Prof. Dr. Kathrin Flaßkamp
M.Sc. Matthias Hoffmann
Systems Modeling and Simulation
Saarland University

Optimization – Project 2 – WS 21/22

Constrained and unconstrained optimization

Project task 2.1 – Multiobjective optimization methods

In the lecture you were presented four methods for calculating Pareto optimal points:

a) weighted-sum method

b) reference-point method

c) ϵ-constraint method

d) Pascoletti-Serafini scalarization

Implement at least three of these methods in the following way. The method-function should take as one input the hyperparameter (weights in WS, reference point in RPP, ...) and give out a `problem`-struct that can be fed to the `fmincon` call (for more information on this struct refer to the fmincon documentation). A call of the weighted-sum method function could look like this:

```
ws_problem = get_ws_problem(cost_fun, weights, ...)
ws_sol = fmincon(ws_problem)
```

For checking whether your algorithm works you can use the multiobjective optimization problem:

$$\min_{x \in \mathbb{R}^2} [x_1 \; x_2]^\intercal$$
$$\text{s.t. } ||x - [1 \; 1]^\intercal||_2^2 \leq 1$$

*Remark: The algorithms do **not** have to work for $k > 2$ objectives.*

Project task 2.2 – Computing the Pareto front

Consider the constrained multi-objective optimization problem

$$\min\ F(x) = \begin{pmatrix} f_1(x) \\ \\ f_2(x) \end{pmatrix} = \begin{pmatrix} \frac{1}{2}\left(\sqrt{1+(x_1+x_2)^2} + \sqrt{1+(x_1-x_2)^2} + x_1 - x_2\right) + \lambda e^{-(x_1-x_2)^2} \\ \frac{1}{2}\left(\sqrt{1+(x_1+x_2)^2} + \sqrt{1+(x_1-x_2)^2} - x_1 + x_2\right) + \lambda e^{-(x_1-x_2)^2} \end{pmatrix}$$

$$\text{s.t.} \quad -1.2 \leq x \leq 1.2,$$

where $\lambda = 0.6$.

a) Compute the utopia and nadir point.

b) Download the `get_*_params`-functions we prepared for you. Open them and read the code to understand how they generate the hyperparameters from the extreme points, utopia point, and nadir point and how to use them.

c) Implement the cost functions and constraints given in the problem above.

d) Generate Pareto fronts with the algorithms implemented in project task 2.1 and explain the resulting discretization of the Pareto front you see.

Please send your solutions to these tasks to matthias.hoffmann@uni-saarland.de by 17.01.2022 zipped in a file named "firstname_lastname_matrikelnr_2.zip". You can only pass if you work on all tasks and describe your problems in unsolved subtasks.