EVOLUTIONARY MULTIOBJECTIVE OPTIMIZATION BASED ON GAUSSIAN PROCESS MODELING

Miha Mlakar

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

Supervisor: Prof. Dr. Bogdan Filipič, Jožef Stefan Institute, Ljubljana, Slovenia

Evaluation Board:
Assist. Prof. Dr. Gregor Papa, Chair, Jožef Stefan Institute, Ljubljana, Slovenia
Prof. Dr. Zoran Bosnić, Member, University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia
Prof. Dr. Marjan Mernik, Member, University of Maribor, Faculty of Electrical Engineering and Computer Science, Maribor, Slovenia

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Miha Mlakar

EVOLUTIONARY MULTIOBJECTIVE OPTIMIZATION BASED ON GAUSSIAN PROCESS MODELING

Doctoral Dissertation

EVOLUCIJSKA VEČKRITERIJSKA OPTIMIZACIJA NA OSNOVI MODELIRANJA Z GAUSSOVIMI PROCESI

Doktorska disertacija

Supervisor: Prof. Dr. Bogdan Filipič

Ljubljana, Slovenia, April 2015

To Kristina and Maks and in memory of my father Janez

Acknowledgments

For successful completion of this dissertation I have to thank many individuals who made it possible and facilitated my research work. Regardless of the exposure and the order, I am sincerely grateful to everyone, even those who are not listed below.

First and foremost, I would like to thank my supervisor Prof. Bogdan Filipič. He provided me guidance, support, understanding, and professional assistance of the most valuable kind. His expertise and tendency towards perfection improved my research skills and prepared me for the future challenges.

I would also like to thank Tea Tušar for being my unofficial co-supervisor. Our extensive discussions and insightful explorations have been of great value for this work.

Special thanks to Prof. Marjan Mernik and Assist. Prof. Gregor Papa who have followed my doctoral research from the start and for their research inputs and helpful suggestions during my study. I am also grateful to Prof. Zoran Bosnić for his comments that improved this dissertation.

Thanks to all the participants of the AVN workshops, where most of this work was first presented. They always shared useful ideas and our discussions inspired my work. Further, I also wish to acknowledge the support of Robert Vertnik on the steel-casting use case.

I want to thank the Head of the Department of Intelligent Systems at the Jožef Stefan Institute Prof. Matjaž Gams and all my colleagues (especially those from the Computational Intelligence Group) for creating a pleasant working environment.

I thankfully acknowledge financial support from the Jožef Stefan Institute and the Slovene Research Agency for providing me with funding that made this dissertation possible.

Special thanks to my friend and colleague Dejan Petelin for sharing additional knowledge with me about modeling and for our coffee-break discussions in which we were "saving the world". I also want to thank my classmates and good friends Aleš Tavčar, Boštjan Kaluža and Lucas Benedičič for various research ideas and for helping me make my everyday commitments much easier.

Finally, many thanks to my wife Kristina, my family and friends, for their love, support, encouragement, and advice. They rejoiced with me at the successes and patiently encouraged and supported me in difficult moments.

Again, my sincere thanks to all.

Abstract

Multiobjective optimization is the process of simultaneously optimizing two or more conflicting objectives and is used for solving real-world optimization problems in various fields, from product design to process optimization.

One of the most effective ways of solving problems with multiple objectives is to use multiobjective evolutionary algorithms (MOEAs). MOEAs draw inspiration from adaptation processes occurring in nature. In order to find the best solutions, they perform numerous solution evaluations. If these solution evaluations are time-consuming, the optimization process can take a lot of time.

To obtain the results of such an optimization problem faster (or to obtain them in a reasonable amount of time), surrogate models can be used to approximate the objective functions of the problem. Instead of performing time-consuming exact evaluation to evaluate a solution, the solution can be approximated with a surrogate model. Using solution approximations can significantly accelerate the optimization process, but can also spoil the results if the solution approximations are inaccurate. When comparing approximated solutions, a solution can incorrectly appear to be dominated by inaccurate and over-optimistic approximations.

To reduce the possibility of incorrect comparisons, we propose new relations under uncertainty that, in addition to the approximated values, consider also the confidence intervals for the approximations. The relations under uncertainty were compared with the Pareto dominance relations in the experiments that confirmed that the use of the proposed relations reduces the possibility of incorrect comparisons.

We included the relations under uncertainty in a new MOEA called Differential Evolution for Multiobjective Optimization based on Gaussian Process modeling (GP-DEMO). GP-DEMO is based on Differential Evolution for Multiobjective Optimization (DEMO), a steady-state algorithm known to be very effective in solving numerical multiobjective optimization problems.

GP-DEMO was compared with DEMO and also with another surrogate-model-based MOEA called Generational Evolution Control (GEC). These algorithms were tested on 12 benchmark problems of different complexities and on two real-world problems – optimization of continuous steel casting and finding the correlation between a simulated and a measured electrocardiogram (ECG). The empirical analysis of their results showed that GP-DEMO and DEMO produce similar results, but GP-DEMO needs considerably less exact evaluations. In comparison to GEC, GP-DEMO achieves better results and the number of exact evaluations depends on the type of the optimization problem.

In order to determine when to use GP-DEMO instead of DEMO, we calculated for every test problem how long a single exact solution evaluation should last for the optimization times of GP-DEMO and DEMO to be equal. So for an arbitrary optimization problem we can, depending on the assessed complexity and the duration of a single exact solution evaluation, estimate which of the two algorithms is more suitable.

Povzetek

Z optimizacijskimi problemi se v različnih oblikah pogosto srečujemo v vsakdanjem življenju. Mnogi optimizacijski problemi zahtevajo sočasno optimizacijo več kriterijev, ki si pogosto nasprotujejo. Takim problemom pravimo večkriterijski optimizacijski problemi.

Eden izmed najučinkovitejših načinov reševanja takih problemov je uporaba večkriterijskih evolucijskih algoritmov. Evolucijski algoritmi temeljijo na posnemanju naravne evolucije in se zgledujejo po procesih prilagajanja, ki se dogajajo v naravi. Ti algoritmi med optimizacijskim procesom ovrednotijo veliko število rešitev. Če so ta ovrednotenja računsko zahtevna, lahko optimizacijski proces traja zelo dolgo.

Za pospešitev takega optimizacijskega procesa lahko namesto zamudnega eksaktnega vrednotenja rešitev del rešitev aproksimiramo z nadomestnimi modeli, ki jih dobimo z modeliranjem kriterijev optimizacijskega problema. V primerih, ko je zaradi kompleksnosti optimizacijskega problema kriterije težko modelirati, lahko nadomestni modeli vračajo netočne aproksimacije. Posledica tega je, da so pri primerjavi rešitev lahko dobre, eksaktno ovrednotene rešitve zavržene, ker izgledajo slabše kot napačno, preveč optimistično aproksimirane rešitve. To lahko upočasni iskanje najboljših rešitev oziroma celo prepreči algoritmu, da jih najde.

Za zmanjševanje števila napačnih primerjav smo v disertaciji definirali nove relacije ob negotovosti, ki za primerjavo rešitev poleg aproksimiranih vrednosti upoštevajo tudi intervale zaupanja, ki jih vračajo nekatere aproksimacijske metode. Relacije ob negotovosti smo primerjali z relacijami Pareto dominiranosti in pokazali, da relacije ob negotovosti zmanjšujejo možnost napačnih primerjav.

Relacije ob negotovosti smo nato vključili v nov algoritem, ki smo ga poimenovali Differential Evolution for Multiobjective Optimization based on Gaussian Process modeling (GP-DEMO). Ta algoritem temelji na algoritmu Differential Evolution for Multiobjective Optimization (DEMO), ki je zelo učinkovit pri reševanju numeričnih večkriterijskih optimizacijskih problemov.

Algoritem GP-DEMO smo primerjali z algoritmom DEMO in z algoritmom Generational Evolution Control (GEC), ki je tudi večkriterijski evolucijski algoritem in uporablja nadomestne modele. Algoritme smo primerjali na 12 testnih problemih in na dveh realnih problemih – optimizaciji kontinuirnega ulivanja jekla in iskanju korelacije med simuliranim in izmerjenim elektrokardiogramom (EKG). Analiza rezultatov je pokazala, da GP-DEMO in DEMO dosežeta podobne rezultate, le da GP-DEMO za to potrebuje manj eksaktnih ovrednotenj rešitev. GP-DEMO v primerjavi z algoritmom GEC doseže boljše rezultate, število eksaktno ovrednotenih rešitev pa je odvisno od tipa optimizacijskega problema.

Za vsak v disertaciji uporabljen testni optimizacijski problem smo izračunali, kako dolgo naj bi trajalo posamezno eksaktno ovrednotenje, da bi bilo trajanje optimizacijskega procesa za algoritma GP-DEMO in DEMO enako. Glede na to, da algoritma vračata primerljive rezulate, lahko glede na kompleksnost optimizacijskega problema in trajanje eksaktnega ovrednotenja rešitve za poljuben optimizacijski problem ocenimo, kateri algoritem je primerneje uporabiti.

Contents

List of Figures **xv**

List of Tables **xvii**

List of Algorithms **xix**

Abbreviations **xxi**

Symbols **xxiii**

1 Introduction **1**
1.1 Problem Formulation 1
1.2 Hypotheses and Goals 3
1.3 Methodology 4
1.4 Expected Scientific Contributions 4
1.5 Overview of the Dissertation 5

2 Background and Related Work **7**
2.1 Multiobjective Optimization 7
2.1.1 Hypervolume 9
2.2 Multiobjective Evolutionary Algorithms 10
2.2.1 The NSGA-II Environmental Selection Procedure 12
2.3 Differential Evolution for Multiobjective Optimization 13
2.3.1 The DE Algorithm 13
2.3.2 The DEMO Algorithm 16
2.4 Surrogate-Model-Based Optimization 16
2.4.1 Evolution Control 18
2.4.2 Surrogate-Model-Based Multiobjective Optimization 20
2.5 Comparing Solutions Under Uncertainty 22
2.6 Gaussian Process Models 24
2.6.1 Gaussian Process Modeling 25
2.6.2 Sparse Approximation 26
2.7 Random Forest Models 27

3 Relations for Comparing Solutions Under Uncertainty **31**
3.1 Relations without Uncertainty 31
3.2 Relations under Uncertainty 32
3.2.1 Relations under Uncertainty without Constraints 33
3.2.2 Relations under Uncertainty with Constraints 35
3.3 Comparing Solutions under Uncertainty 36
3.4 Empirical Proof of Concept 39
3.5 Comparing GP and RF Modeling Techniques 42

3.6 Discussion 45

4 The GP-DEMO Algorithm **47**
4.1 Algorithm Structure 47
4.2 Comparing Parents and Candidates under Uncertainty 49
4.3 Selection under Uncertainty 50

5 Numerical Evaluation **53**
5.1 Test Problems 53
5.1.1 Benchmark Problems 53
5.1.2 The Continuous Steel Casting Problem 53
5.1.3 The ECG Problem 55
5.2 Experimental Setup 56
5.3 The Generational Evolution Control (GEC) Algorithm 57
5.4 Results 59

6 Analysis of Results **63**
6.1 Comparing GP-DEMO, DEMO and GEC 63
6.1.1 Benchmark Problems 63
6.1.2 Real-World Problems 66
6.2 Statistics 67
6.3 Border Time 67
6.4 Discussion 67

7 Conclusions **71**
7.1 Summary 71
7.2 Scientific Contributions 72
7.3 Future Work 73

References **75**

Bibliography **81**

Biography **83**

List of Figures

Figure 1.1: A problem that can occur when comparing inaccurately approximated solutions: (a) approximated solutions, (b) approximated solutions and corresponding exactly evaluated solutions. 2

Figure 2.1: The decision and objective space of a MOP. Red points represent decision and objective vectors of the best solutions. 8
Figure 2.2: The objective space of a constrained multiobjective optimization problem. 9
Figure 2.3: Dominance relations and Pareto-optimal front. 10
Figure 2.4: Illustration of hypervolume for a two-dimensional MOP. 11
Figure 2.5: The NSGA-II environmental selection procedure: (a) non-dominated sorting, (b) crowding distance. 13
Figure 2.6: Visual representation of candidate creation with the DE/rand/1/bin strategy. 15
Figure 2.7: Generation-based evolution control. 18
Figure 2.8: Individual-based evolution control. 19
Figure 2.9: Adaptive evolution control. 20
Figure 2.10: Modeling with GP models: in addition to the mean value (prediction), we obtain a 95% confidence band for the underlying function f (shown in gray). 27
Figure 2.11: Modeling with SPGP models: pseudo-inputs (dots) are arbitrarily located, i.e., not a subset of the training data points (circles). 28

Figure 3.1: The bounding box of an objective vector. 33
Figure 3.2: Approximated solutions presented in the objective space using bounding boxes. 34

Figure 4.1: Selection procedure in GP-DEMO: (a) approximated first front, (b) corrected first front. 50

Figure 5.1: A scheme of the steel casting process. 54
Figure 5.2: A sketch of the GEC algorithm. 57
Figure 5.3: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the BNH problem. 59
Figure 5.4: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the OSY problem. 59
Figure 5.5: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the SRN problem. 60
Figure 5.6: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG1 problem. 60
Figure 5.7: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG2 problem. 60

Figure 5.8: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG3 problem. 60
Figure 5.9: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG4 problem. 60
Figure 5.10: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG5 problem. 60
Figure 5.11: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG6 problem. 61
Figure 5.12: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG7 problem. 61
Figure 5.13: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG8 problem. 61
Figure 5.14: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG9 problem. 61
Figure 5.15: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the continuous steel casting problem. 61
Figure 5.16: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the ECG problem. 61

List of Tables

Table 3.1: Relations without uncertainty. 37
Table 3.2: Relations under uncertainty without constraints. 37
Table 3.3: Relations under uncertainty with constraints. 37
Table 3.4: Comparison of the proposed relations with Pareto dominance relations on the Poloni problem (average values over 30 runs). 40
Table 3.5: Comparison of the proposed relations with Pareto dominance relations on the OSY problem (average values over 30 runs). 40
Table 3.6: Comparison of the proposed relations with Pareto dominance relations on the SRN problem (average values over 30 runs). 41
Table 3.7: Comparison of the proposed relations with Pareto dominance relations on the Poloni problem (average values over 30 runs). 43
Table 3.8: Comparison of the proposed relations with Pareto dominance relations on the OSY problem (average values over 30 runs). 43
Table 3.9: Comparison of the proposed relations with Pareto dominance relations on the SRN problem (average values over 30 runs). 44

Table 5.1: Input variables and their boundary constraints. 55
Table 5.2: Output variables, their bounds and desired values. 55
Table 5.3: Reference points for calculating the hypervolume values. 59

Table 6.1: Results on BNH, OSY and SRN test problems. 64
Table 6.2: Results on WFG test problems. 65
Table 6.3: Results on real-world test problems. 66
Table 6.4: The probability of no significant difference between hypervolumes. . . . 68
Table 6.5: Border times (in seconds) for the benchmark test problems. 69

List of Algorithms

Algorithm 2.1: NSGA-II 12
Algorithm 2.2: DE 14
Algorithm 2.3: Candidate creation with the DE/rand/1/bin strategy 15
Algorithm 2.4: DEMO 17
Algorithm 2.5: RF 29

Algorithm 4.1: GP-DEMO 48

Algorithm 5.1: GEC 58

Abbreviations

BB	...	bounding box
DE	...	differential evolution
DEMO	...	differential evolution for multiobjective optimization
EA	...	evolutionary algorithm
ECG	...	electrocardiogram
GEC	...	generational evolution control
GP	...	Gaussian process
GP-DEMO	...	differential evolution for multiobjective optimization based on Gaussian process modeling
MOEA	...	multiobjective evolutionary algorithm
MOP	...	multiobjective optimization problem
NSGA-II	...	non-dominated sorting genetic algorithm II
PCC	...	Pearson correlation coefficient
RF	...	random forest
SPGP	...	sparse Gaussian process using preudo-inputs
WFG	...	a suite of benchmark optimization problems by the Walking Fish Group

Symbols

$\mathbf{f}$	... optimization function
$\mathbf{x}$, $\mathbf{y}$	... decision vectors
X	... decision space
n	... dimensionality of the decision space
$\mathbf{z}$, $\mathbf{w}$	... objective vectors
Z	... objective space
m	... dimensionality of the objective space
$\mathcal{F}$	... feasible objective value region
$\mathcal{P}$	... parent population
$\mathcal{Q}$	... offspring population
$\mathcal{R}$	... union of parent and offspring populations
$\mathcal{S}$	... set of non-dominated solutions
$popSize$	... size of the population
CR	... crossover probability
F	... scaling factor
HV	... hypervolume
$\mathbf{p}$	... parent solution
$\mathbf{c}$	... candidate solution
N	... size of training data
D	... dimension of input vectors
$\mathbf{K}$	... covariance matrix
C	... covariance function
$\mathbf{\Theta}$	... vector of hyperparameters
$\mathcal{L}$	... likelihood
μ	... mean of the distribution
σ	... standard deviation
σ^2	... variance
$\mathbf{k}(\mathbf{x}^*)$	... vector of covariances between the test and the training cases
M	... size of the active set
$\boldsymbol{\varepsilon}$, $\boldsymbol{\delta}$	... confidence vectors
$\underline{\boldsymbol{\varepsilon}}$	... lower-bound confidence vector
$\overline{\boldsymbol{\varepsilon}}$	... upper-bound confidence vector
$\prec$	... Pareto dominance relation
$\preceq$	... weak Pareto dominance relation
$\prec\prec$	... strict Pareto dominance relation
$\not\preceq$	... Pareto non-dominance relation
$\prec_c$	... constrained dominance relation
$\|\|$	... incomparability relation
$\prec_u$	... probable Pareto dominance relation
$\not\preceq_u$	... probable Pareto non-dominance relation

$||_u$ ... probable incomparability relation
$\sim_u$ ... undetermined relation
$\prec_{uc}$ ... probable constrained dominance relation
$\not\prec_{uc}$ ... probable constrained non-dominance relation
$||_{uc}$ ... probable constrained incomparability relation
$\sim_{uc}$ ... undetermined constrained relation

Chapter 1

Introduction

This chapter shortly describes the problems addressed in this dissertation and presents the motivation for our research. Further it presents the hypotheses and goals defined at the beginning of our work and states the expected scientific contributions. The chapter concludes with an overview of the dissertation where each chapter is briefly described to introduce the contents of the dissertation.

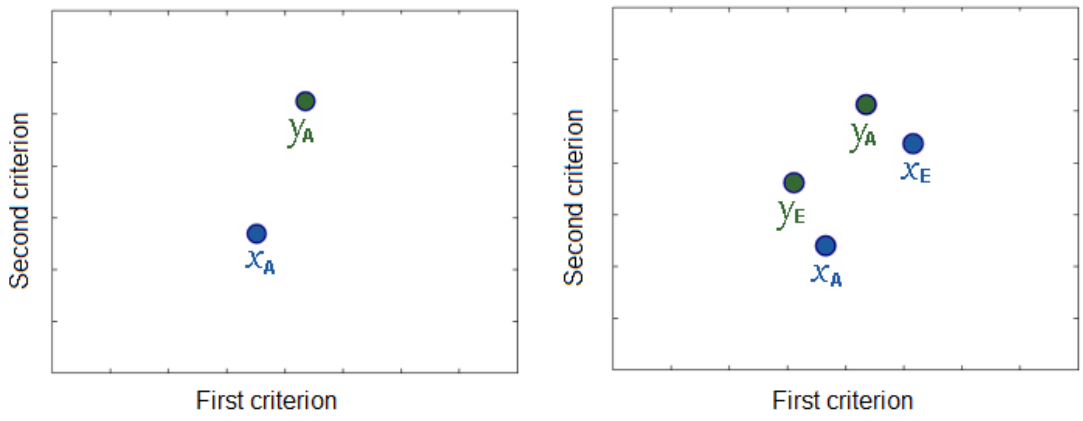
1.1 Problem Formulation

Optimization problems are present in everyday life and can be found in various fields, from product design and process optimization to financial applications. Many of these optimization problems require simultaneous optimization of multiple, often conflicting criteria (or objectives). These problems are called multiobjective optimization problems. The solution to such a problem is not a single one, but a family of solutions, known as the Pareto-optimal set. This set gives the decision-maker an insight into the characteristics of the problem before a single final solution is chosen.

One of the most effective ways of solving multiobjective optimization problems is to use multiobjective evolutionary algorithms (MOEAs) (Deb, 2001). MOEAs are population-based algorithms that draw inspiration from optimization processes that occur in nature. In order to find a Pareto-optimal set, a lot of different solutions have to be assessed (evaluated) during the optimization process. For some optimization problems these solution evaluations can be costly, dangerous or computationally expensive. In such cases the goal is to reduce the number of exactly evaluated solutions, but still find the best solutions. In this dissertation we focus on computationally expensive multiobjective optimization problems, where a single solution evaluation takes a lot of time.

In order to obtain the results of such an optimization problem more quickly (or obtain them in a reasonable amount of time), we can use surrogate models in the optimization process. Surrogate models mimic the behavior of the simulation model used for exact solution evaluations. A surrogate model is constructed based on modeling the response of the simulator from a limited number of previously exactly evaluated solutions. The idea is to build a surrogate model that is as accurate as possible, using as few simulation evaluations as possible. Then instead of using a time-consuming exact evaluation, a solution can be approximated with the surrogate model. Since an individual solution approximation is (much) faster, the whole optimization process can be accelerated.

However, the use of the surrogate models in optimization has two drawbacks. The first one is that the time needed to create and update the surrogate models prolongs the optimization process. If exact solution evaluations are quick, it can happen that the surrogate-model-based optimization takes longer than the optimization without surrogates.



(a) (b)

Figure 1.1: A problem that can occur when comparing inaccurately approximated solutions: (a) approximated solutions, (b) approximated solutions and corresponding exactly evaluated solutions.

The second drawback can be observed if the optimization problem is very complex and therefore hard to model. This results in imprecise surrogate models and the solutions approximated with these models can be inaccurate. As a consequence, good solutions can be discarded from the population because they incorrectly appear to be dominated by the inaccurate and over-optimistic approximations. This can slow the optimization process or even prevent the algorithm from finding the best solutions.

When approximating solutions, some surrogate models, in addition to the approximated values, provide also a distribution, from which the confidence interval of the approximation can be calculated. This confidence interval indicates the region in which the exactly evaluated solution is most likely to appear. The confidence interval width indicates the certainty of the approximation. If the confidence interval is narrow, we can be more certain about the approximation and vice versa.

The solutions represented with confidence intervals, where exact objective values are unknown, are called solutions under uncertainty and the problems with uncertain solutions are called uncertain optimization problems. The source of uncertainty can, in addition to the objective function approximations, be noisy, robust, or time-varying objective functions (Jin & Branke, 2005).

Regardless of the source of uncertainty, it is better if an algorithm takes uncertainty into account. Since this uncertainty offers additional information, it can be effectively used when comparing solutions. Comparing only approximated values without considering uncertainty can for inaccurate approximations cause incorrect comparisons. Such an example is shown in Figure 1.1. Solutions $\mathbf{x}_A$ and $\mathbf{y}_A$ represent approximated values for solutions $\mathbf{x}$ and $\mathbf{y}$ (Figure 1.1 (a)). If we compare approximations and presume that this is a minimization problem, we can see that solutions $\mathbf{x}$ is better than solution $\mathbf{y}$. But if we would exactly evaluate these solutions, we would get different objective values represented with $\mathbf{x}_E$ and $\mathbf{y}_E$ (Figure 1.1 (b)). The comparison of these two solutions shows that solution $\mathbf{y}$ is better than solution $\mathbf{x}$. Therefore comparing only approximated solutions can result in incorrect relations between solutions.

To prevent these unwanted effects, we propose new relations for comparing solutions under uncertainty where, in addition to the approximated values of a solution, their con-

fidence intervals are considered. These relations extend Pareto dominance relations and cover all possible combinations that can occur when comparing solutions represented with confidence intervals. New relations also take into consideration the feasibility of solutions including the uncertainty of feasibility due to the uncertainty of solutions. During the optimization process, some solutions are exactly evaluated and others approximated; therefore, the relations under uncertainty also cover the comparison of approximated solutions with exactly evaluated solutions. The relations under uncertainty can be used to compare solutions in any multiobjective optimization algorithm dealing with solutions represented with confidence intervals.

To check if the use of the proposed relations under uncertainty reduces the possibility of incorrect comparisons due to the inaccurate approximations, we compare them with Pareto dominance relations. The comparison is performed with various surrogate models and on different benchmark problems.

Based on the relations under uncertainty we propose a new surrogate-model-based multiobjective evolutionary algorithm, called Differential Evolution for Multiobjective Optimization based on Gaussian Process modeling (GP-DEMO). This algorithm is an extension of the Differential Evolution for Multiobjective Optimization (DEMO) algorithm (Robič & Filipič, 2005), which uses differential evolution to effectively solve numerical multiobjective optimization problems and, in addition, emphasizes the variation operators. As a surrogate model Gaussian Process modeling is employed to find approximate solution values together with their confidence intervals. All solution comparisons in GP-DEMO are performed with new relations under uncertainty. As it works on the same principles as DEMO, the quality of its results is expected to be similar to the results of DEMO, but with fewer exact solution evaluations.

To thoroughly test the GP-DEMO algorithm, we compare it with another surrogate-model-based algorithm and with DEMO on several benchmark and two real-world optimization problems.

1.2 Hypotheses and Goals

This dissertation investigates the following two hypotheses:

(**H1**) The proposed relations under uncertainty reduce the possibility of incorrect comparisons due to inaccurately approximated solutions.

(**H2**) The proposed surrogate-model-based multiobjective evolutionary algorithm (GP-DEMO) in comparison to other MOEAs produces comparable results with fewer exact evaluations of the original objective functions.

In accordance with the purpose of the doctoral dissertation and to verify the hypotheses, the following goals are expected to be achieved:

- improvement of the surrogate-model-based multiobjective optimization methodology in terms of achieving high-quality results with as few exact solution evaluations as possible,
- providing a survey of the state-of-the-art approaches on comparing solutions under uncertainty and on computationally expensive optimization problems,
- analyzing the pros and cons of the Pareto dominance relations for comparing solutions under uncertainty,

- determining the impact of the proposed relations under uncertainty on reducing the possibility for a surrogate-model-based algorithm to get stuck in a local optimum or find suboptimal solutions due to inaccurate approximations,
- systematic evaluation and comparison of results from several MOEAs with and without surrogate models,
- analyzing the potential of the proposed approach in real-world applications.

1.3 Methodology

To provide the empirical support for the hypotheses and meet the goals, we use a standard methodological framework and various methods from the field of machine learning and multiobjective optimization. The methodology used in this dissertation includes:

- Literature survey. The existing work in the field of surrogate-model-based evolutionary algorithms and also the techniques for comparing solutions represented with confidence intervals will be reviewed.
- The design and definition of new relations under uncertainty. The relations will cover all possible scenarios for comparing feasible and infeasible solutions represented with confidence intervals.
- Design and implementation of an algorithm for measuring the number of incorrect solution comparisons. This algorithm will be used for comparing the relations under uncertainty with Pareto dominance relations.
- Design and implementation of a new surrogate-model-based MOEA. By combining our ideas with existing approaches, the new algorithm based on relations under uncertainty will be designed.
- Empirical evaluation of various algorithms on benchmark and real-world problems. The algorithms will be evaluated and their performance analyzed and visually presented. Moreover, the comparison will be made using various measures.
- Statistical analysis of results. Due to stochastic nature of the evolutionary algorithms, the t-test statistical hypothesis test will be used to determine if the algorithms differ significantly.

1.4 Expected Scientific Contributions

Fulfilling the goals and providing empirical support for the hypotheses requires a detailed examination of the following research areas:

- evolutionary computation,
- multiobjective optimization,
- surrogate-model-based optimization,
- surrogate modeling and solution approximations,
- comparison of solutions under uncertainty.

Scientific contributions of the dissertation are expected at the intersection of these research areas. We foresee the following scientific contributions:

- Defining new relations for comparing solutions under uncertainty that are suited for comparing solutions represented with approximated values and confidence intervals.
- Confirming that the new relations under uncertainty, in comparison to Pareto dominance relations, reduce the possibility of incorrect solution comparisons.
- Designing a surrogate-model-based MOEA called GP-DEMO. This algorithm is suited for solving multiobjective optimization problems where solution evaluations are computationally expensive.
- Setting guidelines on when, depending on the type of optimization problem, the GP-DEMO algorithm should be used.

1.5 Overview of the Dissertation

Chapter 2 describes the algorithms, methods and approaches related to our research. It starts with an explanation of basic concepts of multiobjective optimization and MOEAs. Next, the DEMO algorithm with its advantages and disadvantages is described. Since GP-DEMO uses surrogate models and compares solutions under uncertainty, these fields are thoroughly described and various approaches and concepts are presented. In addition, two often used modeling techniques, Gaussian Process (GP) modeling and Random Forest (RF) modeling, are presented.

In Chapter 3, the proposed relations under uncertainty are described. They extend the Pareto dominance relations, so the latter are presented as well. The process of comparing solutions under uncertainty is demonstrated for unconstrained and constrained multiobjective optimization problems. Further, the hypothesis that using the new relations under uncertainty reduces the number of incorrect comparisons is tested. To test this hypothesis, we compared the relations under uncertainty with Pareto dominance relations. This experiment was also used to compare two modeling techniques, RF and GP. The discussion of results from both comparisons concludes Chapter 3.

Chapter 4 describes the newly designed surrogate-model-based evolutionary algorithm GP-DEMO. This algorithm extends the DEMO and uses relations under uncertainty to compare the solutions. The outline of the GP-DEMO algorithm with its pros and cons is presented. Because of dealing with uncertain solution values, the procedure of comparing the parent and candidate solutions and the selection procedure are modified. For both procedures the implementation and properties are described in detail.

Chapter 5 starts with the description of benchmark and real-world problems that were used for testing. Next, the experimental setup is presented with the interpretation of the settings selected. For the sake of fair evaluation of the GP-DEMO algorithm, we compared it with another surrogate-model-based algorithm GEC. Its structure and characteristics are also fully described in this chapter. The chapter ends with the presentation of the results gained with all algorithms. For every algorithm the results are presented with tables and the best fronts are drawn.

The analysis of the results is performed in Chapter 6. The algorithms are pairwise compared and the statistical analyses are presented. To determine which algorithm is more suitable for a certain type of optimization problems, the optimization times and the quality of results were combined to develop the strategy that, depending on the complexity and exact evaluation times, suggests the appropriate algorithm.

Chapter 7 concludes the dissertation with the overview of the work done, the scientific contributions of the dissertation and the ideas for future work.

Chapter 2

Background and Related Work

In this chapter we explain the main concepts, algorithms and measures used in this dissertation, and review the state of the art in the fields of surrogate-model-based multiobjective optimization and comparison of solutions under uncertainty.

We start with explaining the basic multiobjective optimization principles and dominance relation between solutions with conflicting objectives. Then, we describe the multiobjective evolutionary algorithms (MOEAs) and identify the reasons for their suitability for solving multiobjective optimization problems. We present MOEAs basic principles and outline a very popular MOEA algorithm, Non-dominated Sorting Genetic Algorithm II (NSGA-II), and its environmental selection procedure. This procedure is used also in the DEMO algorithm and with some modifications also in the GP-DEMO algorithm.

Further, we describe the algorithms Differential Evolution (DE) and DEMO, since DEMO is used as the base for GP-DEMO.

Since GP-DEMO uses surrogate models and relations for comparing solutions under uncertainty, we review these research fields and describe the methods and techniques used by other authors.

The chapter concludes with presenting two different modeling techniques that, in addition to the approximations, return also the confidence in the approximations. These techniques are GP modeling and RF modeling. Both techniques were used in the experiments in this dissertation and GP modeling is also used in GP-DEMO.

2.1 Multiobjective Optimization

In multiobjective optimization, we wish to simultaneously optimize several (possibly conflicting) objectives. The scenario considered in this dissertation involves an arbitrary multiobjective optimization problem (MOP) with m objectives which are, without loss of generality, all to be minimized and all equally important, i.e., no additional knowledge about the problem is available. A MOP consists of finding the minimum of the function

$$\mathbf{f} : X \to Z \tag{2.1}$$

$$\mathbf{f} : (x_1, ..., x_n) \mapsto (f_1(x_1, ..., x_n), ..., f_m(x_1, ..., x_n)), \tag{2.2}$$

subject to:

- boundary constraints:

$$x_i^{min} \leq x_i \leq x_i^{max}; \ (i = 1, ..., n), \tag{2.3}$$

- constraints on decision variables:

$$g_j(\mathbf{x}) \leq 0;\ (j = 1, 2, ...k), \tag{2.4}$$

- constraints on objectives:

$$h_j(\mathbf{f}(\mathbf{x})) \leq 0;\ (j = 1, 2, ...l), \tag{2.5}$$

where n is the number of variables, m is the number of objectives, k is the number of constraints on decision variables, and l is the number of constraints on objectives. Each solution $\mathbf{x} = (x_1, ..., x_n) \in X$ is called a *decision vector*, while the corresponding element $\mathbf{z} = \mathbf{f}(\mathbf{x}) \in Z$ is an *objective vector* (see Figure 2.1).

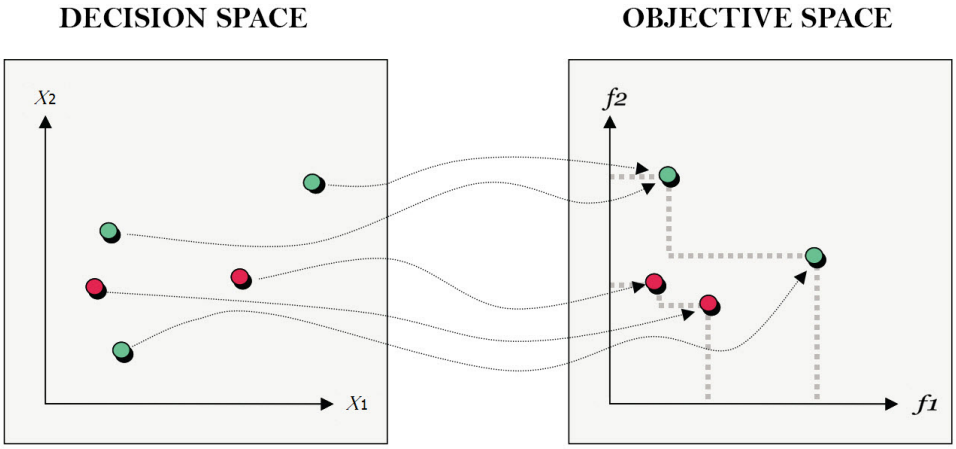


Figure 2.1: The decision and objective space of a MOP. Red points represent decision and objective vectors of the best solutions.

The boundary constraints define the *search region* of an optimization problem by setting the lower bounds $\mathbf{x}^{\mathbf{min}}$ and the upper bounds $\mathbf{x}^{\mathbf{max}}$ for the variables. Inside the search region, the constraints on decision variables further define the feasibility of solutions. An example of such a constraint would be that the sum of two variables should not exceed a predefined value. Since these constraints can be complex, the region they define can also be complex. As an example, the red contour in Figure 2.2 that represents this region is drawn as a complex shape. The constraints on objectives limit the feasibility of the objective values. An example of a constraint on objectives would be to set a maximum budget and a minimum top speed in the optimization problem of finding a fast and cheap car. The constraints on objectives are typically not very complex, hence the region defined by these constraints is fairly simple. We call this region *feasible objective value region*; in Figure 2.2 it is surrounded by the blue and green lines.

If all constraints are satisfied, the solution is called *feasible*; otherwise it is *infeasible*. All feasible solutions in the decision space constitute the *feasible region*. The mapping of this region in the objective space is called *feasible region image* and this region is marked with black hatching in Figure 2.2. The feasible solutions of a MOP that are the best with regard to all objectives create a front of solutions called the *Pareto optimal front*, which is indicated by the green line in Figure 2.2.

In multiobjective optimization the comparison of two solutions is based on the concept of domination (Deb, 2001). Solution $\mathbf{x}^1$ dominates solution $\mathbf{x}^2$, i.e., $(\mathbf{x}^1 \prec \mathbf{x}^2)$, if:

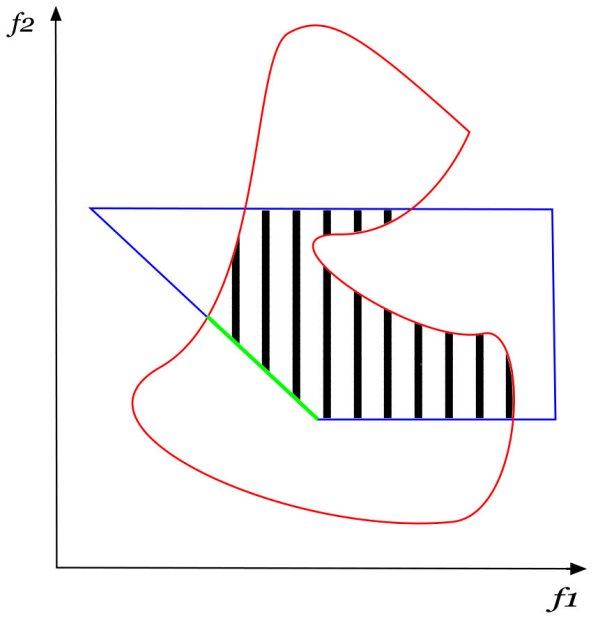


Figure 2.2: The objective space of a constrained multiobjective optimization problem.

1. $\mathbf{x}^1$ is not worse than $\mathbf{x}^2$ in any of the objectives,
2. $\mathbf{x}^1$ is strictly better than $\mathbf{x}^2$ in at least one objective.

When comparing two solutions, one solution can be better (or not worse) than the other solution in all objectives, or better in some objectives and worse in others. In the first case, the first solution is better/dominates the second solution (the second solution is dominated by the first one), while in the second case no solution is better since the two solutions are incomparable. If a solution is not dominated by any other solution, it is called a non-dominated solution. When all feasible solutions are taken into consideration, the non-dominated solutions are Pareto-optimal solutions forming Pareto-optimal front (see Figure 2.3).

The concept of domination between two solutions makes it possible to compare solutions with multiple objectives and is used by most MOEAs.

Since dominated solutions do not improve the results of the MOP, the task of the multiobjective optimization is to find a set of non-dominated solutions that form Pareto-optimal front. Although the goal is to find Pareto-optimal solutions, the obtained non-dominated solutions might not be Pareto-optimal, due to the complexity of the problem (algorithm cannot find them) or limited number of available solution evaluations.

In addition to finding solutions as close as possible to the Pareto-optimal front, another goal of the multiobjective optimization algorithms is to find solutions that are as diverse as possible. The spread and distribution of solutions on the non-dominated front assure a good set of trade-off solutions among objectives.

2.1.1 Hypervolume

The hypervolume indicator, originally proposed in (Zitzler & Thiele, 1998), is one of the most frequently used quality indicators to asses the performance of multiobjective optimizers. It can be used to compare non-dominated fronts or, if Pareto-optimal front is known, evaluate the closeness of a non-dominated front to it. The hypervolume value is obtained by calculating the volume covered by a set of non-dominated solutions $\mathcal{S}$. More precisely,

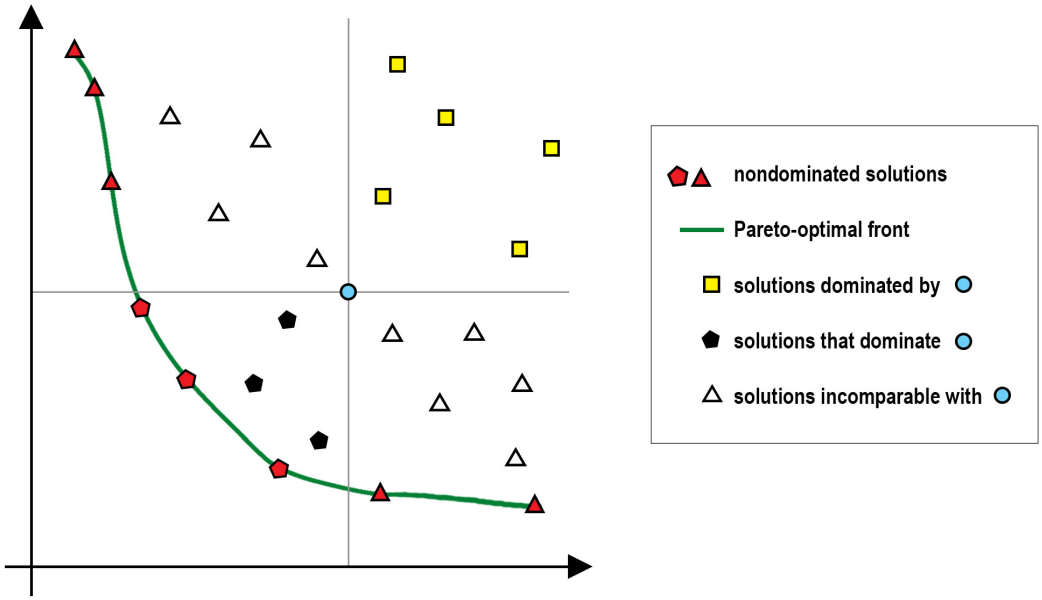


Figure 2.3: Dominance relations and Pareto-optimal front.

it calculates the volume between the solutions and a reference point. The coordinates of the reference point are usually set as the maximum feasible values of the objectives or as the maximum values of the objectives in the set of nondominated solutions. In this dissertation the reference point was determined gradually in a way that it covered all obtained non-dominated solutions and that the region it defined was as small as possible.

The procedure for calculating hypervolume constructs a hypercube v_i for each solution $i \in \mathcal{S}$, where the diagonal corners of the hypercube are the reference point and the solution. Hypervolume HV is the volume of the union of hypercubes:

$$HV = volume(\bigcup_{i}^{|\mathcal{P}|} v_i) \tag{2.6}$$

An illustration of calculated hypervolume is shown in Figure 2.4.

Hypervolume can be calculated during the optimization process and can thus measure the quality of the current non-dominated solutions. By measuring hypervolume, we can detect if the algorithm has found the Pareto-optimal front (if it is known), or check if hypervolume is not changing anymore and stop the optimization process. But it has to be noted that the calculation of hypervolume is computationally expensive, especially in MOPs with many objectives, so frequent calculation of hypervolume can notably increase the optimization time.

2.2 Multiobjective Evolutionary Algorithms

The term evolutionary algorithm (EA) stands for a class of stochastic optimization methods that imitate the principles of the Darwinian theory of evolution. The EAs search for the optimal solutions by simultaneously processing a set of solutions called population. Using strong simplifications, solutions in the population are modified using two basic principles:

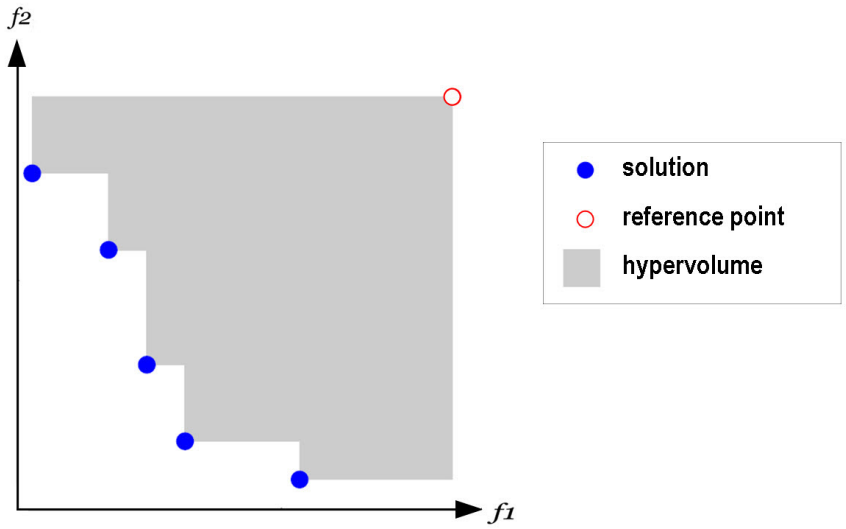


Figure 2.4: Illustration of hypervolume for a two-dimensional MOP.

selection and variation. While selection of the best solutions mimics the competition for reproduction and survival of the fittest, the other principle, variation, imitates the natural capability of creating "new" living beings by means of recombination (crossover) and mutation. By applying selection, crossover and mutation to a population of individuals, they create better and better offspring individuals and thus drive the optimization process towards optimal solutions.

Although the underlying mechanisms are simple, these algorithms have proven themselves as general, robust and powerful search algorithms (Back, Hammel, & Schwefel, 1997). They are (i) often applied to problems with very large and highly complex search spaces, (ii) especially applicable for parallelization, and (iii) known to be robust and capable of handling all types of optimization problems, including problems with discrete search space.

Evolutionary algorithms are very suitable for solving MOPs since they work with a population of solutions, which means they should be able to capture a population of Pareto-optimal solutions in a single optimization run. In addition, MOEAs do not require parameters, such as weight vectors, since the multiobjective problem is not transformed into a single-objective problem. Moreover, they are able to simultaneously find a diverse set of multiple non-dominated solutions.

Probably the most widely-used MOEA is the Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Deb, Pratap, Agarwal, & Meyarivan, 2002). This algorithm was applied to many real-world problems and contains all the basic concepts of the evolutionary algorithms, i.e., selection, crossover and mutation. NSGA-II is outlined in Algorithm 2.1.

NSGA-II starts by initializing the first parent population P_0 with randomly created individuals and sets the first offspring population Q_0 to be empty. While the stopping criterion is not met, in every generation the population Q_t is created from the population P_t. To create the offspring population, the pairs of solutions are selected from P_t using crowded tournament selection (Deb et al., 2002). These solutions are then combined using crossover and further modified using mutation. The new solutions are then added to Q_t. After filling the offspring population Q_t, the individuals from P_t and Q_t are joined to create a union of solutions R_t. From R_t the new population P_{t+1} is filled with the best solutions from R_t. The best solutions from R_t are chosen according to the environmental selection

Algorithm 2.1: NSGA-II

Result: A set of non-dominated solutions

Create the initial parent population $\mathcal{P}_0$ of random individuals;
Evaluate the solutions in $\mathcal{P}_0$;
Set the first offspring population $\mathcal{Q}_0$ to be empty;
while *stopping criterion not met* **do**
  for each pair of individuals $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$ from $\mathcal{P}_t$ **do**
    Modify the individuals $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$ with crossover;
    Modify the individuals $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$ with mutation;
    Add individuals $\mathbf{p}^i$ and $\mathbf{p}^{i+1}$ to the population $\mathcal{Q}_t$;
  end
  Combine parent and offspring populations $\mathcal{R}_t = \mathcal{P}_t \cup \mathcal{Q}_t$;
  Apply an environmental selection procedure on $\mathcal{R}_t$ to get the best *popSize* solutions (see Subsection 2.2.1);
  Use these *popSize* best solutions to create population $\mathcal{P}_{t+1}$;
end

procedure based on non-dominated sorting and the crowding distance metric. Since this environmental selection procedure is used also in DEMO and GP-DEMO algorithms, we describe it in greater detail.

2.2.1 The NSGA-II Environmental Selection Procedure

By combining current and offspring solutions, the union is usually enlarged after every generation. In order to maintain constant population size throughout the optimization process, the environmental selection procedure has to be applied to discard some solutions. The environmental selection procedure is based on the *non-dominated sorting* and the *crowding distance metric*.

For the non-dominated sorting the individuals in the population are ranked according to the number of individuals that dominate them. All non-dominated individuals are allocated into the first front and then the non-dominated sorting is applied again to the remaining individuals. In this way, we get a unique sequence of fronts, where individuals from precedent fronts are preferred to those from subsequent fronts. The new (reduced) population is filled with the individuals from the best fronts. If a front cannot fit into population entirely, the individuals from this front are further ranked according to the crowding distance metric.

The crowding distance metric calculates distances between neighboring solutions and thus identifies and promotes individuals from the less crowded regions. For the individual $\mathbf{x}^i$, the distance $d_j(\mathbf{x}^i)$ between its neighboring individuals $\mathbf{x}^{i-}$ and $\mathbf{x}^{i+}$ for objective j is calculated as:

$$d_j(\mathbf{x}^i) = \frac{f_j(\mathbf{x}^{i+}) - f_j(\mathbf{x}^{i-})}{f_j^{max} - f_j^{min}}, \tag{2.7}$$

where f_j^{max} and f_j^{min} denote the maximum and minimum value of objective j. The two solutions with extreme values of f_j are assigned the largest possible d_j, which ensures that the spread of solutions is as wide as possible. The crowding distance for solution $\mathbf{x}^i$ is then defined as:

$$c(\mathbf{x}^i) = \sum_{j=1}^{m} d_j(\mathbf{x}^i). \tag{2.8}$$

After calculating the crowding distance for all solutions on the selected front, the solutions with the largest crowding distance are included in the next population.

An example of carrying out the environmental selection can be seen in Figure 2.5. The example presents the case where the population size is 7. After the non-dominated sorting (Figure 2.5 (a)) the whole first front and three solutions from the second front are added to the next population. To determine which three solutions from the second front are added, the crowding distances for solutions on this front are calculated (Figure 2.5 (b) shows the calculation of the crowding distance metric for solution $\mathbf{x}^i$). The solutions marked with black dots have the largest crowding distances and are added to the population. Other solutions are discarded.

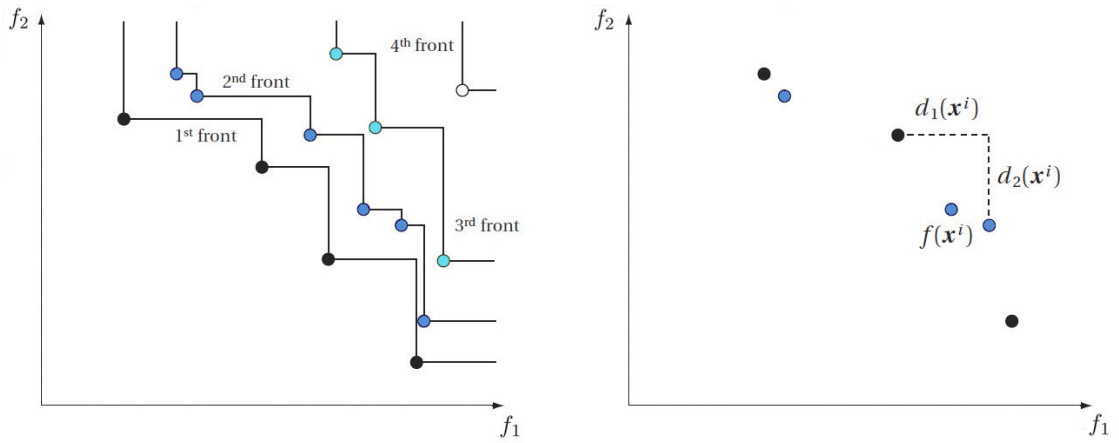


Figure 2.5: The NSGA-II environmental selection procedure: (a) non-dominated sorting, (b) crowding distance.

2.3 Differential Evolution for Multiobjective Optimization

In this section we present the DEMO algorithm that is further used as a base for the GP-DEMO. Since the basic mechanisms for creating and mutating solutions are taken from the single-objective DE algorithm, the latter are presented first.

2.3.1 The DE Algorithm

DE is an evolutionary algorithm proposed by Price and Storn (1997). It is a simple population-based algorithm that encodes solutions as vectors. For creating new solutions from the existing ones the algorithm uses operations such as vector addition, scalar multiplication and exchange of components (crossover).

DE, as evolutionary algorithms in general, starts with a population of randomly chosen solutions. From this initial population better and better solutions are obtained during the optimization process. In every generation for every solution (called parent) in the population $\mathcal{P}$, a so-called candidate solution is constructed using one of the many possible

strategies. After creating the candidate, it is evaluated and compared to its parent. If the candidate is better than or equal to its parent, it replaces the parent in the population. Otherwise, the candidate is discarded. After creating, evaluating and comparing all the candidates, the solutions in the population $\mathcal{P}$ are randomly enumerated so that the order of parents changes. When the stopping criterion is met, the best solution from $\mathcal{P}$ is returned. The outline of the DE algorithm is shown in Algorithm 2.2.

Algorithm 2.2: DE

Result: The best solution from $\mathcal{P}$

```
Create the initial population P_0 of random individuals;
Evaluate the solutions in P_0;
while stopping criterion not met do
    for each individual p^i (i = 1, ..., popSize) from P do
        Create candidate c from parent p^i;
        Evaluate c;
        if c is better than p^i then
            Replace p^i with c;
        end
        else
            Discard c;
        end
    end
    Randomly enumerate the individuals in P;
end
```

The creation of candidates in DE follows one of the so-called DE-strategies. DE strategies are written using the DE/x/y/z notation, where x represents the method of selecting the first solution $\mathbf{x}^{i_1}$, which can be selected randomly (rand) or as the best vector so far (best); y is the number of difference vectors used; and z defines the type of crossover which can be binomial (bin) or exponential (exp) (Price, Storn, & Lampinen, 2006).

In this dissertation we use the DE/rand/1/bin strategy. This strategy is most frequently used and the DE using it is also referred to as classic DE (Price et al., 2006). Algorithm 2.3 outlines the candidate creation using this strategy, and Figure 2.6 shows its visual representation.

The creation of a candidate using this strategy is the following. From the current population $\mathcal{P}$ and for the current parent $\mathbf{x}^i$, the candidate $\mathbf{c}$ is constructed using three randomly chosen solutions $\mathbf{x}^{i_1}$, $\mathbf{x}^{i_2}$ and $\mathbf{x}^{i_3}$:

$$\mathbf{c} = \mathbf{x}^{i_1} + F(\mathbf{x}^{i_2} - \mathbf{x}^{i_3}), \tag{2.9}$$

where i, i_1, i_2 and i_3 are pairwise different and F is a scaling factor for the difference vector $\mathbf{x}^{i_2} - \mathbf{x}^{i_3}$. This step is often referred to as *mutation*. Following the mutation, the candidate is subject to *binomial crossover* with its parent. During binomial crossover some components of the parent are copied to the corresponding positions in the candidate. Let l be a random integer from $(1, \ldots, n)$, where n is the number of variables. After binomial crossover, each candidate's component c_k is equal to:

$$c_k = \begin{cases} c_k & \text{if } k = l \text{ or } r_k \leq CR, \\ x_k^i & \text{otherwise,} \end{cases} \tag{2.10}$$

where each r_k is chosen randomly from the [0,1] interval, and CR is the crossover probability used to determine the influence of the parent. Smaller CR means that the candidate inherits most components from its parent. But if $CR = 0$, the candidate does not inherit all components, since the component c_l always remains intact for a randomly chosen l.

Algorithm 2.3: Candidate creation with the DE/rand/1/bin strategy

Result: Candidate $\mathbf{c}$

for *solution* $\mathbf{x}^i$ (parent) from the population $\mathcal{P}$ **do**
  Randomly choose three solutions $\mathbf{x}^{i_1}$, $\mathbf{x}^{i_2}$ and $\mathbf{x}^{i_3}$ from $\mathcal{P}$, where i, i_1, i_2 and i_3 are pairwise different;
  Calculate candidate $\mathbf{c}$ as $\mathbf{c} = \mathbf{x}^{i_1} + F(\mathbf{x}^{i_2} - \mathbf{x}^{i_3})$;
  Modify $\mathbf{c}$ by binomial crossover with $\mathbf{x}^i$ using probability CR;
  Repair $\mathbf{c}$ if necessary;
end

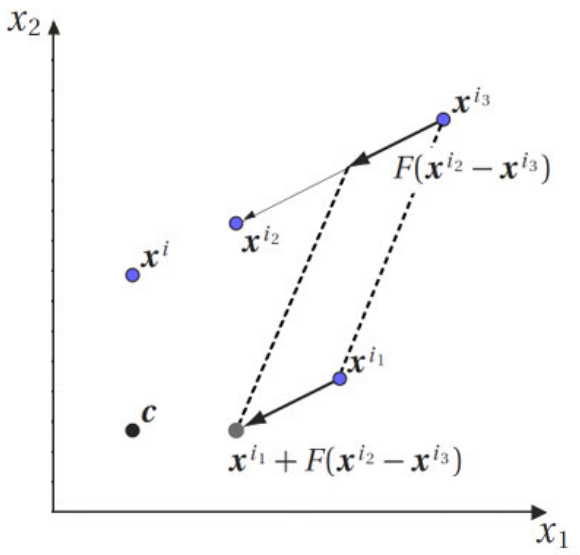


Figure 2.6: Visual representation of candidate creation with the DE/rand/1/bin strategy.

Sometimes, the newly created candidate falls out of bounds of the decision space. In such cases, a repair method can be used. Repairing infeasible candidates requires a repair procedure that modifies a given decision vector in such a way that it does not violate constraints. We solve this problem by replacing the candidate decision value violating the boundary constraints with the closest boundary value. In this way, the candidate gets inside the predefined search region and becomes feasible with only a small alteration. This ensures that, in order to find solutions not violating boundary constraints, the creation of new solutions is not necessary. But on the other hand, the repair procedure is biased for problems where the optimal solution lies on one of the bounds of the decision space.

A repair procedure is needed also in the case when the decision variables are not continuous. In the discrete decision space the operations of vector addition and scalar multiplication can result in a point that is not part of the discrete decision space. In such cases we adopt the Lamarckian repair (Ishibuchi, Kaige, & Narukawa, 2005), where the

candidate's values are rounded to the nearest point in the decision space.

The newly obtained good solutions are immediately included in the creation of new candidates, which speeds up the convergence to the optimal solutions. Moreover, since no solution can be removed from the population unless a better solution is found, DE implicitly incorporates elitism that assures the preservation of good solutions. Besides its good convergence properties and suitability for parallelization, perhaps the key advantage of DE is its conceptual simplicity and ease of use.

2.3.2 The DEMO Algorithm

The DEMO algorithm proposed by Robič and Filipič (2005) is multiobjective evolutionary algorithm based on DE. Like DE, DEMO is easy to understand and implement, and very effective on numerical problems. DEMO is a steady-state evolutionary algorithm that adds candidate solutions directly to the existing population. Since they are immediately used for generating new solutions, the algorithm's convergence is accelerated.

The idea of DEMO is to use DE for exploring the decision space and environmental selection procedure to select the best individuals for the next population. Various environmental selection procedures can be used in DEMO. In this dissertation the environmental selection procedure from the NSGA-II algorithm (Deb et al., 2002) presented in Subsection 2.2.1 is used.

The outline of DEMO is shown in Algorithm 2.4. Similarly to DE, the algorithm starts with a population $\mathcal{P}$ of $popSize$ randomly created solutions. In each generation the following steps are repeated. For every solution (called parent) in $\mathcal{P}$ the candidate is created using the DE/rand/1/bin strategy. The candidate is then evaluated and compared to its parent. The Pareto dominance relations are used to determine the dominance relation between the solutions. The candidate replaces the parent if it dominates the parent. If the parent dominates the candidate, the candidate is discarded. Otherwise (if the candidate and parent are incomparable), the candidate is added to the population. After repeating this step for all $popSize$ solutions in $\mathcal{P}$, the population has enlarged and needs to be truncated back to the size $popSize$ using the environmental selection procedure. In DEMO, the environmental selection procedure from the NSGA-II algorithm is used (see Subsection 2.2.1). In the end of every generation, the solutions from $\mathcal{P}$ are randomly enumerated so that the order of parents changes in the next generation. After meeting the stopping criterion, the output of DEMO consists of a front of non-dominated solutions from $\mathcal{P}$.

Because DEMO immediately discards the dominated solution in the comparison between the candidate and its parent, its population size rarely reaches $2 * popSize$ before environmental selection. Therefore, the computational complexity of the employed environmental selection procedure is often smaller than the complexity of the same environmental selection procedure applied in other MOEAs, for example in NSGA-II.

The DEMO algorithm is characterized by the same advantages and limitations as DE. When comparing it to other MOEAs, DEMO achieved significantly better results (Tušar, 2007; Tušar & Filipič, 2007). However, it is important to note that both DE and DEMO are limited to vector representation of solutions and can therefore only be used in numerical optimization.

2.4 Surrogate-Model-Based Optimization

Most engineering optimization problems require experiments and/or simulations to evaluate objective and constraint functions. For many real world problems, however, a single simulation can take minutes, hours, or even days to complete. As a result, an optimization

Algorithm 2.4: DEMO

Result: A set of non-dominated solutions

Create the initial population $\mathcal{P}$ of random individuals;
Evaluate the solutions in $\mathcal{P}$;
while *stopping criterion not met* **do**
  for each individual $\mathbf{p}^i$ ($i = 1, \ldots,$ *popSize*) from $\mathcal{P}$ **do**
    Create candidate $\mathbf{c}$ from parent $\mathbf{p}^i$;
    Calculate the objectives of $\mathbf{c}$;
    if $\mathbf{c}$ *dominates* $\mathbf{p}^i$ **then**
      Replace $\mathbf{p}^i$ with $\mathbf{c}$;
    end
    else if $\mathbf{p}^i$ *dominates* $\mathbf{c}$ **then**
      Discard $\mathbf{c}$;
    end
    else
      Add $\mathbf{c}$ to $\mathcal{P}$;
    end
  end
  if *there are more than popSize individuals in* $\mathcal{P}$ **then**
    Apply an environmental selection procedure to get the best *popSize* individuals (see Subsection 2.2.1);
  end
  Randomly enumerate the individuals in $\mathcal{P}$;
end

task that requires thousands or even millions of simulation evaluations becomes almost impossible. One way of reducing this burden is by constructing approximation models, known as surrogate models (sometimes called also meta-models), that mimic the behavior of the simulation model as closely as possible while being computationally cheap(er) to evaluate. A surrogate model is constructed based on modeling the response of the simulator from a limited number of previously evaluated data points. The idea is to build a surrogate model that is as accurate as possible, using as few simulation evaluations as possible. The surrogate model is then used during the optimization processes to approximate the solution values instead of exactly evaluating them.

In the literature various modeling techniques are utilized for building the surrogate models. For single- and multiobjective optimization similar techniques are used. Some of the most commonly used ones are the Response Surface Method (Myers & Montgomery, 1995), Radial Basis Function (Hardy, 1971), Neural Network (Specht, 1990), Krigging (Stein, 1999) and various types of Gaussian Process modeling (Rasmussen & Williams, 2006; Quinonero-Candela, Rasmussen, & Williams, 2007). These techniques typically return only one approximated value, which is why in multiobjective optimization these techniques need to be slightly modified or several surrogate models have to be used, so that each surrogate model approximates one objective (Jin, 2011).

The essential question to answer in surrogate-model-based optimization is which individuals should be chosen to be exactly evaluated and which to be approximated with a surrogate model. As we assume exact evaluations to be time-consuming, the question is how to adapt the number of exactly evaluated individuals so that the time needed for optimization process can be reduced as much as possible, while the algorithm still finds

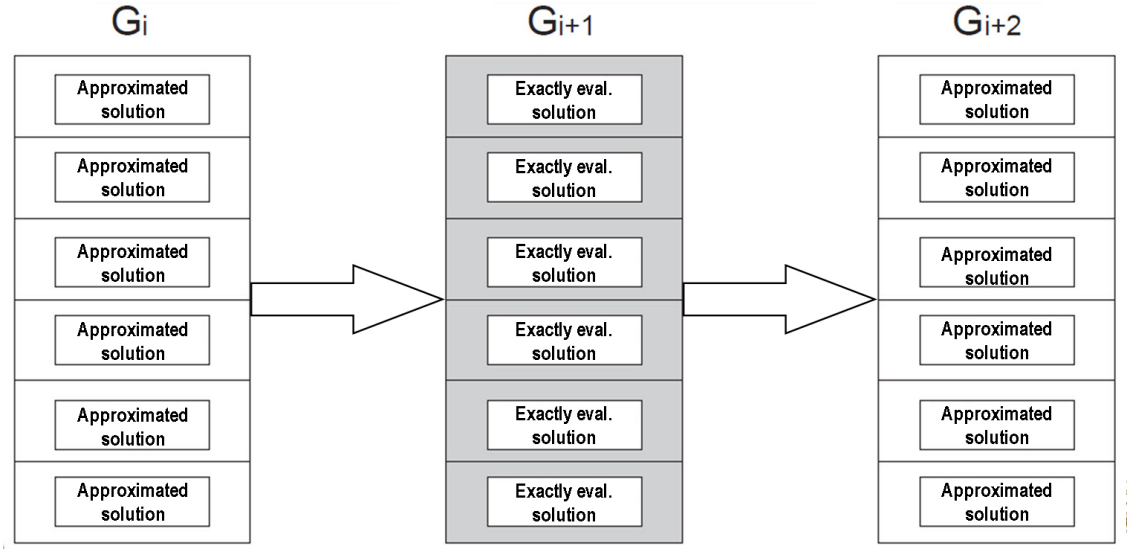


Figure 2.7: Generation-based evolution control.

the (near-)optimal solutions.

In the literature there are many approaches that include various algorithms and various modeling techniques. The main difference between the approaches is how and when the surrogate models are used in the optimization process. In evolutionary algorithms the mechanism that controls the use of surrogate models is called evolution control (Jin, 2003).

2.4.1 Evolution Control

Using evolution control we can set the balance between (i) solving the optimization problem as fast as possible by approximating as many solutions as possible, or (ii) by exactly evaluating a lot of solutions to ensure the high accuracy of the surrogate model to prevent inaccurate approximations and minimize the possibility of getting stuck in a local optimum.

The evolution control can be either fixed or adaptive. In *fixed evolution control*, the number of exact solution evaluations performed during optimization is known in advance and is invariant to the optimization problem. Fixed evolution control can be further divided into generation-based and individual-based evolution control.

In *generation-based evolution control* (Figure 2.7) during the optimization process in some generations all solutions are approximated and in the others exactly evaluated (Deb & Nain, 2007). The alternation between approximations and exact evaluations is usually done in a way that a few generations of exact evaluations are typically followed by a few generations of approximations. During the generations with the exact evaluations the newly exactly evaluated solutions are added to the set of all exactly evaluated solutions. This set of solutions is then used to train or update the surrogate model. The surrogate model is then used for solution approximations in the next generations.

With *individual-based evolution control* (Figure 2.8), solution evaluations (exact or approximations) are determined individually and not for the whole generation. In every generation some (usually best) solutions are exactly evaluated while the others are approximated (Grierson & Pak, 1993). Newly exactly evaluated solutions are then used to update the surrogate model. This approach is convenient for single-objective optimization problems, but not for multiobjective optimization with sets of non-dominated solutions. Exactly evaluating best solutions would mean that all non-dominated solutions would have

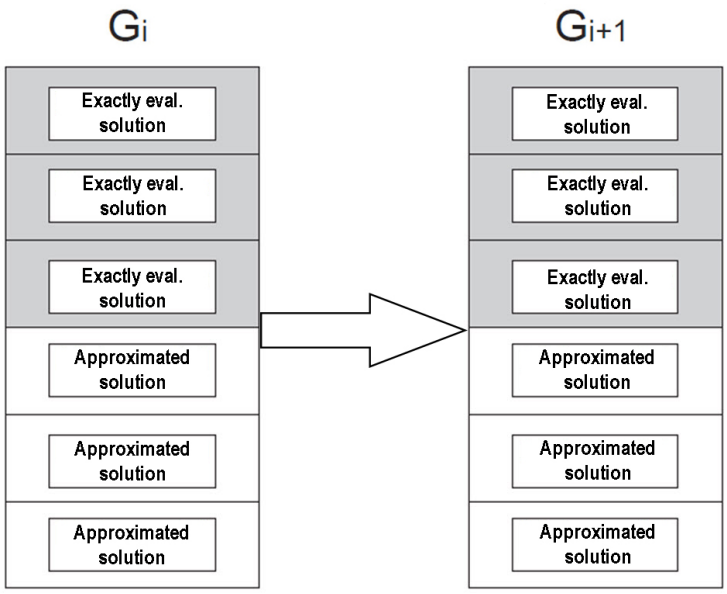


Figure 2.8: Individual-based evolution control.

to be exactly evaluated, which could result in exactly evaluating most solutions. Therefore, for multiobjective problems, we could exactly evaluate a random number of solutions, but this would result in a lot of unnecessary exact evaluations, and that is why individual-based evolution control is not very suitable for multiobjective optimization.

With both generation-based and individual-based evolution control the number of exact evaluations is known before the optimization process and is invariant to the problem. This means that even if an optimization problem is very simple and can be modeled very accurately, the number of exact evaluations remains the same. For such problems the number of exactly evaluated solutions could be reduced by mostly using solution approximations and the obtained results would not change due to accurate approximations.

The approach where the number of exactly evaluated solutions is not known in advance, but depends on the accuracy of the surrogate model for a given problem, is called *adaptive evolution control* (Figure 2.9). For simpler problems, a small number of solutions get exactly evaluated, and for more complex problems, where a small number of exactly evaluated solutions could lead to inaccurate approximations, the number of exactly evaluated solutions is higher.

In the literature adaptive evolution control can be found in a lot of approaches. For example, one can exactly evaluate only solutions that are approximated as better than the existing solutions (A. Zhou & Q. Zhang, 2010). Another approach would be to use the surrogate model to find the best solutions, exactly evaluate them, update the surrogate model with them and repeat the procedure until the found solutions are not changing any more (Z. Zhou, Ong, Nair, Keane, & Lum, 2007). Another possible use of adaptive evolution control would be to use the confidence of the approximations for determining which solutions are perspective and should get exactly evaluated (Ulmer, Streichert, & Zell, 2003; Jones, Schonlau, & Welch, 1998).

These are just a few examples to illustrate the vast possibilities when using adaptive evolution control. Since most of the modern algorithms use adaptive evolution control, we focus our overview on those algorithms.

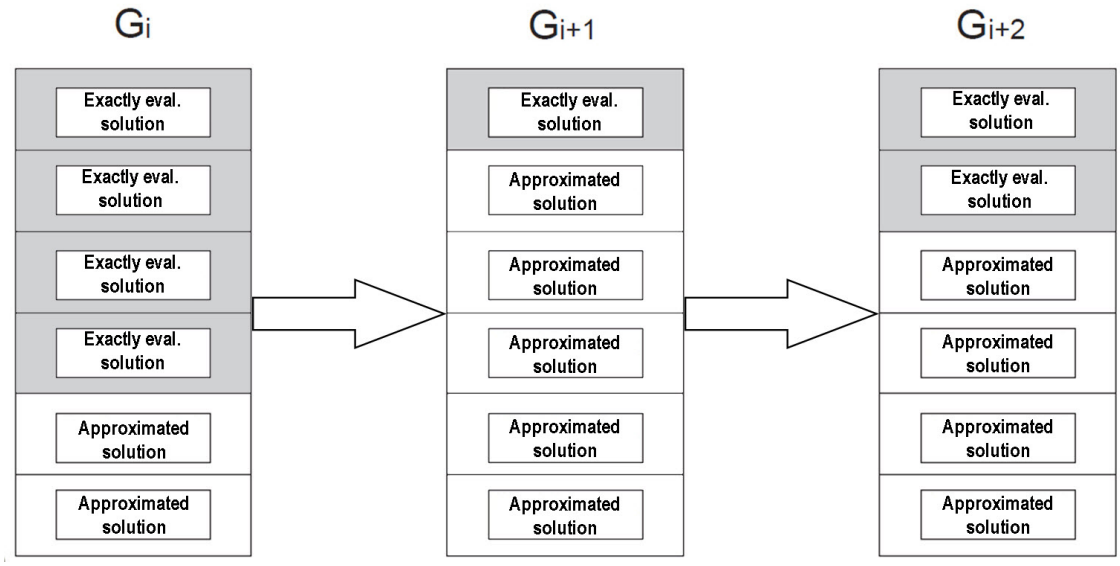


Figure 2.9: Adaptive evolution control.

2.4.2 Surrogate-Model-Based Multiobjective Optimization

In single-objective optimization, the usage of surrogate models is well established and has proven to be successful. In the literature many algorithms and various modeling techniques are used to solve benchmark and real-world problems (J. Zhang & Sanderson, 2007; Emmerich, Giotis, Özdemir, Bäck, & Giannakoglou, 2002). The results typically show that the surrogate-model-based optimization in comparison to optimization without surrogates provides comparable results in fewer exact solution evaluations (Jin, Olhofer, & Sendhoff, 2001; Z. Zhou et al., 2007).

Similarly, the use of DE in combination with surrogate models also produced promising results (J. Zhang & Sanderson, 2007). The authors presented an algorithm based on DE that generates multiple offspring for each parent and chooses the promising ones based on the confidence and the approximation value of the current surrogate model.

This dissertation specializes in multiobjective optimization where the integration and application of surrogate models is different than in the surrogate-model-based single-objective optimization. Since the algorithms and approaches used for multiobjective optimization problems are different, we focus our related work survey to the surrogate-model-based multiobjective evolutionary algorithms, where the problem of finding (near-)optimal solutions is even more challenging than in single-objective optimization.

Approaches in the literature differ in terms of which solutions are approximated and how they use the approximations. Surrogate models can aim at either a global approximation of the objective function, or a local one, focusing on the neighborhood of the best current individuals. Of course, like in (Z. Zhou et al., 2007), the combination of local and global surrogate models can be used for solving optimization problems.

In surrogate-model-based optimization there are two main ways of using surrogate models: as a part of memetic search or to pre-select the promising individuals which are then exactly evaluated (Pilat & Neruda, 2012).

In a memetic algorithm, an additional algorithm (e.g., a gradient-based or an evolutionary algorithm) is used to find (near-)optimal solutions using the surrogate model. Once this optimum is found, the best solutions are exactly evaluated and used for updating the model. In (Pilat & Neruda, 2011), aggregated surrogate models are used in a memetic

algorithm. The model is based on the distance to the currently known, non-dominated set and is used to find new, non-dominated individuals using local search. In memetic algorithms, especially if the surrogate model is not very accurate, a local optimum is often found instead of the global optimum.

In the case of pre-selecting the promising individuals, the surrogate model is used to find the promising individuals or drop the low-quality ones even before they are exactly evaluated, thus reducing the number of exact evaluations. For example, OEGADO (Chafekar, Shi, Rasheed, & Xuan, 2005) creates a surrogate model for each of the objectives. The best solutions in every objective get also approximated on other objectives, which helps with finding trade-off individuals. The best individuals are then exactly evaluated and used to update the models. ParEGO (Knowles, 2006) uses the weighted sum of the objective functions to perform local search. The weights are generated randomly for each iteration. When a different model is used for each of the functions, the conversion from the multiobjective problem to the single-objective one has to be performed (or a multiobjective optimizer has to be used on the models). Moreover, if there are several models, their errors can add up, as well as the time needed to train the models.

Surrogate models are also used to rank and filter out offspring according to Pareto-related indicators like hypervolume (Emmerich, Giannakoglou, & Naujoks, 2006), or a weighted sum of the objectives (Taboada, Baheranwala, Coit, & Wattanapongsakorn, 2007). The problem with the methods that use hypervolume as a way of finding promising solutions is the hypervolume calculation time, especially on many objectives. Another possibility is described in (Li et al., 2009), where the authors present an algorithm that calculates only non-dominated solutions or solutions that can, because of variance, become non-dominated.

Some surrogate models, in addition to the approximation value, also return the certainty of the prediction. The use of this confidence information can help increase the accuracy of the surrogate model. The early work on uncertainty measures for solution approximations using GP modeling is presented in (Zilinskas, 1980) and (Mockus, Tiesis, & Zilinskas, 1978). There are various approaches that use confidence information with surrogate-model-based optimization algorithms. In (Emmerich et al., 2002), the authors use confidence information to guide the search towards less explored regions in the search space. The confidence of the prediction with the approximated value can also be used to calculate the criterion of expected improvement. A comprehensive discussion of how to use the criterion of expected improvement including the confidence bounds in evolutionary multiobjective optimization is found in (Emmerich, 2005). Approaches to applying this criterion are analyzed in (Wagner, Emmerich, Deutz, & Ponweiser, 2010). An example of an algorithm that uses an approximated value and variance to calculate the criterion of expected improvement to decide which solutions should be exactly evaluated is presented in (Jones et al., 1998). In the same paper it was also shown that in single-objective optimization the approximated solutions with higher variance have higher expected improvement. In (Emmerich, Deutz, & Klinkenberg, 2011), it was later shown that this holds also for biobjective expected improvement.

When comparing approximated solutions with unknown exact values (solutions under uncertainty), the comparison techniques can be modified to consider this uncertainty. The certainty of the prediction obtained with some modeling technique can be included in the comparison as an additional knowledge to prevent incorrect comparisons or minimize their number. These techniques for comparing solutions under uncertainty are presented in the next section.

2.5 Comparing Solutions Under Uncertainty

Comparison of solutions is an essential step of the optimization process. Comparing solutions helps determine which solution is better and therefore appropriate to drive the optimization process further, and which one is worse and should be replaced with a better solution. The comparison of solutions in single-objective optimization is straightforward. Either both solutions have the same objective values, or one solution is better than the other, which means that deciding which solution is better is trivial.

In multiobjective optimization we wish to simultaneously optimize several conflicting objectives. Here, one solution can be better in some objectives and worse in others. Consequently, the comparison of solutions and therefore the whole optimization process becomes more challenging.

When solving real-world optimization problems, it is often not possible to determine the objective values without uncertainties. The nature of uncertainties depends on the problem. In (Jin & Branke, 2005), four types of uncertainty sources are mentioned. The first one is the *noisy fitness functions*, where the same input parameters return different objective values. The second one is the requirement for *solution robustness*, where the quality of the obtained solutions should be robust against environmental changes or deviations from the optimal point. The third type is the *approximated fitness*, where the fitness functions suffer from approximation errors. The fourth and final type is the *time-varying fitness functions*, where the optimum of the problem to be solved changes over time and, thus, the optimizer should be able to track the optimum continuously.

Regardless of the uncertainty origin, the techniques for comparing solutions under uncertainty and determining their domination status are similar. Two different approaches are used when comparing solutions under uncertainty. The first one is to take the approximated value and variance and transform them into one value and then compare these single values. The second approach is to calculate the confidence interval and then directly compare the solutions represented with confidence intervals.

An example of the *first approach* can be found in (Fieldsend & Everson, 2005), where probabilistic dominance is defined and, for comparing solutions, the probability of dominance is used rather than outright dominance. If the probability that one solution dominates the other is higher than the specified degree of confidence, then this solution is said to dominate the other. This probabilistic dominance allows the use of the usual deterministic elitist algorithms with certain degree of confidence in the results. The methods to calculate the probability of dominance vary, depending on the types of uncertainty.

Similarly, in (Gong, Qin, & Sun, 2010) the authors define the dominance relation between solutions based on the probabilities of one solution objective being better than the same objective of another solution. For solutions with multiple objectives, the hypercuboids are defined and, similarly, comparing their volume and the center point can determine the probability of one solution being better than the other. To select diverse solutions, the paper also redefines the crowding distance defined in (Deb et al., 2002) based on the location and the volume of the hyper-cuboids of these solutions.

Another example of this approach is presented in (Basseur & Zitzler, 2006), where each solution is inherently associated with a probability distribution over the objective space. A probabilistic model that combines quality indicators and uncertainty is created and then used to calculate the expected value for each solution.

In the *second approach*, the solutions represented with approximated values and confidence intervals are compared to determine the relation between the solutions.

In (Trautmann, Mehnen, & Naujoks, 2009) and (Voß, Trautmann, & Igel, 2010), the authors tackle a noisy optimization problem with an algorithm that evaluates every solution several times (and, if necessary, performs additional evaluations to reduce the uncertainty)

and calculates the mean value and standard deviation for these evaluations. Modified Pareto dominance relations are defined for comparing solutions in uncertain environments. The Pareto dominance relations are modified in a way that solution $\mathbf{x}$ dominates solution $\mathbf{y}$ if, for every objective, the mean value plus standard deviation of $\mathbf{x}$ dominates the mean value minus standard deviation of $\mathbf{y}$. If this is not the case, the solutions are non-dominated. To avoid having too many non-dominated solutions, the promising solutions are additionally evaluated to make the standard deviation smaller.

In (Soares, Guimaraes, Maia, Vasconcelos, & Jaulin, 2009), a robust multiobjective evolutionary algorithm was developed for solving optimization problems in which solutions should be invariant to small input changes. The uncertain parameters are represented with intervals, which results in solution objectives also being represented with intervals. The algorithm for comparing solutions then compares the worst-case scenario values of objectives, i.e., the values at the border of the interval.

In (Babbar, Lakshmikantha, & Goldberg, 2003), the authors tackled noisy optimization problems with a modified NSGA-II algorithm (Deb et al., 2002) for handling solutions with uncertainty. The procedure for obtaining the rank of solutions is transformed so that it also considers the variance of solutions. Dominated solutions can also be ranked on the Pareto frontier, if the distance to any non-dominated solution, calculated from the fitness values and variances of solutions, is smaller than the threshold called Neighborhood Restriction Factor. During the optimization process, this factor becomes smaller and the number of evaluations taken for non-dominated solutions increases, resulting in a smaller variance and a more precise set of non-dominated solutions.

A concept of comparing solutions with uncertain objectives represented with intervals is presented in (Teich, 2001). The authors define the extension of Pareto dominance based on a theory of probabilistic dominance. They present a case where objective values are continuously and uniformly distributed inside the interval and by comparing the distributions the probability of dominance is calculated. The approach is then implemented in the modified SPEA (Zitzler & Thiele, 1999) algorithm.

Another concept of comparing solutions under uncertainty is presented in (Oumayma, Nahla, & Talbi, 2013), where new Pareto relations are defined using a possibilistic framework. The solutions characterized by a particular *possibility distribution* are represented with triangular possibility distributions, i.e., a triplet of values: most plausible value and lower and upper borders of distribution that represent the least plausible values. Based on this representation, the authors define strong Pareto dominance, weak Pareto dominance, and indifference (where it cannot be determined which solution looks better). They used this approach on a vehicle routing problem with uncertain demands.

A more theoretical approach to the solution comparison under uncertainty is presented in (Limbourg, 2005) for optimization problems where the uncertainty of solutions cannot be reduced by the sampling methods. The solutions are represented with intervals, and new relations are defined for comparing those intervals. The authors define certain and uncertain domination criterion for comparing intervals. On this basis they suggest a strong Pareto dominance relation in cases where the dominance relation can be determined, and weak Pareto dominance relation when the domination relation cannot be determined because of uncertainty. In this case, the expected values for every solution are assumed and these values are then compared.

In (Gunter, 2001), a partial order approach is suggested to enable the comparison of solutions represented with confidence intervals. This approach does not differentiate between the cases in which the upper border of one interval dominates the lower border of another interval and the cases in which some part of intervals overlap. A very similar approach to handle solutions represented with intervals, called imprecise Pareto relations,

is presented in (Limbourg & Aponte, 2005).

Bounding boxes representing multiobjective solutions with confidence intervals are defined in (Emmerich & Naujoks, 2004) (they are described in greater detail in Section 3.2). The authors present various comparison strategies, but in all strategies the comparison of bounding boxes is simplified to the comparison of bounding box bounds. The individuals are compared to all solutions in the population and individuals with a small probability of being competitive are rejected, while individuals with a high probability of being better are exactly evaluated.

To our knowledge, none of these methods systematically covers all aspects of comparing (constrained) multiobjective solutions with confidence intervals, which is one of the contributions of this dissertation.

In surrogate-model-based optimization the GP and RF modeling are two well-known modeling techniques that, in addition to the approximated value, return also the confidence interval for the approximation which is then used in the comparison under uncertainty. Both methods are described in detail in the following sections.

2.6 Gaussian Process Models

The GP models are probabilistic non-parametric models based on the principles of Bayesian probability, which can be used for both regression and classification problems. The name GP models refers to the assumption that a prior distribution on the function to be modeled is a stochastic process with a normal distribution, i.e., a Gaussian Process.

GPs were first used in 1940's for time series prediction (N. Wiener, 1949; Kolmogorov, 1941). In 1970's, GPs were widely used in the field of geostatistics where the method is known as Kriging (Matheron, 1973), named after a mining engineer Krige (Krige, 1951). The Kriging method is similar to GP regression from mathematical point of view, but as a geostatistical method it is mostly used for two- and three-dimensional input spaces and tends to ignore any probabilistic interpretations (Cressie, 1993; Stein, 1999). It should be noted that Kriging uses a prediction error minimization for modeling, which in principle does not consider uncertainty. To take uncertainty into account, additional computations are needed (Kleijnen & Beers, 2004; Sacks, Welch, Mitchell, & Wynn, 1989). On the other hand, GPs use a Bayesian inference which interprets the uncertainty in the probabilistic way. A Bayesian description of GPs to define prior distributions over functions was presented and applied to one-dimensional curve fitting in (O'Hagan, 1978). In the machine learning community, GPs were introduced for supervised learning by Williams and Rasmussen (1996). They were inspired by Neal (1996), where it was shown that under certain conditions Bayesian Neural Networks converge to GPs in the limit of an infinite number of units. Since the introduction of GPs in supervised learning (Williams & Rasmussen, 1996), GP models have been used for modeling in various fields, e.g., biological systems (Ažman & Kocijan, 2007; Južnič-Zonta, Kocijan, Flotats, & Vrečko, 2012), environmental systems (Grašič, Mlakar, & Božnar, 2006), chemical engineering (Likar & Kocijan, 2007) and many others.

GP models differ from most other black-box identification approaches in that they do not try to approximate the modeled system by fitting the parameters of the selected basis functions, but rather by searching for relationships among the measured data. The output of GP models is a normal distribution expressed in terms of the mean and the variance. The mean value represents the most likely output, and the variance can be interpreted as a measure of its confidence. The obtained variance, which depends on the amount and the quality of the available training data, provides important information when it comes to distinguishing GP models from other computational intelligence methods.

As GP models are, due to their probabilistic nature, suitable for interpolation, i.e., when data is missing, and in addition to the mean value also provide variance, they have already been used in stochastic optimization with surrogate models (Viana, Haftka, & Watson, 2012).

2.6.1 Gaussian Process Modeling

A GP is a collection of random variables that have a joint multivariate Gaussian distribution. Assuming a relationship of the form $y = f(\mathbf{x})$ between input $\mathbf{x}$ and output y, we have $y_1, \ldots, y_N \sim \mathcal{N}(0, \mathbf{K})$, where $K_{pq} = \mathrm{Cov}(y_p, y_q) = C(\mathbf{x}_p, \mathbf{x}_q)$ gives the covariance between the output points corresponding to the input points $\mathbf{x}_p$ and $\mathbf{x}_q$. Thus, the mean $\mu(\mathbf{x})$ and the covariance function $C(\mathbf{x}_p, \mathbf{x}_q)$ fully specify the GP.

The value of the covariance function $C(\mathbf{x}_p, \mathbf{x}_q)$ expresses the correlation between the individual outputs $f(\mathbf{x}_p)$ and $f(\mathbf{x}_q)$ with respect to inputs $\mathbf{x}_p$ and $\mathbf{x}_q$. It should be noted that the covariance function $C(\cdot, \cdot)$ can be any function that generates a positive semi-definite covariance matrix.

A commonly used covariance function is a composition of the square exponential covariance function with "automatic relevance determination" (ARD) hyperparameters (MacKay, 1998) and the constant covariance function assuming white noise:

$$C(\mathbf{x}_p, \mathbf{x}_q) = v_1 \exp\left[-\frac{1}{2}\sum_{d=1}^{D} w_d (x_{dp} - x_{dq})^2\right] + \delta_{pq} v_0, \tag{2.11}$$

where w_d, v_1 and v_0 are the hyperparameters of the covariance function, D is the input dimension, and $\delta_{pq} = 1$ if $p = q$ and 0 otherwise. It should be noted that this covariance function is infinitely differentiable and therefore very smooth. In (Stein, 1999), it is argued that such strong smoothness assumptions are unrealistic for modeling many physical processes, but probably this covariance function is still the most widely used pragmatic choice. However, other forms and combinations of covariance functions suitable for various applications can be found in (Rasmussen & Williams, 2006). The hyperparameters can be written as a vector $\boldsymbol{\Theta} = [w_1, \ldots, w_D, v_1, v_0]^T$. The hyperparameters w_d indicate the importance of individual inputs. If w_d is zero or near zero, it means the inputs in dimension d contain little information and could possibly be neglected.

To accurately reflect the correlations present in the training data, the hyperparameter values of the covariance function need to be optimized. Due to the probabilistic nature of the GP models, instead of minimizing the model error, the probability of the model is maximized.

Consider a set of N D-dimensional input vectors $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]^T$ and a vector of output data $\mathbf{y} = [y_1, y_2, \ldots, y_N]$. Based on the data $(\mathbf{X}, \mathbf{y})$, and given a new input vector $\mathbf{x}^*$, we wish to find the predictive distribution of the corresponding output y^*. From the training set $\mathbf{X}$, a covariance matrix $\mathbf{K}$ of size $N \times N$ is determined. The overall problem of learning unknown parameters from data corresponds to the predictive distribution $p(y^*|\mathbf{y}, \mathbf{X}, \mathbf{x}^*)$ of the new target y, given the training data $(\mathbf{y}, \mathbf{X})$ and a new input $\mathbf{x}^*$. In order to calculate this posterior distribution, a prior distribution over the hyperparameters $p(\boldsymbol{\Theta}|\mathbf{y}, \mathbf{X})$ needs to be defined, followed by the integration of the model over the hyperparameters

$$p(y^*|\mathbf{y}, \mathbf{X}, \mathbf{x}^*) = \int p(y^*|\boldsymbol{\Theta}, \mathbf{y}, \mathbf{X}, \mathbf{x}^*) p(\boldsymbol{\Theta}|\mathbf{y}, \mathbf{X}) d\boldsymbol{\Theta}. \tag{2.12}$$

The computation of such integrals can be difficult due to the intractable nature of the non-linear functions, therefore, the general practice for estimating hyperparameter values

is the maximum-likelihood estimation, i.e., minimizing the following negative log-likelihood function:

$$\mathcal{L}(\mathbf{\Theta}) = -\frac{1}{2}\log(|\mathbf{K}|) - \frac{1}{2}\mathbf{y}^T\mathbf{K}^{-1}\mathbf{y} - \frac{N}{2}\log(2\pi). \tag{2.13}$$

GP models can be easily utilized for regression calculation. Based on the training set $\mathbf{X}$, a covariance matrix $\mathbf{K}$ of size $N \times N$ is calculated. The aim is to find the distribution of the corresponding output y^* for some new input vector $\mathbf{x}^* = [x_1(N+1), x_2(N+1), \ldots, x_D(N+1)]$.

The predictive distribution of the output for a new test input has a normal probability distribution with a mean and variance

$$\mu(y^*) = \mathbf{k}(\mathbf{x}^*)^T\mathbf{K}^{-1}\mathbf{y}, \tag{2.14}$$

$$\sigma^2(y^*) = \kappa(\mathbf{x}^*) - \mathbf{k}(\mathbf{x}^*)^T\mathbf{K}^{-1}\mathbf{k}(\mathbf{x}^*), \tag{2.15}$$

where $\mathbf{k}(\mathbf{x}^*) = [C(\mathbf{x}_1, \mathbf{x}^*), \ldots, C(\mathbf{x}_N, \mathbf{x}^*)]^T$ is the $N \times 1$ vector of covariances between the test and the training cases, and $\kappa(x^*) = C(\mathbf{x}^*, \mathbf{x}^*)$ is the covariance between the test input itself.

As can be seen from (2.15), the GP model, in addition to a mean value, also provides information about the confidence of prediction using the variance. Usually, the confidence of the prediction is depicted with a 2σ interval, which corresponds to about 95% of the confidence interval. Considering the confidence intervals of all predictions, we obtain a confidence band, shown in gray in the example in Figure 2.10. By indicating a wider confidence band around the predicted mean, it highlights the areas of the input space where the prediction quality is poor due to the lack of data or noisy data.

2.6.2 Sparse Approximation

A noticeable drawback of "full" GP modeling is the computation load that increases with the third power of the amount of input data due to the calculation of the inverse of the covariance matrix. This computational complexity restricts the amount of training data to, at most, a few thousand instances. As multiobjective evolutionary algorithms usually require more than a few thousand evaluations, which serve as the training data for GP models, "full" GP modeling does not seem to be viable for our needs.

To overcome the computational-limitation issues and consequently make the method viable for large-scale dataset applications, such as stochastic optimization, numerous authors have suggested various sparse approximations. A survey of such methods can be found in (Quinonero-Candela et al., 2007; Quinonero-Candela & Rasmussen, 2005). A common property of the sparse-approximation methods is that they try to retain the bulk of the information contained in the full training dataset, but reduce the size of the covariance matrix so as to facilitate a less computationally demanding implementation of the GP model. Usually, this subset of the training data is called the active set. The computational complexity of such algorithms is $O(NM^2)$, where N is the amount of training data and M is the size of the active set.

We decided to use a state-of-the-art sparse-approximation method named Sparse Gaussian Processes using Pseudo-inputs (SPGP) (Snelson & Ghahramani, 2006), which is in general determined as a fully independent training conditional approximation (Quinonero-Candela & Rasmussen, 2005; Quinonero-Candela et al., 2007). The idea of this method is that instead of selecting a subset of the training data, it rather optimizes the locations of M pseudo-inputs, as this seems to be easier to solve than the discrete subset selection problem. The pseudo-input locations are optimized based on the covariances between the training data points and the pseudo-inputs.

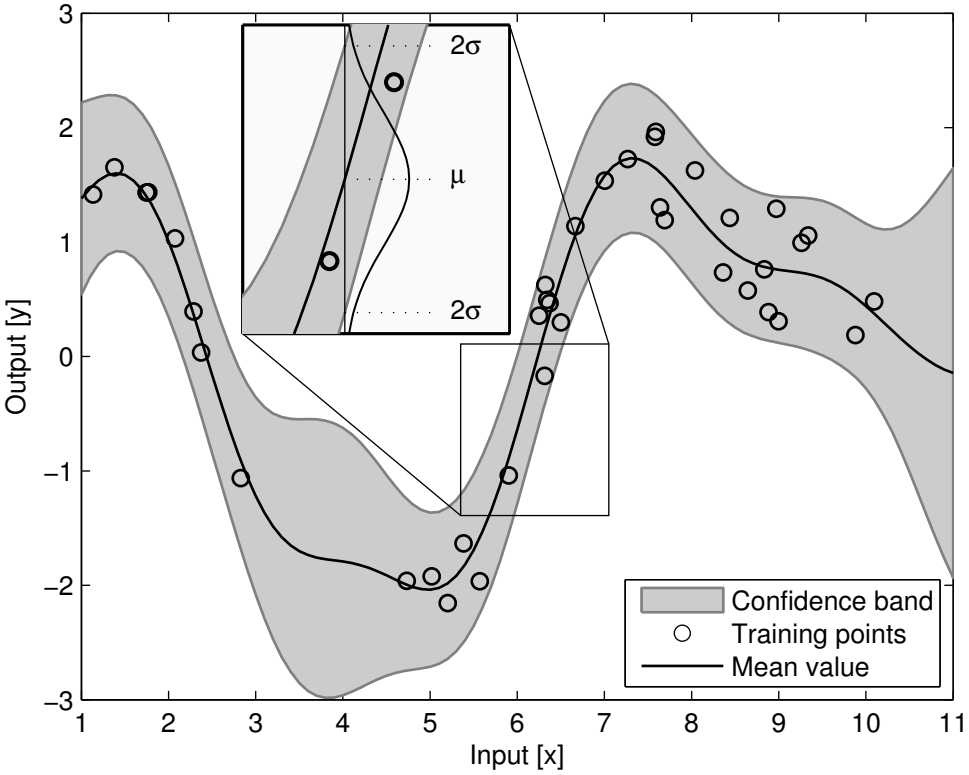


Figure 2.10: Modeling with GP models: in addition to the mean value (prediction), we obtain a 95% confidence band for the underlying function f (shown in gray).

It should be noted that due to fewer data points being incorporated into the model (covariance matrix) and their arbitrary locations, the posterior of the SPGP model, especially the variance, can be somewhat different to the posterior of a "full" GP model. Such a case is illustrated in Figure 2.11. It is clear that the mean value is very similar to the mean value obtained with the "full" GP model (Figure 2.10), but the variance (95% confidence interval) is distinctly different from the variance obtained by the "full" GP model (Figure 2.10).

2.7 Random Forest Models

RF is an algorithm for classification and regression developed by Breiman (2001) that uses an ensemble of decision trees to make predictions. If the RF is used for classification, the classification trees are used, otherwise the regression trees are used. The difference between decision and regression trees is in the values contained in the leaves of the tree. In classification trees, the leaves contain class labels and in regression trees a (real) number is set instead. Since solving MOPs requires finding objective values that are continuous, the regression trees have to be used, thus RF for regression is described and used in this dissertation.

The RF algorithm is a slightly modified version of the bagging (bootstrap aggregation) method (Breiman, 1996). In bagging an ensemble of trees is constructed, where each tree is built using a different bootstrap sample of the data. Bootstrapping means creating new training sets of size N' by randomly sampling the given set (of size N) $N' \leq N$ times with

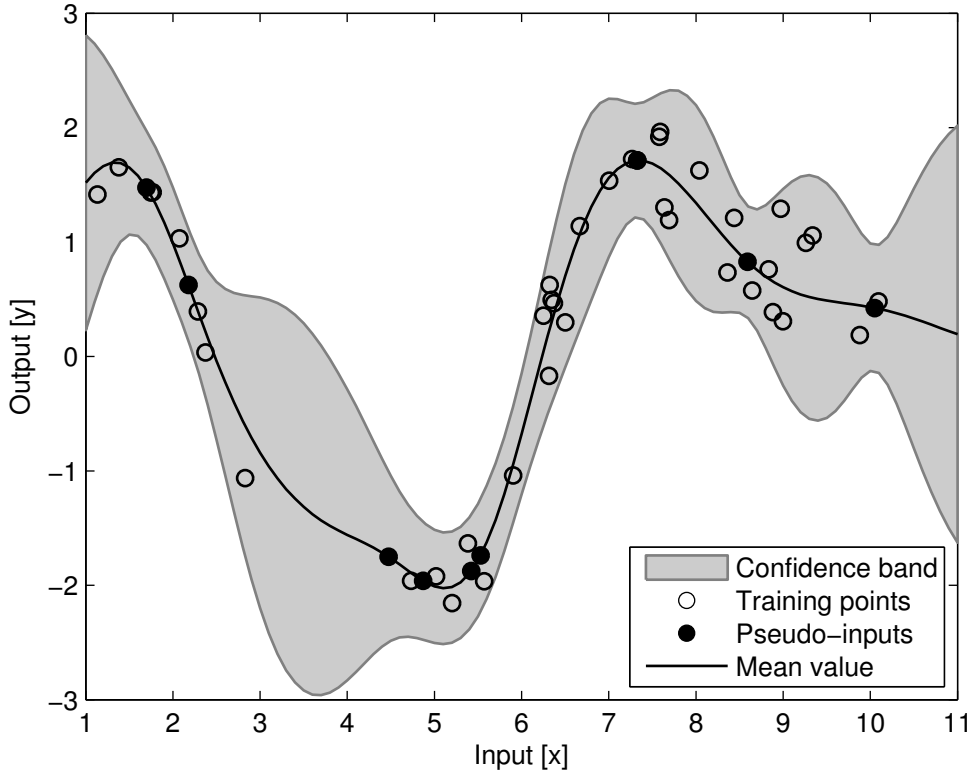


Figure 2.11: Modeling with SPGP models: pseudo-inputs (dots) are arbitrarily located, i.e., not a subset of the training data points (circles).

replacement. This ensures that each tree is trained on a slightly different version of the data, which helps ensuring the trees are uncorrelated. For regression, the final prediction for bagging is obtained by calculating the mean of all predictions from all trees built.

The difference between RF and bagging is that RF adds additional layer of randomness by changing how the regression trees are constructed. Algorithm 2.5 presents the overview of the RF modeling technique.

In the RF algorithm, as in bagging, each regression tree in the forest is trained using a bootstrapped sample of the training data. Each tree is built recursively, starting from the root and unpruned (grown fully), so as to obtain low-bias trees. At each node a randomized search procedure is applied to determine how the node should be split. The splits are usually chosen according to the squared error. If by splitting the node the squared error decreases, the node is split, otherwise the node is not split. When the best split is found, the decision space is partitioned according to the chosen split, and the procedure is applied recursively to split the left and right children. Splitting continues until no acceptable split can be found or the children contain fewer than the predefined number of minimal data points in leaves.

In contrast to standard trees, where each node is split choosing among all variables, in RF each node is split using a subset of variables randomly chosen at that node. This somewhat counter-intuitive strategy turns out to perform very well compared to many other modeling techniques, including discriminant analysis, support vector machines and neural networks, and is robust against overfitting (Breiman, 2001). In the same paper the authors argue that the RF algorithm achieves:

Algorithm 2.5: RF

Result: A random forest model

Let N_{trees} be the number of trees to build;
for each of N_{trees} iterations **do**
    Select a new bootstrap sample from training set;
    Grow a regression tree on this bootstrap;
    At each internal node randomly select a subset of variables and determine the best split using only these variables;
    Do not perform cost complexity pruning. Save tree as is, alongside those built thus far;
end

- exceptional prediction accuracy,
- low bias,
- low variance.

In addition, the RF algorithm (i) can be used when there are many more variables than observations, (ii) is very user-friendly in the sense that it has only few parameters to choose (the number of variables in the random subset at each node, the number of trees in the forest and the minimal number of solutions in leaves), and (iii) is usually not very sensitive to their values.

Chapter 3

Relations for Comparing Solutions Under Uncertainty

In optimization algorithms, finding relations between solutions is necessary to determine which solutions are better and which are worse. For multiobjective optimization problems without uncertainty the Pareto dominance relations are used. For problems with uncertainty modified relations have to be used to additionally include the uncertainty in the comparison of solutions (Mlakar, Tušar, & Filipič, 2013b, 2014b).

In this chapter we propose new relations under uncertainty for comparing solutions represented with approximated values and confidence intervals. The use of relations under uncertainty is demonstrated for unconstrained and constrained problems. In addition, we compare the new relations under uncertainty with the Pareto dominance relations, and measure the number of incorrect comparisons performed with each kind of relations. We test the hypothesis that, in comparison to Pareto dominance relations, the use of relations under uncertainty reduces the number of incorrect comparisons. Since the number of incorrect comparisons depends also on the accuracy of the surrogate model, we use two different modeling techniques, i.e., GP and RF modeling, and compare the results gained with each one.

The relations without uncertainty and the relations under uncertainty are defined on the MOP described in Section 2.1. For an arbitrary solution $\mathbf{x}$ with n variables and m objectives, $\mathbf{x} = (x_1, ..., x_n) \in X$ is a *decision vector*, while the corresponding element $\mathbf{z} = \mathbf{z}(\mathbf{x}) = (z_1, ..., z_m) \in Z$ is an *objective vector*.

Since the relations under uncertainty are based on the Pareto dominance relations, the latter concept is described first.

3.1 Relations without Uncertainty

In this section we consider the case where all solutions of a MOP are exactly evaluated, that is they are without uncertainty.

Definition 3.1 (Pareto dominance). The objective vector $\mathbf{z}$ *dominates* the objective vector $\mathbf{w}$, $\mathbf{z} \prec \mathbf{w}$, iff $z_j \leq w_j$ for all $j \in \{1, ..., m\}$ and $z_k < w_k$ for at least one $k \in \{1, ..., m\}$.

Definition 3.2 (Weak Pareto dominance). The objective vector $\mathbf{z}$ *weakly dominates* the objective vector $\mathbf{w}$, $\mathbf{z} \preceq \mathbf{w}$, iff $z_j \leq w_j$ for all $j \in \{1, ..., m\}$.

Definition 3.3 (Strict Pareto dominance). The objective vector $\mathbf{z}$ *strictly dominates* the objective vector $\mathbf{w}$, $\mathbf{z} \prec\prec \mathbf{w}$, iff $z_j < w_j$ for all $j \in \{1, ..., m\}$.

When $\mathbf{z} = \mathbf{f}(\mathbf{x}), \mathbf{w} = \mathbf{f}(\mathbf{y})$ and $\mathbf{z}$ (weakly or strictly) dominates $\mathbf{w}$, we say that solution $\mathbf{x}$ (weakly or strictly) dominates solution $\mathbf{y}$. In other words, solution $\mathbf{x}$ is equal to or better than solution $\mathbf{y}$. The weak Pareto dominance is a natural generalization of the $\leq$ relation, and the strict Pareto dominance is the natural generalization of the $<$ relation.

Definition 3.4 (Incomparability). The objective vectors $\mathbf{z}$ and $\mathbf{w}$ are *incomparable*, $\mathbf{z} \parallel \mathbf{w}$, iff $\mathbf{z} \npreceq \mathbf{w}$ and $\mathbf{w} \npreceq \mathbf{z}$.

Again, if $\mathbf{z}$ and $\mathbf{w}$ are incomparable, solutions $\mathbf{x}$ and $\mathbf{y}$ are incomparable.

The above-defined relations are usually used only when solving problems without constraints where all solutions are feasible. For cases where the solutions can also be infeasible, the Pareto dominance relation is slightly modified (Deb et al., 2002) as follows.

Definition 3.5 (Constrained dominance). The objective vector $\mathbf{z}$ *constrained-dominates* the objective vector $\mathbf{w}$, $\mathbf{z} \prec_c \mathbf{w}$, if any of the following conditions is true:

1. Solution $\mathbf{z}$ is feasible and solution $\mathbf{w}$ is not.
2. Solutions $\mathbf{z}$ and $\mathbf{w}$ are both infeasible, but solution $\mathbf{z}$ has a smaller overall constraint violation.
3. Solutions $\mathbf{z}$ and $\mathbf{w}$ are feasible and solution $\mathbf{z}$ Pareto dominates solution $\mathbf{w}$.

When $\mathbf{z} = \mathbf{f}(\mathbf{x}), \mathbf{w} = \mathbf{f}(\mathbf{y})$ and $\mathbf{z}$ constrained-dominates $\mathbf{w}$, we say that solution $\mathbf{x}$ constrained-dominates solution $\mathbf{y}$.

The effect of using the constrained dominance principle is that any feasible solution is better than any infeasible solution and that of the two infeasible solutions the one closer to the feasibility region is better.

3.2 Relations under Uncertainty

In this section we consider the case where the objective values of the solutions are represented with the approximated values and confidence intervals for each approximation. In such a case, the standard relations described previously are not suitable and must be adapted to accommodate the uncertainty. Every solution $\mathbf{x}$ is represented with a vector of approximated objective values $\mathbf{z} = \mathbf{f}(\mathbf{x}) = (z_1, z_2, ..., z_m)$ and a confidence vector $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, ..., \varepsilon_m)$. For the objective z_i the confidence interval is equal to $[z_i - \varepsilon_i, z_i + \varepsilon_i]$. In order to be able to compare the solutions represented in this way, the relations between the solutions under uncertainty are defined on the *bounding boxes* (BBs) of their objective values. From the vectors of the approximated values and the confidence intervals, the bounding box of an objective vector $\mathbf{z}$ is designed as (Figure 3.1):

$$\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) = [z_1 - \varepsilon_1, z_1 + \varepsilon_1] \times [z_2 - \varepsilon_2, z_2 - \varepsilon_2] \times ... \times [z_m - \varepsilon_m, z_m - \varepsilon_m]. \tag{3.1}$$

This definition of BB presumes that the confidence intervals are symmetric. This is not always the case, e.g., because of non-symmetric form of noise. Instead of considering just confidence vector $\boldsymbol{\varepsilon}$, we could define lower-bound confidence vector $\underline{\boldsymbol{\varepsilon}} = (\underline{\varepsilon_1}, \underline{\varepsilon_2}, ..., \underline{\varepsilon_m})$ and upper-bound confidence vector $\overline{\boldsymbol{\varepsilon}} = (\overline{\varepsilon_1}, \overline{\varepsilon_2}, ..., \overline{\varepsilon_m})$. For the objective z_i the confidence interval would then be equal to $[z_i - \underline{\varepsilon_i}, z_i + \overline{\varepsilon_i}]$, and the definition of the bounding box that considers the non-symmetric uncertainty intervals would be:

$$\mathrm{BB}(\mathbf{z}, \underline{\boldsymbol{\varepsilon}}, \overline{\boldsymbol{\varepsilon}}) = [z_1 - \underline{\varepsilon_1}, z_1 + \overline{\varepsilon_1}] \times [z_2 - \underline{\varepsilon_2}, z_2 + \overline{\varepsilon_2}] \times ... \times [z_m - \underline{\varepsilon_m}, z_m + \overline{\varepsilon_m}]. \tag{3.2}$$

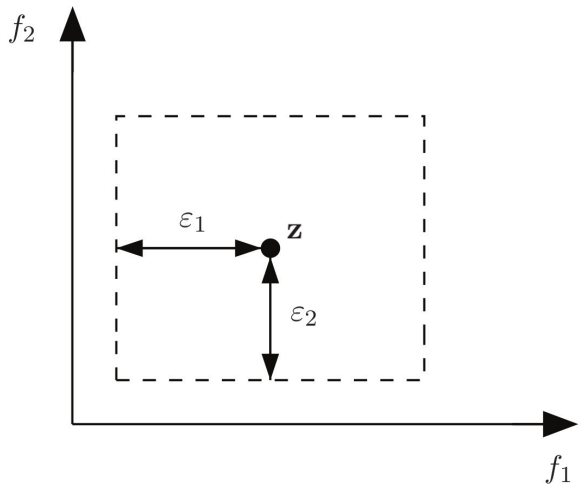


Figure 3.1: The bounding box of an objective vector.

However, since the relations under uncertainty are indifferent to the shape and size of the bounding boxes, we can for the sake of simplicity assume that the confidence interval is symmetric.

In addition to bounding boxes, where every objective has its own confidence interval, multiobjective solutions with uncertainty can also be represented with ellipsoids. Representation with ellipsoids restricts all objectives from obtaining their worst-case values simultaneously. But since comparing multiobjective solutions is performed by comparing pairs of objectives, where the confidence of each objective is inspected, we adopt the approach with bounding boxes.

We handle relations under uncertainty without constraints and with constraints separately.

3.2.1 Relations under Uncertainty without Constraints

Definition 3.6 (Probable Pareto dominance). The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ *probably dominates* the bounding box $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \prec_u \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, iff for every $\mathbf{z}' \in \mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ and every $\mathbf{w}' \in \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$: $\mathbf{z}' \prec w'$.

If $\mathbf{z} = \mathbf{f}(\mathbf{x})$ with confidence vector $\boldsymbol{\varepsilon}$, $\mathbf{w} = \mathbf{f}(\mathbf{y})$ with confidence vector $\boldsymbol{\delta}$, and $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \prec_u \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, then solution $\mathbf{x}$ probably dominates solution $\mathbf{y}$ ($\mathbf{x} \prec_u \mathbf{y}$). In other words, $\mathbf{x}$ dominates $\mathbf{y}$ with (high) confidence (depending on $\boldsymbol{\varepsilon}$ and $\boldsymbol{\delta}$).

To test whether $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ probably dominates $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, it is enough to check if the corner point $(z_1 + \varepsilon_1, z_2 + \varepsilon_2, \ldots, z_m + \varepsilon_m)$ dominates the corner point $(w_1 - \delta_1, w_2 - \delta_2, \ldots, w_m - \delta_m)$. If it does, then $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \prec_u \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$.

Figure 3.2 shows the objective values $\mathbf{z}^1, \ldots, \mathbf{z}^5$ and their bounding boxes. We can see that $\mathbf{z}^1$ probably dominates solution $\mathbf{z}^4$ ($\mathbf{z}^1 \prec_u \mathbf{z}^4$).

Analogously, other relations under uncertainty can be defined.

Definition 3.7 (Probable Pareto non-dominance). The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ *is probably non-dominated* by the bounding box $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \not\prec_u \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, iff for every $\mathbf{z}' \in \mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ and $\mathbf{w}' \in \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$: $\mathbf{z}' \prec \mathbf{w}'$ or $\mathbf{z}' \parallel \mathbf{w}'$.

Several examples of probable Pareto non-dominance can be seen in Figure 3.2: $\mathbf{z}^1 \not\prec_u \mathbf{z}^2, \mathbf{z}^1 \not\prec_u \mathbf{z}^3, \mathbf{z}^1 \not\prec_u \mathbf{z}^4, \mathbf{z}^1 \not\prec_u \mathbf{z}^5, \mathbf{z}^2 \not\prec_u \mathbf{z}^4, \mathbf{z}^3 \not\prec_u \mathbf{z}^4$.

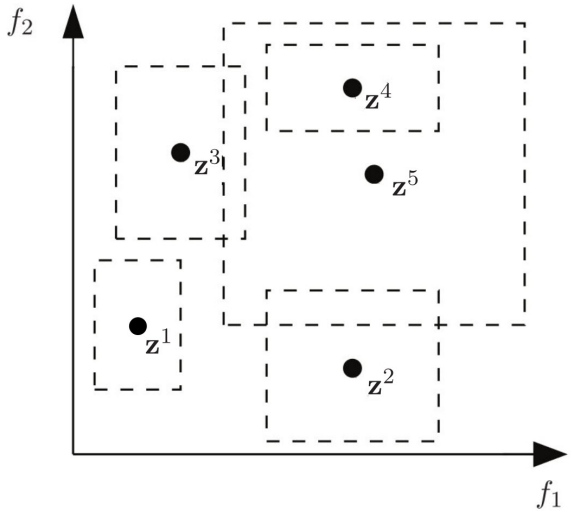


Figure 3.2: Approximated solutions presented in the objective space using bounding boxes.

If $\mathbf{z} = \mathbf{f}(\mathbf{x})$ with confidence vector $\boldsymbol{\varepsilon}$, $\mathbf{w} = \mathbf{f}(\mathbf{y})$ with confidence vector $\boldsymbol{\delta}$, and $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ $\not\prec_u \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, we say that solution $\mathbf{x}$ is probably non-dominated by solution $\mathbf{y}$ ($\mathbf{x} \not\prec_u \mathbf{y}$). This means it is expected that either $\mathbf{x}$ dominates $\mathbf{y}$ or that the solutions are incomparable.

Definition 3.8 (Probable incomparability). The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ is *probably incomparable* with the bounding box $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ $||_u$ $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, iff for every $\mathbf{z}' \in \mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ and $\mathbf{w}' \in \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$: $\mathbf{z}' \,||\, \mathbf{w}'$.

Again, two solutions $\mathbf{x}$ and $\mathbf{y}$ are probably incomparable when their corresponding bounding boxes are probably incomparable ($\mathbf{x} \,||_{\mathrm{u}}\, \mathbf{y}$). In Figure 3.2, $\mathbf{z}^2$ is probably incomparable with $\mathbf{z}^3$.

Finally, when none of the presented relations under uncertainty applies, two solutions are in an undetermined relation.

Definition 3.9 (Undetermined relation). The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ is in an *undetermined relation* with the bounding box $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ $\sim_u \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, iff $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \cap \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta}) \neq \emptyset$.

If $\mathbf{z} = \mathbf{f}(\mathbf{x})$ with confidence vector $\boldsymbol{\varepsilon}$, $\mathbf{w} = \mathbf{f}(\mathbf{y})$ with confidence vector $\boldsymbol{\delta}$, and $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ $\sim_u \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, we say that solution $\mathbf{x}$ is in an undetermined relation with solution $\mathbf{y}$ ($\mathbf{x} \sim_u \mathbf{y}$). This means it is expected that either one solution weakly dominates the other or that the solutions are incomparable. In Figure 3.2, $\mathbf{z}^5$ is in an undetermined relation with $\mathbf{z}^2$, $\mathbf{z}^3$ and $\mathbf{z}^4$.

Two implications can be found between relations under uncertainty. If solution $\mathbf{x}$ probably dominates solution $\mathbf{y}$, then the solution $\mathbf{x}$ is also probably non-dominated by the solution $\mathbf{y}$:

$$\mathbf{x} \prec_u \mathbf{y} \Rightarrow \mathbf{x} \not\prec_u \mathbf{y}. \tag{3.3}$$

Similarly, probable incomparability implies probable Pareto non-dominance:

$$\mathbf{x} \,||_{\mathrm{u}}\, \mathbf{y} \Rightarrow \mathbf{x} \not\prec_u \mathbf{y}. \tag{3.4}$$

If all the solutions are exactly evaluated, that is all their corresponding confidence interval widths equal zero, the relations presented in this section directly translate to those described in Section 3.1.

3.2.2 Relations under Uncertainty with Constraints

Similarly to the Pareto dominance relations (Section 3.1), the relations under uncertainty without constraints (Subsection 3.2.1) are usually applied only if all solutions are feasible. To compare solutions represented with BBs where the feasibility of solutions is uncertain, we need to define a measure of feasibility for solutions represented with BBs. Since BBs are defined on the objective space, we only need to check the feasibility of BBs against constraints on objectives that define the feasible objective value region $\mathcal{F}$. We assume that before checking these constraints the solution has already met constraints on decision values and boundary constraints.

Definition 3.10 (Probable feasibility). The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ is *probably feasible* if $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \cap \mathcal{F} = \mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$, where $\mathcal{F}$ is the feasible objective value region of the problem.

Definition 3.11 (Probable infeasibility). The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ is *probably infeasible* if $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \cap \mathcal{F} = \emptyset$, where $\mathcal{F}$ is the feasible objective value region of the problem.

Definition 3.12 (Undetermined feasibility). The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ has *undetermined feasibility* if $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \cap \mathcal{F} \neq \mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ and $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \cap \mathcal{F} \neq \emptyset$, where $\mathcal{F}$ is the feasible objective value region of the problem.

In the unlikely case of very complex constraints on objectives, it can be difficult to implement and calculate the intersection between BB and $\mathcal{F}$. However, the procedure can be simplified by checking the feasibility only for the points on the vertices of the BB. If all the points are feasible, we can say that the solution is probably feasible; if they are all infeasible, the solution is probably infeasible; and if some points are feasible and others are not, we can say the solution has undetermined feasibility. We can assume this simplification since the widths of the confidence intervals are relatively small and we can presume that the vertices represent the whole BB sufficiently well.

To compare feasible and infeasible solutions represented with BBs, we define the following four relations under uncertainty with constraints.

Definition 3.13 (Probable constrained dominance). The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ *probably constrained-dominates* the bounding box $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \prec_{uc} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, if any of the following conditions is true:

1. The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ is probably feasible and the bounding box $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$ is probably infeasible.
2. The bounding boxes $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ and $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$ are both probably infeasible, but the objective vector $\mathbf{z}$ has a smaller overall constraint violation.
3. The bounding boxes $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ and $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$ are both probably feasible, and $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \prec_{\mathrm{u}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$.
4. The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ is probably feasible, the bounding box $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$ has undetermined feasibility and $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \prec_{\mathrm{u}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$.

If $\mathbf{z} = \mathbf{f}(\mathbf{x})$ with confidence vector $\boldsymbol{\varepsilon}$, $\mathbf{w} = \mathbf{f}(\mathbf{y})$ with confidence vector $\boldsymbol{\delta}$, and $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \prec_{\mathrm{uc}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, then the solution $\mathbf{x}$ probably constrained-dominates the solution $\mathbf{y}$ ($\mathbf{x} \prec_{\mathrm{uc}} \mathbf{y}$).

Definition 3.14 (Probable constrained non-dominance). The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ *is probably constrained-non-dominated* by the bounding box $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \not\succ_{\mathrm{uc}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, if any of the following conditions is true:

1. The bounding boxes $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ and $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$ are probably feasible, and $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \not\prec_{\mathrm{u}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$.

2. The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ is probably feasible, and the bounding box $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$ has undetermined feasibility, and either $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \not\prec_{\mathrm{u}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$ or $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \,||_{\mathrm{u}}\, \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$.

If $\mathbf{z} = \mathbf{f}(\mathbf{x})$ with confidence vector $\boldsymbol{\varepsilon}$, $\mathbf{w} = \mathbf{f}(\mathbf{y})$ with confidence vector $\boldsymbol{\delta}$, and $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \not\prec_{\mathrm{uc}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, then the solution $\mathbf{x}$ is probably constrained-non-dominated by the solution $\mathbf{y}$ ($\mathbf{x} \not\prec_{\mathrm{uc}} \mathbf{y}$).

Definition 3.15 (Probable constrained incomparability). The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ *is probably constrained-incomparable* with the bounding box $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \,||_{\mathrm{uc}}\, \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, if any of the following conditions are true:

1. The bounding boxes $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ and $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$ are probably feasible and $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \,||_{\mathrm{u}}\, \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$.

2. The bounding boxes $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ and $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$ are both probably infeasible, and both objective vectors $\mathbf{z}$ and $\mathbf{w}$ have the same overall constraint violation.

Two solutions $\mathbf{x}$ and $\mathbf{y}$ are probably constrained-incomparable when the corresponding bounding boxes are probably constrained-incomparable ($\mathbf{x} \,||_{\mathrm{uc}}\, \mathbf{y}$).

Definition 3.16 (Undetermined constrained relation). The bounding box $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon})$ is in an *undetermined constrained relation* with the bounding box $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, $\mathrm{BB}(\mathbf{z}, \boldsymbol{\varepsilon}) \sim_{\mathrm{uc}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$, if the two bounding boxes are not in any other constrained relation under uncertainty.

Again, two solutions $\mathbf{x}$ and $\mathbf{y}$ are in an undetermined constrained relation when the corresponding bounding boxes are in an undetermined constrained relation ($\mathbf{x} \sim_{\mathrm{uc}} \mathbf{y}$).

When two solutions are in an undetermined constrained relation, the three following outcomes are possible: (1) the first solution dominates the second one, (2) the second solution dominates the first one, or (3) the solutions are incomparable. We present a possible scenario to clarify why the solutions can be in an undetermined constrained relation due to their feasibility. We compare solution $\mathbf{x}$ with undetermined feasibility, and solution $\mathbf{y}$ with probable feasibility, and $\mathbf{x}$ is probably non-dominated by $\mathbf{y}$. This means that if we were to exactly evaluate solution $\mathbf{x}$ and it would be infeasible, the solution $\mathbf{y}$ would dominate the solution $\mathbf{x}$. This implies that the solutions can be in any relation; hence, by definition, they are in an undetermined constrained relation. Similarly, there are also other cases in which solutions are in an undetermined constrained relation and we need to exactly evaluate at least one of the solutions.

All relations for comparing solutions mentioned in this dissertation are summarized in Tables 3.1–3.3.

3.3 Comparing Solutions under Uncertainty

In iterative optimization algorithms, the process of gradual solution improvement is based on solution comparisons. By comparing solutions, the algorithm finds which solutions are better and promotes them further, while those that are found worse are discarded.

In this section we show the use of constrained relations under uncertainty for comparing two solutions represented with BBs. This comparison can be implemented in any multiobjective optimization algorithm. However, since every algorithm applies a specific search strategy, we present how the relations under uncertainty can be used instead of Pareto dominance relations.

Table 3.1: Relations without uncertainty.

Relation name	Relation symbol	Meaning
Pareto dominance	$\mathbf{z} \prec \mathbf{w}$	$\mathbf{z}$ dominates $\mathbf{w}$
Weak Pareto dominance	$\mathbf{z} \preceq \mathbf{w}$	$\mathbf{z}$ weakly dominates $\mathbf{w}$
Strict Pareto dominance	$\mathbf{z} \prec\prec \mathbf{w}$	$\mathbf{z}$ strictly dominates $\mathbf{w}$
Incompatibility	$\mathbf{z} \,\|\|\, \mathbf{w}$	$\mathbf{z}$ is incomparable with $\mathbf{w}$
Constrained dominance	$\mathbf{z} \prec_{\mathrm{c}} \mathbf{w}$	$\mathbf{z}$ constrained-dominates $\mathbf{w}$

Table 3.2: Relations under uncertainty without constraints.

Relation name	Relation symbol	Meaning
Probable Pareto dominance	$\mathrm{BB}(\mathbf{z}, \varepsilon) \prec_{\mathrm{u}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$	$\mathrm{BB}(\mathbf{z}, \varepsilon)$ probably dominates $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$
Probable Pareto non-dominance	$\mathrm{BB}(\mathbf{z}, \varepsilon) \not\prec_{\mathrm{u}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$	$\mathrm{BB}(\mathbf{z}, \varepsilon)$ is probably non-dominated by $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$
Probable incomparability	$\mathrm{BB}(\mathbf{z}, \varepsilon) \,\|\|_{\mathrm{u}}\, \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$	$\mathrm{BB}(\mathbf{z}, \varepsilon)$ is probably incomparable with $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$
Undetermined relation	$\mathrm{BB}(\mathbf{z}, \varepsilon) \sim_{\mathrm{u}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$	$\mathrm{BB}(\mathbf{z}, \varepsilon)$ is in an undetermined relation with $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$

Table 3.3: Relations under uncertainty with constraints.

Relation name	Relation symbol	Meaning
Probable constrained dominance	$\mathrm{BB}(\mathbf{z}, \varepsilon) \prec_{\mathrm{uc}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$	$\mathrm{BB}(\mathbf{z}, \varepsilon)$ probably constrained-dominates $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$
Probable constrained non-dominance	$\mathrm{BB}(\mathbf{z}, \varepsilon) \not\prec_{\mathrm{uc}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$	$\mathrm{BB}(\mathbf{z}, \varepsilon)$ is probably constrained-non-dominated by $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$
Probable constrained incomparability	$\mathrm{BB}(\mathbf{z}, \varepsilon) \,\|\|_{\mathrm{uc}}\, \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$	$\mathrm{BB}(\mathbf{z}, \varepsilon)$ is probably constrained-incomparable with $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$
Undetermined constrained relation	$\mathrm{BB}(\mathbf{z}, \varepsilon) \sim_{\mathrm{uc}} \mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$	$\mathrm{BB}(\mathbf{z}, \varepsilon)$ is in an undetermined constrained relation with $\mathrm{BB}(\mathbf{w}, \boldsymbol{\delta})$

Nevertheless, it is to be noted that straightforward use of relations under uncertainty instead of Pareto dominance relations is not always possible. When the confidence intervals (at least one) are overlapping, confidence interval reduction procedures have to be applied in order to be able to determine the result of comparison. These additional procedures differ regarding of the type of uncertainty. For example, in the case of surrogate-model-based optimization the additional procedure is exact evaluation, and in the case of noisy optimization the procedure includes additional evaluations that result in reducing the width of the confidence interval. In cases where the width of the confidence interval cannot be changed and the relations between solutions are unknown, another approach needs to be taken, for example, comparison of the approximated values instead of comparing BBs.

When comparing solution $\mathbf{z}$ with confidence vector $\boldsymbol{\varepsilon}$ and solution $\mathbf{w}$ with confidence vector $\boldsymbol{\delta}$, we consecutively check the four possibilities listed below.

1. If $\mathbf{z} \prec_{\text{uc}} \mathbf{w}$, we can consider $\mathbf{z}$ and $\mathbf{w}$ to be in the Pareto dominance relation ($\mathbf{z} \prec \mathbf{w}$).

   Here the solution $\mathbf{z}$ is probably better than the solution $\mathbf{w}$; therefore, no confidence interval reduction is necessary as it would probably not change the dominance relation.

2. If $\mathbf{z} \,||_{\text{uc}}\, \mathbf{w}$, we can consider $\mathbf{z}$ and $\mathbf{w}$ to be incomparable ($\mathbf{z} \,||\, \mathbf{w}$).

   In this case, solutions $\mathbf{z}$ and $\mathbf{w}$ are probably constrained-incomparable. Even if both solutions were exactly evaluated, they would probably still be incomparable and the algorithm would probably still keep both solutions. Hence, no confidence interval reduction is needed.

3. If $\mathbf{z} \not\prec_{\text{uc}} \mathbf{w}$, the algorithm checks $\boldsymbol{\varepsilon}$. If $\boldsymbol{\varepsilon} \neq \mathbf{0}$, the algorithm performs confidence interval reduction on $\mathbf{z}$ and compares the solutions again. If $\boldsymbol{\varepsilon} = \mathbf{0}$, the algorithm performs confidence interval reduction on solution $\mathbf{w}$ and compares the solutions again.

   In this case, the solution $\mathbf{z}$ is probably better in at least one objective and probably not worse in the others. In order to determine whether either solution $\mathbf{z}$ dominates solution $\mathbf{w}$ or they are incomparable, (at least) for one solution the confidence interval reduction needs to be performed. Because $\mathbf{z}$ is more promising, its confidence intervals are checked. If their widths are different from zero, which means that the solution is approximated, the algorithm performs confidence interval reduction on $\mathbf{z}$ and then compares the solutions again. If the confidence interval widths are equal to zero, which means that solution $\mathbf{z}$ is exactly evaluated, then, in order to be able to compare the solutions, the algorithm performs confidence interval reduction on $\mathbf{w}$ and compares the solutions again.

4. If $\mathbf{z} \sim_{\text{uc}} \mathbf{w}$, the algorithm checks the feasibility of solutions. If both solutions have undetermined feasibility, the algorithm randomly chooses one solution and performs confidence interval reduction on it. If one solution has undetermined feasibility, the algorithm performs confidence interval reduction on that solution and compares the solutions again. If both solutions are probably feasible, the algorithm checks the confidence vector of a randomly picked solution. If it is not equal to zero, the algorithm performs confidence interval reduction on this solution and compares the solutions again. If the confidence vector is equal to zero, the algorithm performs confidence interval reduction on the other solution and compares the solutions again.

   In this case, the only way to find out which solution is better is to perform confidence interval reduction on (at least) one solution. Because solutions near the borders of

the feasibility region are usually better, the algorithm first checks and performs confidence interval reduction on these solutions. If both solutions are probably feasible, the algorithm checks whether the first solution is exactly evaluated. If it is not, the algorithm performs confidence interval reduction on it. If it is, the algorithm performs additional confidence interval reduction on the other solution and then compares the solutions again.

3.4 Empirical Proof of Concept

In this section we test the hypothesis that by using the new relations under uncertainty the number of incorrect comparisons is reduced. We experimentally compared multiobjective solutions with uncertainty where the uncertainty comes from solution approximations gained with surrogate models. To be able to compare the number of incorrect comparisons, every solution comparison was performed with relations under uncertainty and with Pareto dominance relations. In addition to comparing approximated solution values, we also compared the exact solution values. This allowed us to monitor the accuracy of comparison of uncertain solutions.

The solutions selected for testing were not generated randomly, but rather produced by the well-known NSGA-II algorithm (Deb et al., 2002). This ensured that the solution comparisons were similar to the comparisons performed in multiobjective evolutionary algorithms and thus provided relevant results.

In every generation NSGA-II creates a new set of solutions, adds them to the current ones and then performs selection on the union to identify the most promising ones. The selection procedure includes comparing every solution with all other solutions to determine its dominance status. On these comparisons we compared the relations under uncertainty with the Pareto dominance relations.

The comparison was performed on three benchmark problems. One is the Poloni optimization problem (Poloni, Giurgevich, Onesti, & Pediroda, 2000) and two are from (Deb, 2001), called OSY and SRN. All of them are two-objective problems.

GP modeling (Rasmussen & Williams, 2006) was used to build surrogate models for solution approximations. For the confidence interval width of the approximation we used two standard deviations (2σ), which corresponds to about 95% of the normal distribution of the approximations. To test the correlation between the surrogate model accuracy and the incorrect comparisons, five different models of increasing accuracy were built – each on larger number of solutions.

The algorithm parameter values used in the experiments were the same for all three problems. They were set as follows:

- population size: 100,
- number of generations: 100,
- number of runs: 30.

For every problem and for every model we calculated the number of incorrect comparisons for each comparison technique. In addition, we calculated the average confidence interval width, and for relations under uncertainty also the number of cases where, in order to be able to compare the solutions, confidence interval reduction procedures (in our case exact evaluations of approximated solutions) were performed.

The results averaged over 30 runs are presented in Tables 3.4–3.6. These results show that by increasing the number of solutions used for building the surrogate model the accuracy of the model increases and the number of incorrect comparisons decreases. The

Table 3.4: Comparison of the proposed relations with Pareto dominance relations on the Poloni problem (average values over 30 runs).

Relation type	Solutions used for surrogate model	Number of comparisons	Incorrect comparisons	Number of comparisons with confidence interval reductions	Proportion of confidence interval reductions [%]	Confidence interval width
Relations under uncertainty	20	3,940,200	1,515	3,635,805	92	26.25
	30		682	3,152,124	80	15.41
	50		138	1,218,337	31	1.29
	100		65	672,384	17	0.012
	200		13	549,380	14	0.002
Pareto dominance relations	20	3,940,200	367,684	/	/	26.25
	30		159,945	/	/	15.41
	50		22,032	/	/	1.29
	100		2,309	/	/	0.012
	200		1,219	/	/	0.002

Table 3.5: Comparison of the proposed relations with Pareto dominance relations on the OSY problem (average values over 30 runs).

Relation type	Solutions used for surrogate model	Number of comparisons	Incorrect comparisons	Number of comparisons with confidence interval reductions	Proportion of confidence interval reductions [%]	Confidence interval width
Relations under uncertainty	20	3,940,200	74,181	2,289,682	58	42.81
	30		21,861	1,934,212	49	25.98
	50		19,342	1,426,775	36	25.05
	100		144	712,298	18	0.07
	200		152	271,821	7	0.03
Pareto dominance relations	20	3,940,200	336,049	/	/	42.81
	30		136,357	/	/	25.98
	50		49,790	/	/	25.05
	100		1,736	/	/	0.07
	200		1,453	/	/	0.03

Table 3.6: Comparison of the proposed relations with Pareto dominance relations on the SRN problem (average values over 30 runs).

Relation type	Solutions used for surrogate model	Number of comparisons	Incorrect comparisons	Number of comparisons with confidence interval reductions	Proportion of confidence interval reductions [%]	Confidence interval width
Relations under uncertainty	20	3,940,200	7,407	2,703,783	69	50.03
	30		16	2,338,535	59	0.074
	50		2	749,258	19	0.099
	100		3	359,952	9	0.022
	200		11	183,625	5	0.009
Pareto dominance relations	20	3,940,200	188,401	/	/	50.03
	30		161	/	/	0.074
	50		543	/	/	0.099
	100		645	/	/	0.022
	200		648	/	/	0.009

reason for the high number of incorrect comparisons using the models built on smaller number of solutions is in the fact that the solutions used for building the surrogate models do not cover the decision space well enough. Due to the lack of information, the solution approximations can be incorrect by a large margin. This can also result in the exact solution values falling out of the bounding boxes. This reflects in some incorrect comparisons also encountered with the relations under uncertainty.

With the increasing number of solutions used for building the surrogate model the average confidence interval width also gets narrower. The narrower the confidence intervals, the smaller the bounding boxes and the number of required additional confidence interval reductions.

Examining the number of incorrect comparisons for the two relation types, we can see that by using the Pareto dominance relations the number of incorrect comparisons is from 3 to 243 times higher than by using relations under uncertainty. Regardless of the accuracy of the surrogate model, we can conclude that by using relations under uncertainty the number of incorrect comparisons is reduced.

As we can see, in order to reduce the number of incorrect comparisons, we have to perform additional confidence interval reductions. This in turn increases the total optimization time, hence a balance between the number of incorrect comparisons and the time spent performing additional confidence interval reductions needs to be found.

3.5 Comparing GP and RF Modeling Techniques

Since the accuracy of the surrogate model in the surrogate-model-based optimization is very important, we decided to compare two different modeling techniques (Mlakar, Tušar, & Filipič, 2014a). In addition to GP we performed the same testing also with the RF methodology. The reason to choose RF was the fact that the methodology is well-known and that it (unlike other methods, such as support vector machines and artificial neural networks), in addition to the approximated solution value, returns also the confidence interval of the approximation.

The testing procedure was the same as with GP, so we used the same optimization problems and also used relations under uncertainty and Pareto dominance relations.

With this additional testing we (i) check if the hypothesis that the relations under uncertainty reduce the number of incorrect comparisons is independent of the surrogate model, and (ii) compare the accuracy of the surrogate models to see which technique is more suitable to use.

For testing purposes we used the RF algorithm as proposed in (Breiman, 2001), implemented in program R (Liaw & M. Wiener, 2002). The number of trees used for building RF was 10,000, and the minimum number of elements in leaves was set to 1. Since building an RF surrogate model is faster than building a GP surrogate model, we, in addition to 20, 30, 50, 100 and 200 exactly evaluated solutions, also built an RF surrogate model from 1,000 exactly evaluated solutions. We tested how much the larger RF surrogate model built from 1,000 exactly evaluated solutions increases the accuracy of the approximations.

The results averaged over 30 runs are presented in Tables 3.7–3.9. The results produced by RF show similar trends as the results by GP. If we look at the number of incorrect comparisons, it can be seen that also with RF the relations under uncertainty reduce the number of incorrect comparisons. Since this is also the case with GP and is independent of the accuracy of a surrogate model, we thus provide empirical support for the hypothesis that the use of relations under uncertainty reduces the number of incorrect comparisons.

If we compare the results gained with the two modeling techniques (Tables 3.4–3.6), we can also see some differences. The main difference is in the width of the confidence

Table 3.7: Comparison of the proposed relations with Pareto dominance relations on the Poloni problem (average values over 30 runs).

Relation type	Solutions used for surrogate model	Number of comparisons	Incorrect comparisons	Number of comparisons with confidence interval reductions	Proportion of confidence interval reductions [%]	Confidence interval width
Relations under uncertainty	20	3,940,200	22,497	3,906,474	99	32.89
	30		5,206	3,937,230	99	31.83
	50		2,180	3,935,723	99	28.42
	100		125	3,930,277	99	23.97
	200		4	3,909,386	99	19.76
	1,000		2	3,619,402	92	12.11
Pareto dominance relations	20	3,940,200	1,021,750	/	/	32.89
	30		965,491	/	/	31.83
	50		1,043,216	/	/	28.42
	100		894,889	/	/	23.97
	200		733,044	/	/	19.76
	1,000		379,928	/	/	12.11

Table 3.8: Comparison of the proposed relations with Pareto dominance relations on the OSY problem (average values over 30 runs).

Relation type	Solutions used for surrogate model	Number of comparisons	Incorrect comparisons	Number of comparisons with confidence interval reductions	Proportion of confidence interval reductions [%]	Confidence interval width
Relations under uncertainty	20	3,940,200	1	2,663,597	68	842.41
	30		0	2,663,597	68	789.04
	50		0	2,663,597	68	767.92
	100		0	2,663,597	68	720.44
	200		0	2,663,597	68	677.79
	1,000		0	2,663,597	68	548.19
Pareto dominance relations	20	3,940,200	885,416	/	/	842.41
	30		770,439	/	/	789.04
	50		810,251	/	/	767.92
	100		683,578	/	/	720.44
	200		661,919	/	/	677.79
	1,000		555,983	/	/	548.19

Table 3.9: Comparison of the proposed relations with Pareto dominance relations on the SRN problem (average values over 30 runs).

Relation type	Solutions used for surrogate model	Number of comparisons	Incorrect comparisons	Number of comparisons with confidence interval reductions	Proportion of confidence interval reductions [%]	Confidence interval width
Relations under uncertainty	20	3,940,200	18	3,384,351	86	359.51
	30		0	3,385,285	86	350.55
	50		0	3,385,242	86	308.94
	100		0	3,384,910	86	266.55
	200		0	3,378,456	86	224.77
	1,000		0	3,133,626	79	139.89
Pareto dominance relations	20	3,940,200	387,854	/	/	359.51
	30		425,691	/	/	350.55
	50		365,606	/	/	308.94
	100		288,611	/	/	266.55
	200		216,634	/	/	224.77
	1,000		136,656	/	/	139.89

intervals. The RF surrogate models are less confident in their approximations, which can be seen by wider confidence intervals. Consequently, the number of comparisons with confidence interval reductions is much higher than this number gained with GP.

In addition to wider confidence intervals, the RF surrogate models are also less accurate. If we compare the number of incorrect comparisons performed with Pareto dominance relations where the bounding boxes are not considered, we can see that the number of incorrect comparisons is higher with the RF surrogate models.

Another difference is in the correlation between the number of solutions used for building the surrogate model and the accuracy of the surrogate model. By increasing the number of solutions used, the RF surrogate models do not improve as quickly as the GP models. Even in cases where 1,000 solutions were used for building the RF surrogate models, the confidence interval widths are not greatly reduced, and the intervals are still much wider in comparison to the confidence intervals gained with the GP models built from 200 solutions.

Comparing the number of incorrect comparisons, we can see that by using relations under uncertainty with RF the results are slightly better than with GP. But the reason for that is in the fact that, because of relatively wide confidence intervals, the number of confidence interval reductions with RF is very high. As a result, most solutions have to be exactly evaluated in order to perform the comparisons. Hence, the reason for a lower number of incorrect comparisons is not in the better surrogate model, but in the fact that more solutions are exactly evaluated and are therefore without uncertainty. Since in surrogate-model-based optimization exactly evaluated solutions are typically "expensive", the modeling technique that exactly evaluates most of the solutions is not very suitable.

To summarize, because of better accuracy and narrower confidence intervals, which result in a lower number of additional confidence interval reductions, GP is a more appropriate modeling technique to use in the surrogate-model-based optimization than RF.

3.6 Discussion

In this chapter we proposed new relations for comparing solutions under uncertainty. The relations under uncertainty are defined on bounding boxes that are based on approximated values and confidence intervals. These relations extend the Pareto dominance relations and, in addition to the confidence intervals, also consider the feasibility of solutions. We presented all possible combinations that can occur when comparing solutions under uncertainty with uncertain feasibility. It was also shown how the new relations under uncertainty can be used for solution comparison in an arbitrary multiobjective optimization algorithm. The comparison of relations under uncertainty with the Pareto dominance relations provided empirical support for our hypothesis that comparing solutions using the proposed relations under uncertainty reduces the possibility of incorrect comparisons. In addition, we also compared GP and RF modeling techniques with relations under uncertainty. The GP modeling provided more accurate approximations with smaller confidence intervals and proved to be more appropriate for surrogate-model-based optimization. Since this method is more suitable, we also used it with the proposed GP-DEMO algorithm described in the next chapter.

Chapter 4

The GP-DEMO Algorithm

As shown in the previous chapter, the use of new relations under uncertainty reduces the possibility of incorrect comparisons of inaccurately approximated solutions. Because of that we decided to design a new surrogate-model-based optimization algorithm with relations under uncertainty used for comparing solutions (Mlakar, Petelin, Tušar, & Filipič, 2013a, 2015). Since evolutionary algorithms without surrogates are known to be capable of finding good solutions to hard optimization problems, the idea is to develop a surrogate-model-based algorithm that uses the same techniques in combination with approximated solutions. For this reason we took an existing evolutionary multiobjective optimization algorithm and modified it for the use with relations under uncertainty. The other parts of optimization procedure, like generating new solutions and selecting the best ones, were kept unchanged. Since relations under uncertainty reduce the possibility of incorrect solution comparisons, we expect to achieve similar results regardless of the difficulty of the problem and of the inaccuracy of the surrogate model approximations. In addition, the algorithm should perform fewer exact solution evaluations during the optimization process due to the use of a surrogate model. In optimization problems where exact solution evaluations are time-consuming, the algorithm should find comparable results faster.

4.1 Algorithm Structure

The GP-DEMO algorithm for surrogate-model-based optimization is, as the name suggests, built upon the DEMO algorithm (see Subsection 2.3.2). The difference between the algorithms is that GP-DEMO approximates all newly created solutions with surrogate models and uses relations under uncertainty to compare them. In cases where the relations between solutions cannot be determined due to the uncertainty, (at least) one solution gets exactly evaluated. This way the best solutions are always kept in the population and worse ones are discarded. Algorithm 4.1 presents the GP-DEMO pseudo-code.

After performing initial tests, we decided to use the SPGP sparse approximation method (see Subsection 2.6.2) for GP modeling as it is of a much lower computational complexity than "full" GP modeling. Although the SPGP is a sparse approximation method, updating the model is a relatively slow operation. Therefore, we decided not to update the model after every newly exactly evaluated solution, but only after every generation. All newly exactly evaluated solutions in the last generation are then used to update the surrogate model. Such an approach seems natural for evolutionary algorithms and can be interpreted as batch learning. It should be noted that only exact evaluations are included in updating the model. This means that after any generation, in the worst case, the number of exactly evaluated solutions is equal to the population size, and in the best case, no solution is exactly evaluated and, as a result, there is no need to update the model. The

Algorithm 4.1: GP-DEMO

Result: A set of exactly evaluated non-dominated solutions

```
Create the initial population P of random individuals;
Exactly evaluate the solutions in P;
Build the initial GP model;
while stopping criterion not met do
    for each individual p_i (i = 1, ..., popSize) from P do
        Create candidate c from parent p_i;
        Approximate c with the GP model;
        Compare c and p_i under uncertainty and keep either the best one or both
        solutions (see Section 4.2);
    end
    if there are more than popSize individuals in P then
        Use selection procedure under uncertainty (see Section 4.3);
    end
    Update the GP model from the set of exactly evaluated solutions;
    Randomly enumerate the individuals in P;
end
Exactly evaluate all approximated solutions on the first front;
```

number of exactly evaluated solutions depends on the quality of the model. In general, the more accurate the model, the lower the number of exactly evaluated solutions.

The update of the SPGP model is implemented as some kind of windowing technique. When updating the surrogate model, n last exactly evaluated solutions are used, where n is the window size. As we use the SPGP sparse approximation for GP modeling, only m exactly evaluated solutions are used as the active set. These m solutions are then used to build the surrogate model. Both parameters, n and m, are the algorithm parameters. It should be noted that the active set is calculated from scratch during every update, which means that the active set of the previous model is not used as the initial active set in the model update. Due to the nature of the optimization process, we do not want to keep the solutions from the whole decision space, but the solutions near the Pareto optimal front. Nevertheless, the GP model's hyperparameters are preserved, as their values are not supposed to change much with each generation/update, and therefore the model optimization converges much more quickly.

Since GP approximates only one output value, we needed to create a separate surrogate model for every objective. The active set and the hypervolume values are set differently for every surrogate model.

As already mentioned, GP-DEMO is based on DEMO, but uses relations under uncertainty and solutions are represented with (approximated) values and confidence intervals. Hence some parts of the DEMO algorithm need to be modified. The procedure for creating a candidate solution from the parent solution is the same as in DEMO, but the comparison of candidate and parent solutions is modified. The procedure is described in Section 4.2.

After comparing candidates to parents and deciding which solutions to keep and which to discard, the next modified part in GP-DEMO is the selection procedure. Here the algorithm limits the number of solutions to the population size. The best solutions are preserved and among the best the spread of solutions is also considered. Since solutions are represented with confidence intervals, this selection procedure needs to be performed differently. The procedure is described in detail in Section 4.3.

4.2 Comparing Parents and Candidates under Uncertainty

Comparison of the candidate and parent solutions in the GP-DEMO algorithm is based on the relations under uncertainty (Chapter 3). In Section 3.3 we described the procedure for comparing two solutions under uncertainty in an arbitrary multiobjective optimization algorithm. Here we specialize this procedure for comparing candidate and parent solutions in a surrogate-model-based multiobjective optimization algorithm. When confidence intervals of at least one objective of two solutions are overlapping, exact solution evaluations are performed in order to be able to determine the result of the comparison.

When comparing candidate $\mathbf{c}$ with confidence vector $\boldsymbol{\varepsilon}$ and parent $\mathbf{p}$ with confidence vector $\boldsymbol{\delta}$, we consecutively check the six possibilities listed below.

1. If $\mathbf{c} \prec_{\text{uc}} \mathbf{p}$, solution $\mathbf{c}$ is added to the population and solution $\mathbf{p}$ is discarded.

   Here the solution $\mathbf{c}$ is probably better than the solution $\mathbf{p}$, therefore no additional evaluations are necessary as they would probably not change the dominance relation.

2. If $\mathbf{p} \prec_{\text{uc}} \mathbf{c}$, solution $\mathbf{p}$ is added to the population and solution $\mathbf{c}$ is discarded.

   This case is similar to the previous one.

3. If $\mathbf{c} \,||_{\text{uc}}\, \mathbf{p}$, both solutions are added to the population.

   In this case, solutions $\mathbf{c}$ and $\mathbf{p}$ are probably incomparable. Even if both solutions were exactly evaluated and thus with confidence interval widths equal to zero, they would probably still be incomparable and the algorithm would still add both solutions to the population. Hence, no additional evaluations are needed in this case.

4. If $\mathbf{c} \not\prec_{\text{uc}} \mathbf{p}$, the algorithm checks $\boldsymbol{\varepsilon}$. If $\boldsymbol{\varepsilon} \neq \mathbf{0}$, the algorithm exactly evaluates $\mathbf{c}$ and compares the solutions again. If $\boldsymbol{\varepsilon} = \mathbf{0}$, the algorithm exactly evaluates $\mathbf{p}$ and compares the solutions again.

   In this case, solution $\mathbf{p}$ is probably better in at least one objective and not worse in the others. In order to determine if either solution $\mathbf{c}$ dominates solution $\mathbf{p}$ or they are incomparable, (at least) one solution needs to be exactly evaluated. Because $\mathbf{c}$ looks more promising, its confidence interval width is checked. If it is different from zero, meaning that the solution is approximated, the algorithm exactly evaluates solution $\mathbf{c}$ and then compares the solutions again. If the confidence interval width is equal to zero, meaning that $\mathbf{c}$ is exactly evaluated, then, in order to be able to compare the solutions, the algorithm exactly evaluates solution $\mathbf{p}$ and compares the solutions again.

5. If $\mathbf{p} \not\prec_{\text{uc}} \mathbf{c}$, the algorithm checks $\boldsymbol{\delta}$. If $\boldsymbol{\delta} \neq \mathbf{0}$, the algorithm exactly evaluates $\mathbf{p}$ and compares the solutions again. If $\boldsymbol{\delta} = \mathbf{0}$, the algorithm exactly evaluates $\mathbf{c}$ and compares the solutions again.

   This case is similar to the previous one, except that the solution $\mathbf{p}$ is now more promising.

6. If $\mathbf{c} \sim_{\text{uc}} \mathbf{p}$, the algorithm checks $\boldsymbol{\varepsilon}$. If $\boldsymbol{\varepsilon} \neq \mathbf{0}$, the algorithm exactly evaluates $\mathbf{c}$ and compares the solutions again. If $\boldsymbol{\varepsilon} = \mathbf{0}$, the algorithm exactly evaluates $\mathbf{p}$ and compares the solutions again.

   In this case, the bounding boxes are overlapping and the only way to determine which solution is better is to exactly evaluate (at least) one solution. Because the candidate (offspring) has the potential to be better than the parent, the algorithm first checks if it is exactly evaluated. If it is not, the algorithm exactly evaluates it. If it is, the algorithm exactly evaluates the parent and then compares the solutions again.

4.3 Selection under Uncertainty

The selection procedure in a multiobjective evolutionary algorithm keeps the size of the population constant and tries to uniformly spread the solutions on the front. The selection procedure in GP-DEMO is based on the selection procedure used in NSGA-II (Algorithm 2.1). This selection procedure involves non-dominated sorting and ranking using the crowding distance metric. In non-dominated sorting all the non-dominated individuals are allocated into the first front and then the non-dominated sorting is applied again to the remaining individuals. In order to identify solutions of the first non-dominated front, each solution is compared with every other solution in the population to find out if it is dominated. At this stage, all individuals in the first non-dominated front are found. In order to find the individuals in the next front, the solutions of the first front are discounted temporarily and the above procedure is repeated. This is repeated also for finding the third and the following fronts. In this way, a sequence of fronts is obtained, where the individuals from the preceding fronts are preferred to those from the subsequent fronts.

The new population is filled in turn with the individuals from the best fronts. If a whole front cannot fit into the population entirely, the individuals from this front are further ranked according to the crowding distance metric.

Sorting based on crowding distance metric prefers individuals from less crowded regions of the objective space to ensure a good spread of solutions. To get an estimate of the density of solutions surrounding a particular solution in the population, the average distance of two solutions on either side of this solution along each of the objectives is calculated. The solutions with the largest crowding distance are included in the next population. If a solution is on the edge of a front its crowding distance is set to maximum. This ensures that the front is as wide as possible.

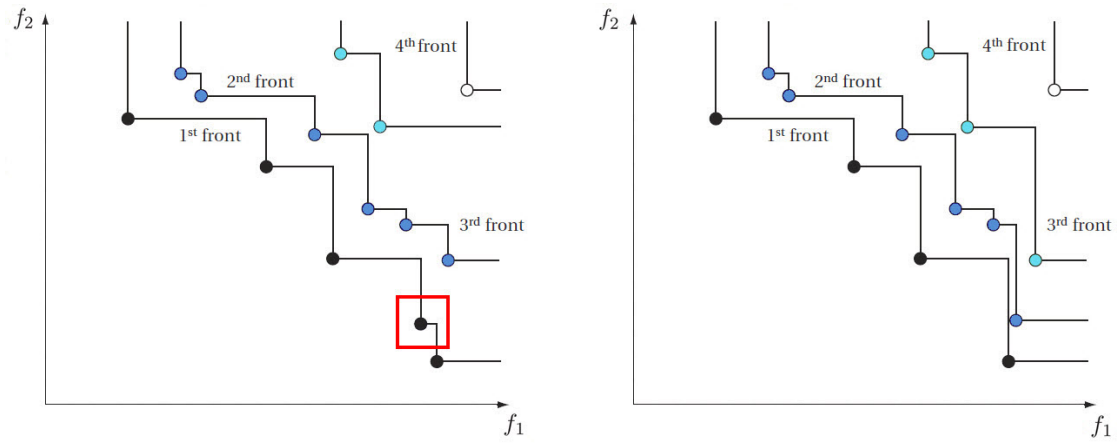


Figure 4.1: Selection procedure in GP-DEMO: (a) approximated first front, (b) corrected first front.

When performing non-dominated sorting with inaccurately approximated solutions, worse solutions can be incorrectly found as dominating better solutions. This results in promising solutions being dominated and possibly discarded. As a consequence the process of finding the best and evenly spread non-dominated solutions is misled. To prevent this from happening, the NSGA-II selection procedure is slightly modified for the use in GP-DEMO.

The main idea of selection procedure in GP-DEMO is to ensure that the solutions on the first front are actually non-dominated. So when comparing solutions with each other the relations under uncertainty are used to determine the dominance relation between solutions. When comparing two solutions if, because of uncertainty, we cannot determine if a solution is dominated, this solution is marked. After comparing all solutions, the possibly dominated solutions are non-dominated but also marked solutions. To ensure that the solutions on the first front are really non-dominated, we exactly evaluate these possibly dominated solutions and again compare them with other solutions.

With this approach we ensure that the front of non-dominated solutions is always accurate, there are no deficiencies in the optimization process, and the possibility of getting stuck in the local optima because of inaccurate approximations is reduced.

An example where the additional exact evaluation changes the first front is shown in Figure 4.1. The examined solution is presented with the black dot representing approximated value and a red bounding box around it (Figure 4.1 (a)). The relation under uncertainty between this solution and the solution underneath it is the probable Pareto non-dominance. So unless we exactly evaluate the examined solution we cannot know whether it is dominated or incomparable. After exact evaluation of this solution it turns out that it is dominated and does not belong to the first front (Figure 4.1 (b)).

This procedure of keeping only non-dominated solutions on the first front is very important in the later stages of optimization when usually all solutions in the population are non-dominated. With this procedure we discard the dominated solutions and thus accelerate the optimization process.

When the number of solutions on the front is greater than the size of population, the ranking using crowding distance metric has to be calculated. The calculation of crowding distance metric is not modified in GP-DEMO. To calculate the crowding distance metric, the approximated objective values without bounding boxes are used. In this step, the algorithm does not exactly evaluate any more approximated solutions to spare as many exact evaluations as possible.

Chapter 5

Numerical Evaluation

To check the quality of the GP-DEMO algorithm, we tested it on various benchmark and real-world optimization problems. This chapter describes the test problems, presents the settings used for testing, describes the surrogate-model-based algorithm GEC used to compare the results, and shows the results of the optimization.

5.1 Test Problems

The test problems used in this dissertation are all minimization problems and can be divided into two groups. The first group consists of known benchmark problems and the second group consists of two real-world problems. The benchmark problems were divided into easier and more complex problems to test all the aspects of the algorithms. The first of the two real-world problems is optimization of the continuous steel casting process and the second one is the problem of finding the best correlation between a simulated and a measured electrocardiogram (ECG). Most of these test problems are two-objective, with the exception of steel casting optimization which involves three objectives.

5.1.1 Benchmark Problems

The benchmark problems are further divided into two subgroups. The first subgroup consists of three problems from (Deb, 2001) called BNH, OSY and SRN. All the problems are constrained and have two objectives. These three problems are relatively simple and are used to measure how many exact evaluations can be saved with surrogate-model-based algorithms in comparison to DEMO.

The second subgroup consists of the WFG test problems introduced in (Huband, Barone, While, & Hingston, 2005). The WFG toolkit is used to construct a suite of problems that provides a thorough test for optimizers. The nine WFG problems, WFG1–WFG9, are formulated in such a manner that each poses a different challenge to the optimizers. The WFG toolkit tests the abilities of surrogate-model-based algorithms to find solutions comparable to the ones gained with DEMO on simple as well as complex problems.

The time needed to exactly evaluate a single solution for any of the benchmark problems is just a few milliseconds, hence exactly evaluating all solutions takes a few seconds.

5.1.2 The Continuous Steel Casting Problem

The continuous casting of steel is a very complex metallurgical process where molten steel is cooled and shaped into semi-manufactures of desired dimensions. The main components

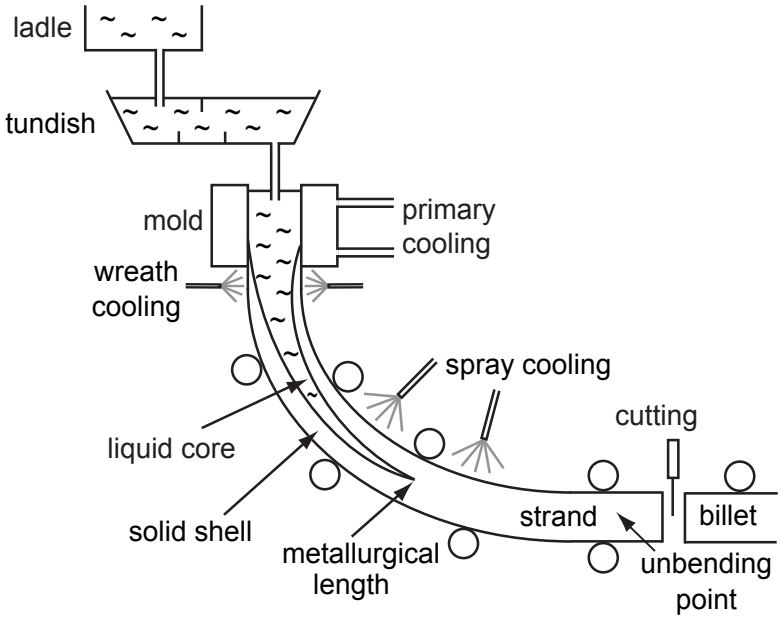


Figure 5.1: A scheme of the steel casting process.

of the casting system (schematically shown in Figure 5.1) are the ladle, tundish, mold and cooling subsystems (Robič & Filipič, 2004).

The process of steel casting starts with molten steel being poured into a ladle from an electric furnace and then led through the tundish that acts as a buffer for the liquid metal, which is then drained into an open-base copper mold. The water-cooling inside the mold cools the mold and the hot steel starts solidifying in contact with it. The water flowing through the channels in the walls of the mold cools the steel. The channels represent the primary cooling subsystem.

Molten steel with a thin solid shell, now called the strand, exits the base of the mold into a spray chamber where it is immediately supported by closely spaced water-cooled rollers. The strand is sprayed with water in the wreath and spray cooling areas in order to increase the rate of solidification. Together, the wreath and spray cooling areas represent the secondary cooling subsystem.

When the steel exits the casting system, it is cut into billets of the desired length. The length of the liquid core in the strand is called the metallurgical length. The metallurgical length, the thickness of the solid shell at the mold exit, and the strand surface temperature at the unbending point have a large effect on the quality of the cast steel.

The optimization problem involves input variables (process parameters), output variables, and the desired output values, determined by the metallurgists at the plant (Mlakar, Tušar, & Filipič, 2012). The task is to find the input variable settings resulting in values of the output variables as close as possible to the desired values. Based on empirical knowledge in the steel production domain, such settings result in high-quality steel.

Since the process of steel casting is expensive, time-consuming and could also be dangerous, it is necessary to have a model to make the optimization of the parameters of the steel casting possible (Šarler et al., 2011). To model the casting, the numerical model of steel casting (Vertnik & Šarler, 2009) was used. The four input variables of this numerical model that are being optimized are the casting speed, the mold outlet coolant temperature, the wreath system coolant flow, and the spray system coolant flow. The lower and the upper bounds for these variables were also determined by the metallurgists (see Table 5.1)

Table 5.1: Input variables and their boundary constraints.

Variable	Lower bound	Upper bound
Casting speed [m/min]	1.50	2.00
Mold outlet coolant temperature [°C]	33	35
Wreath system coolant flow [l/min]	10	40
Spray system coolant flow [l/min]	25	65

Table 5.2: Output variables, their bounds and desired values.

Variable	Lower bound	Upper bound	Desired value
Metallurgical length [m]	10	11	10
Shell thickness [mm]	11	15	13
Surface temperature [°C]	1,115	1,130	1,122.5

and considered as boundary constraints in optimization.

Given the input parameters, the simulator computes the three output variables that are essential for the quality of cast steel: the metallurgical length, the shell thickness and the surface temperature at the unbending point. The lower and upper bounds for the output variables and their desired values are listed in Table 5.2. As an optimization criterion, the difference between the output variable produced by the numerical simulator and its desired value is considered. Thus, the goal is to find such values of the input variables that all the criteria would be 0 or as close to 0 as possible.

The time needed to exactly evaluate a single setting of the input variables, i.e., to numerically simulate the steel casting process, is approximately 2 minutes on a 3.4-GHz Intel Core i7 computer with 8 GB RAM.

5.1.3 The ECG Problem

The second real-world test problem is the problem of finding the best correlation between a simulated and a measured ECG. An ECG is a diagnostic and monitoring tool that records heart activity by measuring, on the body surface, the electrical currents originating in the heart. Modeling the electrical activity of a human heart provides useful insight into the ECG generating mechanisms that can in turn be used to further the understanding of the ECG and improve its diagnostic benefits.

For this problem the ECG simulator presented in (Depolli, Avbelj, & Trobec, 2008) was used. The simulator uses a simplified heart cell model consisting of the action potential (AP), a function which defines the heart cells' electrical activity. Since we focus only on the difference between the T waves of the ECG traces, the full resolution and complexity of this simulator is not needed. Thus, a coarse model consisting of eight times fewer heart cells than the original model was used, enabling faster simulation.

The input parameters (variables of the optimization problem) of the simulator consist of two groups of four parameters. Every group defines the AP of the heart's cell layer.

This optimization problem includes two objectives. For every objective, we first calculate the Pearson correlation coefficient (Rodgers & Nicewander, 1988) between the simulated ECG and the ECG measured on one of the two location points on the body. Then, in order to get a minimization problem, we calculate the objective value (f) by subtracting the Pearson correlation coefficient (PCC) from 1:

$$f = 1 - PCC. \tag{5.1}$$

When the simulated and the measured ECG are fully correlated, their Pearson correlation coefficient is equal to 1 and the objective value is equal to 0.

Solving this optimization problem consists of finding the right combination of variable values for setting the APs of the simulated heart in such a way that the simulated ECG is as close as possible to the measured ECG.

The time needed to exactly evaluate a single solution of the ECG problem is approximately 15 seconds on a 3.4-GHz Intel Core i7 computer with 8 GB RAM.

5.2 Experimental Setup

For the purpose of determining the quality of the results obtained with the GP-DEMO algorithm, a comparison was made with the DEMO algorithm and with the surrogate-model-based algorithm called Generational Evolution Control (GEC) described in the next section.

The algorithm parameter values used for the testing were the same for all three algorithms. They were set as follows:

- maximum number of solution evaluations: 10,000,
- population size: 100,
- weight: 0.5,
- crossover probability: 0.3,
- selection method: non-dominated sorting and crowding distance as in NSGA-II (see Subsection 2.2.1).

The maximum number of solution evaluations for the continuous steel casting problem was 3,000 instead of 10,000 in order to save time and because after 3,000 solution evaluations the hypervolume did not increase any more.

The GP-DEMO and the GEC algorithms used GP modeling to create the surrogate models. The modeling technique and the parameter values used were the same in both algorithms (see Subsection 2.6.1). The width of the confidence interval was equal to two standard deviations. This means that the probability the exactly evaluated solution is within the confidence interval of the approximated solution is 95%. The sizes of the active set and the window set in GP modeling were determined after trying different settings and were chosen as the best compromise between the time needed to build the model and the precision of this model for the approximation. The parameter values were the following:

- active set size: 350,
- window set size: 500.

The window set is relatively small. This ensures that during the optimization process, when approaching the optimum, the surrogate model uses only exactly evaluated solutions close to the optimum. This locality enables the surrogate model to be more precise, thus making the confidence interval narrower.

The experiments were run 20 times for each of the benchmark problems and 10 times for each of the real-world problems because of similar results and in order to save time.

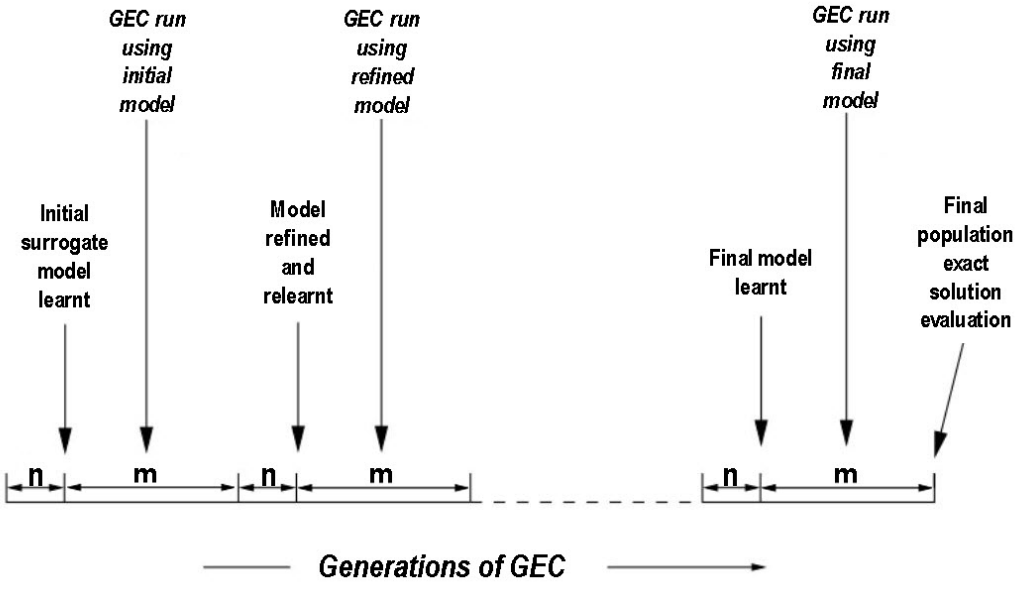


Figure 5.2: A sketch of the GEC algorithm.

5.3 The Generational Evolution Control (GEC) Algorithm

In addition to comparing GP-DEMO to DEMO, we also compared it to another surrogate-model-based algorithm. In this comparison we focused on the quality of results and also on reducing the number of performed exact evaluations.

The algorithm that we call Generational Evolution Control (GEC) is based on the NSGA-II-ANN algorithm from (Deb & Nain, 2007). The basic idea of NSGA-II-ANN is that during the optimization process in some generations all the solutions are exactly evaluated, while in others, all the solutions are approximated with a surrogate model. The schematic presentation of this algorithm is shown in Figure 5.2, where n is the number of generations where only exact evaluations are used and m is the number of generations where only approximations are used. In their paper, the authors describe different versions of the NSGA-II-ANN algorithm. For the purpose of this research we chose the version of the NSGA-II-ANN algorithm that the authors claimed to be better than other versions. In this version the number of generations for exact evaluations is three, followed by seven generations of approximated solutions. This combination is then repeated during the whole optimization process.

To be able to perform a fair comparison, some modifications had to be made to the NSGA-II-ANN algorithm, and this modified algorithm is called GEC. In the GEC algorithm, GP modeling was used for building the surrogate models instead of the artificial neural network (ANN), because the use of a different surrogate model would influence the results. We also used the same modeling parameters (active set size, window set size) for GP-DEMO and GEC. For creation and comparison of solutions, instead of techniques used in NSGA-II, GEC uses techniques implemented in GP-DEMO. These are the same techniques that are used in DEMO. This ensured the same optimization procedure was used in DEMO, GP-DEMO and GEC hence providing a fair comparison.

At the end of the optimization process with GEC, in order to get a comparable hypervolume and the correct number of non-dominated solutions, all the approximated non-dominated solutions were exactly evaluated. This ensured that the front of non-dominated

solutions was accurate and not approximated. Since this was done also with the GP-DEMO algorithm, the final set of non-dominated solutions can be compared between the algorithms and also with set of non-dominated solutions obtained with DEMO.

Algorithm 5.1 presents the pseudo-code of the GEC algorithm. There are two major differences between GEC and GP-DEMO. The first one is based on the relations used for comparing solutions. GP-DEMO uses relations under uncertainty and GEC uses Pareto dominance relations without considering confidence intervals. The second major difference is in the evolution control that determines which solutions are approximated and which exactly evaluated. GP-DEMO uses adaptive evolution control that considers confidence interval widths in determining which solutions are exactly evaluated. GEC, on the other hand, uses the static approach that is the same for all problems. It combines a predefined number of exact solution evaluations with the solution approximations performed using the surrogate model.

Algorithm 5.1: GEC

Result: A set of exactly evaluated non-dominated solutions

```
Create the initial population P of random individuals;
Exactly evaluate the solutions in P;
Build the initial GP model;
while stopping criterion not met do
    if exact solution evaluations are used in this generation then
        for each individual p_i (i = 1, ..., popSize) from P do
            Create candidate c from parent p_i;
            Exactly evaluate c;
            Compare c and p_i using Pareto dominance relations;
        end
        if there are more than popSize individuals in P then
            Use selection procedure as in NSGA-II;
        end
        Add all newly exactly evaluated solutions to the set of exactly evaluated
        solutions;
        if in the next generation solution approximation are used then
            Update the GP model from the set of exactly evaluated solutions;
        end
        Randomly enumerate the individuals in P;
    end
    else
        for each individual p_i (i = 1, ..., popSize) from P do
            Create candidate c from parent p_i;
            Approximate c with the GP model;
            Compare c and p_i using Pareto dominance relations;
        end
        if there are more than popSize individuals in P then
            Use selection procedure as in NSGA-II;
        end
        Randomly enumerate the individuals in P;
    end
end
Exactly evaluate all approximated solutions on the front;
```

5.4 Results

For every problem, for every run and for every algorithm four different measures were obtained:

- number of exact evaluations performed during the optimization process,
- final hypervolume,
- duration of the optimization process,
- number of non-dominated solutions on the final front.

The reference points used for calculating the hypervolume values are presented in Table 5.3. Each point was selected in a way that it covered all non-dominated solutions, and that the region it defined was as small as possible.

Table 5.3: Reference points for calculating the hypervolume values.

Test problem	Hypervolume reference point
BNH	(150, 50)
OSY	(0, 80)
SRN	(250, 50)
WFG1 - WFG9	(10, 10)
Continuous casting	(1, 2, 15)
ECG	(2, 2)

For every problem and for every tested algorithm the best final fronts (according to the hypervolume) are plotted in Figures 5.3–5.15.

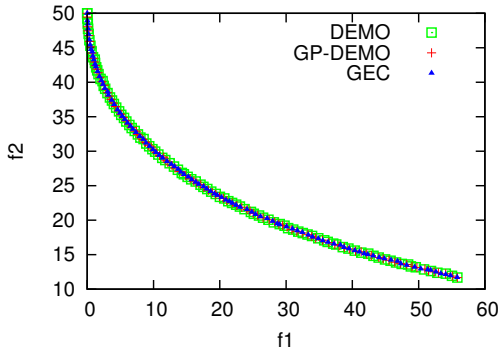


Figure 5.3: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the BNH problem.

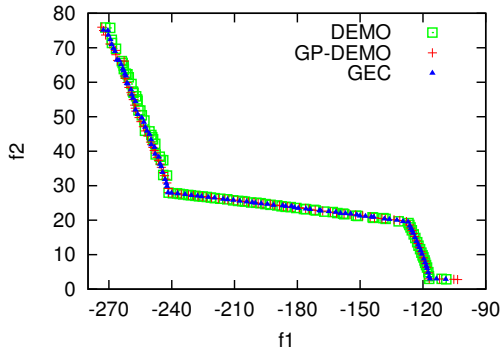


Figure 5.4: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the OSY problem.

The analysis of the results with the detailed comparison of all three algorithms is presented in Section 6.1. The results averaged over all the runs and the standard deviations (σ) of the final hypervolumes are presented in Tables 6.1–6.3.

We also performed a statistical analysis of the hypervolumes obtained with the algorithms on the test problems. The results of this statistical analysis and the statistical significance of these results are presented in Section 6.2.

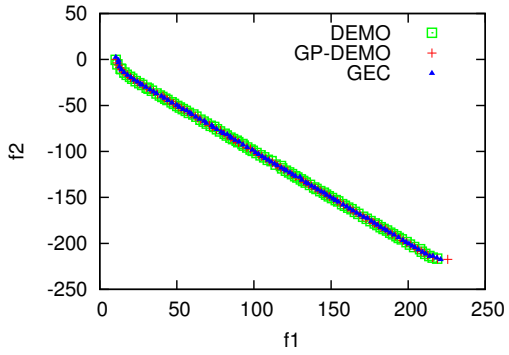


Figure 5.5: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the SRN problem.

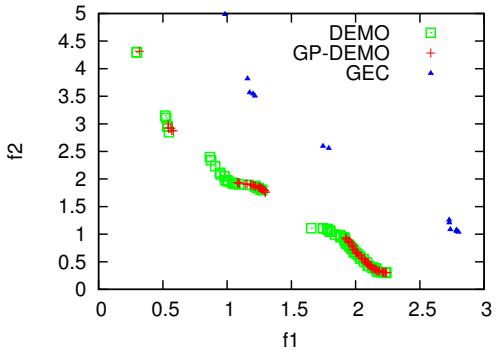


Figure 5.6: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG1 problem.

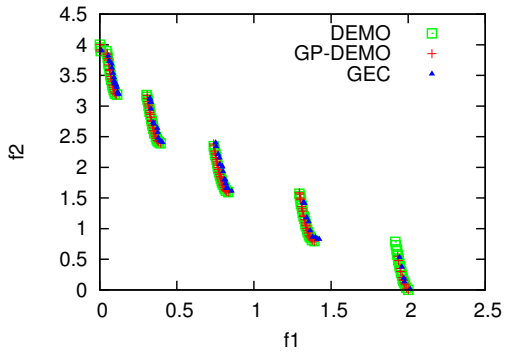


Figure 5.7: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG2 problem.

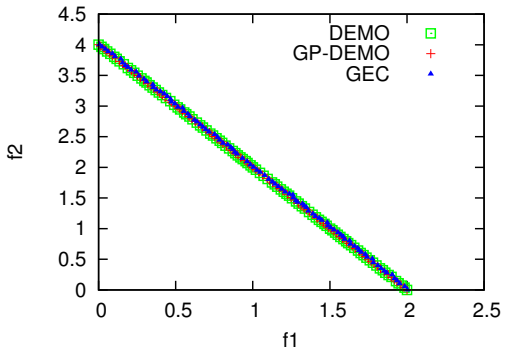


Figure 5.8: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG3 problem.

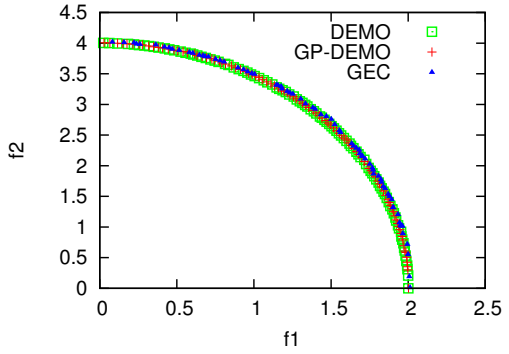


Figure 5.9: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG4 problem.

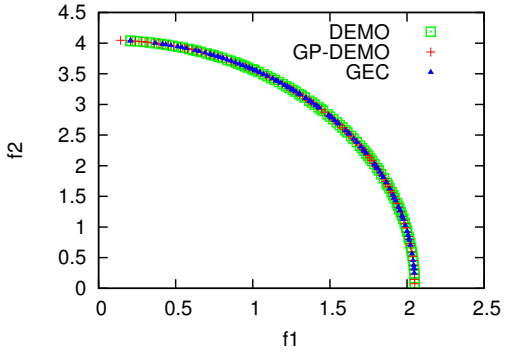


Figure 5.10: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG5 problem.

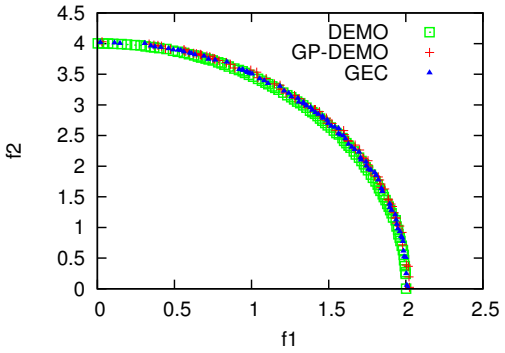


Figure 5.11: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG6 problem.

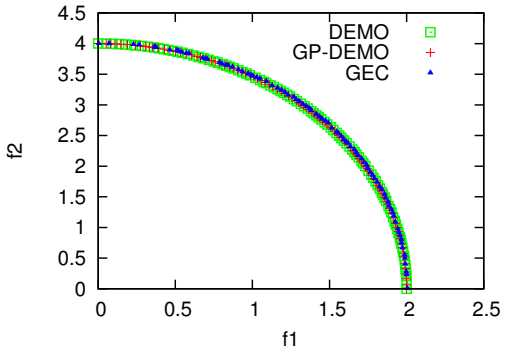


Figure 5.12: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG7 problem.

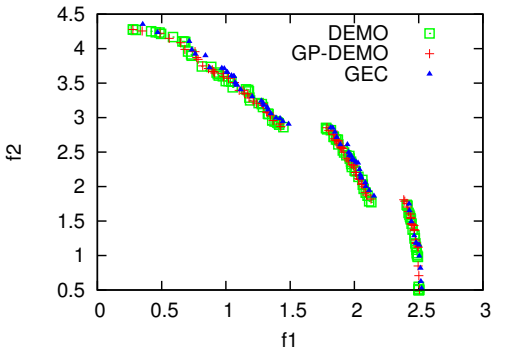


Figure 5.13: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG8 problem.

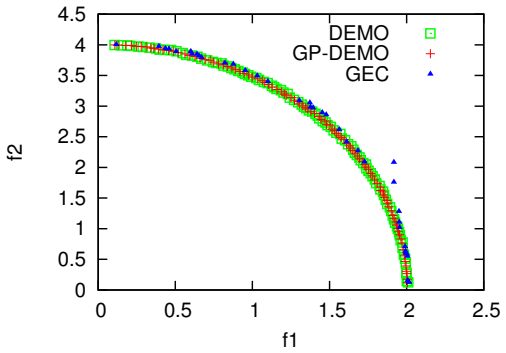


Figure 5.14: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the WFG9 problem.

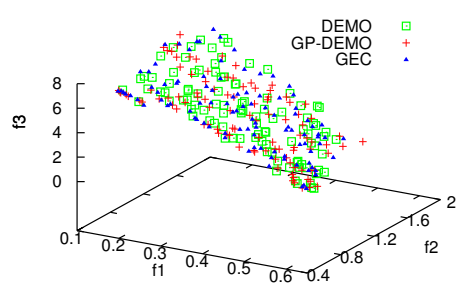


Figure 5.15: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the continuous steel casting problem.

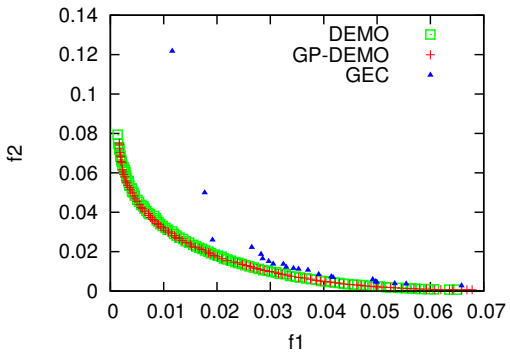


Figure 5.16: Fronts of non-dominated solutions found by DEMO, GP-DEMO and GEC on the ECG problem.

Chapter 6

Analysis of Results

The evaluation of the GP-DEMO algorithm was done by comparing the algorithm with two other multiobjective evolutionary algorithms. The first one is DEMO, which is known to be very effective and was used for comparing the quality of the results. The second one is GEC, a surrogate-model-based multiobjective evolutionary algorithm that was used to compare both the quality of the results and the number of exactly evaluated solutions performed during the optimization process.

6.1 Comparing GP-DEMO, DEMO and GEC

We first compare the results gained on the benchmark problems and then on the real-world problems. Since the benchmark problems are not computationally expensive, we primarily focus on the quality of results and less on the optimization time. The benchmark problems cover various optimization problems so the relation between the problem complexity and the efficiency of the surrogate-model-based algorithms can be seen.

With real-world optimization problems, the exact evaluations are computationally expensive, so the surrogate-model-based algorithms should find results faster.

6.1.1 Benchmark Problems

The first set of benchmark optimization problems that was used for the testing (BNH, OSY and SRN) is composed of relatively simple optimization problems. The results are presented in Table 6.1. Because of the simplicity of the problems, the GP modeling technique is able to create very precise models of their objective functions. Therefore, the approximated solutions have very narrow confidence intervals and are rarely required to be exactly evaluated. The comparison of the solutions with very narrow confidence intervals can usually be done without exactly evaluating them. This results in GP-DEMO and GEC getting almost the same hypervolume as DEMO, but with fewer exactly evaluated solutions. The statistical analysis (Section 6.2) shows that there is no significant difference between the algorithms. On these problems, GP-DEMO exactly evaluated only between 3% (BNH and SRN) and 15% (OSY) of all evaluated solutions. The particularity of the GEC algorithm is that the number of exact evaluations performed during the optimization process differs very little from one problem to another. Since in three out of ten generations the newly created solutions are exactly evaluated and the maximum number of evaluated solutions is equal to 10,000, the number of exactly evaluated solutions is at least 3,000. Additional exact evaluations are performed at the end of the optimization process where all the approximated solutions are exactly evaluated. Because of this strategy the

number of exact evaluations performed with GEC varies between 3,000 and 3,100 on all test problems, which is a little more than 30% of all the evaluated solutions.

Table 6.1: Results on BNH, OSY and SRN test problems.

Problem	Algorithm	Number of exactly evaluated solutions	Hypervolume $\pm$ σ	Optimization time [h:min:s]	Non-dominated solutions
BNH	DEMO	10,000	0.6880 $\pm$ 0.0001	00:00:05	100
	GP-DEMO	401	0.6880 $\pm$ 0.0002	00:05:47	100
	GEC	3,080	0.6879 $\pm$ 0.0002	00:14:18	100
OSY	DEMO	10,000	0.9645 $\pm$ 0.0015	00:00:05	100
	GP-DEMO	1,485	0.9645 $\pm$ 0.0015	00:24:36	100
	GEC	3,078	0.9645 $\pm$ 0.0014	00:23:08	99
SRN	DEMO	10,000	0.9550 $\pm$ 0.0050	00:00:05	100
	GP-DEMO	267	0.9559 $\pm$ 0.0049	00:02:49	100
	GEC	3,091	0.9558 $\pm$ 0.0050	00:14:53	100

The WFG test problems were the second set of benchmark problems. The results are presented in Table 6.2. These problems are known to be hard optimization problems designed to thoroughly test any optimization algorithm. Because of their complexity, the WFG objective functions are very difficult to model. This results in two challenges. The first one is that the confidence intervals of the approximated solutions are wider than on the previous set of test problems. Therefore, many solutions need to be exactly evaluated and the reduction in the number of exactly evaluated solutions was only between 10% and 80%. As the second challenge, the GP model can in some cases be overconfident in what are actually very inaccurate approximations. Such cases occur when the modeled function has sudden changes in values and the GP model identifies these changes as outliers or noise. Since due to its probabilistic nature the GP model smooths them, the approximation is incorrect. When such changes are relatively large compared to the "usual" changes, the variance of the approximation is small and therefore the exactly evaluated value is not inside the confidence interval. In order to prevent solutions from falling out of the confidence interval, we could make the confidence interval wider, e.g., three standard deviations, but since this happens rarely and the final results are still reasonable, we did not change the width of the confidence interval. This case occurred mainly on the WFG1 problem and the hypervolume obtained with GP-DEMO was slightly worse than the hypervolume obtained with DEMO (Figure 5.6). The hypervolume values on other WFG problems are very similar. In some cases, when the fitness function is very complex and the GP model smooths it, even though the exactly evaluated values are inside confidence intervals, the variance is too big, so many solutions get exactly evaluated.

The results gained with GEC on the WFG test problems are the worst on all test problems. The final fronts have fewer solutions and lower hypervolume values. Because the GEC algorithm does not use the confidence intervals when comparing solutions, the mistakes, where an approximated solution incorrectly dominates the other solution, occur more often. Because the surrogate models for the WFG test problems are not very accurate, the approximation errors are larger and more frequent. On all WFG test problems, just before the end of the optimization process, the final fronts gained with GEC consisted of 100 solutions and their hypervolumes were competitive. After exactly evaluating these solutions, the results got worse. The reason why algorithms like GEC face this difficulty is that early in the optimization process an incorrectly approximated solution appears to be very good. This solution then prevents other high-quality solutions from staying in the population because it dominates them. At the end of the optimization process this solution

Table 6.2: Results on WFG test problems.

Problem	Algorithm	Number of exactly evaluated solutions	Hypervolume ± σ	Optimization time [h:min:s]	Non-dominated solutions
WFG1	DEMO	10,000	0.9362 ± 0.0070	00:00:10	72
	GP-DEMO	8,291	0.8932 ± 0.0154	02:42:01	51
	GEC	3,100	0.7557 ± 0.0258	00:38:08	11
WFG2	DEMO	10,000	0.9644 ± 0.0001	00:00:09	100
	GP-DEMO	3,991	0.9640 ± 0.0021	02:22:54	100
	GEC	3,003	0.9607 ± 0.0008	00:38:11	44
WFG3	DEMO	10,000	0.9594 ± 0.0000	00:00:10	100
	GP-DEMO	4,864	0.9594 ± 0.0002	02:21:26	100
	GEC	3,023	0.9578 ± 0.0014	00:37:26	85
WFG4	DEMO	10,000	0.9342 ± 0.0013	00:00:10	100
	GP-DEMO	3,508	0.9298 ± 0.0033	02:33:52	94
	GEC	3,025	0.9252 ± 0.0042	00:37:13	25
WFG5	DEMO	10,000	0.9150 ± 0.0021	00:00:09	100
	GP-DEMO	6,710	0.9157 ± 0.0020	02:10:04	100
	GEC	3,083	0.9123 ± 0.0037	00:38:56	73
WFG6	DEMO	10,000	0.9365 ± 0.0001	00:00:09	100
	GP-DEMO	2,003	0.9228 ± 0.0075	02:24:44	71
	GEC	3,019	0.9307 ± 0.0033	00:37:33	70
WFG7	DEMO	10,000	0.9368 ± 0.0000	00:00:09	100
	GP-DEMO	7,897	0.9368 ± 0.0000	02:52:01	100
	GEC	3,080	0.9147 ± 0.0152	00:38:42	23
WFG8	DEMO	10,000	0.8655 ± 0.0020	00:00:10	98
	GP-DEMO	3,273	0.8641 ± 0.0042	02:22:14	89
	GEC	3,022	0.8560 ± 0.0086	00:36:33	57
WFG9	DEMO	10,000	0.9206 ± 0.0009	00:00:09	100
	GP-DEMO	8,988	0.9203 ± 0.0005	02:42:01	100
	GEC	3,077	0.8880 ± 0.0194	00:35:18	13

is exactly evaluated as a low-quality solution, lowering the hypervolume since a part of the decision space gets weakly covered.

In solving any optimization problem, the solutions in the population tend to get closer together since they all get located on the non-dominated front. Although the solutions are closer, this usually does not affect the number of exact evaluations needed to compare them. If the widths of the confidence intervals stayed the same, the smaller distances would cause the overlap in the confidence intervals and the increase in the number of exact evaluations. But during the optimization process also the accuracy of the surrogate models increases. But since the surrogate models are built from recently evaluated solutions, they adapt to a specific region where the solutions are and increase in accuracy there. The confidence intervals widths of approximations are narrowed and even though the solutions are closer, their confidence intervals do not overlap.

Comparing the times needed for optimization, we can see that DEMO finds final fronts in a few seconds. With GP-DEMO and GEC the optimization times are longer because of the time spent building the surrogate models. The time needed for updating the surrogate models during the optimization process depends on the number of exactly evaluated solutions that are used for training, the complexity of the objective function, and also of the number of objectives, because for every objective a separate surrogate model is created. Since the exact evaluation of 10,000 solutions for benchmark optimization problems takes

just a few seconds, the time needed to update the surrogate models is approximately the same as the optimization time.

6.1.2 Real-World Problems

The analysis of the results from the real-world test problems shows similar findings. The difference with real-world problems is that the whole optimization process takes longer because every exact solution evaluation is computationally expensive. The results are presented in Table 6.3.

Table 6.3: Results on real-world test problems.

Problem	Algorithm	Number of exactly evaluated solutions	Hypervolume $\pm$ σ	Optimization time [h:min:s]	Non-dominated solutions
Continuous casting	DEMO	3,000	0.5084 $\pm$ 0.0113	112:22:37	100
	GP-DEMO	950	0.5117 $\pm$ 0.0057	36:20:05	100
	GEC	1,090	0.5058 $\pm$ 0.0123	39:28:39	97
ECG	DEMO	10,000	0.9988 $\pm$ 0.0001	39:21:19	100
	GP-DEMO	8,135	0.9986 $\pm$ 0.0001	36:35:18	100
	GEC	3,100	0.9937 $\pm$ 0.0045	13:16:04	22

The first real-world problem is the problem of optimizing the continuous casting process in order to get the best possible quality of cast steel. The GP modeling technique was able to create an accurate surrogate model of the continuous steel casting process such that the hypervolume and the size of the final front were almost identical for all three algorithms. The GP-DEMO algorithm exactly evaluated just a little over 30% of all the evaluated solutions. This percentage would be even lower if the stopping criterion would allow more than 3,000 solution evaluations, because at the beginning the surrogate model is not as accurate as it becomes after the updates. With more precise surrogate models, the confidence intervals become narrower and less exact evaluations are needed.

The analysis of the results on the ECG problem shows similar algorithm behavior as seen on the WFG test problems. The surrogate models created during the optimization process approximate solutions quite well. However, because the solutions on the front are very close to zero, and also very close to each other, the confidence intervals of approximations still turn out to be relatively wide. Due to this GP-DEMO achieves only a small saving in the number of exactly evaluated solutions, but still produces the results as good as the ones gained with DEMO. The GEC algorithm has difficulties once again in reaching the same quality of results, and the number of non-dominated solutions on the final front is small.

When comparing the optimization times on real-world problems, we can see that DEMO is now the slowest algorithm. The surrogate-model-based algorithms are designed for hard numerical problems, so they perform best on these kinds of problems. The difference in the optimization times depends on the times needed to build the surrogate models and of the complexity of the problem which results in the number of exact evaluations performed. On the continuous steel casting problem, because a single exact solution evaluation takes more than two minutes, the time needed for optimization with GP-DEMO (and also GEC) is one and a half day, while the optimization with DEMO takes four and a half days, which is three days longer. The ECG problem is more complex to model and a single exact solution evaluation takes less time, hence the difference between GP-DEMO and DEMO is not that big. On this problem the GEC has the shortest optimization time, but at the cost of worse quality of results.

6.2 Statistics

We performed also a statistical analysis of the hypervolumes obtained with the algorithms on the test problems. The idea was to compare the hypervolumes and determine if they differentiate significantly. The significance of these results was checked with the t-test (Zimmerman, 1997) statistical hypothesis test. The t-test compares the actual difference between two means in relation to the variation in the data (expressed as the standard deviation of the difference between the means).

The null hypothesis was that there is no statistically significant difference between the results. The results of this statistical analysis are presented in Table 6.4. They show the probability that there is no significant difference between the hypervolumes obtained with algorithms. For example, the comparison of hypervolumes gained with GP-DEMO and DEMO on the BNH test problem shows that the probability of no significant difference between the hypervolumes is 66%. To reject the null hypothesis, meaning that there is a significant difference between the results, the probability is usually set to be below 5% (or sometimes 1%).

6.3 Border Time

Because the duration of the optimization process is important, we performed an analysis in order to determine in which cases GP-DEMO would be faster than DEMO. In addition to the duration of a single exact solution evaluation, the optimization time depends also on the complexity of the optimization problem. More complex optimization problems are usually very hard to model and the surrogate models are not very accurate. These models are typically not very confident in the solution approximations, thus resulting in solutions under uncertainty having wider confidence intervals. Wider confidence intervals result in more exact solution evaluations performed and for this reason the optimization time on complex optimization problems is longer.

The border time for a specific optimization problem tells us how long a single exact solution evaluation should last for the optimization times of GP-DEMO and DEMO to be equal. Since the benchmark problems cover optimization problems with various complexities, we calculated the border time for every benchmark problem. The calculated border times are shown in Table 6.5. For example, for the BNH problem, if a single exact solution evaluation took 0.04 seconds, GP-DEMO and DEMO would be equally fast. Of course, if the border time was longer, GP-DEMO would be faster.

We can see that the border times vary from less than a second on easier problems to a few seconds on harder problems, where more solutions are exactly evaluated with GP-DEMO. The border times could be further reduced by optimizing the GP modeling procedure. For example, the update of the surrogate models could be made parallel for every objective and also the update of the GP models could perhaps be performed less often.

When solving a new optimization problem, if the time needed for a single exact solution evaluation is known and the difficulty of the optimization problem can be estimated, these border times can help us choose which algorithm to use in order to get good results as fast as possible.

6.4 Discussion

To asses the performance of the proposed surrogate-model-based algorithm GP-DEMO, we tested it on a set of benchmark problems and real-world problems and compared its

Table 6.4: The probability of no significant difference between hypervolumes.

Test problems	Comparisons between algorithms		
		GP-DEMO	GEC
BNH	DEMO	0.6599	0.0389
	GP-DEMO		0.0883
		GP-DEMO	GEC
OSY	DEMO	0.8636	0.8351
	GP-DEMO		0.9740
		GP-DEMO	GEC
SRN	DEMO	0.5665	0.6407
	GP-DEMO		0.9165
		GP-DEMO	GEC
WFG1	DEMO	$1.0 \cdot 10^{-11}$	$2.8 \cdot 10^{-19}$
	GP-DEMO		$1.4 \cdot 10^{-19}$
		GP-DEMO	GEC
WFG2	DEMO	0.1084	$3.6 \cdot 10^{-14}$
	GP-DEMO		$7.5 \cdot 10^{-6}$
		GP-DEMO	GEC
WFG3	DEMO	0.1484	$4.4 \cdot 10^{-5}$
	GP-DEMO		$6.6 \cdot 10^{-5}$
		GP-DEMO	GEC
WFG4	DEMO	$6.5 \cdot 10^{-6}$	$3.6 \cdot 10^{-9}$
	GP-DEMO		0.0004
		GP-DEMO	GEC
WFG5	DEMO	0.3109	0.0070
	GP-DEMO		0.0011
		GP-DEMO	GEC
WFG6	DEMO	$1.2 \cdot 10^{-7}$	$1.4 \cdot 10^{-7}$
	GP-DEMO		0.0002
		GP-DEMO	GEC
WFG7	DEMO	0.0047	$3.2 \cdot 10^{-6}$
	GP-DEMO		$3.3 \cdot 10^{-6}$
		GP-DEMO	GEC
WFG8	DEMO	0.1845	$8.2 \cdot 10^{-5}$
	GP-DEMO		0.0007
		GP-DEMO	GEC
WFG9	DEMO	0.1458	$4.3 \cdot 10^{-7}$
	GP-DEMO		$5.6 \cdot 10^{-7}$
		GP-DEMO	GEC
Continuous casting	DEMO	0.4456	0.6487
	GP-DEMO		0.2143
		GP-DEMO	GEC
ECG	DEMO	0.0142	0.0060
	GP-DEMO		0.0070

Table 6.5: Border times (in seconds) for the benchmark test problems.

Test problem	Border time [s]
BNH	0.04
OSY	0.2
SRN	0.02
WFG1	5.7
WFG2	1.4
WFG3	1.6
WFG4	1.4
WFG5	3.4
WFG6	1.1
WFG7	4.9
WFG8	1.3
WFG9	9.5

results with the results obtained with DEMO and GEC. The measures that we compared were the number of exactly evaluated solutions, hypervolume value, optimization time and the number of non-dominated solutions on the final front.

Comparing the hypervolume and the number of non-dominated solutions on the final fronts, DEMO, as expected, obtained the best results on all problems. With GP-DEMO we wanted to achieve similar results but with a smaller number of exact evaluations. The comparison of gained results shows that on the simpler problems the quality of results is almost the same and the number of exactly evaluated solutions is substantially lower. On the more complex optimization problems GP-DEMO still achieves similar results to DEMO. The statistical tests show that on some WFG problems there is a statistically significant difference between the results, but the difference is very very small. Also the average number of non-dominated solutions on the final front is similar, and the presentation of the best fronts in Figures 5.6–5.14 shows that the fronts are almost overlapping. The small difference in the results is caused because of the inaccurate approximations, where after exactly evaluating the solutions, the objective values fall out of the predicted confidence intervals. The reason for that is in the complexity of the problem that includes fast changes in objective function. This complexity cannot be followed by GP, so it is seen as an outlier or noise and, due to the probabilistic nature, smoothed.

To put GP-DEMO results in perspective, we compared them with GEC. If the GEC algorithm got similar results, that would mean that GP-DEMO procedure is not better and that the relations under uncertainty do not prevent the worsening of results due to the inaccurate approximations.

On simpler problems the results are similar, which is expected. The surrogate models are very accurate and the number of incorrect comparisons is very low (tests performed in Section 3.4) so the use of relations under uncertainty does not influence the results that much. On the other hand, with the more complex optimization problems the surrogate models are not very accurate. This causes many incorrect comparisons due to the inaccurately approximated solutions. The hypervolumes obtained with GEC are therefore worse on almost all WFG problems. In addition, the number of non-dominated solutions is also considerably lower. Since both GP-DEMO and GEC use the same modeling technique, this difference is due to relations under uncertainty used in GP-DEMO.

Comparison of the optimization times shows that DEMO is the most suitable for the benchmark problems, but it is not suitable for computationally expensive real-world problems. In contrast, both surrogate-model-based evolutionary multiobjective algorithms are more suitable for computationally expensive real-world problems, because the time needed

for training and updating the surrogate models on the benchmark problems takes most of the optimization time.

Let us sum up the comparisons. Firstly, GP-DEMO performed better than GEC, since it spent a smaller number of exact evaluations on easier problems and produced better results on more complex problems. Secondly, if the solution evaluations are very fast, the DEMO algorithm performs best, but on real-world problems, where exact solution evaluations take more time, GP-DEMO becomes more suitable because the optimization times are shorter and the quality of the results is comparable.

In order to be able to determine when to choose GP-DEMO instead of DEMO, we decided to calculate border times for the benchmark problems. The border time defines how long a single exact solution evaluation should last in order for the optimization times of GP-DEMO and DEMO to be equal. If a single exact solution evaluation is longer than the border time, the optimization time of DEMO is longer than the optimization time of GP-DEMO. The border time depends on the complexity of the problem and it varies from less than a second to a few seconds. Therefore, we can say that in the case of a multiobjective optimization problem where a single exact solution evaluation takes more than the expected border time and the quality of the results is important, GP-DEMO is the algorithm to be used.

Chapter 7

Conclusions

We conclude the dissertation with an overview of the work done. Further, we state the achieved scientific contributions and present the ideas for future work.

7.1 Summary

Optimization problems with time-consuming solution evaluations are usually solved with surrogate-model-based optimization algorithms. If an optimization problem is very complex and hard to model, solution approximations can be inaccurate. Comparison of inaccurately approximated solutions can result in incorrect relations between solutions. As a consequence, high-quality solutions can be discarded, or the applied optimization algorithm can get stuck in a local optimum.

To overcome these difficulties, this dissertation proposed a surrogate-model-based multiobjective evolutionary algorithm that, in comparison to other multiobjective evolutionary algorithms (MOEAs), finds comparable solutions with less exact solution evaluations. To achieve this, we defined new relations under uncertainty for comparing solutions represented with approximated values and confidence intervals. Based on these relations we developed a new surrogate-model-based multiobjective evolutionary algorithm called GP-DEMO.

Relations under uncertainty extend the Pareto dominance relations and, in addition to the confidence intervals, also consider the feasibility of solutions. They are defined on bounding boxes that are calculated from approximated values and confidence intervals.

We compared the relations under uncertainty with the Pareto dominance relations and provided empirical support for the hypothesis that comparing solutions using the newly proposed relations reduces the possibility of incorrect comparisons and prevents inaccurate approximations from spoiling the results. Furthermore, we compared the RF and the GP modeling techniques and showed that GP modeling is more suitable to use as a surrogate model. In addition, we also demonstrated how the new relations can be applied to compare solutions in an arbitrary multiobjective optimization algorithm.

The relations under uncertainty are also used in the GP-DEMO algorithm for all solution comparisons. GP-DEMO is based on the DEMO algorithm and uses GP modeling to build surrogate models for solution approximations. Due to uncertainty the procedures for comparing the candidate and parent solutions, and the selection procedure for maintaining the population size are modified. Both procedures determine which solutions get exactly evaluated to reduce the number of incorrect comparisons, and find comparable results to DEMO with as few exact evaluations as possible.

We compared GP-DEMO with two other multiobjective evolutionary algorithms. The first one was DEMO and the second one was another surrogate-model-based algorithm

called Generational Evolution Control (GEC). We chose these two algorithms in order to compare both the quality of the results and the reduction of the number of exact evaluations performed. The algorithms were tested on 12 benchmark and two computationally expensive real-world problems.

The comparison provided empirical support for the hypothesis that GP-DEMO in comparison to other MOEAs produces comparable results with fewer exact solution evaluations. The empirical analysis of the results showed that GP-DEMO and DEMO achieve similar hypervolumes and similar number of non-dominated solutions, but GP-DEMO needs considerably less exact evaluations. In addition, GP-DEMO achieves better results than GEC. Which algorithm needs less exact evaluations depends on the type of optimization problem.

In order to determine when to use GP-DEMO instead of DEMO, we calculated border times. The border time for a specific optimization problem tells us how long a single exact solution evaluation should last, in order for the optimization times of GP-DEMO and DEMO to be equal. The border time correlates mainly with the number of exactly evaluated solutions which depends on the ability to accurately model the optimization problem. If the optimization problem is simple and can be modeled accurately, the approximated solutions have narrow confidence intervals and the number of exact solution evaluations is small, and thus the border time for this problem is short. If the optimization problem is very complex and hard to model, the border time is longer.

Depending on the appraised complexity of the problem, the border time can be estimated. Therefore, if a single exact solution evaluation takes more than the estimated border time and the quality of the results is important, we can conclude that for this problem GP-DEMO is a more appropriate choice than DEMO.

7.2 Scientific Contributions

The contributions of this dissertation to science closely match our initial expectations. The presented work led to the following original scientific contributions:

- **Definition of new relations for comparing solutions under uncertainty.** The relations under uncertainty are based on bounding boxes that are calculated from approximated values and confidence intervals. New relations extend Pareto dominance relations and are suitable for comparing solutions under uncertainty and can also be used for comparing solutions with uncertain feasibility. In addition, the relations under uncertainty also offer the possibility of determining dominance status of solutions without necessarily knowing their exact values.

- **Reduced possibility of incorrect solution comparisons by using relations under uncertainty.** We performed extensive testing on several multiobjective optimization problems with various surrogate models. We confirmed that in comparison to Pareto dominance relations the new relations under uncertainty reduce the possibility of incorrect solution comparisons.

- **A new surrogate-model-based multiobjective optimization algorithm (GP-DEMO).** The algorithm is suited for solving multiobjective optimization problems where exact solution evaluations are computationally expensive. The algorithm relies on relations under uncertainty and tends to find solutions comparable to other multiobjective evolutionary algorithms, but reduces the number of the needed exact solution evaluations.

- **Guidelines on when to use the GP-DEMO algorithm.** Depending on the type of the optimization problem and the time needed for a single exact solution

evaluation, we proposed common guidelines on when GP-DEMO should be used instead of DEMO.

7.3 Future Work

Despite successfully carrying out the experiments and proving the efficiency of relations under uncertainty and the GP-DEMO algorithm, we detected several possibilities for improving the results.

The focus of future work will be on improving the GP-DEMO algorithm. A possibility for its improvement is to speed up the optimization process by updating surrogate models in parallel. Since we build a separate surrogate model for every objective, the time needed to build and update them in parallel would be reduced significantly.

In addition to the speed improvement, application of other surrogate models could be tested. Since different modeling techniques could be suitable for different optimization problems, choosing an appropriate surrogate model could improve the results. There are various modeling techniques that are known to be very effective, but their approximations do not return confidence intervals. To be able to use them with GP-DEMO, we have to adapt them in order to estimate confidence intervals for the approximations (Bosnić & Kononenko, 2009). One way of doing that is to use the local sensitivity analysis (Bosnić & Kononenko, 2008) that aims at determining how much the variation of input can influence the output. The approach locally modifies the learning set in a controlled manner in order to explore the sensitivity of the surrogate model in a particular part of the objective space. The sensitivity is thus related to changes of the approximations of the surrogate model when the learning set is slightly modified. From the sensitivity the confidence interval of the approximation can be obtained.

The next possibility to improve GP-DEMO would be to dynamically change the condition that determines when to update surrogate models. Currently, surrogate models are updated after every algorithm generation. At the beginning of the optimization process when there are not many exactly evaluated solutions, only a few newly exactly evaluated solutions can greatly improve the accuracy of the surrogate model. In the final stages of the optimization process, the new solutions are usually very close to the existing ones, hence the new solutions usually do not change the surrogate model a lot. To take advantage of this property, we could update surrogate models adaptively – more often at the early stages of the optimization process and less often later on.

Another possibility to further improve the results would be to apply several surrogate models instead of just one. Every solution could be approximated with all included models, and the surrogate model returning the narrower confidence interval would be considered as the most confident and thus its approximation would be used. For every objective several surrogate models, each suitable for a specific type of optimization problems, would be built. As a result, combined approximations would be more accurate irrespectively of the type of the optimization problem.

Alternatively, two surrogate models could be used at the same time, one as a local and the other as a global model. The modeling techniques can either be different (e.g., one fast and suitable for modeling more solutions, and the other slower and more accurate even with less solutions) or the same but with a different set of solutions applied to every technique. For the global model we would use all exactly evaluated solutions and try to get the best spread of solutions to obtain a comprehensive overview of the whole objective space. For the local model we would be concentrating only on the current best solutions and thus the surrogate model would be very accurate around non-dominated solutions. During the optimization process, the local model would be used more often for approximations, while

the global model would be used in cases where solutions would fall out of the region where the local model is very accurate. If in these cases only the local model was used, it could, due to the lack of other solutions in the neighborhood, treat these solutions as noise. This would result in very inaccurate approximations and the actual solution values could be out of the predicted confidence intervals, which could result in promoting worse solutions and discarding high-quality solutions. The use of global model would prevent these difficulties and improve the optimization process.

Using more than one surrogate model can improve optimization results and reduce the number of exact evaluations. But on the other hand, the time needed for building additional surrogate models and the time needed for additional solution approximations increase the overall optimization time. Hence, the application of this approach has to be additionally tested to determine its suitability.

References

Ažman, K. & Kocijan, J. (2007). Application of Gaussian processes for black-box modelling of biosystems. *ISA Transactions*, *46*(4), 443–457.

Babbar, M., Lakshmikantha, A., & Goldberg, D. E. (2003). A modified NSGA-II to solve noisy multiobjective problems. In *Genetic and evolutionary computation conference (GECCO), late-breaking papers* (pp. 21–27).

Back, T., Hammel, U., & Schwefel, H.-P. (1997). Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary computation*, *1*(1), 3–17.

Basseur, M. & Zitzler, E. (2006). A preliminary study on handling uncertainty in indicator-based multiobjective optimization. In *Applications of evolutionary computing* (Vol. 3907, pp. 727–739). Lecture Notes in Computer Science. Springer, Berlin.

Bosnić, Z. & Kononenko, I. (2008). Estimation of individual prediction reliability using the local sensitivity analysis. *Applied Intelligence*, *29*(3).

Bosnić, Z. & Kononenko, I. (2009). An overview of advances in reliability estimation of individual predictions in machine learning. *Intelligent Data Analysis*, *13*(2), 385–401.

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*(2), 123–140.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Chafekar, D., Shi, L., Rasheed, K., & Xuan, J. (2005). Multiobjective GA optimization using reduced models. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications*, *35*(2), 261–265.

Cressie, N. A. (1993). *Statistics for spatial data*. Wiley, New York.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. Wiley, New York.

Deb, K. & Nain, P. (2007). An evolutionary multi-objective adaptive meta-modeling procedure using artificial neural networks. In *Evolutionary computation in dynamic and uncertain environments* (Vol. 51, pp. 297–322). Studies in Computational Intelligence. Springer, Berlin.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197.

Depolli, M., Avbelj, V., & Trobec, R. (2008). Computer-simulated alternative modes of U-wave genesis. *Journal of Cardiovascular Electrophysiology*, *19*(1), 84–89.

Emmerich, M. (2005). *Single- and multi-objective evolutionary design optimization assisted by Gaussian random field metamodels* (PhD Thesis, University of Dortmund).

Emmerich, M., Deutz, A., & Klinkenberg, J. (2011). Hypervolume-based expected improvement: monotonicity properties and exact computation. In *2011 IEEE congress on evolutionary computation (CEC)* (pp. 2147–2154). IEEE, Piscataway.

Emmerich, M., Giannakoglou, K. C., & Naujoks, B. (2006). Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, *10*(4), 421–439.

Emmerich, M., Giotis, A., Özdemir, M., Bäck, T., & Giannakoglou, K. (2002). Metamodel-assisted evolution strategies. In *Parallel problem solving from nature – PPSN VII* (Vol. 2439, pp. 361–370). Lecture Notes in Computer Science. Springer, Berlin.

Emmerich, M. & Naujoks, B. (2004). Metamodel assisted multiobjective optimisation strategies and their application in airfoil design. In *Adaptive computing in design and manufacture VI* (pp. 249–260). Springer, London.

Fieldsend, J. E. & Everson, R. M. (2005). Multi-objective optimisation in the presence of uncertainty. In *2005 IEEE congress on evolutionary computation (CEC)* (Vol. 1, pp. 243–250).

Gong, D., Qin, N., & Sun, X. (2010). Evolutionary algorithms for multi-objective optimization problems with interval parameters. In *2010 IEEE fifth international conference on bio-inspired computing: theories and applications (BIC-TA)* (pp. 411–420).

Grašič, B., Mlakar, P., & Božnar, M. Z. (2006). Ozone prediction based on neural networks and Gaussian processes. *Nuovo Cimento C*, *29*, 651–661.

Grierson, D. E. & Pak, W. H. (1993). Optimal sizing, geometrical and topological design using a genetic algorithm. *Structural Optimization*, *6*(3), 151–159.

Gunter, R. (2001). A partial order approach to noisy fitness functions. In *2001 IEEE congress on evolutionary computation (CEC)* (Vol. 1, pp. 318–325).

Hardy, R. L. (1971). Multiquadric equations of topography and other irregular surfaces. *Journal of Geophysical Research*, *76*(8), 1905–1915.

Huband, S., Barone, L., While, L., & Hingston, P. (2005). A scalable multi-objective test problem toolkit. In *2005 evolutionary multi-criterion optimization (EMO)* (Vol. 3410, pp. 280–295). Lecture Notes in Computer Science. Springer, Berlin.

Ishibuchi, H., Kaige, S., & Narukawa, K. (2005). Comparison between lamarckian and baldwinian repair on multiobjective 0/1 knapsack problems. In *Evolutionary multi-criterion optimization* (pp. 370–385). Springer.

Jin, Y. (2003). A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, *9*, 3–12.

Jin, Y. (2011). Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, *1*(2), 61–70.

Jin, Y. & Branke, J. (2005). Evolutionary optimization in uncertain environments – A survey. *IEEE Transactions on Evolutionary Computation*, *9*(3), 303–317.

Jin, Y., Olhofer, M., & Sendhoff, B. (2001). Managing approximate models in evolutionary aerodynamic design optimization. In *2001 IEEE congress on evolutionary computation (CEC)* (Vol. 1, pp. 592–599). IEEE, Piscataway.

Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, *13*(4), 455–492.

Južnič-Zonta, Ž., Kocijan, J., Flotats, X., & Vrečko, D. (2012). Multi-criteria analyses of wastewater treatment bio-processes under an uncertainty and a multiplicity of steady states. *Water Research*, *46*(18), 6121–6131.

Kleijnen, J. P. C. & Beers, W. C. M. V. (2004). Application-driven sequential designs for simulation experiments: Kriging metamodelling. *Journal of the Operational Research Society*, *55*(8), 876–883.

Knowles, J. (2006). ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, *10*(1), 50–66.

Kolmogorov, A. N. (1941). Interpolation und extrapolation von stationären zufälligen folgen. *Bulletin Academy Science USSR Series in Mathematics*, *5*, 3–14.

Krige, D. G. (1951). A statistical approach to some basic mine valuation problems on the Witwatersrand. *Journal of the Chemical, Metallurgical and Mining Society of South Afrika*, *52*(6), 119–139.

Li, G., Li, M., Azarm, S., Hashimi, S. A., Ameri, T. A., & Qasas, N. A. (2009). Improving multi-objective genetic algorithms with adaptive design of experiments and online metamodeling. *Structural and Multidisciplinary Optimization*, *37*(5), 447–461.

Liaw, A. & Wiener, M. (2002). Classification and regression by randomForest. *R News*, *2*(3), 18–22.

Likar, B. & Kocijan, J. (2007). Predictive control of a gas-liquid separation plant based on a Gaussian process model. *Computers & Chemical Engineering*, *31*(3), 142–152.

Limbourg, P. (2005). Multi-objective optimization of problems with epistemic uncertainty. In *2005 evolutionary multi-criterion optimization (EMO)* (Vol. 3410, pp. 413–427). Lecture Notes in Computer Science. Springer, Berlin.

Limbourg, P. & Aponte, D. E. S. (2005). An optimization algorithm for imprecise multi-objective problem functions. In *2005 IEEE congress on evolutionary computation (CEC)* (Vol. 1, pp. 459–466).

MacKay, D. J. C. (1998). Introduction to Gaussian processes. In *Neural networks and machine learning, NATO ASI Series* (pp. 133–166). Kluwer Academic Press.

Matheron, G. (1973). The intrinsic random functions and their applications. *Advances in Applied Probability*, *5*, 439–468.

Mlakar, M., Petelin, D., Tušar, T., & Filipič, B. (2013a). GP-DEMO: Differential evolution for multiobjective optimization based on Gaussian process models. In *Proceedings of the 22nd international conference on multiple criteria decision making (MCDM)* (p. 203). Malaga, Spain.

Mlakar, M., Petelin, D., Tušar, T., & Filipič, B. (2015). GP-DEMO: Differential evolution for multiobjective optimization based on Gaussian process models. *European Journal of Operational Research*, *243*(2), 347–361.

Mlakar, M., Tušar, T., & Filipič, B. (2012). Discrete vs. continuous multiobjective optimization of continuous casting of steel. In *Proceedings of the genetic and evolutionary computation conference, GECCO '12* (pp. 587–590). ACM, NY.

Mlakar, M., Tušar, T., & Filipič, B. (2013b). Primerjava rešitev ob negotovosti v večkriterijski optimizaciji. In *Proceedings of the 16th international multiconference information society, IS 2013* (Vol. A, pp. 86–89). Jožef Stefan Institute, Ljubljana.

Mlakar, M., Tušar, T., & Filipič, B. (2014a). Comparing random forest and gaussian process modeling in the GP-DEMO algorithm. In *Proceedings of the 17th international multiconference information society, IS 2014* (Vol. A, pp. 66–69). Jožef Stefan Institute, Ljubljana.

Mlakar, M., Tušar, T., & Filipič, B. (2014b). Comparing solutions under uncertainty in multiobjective optimization. *Mathematical Problems in Engineering*, *2014*. doi:10.1155/2014/817964

Mockus, J., Tiesis, V., & Zilinskas, A. (1978). The application of bayesian methods for seeking the extremum. In *Towards global optimization 2* (pp. 117–129). North-Holland, Amsterdam.

Myers, R. H. & Montgomery, D. C. (1995). *Response surface methodology: Process and product optimization using designed experiments*. Wiley, New York.

Neal, R. M. (1996). Bayesian learning for neural networks. In *Lecture notes in statistics 118*. Springer, Berlin.

O'Hagan, A. (1978). Curve fitting and optimal design for prediction (with discussion). *Journal of the Royal Statistical Society Series B*, *40*(1), 1–42.

Oumayma, B., Nahla, B. A., & Talbi, E.-G. (2013). A possibilistic framework for solving multi-objective problems under uncertainty: Definition of new Pareto optimality. In *IEEE 27th international symposium on parallel and distributed processing workshops and PhD forum* (pp. 405–414). IEEE Computer Society.

Pilat, M. & Neruda, R. (2011). ASM-MOMA: Multiobjective memetic algorithm with aggregate surrogate model. In *2011 IEEE congress on evolutionary computation (CEC)* (pp. 1202–1208). IEEE, Piscataway.

Pilat, M. & Neruda, R. (2012). An evolutionary strategy for surrogate-based multiobjective optimization. In *2012 IEEE congress on evolutionary computation (CEC)* (pp. 1–7). IEEE, Piscataway.

Poloni, C., Giurgevich, A., Onesti, L., & Pediroda, V. (2000). Hybridization of a multi-objective genetic algorithm, a neural network and a classical optimizer for a complex design problem in fluid dynamics. *Computer Methods in Applied Mechanics and Engineering*, *186*(2), 403–420.

Price, K. & Storn, R. M. (1997). Differential evolution – A simple evolution strategy for fast optimization. *Dr. Dobb's Journal*, *22*(4), 18–24.

Price, K., Storn, R. M., & Lampinen, J. A. (2006). *Differential evolution: A practical approach to global optimization*. Springer.

Quinonero-Candela, J. & Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *The Journal of Machine Learning Research*, *6*, 1939–1959.

Quinonero-Candela, J., Rasmussen, C. E., & Williams, C. K. I. (2007). *Approximation methods for Gaussian process regression* (tech. rep. No. MSR-TR-2007-124). Microsoft Research.

Rasmussen, C. E. & Williams, C. K. I. (2006). *Gaussian processes for machine learning*. MIT Press, Cambridge.

Robič, T. & Filipič, B. (2004). In search for an efficient parameter tuning method for steel casting. In *Proceedings of the international conference on bioinspired optimization methods and their applications – BIOMA 2004* (pp. 83–94). Jožef Stefan Institute, Ljubljana.

Robič, T. & Filipič, B. (2005). DEMO: Differential evolution for multiobjective optimization. In *2005 evolutionary multi-criterion optimization (EMO)* (Vol. 3410, pp. 520–533). Lecture Notes in Computer Science. Springer, Berlin.

Rodgers, J. L. & Nicewander, W. A. (1988). Thirteen ways to look at the correlation coefficient. *The American Statistician*, *42*(1), 59–66.

Sacks, J., Welch, W. J., Mitchell, T. J., & Wynn, H. P. (1989). Design and analysis of computer experiments. *Statistical Science*, *4*(4), 409–423.

Šarler, B., Belič, I., Filipič, B., Grešovnik, I., Hanoglu, U., Jenko, M., ... Vertnik, R. (2011). Simulation and optimisation of casting, rolling and heat treatment processes for competitive production of topmost steel. In *Proceedings of the 19th conference on materials and technology* (p. 25). Inštitut za kovinske materiale in tehnologije, Ljubljana.

Snelson, E. & Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In *Advances in neural information processing systems 18* (pp. 1257–1264). MIT Press, Cambridge.

Soares, G. L., Guimaraes, F. G., Maia, C. A., Vasconcelos, J. A., & Jaulin, L. (2009). Interval robust multi-objective evolutionary algorithm. In *2009 IEEE congress on evolutionary computation (CEC)* (pp. 1637–1643).

Specht, D. F. (1990). Probabilistic neural networks. *Neural Networks*, *3*(1), 109–118.

Stein, M. L. (1999). *Interpolation of spatial data: Some theory for kriging.* Springer Series in Statistics. Springer, New York.

Taboada, H. A., Baheranwala, F., Coit, D. W., & Wattanapongsakorn, N. (2007). Practical solutions for multi-objective optimization: An application to system reliability design problems. *Reliability Engineering & System Safety*, *92*(3), 314–322.

Teich, J. (2001). Pareto-front exploration with uncertain objectives. In *2001 evolutionary multi-criterion optimization (EMO)* (Vol. 1993, pp. 314–328). Lecture Notes in Computer Science. Springer.

Trautmann, H., Mehnen, J., & Naujoks, B. (2009). Pareto-dominance in noisy environments. In *2009 IEEE congress on evolutionary computation (CEC)* (pp. 3119–3126).

Tušar, T. (2007). *Design of an algorithm for multiobjective optimization with differential evolution* (M.Sc. Thesis, Faculty of Computer and Information Science, University of Ljubljana).

Tušar, T. & Filipič, B. (2007). Differential evolution versus genetic algorithms in multiobjective optimization. In *2007 evolutionary multi-criterion optimization (EMO)* (Vol. 4403, pp. 257–271). Lecture Notes in Computer Science. Springer.

Ulmer, H., Streichert, F., & Zell, A. (2003). Evolution strategies assisted by Gaussian processes with improved preselection criterion. In *2003 IEEE congress on evolutionary computation (CEC)* (Vol. 1, pp. 692–699).

Vertnik, R. & Šarler, B. (2009). Simulation of continuous casting of steel by a meshless technique. *International Journal of Cast Metals Research*, *22*(1–4), 311–313.

Viana, F. A., Haftka, R. T., & Watson, L. T. (2012). Efficient global optimization algorithm assisted by multiple surrogate techniques. *Journal of Global Optimization*, 1–21.

Voß, T., Trautmann, H., & Igel, C. (2010). New uncertainty handling strategies in multi-objective evolutionary optimization. In *2010 parallel problem solving from nature – PPSN XI* (Vol. 6239, pp. 260–269). Lecture Notes in Computer Science. Springer, Berlin.

Wagner, T., Emmerich, M., Deutz, A., & Ponweiser, W. (2010). On expected-improvement criteria for model-based multi-objective optimization. In *Parallel problem solving from nature – PPSN XI* (Vol. 6238, pp. 718–727). Lecture Notes in Computer Science. Springer, Berlin.

Wiener, N. (1949). *Extrapolation, interpolation and smoothing of stationary time series.* MIT Press, Cambridge.

Williams, C. K. I. & Rasmussen, C. E. (1996). Gaussian processes for regression. In *Advances in neural information processing systems 8* (pp. 514–520). MIT Press, Cambridge.

Zhang, J. & Sanderson, A. C. (2007). DE-AEC: A differential evolution algorithm based on adaptive evolution control. In *2007 IEEE congress on evolutionary computation (CEC)* (pp. 3824–3830). IEEE, Piscataway.

Zhou, A. & Zhang, Q. (2010). A surrogate-assisted evolutionary algorithm for minimax optimization. In *2010 IEEE congress on evolutionary computation (CEC)* (pp. 1–7). IEEE, Piscataway.

Zhou, Z., Ong, Y. S., Nair, P. B., Keane, A. J., & Lum, K. Y. (2007). Combining global and local surrogate models to accelerate evolutionary optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications*, *37*(1), 66–76.

Zilinskas, A. (1980). On the use of statistical models of multimodal functions for the construction of the optimization algorithms. In *Optimization techniques* (Vol. 23, pp. 138–147). Lecture Notes in Control and Information Sciences. Springer, Berlin.

Zimmerman, D. W. (1997). Teacher's corner: A note on interpretation of the paired-samples t Test. *Journal of Educational and Behavioral Statistics*, *22*(3), 349–360.

Zitzler, E. & Thiele, L. (1998). Multiobjective optimization using evolutionary algorithms – A comparative case study. In *Parallel problem solving from nature – PPSN V* (pp. 292–301). Springer.

Zitzler, E. & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, *3*(4), 257–271.

Bibliography

Publications Related to the Thesis

Journal Articles

Mlakar, M., Petelin, D., Tušar, T., & Filipič, B. (2015). GP-DEMO: Differential evolution for multiobjective optimization based on Gaussian process models. *European Journal of Operational Research*, *243*(2), 347–361.

Mlakar, M., Tušar, T., & Filipič, B. (2014b). Comparing solutions under uncertainty in multiobjective optimization. *Mathematical Problems in Engineering*, *2014*. doi:10.1155/2014/817964

Conference Papers

Mlakar, M., Petelin, D., Tušar, T., & Filipič, B. (2013a). GP-DEMO: Differential evolution for multiobjective optimization based on Gaussian process models. In *Proceedings of the 22nd international conference on multiple criteria decision making (MCDM)* (p. 203). Malaga, Spain.

Mlakar, M., Tušar, T., & Filipič, B. (2012). Discrete vs. continuous multiobjective optimization of continuous casting of steel. In *Proceedings of the genetic and evolutionary computation conference, GECCO '12* (pp. 587–590). ACM, NY.

Mlakar, M., Tušar, T., & Filipič, B. (2013b). Primerjava rešitev ob negotovosti v večkriterijski optimizaciji. In *Proceedings of the 16th international multiconference information society, IS 2013* (Vol. A, pp. 86–89). Jožef Stefan Institute, Ljubljana.

Mlakar, M., Tušar, T., & Filipič, B. (2014a). Comparing random forest and gaussian process modeling in the GP-DEMO algorithm. In *Proceedings of the 17th international multiconference information society, IS 2014* (Vol. A, pp. 66–69). Jožef Stefan Institute, Ljubljana.

Šarler, B., Belič, I., Filipič, B., Grešovnik, I., Hanoglu, U., Jenko, M., ... Vertnik, R. (2011). Simulation and optimisation of casting, rolling and heat treatment processes for competitive production of topmost steel. In *Proceedings of the 19th conference on materials and technology* (p. 25). Inštitut za kovinske materiale in tehnologije, Ljubljana.

Other Publications

Conference Papers

Filipič, B., Mlakar, M., Dovgan, E., & Tušar, T. (2013). Razvoj sistema za računalniško podprto evidentiranje in sestavljanje fragmentov stenskih poslikav. In *Proceedings of the 14th international multiconference information society, IS 2011* (Vol. A, pp. 45–48). Jožef Stefan Institute, Ljubljana.

Mlakar, M. & Filipič, B. (2011). Implementacija programa za evidentiranje in pomoč pri sestavljanju fragmentov stenskih poslikav. In *3rd Jožef Stefan international postgraduate school students conference* (pp. 104–109). Jožef Stefan International Postgraduate School, Ljubljana.

Mlakar, M., Koblar, V., & Filipič, B. (2010). Optimizacija klasifikacijske točnosti in velikosti odločitvenih dreves za napovedovanje kakovosti grafitnih polizdelkov. In *Proceedings of the 19th international electrotechnical and computer science conference, ERK 2010* (pp. 112–119). IEEE, Ljubljana.

Biography

Miha Mlakar was born in Ljubljana, Slovenia. In elementary school he obtained the gold Zois award in mathematics and received the Zois scholarship for gifted students for the duration of secondary and undergraduate education. After elementary school, he attended the Poljane grammar school in Ljubljana, where he obtained several awards in national competitions in mathematics and physics.

He graduated in 2010 from the Faculty of Computer and Information Science, University of Ljubljana, Slovenia, with the thesis entitled *Accuracy of cancer diagnosis models inferred by machine learning from gene expression data sets.*

Since 2010 he has been working as a research assistant at the Department of Intelligent Systems of the Jožef Stefan Institute in Ljubljana. His research interests focus on evolutionary algorithms for single- and multiobjective optimization, machine learning, data mining, and applications of these techniques in optimization of production processes and engineering design.

In 2010 he enrolled in the Jožef Stefan International Postgraduate School, where he is currently pursuing a Ph.D. in Information and Communication Technologies. In accordance with his dissertation topic his research interests now also include surrogate-model-based optimization and optimization under uncertainty.