(\mathbf{x}^1) \prec F(\mathbf{x}^2) \prec F(\mathbf{x}^3)$.
- Third, the feasible set S must be compact.
- Fourth, the objective functions f_i must be lower semi-continuous in their domain.
- For the ones interested in a more rigorous explanation of these conditions, reading Chapter 3.2 in [2] is recommended.

Trade-offs I

- As we have seen, multiple (weakly) Pareto optimal solutions can exists for a given multiobjective optimization problem.
- Moreover, these solutions can only be partially ordered at best, in other words, they are mathematically incomparable. A decision maker must choose the best solution among the Pareto optimal ones.
- It is clear that we cannot switch from one Pareto optimal solution to another without having to make a **trade-off** in regard to at least one objective.
- This is a **unique characteristic of multiobjective optimization problems**: multiple optimal solutions exists, but one cannot switch from one solution to another without making a trade-off.

Trade-offs II

Trade-offs between two (feasible) solutions, $\mathbf{x}^1$ and $\mathbf{x}^2$, to a multiobjective optimization problem can be formalized as a **trade-off matrix** Λ with elements:

Trade-off matrix elements

$$\Lambda_{ij} = \frac{f_i(\mathbf{x^1}) - f_i(\mathbf{x^2})}{f_j(\mathbf{x^1}) - f_j(\mathbf{x^2})}, \ i, j \in [1, k]. \tag{6}$$

Trade-offs III

Think for a moment

When computing trade-offs (6), in which given case should we be extra careful?

Trade-offs IV

Example 1: computing the trade-off matrix for a problem with $k = 3$ objectives and two feasible solutions.

- Consider the two following feasible, Pareto optimal solutions, $\mathbf{x}^1$ and $\mathbf{x}^2$, with corresponding objective vectors $\mathbf{z}^1$ and $\mathbf{z}^2$:

$$\begin{aligned}\mathbf{z}^1 &= (4, 7, 4),\\ \mathbf{z}^2 &= (2, 10, 3).\end{aligned}$$

- The trade-off matrix (6) is then

$$\Lambda = \begin{pmatrix} 1 & -0.6\overline{66} & 2 \\ -1.5 & 1 & -3 \\ 0.5 & -0.3\overline{33} & 1 \end{pmatrix}. \tag{7}$$

Trade-offs V

- From the computed matrix (7), we can deduce some of its properties:
  1. $\text{diag}(\Lambda) = \mathbb{1}$;
  2. $\Lambda_{ij} = \Lambda_{ji}^{-1}$; and
  3. An element Λ_{ij} reflects the rate of change: how much does objective i change when we increment objective j by one unit?
- Practical tips: due to property 2., it is enough to compute either the lower or upper triangle of Λ. If the denominator of some element Λ_{ij} seems to (numerically) approach infinity, it can be assumed that $f_j(\mathbf{x}^1) = f_j(\mathbf{x}^2)$ in (6).

Proper Pareto optimality I

There are two more useful concepts, related to trade-offs, to define when it comes to the optimality of solutions to a multiobjective optimization problem.

Proper Pareto optimality

A solution $\mathbf{x}^*$ is said to be properly Pareto optimal if it is Pareto optimal and if there is some positive, non-zero $M \in \mathbb{R}$ such that for each f_i and $\mathbf{x} \in S$ satisfying $f_i(\mathbf{x}) \leq f_i(\mathbf{x}^*)$, there exists at least one f_j such that $f_j(\mathbf{x}^*) \leq f_j(\mathbf{x})$ and

$$\frac{f_i(\mathbf{x}^*) - f_i(\mathbf{x})}{f_j(\mathbf{x}) - f_j(\mathbf{x}^*)} \leq M. \tag{8}$$

Proper Pareto optimality II

ε-proper Pareto optimality

A solution $\mathbf{x}^*$ is ε-properly Pareto optimal if

$$(F(\mathbf{x}^*) - \mathbb{R}^k_\varepsilon \backslash \mathbf{0}) \cap Z = \emptyset, \tag{9}$$

where $\mathbb{R}^k_\varepsilon = \{F(\mathbf{x}^*) \in \mathbb{R}^k | \max_{i \in [1,k]} f_i(\mathbf{x}^*) + \varepsilon \sum_{i=1}^k f_i(\mathbf{x}^*) \geq 0\}$, and ε is a scalar value.

- Proper Pareto optimality can be defined in other ways as well.
- The point of definitions (8) and (9) is to limit the set of Pareto optimal solutions to a set where infinite or large enough trade-offs are not allowed.
  - E.g., the denominator in (6) and (8) is very small, then the value of Λ_{ij} approaches positive or negative infinity.
  - This means that the value of the objective function f_j in question for the two solutions being compared are very close to each other, which make render the trade-off in question not very interesting from the perspective of a DM, for example.

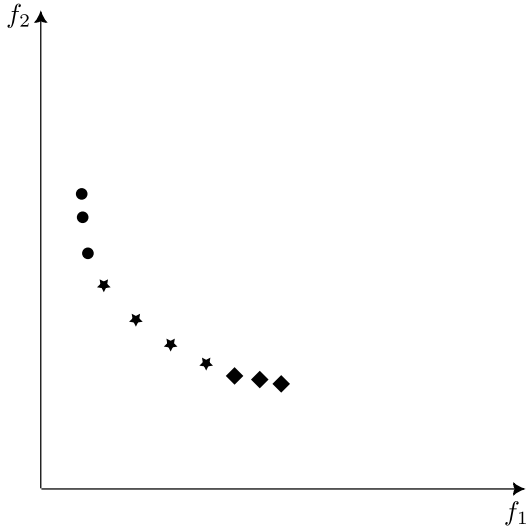
Proper Pareto optimality III

- From a decision-making perspective, eliminating solutions from the Pareto optimal solutions with these very large or infinite trade-offs, is desirable.
- There are also other ways to determine proper Pareto optimality. It is not as a clear concept as Pareto optimality and weak Pareto optimality.

Proper Pareto optimality IV

Think for a moment

Which of the following solutions (circles, stars, or diamonds) can be considered to be properly Pareto optimal?



Pareto optimal fronts

Pareto optimal fronts I

- We have already seen that a multiobjective optimization problem can have multiple Pareto optimal solutions.
- The set of all Pareto optimal solutions to a multiobjective optimization problem can be defined as:

The Pareto optimal set and front

The set P consisting of all Pareto optimal solutions to a multiobjective optimization problem (1) is known as the **Pareto optimal set**. The image of the Pareto optimal set P is defined as the **Pareto optimal front** Z^{Pareto}. The Pareto optimal set is a subset of the feasible solutions S, i.e., $P \subseteq S$. Furthermore, if we define Z as the image of the feasible set, then the Pareto optimal front is a subset of this set, i.e., $Z^{Pareto} \subseteq Z$.

Pareto optimal fronts II

- **Be careful**, in the literature, the Pareto optimal set and front may be named differently. For instance, efficient is sometimes used instead of Pareto: efficient set and efficient front.
- Sometimes the Pareto optimal set and solution may refer directly to the objective space.
- We rarely know the true Pareto optimal set and fronts of a multiobjective optimization problem. Therefore, we often deal with representations or approximations of it.
- We will talk about Pareto optimal fronts from now on, which are more interesting from a DM and decision-making perspectives.

Pareto optimal fronts III

- Lower (best) and upper (worst) of the objective function values present in a Pareto optimal front are characterized by the **ideal** and **nadir** points, respectively. These are defined below.

Ideal point

The ideal point $\mathbf{z}^*$ of a Pareto optimal front for some problem (1) is defined as the vector consisting of the lowest individual values for each objective function when each objective function is minimized separately.

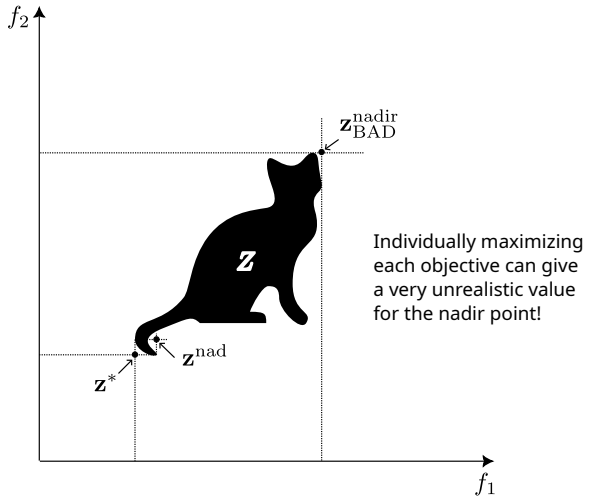
Nadir point

The nadir point $\mathbf{z}^{\text{nadir}}$ is defined as the highest individual values found for each objective function in the Pareto optimal front for some problem (1).

Pareto optimal fronts IV

- Notice that the ideal point can be computed while computing the nadir point generally requires information on the full extent of the Pareto optimal front.
- Consequently, the ideal point is trivial to compute (just minimize each objective function separately), but the nadir point can cause headaches.
- **A word of caution**: if the objective vector found by minimizing an objective function individually is not unique, i.e., the objective function has multiple optima, then the resulting objective vector may or may not be Pareto optimal!
- Why do not we just maximize each objective function to find the nadir point? Because the nadir point should characterize the upper bound of the Pareto optimal front. Maximizing each objective will yield generally incorrect results.
- To find the **true** nadir point, the whole Pareto optimal front must be known. Computing the full front is generally infeasible in practice.

Pareto optimal fronts V



Individually maximizing each objective can give a very unrealistic value for the nadir point!

Pareto optimal fronts VI

- A simple method to approximate the nadir point of a Pareto optimal front is to use the **payoff-table method**.
- After computing the ideal point by minimizing each objective function individually, the resulting objective vectors can be arranged in a table (square matrix): one vector on each row.
- Then, the maximum value can be taken from each column. These values represent an approximation of the nadir point.

Pareto optimal fronts VII

Example 2: approximating the nadir point using the payoff-table method.

- Suppose we have computed the ideal point of a problem with $k = 3$ objectives by minimizing each objective function individually: minimizing the first objective function yields $\mathbf{z}^1 = (-3.2, 4.3, -2.2)$, the second $\mathbf{z}^2 = (3.9, 2.2, -2.5)$, and the third $\mathbf{z}^3 = (3.1, 2.9, -4.2)$.
- Arranging the objective vectors in a table $(\mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3)^\mathsf{T}$, we have the payoff-table:

$$\begin{pmatrix} -3.2 & 4.3 & -2.2 \\ 3.9 & 2.2 & -2.5 \\ 3.1 & 2.9 & -4.2 \end{pmatrix} \tag{10}$$

Think for a moment

What is the ideal point, and what is the nadir point, according to the computed payoff-table?

Pareto optimal fronts VIII

- Generally, the payoff-table method will yield nadir points that are either far too low (optimistic) or far too high (pessimistic). This is important to keep in mind when using the method.
- The main advantage of the payoff-table method is its simplicity. That is it.
- In practice, the nadir point can be guessed if the payoff-table method fails to yield anything useful.
- In some cases, a DM may also be able to provide an estimate of the nadir point based on their domain expertise.

Pareto optimal fronts IX

When it comes to the feasible set S and the image of the set Z of a multiobjective optimization problem (1), there are some important results to keep in mind.

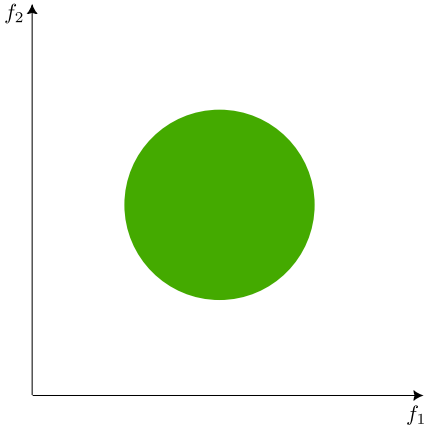
Convex multiobjective optimization problem

A multiobjective optimization problem is said to be convex if the feasible region S of the problem is convex and if all the objective functions (and consequently Z) are convex.

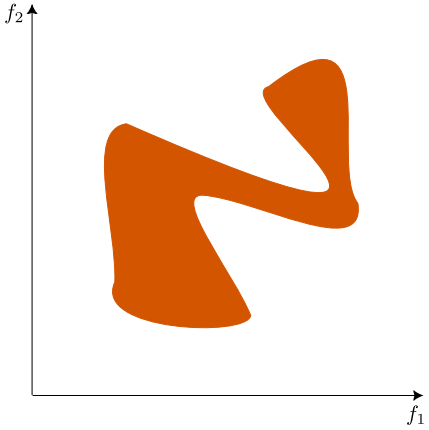
Pareto optimal solution for a convex problem

Any locally Pareto optimal solution for a convex multiobjective optimization problem is also globally Pareto optimal.

Pareto optimal fronts X



Convex



Non-convex

Pareto optimal fronts XI

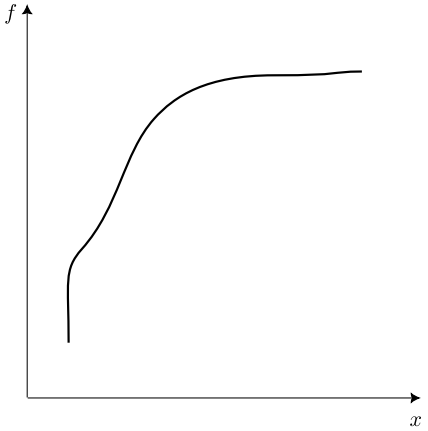
Quasiconvex function

A function $f : \mathbb{R}^n \to \mathbb{R}$ is quasiconvex if $f(\beta \mathbf{x}^1 + (1-\beta)\mathbf{x}^2) \leq \max(f(\mathbf{x}^1), f(\mathbf{x}^2))$ for $0 \leq \beta \leq 1$ and for all $\mathbf{x}^1, \mathbf{x}^2 \in \mathbb{R}^n$.

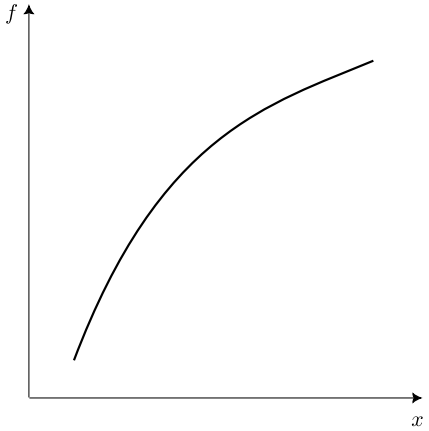
Strictly quasiconvex function

A function $f : \mathbb{R}^n \to \mathbb{R}$ is strictly quasiconvex if $f(\beta \mathbf{x}^1 + (1-\beta)\mathbf{x}^2) < \max(f(\mathbf{x}^1), f(\mathbf{x}^2))$ for $0 < \beta < 1$ and for all $\mathbf{x}^1, \mathbf{x}^2 \in \mathbb{R}^n$.

Pareto optimal fronts XII



Quasiconvex



Strictly quasiconvex

Pareto optimal fronts XIII

The concept of quasiconvexity and strict quasiconvexity leads to the following important results.

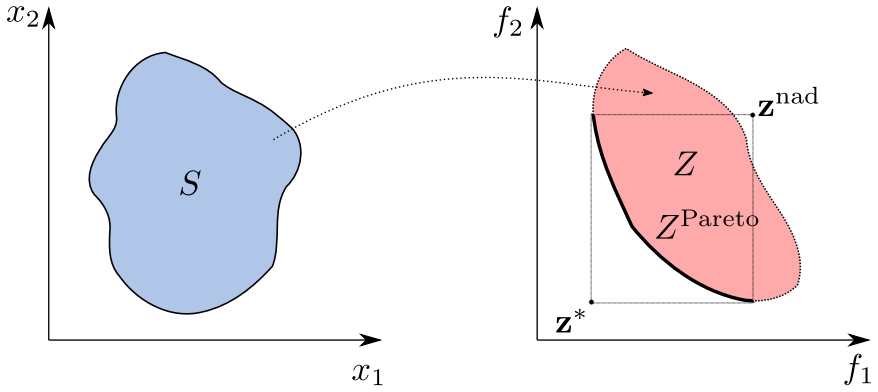
Relaxed relation between local and global Pareto optimal solutions

If a multiobjective optimization problem has a convex feasible region S and quasiconvex objective functions with at least one strictly quasiconvex objective function, then each locally Pareto optimal solution is also globally Pareto optimal.

- When the multiobjective optimization problem has a non-convex feasible region S and non-linear objective functions, **a locally Pareto optimal solution may be or may not be globally Pareto optimal**.

Pareto optimal fronts XIV

$$\min_{\mathbf{x} \in S} \{f_1(\mathbf{x}), f_2(\mathbf{x})\}$$



Scalarization and scalarization functions

Scalarization I

- One approach to finding Pareto optimal solutions to a multiobjective optimization problem (1) is to **scalarize** it.
- By scalarizing, we **transform** the original problem from a multiobjective optimization problem to a single-objective optimization problem.
  - I.e.: from $\mathbb{R}^n \to \mathbb{R}^k$ to $\mathbb{R}^n \to \mathbb{R}$.
- We do this because we know how to solve single-objective optimization problems!

Think for a moment

What sort of issues do you think can arise in transforming a multiobjective optimization problem to a single-objective optimization one?

Scalarization II

As an example of scalarization, consider the ε-constraint method:

ε-constraint method

$$\begin{aligned} \min \quad & f_i(\mathbf{x}) \\ \text{s.t.} \quad & f_j(\mathbf{x}) \leq \varepsilon_j, \forall\, j \in [1,k], j \neq i, \\ & \mathbf{x} \in S. \end{aligned} \tag{11}$$

- In the ε-constraint method we **select one objective to be minimized** and then include the remaining objectives as constraints to the problem where the value of the objective functions are not to exceed certain $\varepsilon_j \in \mathbb{R}$ values.
- In the general case, the ε-constraint method **can find any Pareto optimal solution**.

Scalarization III

Think for a moment

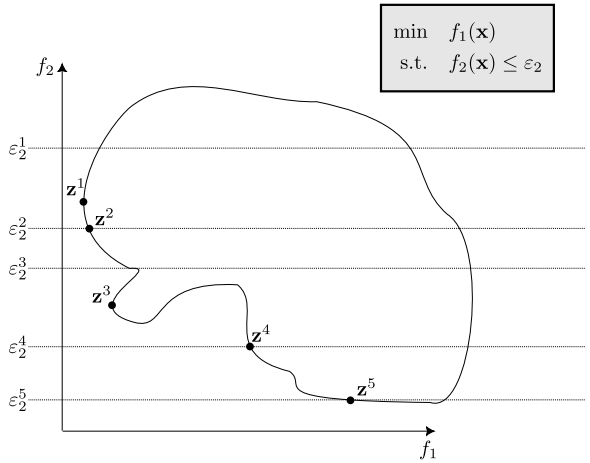
What challenges do you see arising with the use of the ε-constraint method?

Scalarization IV

- **Setting the ε_j values can be challenging** in the ε-constraint method since the weights are not to exceed the minimum and maximum values of the corresponding objective function f_j.
- To make sure a solution to the ε-constraint problem is Pareto optimal, **the solution must be a solution to the problem for each** $i \in [1, k]$. Otherwise the solution is weakly Pareto optimal.

Scalarization V

The working principle of the ε-constraint method in the two-dimensional case:



Scalarization VI

Another example of scalarization is the **weighted sum method**:

Weighted sum method

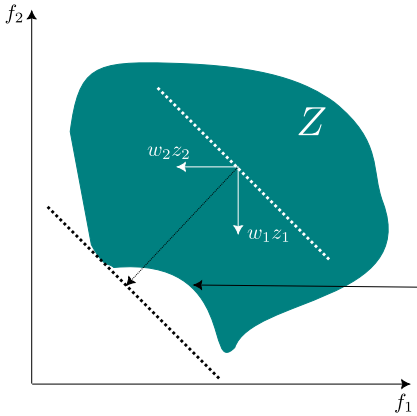
$$\min_{\mathbf{x} \in S} \sum_{i=1}^{k} w_i f_i(\mathbf{x}), \tag{12}$$

where $\mathbf{w}$ is a vector of weights w_i, and $\sum_{i=1}^{k} w_i = 1$ and $w_i \geq 0 \; \forall i \in [1, k]$.

- Weighted sum can only find Pareto solutions at vertices of the Pareto front; solutions on edges are missed.
- It cannot capture all Pareto solutions even in convex problems and fails completely for non-convex co-domains.
- The solutions found by the weighted sum method are Pareto optimal when the weights are positive and non-zero ($w_i > 0$) and unique.

Scalarization VII

Graphical illustration of the weighted sums method in the two-dimensional case:



The weights dicate the slope of a "search line". It moves towards the origin, until only one objective vector intersects with the line. Thus, no matter how we select the weights, the weighted sum method cannot find the solutions in the concave part :(

Scalarization VIII

Think for a moment

Despite it apparent drawbacks, the weighted sum method remains perhaps the most popular scalarization method, or method in general, to tackle multiobjective optimization problems. Sometimes, it is, very much mistakenly, regarded synonymous to multiobjective optimization! Why is this the case? What do you believe its perceived advantages could be?

Scalarization functions I

- We can also **consider functions** that can be used to transform a multiobjective optimization problem (1) into a single-objective optimization problem.
- This kind of functions are known as **scalarization functions** s and can be defined as follows:

Scalarization function

$$s : \mathbb{R}^k \to \mathbb{R}. \tag{13}$$

Scalarization functions II

- A scalarization function s is a function of objective values and some additional parameters $\mathbf{p}$.
- Utilizing a scalarization function, we can formulate **a scalarized (multiobjective optimization) problem**:

Scalarized problem

$$\begin{aligned} &\min s(F(\mathbf{x}); \mathbf{p}) \\ &\text{s.t. } \mathbf{x} \in S, \end{aligned} \tag{14}$$

where $\mathbf{p}$ are additional parameters supplied to the scalarization function.

Scalarization functions III

Think for a moment

What kind of considerations and limitations do you think applying scalarization functions to scalarize a multiobjective optimization problem can emerge and should be considered?

Scalarization functions IV

- Desirable properties have been defined for scalarization functions (13):
  1. When minimized, a scalarization function can be used to find any Pareto optimal solution.
  2. Every solution resulting from minimizing a scalarization function is Pareto optimal.
  3. If aspiration levels are used (covered shortly), the solution found by solving the scalarized problem is satisfying given that the aspiration levels are feasible.

Scalarization functions V

The scalarization function used in the reference point method [3] is the *achievement scalarizing function* defined as follows:

Achievement scalarizing function

$$s_{\text{ASF}}(F(\mathbf{x}); \mathbf{w}, \bar{\mathbf{z}}) = \max_{i \in [1,k]} \left[w_i \left(f_i(\mathbf{x}) - \bar{z}_i \right) \right] + \rho \sum_{i=1}^{k} w_i \left(f_i(\mathbf{x}) - \bar{z}_i \right), \tag{15}$$

where $\bar{\mathbf{z}}$ is known as **a reference point**, which consists of **aspiration levels** z_i, and ρ is some small positive scalar (e.g, $\rho \sim 10^{-6}$).

- The version of the achievement scalarizing function is its **augmented version**. The augmentation comes from the summation term, which makes the scalarization function order approximating guaranteeing its solutions to be Pareto optimal (actually ε-properly Pareto optimal!). Without the augmentation term, the solutions are only weakly Pareto optimal.

Scalarization functions VI

In the STOM method [4], the following scalarization function is used:

STOM scalarization function

$$s_{\text{STOM}}(F(\mathbf{x}); \bar{\mathbf{z}}, \mathbf{z}^{**}, \rho) = \max_{i \in [1,k]} \left[\frac{f_i(\mathbf{x}) - z_i^{**}}{\bar{z}_i - z_i^{**}} \right] + \rho \sum_{i=1}^{k} \frac{f_i(\mathbf{x})}{\bar{z}_i - z_i^{**}}. \tag{16}$$

- In (16), $\mathbf{z}^{*}*$ is the *utopian point* of the multiobjective optimization problem being solved, it is defined as the a slightly better ideal point with each element defined as $z_i^{**} = z_i^{*} - \rho_i^{*}$, where ρ_i^{*} is some small positive scalar.
- Notice that by setting $w_i = (\bar{z}_i - z_i^{**})^{-1}$ in (15), we get the scalarization function in STOM (16).

Scalarization functions VII

Think for a moment

Why do we need an utopian point in the STOM scalarization function (16)?

Scalarization functions VIII

Let us examine one more scalarization function, function used in the GUESS method [5]:

GUESS scalarizing function

$$s_{\text{GUESS}}(F(\mathbf{x}); \bar{\mathbf{z}}, \mathbf{z}^{\text{nad}}) = \max_{i \in [1,k]} \left[\frac{f_i(\mathbf{x}) - z_i^{\text{nad}}}{z_i^{\text{nad}} - \bar{z}_i} \right]. \tag{17}$$

- Notice the lack on an augmentation term in (17). This version of the GUESS scalarizing function is **order representing**, thus solving the respective scalarized problem results in the solution to be guaranteed to be only weakly Pareto optimal.
- (17) can be augmented with an augmentation term to guarantee its solutions to be (properly) Pareto optimal.

Scalarization functions IX

Some scalarization functions can be generalize, which the GLIDE scalarization [6] is able to achieve:

GLIDE scalarization function

$$\text{GLIDE} \quad \begin{cases} \min & \alpha + \rho \sum_{i=1}^{k} \omega_i^h \left(f_i(\mathbf{x}) - q_i^h \right) \\ \text{s.t.} & \mu_i^h \left(f_i(\mathbf{x}) - q_i^h \right) \leq \alpha \quad (i = 1, \ldots, k), \\ & f_i(\mathbf{x}) \leq \varepsilon_i^h + s_\varepsilon \cdot \Delta\varepsilon_i^h \quad (i = 1, \ldots, k), \\ & \mathbf{x} \in S. \end{cases} \tag{18}$$

Scalarization functions X

There is also GLIDE2 [7] and its two variants:

GLIDE2 differentiable

$$\text{GLIDE2-dif} \quad \begin{cases} \min & \alpha + \rho \sum_{i=1}^{k} \omega_i^h \left(f_i(\mathbf{x}) - q_i^h \right) & \\ \text{s.t.} & \mu_i^h \left(f_i(\mathbf{x}) - q_i^h \right) \leq \alpha & \text{for } i \in I_\alpha^h, \\ & f_i(\mathbf{x}) \leq \varepsilon_i^h + s_\varepsilon \cdot \Delta\varepsilon_i^h & \text{for } i \in I_\varepsilon^h, \\ & \mathbf{x} \in S, & \end{cases} \tag{19}$$

Scalarization functions XI

GLIDE2 non-differentiable

$$\text{GLIDE2-ndif} \quad \begin{cases} \min & \max\limits_{i \in I_\alpha^h} \left\{ \mu_i^h \left(f_i(\mathbf{x}) - q_i^h \right) \right\} + \rho \sum\limits_{i=1}^{k} \omega_i^h \left(f_i(\mathbf{x}) - q_i^h \right) \\ \text{s.t.} & f_i(\mathbf{x}) \leq \varepsilon_i^h + s_\varepsilon \cdot \Delta \varepsilon_i^h \qquad \text{for } i \in I_\varepsilon^h, \\ & \mathbf{x} \in S. \end{cases} \tag{20}$$

Scalarization functions XII

- We saw a few scalarization functions, but many more exist.
- For a list and comparison of different scalarization functions, see [8] for example.
- Remember that the same preferences (e.g., a reference point) may yield different solutions when utilized with different scalarization functions!

Solving scalarized problems I

Think for a moment

We have mentioned properties and guarantees of (Pareto) optimality for some scalarization functions. However, when solving the scalarized problem, what needs to be considered?

Solving scalarized problems II

- After scalarizing a multiobjective optimization problem, it needs to be solved to yield results.
- One needs to be careful and remember that most optimization algorithms find local solutions. Consequently, the Pareto optimal solutions found by solving a scalarized problem can also be local.
- Sometimes global optimizers can be used, especially when gradient information on the objective functions is available.
- In real-life problems, we often end-up with opaque-box functions, in which case nothing certain can be said about the quality of the Pareto optimal solutions found.
- In this last case, it is better to always assume that the solutions are only locally Pareto optimal at best!

Scalarization-based methods

Scalarization based methods

- Multiobjective optimization methods based on scalarization (whether by using scalarizing functions or other means to scalarize the multiobjective optimization problem being solved) can be used to support a DM in finding their most preferred solution.
- These methods are sometimes also called *MCDM methods* to distinguish them from population-based methods (covered later during the course).
- This is not to say that heuristics cannot be used to solve the scalarized problems present in MCDM methods—sometimes it is necessary, e.g., in when objective functions are opaque-boxes.
- However, when applicable, we should use exact methods to solve scalarized problems to increase the accuracy of the solutions found.

Non-interactive methods

- We already saw two non-interactive methods when discussing scalarization: the ε-constraint method (11) and the weighted sums method (12).
- In the ε-constraint method, the upper bounds ε_i can be set **a priori** by a DM.
- Likewise, in the weighting method, the weights w_i can be specified by a DM before the optimization process.
- However, specifying upper bounds and weights is not necessarily very intuitive for the DM.
- **Aspiration levels**, and **reference points**, can be argued to be more intuitive from the perspective of a DM compared to weights and ε-values. We will focus on methods based on reference points.

Non-interactive methods: Goal programming I

- Goal programming is one of the first methods created specifically to solve multiobjective optimization problems.
- In goal programming, a DM sets **aspiration levels** $\bar{\mathbf{z}}$, then, the deviations of the objective functions $f_i(\mathbf{x})$ from the aspiration levels are minimized.
- Using the aspiration levels, **goals** can be formulated as $f_i(\mathbf{x}) \leq \bar{z}_i$, or as equalities or ranges.
- The deviation from a goal can be defined as $\delta_i = \bar{z}_i - f_i(\mathbf{x})$. The deviation can be either positive or negative. Consequently, the deviation can be expressed as the difference between two positive values $\delta_i = \delta_i^- - \delta_i^+$, where $\delta_i^+ \delta_i^- = 0$.
- Therefore, the goal can be represented as $\bar{z}_i = f_i(\mathbf{x}) + \delta_i^- - \delta_i^+$.
- When minimizing, it is sufficient to minimize the δ_i^+ deviations.

Non-interactive methods: Goal programming II

A **weighted goal programming** problem can be formulated as follows:

Weighted goal programming problem

$$\begin{aligned} \min \quad & \sum_{i=1}^{k} w_i \delta_i^+ \\ \text{s.t.} \quad & f_i(\mathbf{x}) - \delta_i^+ \leq \bar{z}_i \ \forall i \in [1,k], \\ & \delta_i^+ \geq 0 \ \forall \in [1,k], \\ & \mathbf{x} \in S. \end{aligned} \tag{21}$$

- Notice that in (21) both $\mathbf{x}$ and δ_i^+ are variables.

Non-interactive methods: Goal programming III

It is also possible to formulate a min-max goal programming problem utilizing goals:

Min-max goal programming problem

$$\begin{aligned} \min \quad & \max_{i \in [1,k]} \delta_i^+ \\ \text{s.t.} \quad & f_i(\mathbf{x}) - \delta_i^+ \leq \bar{z}_i \ \forall i \in [1,k], \\ & \mathbf{x} \in S. \end{aligned} \tag{22}$$

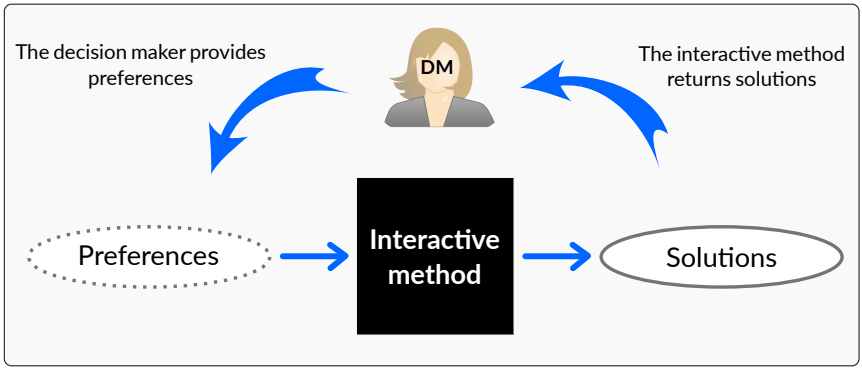
Think for a moment

If the aspiration levels given in goal programming $\bar{\mathbf{z}}$ are feasible (i.e., they are contained in Z, the image of the feasible region), what issue will arise given the optimality of the solutions found?

Interactive methods I

- For the rest of this lecture, we will be discussing a few different **interactive methods**.
- An interactive method is a method for solving multiobjective optimization problems in a manner where the DM is involved **interactively** in the solution process. One of the tasks of the DM is to provide **preference information**.
- An interactive method is more than just an optimization procedure. During an interactive method, the DM has an opportunity to **learn** about and **explore** the multiobjective optimization problem being solved.
- The DM can also learn about their own preferences and fine tune them to find their best solution.

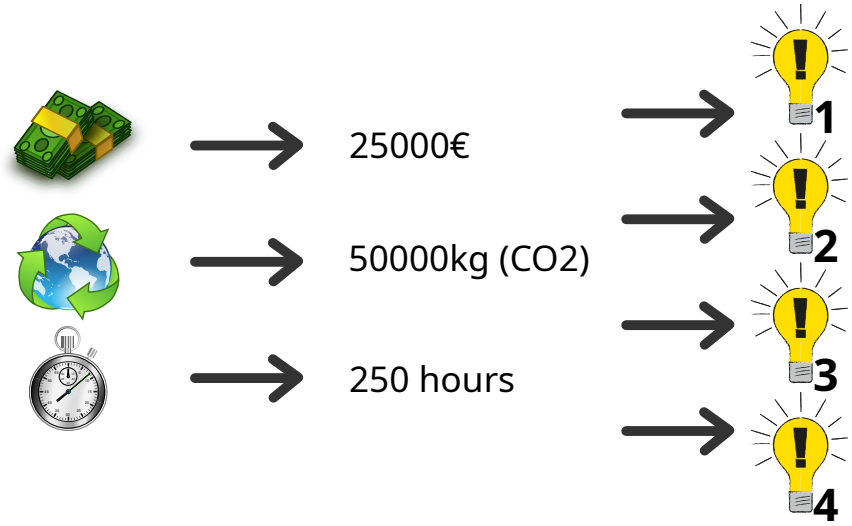
Interactive methods II



Interactive methods: The reference point method I

- In the reference point method [3], in each iteration, a DM provides a **reference point $\bar{\mathbf{z}}$**, which is then used to minimize the achievement scalarizing function (15).
- In addition, the reference point is perturbed, creating k additional reference points, which are again used to minimize the achievement scalarizing function (15).
- The DM is then shown $k + 1$ solutions, from which they can either choose their most preferred solution and stop, or the DM may provide a new reference point and a new set of $k + 1$ solutions are computed, which starts a new iteration.

Interactive methods: The reference point method II



Interactive methods: The reference point method III

- The reference point method gives the DM an idea how the solutions look around the reference point given. The farther the reference point is from the Pareto front, the more spread the $k + 1$ solutions will be.
- The reference point method can be regarded as a generalization of goal programming, but **it can handle both feasible and infeasible reference points!**
- The method is easy to implement and readily modified by changing the scalarizing function used.
- The main issue with the method is that the DM has little to no support in providing the aspiration levels for the reference point. At best, they may have knowledge of the extent of the Pareto optimal from knowing the ideal and, possibly, the nadir point of the problem.

Interactive methods: The reference point method IV

Think for a moment

Does the reference point method have any parameters you can think of?

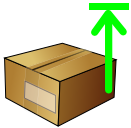
Interactive methods: The synchronous NIMBUS method I

- In the synchronous NIMBUS method [9] a DM provides preference information by **classifying** each objective function of the current solution $f(\mathbf{x}^c) = \mathbf{z}$ in an iteration in one of five classes:
  1. objective functions whose value is desired to improve from their current value: $I^{<}$;
  2. objective functions whose value is desired to improve until some level $\hat{z}_i$ from their current value: $I^{\leq}$;
  3. objective functions whose current value is acceptable: $i^{=}$;
  4. objective functions whose current value can be impaired until some level ε_i: $I^{\geq}$; and
  5. objective functions whose value is allowed to change freely: $I^{\diamond}$.
- Each of the five classes is a set that contains objective functions classified by a DM. The values for $\hat{z}_i$ and ε_i are also provided by the DM.
- For the classifications to be valid, it is required that $I^{<} \cup I^{\leq} \neq \emptyset$ and $I^{\geq} \cup I^{\diamond} \neq \emptyset$.

Interactive methods: The synchronous NIMBUS method II

- Based on the classifications provided by the DM, a reference point is then formulated, which is required later in the scalarizing functions used in NIMBUS:
  - $\bar{z}_i = z_i^*$ for $i \in I^<$;
  - $\bar{z}_i = \hat{z}_i$ for $i \in I^\leq$;
  - $\bar{z}_i = z_i$ for $i \in I^=$;
  - $\bar{z}_i = \varepsilon_i$ for $i \in I^\geq$;
  - $\bar{z}_i = z_i^{\text{nad}}$ for $i \in I^\diamond$.
- Notice that we require information on the ideal and nadir points of the problem to be able to formulate the reference point.

Interactive methods: The synchronous NIMBUS method III



500 units

200 €/kwh

50000kg (CO2)

Interactive methods: The synchronous NIMBUS method IV

- The DM can also choose how many new solutions are computed based on the classifications, up to a maximum of four.
- The solutions are computed utilizing different scalarization functions. At least one new solution is always computed utilizing the NIMBUS scalarization function (23).

NIMBUS scalarization function

$$
\begin{aligned}
\min \quad & \max_{\substack{i \in I^{<} \\ j \in I^{\leq}}} \left[\frac{f_i(\mathbf{x}) - z_i^*}{z_i^{\text{nad}} - z_i^{**}}, \frac{f_j(\mathbf{x}) - \hat{z}_j}{z_j^{\text{nad}} - z_j^{**}} \right] + \rho \sum_{i=1}^{k} \frac{f_i(\mathbf{x})}{z_i^{\text{nad}} - z_i^{**}} \\
\text{s.t.} \quad & f_i(\mathbf{x}) \leq f_i(\mathbf{x}^c) \text{ for each } i \in I^{<} \cup I^{\leq} \cup I^{=}, \\
& f_i(\mathbf{x}) \leq \varepsilon_i \text{ for each } i \in I^{\geq}, \\
& \mathbf{x} \in S.
\end{aligned}
\tag{23}
$$

Interactive methods: The synchronous NIMBUS method V

- The three other solutions are computed utilizing the scalarization functions used in the reference point method (15), the GUESS method (17), and the STOM method (16).
- When using the scalarization function of the reference point method, we set the weights to be $w_i = (z_i^{\text{nad}} - z_i^{**})^{-1}$.
- When using the scalarization function of the GUESS method, we require that $i \notin I^{\diamond}$, and we add an augmentation term.
- The scalarization function used in the STOM method is used as presented.

Interactive methods: The synchronous NIMBUS method VI

- Additionally, when using NIMBUS, the DM has the option to save interesting solutions to an **archive**.
- The DM has also the option to see a selected number of **intermediate solutions** computed between two previous solutions.
- Synchronous NIMBUS has an old web-based interface: WWW-NIMBUS. It is the first web-based interface that was developed for an interactive multiobjective optimization method. The interface is available here: `http://nimbus.mit.jyu.fi/`
- We also have more modern versions!

Interactive methods: The synchronous NIMBUS method VII

Think for a moment

Can you see any issues arising on the side of the DM when utilizing a method like synchronous NIMBUS?

Interactive methods: The NAUTILUS family of methods I

- The **NAUTILUS family** of interactive multiobjective optimization methods [10] is a special one in that **each solution process starts from the nadir point** and gradually approaches the Pareto optimal front until it is reached.
- Starting from the nadir point allows the DM to see improvements in all objective function values, gradually approaching the Pareto optimal front. This is why NAUTILUS methods can be regarded as **trade-off free** methods.
- By allowing the DM to improve each objective function value in subsequent iterations can reduce the effects of the **anchoring bias**, for instance.

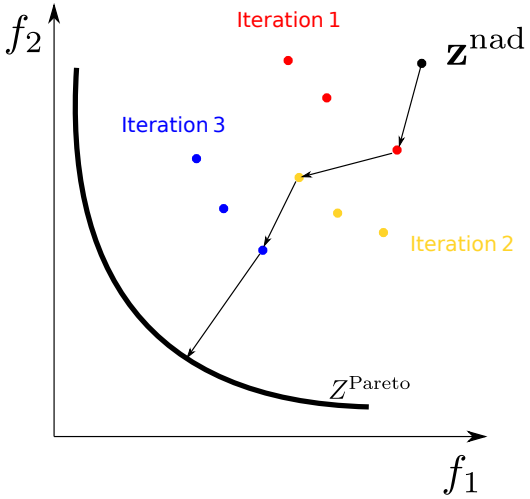
Interactive methods: E-NAUTILUS I

- In the E-NAUTILUS method [11], the DM starts the solution process from the nadir point. Then, in each iteration, a number of intermediate, non-dominated points, is generated. Non-dominated means that none of the generated intermediate solutions dominates another.
- The DM can select the number of iterations and the number of intermediate solution generated.
- From the intermediate points, the DM selects their preferred one. Then, a new set of intermediate points closer to the Pareto optimal front is generated based on the selected point.
- The DM has also the option to step back to a intermediate point generated in a prior iteration.

Interactive methods: E-NAUTILUS II

- This process continues until a solution on the Pareto optimal front is reached. The DM can set the number of total iterations and the number of intermediate points generated in each iteration. The number of remaining iterations can also be changed during the method.

Interactive methods: E-NAUTILUS III



Interactive methods: E-NAUTILUS IV

- With NAUTILUS methods in general, a representation of the Pareto optimal front is often used instead of the true front.
- Therefore, once a solution is reached in E-NAUTILUS, a post-processing stage follows.
- In the post-processing stage, an accurate model of the multiobjective optimization problem being solved is used and, for example, some scalarizing function is used to solve said problem by utilizing the solution found by the DM as a reference point.
- How to calculate representations of the Pareto optimal front, is discussed in a later lecture.

Interactive methods: E-NAUTILUS V

Think for a moment

Why do you think representations (e.g., approximations) of the Pareto front are used in NAUTILUS methods, and some other methods as well? In general, what are the benefits of using a representation of a front instead of the true front in interactive methods?

Interactive methods: NAUTILUS Navigator I

- NAUTILUS Navigator is an interactive multiobjective optimization **navigation method**.
- In NAUTILUS Navigator, the DM can see in real-time how the reachable ranges of the objective function values change as the process approaches gradually the Pareto optimal front.
- The DM can stop and start the navigation process at any time. The DM can guide the process by setting aspiration levels and upper bounds for each objective function.
- The idea of the method is best demonstrated with the video found here: `https://www.youtube.com/watch?v=gjvIG8PiPBo`
- Once a solution is found, a post-processing phase follows like in E-NAUTILUS, if a representation of the Pareto optimal front was used.

Interactive methods: NAUTILUS Navigator II

- Navigation methods are interesting because they allow the DM to see how the optimization process evolves in real time.
- Other examples of navigation methods are Pareto navigator [12] and Pareto race [13], for instance.
- In a later lecture, we will also discuss a recently developed new navigation method: O-NAUTILUS [14], which is suitable for solving computationally expensive problems.

Interactive methods: Conclusions I

- We saw a glimpse of examples of interactive multiobjective optimization methods. Many more methods exist in addition to the ones discussed so far.
- For an overview of existing interactive methods, see, e.g., [15].
- From a decision-making perspective, it is important to have different interactive methods because **different DMs can have different requirements and needs.**
  - For instance, one DM may prefer to supply aspiration levels, while another DM might was to classify objective functions.
  - A good decision support system should be able to adhere to the needs of its users.
- It is also important to keep in mind that different interactive methods can produce different solutions with the same preference information!

Interactive methods: Conclusions II

- Lastly, let us note a big problem with interactive methods: they are very challenging to compare to each other!
- As the DM is in a central role in interactive methods, the human aspects cannot be ignored. For example, a DM may prefer one method to another solely based on the interface of the method. But is preferring the interface of a method over another the same as preferring the method underneath the interface to another?
- There is completed and ongoing work on trying to come up with systematic ways to compare interactive methods. Some with human DMs, some with artificial DMs, or ADMs. See, for example, [16] for a recent review on how the performance of interactive methods have been assessed in the past.
- Ongoing work: we are researching in our group an experimental framework to compare interactive methods with human DMs. Some of our most recent progress can be found in a recent publication [17].

Some open research questions

- How to compare interactive methods?
- How to compute the nadir point for multiobjective optimization problems with $k > 2$?
- How to model and mitigate various cognitive biases affecting the DM, such as the anchoring bias?
- How to transform preference information from one form to another? E.g., how to express ranges as a reference point?
- How to choose the best scalarizing function in scalarization-based methods?
- How to best visualize solutions of a multiobjective optimization problem to a DM?
- How to best support DMs in providing preference information in interactive methods?
- And many more...

Bibliography I

[1] Kaisa Miettinen. *Nonlinear multiobjective optimization*. Boston: Kluwer Academic Publishers, 1999.

[2] Yoshikazu Sawaragi, HIROTAKA NAKAYAMA, and TETSUZO TANINO. *Theory of multiobjective optimization*. Elsevier, 1985.

[3] A. P. Wierzbicki. "The Use of Reference Objectives in Multiobjective Optimization". In: *Multiple Criteria Decision Making, Theory and Applications*. Ed. by G. Fandel and T. Gal. Berlin: Springer, 1980, pp. 468–486. DOI: 10.1007/978-3-642-48782-8_32.

[4] Hirotaka Nakayama. "Aspiration Level Approach to Interactive Multi-Objective Programming and Its Applications". In: *Advances in Multicriteria Analysis*. Ed. by Panos M. Pardalos, Yannis Siskos, and Constantin Zopounidis. Boston, MA: Springer, 1995, pp. 147–174. ISBN: 978-1-4757-2383-0. DOI: 10.1007/978-1-4757-2383-0_10.

Bibliography II

[5] John Telfer Buchanan. "A naive approach for solving MCDM problems: The GUESS method". In: *Journal of the Operational Research Society* 48.2 (1997), pp. 202–206.

[6] Mariano Luque, Francisco Ruiz, and Kaisa Miettinen. "Global formulation for interactive multiobjective optimization". In: *Or Spectrum* 33.1 (2011), pp. 27–48.

[7] Francisco Ruiz, Mariano Luque, and Kaisa Miettinen. "Improving the computational efficiency in a global formulation (GLIDE) for interactive multiobjective optimization". In: *Annals of Operations Research* 197.1 (2012), pp. 47–70.

[8] Kaisa Miettinen and Marko M. Mäkelä. "On scalarizing functions in multiobjective optimization". In: *OR Spectrum* 24.2 (2002), pp. 193–213. DOI: 10.1007/s00291-001-0092-9.

[9] Kaisa Miettinen and Marko M. Mäkelä. "Synchronous approach in interactive multiobjective optimization". In: *European Journal of Operational Research* 170.3 (2006), pp. 909–922. DOI: 10.1016/j.ejor.2004.07.052.

Bibliography III

[10] Kaisa Miettinen and Francisco Ruiz. "NAUTILUS framework: towards trade-off-free interaction in multiobjective optimization". In: *Journal of Business Economics* 86.1-2 (2016), pp. 5–21. DOI: 10.1007/s11573-015-0786-0.

[11] Ana B. Ruiz et al. "E-NAUTILUS: A decision support system for complex multiobjective optimization problems based on the NAUTILUS method". In: *European Journal of Operational Research* 246.1 (2015), pp. 218–231. DOI: 10.1016/j.ejor.2015.04.027.

[12] Petri Eskelinen et al. "Pareto navigator for interactive nonlinear multiobjective optimization". In: *OR spectrum* 32.1 (2010), pp. 211–227.

[13] Pekka Korhonen and Jyrki Wallenius. "A Pareto race". In: *Naval Research Logistics (NRL)* 35.6 (1988), pp. 615–623.

[14] Bhupinder Singh Saini et al. "Optimistic NAUTILUS navigator for multiobjective optimization with costly function evaluations". In: *Journal of Global Optimization* (2022), pp. 1–25.

Bibliography IV

[15] Bin Xin et al. "Interactive Multiobjective Optimization: A Review of the State-of-the-Art". In: *IEEE Access* 6 (2018), pp. 41256–41279. DOI: 10.1109/ACCESS.2018.2856832.

[16] Bekir Afsar, Kaisa Miettinen, and Francisco Ruiz. "Assessing the performance of interactive multiobjective optimization methods: a survey". In: *ACM Computing Surveys (CSUR)* 54.4 (2021), pp. 1–27.

[17] Bekir Afsar et al. "An experimental design for comparing interactive methods based on their desirable properties". In: *Annals of Operations Research* 338.2 (2024), pp. 835–856.