

DESDEO: The framework for interactive multiobjective optimization

Giovanni Misitano

`giovanni.a.misitano@jyu.fi`

University of Jyväskylä
The multiobjective optimization group

July 29, 2022



JYVÄSKYLÄN YLIOPISTO
UNIVERSITY OF JYVÄSKYLÄ



- 1 Introduction
- 2 Background
- 3 DESDEO
 - What is DESDEO about?
 - desdeo-problem
 - desdeo-tools
 - desdeo-emo
 - desdeo-mcdm
 - Hybridization
 - User interface(s)
- 4 What I want you to remember
- 5 Bibliography

- 1 Introduction
- 2 Background
- 3 DESDEO
- 4 What I want you to remember
- 5 Bibliography

- Many real-life problems can be modeled as multiobjective optimization problems.
- These problems have many solutions with different real-life consequences.
- Involving a human decision maker is vital to find the best solution(s).
 - In this regard, interactive methods for multiobjective optimization are especially important to consider.

- Software for experimenting and implementing interactive multiobjective optimization methods is lacking.
- Frameworks containing both scalarization based and evolutionary methods have not existed before.
- A central hub for implementations of various interactive methods is needed.

- DESDEO makes it easy for developers and researchers alike to utilize existing methods and implement their own.
- DESDEO contains both scalarization based and evolutionary methods.
- DESDEO has the largest collection of interactive methods, at least as far as we know.

- 1 Introduction
- 2 Background
- 3 DESDEO
- 4 What I want you to remember
- 5 Bibliography

Multiobjective optimization problems

- A multiobjective optimization problem has many conflicting objectives that are to be optimized simultaneously [1].

Multiobjective optimization problem

A multiobjective optimization problem can be defined as

$$\min F(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_k(\mathbf{x})), \quad (1)$$

where $f_1 \dots f_i$, $i \in [1, k]$ are objective functions and \mathbf{x} is a decision variable vector. The vectors \mathbf{x} can be subject to both **box-constraints** and **function constraints**. Feasible \mathbf{x} belong to the *feasible variable space* S or $\mathbf{x} \in S$.

Box-constraints

$$x_i^{\text{low}} \leq x_i \leq x_i^{\text{high}}, x_i \in \mathbf{x} \quad (2)$$

Function constraints

$$\begin{aligned} g(\mathbf{x}) - \delta_g &> 0 \\ h(\mathbf{x}) - \delta_h &= 0 \\ \delta_g, \delta_h &\in \mathbb{R} \end{aligned} \quad (3)$$

- In (2) x_i^{low} and x_i^{high} are the lower and higher limits for the i th element in \mathbf{x} , respectively.
- In (3) δ_g and δ_h are scalar values which should be exceeded or be exactly matched by $g(\mathbf{x})$ and $h(\mathbf{x})$, respectively.

Domination

For two feasible solutions $\mathbf{x}^1, \mathbf{x}^2 \in S$, \mathbf{x}^1 is said to dominate \mathbf{x}^2 if and only if $f_i(\mathbf{x}^1) \leq f_i(\mathbf{x}^2)$ for all $i = 1 \dots k$, and $f_j(\mathbf{x}^1) < f_j(\mathbf{x}^2)$ for at least some index $j = 1 \dots k$.

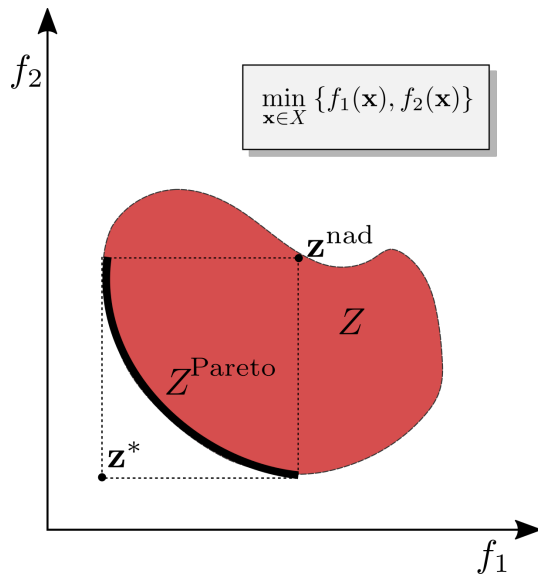
Pareto optimality

A solution \mathbf{x}^* is said to be Pareto optimal if and only if there exists no other solution in S that dominates it. The set of objective vectors corresponding to the Pareto optimal solutions is known as the Pareto optimal front.

Ideal and nadir points

The ideal \mathbf{z}^* and nadir \mathbf{z}^{nad} points represent the best (lowest) and worst (highest) values of the objective function values on the Pareto optimal front, respectively.

More definitions



- One way to solve multiobjective optimization problems is to transform them into single-objective optimization problems by scalarizing them using some scalarizing function s :

Scalarized problem

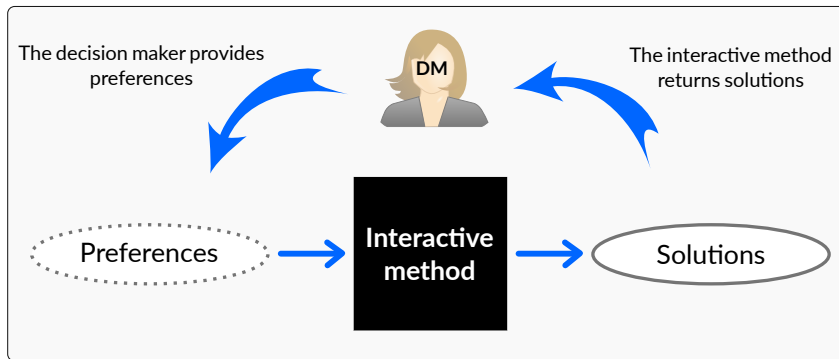
$$\begin{aligned} \min \quad & s(\mathbf{F}(\mathbf{x}); \mathbf{p}) \\ \text{subject to} \quad & \mathbf{x} \in S, \end{aligned} \tag{4}$$

where \mathbf{p} is a set of additional parameters given to the scalarizing function s .

- Examples of scalarizing functions can be found in [2], for instance.

- Another way to solve multiobjective optimization problems is to use evolutionary methods [3].
- These methods utilize evolutionary operations (cross-over, mutation, recombination) to evolve a population of solutions towards Pareto optimality.
- Being heuristic in their nature, these methods cannot guarantee the Pareto optimality of any solution found.
- Evolutionary methods are often the best way to get at least approximate solutions to most real-life problems.

- As said, multiobjective optimization problems have multiple solutions. Usually, we are interested in the Pareto optimal solutions.
- The concept of a best solution is therefore subjective to the preferences of a decision maker (DM) who is often a domain expert.
- The DM can provide their preferences before, after, or during a solution process.
 - We are interested in the last option, which is also known as interactive multiobjective optimization.



- One of the major benefits of interactive methods is that they allow the DM to learn about their own preferences and the multiobjective optimization problem.

1 Introduction

2 Background

3 DESDEO


- What is DESDEO about?
- desdeo-problem
- desdeo-tools
- desdeo-emo
- desdeo-mcdm
- Hybridization
- User interface(s)

4 What I want you to remember

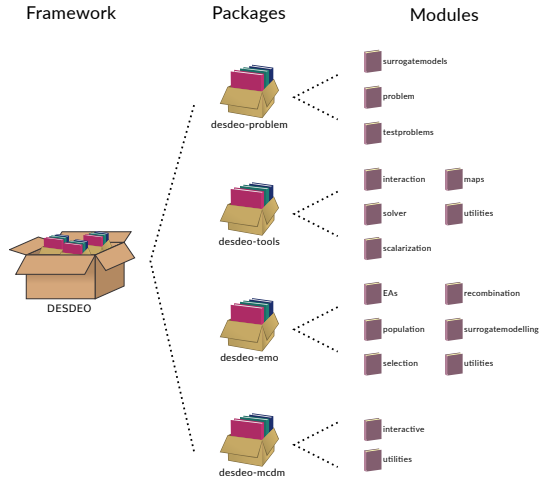
5 Bibliography

What is DESDEO about?

- The goal of DESDEO [4] is to provide a central hub for interactive multiobjective optimization methods.
- DESDEO is open source, modular, and implemented in Python. DESDEO is available on GitHub¹.
- Our aim is to provide tools for developers and researchers alike to start experimenting with interactive multiobjective optimization without having to start from scratch.
- Thus, DESDEO is built with a modular structure consisting of packages and modules.
- DESDEO's homepage: <https://desdeo.it.jyu.fi/>

¹<https://github.com/industrial-optimization-group/DESDEO> 

Structure of DESDEO



- The `desdeo-problem` package has tools and utilities to define multiobjective optimization problems.
- Problems based on data, problems based on analytical formulations, and problems based on surrogates are supported.
- Some pre-defined problems also exist, such as the DTLZ problems [5].
- GitHub page:
<https://github.com/industrial-optimization-group/desdeo-problem>

- The desdeo-tools package contains tools and utilities used across DESDEO.
- For instance, it provides ways to scalarize multiobjective optimization problems, provides tools to help interact with methods, and mappings to further transform multiobjective optimization problems.
 - For instance, to a so-called preference incorporated space [6].
- GitHub page:
<https://github.com/industrial-optimization-group/desdeo-tools>

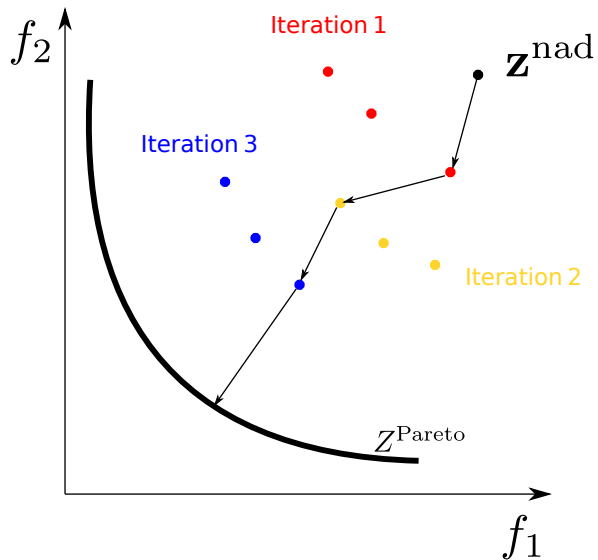
- The desdeo-emo package contains evolutionary methods, both interactive and non-interactive.
- Interactive versions of RVEA and NSGA-III, for instance.
 - How these methods have been made interactive is described in [7].
- Types of preferences accepted by evolutionary methods implemented are reference points, preferred solutions, non-preferred solutions, upper and lower bounds.
- GitHub page:
<https://github.com/industrial-optimization-group/desdeo-emo>

- In addition to the two methods listed, many other evolutionary methods are implemented in desdeo-emo.
- Evolutionary operators are easy to combine into new methods or to be swapped in existing methods.
- There are also some surrogate models available that are specifically evolved via the evolutionary algorithms present in desdeo-emo.

- The `desdeo-mcdm` package contains interactive methods based on scalarization.
- The type of preferences required by the methods in this package vary wildly.
- Some of these methods are presented in more detail on the following slides.
- GitHub page:
<https://github.com/industrial-optimization-group/desdeo-mcdm>

- A DM is shown a solution and is asked to classify its objectives up to five classes so that each objective may:
 - ① Stay as it is
 - ② Change freely
 - ③ Improve
 - ④ Improve until some value
 - ⑤ Worsen until some value
- Based on the classifications, new solutions are then computed, which the DM may explore.
- A new iteration starts with the classification of the objectives based on a solution the DM chose in the previous iteration.

- In the NAUTILUS family of methods, the general idea is to start from a non-Pareto optimal solution and approach the Pareto front gradually.
- This way, the DM does not have to make trade-offs and will always see improvements in each objective. This is why these methods are sometimes called trade-off free methods as well.
- The main benefit of NAUTILUS methods is that they help the DM avoid anchoring effects.



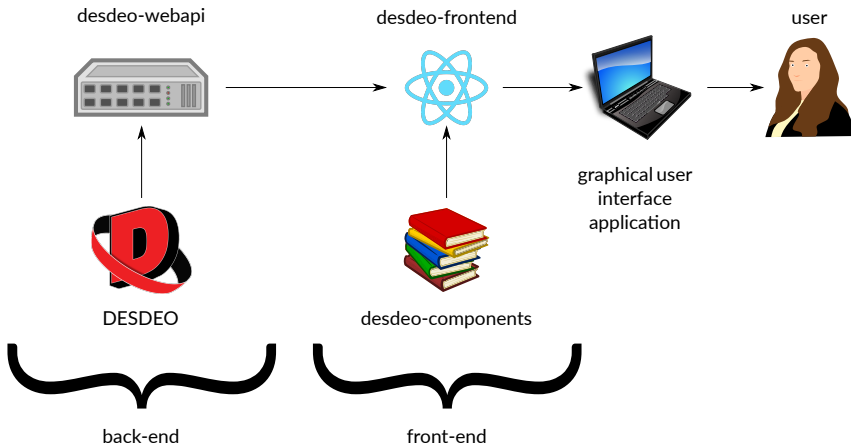
- Having both evolutionary and scalarization based methods in DESDEO allows the combination of the two in what is known as hybridization.
- For instance, we can calculate an approximation of the Pareto optimal front using some evolutionary method and then use that front in some scalarization based method.
 - For instance, we can use NSGA-III to calculate the front and then use that front in E-NAUTILUS.
- We are also able to change method in any iteration of an interactive method. This is useful if the DM wishes to change the type of preference they want to express during a solution process, for example.

Why user interfaces?

- Another big lack in the field of interactive multiobjective optimization is the absence of interfaces.
- Interactive methods vary and have some very specific needs which makes the development of interfaces challenging.
- Just like DESDEO, we would like the interfaces to be also modular and reusable. This is because apart from the methods themselves, DMs also have specific needs. Moreover, the problem itself might have some specific needs when it comes to visualizations.
- It is pretty clear at this point that we cannot develop just one interface and call it a day. We need to provide general tools instead that others can use to meet their own needs and goals.

- Thus far, we have started developing an interface for the methods found in DESDEO. Our aim is to build modular building blocks from which we construct our interface and which others are free to use to meet their own needs.
- We did not want the interfaces to be limited to the Python ecosystem, which is why we are building a web API that can be used to interface into the methods present in DESDEO.
- The interface itself is developed in Typescript as a React application and custom (interactive) visualizations are being built using D3.
- We are currently planning to redesign the whole interface that we have now.
- Here are some examples of what we have achieved:

The DESDEO webstack



- desdeo-webapi –
<https://github.com/industrial-optimization-group/desdeo-webapi>
- desdeo-components – <https://github.com/industrial-optimization-group/desdeo-components>
- desdeo-frontend –
<https://github.com/industrial-optimization-group/desdeo-frontend>
- TypeScript – <https://www.typescriptlang.org/>
- React – <https://reactjs.org/>
- D3.js – <https://d3js.org/>

What I want you to remember

- 1 Introduction
- 2 Background
- 3 DESDEO
- 4 What I want you to remember
- 5 Bibliography

Remember this

If you have any needs to either utilize or implement interactive multiobjective optimization methods, either scalarization based or evolutionary, remember the existence of DESDEO. It might already have what you are looking for. If not, it is very probable that it has at least some of the tools you need already implemented so that you do not have to start from scratch. Anybody is welcome to contribute to DESDEO.

- 1 Introduction
- 2 Background
- 3 DESDEO
- 4 What I want you to remember
- 5 Bibliography

- [1] Kaisa Miettinen. *Nonlinear multiobjective optimization*. Boston: Kluwer Academic Publishers, 1999.
- [2] 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.
- [3] J. Branke et al., eds. *Multiobjective Optimization: Interactive and Evolutionary Approaches*. Berlin: Springer, 2008.
- [4] G. Misitano et al. “DESDEO: The Modular and Open Source Framework for Interactive Multiobjective Optimization”. In: *IEEE Access* 9 (2021), pp. 148277–148295. DOI: 10.1109/ACCESS.2021.3123825.

- [5] K. Deb et al. “Scalable multi-objective optimization test problems”. en. In: *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*. Vol. 1. Honolulu, HI, USA: IEEE, 2002, pp. 825–830. DOI: 10.1109/CEC.2002.1007032. URL: <http://ieeexplore.ieee.org/document/1007032/>.
- [6] Bhupinder Singh Saini, Jussi Hakanen, and Kaisa Miettinen. “A New Paradigm in Interactive Evolutionary Multiobjective Optimization”. In: *Parallel Problem Solving from Nature – PPSN XVI*. Ed. by Thomas Bäck et al. Cham: Springer International Publishing, 2020, pp. 243–256.
- [7] Jussi Hakanen et al. “Connections of reference vectors and different types of preference information in interactive multiobjective evolutionary algorithms”. In: *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*. Athens, Greece: IEEE, 2016, pp. 1–8. DOI: 10.1109/SSCI.2016.7850220.

- [8] 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.
- [9] 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.
- [10] 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.

Thank you!

Questions?

Connect with me on LinkedIn:

<https://www.linkedin.com/in/giovanni-misitano-4b2b891b1>

