Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Campus Zacatenco

Computer Science Department

Evolutionary and Memetic Strategies for the Treatment of Multi-objective and Parameter Dependent Multi-objective Optimization Problems.

A dissertation presented by

Víctor Adrián Sosa Hernández

as the fulfillment of the requirement for the degree of

Ph. D. in Computer Science

Advisor:

Dr. Oliver Steffen Schütze

Mexico, Mexico City. **December 2017**

Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional

Unidad Zacatenco

Departamento de Computación

Estrategias Meméticas y Evolutivas para el Tratamiento de Problemas de Optimización Multi-objetivo y Dependientes de Parámetros Multi-objetivo.

Tesis que presenta

Víctor Adrián Sosa Hernández

Para obtener el Grado de

Doctor en Ciencias en Computación

Advisor:

Dr. Oliver Steffen Schütze

México, Ciudad de México. **Diciembre 2017**

Resumen

Hoy en día, en muchas aplicaciones del mundo real relacionadas con la industria, finanzas y otras ramas se requiere el optimizar múltiples objetivos al mismo tiempo. Los problemas de optimización multi-objetivo(POM) representan tales problemas donde múltiples objetivos deben optimizarse de forma concurrente. Además, existen ciertos tipos de problemas relacionados con aplicaciones del mundo real donde además de optimizar múltiples objetivos se requiere considerar el efecto de parámetros externos dentro del proceso de optimización. Estos parámetros externos pueden representar por ejemplo las condiciones del ambiente como el clima, temperatura o viento. Para este caso en particular la condiciones ambientales no pueden ser optimizadas ni tampoco ignoradas dado que pueden afectar a los objetivos si un cambio se llega a presentar. Un problema de optimización multi-objetivo dependiente de parámetros (POMP) consiste en un POM donde adicionalmente algunos parámetros externos $\lambda \in \mathbb{R}^l$ no pueden ser incluidos dentro del diseño de un objeto (por ejemplo, el viento no puede ser incluido dentro del diseño de un carro óptimo). Durante la última década los algoritmos evolutivos multi-objetivo (AEMOs) se volvieron populares para el tratamiento de POMs. Entre los diversos AEMOs, hay una tendencia reciente dentro del diseño de estos algoritmos de incluir indicadores de desempeño dentro del mecanismo de selección o como estimadores de densidad. Estos indicadores de desempeño son herramientas que miden la calidad de un aproximación producida por un AEMO, por lo tanto, al incluir los indicadores de desempeño dentro del proceso de optimización se mejora la calidad de la aproximación generada de acuerdo al indicador seleccionado. Varias ventajas de estos algoritmos incentivan su uso para el tratamiento de POMs, sin embargo, también existen ciertas desventajas al utilizarlos. Por ejemplo, generalmente los AEMOs tienden a converger lentamente, lo que es una desventaja severa para este tipo de algoritmos. Este problema provoca el uso de un alto número de evaluaciones de función del POM para obtener una representación adecuada del conjunto de interés. Como posible solución a este problema, se propuso el uso de estrategias meméticas donde un AEMO es combinado con una técnica de búsqueda local para mejorar su desempeño. Finalmente, como se menciono anteriormente, existen una gran variedad de AEMOs para el tratamiento de POMs, sin embargo, para el caso de los POMPs existe una escasez de técnicas especializadas diseñadas para obtener un conjunto de soluciones del problema completo.

A lo largo de este trabajo de tesis, nos enfocaremos en el tratamiento numérico de POMs y POMPs. Primeramente, presentaremos dos técnicas de búsqueda local el método de búsqueda dirigida basado en hypervolumen (MBDH) y el método de Newton basado en hypervolumen (MNH). Ambas técnicas incorporán el uso del indicador del indicador de desempeño hypervolumen para mejorar la convergencia y la calidad de las aproximaciones generadas bajo este indicador. Una característica importante de ambas técnicas es que fueron diseñadas para conjuntos y no solamente para un mejorar un solo punto. Dicha característica convierte ambas técnicas en candidatos perfectos para ser combinados junto con un AEMO. Posteriormente, presentaremos dos estrategias meméticas las cuales integran tanto al MBDH como al MNH. Las estrategias meméticas desarrolladas muestran alcanzar de forma más rápida el conjunto solución y de igual formar mejorar el valor del indicador hypervolume de la aproximación final. Lo anterior incluso es realizado utilizando un número menor de evaluaciones de función. Por último, para contribuir a reducir la falta de técnicas para el tratamiento de POMPs, presentaremos un algoritmo evolutivo basado en descomposición. Esta técnica incluirá observaciones hechas en un estudio presentado dentro de esta tesis sobre el comportamiento de la búsqueda local estocástica dentro de los POMPs. El algoritmo demuestra ser capaz de obtener una aproximación del conjunto de solución completo explotando la interacción existente entre los elementos de la población actual inclusive si ellos pertenecen a un valor de λ diferente.

Abstract

Nowadays, many real world applications related to industry, medicine, finance, among others require optimizing multiple objectives at the same time. A multi-objective optimization problem (MOP) represents such a problem of optimizing not only one but multiple objectives concurrently. Even more, there is a certain type of problems related also to real world applications in which furthermore of optimizing multiple objectives one has to consider the effect of external parameters on the optimization process. These external parameters can represent for instance environment conditions such as weather, temperature or side wind. In this case, environment conditions cannot be optimized nor neglected since they can affect the objectives if a change is presented. A parameter dependent multi-objective optimization problem (PMOP) consists of a MOP where in addition several external parameters $\lambda \in \mathbb{R}^l$ cannot be influenced for the design of an object (e.g., the side wind in the design of an 'optimal' car). During the last decade, multi-objective evolutionary algorithms (MOEAs) has become very popular for the treatment of MOPs. Among them, there is a recent trend in the design of algorithms which consist in including performance indicators into the selection mechanism or as a density estimator. These performance indicators are tools to measure the quality of the approximation set produced by a MOEA, then by including them into the optimization process one, can improve the quality of our approximation according to the selected indicator. Many advantages of these algorithms encourage their use for the treatment of MOPs, however, they still have drawbacks. For instance in general, MOEAs tend to converge slowly, what is a severe drawback of this kind of algorithms. This drawback leads to use a relatively high number of function evaluations to obtain a suitable representation of the set of interest. As a possible remedy, researchers have proposed the use of memetic strategies where a MOEA is hybridized with a local search technique in order to improve its performance. Finally, as we mentioned before, there is a big variety of MOEAs which address the treatment of MOPs, nevertheless, in the case of PMOPs, there is still a lack of specialized techniques designed for solving them as a whole.

Throughout this thesis work, we aim for the numerical treatment of both multi-objective and parameter dependent multi-objective optimization problems. First, we present two different local search techniques the hypervolume based directed search (HVDS) and the hypervolume Newton method (HNM). Both techniques incorporate the use of the hypervolume performance indicator to improve the convergence and also the quality of the approximation according to this indicator. A remarkable property of both techniques is that they are designed for sets and not exclusively for a single point which makes them the perfect candidates to be coupled with a MOEA. Next, two memetic strategies are presented which integrate HVDS and HNM, respectively. The presented memetic strategies show to reach faster the solution set of a MOP and also to improve the hypervolume indicator value of the produced approximation, even using fewer function evaluations. Finally, in order to fill the gap of evolutionary algorithms to tackle PMOPs as a whole, we present the PMOEA/D. This technique incorporates observations made over stochastic local search within PMOPs presented in this thesis work. The algorithm shows to be able to compute an approximation of the whole solution set by exploiting the interactions between all the elements in the current population even if they belong to a different λ value.

Agradecimientos

Primero que nada quiero agradecerle a Dios que me ha permitido hasta el día de hoy cumplir sueños y metas.

Doy las gracias al CONACyT por la beca que me brindó para realizar mis estudios de doctorado.

Agradezco al CINVESTAV-IPN y en especial al Departamento de Computación por aceptarme en su programa de doctorado. Le doy gracias a mis profesores por compartir sus conocimientos conmigo, al personal administrativo por realizar siempre un excelente trabajo y por ayudarme en cada trámite que tuve que hacer, también agradezco a los auxiliares de investigación por su ayuda y apoyo dentro del departamento. De igual forma, a mis amigas y amigos, y compañeros dentro del departamento les doy gracias por su compañía, consejo y buenos momentos.

Quiero darle un especial agradecimiento al Dr. Oliver Schütze, por la oportunidad que me brindó para realizar mi preparación desde la maestría y así poder seguir con el doctorado. Agradezco su guía y apoyo durante la realización de esta etapa académica. Gracias por su tiempo y dedicación para revisar todo el trabajo dentro de esta tesis.

Agradezco a la Dra. Heike Trautmann (Universidad de Müster), al Dr. Günter Rudolph (TU-Dortmund), y al Dr. Michael Emmerich (LIACS) por darme la oportunidad de colaborar con ellos en múltiples artículos. Gracias por su valiosa ayuda y consejo en la realización de este trabajo.

Quiero darle las gracias a los doctores Luis Gerardo de la Fraga, Amilcar Meneses Viveros, Saúl Zapotecas Martínez, y Carlos Artemio Coello Coello por formar parte del comité evaluador de mi trabajo de tesis. Sus comentarios ayudaron a mejorar este trabajo escrito.

A mis padres Isabel y Víctor por su amor, apoyo, y consejo que me han dado durante estos 28 años. Nunca terminaré de agradecerles y pagarles todo lo que han hecho por mí, por esa razón tengan por seguro que seguiré creciendo como persona, y me seguiré superando día tras día para seguir llenándolos de orgullo y satisfacciones.

Le doy las gracias a mis hermanos Andrés y Valeria, por su apoyo, ya que ellos también son una razón para yo seguir adelante. No dejen de luchar y superarse nunca.

A toda mi familia, abuelos, tíos y primos les agradezco ya que durante toda mi vida me han acompañado y enseñado muchas cosas importantes.

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

In many real world applications, several conflicting objectives have to be considered concurrently leading to a multi-objective optimization problem (MOP). For instance, consider the performance of a vehicle over a test road with respect to the speed and the energy consumption. Both objectives have to be optimized, however, there exists a clear conflict between them which negatively affects one if the other is optimized. This leads to the fact that there is typically an entire set of different optimal solutions. The solution set $P_{\mathbf{x}}$ of a MOP, the Pareto set, typically forms a $(k-1)-$dimensional object, where k is the number of objective functions involved in the problem. The image of the Pareto set $P_{\mathbf{f}}$ is called the Pareto front. In certain cases, external parameters modify the solution set of a MOP. For instance, the performance of the vehicle can be affected by weather conditions (e.g., side wind, rain) or road conditions (e.g., sliding friction). Such external parameters cannot be optimized neither neglected but, they have to be considered when solving the problems. The parameter dependent multi-objective optimization problems (PMOPs) consider this extended case when external parameters come into play. The solution set $P_{\mathbf{x}_\Lambda}$ of a PMOP forms a $(k-1+l)-$dimensional object, where l is the dimension of Λ and k the number of objective functions.

In order to measure the quality of an approximation set for both kind of problems (MOPs and PMOPs), one can use certain performance indicators [2–9]. Such performance indicators can transform a MOP into a scalar optimization problem (SOP), which can be advantageous at least when one solves the problem. The hypervolume indicator is one the most widely used indicator in the field of multi-objective optimization. One of its remarkable features is that it is Pareto compliant, however, certain drawbacks such as its high complexity when it is computed in higher dimension with respect to the objective functions caused that many researchers look for alternatives.

There are more other performance indicators such as e.g. the averaged Hausdorff distance Δ_p, $R2$, IGD_+, among others.

For the treatment of MOPs especially the use of multi-objective evolutionary algorithms (MOEAs) have caught the interest of many researchers. Among MOEAs, there is a recent trend in the design of algorithms that are based on a particular performance indicator, the so-called indicator-based evolutionary algorithms. It is known that MOEAs in general tend to converge slowly, what is a severe drawback of this kind of algorithms. This drawback leads to use a relatively high number of function evaluations to obtain a suitable representation of the set of interest. As a possible remedy, researchers have proposed memetic strategies. These memetic algorithms consist in hibridizing MOEAs with local search strategies based on mathematical techniques in order to improve the algorithm performance or even the quality of the final approximation.

The aim of this thesis project is to develop algorithms which push the state-of-the-art for the numerical treatment of MOPs and PMOPs. In particular, we aim to develop memetic strategies based on the hypervolume performance indicator to tackle MOPs. Then, a new evolutionary algorithm to tackle PMOPs based on the stochastic local search observations will be presented in order to produce an approximation of the whole family of Pareto sets and fronts for a given PMOP.

1.1 Motivation

Usually, the solution set of both types of problems MOPs and PMOPs cannot be obtained analytically. Then, the necessity of efficient methods to compute a suitable finite size approximation of the solution set arises. In the case of MOPs faster and reliable procedures are desired to generate an approximation of the solution set, while for PMOPs it is expected to obtain algorithms that are able to compute a discretization of the entire family of Pareto sets and fronts in one single run. As a remedy, we intend to hybridize evolutionary algorithms based on indicators using novel local search strategies. In the case of the treatment of PMOPs the lack of methods to generate a complete approximation of the solutions set gives us a clear opportunity to develop new evolutionary strategies able to perform this task. Finally, in most of the cases, performance indicators are used to assess the produced approximation by algorithms, even some algorithms include them in their selection operators. Therefore, we have to consider the given performance indicator inside the search processes when new local search techniques, evolutionary approaches, and memetic strategies are designed.

1.2 The Problem

The design of evolutionary and memetic algorithms for improving performance indicators values which push the state-of-the-art for the treatment of MOPs and PMOPs.

1.3 General and Particular Aims

General Aim

To design novel evolutionary and memetic algorithms for the numerical treatment of multi-objective and parameter dependent multi-objective optimization problems by means of techniques based on performance indicators.

Particular Aims

1. To develop a memetic strategy for MOPs where the local search technique is based on the directed search method.

2. To consider the hypervolume gradient inside a local searcher within memetic algorithms and as standalone set based local searcher.

3. To investigate the convergence properties of the methods based on the hypervolume gradient.

4. To develop memetic strategies for MOPs based on the hypervolume indicator.

5. To develop local search techniques that consider mechanism for constraint handling.

6. To investigate the behavior of stochastic local search over PMOPs.

7. To design an evolutionary technique for the non-sequential treatment of a given PMOP.

1.4 Final Contributions

Algorithms

- Local search algorithms for the treatment of MOPs:
  - The Hypervolume based Directed Search for general MOPs (HVDS).
  - The Hypervolume Newton Method for bi-objective optimization problems (HNM).
  - The Hypervolume Newton Method for inequality constrained bi-objective optimization problems.
  - The Hypervolume Newton Method for equality constrained bi-objective optimization problems.
- Memetic strategies for the treatment of MOPs:
  - The memetic version of the SMS-EMOA using HVDS for $k = 2, 3$.
  - The memetic version of the SMS-EMOA using HNM for $k = 2$.
- Stochastic techniques for the treatment of PMOPs:
  - Simple Neighborhood Search for PMOPs.
  - PMOEA/D for PMOPs.

Conference Papers

- Víctor Adrián Sosa Hernández, Oliver Schütze, Michael Emmerich: Hypervolume Maximization via Set Based Newton´s Method. In EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V Advances in Intelligent Systems and Computing Volume 288, Springer, Beijing, China, 2014, pp 15-28.

- Víctor Adrián Sosa Hernández, Oliver Schütze, Heike Trautmann, Günter Rudolph: On the Behavior of Stochastic Local Search Within Parameter Dependent MOPs. In the Evolutionary Multi-Criterion Optimization: 8th International Conference (EMO), Springer, Guimares, Portugal, 2015, pp 126-140.

Book Chapters

- Víctor Adrián Sosa Hernández, Adriana Lara, Heike Trautmann, Günter Rudolph, and Oliver Schütze: The Directed Search Method for Unconstrained Parameter Dependent Multi-objective Optimization Problems. In O. Schütze et al. (eds.), Numerical and Evolutionary Optimization - NEO 2015, Springer, 2016, pp 281-330.

Journal Papers (JCR)

- Oliver Schütze, Víctor Adrián Sosa Hernández, Günter Rudolph, and Heike Trautmann: The Hypervolume based Directed Search Method for Multi-Objective Optimization Problems. Journal of Heuristics, June 2016, Volume 22, Issue 3, pp 273-300.

- Víctor Adrián Sosa Hernández, Oliver Schütze, Hao Wang, André Deutz, and Michael Emmerich: The Set-Based Hypervolume Newton Method for Bi-Objective Optimization. Submitted to the IEEE Transactions on Cybernetics, August 2017.

Collaboration Papers

- Sergio Alvarado, Adriana Lara, Víctor Adrián Sosa Hernández and Oliver Schütze: An effective mutation operator to deal with multi-objective constrained problems: SPM. In the 13th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), Mexico City, Mexico, 2016, pp. 1-6.

- Saúl Zapotecas Martínez, Víctor Adrián Sosa Hernández, Hernán Aguirre, Kiyoshi Tanaka, and Carlos Artemio Coello Coello: Using a Family of Curves to Approximate the Pareto Front of a Multi-Objective Optimization Problem. In the International Conference on Parallel Problem Solving from Nature (PPSN), Springer, Ljubljana, Slovenia, 2014, pp 682-691.

- Adriana Lara, Sergio García, Lourdes Uribe, Víctor Adrián Sosa Hernández, H. Wang, and Oliver Schütze: On the Choice of Neighborhood Sampling to Build Effective Search Operators for Constrained MOPs. Submitted to the Memetic Computing Journal.

1.5 Organization of this work

The remainder of this thesis work is organized as follows: in Chapter 2, we state some theoretical background to understand the presented work. Furthermore, we review some mathematical programming and stochastic techniques for the treatment of optimization problems. Also, we present the related work to local search strategies. Chapter 3 is used to present a new version of the HVDS for general number objectives. We design a new region division for MOPs with several objectives, then the HVDS is coupled with an evolutionary strategy to create a novel memetic algorithm. Numerical results show the advantages of integrating our local search strategy. In Chapter 4, we firstly investigate the behavior related to the hypervolume gradient flow in several benchmark functions. Also, we present the hypervolume Newton method (HNM) as a local search strategy for unconstrained and constrained (equality and inequality constrained) problems. We present the integration of the HNM into an evolutionary approach to create a new memetic algorithm for refine final approximations. Numerical results will show the performance of our method against other MOEAs. An investigation of the effect of stochastic neighborhood search (SLS) in the context of PMOPs is presented in Chapter 5. Then, we present an evolutionary technique based on decomposition to compute an approximation of the whole family of solution sets of a given PMOP. Results show the advantage of integrating the obtained knowledge by the investigation of SLS into the evolutionary technique. Finally, we draw some conclusions and give some future paths for continuing our research.

2 | Background

This chapter presents the basic concepts and definitions related to this thesis project. The aim is to ensure that the reader has a better understanding of the content of this work. First, we start by stating the theoretical background, and then we discuss the related state-of-the-art.

2.1 Theoretical Background

2.1.1 Single-objective Optimization

A general single-objective optimization problem (SOP) is defined as the process to minimize (or maximize) an objective function f according a decision vector $\mathbf{x}$. The problem is formulated as follows:

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) \tag{2.1}$$

$$\begin{aligned} \text{s.t}\quad g_i(\mathbf{x}) \leq& 0 \quad i = 1, \ldots, p \\ h_j(\mathbf{x}) =& 0 \quad j = 1, \ldots, q, \end{aligned}$$

where $f : S \subseteq \mathbb{R}^n \to \mathbb{R}$ is the objective function, $\mathbf{x} = (x_1, ..., x_2)^T \in \mathbb{R}^n$ is a vector of decision variables and n defines the dimension of the decision space. The functions g_i and h_i represent the inequality and equality constraints, respectively. The feasible region S is implicitly given by the constraints:

$$S := \{\mathbf{x} \mid g_i(\mathbf{x}) \leq 0 \ i = 1, ..., p \text{ and } h_j(\mathbf{x}) = 0 \ j = 1, ..., q\}. \tag{2.2}$$

If the function f is continuous and twice differentiable, its gradient at $\mathbf{x}$ can be defined as:

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f(\mathbf{x})}{\partial x_1}, \ldots, \frac{\partial f(\mathbf{x})}{\partial x_n} \right)^T \in \mathbb{R}^n, \tag{2.3}$$

where $\frac{\partial f(\mathbf{x})}{\partial x_i}$ denotes the partial derivative of the function f with respect to the variable x_i. The Hessian matrix can be expressed as follows:

$$\nabla^2 f(\mathbf{x}) = \begin{pmatrix} \frac{\partial}{\partial x_1}\left(\frac{\partial f(\mathbf{x})}{\partial x_1}\right) & \cdots & \frac{\partial}{\partial x_1}\left(\frac{\partial f(\mathbf{x})}{\partial x_n}\right) \\ \vdots & \ddots & \vdots \\ \frac{\partial}{\partial x_n}\left(\frac{\partial f(\mathbf{x})}{\partial x_1}\right) & \cdots & \frac{\partial}{\partial x_n}\left(\frac{\partial f(\mathbf{x})}{\partial x_n}\right) \end{pmatrix} \in \mathbb{R}^{n \times n}, \tag{2.4}$$

where $\frac{\partial}{\partial x_i}\left(\frac{\partial f(\mathbf{x})}{\partial x_j}\right)$ represents the second partial derivative of the function f with respect to the variables x_i and x_j.

In the following, we introduce some definitions related to the solution when solving SOPs. A global minimizer is defined as follows:

Definition 1. *A point* $\mathbf{x}^* \in S$ *is a global minimizer of the problem stated in Equation* (2.1) *if*

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \text{ for all } \mathbf{x} \in S. \tag{2.5}$$

In most of the cases, to find a global minimizer is a difficult task since we cannot explore the whole decision space. In general, the mathematical methods which tackle SOPs tend to find local solutions, however, for certain cases these solutions can satisfy the needs of a decision maker (even if they are not global). The definition of a local minimizer is as follows:

Definition 2. *A point* $\mathbf{x}^* \in S$ *is a local minimizer of the problem stated in Equation* (2.1) *if for a certain neighborhood* $\mathcal{N}(\mathbf{x}^*)$*,* $f(\mathbf{x}^*) \leq f(\mathbf{x})$ *for all* $\mathbf{x} \in \mathcal{N}(\mathbf{x}^*)$*.*

The neighborhood $\mathcal{N}(\mathbf{x}^*)$ can be represented by an open ball $B(\mathbf{x}^*, \gamma)$ with center $\mathbf{x}^*$ and radius $\gamma > 0$,

$$\mathcal{N}(\mathbf{x}^*) = B(\mathbf{x}^*, \gamma) := \{\mathbf{x} \in \mathbb{R}^n | \ ||\mathbf{x}^* - \mathbf{x}|| < \gamma\}. \tag{2.6}$$

Figure 2.1 depicts a graphical example of a SOP, the figure contains both a global minimizer (circle) and a local minimizer (triangle) that is not globally optimal.

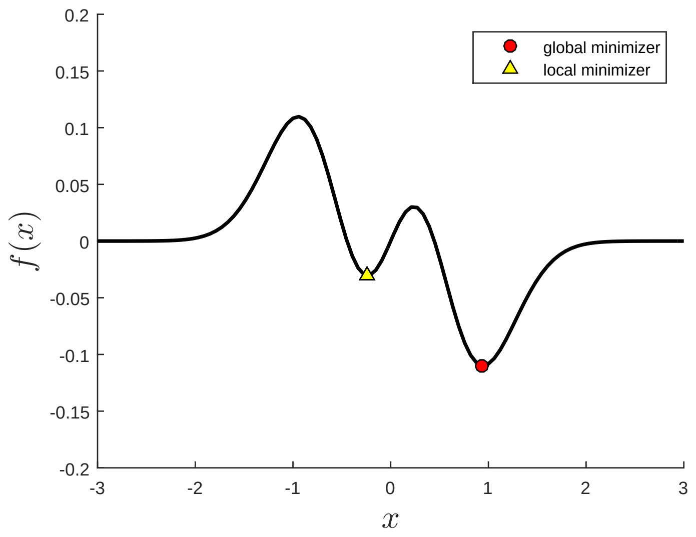


Figure 2.1: Example of local and global minimizers of a SOP.

In order to identify if $\mathbf{x}^*$ is a local minimizer, it is needed to examine all the points inside the neighborhood $\mathcal{N}(\mathbf{x}^*)$ which seems to be a difficult task. However, under certain conditions there is a more efficent and practical process. When f is continuous and twice differentiable, we can determine if $\mathbf{x}^*$ is a local minimizer or not by examining its gradient and Hessian matrix. The optimality conditions developed by Karush, Kuhn, and Tucker [10, 11] allow us to state the necessary condition for optimality.

Theorem 1 (KKT necessary condition)**.** *Let* $\mathbf{x}^*$ *be a local minimizer of a SOP as stated in Equation* (2.1)*; then there exist constants* $\kappa_1, \ldots, \kappa_p$ *and* $\varrho_1, \ldots, \varrho_q$ *that satisfy:*

$$\begin{aligned} \nabla f(\mathbf{x}^*) - \sum_{i=1}^{p} \kappa_i \nabla g_i(\mathbf{x}^*) - \sum_{j=1}^{q} \varrho_j \nabla h_j(\mathbf{x}^*) = 0 \\ g_i(\mathbf{x}) \leq 0 \quad i = 1, \ldots, p \\ h_j(\mathbf{x}) = 0 \quad j = 1, \ldots, q \\ \kappa_i g_i(\mathbf{x}) = 0 \quad i = 1, \ldots, p \\ \kappa_i \geq 0 \quad i = 1, \ldots, p. \end{aligned} \tag{2.7}$$

The scalar quantities κ_i and ϱ_j stated in Equation (2.7) are called the Lagrange multipliers for the constraints [12]. In the case that $p = 0$ and $q = 0$ the problem stated in Equation (2.1) is an unconstrained SOP of the following form:

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) \tag{2.8}$$

For that case, the necessary condition is reduced to $\nabla f(\mathbf{x}^*) = \mathbf{0}$.

2.1.2 Multi-Objective Optimization

In many problems it is required that more than one objective function has to be minimized (or maximized) concurrently, leading to a multi-objective optimization problem (MOP). Formally, a continuous MOP can be defined as follows:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \mathbf{f}(\mathbf{x}), \tag{2.9}$$

$$\begin{aligned} \text{s.t}\quad g_i(\mathbf{x}) \leq& 0 \quad i = 1, \ldots, p \\ h_j(\mathbf{x}) =& 0 \quad j = 1, \ldots, q, \end{aligned}$$

where $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))^T$ is a vector of k objective functions, $f_i : S \subseteq \mathbb{R}^n \to \mathbb{R},\ i = 1, \ldots, k$, is the i-th objective function, $\mathbf{x} \in \mathbb{R}^n$ is a vector of decision variables and n defines the dimension of the decision space. The inequality and equality constraints are given by the functions g_i and h_i, respectively. S represents the feasible region as defined above in Equation (2.2). Throughout this work, it is assumed that each objective function is continuous and twice differentiable almost everywhere in the decision space.

Pareto Optimality

In the following, the concept of optimality is defined for MOPs based on the work of Pareto [13] and Edgeworth [14].

Definition 3. *(a) Let* $\mathbf{v}$, $\mathbf{w} \in \mathbb{R}^k$*. Then the vector* $\mathbf{v}$ *is less than* $\mathbf{w}$ ($\mathbf{v} <_p \mathbf{w}$)*, if* $v_i < w_i$ *for all* $i \in \{1, 2, \ldots, k\}$*. The relation* $\leq_p$ *is defined analogously.*

(b) *A vector* $\mathbf{y} \in \mathbb{R}^n$ *is dominated by a vector* $\mathbf{x} \in \mathbb{R}^n$ $(\mathbf{x} \prec \mathbf{y})$ *with respect to the definition of a MOP if*

$$\mathbf{f}(\mathbf{x}) \leq_p \mathbf{f}(\mathbf{y}) \text{ and } \mathbf{f}(\mathbf{x}) \neq \mathbf{f}(\mathbf{y}),$$

else $\mathbf{x}$ *is called non-dominated by* $\mathbf{y}$.

(c) *A point* $\mathbf{x} \in S$ *is called (Pareto) optimal or a Pareto point if there exists no* $\mathbf{y} \in S$ *which dominates* $\mathbf{x}$.

If all the objectives $f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})$ and constraints $g_1(\mathbf{x}), ..., g_p(\mathbf{x}), h_1(\mathbf{x}), ..., h_q(\mathbf{x})$ are differentiable, the following theorem of Kuhn and Tucker [10, 11] states the necessary condition for Pareto optimality for unconstrained MOPs.

Theorem 2 (KKT necessary condition)**.** *Let* $\mathbf{x}^* \in S$ *be a Pareto point of a MOP as stated in Equation* (2.9)*; then there exists constants* $\alpha_1, \ldots, \alpha_k$*,* $\kappa_1, \ldots, \kappa_p$ *and* $\varrho_1, \ldots, \varrho_q$ *that satisfy:*

$$\begin{aligned} \sum_{i=1}^{k} \alpha_i \nabla f_i(\mathbf{x}^*) - \sum_{j=1}^{p} \kappa_j \nabla g_j(\mathbf{x}^*) - \sum_{m=1}^{q} \varrho_m \nabla h_m(\mathbf{x}^*) = 0 \\ g_j(\mathbf{x}) \leq 0 \quad j = 1, \ldots, p \\ h_m(\mathbf{x}) = 0 \quad m = 1, \ldots, q \\ \sum_{i=1}^{k} \alpha_i = 1 \\ \alpha_i \geq 0 \quad i = 1, \ldots, k \\ \kappa_j g_j(\mathbf{x}) = 0 \quad j = 1, \ldots, p \\ \kappa_j \geq 0 \quad j = 1, \ldots, p. \end{aligned} \tag{2.10}$$

Similar to the SOP case the scalar values κ_j and ϱ_m denote the Lagrange multipliers for the constraints. If the problem has no constraints ($p = 0$ and $q = 0$) then the problem stated in Equation (2.9) is an unconstrained MOP of the following form:

$$\min_{\mathbf{x} \in \mathbb{R}^n} \mathbf{f}(\mathbf{x}) \tag{2.11}$$

then, the following theorems hold for this case.

Theorem 3. *Let $\mathbf{x}^* \in S$ be a Pareto point of a MOP as stated in Equation (2.11); then there exists a vector $\alpha \in \mathbb{R}^k$ with $\alpha_i \geq 0$, $i = 1, \ldots, k$, and $\sum_{i=1}^k \alpha_i = 1$ such that*

$$\sum_{i=1}^{k} \alpha_i \nabla f_i(\mathbf{x}^*) = 0. \tag{2.12}$$

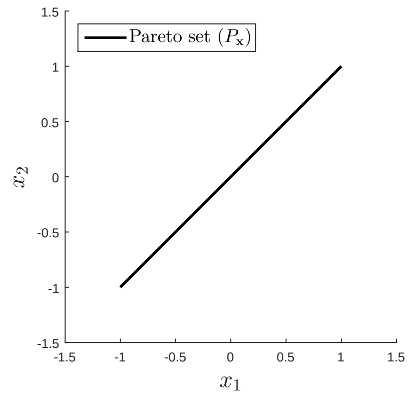
The theorem claims that the vector of zeros can be written as a convex combination of the gradients of the objectives at every Pareto point. Equation (2.12) is not a sufficient condition for Pareto optimality, but points which satisfy Equation (2.12) are certainly 'Pareto candidates'.

Definition 4. *Given an unconstrained MOP as stated in (2.11), a point $\mathbf{x} \in S$ is called a Karush-Kuhn-Tucker point (KKT point) if there exist scalars $\alpha_1, \alpha_2, \ldots, \alpha_k \geq 0$ such that $\sum_{i=1}^k \alpha_i = 1$ and that Equation (2.12) is satisfied.*

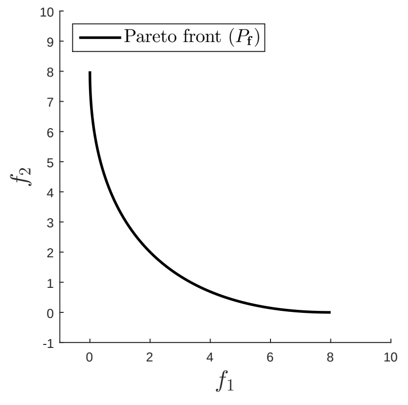
The set of all Pareto points is called the Pareto set, denoted by $P_\mathbf{x}$ and its image $P_\mathbf{f} = \mathbf{f}(P_\mathbf{x})$ the Pareto front. As example consider the following bi-objective problem MOP1 ([15]) whose Pareto front is convex:

$$\begin{aligned} &f_1, f_2 : \mathbb{R}^2 \to \mathbb{R} \\ &f_1(\mathbf{x}) = (x_1 - 1)^2 + (x_2 - 1)^2 \\ &f_2(\mathbf{x}) = (x_1 + 1)^2 + (x_2 + 1)^2. \end{aligned} \tag{2.13}$$

In Figure 2.2, we show both sets of interest (Pareto set and front) of the Problem (2.13).



(a) Desicion space



(b) Objective space

Figure 2.2: Example of the sets of interest (Pareto set and front) of a MOP.

2.1.3 Parameter Dependent Multi-Objective Optimization

In some real-world applications the solution set of a MOP depends on external parameters $\lambda \in \mathbb{R}^l$. The external parameter λ cannot be optimized but it has to be considered when the solution set is being computed. For instance, the speed and fuel-consumption are two objectives which are in conflict when engineers design a car. In this case, engineers want to achieve the best trade-off between these objectives, however, they have to be aware of considering external parameters such as side-wind, humidity, heat, among others when both objectives are measured. The influence of these external parameters will affect the solution set of the given problem leading to the parameter dependent multi-objective optimization problems (PMOPs) which can be defined as follows:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \mathbf{f}_\lambda(\mathbf{x}), \tag{2.14}$$

where $\mathbf{f}_\lambda$ is defined as a vector of objective functions that depends on an external parameter $\lambda \in \Lambda$ and

$$\begin{aligned} &\mathbf{f}_\lambda : S \subseteq \mathbb{R}^n \to \mathbb{R}^k, \\ &\mathbf{f}_\lambda(\mathbf{x}) = (f_1(\mathbf{x},\lambda), \ldots, f_k(\mathbf{x},\lambda))^T. \end{aligned} \tag{2.15}$$

$\Lambda \subseteq \mathbb{R}^l$ specifies the subspace where the external parameter is defined. Note that for every fixed value of λ the problem stated in Equation (2.14) can be seen as a classical

MOP. Thus, the solution set, unlike MOPs, consists of an entire family of Pareto sets defined as follows:

$$P_{\mathbf{x}_\lambda} := \{(\mathbf{x}, \lambda) \in \mathbb{R}^{n+l} | \mathbf{x} \text{ is a Pareto point of } \mathbf{f}_\lambda \text{ and } \lambda \in \Lambda\}. \tag{2.16}$$

The according family of Pareto fronts is denoted by $P_{\mathbf{f}_\lambda}$. As a general example, we consider here the following parameter dependent multi-objective optimizatio problem PMOP1 ([16]):

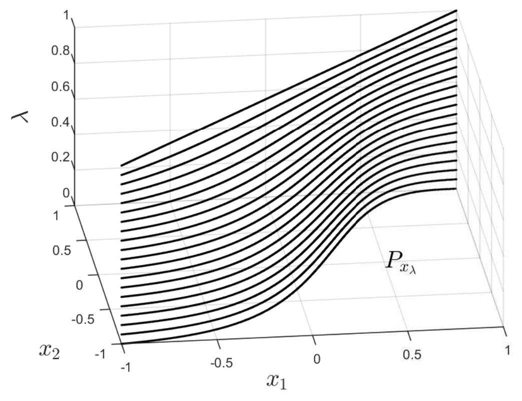
$$\begin{aligned} \mathbf{f}_\lambda &: \mathbb{R}^2 \to \mathbb{R}^2 \\ \mathbf{f}_\lambda(\mathbf{x}) &:= (1-\lambda)\mathbf{f}_1(\mathbf{x}) + \lambda \mathbf{f}_2(\mathbf{x}), \end{aligned} \tag{2.17}$$

where $\lambda \in [0,1]$ and $\mathbf{f}_1, \mathbf{f}_2 : \mathbb{R}^2 \to \mathbb{R}^2$,

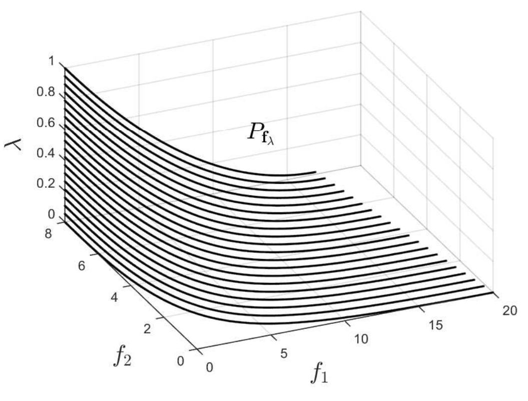
$$\mathbf{f}_1(x_1, x_2) = \begin{pmatrix} (x_1-1)^4 + (x_2-1)^2 \\ (x_1+1)^2 + (x_2+1)^2 \end{pmatrix},$$

$$\mathbf{f}_2(x_1, x_2) = \begin{pmatrix} (x_1-1)^2 + (x_2-1)^2 \\ (x_1+1)^2 + (x_2+1)^2 \end{pmatrix}.$$

This problem is a convex homotopy of the MOPs $\mathbf{f}_1$ and $\mathbf{f}_2$ which have both convex Pareto fronts. Figure 2.3 depicts both families of the Problem (2.17) .



(a) Decision space



(b) Objective space

Figure 2.3: Example of the solution sets (families of Pareto sets and fronts) of a PMOP.

2.1.4 Convergence Rates

Here, we include definitions related to the types of convergence. Let $\{\mathbf{x}_i\}$ be a sequence in $\mathbb{R}^n$ that converges to $\mathbf{x}^*$.

- The convergence is linear if there is a constant $c \in (0, 1)$ such that

$$\frac{||\mathbf{x}_{i+1} - \mathbf{x}^*||}{||\mathbf{x}_i - \mathbf{x}^*||} \leq c, \tag{2.18}$$

  for all i sufficiently large.

- The convergence is superlinear if

$$\lim_{i \to \infty} \frac{||\mathbf{x}_{i+1} - \mathbf{x}^*||}{||\mathbf{x}_i - \mathbf{x}^*||} = 0. \tag{2.19}$$

- The convergence is quadratic if there is a constant c not necessarily less than 1 such that

$$\frac{||\mathbf{x}_{i+1} - \mathbf{x}^*||}{||\mathbf{x}_i - \mathbf{x}^*||^2} \leq c, \tag{2.20}$$

  for all i sufficiently large.

2.2 Mathematical Programming Techniques for Optimization

To give a brief insight into the developments related to the numerical treatment of SOPs and MOPs. This section describes some available methods based on mathematical programming techniques to find optimal solutions for a given optimization problem (SOP or MOP).

2.2.1 Mathematical Techniques for SOPs

There are two classes of mathematical programming algorithms: line search and trust region methods. Throughout this work, we will only address line search strategies. For a thorough description of trust region methods we refer for instance to [12].

Line search methods for solving SOPs require a starting point, here denoted by $\mathbf{x}_0$. Once the starting point is given the algorithms perform a sequence of iterations that finalizes when no improvement is reached. Each iteration of the line search method computes a search direction ν_i and then decides how far to move along that direction. The iteration is given by

$$\mathbf{x}_{i+1} = \mathbf{x}_i + t_i \nu_i, \tag{2.21}$$

where $t_i > 0$ is called the step size. The success of this class of methods lies on the correct choice of both the direction ($\nu_i \in \mathbb{R}^n$) and the step size ($t_i > 0$). A simple stopping condition for line search algorithms is the reduction of f, that is, $f(\mathbf{x}_i + t_i\nu_i) < f(\mathbf{x}_i)$, however, it is not sufficient to guarantee convergence toward $\mathbf{x}^*$. A popular inexact line search condition stipulates that t_i should first of all give sufficient decrease in the objective function f, as measured by the following inequality:

$$f(\mathbf{x}_i + t_i\nu_i) \leq f(\mathbf{x}_i) + c_1 t_i \nabla f(\mathbf{x}_i)^T \nu_i, \tag{2.22}$$

for some constant $c_1 \in (0, 1)$. This means that the reduction in f should be proportional to both the step size t_i and the directional derivative $\nabla f(\mathbf{x}_i)^T \nu_i$. Equation (2.22) is called the Armijo condition. The sufficient decrease condition is not enough by itself, then to ensure the algoritm performs a reasonable progress it is introduced another requirement called the curvature condition which requires t_i to satisfy

$$\nabla f(\mathbf{x}_i + t_i\nu_i)^T \nu_i \geq c_2 \nabla f(\mathbf{x}_i)^T \nu_i, \tag{2.23}$$

for some constant $c_2 \in (c_1, 1)$, where c_1 is the constant of Equation (2.22). The sufficient decrease and curvature conditions are known collectively as the Wolfe conditions. Algorithm 1 represents the general framework for descent methods using line search.

Steepest Descent Method

A natural choice for a search direction is

$$\nu := -\nabla f(\mathbf{x}). \tag{2.24}$$

This is because among all the directions we could move $\mathbf{x}_i$, it is the one along which f decreases most rapidly [12] at least for small step sizes. The computation of t_i

Algorithm 1 General descent method

```
Require: A given starting point $\mathbf{x}_0 \in \mathbb{R}^n$.
Ensure: A candidate solution $\mathbf{x}_f$ for a given SOP.
1: Set $i := 0$
2: do
3:     Determine a descent direction $\nu_i \in \mathbb{R}^n$.
4:     Choose a step size $t_i \in \mathbb{R}_+$.
5:     Set $\mathbf{x}_{i+1} := \mathbf{x}_i + t_i\nu_i$
6:     Set $i := i + 1$.
7: while stopping criterion is satisfied
8: return $\mathbf{x}_f = \mathbf{x}_{i-1}$
```

can be done in different ways, for instance via exact or backtracking line search. One advantage of this method is that it only requires the gradient of the function. However, for difficult problems the method can be very slow [12]. We can take the norm of the gradient $||\nabla f(\mathbf{x})||_2 \leq tol$ as a stopping criterion where *tol* is a small and positive value. By using the general framework in Algorithm 1, we can use Equation (2.24) to determine ν_i.

The Newton Method

One of the most important search directions within line search strategies is the Newton direction which is derived from a second-order Taylor series approximation. Newton method assumes that f is twice continuously differentiable within $\mathbb{R}^n$. Given an unconstrained SOP a Newton step (or the Newton function) is defined as follows:

$$\begin{aligned} \mathbf{N} :& \mathbb{R}^n \to \mathbb{R}^n \\ \mathbf{N}(\mathbf{x}) :&= \mathbf{x} - \nabla^2 f(\mathbf{x})^{-1} \nabla f(\mathbf{x}). \end{aligned} \tag{2.25}$$

The Newton direction for the current point $\mathbf{x}$ is given then by

$$\nu_{\mathbf{N}} = -\nabla^2 f(\mathbf{x})^{-1} \nabla f(\mathbf{x}). \tag{2.26}$$

Thus, the Newton method is defined as

$$\begin{aligned} &\mathbf{x}_0 \in \mathbb{R}^n \\ &\mathbf{x}_{i+1} = \mathbf{N}(\mathbf{x}_i), \text{ for } \quad i = 0, 1, 2, \ldots. \end{aligned} \tag{2.27}$$

Unlike the steepest descent direction, a natural step size associated to the Newton direction is $t_i = 1$, since it is accepted by the Wolfe conditions for all large i (see [12] for more a detailed explanation). Nevertheless, a possible adjustment can be done. In most of the cases Newton's method will provide fast rates than local convergence, typically quadratic. However, its main disadvantage is that it requires the computation of the Hessian $\nabla^2 f(\mathbf{x})$ in each step. This is due to the process to compute this matrix, that in most of the cases can be computationally expensive.

2.2.2 Mathematical Programming Techniques for MOPs

In the following, we describe some mathematical programming techniques applied to the context of MOPs. Similar to SOPs these techniques need also a starting guess. The aim is to find points over the set of interest which is given by the Pareto set/front.

Scalarization Methods

One common way to solve MOPs is via scalarization methods, i.e., to transform the original problem (defined in Equation (2.9)) into a SOP of the form:

$$\min_{\mathbf{x}\in S} f_\alpha(\mathbf{x}), \tag{2.28}$$

where $f_\alpha : S \to \mathbb{R}$ and $\alpha \in \mathbb{R}^k$ is an external parameter. Note that for a given value of α the solution of Equation (2.28) is typically a single point rather than a $(k-1)$-manifold. In order to produce a finite size approximations of the sets of interest (Pareto set and front) one can solve a sequence of optimization problems [17–21]. In other words, one has to solve Equation (2.28) for a suitable set $A := \{\alpha^{(1)}, \ldots, \alpha^{(m)}\} \subset \mathbb{R}^k$ of external parameters where m defines the size of the approximation (see [22]).

The general advantage of the use of scalarization methods is that they can be tackled by any SOP solver. Nevertheless, scalarization methods only produced one single solution. In the following, some scalarization methods are introduced:

Weighted Sum Method. Probably, the first proposed scalarization method is the Weighted Sum Method [17, 18]. The underlying idea is to assign to each objective a certain weight $\omega_i \geq 0$, and to minimize the resulting weighted sum. Given a problem of the form of Equation (2.9), the weighted sum problem can be stated as follows:

$$\min_{\mathbf{x}\in S} \sum_{i=1}^{k} \omega_i f_i(\mathbf{x}), \tag{2.29}$$

where $\omega_i \geq 0$ for all $i = 1, \dots, k$ and $\sum_{i=1}^{k} \omega_i = 1$ [23].

Several theoretical results are presented concerning Equation (2.29):

Theorem 4 ([17,18]). *The solution of the problem stated in Equation* (2.29) *is weakly Pareto optimal.*

Theorem 5 ([17,18]). *The solution of the problem stated in Equation* (2.29) *is Pareto optimal if the weighting coefficients are positive, that is $\omega_i > 0$ for all $i = 1, ..., k$.*

The weakness of this method is that only if the Pareto front is convex all points on this set can be reached by solving Equation (2.25) for a particular value of ω. In other cases optimal solutions $\mathbf{x}^*$ where $f(\mathbf{x}^*)$ lies on a concave portion of the Pareto front, can be difficult to find.

Weighted Tchebycheff Method. The method proposed in [19] finds a point whose image is as close as possible to a given reference point $\mathbf{z} \in \mathbb{R}^k$. For the distance assignment the weighted Tchebycheff metric is mostly used: let $\omega \in \mathbb{R}^k$ with $\omega_i \geq 0$, $i = 1, \dots, k$, and $\sum_{i=1}^{k} \omega_i = 1$, and let $\mathbf{z} = [z_1, \dots, z_k]^T$, then the Weighted Tchebycheff Method reads as follows:

$$\min_{\mathbf{x}\in S} \max_{i=1,\dots,k} \omega_i |f_i(\mathbf{x}) - z_i|. \tag{2.30}$$

Note that the solution of Equation (2.30) depends on the choice of $\mathbf{z}$ as well as on ω. The main advantage of the Weighted Tchebycheff Method is that by a proper choice of these vectors every point on the Pareto front can be reached.

Theorem 6 ([19]). *The solution of Equation* (2.30) *is weakly Pareto optimal if $\omega \in \mathbb{R}^k_+$.*

Theorem 7 ([19]). *Let $\mathbf{x}^* \in S$ be Pareto optimal. Then there exists $\omega \in \mathbb{R}^k_+$ such that $\mathbf{x}^*$ is a solution of Equation* (2.30), *where $\mathbf{z}$ is chosen as the utopian vector of the MOP.*

The utopian vector $f^* = [f_1^*, \dots, f_k^*]^T$ of a MOP consists of the minimal objective values f_i^* of each function f_i. On the other hand, the proper choices of $\mathbf{z}$ and ω might also present delicate problems.

Normal Boundary Intersection Method. In 1998, an alternative scalararization method was proposed by Das and Dennis [20], the Normal Boundary Intersection method (NBI). The method computes a finite size approximation of the Pareto front using the following two steps:

1. Compute the convex hull of individual minima (CHIM) which is the $(k-1)$-simplex connecting the objective values of the minima of each objective.

2. Select points $\mathbf{y}_i$ from the CHIM and compute the point $\mathbf{z}_i^* \in S$ such that the image $f(\mathbf{x}_i^*)$ has the maximal distance from $\mathbf{y}_i$, in the direction normal to the CHIM.

To be more precise, let $\mathbf{x}_i^*$ be a global minimizer of the i-th objective, let $f_i^* := f(\mathbf{x}_i^*)$, and denote

$$\Phi := [f_1^*, \dots, f_k^*] \in \mathbb{R}^{k\times k}. \tag{2.31}$$

Then the CHIM is defined as

$$\text{CHIM} = \left\{ \Phi\omega : \omega \in \mathbb{R}^k : \sum_{i=1}^{k} \omega_i = 1, \omega_i \geq 1, i = 1, \dots, k \right\}. \tag{2.32}$$

The optimization problem in the second step is called the NBI-subproblem. Given an initial value $\Phi\omega = \sum_{i=1}^k \omega_i f_i^*$ and the direction $\mathbf{d} \in \mathbb{R}^k$ which is orthogonal to the CHIM, the NBI-subproblem can be stated in mathematical terms as follows:

$$\max_{\mathbf{x},t} t \tag{2.33}$$

$$\text{s.t } \mathbf{f}(\mathbf{x}_0) + t\mathbf{d} = \mathbf{f}(\mathbf{x})$$
$$\mathbf{x} \in S.$$

Descent Directions

A descent direction is defined as follows: given a point $\mathbf{x} \in \mathbb{R}^n$, a vector $\nu \in \mathbb{R}^n$ is called a descent direction if a search in that direction leads to an improvement of all the objective values. This means, ν is a descent direction of a MOP at a point $\mathbf{x} \in \mathbb{R}^n$ if there exists a $\tilde{t} \in \mathbb{R}_+$ such that

$$\mathbf{f}(\mathbf{x} + t\nu) <_p \mathbf{f}(\mathbf{x}), \ \forall t \in (0, \tilde{t}). \tag{2.34}$$

If all objectives of a MOP are differentiable, then Equation (2.34) is equivalent to

$$\nabla f_i(\mathbf{x})^T \nu < 0, \ i = 1, \ldots, k. \tag{2.35}$$

Hence, if a descent direction ν is given for an initial point $\mathbf{x}$, a solution $\mathbf{x}_{new}$ that dominates $\mathbf{x}$ can be found by performing a line search, i.e.,

$$\mathbf{x}_{new} = \mathbf{x} + t\nu, \tag{2.36}$$

where $t \in \mathbb{R}_+$ is a (sufficiently small) step size. In [24–27], some descent directions have been proposed. In the following, we introduce two descent directions that are important for this work.

Averaged Descent Direction. In [25], a descent direction is proposed for bi-objective optimization problems (i.e., $k = 2$). The Average Descent Direction takes advantage of the descent cone properties of a MOP (see [1]) and it reads as follows:

Theorem 8 ([10])**.** *Let $\mathbf{x} \in \mathbf{R}^n$, $f_1, f_2 : \mathbb{R}^n \to \mathbb{R}$ define a bi-objective unconstrained MOP, and $\nabla f_i \neq 0$ for $i = 1, 2$. Then, the direction*

$$\nu = -\frac{1}{2}\left(\frac{\nabla f_1(\mathbf{x})}{||\nabla f_1(\mathbf{x})||} + \frac{\nabla f_2(\mathbf{x})}{||\nabla f_2(\mathbf{x})||}\right), \tag{2.37}$$

where $||\cdot|| = ||\cdot||_2$, is a descent direction at $\mathbf{x}$ for the MOP.

This descent direction needs only the gradient information unlike other approaches that in most of the cases solve a linear quadratic optimization problem. Nonetheless, the method still has a clear disadvantage namely that it cannot be generalized for more than two objective functions and also it cannot (in its current form) manage constraint information.

Descent Direction of Schäffler, Schultz and Weinzierl. In [24], the authors present an approach for the computation of a descent direction for unconstrained MOPs. This descent direction is defined as follows:

Theorem 9 ([24]). *Let a given MOP as defined in Equation* (2.9) *and the map* $q : \mathbb{R}^n \to \mathbb{R}$ *be defined by*

$$q(\mathbf{x}) = \sum_{i=1}^{k} \tilde{\alpha}_i \nabla f_i(\mathbf{x}), \tag{2.38}$$

where $\tilde{\alpha}$ *is a solution of*

$$\min_{\alpha \in \mathbb{R}} \left\{ \left\| \sum_{i=1}^{k} \alpha_i \nabla f_i(\mathbf{x}) \right\|_2^2 : \alpha_i \geq 0,\ i = 1, \ldots, k,\ \sum_{i=1}^{k} \alpha_i = 1 \right\}. \tag{2.39}$$

Then the following statements hold:

- *Either* $q(\mathbf{x}) = 0$ *or* $-q(\mathbf{x})$ *is a descent direction.*
- *For each* $\tilde{\mathbf{x}} \in \mathbf{R}^n$, *there exists a neighborhood* $\mathcal{N}(\tilde{\mathbf{x}})$ *and a constant* $L_{\tilde{\mathbf{x}}} \in \mathbb{R}_0^+$ *such that*

$$||q(\mathbf{x}) - q(\mathbf{y})||_2 \leq L_{\tilde{\mathbf{x}}} ||\mathbf{x} - \mathbf{y}||_2,\ \forall \mathbf{x}, \mathbf{y} \in \mathcal{N}(\tilde{\mathbf{x}}). \tag{2.40}$$

Note that if $q(\mathbf{x}) = 0$, then $\mathbf{x}$ is a KKT point. While q is computed a test for fulliling the first order necessary condition of optimality is being performed.

Directed Search Method

The Directed Search (DS) Method has been proposed for differentiable MOPs which allows to steer the search process from a given point into a desired direction $\mathbf{d}$ in objective space [28]. To be more precise, given a point $\mathbf{x}_0 \in \mathbb{R}^n$, and $\mathbf{d} \in \mathbb{R}^k$ representing the desired search direction in objective space, a search direction $\nu \in \mathbb{R}^n$ in parameter space is sought such that

$$\lim_{t \to 0} \frac{f_i(\mathbf{x}_0 + t\nu) - f_i(\mathbf{x}_0)}{t} = d_i, \quad i = 1, \ldots, k. \tag{2.41}$$

Equation (2.41) can be stated in matrix vector notation, therefore such a direction vector ν solves the following system of linear equations:

$$J(\mathbf{x}_0)\nu = \mathbf{d}, \tag{2.42}$$

where $J(\mathbf{x})$ denotes the Jacobian of $\mathbf{f}$ at $\mathbf{x}$

$$J(\mathbf{x}) = \begin{pmatrix} \nabla f_1(\mathbf{x})^T \\ \vdots \\ \nabla f_k(\mathbf{x})^T \end{pmatrix} \in \mathbb{R}^{k \times n}. \tag{2.43}$$

Since typically $k << n$, we can assume that the linear system in Equation (2.42) is (highly) under-determined. Among the solutions of Equation (2.42), the one with the least 2-norm can be viewed as the greedy direction for the given context what means by using this direction we are going to obtain the maximum gain. This solution is given by

$$\nu_+ := J(\mathbf{x})^+ \mathbf{d}, \tag{2.44}$$

where $J(\mathbf{x})^+$ denotes the pseudo-inverse of $J(\mathbf{x})$ (we refer e.g. to [12] for an efficient computation of ν_+). Assuming a given direction $\mathbf{d} \in \mathbb{R}^k \backslash \{0\}$ with $d_i \leq 0$, $i = 0, \ldots, k$, a given point $\mathbf{x}_0 \in \mathbb{R}^n$ with $rank(J(\mathbf{x}_0)) = k$ and that the image of $\mathbf{f}$ is bounded from below. If one proceeds to a greedy search in direction $\mathbf{d}$ using Equation (2.44), this leads to the numerical solution of the following initial value problem (starting from solution $\mathbf{x}_0 \in \mathbb{R}^n$):

$$\begin{aligned} \mathbf{x}(0) &= \mathbf{x}_0 \in \mathbb{R}^n \\ \dot{\mathbf{x}}(t) &= J(\mathbf{x}(t))^+ \mathbf{d}, \quad t > 0, \end{aligned} \tag{2.45}$$

where t denotes the time. If $\mathbf{d}$ is a 'descent direction' (i.e., $d_i \leq 0$ for all $i = 1, \ldots, k$ and there exists an index j such that $d_j < 0$), a numerical solution of Equation (2.45) can be viewed as a downhill climber for MOPs.

The downhill climber described above shares many characteristics with the one described in [27], where also possible choices for $\mathbf{d}$ are discussed.

The solutions produced by solving the problem stated in Equation (2.45) are characterized as follows: let $\Upsilon : [0, t_f] \to \mathbb{R}^n$ be such a solution, where t_f is the final value, and let t_c be the smallest value of $t \geq 0$ such that

$$\nexists \nu \in \mathbb{R}^n : J(x(t))\nu = \mathbf{d}. \tag{2.46}$$

t_c is called the critical value and $\Upsilon(t_c)$ the critical point of the problem stated in Equation (2.45). Υ can be divided into two parts: $\Upsilon([0, t_c])$ and $\Upsilon(t_c, t_f)$. The first part $\mathbf{f}(\Upsilon(t))$ yields the desired decay in the direction $\mathbf{d}$. From the critical point $\Upsilon(t_c)$ on a 'best fit' is computed, i.e.,

$$\nu_+(\mathbf{x}(t)) = J(\mathbf{x}(t))^+\mathbf{d} = arg\ min_{\nu \in \mathbb{R}^n} ||J(\mathbf{x}(t))\nu - \mathbf{d}||. \tag{2.47}$$

For the end point $\Upsilon(t_f)$ it holds $J(\Upsilon(t_f))^+d = 0$. The end points are of particular interest since they are KKT points with associated convex weight $\alpha = -\mathbf{d}/||d||_1$, however, the computation of Υ in $[t_c, t_f]$ might be computationally expensive the problem stated in Equation (2.45) is stiff over this second part. Thus, the directed search is restricted to detect points $\Upsilon(t_c)$.

The directed search descent method uses as a stopping criterion the facts that $rank(J(\mathbf{x}_0)) = k$ (by assumption) and that the $rank(J(\mathbf{x}^*)) < k$ (by definition of the critical point $\mathbf{x}^*$). However, it is not possible to detect numerically a critical point using the rank of a matrix, then one can compute

$$cond(J(\mathbf{x})) = ||J(\mathbf{x})|| ||J(\mathbf{x})^+|| = \sigma_1/\sigma_k, \tag{2.48}$$

where σ_1 and σ_k are the biggest and smallest singular values of $J(\mathbf{x})$, respectively. So the process can be stopped if

$$cond(J(\mathbf{x}_i)) \geq tol, \tag{2.49}$$

where $tol \in \mathbb{R}^+$ is a given (large) threshold. For a more thorough explanation we refer to [28].

There exist the possibility to use the DS for performing a search along the Pareto set in a predictor-corrector fashion and the method can even be adapted to be realized gradient free. Different to the other approaches, the 2nd derivative information is not necessary. For the predictor the method uses the orthogonal vector α to the Pareto front which is the convex weight related with a given local Pareto point $\mathbf{x}$ such that $\sum_{i=1}^{k} \alpha_i \nabla f_i(\mathbf{x}) = 0$ under the assumption that $\text{rank}(J(\mathbf{x})) = k - 1$. In this case α is orthogonal to the Pareto front, i.e, $\alpha \perp T_y \partial\mathbf{f}(\mathbb{R}^n)$ where T_y denotes the tangent space of $\partial\mathbf{f}(\mathbb{R}^n)$. Thus, the next idea is a search orthogonal to α, for that reason the QR-factorization of α is necessary to obtain the orthonormal basis of the tangent space. In this way, we obtain an orthogonal matrix $Q = (q_1, \ldots, q_k) \in \mathbb{R}^{k\times k}$ and q_i, $i = 1, \ldots, k$, its column vectors, and $R = (r_{11}, 0, \ldots, 0)^T \in \mathbb{R}^{k\times 1}$ with $r_{11} \in \mathbb{R}\backslash\{0\}$.

Since $\alpha = r_{11}q_1$(i.e., $\alpha \in \text{span}\{q_1\}$) and Q is orthogonal, the vectors $(q_2, \dots, q_k)$ form the orthonormal basis of the orthogonal hyperplane to α.

Using

$$J(\mathbf{x})\nu_i = q_i, \ i = 2, \dots, k, \tag{2.50}$$

we can obtain directions ν_i to have a predictor point without any second gradient information. To perform the corrector step, we can use the predictor point $\mathbf{p}$ and solve Equation (2.45). Other predictor-corrector methods to perform a search along the solution set of a MOP given a local optima solution $\mathbf{x}^*$ as starting point can be found in [29–34].

2.2.3 Hypervolume Gradient

In the following, we introduce some notations for a better understanding of the hypervolume gradient. Throughout this work, we consider approximation sets of size μ to the efficient set $P_{\mathbf{x}}$ by vectors $\mathbf{X} \in \mathbb{R}^{\mu \cdot n}$. Each of the points of the approximation can be obtained as follows: the first point $\mathbf{x}^{(1)} \in S$ is represented by the first n consecutive components of $\mathbf{X}$ and the second one by the n consecutive components, an so on. The $\mu \cdot n$-vector $\mathbf{X}$ which contains the approximation set (or population) is defined as follows:

$$\mathbf{X} = \left(\mathbf{x}^{(1)^\top}, \mathbf{x}^{(2)^\top}, \dots, \mathbf{x}^{(\mu)^\top}\right)^\top, \ \mathbf{x}^{(i)} \in S, \ i = 1, \dots, \mu.$$

In addition, a mapping $\mathbf{F} : \mathbb{R}^{\mu \cdot n} \to \mathbb{R}^{\mu \cdot k}$ is obtained from the objective function vector $\mathbf{f}$ (which is defined in Problem (2.11)):

$$\mathbf{F}(\mathbf{X}) := \left(\mathbf{f}(\mathbf{x}^{(1)})^\top, \mathbf{f}(\mathbf{x}^{(2)})^\top, \dots, \mathbf{f}(\mathbf{x}^{(\mu)})^\top\right)^\top. \tag{2.51}$$

By defining $\mathbf{Y} := \mathbf{F}(\mathbf{X})$, we can represent the image of the $\mu \cdot n$-vector $\mathbf{X}$ as:

$$\mathbf{Y} := (y_1^{(1)}, .., y_k^{(1)}, .., y_1^{(i)}, .., y_k^{(i)}, .., y_1^{(\mu)}, .., y_k^{(\mu)})^T \in \mathbb{R}^{\mu \cdot k}. \tag{2.52}$$

The hypervolume indicator $\mathcal{H}$ is defined as the Lebesgue measure on $\mathbb{R}^k$, which is the region of the dominated set in the reference space (see Equation (2.72)). Then, for a given vector $\mathbf{X}$ representing the approximation set (or population) we define the hypervolume indicator for vectors in $\mathbb{R}^{\mu \cdot n}$ as

$$\mathcal{H}_{\mathbf{F}}(\mathbf{X}) = \mathcal{H}(\mathbf{F}(\mathbf{X})). \tag{2.53}$$

In [35], the authors presented a definition of the hypervolume gradient, then in [36] they extended the definition to dimensions $k > 2$ and provided efficient algorithms for its computation for $k \leq 4$. In both works, they stated that for the mapping $\mathcal{H}_{\mathbf{F}}$ exists a gradient field $\nabla\mathcal{H}_{\mathbf{F}}$ if all partial derivates with respect to $\mathcal{H}_{\mathbf{F}}$ are well defined for a given $\mu \cdot n$-vector $\mathbf{X}$ and if one-sided derivatives are used in case of duplicate coordinates in the objective space. The hypervolume indicator gradient $\nabla\mathcal{H}_{\mathbf{F}}(\mathbf{X})$ of the composition $\mathcal{H}_{\mathbf{F}} = \mathcal{H} \circ \mathbf{F}$ is defined as:

$$\nabla\mathcal{H}_{\mathbf{F}}(\mathbf{X}) = \left(\frac{\partial\mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial x_1^{(1)}}, .., \frac{\partial\mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial x_n^{(1)}}, .., \frac{\partial\mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial x_1^{(\mu)}}, .., \frac{\partial\mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial x_n^{(\mu)}}\right)^T = \left(\frac{\partial\mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(1)}}^\top, \dots, \frac{\partial\mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(\mu)}}^\top\right)^\top. \tag{2.54}$$

Note that each term in the right-hand-side of the above equation is called *sub-gradient*, which is the local hypervolume change rate by moving only one decision vector infinitesimally. Moreover, the sub-gradients can be computed by applying the chain rule [36, 37]:

$$\frac{\partial\mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(i)}} = \frac{\partial \mathbf{Y}}{\partial \mathbf{x}^{(i)}} \frac{\partial\mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{Y}} = \sum_{z=1}^{k} \frac{\partial\mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_z(\mathbf{x}^{(i)})} \nabla f_z(\mathbf{x}^{(i)}). \tag{2.55}$$

In order to visualize the structure of this composition, we present the matrix representation:

$$\underbrace{\left(\left(\begin{pmatrix} \frac{\partial\mathcal{H}}{\partial y_1^{(1)}} \\ \vdots \\ \frac{\partial\mathcal{H}}{\partial y_k^{(1)}} \\ \vdots \\ \frac{\partial\mathcal{H}}{\partial y_1^{(\mu)}} \\ \vdots \\ \frac{\partial\mathcal{H}}{\partial y_k^{(\mu)}} \end{pmatrix}\right) (\mathbf{F}(\mathbf{X}))\right)^T}_{\nabla\mathcal{H}(\mathbf{F}(\mathbf{X}))} \cdot \underbrace{\left(\begin{pmatrix} \nabla\mathbf{f}(\mathbf{x}^{(1)}) & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \nabla\mathbf{f}(\mathbf{x}^{(\mu)}) \end{pmatrix}\right)}_{\nabla\mathbf{F}(\mathbf{X})}. \tag{2.56}$$

According to Equation (2.56), it is clear that the right hand side depends only on the gradient of $\nabla\mathbf{F}$ at the sub-vectors $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\mu)}$. Thus, if our MOP is differentiable, the Jacobian matrix $\nabla\mathbf{F}$ can be computed. In order to compute the left hand side we present the following concepts (see also [36]):

Definition 5. *Let* $\pi_{1,\ldots,\tilde{m},\ldots,k}(\mathbf{y}) \in \mathbb{R}^{k-1}$ *denote the projection of a sub-vector* $\mathbf{y}$ *in the multiset onto the coordinates* $1,\ldots,\tilde{m},\ldots,k$*, where* $\tilde{m}$ *means that* m *is omitted.*

Definition 6. *Let* $i \in \{1,\ldots,\mu\}$*. Let* $\mathcal{H}_{k-1}$ *denote the hypervolume indicator for the* $(k-1)$*-dimensional space with reference space* $[\pi_{1,\ldots,\tilde{m},\ldots,k}(\mathbf{r},\infty)]$*. Let* $Y_{(i)}^{<k}$ *denote the multiset of projections* $\pi_{1,\ldots,\tilde{m},\ldots,k}(\mathbf{y}^{(i)})$ *of subvectors* $\mathbf{y}^{(i)}$ *of a multiset* $\mathbf{Y}$ *with a lower* m*-coordinate than the* m*-coordinate of the subvector* $\mathbf{y}^{(i)}$*.*

$$\frac{\partial \mathcal{H}}{y_m^{(i)}}(Y) = \mathcal{H}_{k-1}(\mathbf{Y}_{(i)}^{<m} \cup \{\pi_{1,\ldots,\tilde{m},\ldots,k}(\mathbf{y}^{(i)})\}) - \mathcal{H}_{k-1}(\mathbf{Y}_{(i)}^{<m}). \tag{2.57}$$

2.2.4 The Gradient-Ascent Hypervolume Method

The hypervolume gradient has been used in [35] as a gradient-ascent hypervolume method (GAH). They described a population based algorithm which allows to perform a steepest ascent procedure with backtracking line search. The method requires a good starting point in order to reach the Pareto front. To be more precise, the steepest ascent method takes as a starting point the final approximation given by the evolutionary algorithm. Then it computes the gradient which represent the direction toward each member of the population have to move. Such movement was used to do a fine-tunning of the solutions.

2.3 Stochastic Search Techniques for Optimization

When gradient information is not available none of the previously described mathematical programming techniques can be applied to the given optimization problem. Stochastic algorithms are methods which use probabilistic transition rules can be a suitable alternative to tackle SOPs, MOPs, and PMOPs without gradient information. Among stochastic search techniques, evolutionary approaches which are inspired in nature have demonstrated to solve a wide range of optimization problems. These techniques evolve a population toward the solution or solution set. The process starts by generating a starting population. Then, evolutionary operators are performed to generate and preserve better solutions in the population. The previous process is made until a stopping criteria is reached or fulfilled.

2.3.1 Evolutionary Techniques to Tackle SOPs

Unlike mathematical programming techniques there are some approaches that have also performed well over problems were gradient information is not available. These techniques include knowledge from evolution, swarm intelligence, nature, among others. Here, we mention several approaches which have been used to tackle SOPs.

EAs [38]. Evolutionary algorithms (EAs) are methods based on natural selection. Such methods start from a given initial population which is evaluated according to a given SOP. In every iteration, the algorithm produces new elements by using evolutionary operators such as recombination and mutation. Then, the current and new population are mixed in order to select the elements which as the best value according the SOP. The process can be repeated until some stopping criterion is satisfied.

ESs [39, 40]. Evolutionary strategies (ESs) similar to EAs use mutation, recombination, and selection operators applied to an individual or a complete population. However, for ESs the mutation operator received more importance than recombination. The aim is to produced better candidate solutions in every iteration of the algorithm by the effect of the evolutionary operators. There exist two basic configurations, (i) $(1+1)$-ES and (ii) $(\mu+1)$-ES, where μ represents the number of elements in the population and the other parameter represents the number of new elements produced by evolutionary operators.

PSO [41]. The particle swarm optimization method (PSO) optimizes a given candidate solution $\mathbf{x}$ with regard to a given SOP. The algorithm uses a population called swarm of a certain number of particles. PSO solves the problem by moving the particles over a decision space according to a formula. This formula considers the particle's position and velocity. The performed movement for each particle is influenced by a local best known position and also guided toward the best known positions within the decision space. The expected result is to move the particle to the region that optimizes the SOP at hand.

DE [42]. The differential evolution (DE) method is an optimizer which seeks to find a solution of a continuous SOP. The basic idea of DE is to perturb a gradient search by a stochastic process. DE evolves a population of candidate solutions and creates a new element using a formula which utilize some of the current elements of the population. Then, the algorithm keep the solutions which has a better objective value while the others are eliminated.

2.3.2 Evolutionary Techniques to Tackle MOPs

When solving a MOP, one has to perform a series of separate runs in the case of using scalarization methods, as the given MOP is transformed into a series of SOPs. In contrast, subdivision techniques (see [25,43–51]) and evolutionary computing algorithms can find several members of the Pareto set in a one single run of the algorithm. The evolutionary computing algorithms that tackle MOPs, are called multi-objective evolutionary algorithms (MOEAs) [22,52–54]. A multi-objective evolutionary algorithm normally shares the structure and concepts analogous to its genetic counterparts. Normally, a structure or individual is an encoded solution to some problem and the set of individuals is called population. Within MOEAs there exist evolutionary operators that act over the population generating solutions with higher fitness. This fitness is going to determine which element will survive for the next generation. The main operators are mutation, recombination and selection [22]. Similar to methods for the treatment of SOPs such as evolutionary algorithms, evolutionary strategies, particle swarm, and differential evolution adding decomposition and indicator based algorithms have been used to tackle MOPs. In most of the cases the strategies use a dominance strategy to select the points that will survive for the next iteration of the method. On the other hand decomposition based algorithms divide the MOPs into subproblems that are solved as a SOP. Finally, indicator based algorithms transform the MOP into a SOP by using a performance indicator. In the following, we present some of the most representative MOEAs.

NSGA-II The non-dominated sorting genetic algorithm II (NSGA-II) was proposed in [55]. This algorithm generates in every step a new population from the current one by using evolutionary operators. Then, both populations are combined to double the original size. Next, NSGA-II performs a rank which divides the elements of each population into s layers or sub-fronts. The first sub-front will contain the best elements of the combination of both populations, and the final one the worst elements with respect to the non-dominance grade. The elements that belong to the first fronts will be considered for the next population. In the case that one sub-front has more elements than needed to achieve the original size the crowding distance is computed to select the elements which are distributed best over the Pareto front.

MOEA/D The multi-objective evolutionary algorithm based on decomposition (MOEA/D) was introduced by Zhang et. al in [56]. The algorithm exploits the idea of solving a MOP by decomposing it into some single objective sub-problems. MOEA/D starts by assuming that under certain conditions it is possible to generate a discretization of the Pareto front by solving several SOPs. The algorithm associates every sub-problem to a neighborhood which contains some of the elements of the whole population. In every iteration the algorithm will select a sub-problem and generate a new element by evolutionary operators. By using a scalarization function such as Penalty Boundary Intersection method (included in [56]) or Tchebycheff, MOEA/D updates the elements that belongs to the associated neighborhood in order to keep the ones with the best fitness value. The optimization process will be repeated until a stopping criteria is reached.

SMS-EMOA It is important to mentioned that the hypervolume is also called S-metric. The S-metric selection evolutionary multi-objective algorithm (SMS-EMOA) was proposed in [57] by Emmerich, Beume and Naujoks. This algorithm adapts an evolutionary strategy to solve a given MOP. Over the algorithm the hypervolume governs the selection operator, which allows to get at the end a good approximation according to that performance indicator. The main idea is to integrate new points in the population by replacing such elements whose contribution to hypervolume is the worst. The algorithm starts with an initial population of μ elements. In every iteration a new individual is generated by means of random variation operators 'crossover' and 'mutation'. The new element is added to the current population population. The selection process starts by generating a non-dominating ranking as is performed in NSGA-II [55]. This ranking separates the population into s different fronts $(G_1, \ldots, G_s)$. Finally, the algorithm computes the contribution of each element that belongs to the worst front G_s according to the hypervolume and deletes the element with the least hypervolume contribution. This process is repeated until a certain number of iterations is reached. A pseudo-code of SMS-EMOA is shown in Algorithm 2.

Algorithm 2 SMS-EMOA algorithm

```
Require: A given MOP.
Ensure: An approximation set $P$ of the solution set of the given MOP.
1: Initialize randomly a population $P \subset S$ with $\mu$ elements.
2: while the stopping-criteria is not reached do
3:     Generate an offspring $\mathbf{x} \in S$ from $P$ by variation.
4:     Integrate $\mathbf{x}$ into the population $P := P \cup \{\mathbf{x}\}$.
5:     Build the ranking of fronts $G_1, \ldots, G_s$ from $P$.
6:     Compute the hypervolume contribution for each point $\mathbf{x}' \in G_h$.
7:     Find and discard from $P$ the point $\mathbf{x}'^*$ with the least hypervolume contribution
   $P := P \backslash \{\mathbf{x}'^*\}$.
8: end while
9: return $P$.
```

2.3.3 Behaviour of Stochastic Local Search within PMOPs

Before introducing some evolutionary techniques for the treatment of a given PMOP, we present an investigation of the behavior of stochastic local search (SLS) for continuous PMOPs developed in [58]. By utilizing a certain relation of SLS with line search methods as used in mathematical programming we will see that—under certain (mild) assumptions on the model—both pressure toward and along the set of interest (in objective space) is already inherent in SLS. Initial studies on a simple set based method that includes SLS underline that the problem to compute a finite size representation of the entire solution set via stochastic search methods such as evolutionary algorithms (EAs) is a well-conditioned problem. The obtained results in [58] are valuable for the design of a specialized EAs included in this thesis work. The results in particular suggest that it might make sense to integrate the entire λ-space into the search which will allow to compute the desired approximation in *one* run of the algorithm which is in contrast to the works included in the next subsection which consider 'λ-slices' in each run.

For our considerations it is advantageous to treat λ —at least formally— within PMOPs as a 'normal' parameter leading to the following problem:

$$F : \mathbb{R}^{n+l} \to \mathbb{R}^{k+l}$$
$$F(\mathbf{x}, \lambda) = \begin{pmatrix} f_1(\mathbf{x}, \lambda) \\ \vdots \\ f_k(\mathbf{x}, \lambda) \\ \lambda \end{pmatrix} = \begin{pmatrix} g_1(\mathbf{x}, \lambda) \\ \vdots \\ g_{k+l}(\mathbf{x}, \lambda) \end{pmatrix}, \tag{2.58}$$

where $g_i : \mathbb{R}^{n+l} \to \mathbb{R}$, $i = 1, \ldots, k + l$. The Jacobian is given by

$$J(\mathbf{x}, \lambda) = \begin{pmatrix} \nabla_{\mathbf{x}} f_1(\mathbf{x}, \lambda)^T & \nabla_{\lambda} f_1(\mathbf{x}, \lambda)^T \\ \vdots & \vdots \\ \nabla_{\mathbf{x}} f_k(\mathbf{x}, \lambda)^T & \nabla_{\lambda} f_k(\mathbf{x}, \lambda)^T \\ 0 & I_l \end{pmatrix} := \begin{pmatrix} J_{\mathbf{x}} & J_{\lambda} \\ 0 & I_l \end{pmatrix} \in \mathbb{R}^{(k+l)\times(n+l)}, \tag{2.59}$$

where

$$J_x = \begin{pmatrix} \nabla_{\mathbf{x}} f_1(\mathbf{x}, \lambda)^T \\ \vdots \\ \nabla_{\mathbf{x}} f_k(\mathbf{x}, \lambda)^T \end{pmatrix} \in \mathbb{R}^{k\times n}, \quad J_{\lambda} = \begin{pmatrix} \nabla_{\lambda} f_1(\mathbf{x}, \lambda)^T \\ \vdots \\ \nabla_{\lambda} f_k(\mathbf{x}, \lambda)^T \end{pmatrix} \in \mathbb{R}^{k\times l}, \tag{2.60}$$

and where I_l denotes the $(l \times l)$-identity matrix.

To understand the behaviour of SLS it is advantageous to see its relation to line search as it is used in mathematical programming: if a point $z_1 = (\mathbf{x}_1, \lambda_1)$ is chosen (at random) from a small neighbourhood of $z_0 = (\mathbf{x}_0, \lambda_0)$, then z_1 can be written as

$$z_1 = z_0 + 1(z_1 - z_0) = z_0 + ||z_1 - z_0|| \frac{z_1 - z_0}{||z_1 - z_0||}. \tag{2.61}$$

Thus, the selection of z_1 can be viewed as a search in direction

$$v := \frac{(z_1 - z_0)}{||z_1 - z_0||}. \tag{2.62}$$

For infinitesimal steps in a direction $\nu \in \mathbb{R}^{n+l}$ (in decision space) the related change in objective space is given by $J(\mathbf{x}_0, \lambda_0)\nu$. To see this, consider the i-th component of $J(\mathbf{x}_0, \lambda_0)\nu$:

$$\begin{aligned}(J(\mathbf{x}_0, \lambda_0)\nu)_i = \lim_{t\to 0} \frac{g_i((\mathbf{x}_0, \lambda_0) + t\nu) - g_i(\mathbf{x}_0, \lambda_0)}{t} &= \langle \nabla g_i(\mathbf{x}_0, \lambda_0), \nu\rangle, \\ i &= 1, \ldots, k + l.\end{aligned} \tag{2.63}$$

For problem (2.58) this direction is given by

$$J\nu = \begin{pmatrix} J_{\mathbf{x}} & J_\lambda \\ 0 & I_l \end{pmatrix} \begin{pmatrix} \nu_{\mathbf{x}} \\ \nu_\lambda \end{pmatrix} = \begin{pmatrix} J_{\mathbf{x}}\nu_x + J_\lambda \nu_\lambda \\ \nu_\lambda \end{pmatrix}, \tag{2.64}$$

where $J = J(\mathbf{x}_0, \lambda_0)$ and $\nu = (\nu_{\mathbf{x}}, \nu_\lambda)$ with $\nu_x \in \mathbb{R}^n$ and $\nu_\lambda \in \mathbb{R}^l$.

Based on these considerations,one can consider different scenarios for SLS that occur in different stages within an evolutionary algorithm.

(a) $(\mathbf{x}, \lambda)$ **'far away' from** $P_{\mathbf{x}_\lambda}$**.** Here we use an observation made in [1] for classical MOPs namely that the 'objectives gradients' may point into similar directions when the decision point $(\mathbf{x}, \lambda)$ is far from the Pareto set. We assume here the extreme case namely that all gradients point into the same direction. For this, let $u := \nabla_{\mathbf{x}} f_i(\mathbf{x}, \lambda)$ and assume that

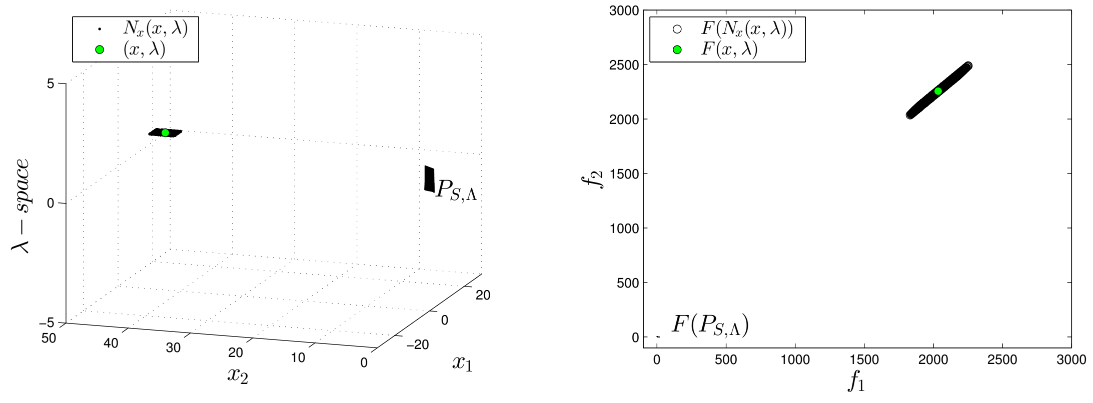
$$\nabla_{\mathbf{x}} f_i(\mathbf{x}, \lambda) = \mu_i u, \; i = 1, \ldots, k, \tag{2.65}$$

where $\mu_i > 0$ for $i = 1, \ldots, k$. Then

$$J_{\mathbf{x}}\nu_{\mathbf{x}} = \begin{pmatrix} \nabla_{\mathbf{x}} f_1(\mathbf{x}, \lambda)^T \nu_{\mathbf{x}} \\ \vdots \\ \nabla_{\mathbf{x}} f_k(\mathbf{x}, \lambda)^T \nu_{\mathbf{x}} \end{pmatrix} = u^T \nu_{\mathbf{x}} \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_k \end{pmatrix}. \tag{2.66}$$

That is, the movement is 1-dimensional regardless of $\nu_{\mathbf{x}}$ which is n-dimensional. Since $J_{\mathbf{x}}\nu_{\mathbf{x}} = 0$ iff $\nu_{\mathbf{x}} \perp u$, the probability is one that for a randomly chosen $\nu_{\mathbf{x}}$ either dominated or dominating solutions are found (and in case a dominated solution is found, the search has simply to be flipped to find dominating solutions).
Thus, for $\nu_\lambda = 0$, which means that the value of λ is not changed in the local search, we obtain for $\mu = (\mu_1, \ldots, \mu_k)^T$

$$J\nu = \begin{pmatrix} u^T \nu_x \mu \\ 0 \end{pmatrix}. \tag{2.67}$$



(a) Decision space (b) Objective space

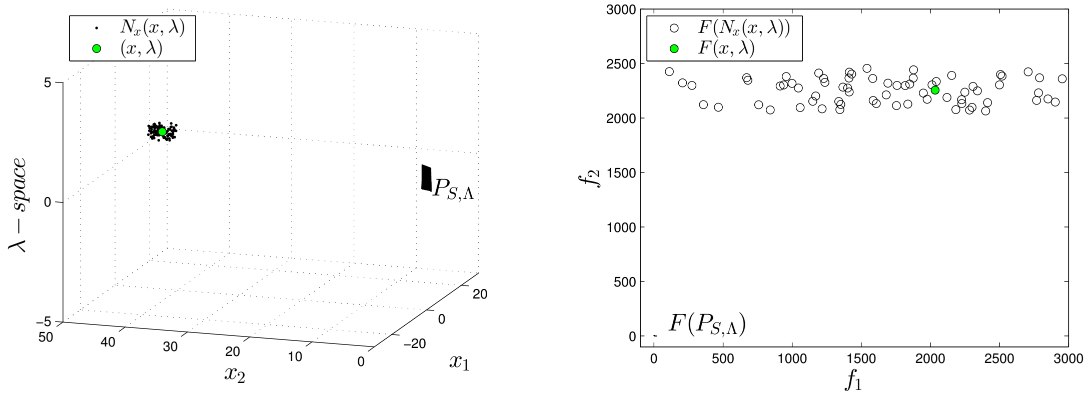
Figure 2.4: SLS for a point that is 'far away' from $P_{\mathbf{x}_\lambda}$ using $\nu_\lambda = 0$.

For $\nu_\lambda \neq 0$, i.e., in the case that the value of λ is changed within the local search, no such physical meaning exists to the best of our knowledge. Nevertheless, the investigation of this problem is still undone.

As an example, consider PMOP1 (2.17). Figures 2.4 and 2.5 show the behavior of SLS for 100 uniformly randomly chosen points near $(\mathbf{x}, \lambda) = (10, 45.2, 0.7)$ for $\nu_\lambda \neq 0$ and $\nu_\lambda = 0$. As neighbourhood the authors chose the infinity norm with radius $r_\mathbf{x} = 2$ in $\mathbf{x}$-space and $r_\lambda = 0.3$ (respectively $r_\lambda = 0$) in λ-space. For the case $\nu_\lambda = 0$ a clear movement toward/against $P_{\mathbf{f}_\lambda}$ can be observed while this is not the case for $\nu_\lambda \neq 0$. Thus, it may make sense to exclude the change of the value of λ in early stages of the search process where the individuals of the populations are supposed to be far away from the set of interest.

(b) $(\mathbf{x}, \lambda)$ **'near' to** $P_{\mathbf{x}_\lambda}$**.** Consider again the extreme case, namely that $\mathbf{x}$ is a Karush-Kuhn-Tucker (KKT) point of F. That is, assume that there exists a convex weight $\alpha \in \mathbb{R}^k$ such that

$$\sum_{i=1}^{k} \alpha_i \nabla_x f_i(\mathbf{x}, \lambda) = J_\mathbf{x}^T \alpha = 0. \tag{2.68}$$



(a) Decision space (b) Objective space

Figure 2.5: SLS for a point that is 'far away' from $P_{\mathbf{x}_\lambda}$ using $\nu_\lambda \neq 0$.

It can be shown ([59]) that the normal vector to the linearized set $P_{\mathbf{f}_\lambda}$ at $(\mathbf{x}, \lambda)$ is given by

$$\eta = \begin{pmatrix} \alpha \\ -{J_\lambda}^T \alpha \end{pmatrix}. \tag{2.69}$$

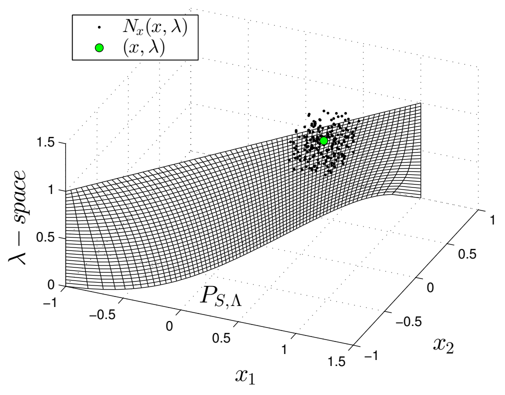
One obtains

$$\langle J\nu, \eta \rangle = \langle \nu, J^T \eta \rangle = \langle \nu, \begin{pmatrix} J_{\mathbf{x}}^T & 0 \\ J_\lambda^T & I_l \end{pmatrix} \begin{pmatrix} \alpha \\ -{J_\lambda}^T \alpha \end{pmatrix} \rangle = \langle \nu, \begin{pmatrix} J_{\mathbf{x}}^T \alpha \\ J_\lambda^T \alpha - J_\lambda^T \alpha \end{pmatrix} \rangle = 0. \tag{2.70}$$

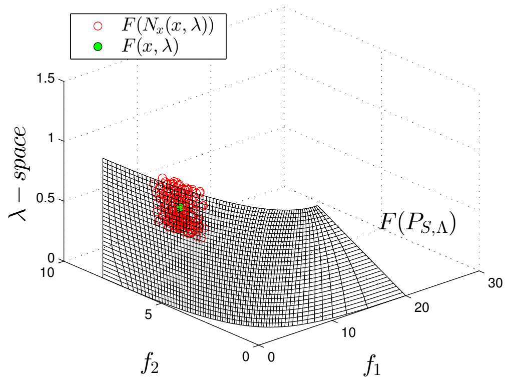
That is, it is either (i) $J\nu = 0$ or (ii) $J\nu$ is a movement orthogonal to η and thus along the linearized set at $F(\mathbf{x}, \lambda)$. If we assume that the rank of J_x is $k-1$, then the rank of J is $k-1+l$ and the dimension of the kernel of J is $n-k+l$. Hence, for a randomly chosen ν the probability is 1 that event (ii) happens.
Equation (2.70) tells us that the movement is orthogonal to the normal vector, but it remains to investigate in which direction of the tangent space the movement is performed. For this, let

$$\eta = QR = (q_1, q_2, \ldots, q_{k+l})R \tag{2.71}$$

be a QR factorization of η. Then, the vectors $q_2, \ldots, q_{k+l}$ form an orthonormal basis of the tangent space. If we assume again that the rank of $J_{\mathbf{x}}$ is $k-1$, then the rank of J is $k-1+l$. Since by Equation (2.70) η is not in the image of J, there exist vectors $\nu_q, \ldots, \nu_{k+l}$ such that $J\nu_i = q_i$, $i = 2, \ldots, k+l$. Thus, a movement via SLS



(a) Decision space



(b) Objective space

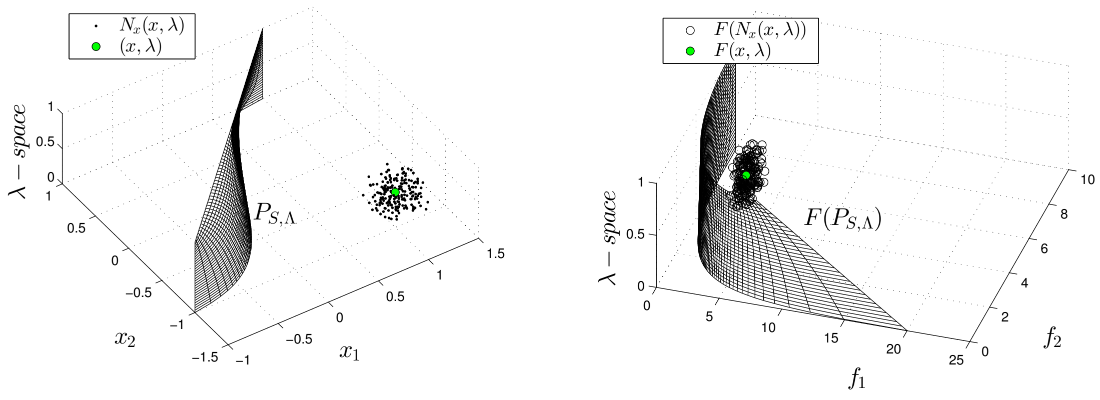
Figure 2.6: SLS for a point that is 'near' to $P_{\mathbf{x}_\lambda}$.

can be performed in all directions of the linearized family of Pareto fronts (i.e., both in $\mathbf{x}$- and λ-direction). Figure 2.6 shows an example for $(\mathbf{x}, \lambda) = (0.44, 0.47, 0.84)^T$ and $r_{\mathbf{x}} = r_\lambda = 0.2$. Again, by construction, no structure in decision space can be observed, but a clear movement along the set of interest can be seen in objective space.

(c) $(\mathbf{x}, \lambda)$ **'in between'.** Apparently, points $(\mathbf{x}, \lambda)$ do not have to be far away from nor near to the set of interest but can be 'in between'. In this case, no clear preference of the movement in objective space can be detected. However, this 'opening' of the search compared to the 1-dimensional movement in early stages of the search is a very important aspect since it allows in principle to find (in the set based context and given a suitable selection mechanism) and spread the solutions. In this case for finding multiple connected components. Figure 2.7 depicts such a scenario for $(\mathbf{x}, \lambda) = (1, -1, 0.5)^T$ and $r_{\mathbf{x}} = r_\lambda = 0.2$.

2.3.4 Evolutionary Techniques to Tackle PMOPs

Many optimization approaches have been used for the treatment of PMOPs, among all of them the evolutionary approaches have proven to be suitable for finding an approximation of the set of interest (family of Pareto sets and fronts) of a given



(a) Decision space (b) Objective space

Figure 2.7: SLS for a point that is 'in between' using $\nu_\lambda \neq 0$.

PMOP (see [9]). In general, the evolutionary algorithms to tackle PMOPs include one or more of the following mechanisms to produce an approximation of the solution set : multi-population schemes ([60]) , co-evolution strategies ([60]), change detection mechanisms ([61–64]), parallel approaches ([65]), diversity based techniques ([61, 66,67]), predictive techniques ([68–70]), and so on. Most of these approaches react to the changes of the external parameter, then when a change is detected the algorithms apply a certain mechanism such as a re-initialization process, in order to compute the new solution set. Algorithm 3 shows a general MOEA framework to tackle a given PMOP in a sequential way. The basic idea of this general framework is to select a given MOEA to evolve a certain population toward the solution set while there is no change detection. In the case that the algorithm detects a change of the external parameter λ it stores the last population and it applies a certain mechanism such as maintain memory, MOEA parameter tuning or a reinitialization process to start once again the optimization process. The process is repeated until the discretized space is covered.

Nevertheless, none of the algorithms which use Algorithm 3 as foundation, considers to treat a PMOP as a whole, instead of waiting for possible changes. In this particular case, we assume a certain number of expected changes during the optimization process, and also the interest of the decision maker over every value of λ. There exists a strong relation of the PMOPs and real world applications that is why

Algorithm 3 General scheme to tackle a given PMOP in a sequential way

```
Ensure: A final approximation $P^{(final)}$ of the family of Pareto sets and fronts of a
    given PMOP.
1: Set $\Lambda := \{\lambda_1, \ldots, \lambda_s\}$.
2: Set $m := 1$, i.e., consider (MOP) for $\lambda := \lambda_m = \lambda_1$.
3: Set $t := 0$.
4: Initialize $P_{m,t} \subset \mathbf{R}^n$.
5: while $m \leq s$ do
6:     if $change()$ then
7:         $P_m^{(final)} := P_{m,t}$
8:         $m := m + 1$
9:         $t := 0$
10:        Maintain memory, tune the MOEA parameters and/or reinitialize the
11:        population $P_{m,t}$.
12:    else
13:        Compute $P_{m,t+1}$ from $P_{m,t}$ using a MOEA for one generation.
14:        $t := t + 1$
15:    end if
16: end while
17: return $P^{(final)} := \{P_1^{(final)}, ..., P_s^{(final)}\}$
```

some approaches have also been adapted to this context such as [71–80].

2.4 Memetic Algorithms

An important drawback of the multi-objective evolutionary algorithms is their rate of convergence, which is typically slow. That is why, the idea of exploiting all available knowledge about a problem is a way to accelerate the search process [81].

Memetic algorithms (MAs) were introduced by Moscato [82] in 1989, who took the roots of this term from the word 'meme' presented by Dawkins in [83]. Speaking now about evolutionary algorithms, the term refers to improve certain elements into the population with a local search mechanism.

MAs are defined as population based approaches composed by an evolutionary computing framework and a set of local search algorithms. The local search strategy is triggered when over the execution of the evolutionary computing framework. The local search strategies improve in this case one or some elements during the global search procedure performed by the MOEA. The combination between a local strategy and a global search using MOEAs is called memetic evolutionary algorithm. Many researchers have reported successful results by the hybridization of the MOEAs. In the recent years different local search techniques have been designed for improving the performance of some evolutionary algorithms by using first or second order derivative methods (e.g. see [26,28,84,85]). Nevertheless, in some cases there is no clear objective of what they are supposed to improve. The integration of an indicator can steer the local search in better directions and also be more effective for improving points.

2.5 Performance Indicators for MOPs and PMOPs

In order to assess the produced outcome set by a MOEA some performance indicators have been proposed. In the following, we present several important indicators for assessing approximation sets produced when solving a given MOP or PMOP.

2.5.1 Hypervolume Indicator

The so-called *dominated hypervolume* (or S-metric) of a population is a commonly accepted indicator [2] for assessing the quality of an approximation. This indicator refers to the size of the space covered or dominated. The hypervolume is described as the Lebesgue measure $\mathcal{L}$ of the union of hypercubes defined by a non-dominated point $\mathbf{y}^{(i)}$ and a reference point $\mathbf{r}$ expressed as $\bigcup[v^{(i)}, \mathbf{r}]$.

Definition 7. *Let $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(\mu)} \in \mathbb{R}^k$ be a non-dominated set and $\mathbf{r} \in \mathbb{R}^k$ such that $\mathbf{y}^{(i)} \prec R$ for all $i = 1, \ldots, \mu$. The value*

$$\mathcal{H}(\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(\mu)}; \mathbf{r}) = \mathcal{L}\left(\bigcup_{i=1}^{\mu}[\mathbf{y}^{(i)}, \mathbf{r}]\right) \tag{2.72}$$

is termed the dominated hypervolume *with respect to* reference point $\mathbf{r}$, *where $\mathcal{L}(\cdot)$ denotes the Lebesgue measure in $\mathbb{R}^k$.*

This measure has a number of appealing properties but determining its value is getting the more computational expensive the larger the number of objectives is considered [86]. The hypervolume has some advantages which are presented in the following:

- If $\mathbf{r}$ is given, $\mathcal{H}(A)$ can be computed without further knowledge,
- $\mathcal{H}$ is Pareto compliant. That is if $(b < a)$ and $A_1 = \{a, a_2, \ldots, a_j\}$ and $A_2 = \{b, a_2, \ldots, a_j\}$, it follows that $\mathcal{H}(A_2) \geq \mathcal{H}(A_1)$ [22].

On the other hand, also this indicator has some disadvantages such as:

- $\mathcal{H}$ is dependent on the choice of $\mathbf{r}$,
- the computation of $\mathcal{H}(A)$ for $k > 2$ is time consuming, since the complexity was estimated as $O(n^{k+1})$ with n being the number of variables in parameter space and k the number of objectives [87].
- the final distribution according to $\mathcal{H}$ is not evenly distributed along the Pareto front [88].

2.5.2 Hausdorff Distance

The Hausdorff distance [3] can also be used as a performance indicator for MOPs. The indicator is described in the following four steps:

(a) The distance between two points $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ is defined as

$$\text{dist}(\mathbf{a}, \mathbf{b}) := ||\mathbf{a} - \mathbf{b}||, \tag{2.73}$$

where $|| \cdot ||$ is a norm on $\mathbb{R}^n$.

(b) The distance between a point $\mathbf{b} \in \mathbb{R}^n$ and a set $A \subset \mathbb{R}^n$ is defined as

$$\text{dist}(\mathbf{b}, A) := \inf_{\mathbf{a} \in A} ||\mathbf{b} - \mathbf{a}||. \tag{2.74}$$

(c) The semi-distance between two sets A, $B \subset \mathbb{R}^n$ is defined as

$$\mathrm{dist}(B, A) = \sup_{\mathbf{b}\in B} \mathrm{dist}(\mathbf{b}, A) = \inf_{\mathbf{a}\in A} \sup_{\mathbf{b}\in B} ||\mathbf{a} - \mathbf{b}||. \tag{2.75}$$

Note that $\mathrm{dist}(A, B)$ does not have to be equal to $\mathrm{dist}(B, A)$, i.e., dist is not symmetric. As example consider $A \subset B$, then $\mathrm{dist}(A, B) = 0$ but not necessarily $\mathrm{dist}(B, A) = 0$.

(d) The Hausdorff distance between A, $B \subset \mathbb{R}^n$

$$d_H(A, B) = \max(\mathrm{dist}(A, B), \mathrm{dist}(B, A))$$

$$= \max(\inf_{\mathbf{a}\in A} \sup_{\mathbf{b}\in B} ||\mathbf{a} - \mathbf{b}||, \inf_{\mathbf{b}\in B} \sup_{\mathbf{a}\in A} ||\mathbf{b} - \mathbf{a}||). \tag{2.76}$$

One of its principal advantages is that d_H defines a metric on the set of compact subsets. However, it is important to mention that d_H is sensitive to single outliers. Such outliers can be generated when using stochastic method such as MOEAs for the approximation of the Pareto set/front.

2.5.3 Generational Distance and Inverted Generational Distance

The Generational Distance (GD) introduced by Van Veldhuizen and Lamont [4] is a value representing how 'far' an approximation A of the Pareto front A is from the true Pareto front P_f. The indicator is defined as follows:

Let

$$A = \{\mathbf{a}_1, \ldots, \mathbf{a}_j\} \tag{2.77}$$

be an archive with j vectors $\mathbf{a}_i \in \mathbb{R}^n$, then

$$\mathrm{GD}(A, P_f) := \frac{1}{j}(\sum_{i=1}^{j} d_i^p)^{1/p}, \tag{2.78}$$

where $p = 2$ and d_i is the Euclidean distance (in objective space) between the image of $\mathbf{a}_i$ and the nearest member of the true Pareto front.

In [5], the Inverted Generational Distance (IGD) is proposed. IGD uses as a reference a discretization of the true Pareto front P_f, and compares each of its elements with respect to the approximation A produced by an algorithm. This intends to reduce some of the problems that occur with the generational distance metric when an algorithm generates very few non-dominated solutions. IGD is defined as follows:

Let $\{\mathbf{y}_1, \dots, \mathbf{y}_z\}$ be a discretization of the Pareto front, $z \in \mathbb{N}$, and A is defined as in Equation (2.77), then

$$\text{IGD}(A, P_f) := \frac{1}{z}(\sum_{i=1}^{z} d_i^p)^{1/p}, \tag{2.79}$$

where d_i denotes the Euclidean distance from $\mathbf{y}_i$ to A.

The disadvantage of GD and IGD lies over the fact that if we increase j or z according Equations (2.78) and (2.79) the approximation appears to be 'better', although the approximation has apparently not change. For more information, we refer to [6].

2.5.4 Averaged Hausdorff Distance

In [6] , a new performance indicator Δ_p is proposed, which can be seen as an 'averaged Hausdorff distance' (AHD) between the image of A and the true Pareto front and the approximation. This new metric is composed by the indicators GD and IGD but with slightly differences that we describe below:

Given a candidate set $A = \{\mathbf{a}_1, \dots, \mathbf{a}_j\}$ and a discretization of the Pareto front $P_f = \{\mathbf{y}_1, \dots, \mathbf{y}_z\}$, then

$$\text{GD}_p(A, P_f) := \left(\frac{1}{j} \sum_{i=1}^{j} d_i^p \right)^{1/p} \tag{2.80}$$

is the averaged semi-distance from the image of A to the discretization of the Pareto front and

$$\text{IGD}_p(A, P_f) := \left(\frac{1}{z} \sum_{i=1}^{z} d_i^p \right)^{1/p} \tag{2.81}$$

is the averaged semi-distance of the $\mathbf{y}_i$'s to the image of A. Finally, we define Δ_p as:

$$\Delta_p := \max(GD_p(A), IGD_p(A)). \tag{2.82}$$

Hence, it holds $\Delta_\infty = d_H$ and Δ_p with $p < \infty$ can be considered as an averaged the Hausdorff distance.

2.5.5 Performance Indicators for PMOPs

As we mentioned, for every fixed value of λ the problem stated in Equation (2.14) can be seen as a classical MOP, in that case, it is possible to use a performance indicator for MOPs to assess the produced approximations. However, instead of having just one scalar value, we will have a set of them depending on s which is the number of values in the discretization of Λ. In order to avoid this, some performance indicators have been proposed and adapted to the context of PMOPs (see [7–9] and the references there in for more information).

2.6 Local Search Strategies based on Performance Indicators

In general, most of the local search strategies integrated mathematical programming techniques for improving the selected points for local search, however, the improvement not always ensures to have a better value according to a performance indicator. In the following, we introduce two local search strategies that aim for improving points according the hypervolume indicator.

2.6.1 Bi-objective HVDS

In [89], the bi-objective hypervolume based directed search method (HVDS) was presented. This algorithm considers a three-region division of the objective space to produce the best improvement for a given candidate solution according to the hypervolume.

In order to assign a point $\mathbf{x}$ into one of the proposed three regions, the algorithm uses some properties of the descent cone of a MOP stated in [1]:

1. It has been observed in [1] that if $\mathbf{x}$ is 'far away' from the Pareto set, then the objective gradients nearly point to the same direction.
2. If $\mathbf{x}$ is 'near' or even in the Pareto set, then the objectives gradients point in opposite directions [1]. Figure 2.8 shows a graphical interpretation of the

previous described properties.

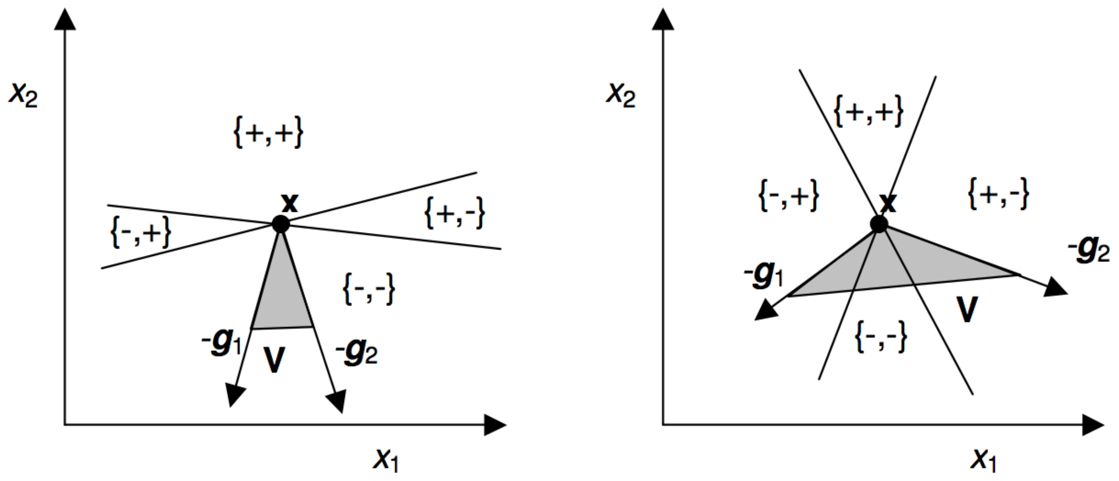


Figure 2.8: Objective gradients and descent cones when $\mathbf{x}$ is 'far away' (left) and when it is 'near' (right) [1].

Therefore, for a bi-objective optimization problem, the method can decide where a point is located within the region division considering the angle between gradients. Let

$$g_i := \nabla f_i(\mathbf{x}),\ i = 1, 2\ , \tag{2.83}$$

then the angle between g_1 and g_2 is defined by

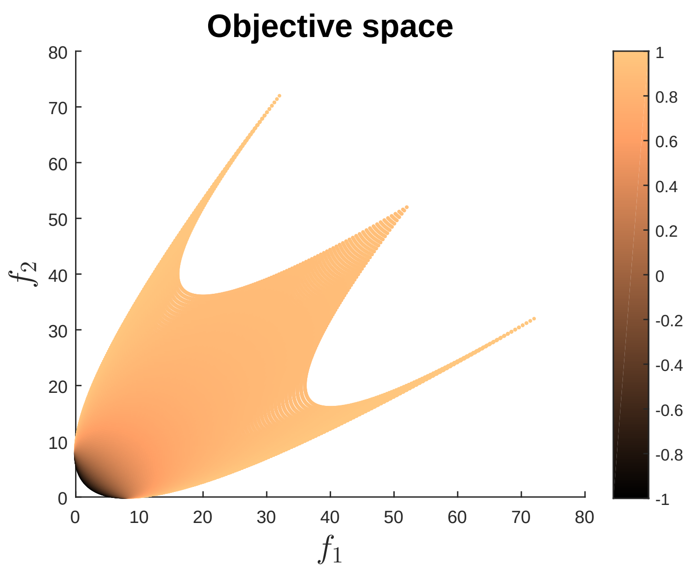
$$\cos\alpha = \frac{g_1^T g_2}{||g_1||||g_2||} \in [-1, 1]. \tag{2.84}$$

The bi-objective HVDS divides the search space into three numerically detected distance regions by setting two values $a, b \in (-1, 1)$ with $b < a$ where:

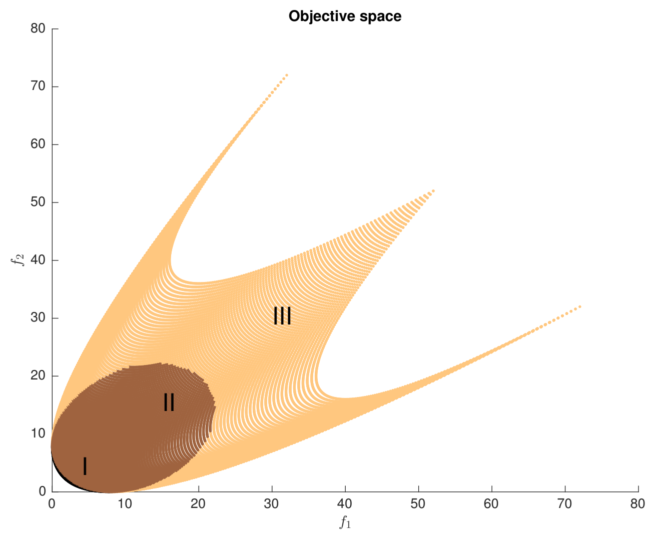
$$\begin{aligned} \mathbf{x} \in I &: \Leftrightarrow \cos\alpha \geq a, \\ \mathbf{x} \in II &: \Leftrightarrow \cos\alpha \in (b, a), \\ \mathbf{x} \in III &: \Leftrightarrow \cos\alpha \leq b. \end{aligned} \tag{2.85}$$

Figure 2.9(a) shows the distribution of $\cos\alpha$ values and Figure 2.9(b) a possible region division of the objective space using using $a = 0.8$ and $b = -0.8$.

Once a point $\mathbf{x}$ is assigned into one of the three regions the local search technique applies one of the following movements:



(a) Values of $\cos \alpha$



(b) Region division

Figure 2.9: Values of $\cos \alpha$ and possible region division for a bi-objective optimization problem.

- Local search in Region I: the algorithm exploits the fact that larger improvements can be done, since both gradients point into the same direction. Then, the Averaged Descent Direction is used coupled with an Armijo-like step size control to move the candidate solution toward the solution set [90].

- Local search in Region II: in the case that the HVDS is applied to one single point the directed search method is used to produce a movement toward the Pareto front by using the following direction in objective space

$$d_{II} = \mathbf{f}(\mathbf{x}) - \mathbf{r}, \tag{2.86}$$

  where $\mathbf{f}(\mathbf{x})$ represents the objective values of $\mathbf{x}$ and $\mathbf{r}$ the selected reference point to compute the hypervolume value. On the other hand, when more than one candidate point is considered to apply the HVDS, they have to be sorted according the value of f_1 in ascending order. Then, the reference point $r_{\mathbf{f}(\mathbf{x}_i)}$ is constructed by

$$\mathbf{r}_{\mathbf{x}_i} = \begin{pmatrix} f_1(\mathbf{x}_{i+1}) \\ f_2(\mathbf{x}_{i-1}) \end{pmatrix}. \tag{2.87}$$

  Hence, for intermediate points ($i \in \{2, ..., l-1\}$), where l is the number of candidate solutions, the direction used in the directed search method is given by

$$d_{II,\mathbf{x}_i} = \mathbf{f}(\mathbf{x}_i) - \mathbf{r}_{\mathbf{x}_i}. \tag{2.88}$$

  Figure 2.10 shows graphically the construction of the reference point for intermediate points.

  For extreme points i.e., $i \in \{1, l\}$, the algorithm proceed as the single point case using Equation (2.86), where the reference point is the one given to measure the hypervolume indicator.

- Local search in Region III: when a candidate point $\mathbf{x}$ is already near to the Pareto front no big improvements can be reached anymore toward this set, however, movements along it could increase the dominated hypervolume. One of the most important advantages of the DS is the possibility to performed a movement in any given direction d. So, by using the DS a movement along the Pareto front can be performed by solving the minimization problem stated in Equation (2.39) to set d_{III} as follows:

$$d_{III} = \begin{pmatrix} -\tilde{a}_1 \\ -\tilde{a}_2 \end{pmatrix}, \tag{2.89}$$

  where $\tilde{\alpha}$ is the solution of the problem stated in Equation (2.39).

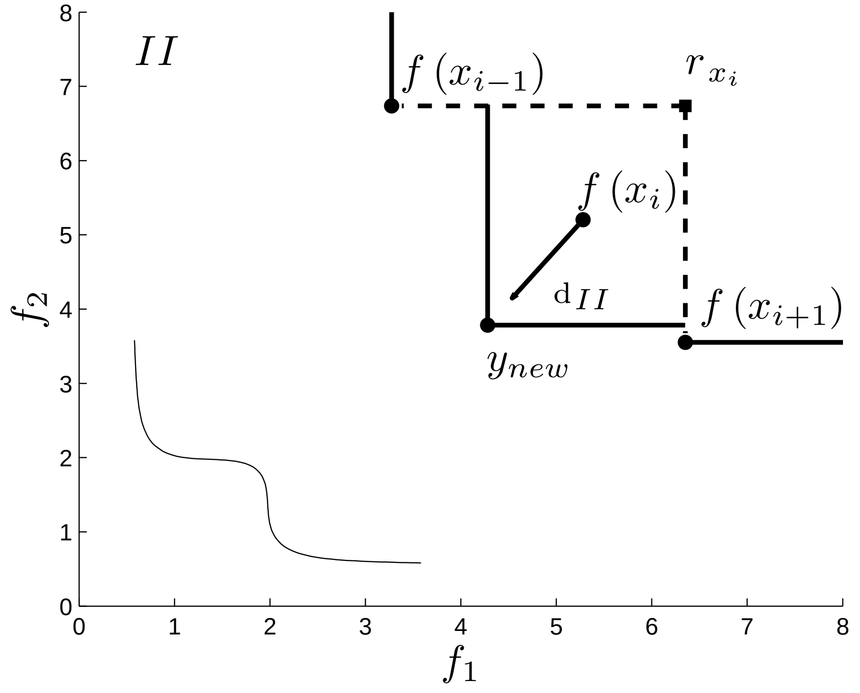


Figure 2.10: Local search in Region II for multiple archive entries.

The three movements are depicted in Figure 2.11 throughout the whole region division in objective space.

The integration of the bi-objective HVDS into a MOEA yields a fast and reliable memetic algorithm to obtain hypervolume approximations of a given MOP. The MOEA used to be coupled with the bi-objective HVDS was the SMS-EMOA an state-of-the-art algorithm. This first attempt only applied the HVDS at the beginning of the optimization process for a certain number of iterations. To be more precise, l elements are injected into the initial population of the SMS-EMOA, where these l elements were produced by the HVDS. The l elements are, in most of the cases, 'near' to the solution set, what makes the algorithm converge faster to the Pareto front. Numerical results showed the advantage of integrating the HVDS into the MOEAs. However, some possible improvements can be done to increase the applicability of the method.

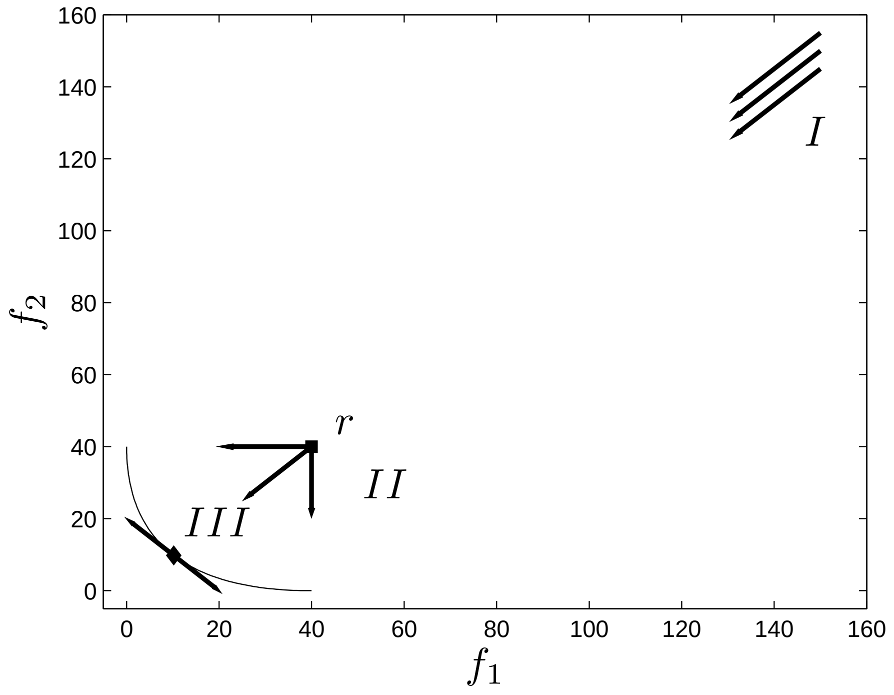


Figure 2.11: Division of the objective space into distance regions.

3 | The Hypervolume based Directed Search Method

During the last years, specialized MOEAs have performed well for the treatment of a wide range of MOPs. The application of such techniques produces a reliable approximation of the solution set of a given MOP. Further, algorithms of this type compute such an approximation in one single run. However, MOEAs present a clear disadvantage, namely they tend to converge slowly causing the use of a high number of function evaluations when the approximation is computed. In order to reduce the number of function evaluations used within the optimization process, researchers have proposed memetic (or hybrid) strategies [82, 85, 91–96]. Techniques of this type hybridize MOEAs using local search strategies to obtain a fast and reliable approximation of the set of interest (Pareto set and front) of a MOP.

MOEAs have implemented different mechanisms to perform the selection process during evolution. One of these mechanisms is the integration of a performance indicator [97–99]. As we mentioned in Chapter 2, a performance indicator measures the quality of an approximation of the Pareto front. Furthermore, it transforms the MOP into a SOP, since the problem is reduced to minimize (or maximize) the scalar value given by evaluating the performance indicator over the approximation. The MOEAs based on a performance indicator aim to generate a set of solutions such that maximize (or minimize) the indicator value. The hypervolume indicator has been widely used within this context, for instance, the SMS-EMOA, known as a powerful state-of-the-art MOEA, maximizes this indicator. The indicator has some desirable properties (see e.g. [100–103]) for instance strict monotonicity and Pareto compliance. On the other hand, approximations obtained by using the hypervolume prefer points in the knee regions for a convex Pareto front and for concave ones the points are accumulated in the extremes [88]. Also, a high computational effort is required when computing the hypervolume value as the number of objectives is increased.

In this chapter, we introduce the Hypervolume based Directed Search Method (HVDS) for general number of objectives. The HVDS is a local search technique that aims to maximize the hypervolume. The proposed technique includes a region division of the objective space applicable to MOPs where their objectives are differentiable. Such a division allows to perform a specific local search procedure in every region according to the position of the candidate solution $\mathbf{x}$. We state different regions according the proximity of the candidate solution $\mathbf{x}$ to the Pareto front: 'far away', 'in between', and 'near'. The integration of the Directed Search method into the local search procedure makes possible to performed movements in any direction d given in objective space. We present the HVDS both as standalone algorithm and as local search engine within SMS-EMOA leading to the hybrid SMS-EMOA-HVDS. We present numerical experiments on several benchmark MOPs with two and three objectives to show the benefit of integrating the HVDS into a MOEA that aims for hypervolume maximization.

3.1 Motivation

The motivation of the proposed local search strategy comes as follows: in most of the cases, we can assume that a given point $\mathbf{x} \in S$ assigned for local search is 'far away' from the Pareto set which happens in early stages of the search process. Then, it is desired for the new iteration $\mathbf{x}_{new}$ to get approached as fast as possible to the set of interest (as a whole) while the position of $\mathbf{x}_{new}$ over this set is rather secondary. On the contrary, the following discussion shows that a steering of the search (e.g. via Directed Search) might be even hindering the overall performance: as we mention in Chapter 2 the work of Brown and Smith [1] stated that the objectives' gradients of a given MOP typically point toward similar directions if $\mathbf{x}$ is 'far away' from the Pareto set (if existing). Consider for simplicity the extreme case, namely that all gradients point into the same direction, i.e., let

$$\nabla f_i(\mathbf{x}) = \vartheta_i \nabla f_1(\mathbf{x}), \quad i = 1, \ldots, k, \tag{3.1}$$

where $\vartheta_i > 0$, $i = 1, \ldots, k$. If $\mathbf{x}_{new}$ is obtained via line search it can be written as

$$\mathbf{x}_{new} = \mathbf{x} + t\nu, \tag{3.2}$$

where $t \in \mathbb{R}_+$ is the step size and $\nu \in \mathbb{R}^n$ the chosen search direction. Note that ν defines the movement in decision space, and that the related movement in objective space is given by $J(\mathbf{x})\nu$ for infinitesimal small step sizes (this can be seen from Equation (2.41)). Thus, in case (3.1) holds, we have

$$J(\mathbf{x})\nu = \begin{pmatrix} \nabla f_1(\mathbf{x})^T \nu \\ \ldots \\ \nabla f_k(\mathbf{x})^T \nu \end{pmatrix} = \nabla f_1(\mathbf{x})^T \nu \cdot \vartheta, \tag{3.3}$$

where $\vartheta = (\vartheta_1, \ldots, \vartheta_k)^T \in \mathbb{R}^k$. Note that the vector in (3.3) is a multiple of ϑ, i.e., we obtain a one-dimensional movement regardless of the choice of ν which is n-dimensional. Though (3.1) and thus (3.3) is apparently only true if the distance of $\mathbf{x}$ to the Pareto set is infinite, the range of directions $\mathbf{d} = J(\mathbf{x})\nu$ in which the search can be steered with a satisfactory step size is expected to be small for points $\mathbf{x}$ that are sufficiently far away from the Pareto set. We refer to [28] for a more thorough discussion. Thus, rather than using DS (or any other steering mechanism) it seems to be wise to perform a greedy search toward the Pareto set.
Second, we consider the other extreme situation, namely that $\mathbf{x}$ is already nearly optimal which happens in later stages of the search process. In that case no further significant improvements can be expected when searching along descent directions. Instead, it seems to be wise to perform the search along the Pareto front.
Finally, points do apparently not have to be 'far away' nor 'near' to the Pareto set. In that case we suggest to search along descent directions that (locally) improve the hypervolume value.

Similar than the bi-objective HVDS, we suggest to divide the decision space S into three regions and then, we define them and proposed the best fit movement for each region:

- **Region** I $\mathbf{x}$ is 'far away' from the Pareto set (denoted by '$\mathbf{x} \in I$'). In that case, a greedy search toward the rough location of the Pareto front is desired.

- **Region** II $\mathbf{x}$ is 'in between', i.e., neither 'far away' nor 'near' to the Pareto front. In that case, a descent direction has to be selected such that a movement in that direction maximizes the hypervolume.

- **Region** III $\mathbf{x}$ is 'near' to the Pareto set. In that case, a search *along* the Pareto front will be performed.

The region division remains the same for the bi-objective HVDS, however, in that first version the mechanism to divide into three regions is restricted to two objectives ($k = 2$). In the following, we propose a new way to realize the region division for general number of objectives and further, we discuss the local search in each region.

3.2 Region Division

In [89], it was proposed to use the angle between gradients to realize the region division, nevertheless, this mechanism only can be applied when $k = 2$. Here, a new strategy is presented in order to perform this region division.

Assume we are given an unconstrained MOP with all its objectives differentiable. Let

$$R_S := \{\mathbf{x} \in S \ : \ \nabla f_i(\mathbf{x}) \neq 0, \ i = 1, \ldots, k\} \tag{3.4}$$

and define $\eta : R_S \to \mathbb{R}$ via

$$\eta(\mathbf{x}) = \left\| \sum_{i=1}^{k} \tilde{\alpha}_i \frac{\nabla f_i(\mathbf{x})}{||\nabla f_i(\mathbf{x})||_2} \right\|_2^2, \tag{3.5}$$

where $\tilde{\alpha}$ is a solution of the following optimization problem

$$\min_{\alpha \in \mathbb{R}} \left\{ \left\| \sum_{i=1}^{k} \alpha_i \frac{\nabla f_i(\mathbf{x})}{||\nabla f_i(\mathbf{x})||_2} \right\|_2^2 \ : \ \alpha_i \geq 0, \ i = 1, \ldots, k, \ \sum_{i=1}^{k} \alpha_i = 1 \right\}. \tag{3.6}$$

The following result is the basis for the region division.

Proposition 10. *Let all objectives f_i, $i = 1, \dots, k$, be continuously differentiable in $\mathbf{x} \in R_S$, η be as in Equation (3.5) and $\mathbf{x} \in R_S$. Then*

(a) $\eta(\mathbf{x}) \in [0, 1]$

(b) $\eta(\mathbf{x}) = 1 \;\Leftrightarrow\; \nabla f_i(\mathbf{x}) = \lambda_i \nabla f_1(\mathbf{x})$ *with* $\lambda_i > 0$ *for* $i = 1, \dots, k$.

(c) $\eta(\mathbf{x}) = 0 \;\;\Leftrightarrow \mathbf{x} \in R_S$ *is a Karush-Kuhn-Tucker (KKT) point.*

(d) $\eta(\mathbf{x})$ *is a continuous mapping.*

Proof. (a) Clearly, $\eta(\mathbf{x}) \geq 0$. Further, it is

$$\eta(\mathbf{x}) = \left\| \sum_{i=1}^{k} \tilde{\alpha}_i \frac{\nabla f_i(\mathbf{x})}{||\nabla f_i(\mathbf{x})||_2} \right\|_2^2 \leq \sum_{i=1}^{k} \tilde{\alpha}_i \left\| \frac{\nabla f_i(\mathbf{x})}{||\nabla f_i(\mathbf{x})||_2} \right\|_2^2 = \sum_{i=1}^{k} \tilde{\alpha}_i = 1. \tag{3.7}$$

(b) "$\Rightarrow$": by strict convexity of the function in Equation (3.6) it follows directly that

$$\frac{\nabla f_1(\mathbf{x})}{\|\nabla f_1(\mathbf{x})\|_2} = \dots = \frac{\nabla f_k(\mathbf{x})}{\|\nabla f_k(\mathbf{x})\|_2} \tag{3.8}$$

which is equivalent to Equation (3.1).

"$\Leftarrow$": let α be a convex weight, then

$$\sum_{i=1}^{k} \alpha_i \frac{\nabla f_i(\mathbf{x})}{||\nabla f_i(\mathbf{x})||_2} = \sum_{i=1}^{k} \alpha_i \frac{\lambda_i \nabla f_1(\mathbf{x})}{||\lambda_i \nabla f_1(\mathbf{x})||_2} = \sum_{i=1}^{k} \alpha_i \frac{\nabla f_1(\mathbf{x})}{||\nabla f_1(\mathbf{x})||_2} = \frac{\nabla f_1(\mathbf{x})}{||\nabla f_1(\mathbf{x})||_2} \tag{3.9}$$

by which it follows that $\eta(x) = 1$.

(c) "$\Rightarrow$": let $\eta(\mathbf{x}) = 0$, then there exists a convex weight α s.t.

$$\sum_{i=1}^{k} \alpha_i \frac{\nabla f_i(\mathbf{x})}{||\nabla f_i(\mathbf{x})||_2} = \sum_{i=1}^{k} \frac{\alpha_i}{||\nabla f_i(\mathbf{x})||_2} \nabla f_i(\mathbf{x}) = 0. \tag{3.10}$$

Thus, choosing $\tilde{w}_i := \alpha_i / \|\nabla f_i(\mathbf{x})\|_2$, $i = 1, \dots, k$, the vector $w := \tilde{w} / \|\tilde{w}\|_1$ is a convex weight with $\sum_{i=1}^{k} w_i \nabla f_i(\mathbf{x}) = 0$, i.e., $\mathbf{x}$ is a KKT point.

"$\Leftarrow$": now let $\mathbf{x} \in R_S$ be a KKT point. That is, there exists a convex weight α s.t.

$$\sum_{i=1}^{k} \alpha_i \nabla f_i(\mathbf{x}) = \sum_{i=1}^{k} \alpha_i \|\nabla f_i(\mathbf{x})\|_2 \frac{\nabla f_i(\mathbf{x})}{||\nabla f_i(\mathbf{x})||_2}. \tag{3.11}$$

Choosing $\tilde{w}_i := \alpha_i \|\nabla f_i(\mathbf{x})\|_2$, $i = 1, \dots, k$, then $w := \tilde{w} / \|\tilde{w}\|_1$ is a convex weight such that $\sum_{i=1}^{k} w_i \nabla f_i(\mathbf{x}) / \|\nabla f_i(\mathbf{x})\|_2 = 0$, i.e., $\eta(\mathbf{x}) = 0$.

(d) follows by construction of η.

□

Thus, by the above result and using the observation made in [1] we can conclude that (i) $\mathbf{x}$ is far away from the Pareto set iff $\eta(\mathbf{x})$ is close to 1, (ii) $\mathbf{x}$ is near to the Pareto set iff $\eta(\mathbf{x})$ is close to 0, and (iii) else $\mathbf{x}$ is in between. Hence, by determining two values $a, b \in (0,1)$ with $a < b$ we can make the region division via

$$\begin{aligned} \mathbf{x} \in I \Leftrightarrow &\ \eta(\mathbf{x}) \geq b \\ \mathbf{x} \in II \Leftrightarrow &\ \eta(\mathbf{x}) \in (a,b) \\ \mathbf{x} \in III \Leftrightarrow &\ \eta(\mathbf{x}) \leq a. \end{aligned} \tag{3.12}$$

As example consider the three objective problem MOP3

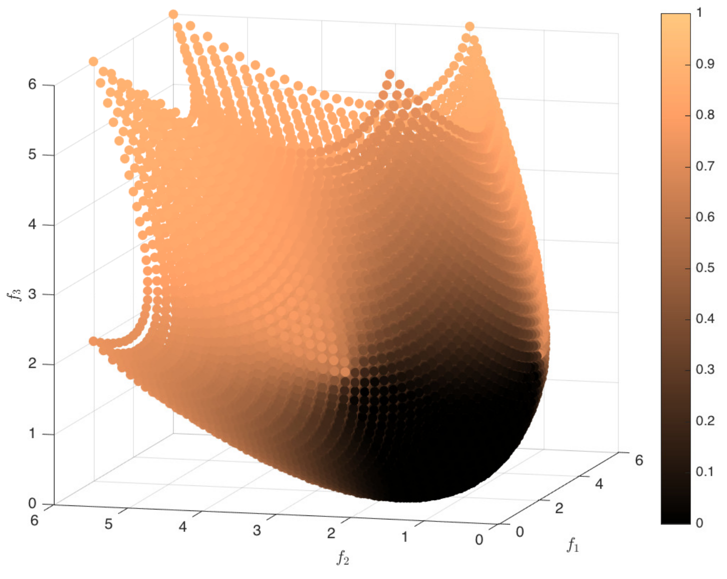
$$\begin{aligned} f_1, f_2, f_3 &: \mathbb{R}^3 \to \mathbb{R} \\ f_i(\mathbf{x}) &= \sum_{j=1}^{3} (x_j - a_j^i)^2, \end{aligned} \tag{3.13}$$

where $a^1 = (1,0,0)^T$, $a^2 = (0,1,0)^T$ and $a^3 = (0,0,1)^T$. Figure 3.1(a) shows the value of η (in objective space for a better visualization) and Figure 3.1(b) a possible division of the image space into the three regions. For the region division we have used $a = 0.1$ and $b = 0.5$. In the next sections a study to choose a and b is presented for a set of benchmark functions.

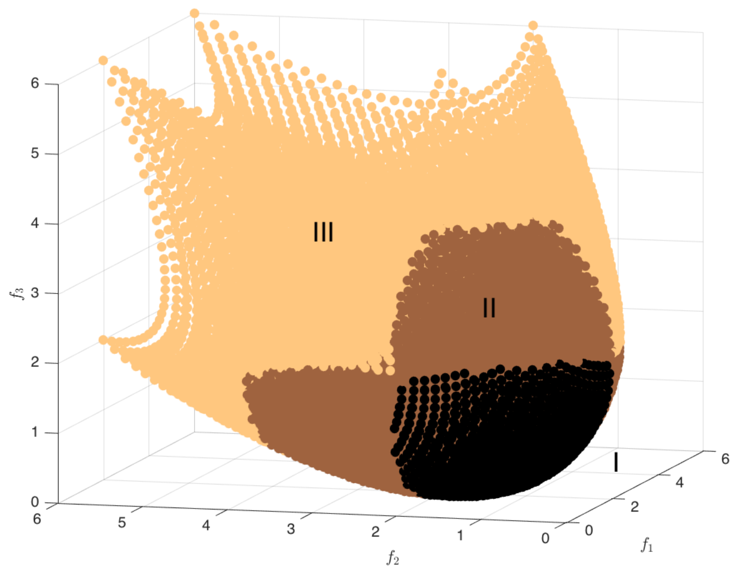
Remark 1. *For the important special case $k = 2$ (i.e., for bi-objective problems) one can simplify the region division by using alternatively the angle between the two gradients (see Chapter 2).*

3.2.1 Specialized Local Search for every Region Division

Once a given point $\mathbf{x}$ for local search is located through the region division, we propose the following search strategy for each of the three regions. First, we consider the extreme case an archive a with only one element ($|A| = 1$) to perform local search and then, we address the general archive case. The general archive represents a generalization of the movement applied to a single point.



(a) Values of η



(b) Region division

Figure 3.1: Values of η and possible region division for MOP (3.13).

One Element Archives

Given an archive $A = \{\mathbf{x}\}$ with one single element, where $\mathbf{x} \in S$ is assigned for local search and a reference point $\mathbf{r} = (r_1, \ldots, r_k)^T$ for measuring the hypervolume indicator, we proposed the following local search movements:

Local search in Region I Within this region explicit steering of the search is a delicate problem. Thus, it seems to be wise to utilize established multi-objective descent methods in order to approach the Pareto set/front as fast as possible. Such methods can be found e.g. in [24, 26, 104].

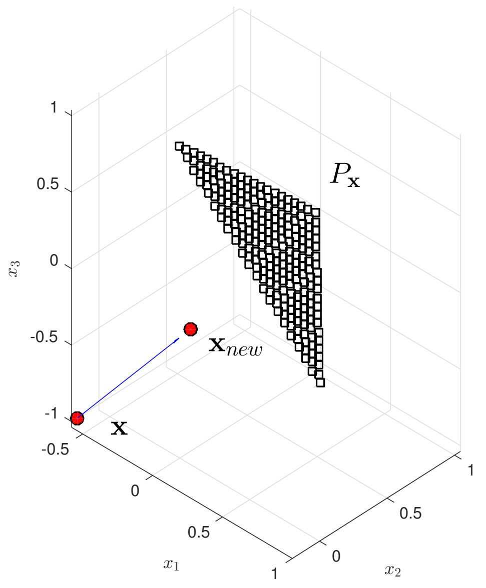
For the treatment of points within this region, we decide to incorporate the descent method proposed in [24] explained in Chapter 2. This method requires to solve an additional optimization problem to compute the search direction. Nevertheless, the computation of the Jacobian (J) of $\mathbf{x}$ is already done by the region division strategy, then, we can solve this problem without calculating once again J. Figure 3.2 shows the performed movement for a given point assigned for local search within Region I, this movement can be observed in both decision variable and objective space. In Region I, the reference point $\mathbf{r}$ is secondary, since it is more important to approach to the solution set as a whole. Then, the movement of a $\mathbf{x}$ is not affected by the position of $\mathbf{r}$.

Local search in Region II If $\mathbf{x}$ is in Region II, the task is to find a descent direction $d_{II} <_p 0$ (in objective space) such that a movement in that direction maximizes the hypervolume. Since the new iterate $\mathbf{x}_{new}$ is performed via DS, we can (idealized) write its image $\mathbf{y}_{new} = \mathbf{f}(\mathbf{x}_{new})$ as

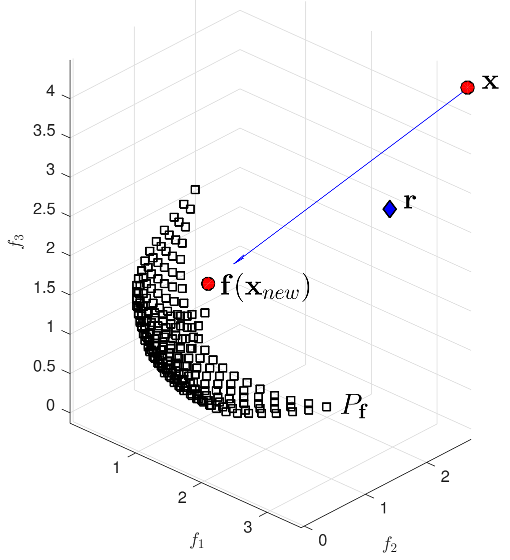
$$\mathbf{y}_{new} = \mathbf{f}(\mathbf{x}) + t\mathbf{d}_{II}, \tag{3.14}$$

where $t \in \mathbb{R}^+$ is a given (fixed) step size and d_{II} has to be chosen such that it solves the k-dimensional problem

$$\begin{aligned} \max_{\mathbf{d} \in \mathbb{R}^k} \mathcal{H}(\mathbf{d}) &= \prod_{i=1}^{k} (r_i - f_i(\mathbf{x}) - td_i) \\ \text{s.t. } ||\mathbf{d}||_2^2 &= 1, \end{aligned} \tag{3.15}$$



(a) Decision variable space



(b) Objective space

Figure 3.2: HVDS in Region I.

which is related to the maximization of the hypervolume (compare to Figure 3.3).

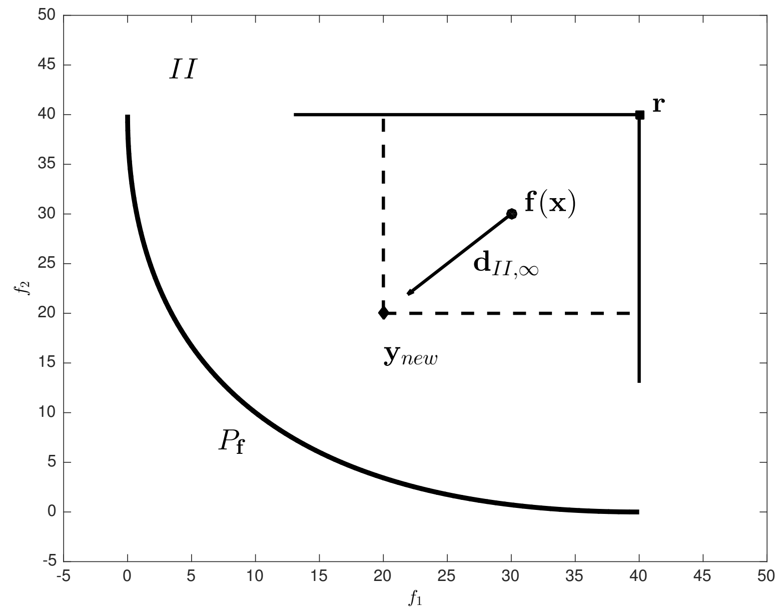
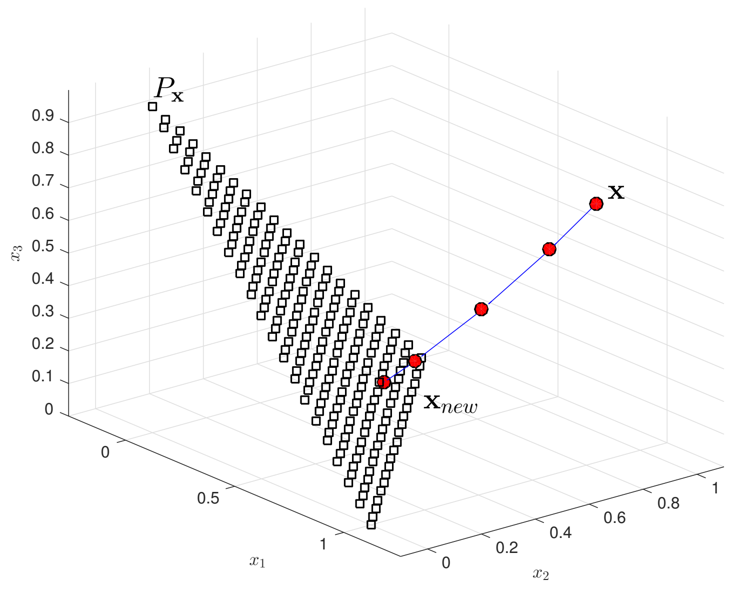


Figure 3.3: Local search in Region II.

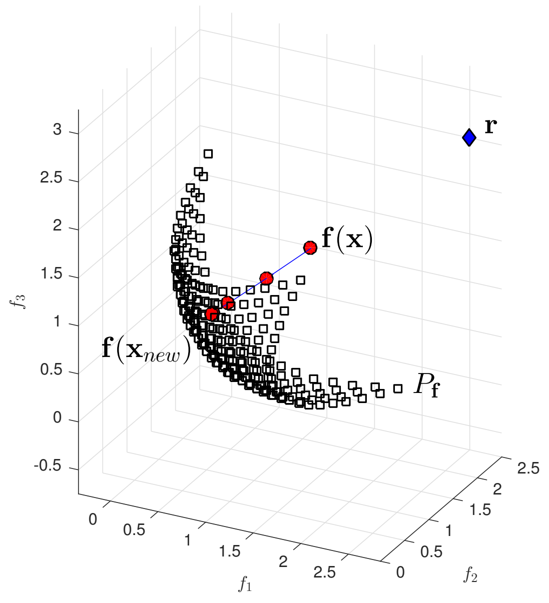
If one replaces the 2-norm by the infinity norm in the constraint of problem stated in Equation (3.15) (which drops the assumption that the movement is done with an equal step in objective space) a straightforward computation shows that

$$\mathbf{d}_{II,\infty} = \mathbf{f}(\mathbf{x}) - \mathbf{r} \tag{3.16}$$

solves the problem (where we assume that $\mathbf{f}(\mathbf{x}) <_p \mathbf{r}$). For our implementations we have used the direction in Equation (3.16) since it is easier to calculate and since we observed that it yields no difference in the performance of the algorithm compared to the solution of problem Equation (3.15). Note, however, that the solution of Equation (3.15) may come with additional CPU time but not with additional function evaluations. Figure 3.4 depicts the path followed by a given $\mathbf{x} \in I$ toward the new point $\mathbf{x}_{new}$. In this case, the reference point $\mathbf{r}$ is used to produce the direction $\mathbf{d}_{II,\infty}$ and then, the DS is applied to produced a sequence of points toward the set of interest.



(a) Decision variable space



(b) Objective space

Figure 3.4: HVDS in Region II.

Local search in Region III If $\mathbf{x}$ is nearly optimal, a search in a descent direction will not lead to significant improvements any more. Instead, it seems to be wise to continue the search along the Pareto set. Here, we suggest to perform a search via DS along the linearized Pareto front (see Figure 3.5). The particular steps of the search are described in the following for KKT points, the steps for nearly optimal solutions are the same.

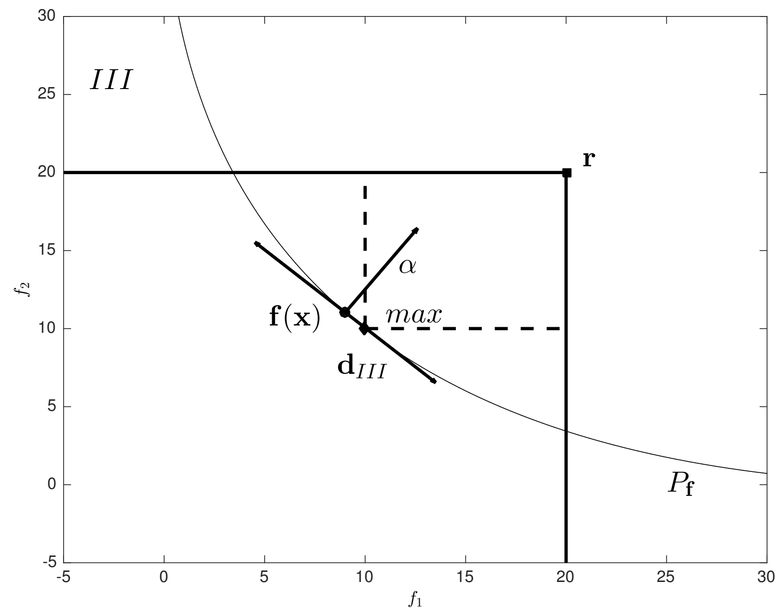


Figure 3.5: Movement along linearized Pareto front in order to improve the hypervolume in Region III.

(1) Compute the KKT weight α associated to α. This can be done by solving Problem (2.39).

(2) It is known that α is orthogonal to the linearized Pareto front at $F(x)$ [29]. Thus, an orthonormal basis of the tangent space can be obtained as follows: compute a QU-factorization of α, i.e.,

$$\alpha = QU = (\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_k)U. \tag{3.17}$$

Since $Q \in \mathbb{R}^{k \times k}$ is orthogonal, the vectors $q_2, \ldots, q_k$ span the desired tangent space and are thus desired search directions in objective space.

(3) Now we can maximize the hypervolume in objective space similar to (3.15). The differences here are that (i) we are restricting the search to directions $\mathbf{d} \in span(q_2, \dots, q_k)$ and (ii) the optimization will not only yield the optimal direction but also the optimal step size (i.e., $\mathbf{y}_{new}$ those corresponding decision vector $\mathbf{x}_{new}$ can be retrieved via DS). The $(k-1)$-dimensional unconstrained optimization problem is given by

$$\max_{\lambda \in \mathbb{R}^{k-1}} \mathcal{H}(\lambda) = \prod_{i=1}^{k} (r_i - f_i(\mathbf{x}) - (Q_2\lambda)_i), \tag{3.18}$$

where

$$Q_2 = (\mathbf{q}_2, \dots, \mathbf{q}_k) \in \mathbb{R}^{k \times k-1}. \tag{3.19}$$

The search direction is thus given by

$$d_{III} = Q_2\lambda^*, \tag{3.20}$$

where λ^* is a solution of Problem (3.18).

(4) Using DS we compute $\mathbf{x}_{new}$ such that $\mathbf{y}_{new} \approx \mathbf{f}(\mathbf{x}) + Q_2\lambda^*$. If the hypervolume value is not increased by $\mathbf{x}_{new}$ we perform the Armijo-like backtracking approach as in [90] using the hypervolume value as objective function.

For the special case $k = 2$ the solution of (3.18) can be expressed analytically if all the elements of the KKT weight are positive. Let $k = 2$ and $\alpha = (\alpha_1, \alpha_2)^T$ be the KKT weights associated to $\mathbf{x}$, then the tangent vector of the linearized Pareto front at $\mathbf{f}(\mathbf{x})$ is e.g. given by $\mathbf{d} = (-\alpha_2, \alpha_1)^T$ and problem (3.18) can be stated as

$$\max_{\lambda \in \mathbb{R}} \mathcal{H}_2(\lambda) = (r_1 - f_1(x) - \lambda d_1) \times (r_2 - f_2(x) - \lambda d_2). \tag{3.21}$$

Proposition 11. *Let $k = 2$ and $\alpha >_p 0$, then the global maximizer of Problem (3.21) is given by*

$$\lambda^* = \frac{d_1 r_2 + d_2 r_1 - d_1 f_2(x) - d_2 f_1(x)}{2 d_1 d_2}. \tag{3.22}$$

Proof. If $\alpha_p >_p 0$, then it follows that also $d_1, d_2 \neq 0$. The first derivative of $\mathcal{H}_2$ is given by

$$\mathcal{H}_2{}'(\lambda) = 2\lambda d_1 d_2 + d_2 f_1(x) - r_1 d_2 + d_1 f_2(x) - d_1 r_2 \tag{3.23}$$

Setting this to zero leads to

$$\lambda^* = \frac{d_1 r_2 + d_2 r_1 - d_1 f_2(x) - d_2 f_1(x)}{2 d_1 d_2}. \tag{3.24}$$

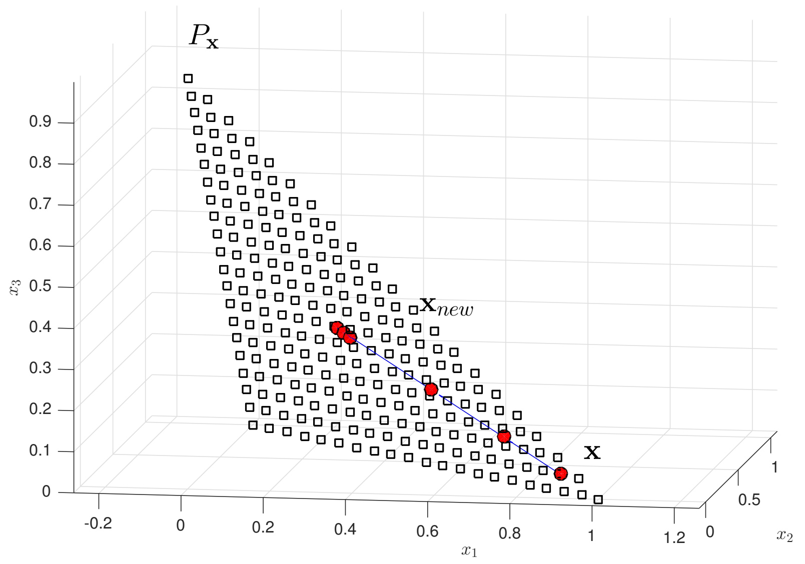
Further, the second derivative at λ^* is given by

$$vol_2\,''(\lambda^*) = 2 d_1 d_2 < 0. \tag{3.25}$$

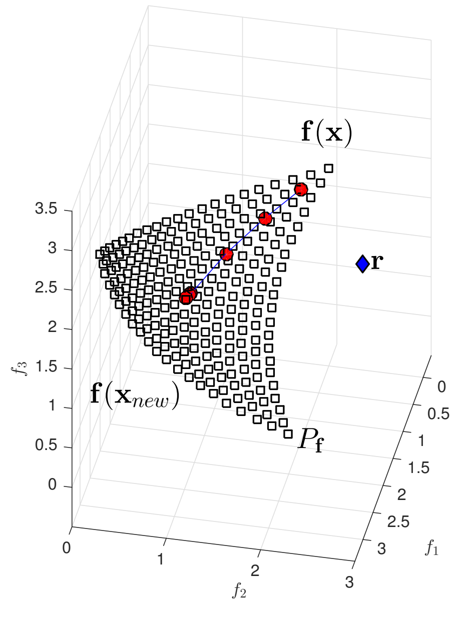
The negativity holds since $\alpha >_p 0$ and by construction of $\mathbf{d}$, and the claim follows. $\square$

Remark 2. *We stress that due to the linearization of the Pareto front at* $\mathbf{f}(\mathbf{x})$ *it can happen that the new iterate* $\mathbf{x}_{new}$ *is located in Region II leading to a kind of oscillating behavior in the sequence of iterates. We have, however, not observed any instabilities in our computations. Further, if* $\mathbf{x}$ *is already near to the optimal solution, such oscillations do typically not happen.*

Figure 3.6 shows the movement of a point $\mathbf{x}$ for local search located in Region III. The aim is to reach the location over the Pareto front which maximizes the hypervolume indicator.



(a) Decision variable space



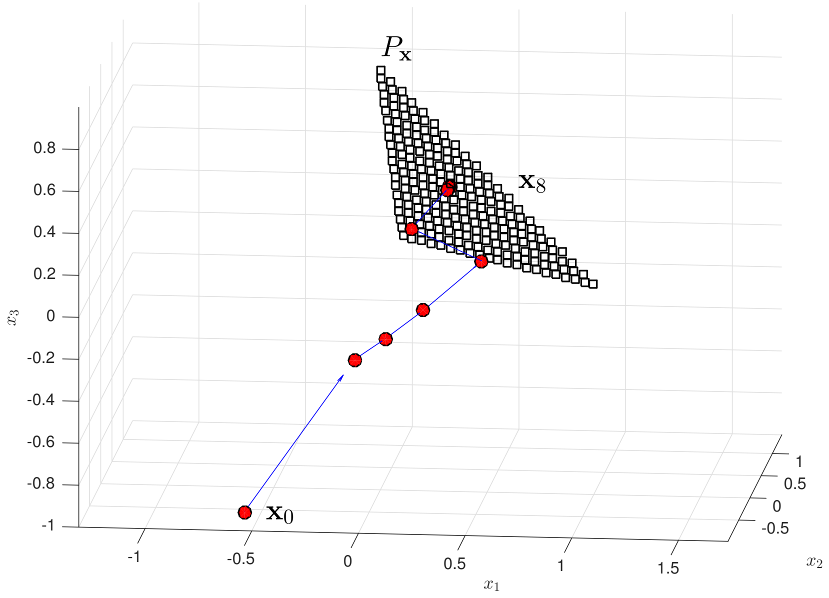
(b) Objective space

Figure 3.6: HVDS in Region I.

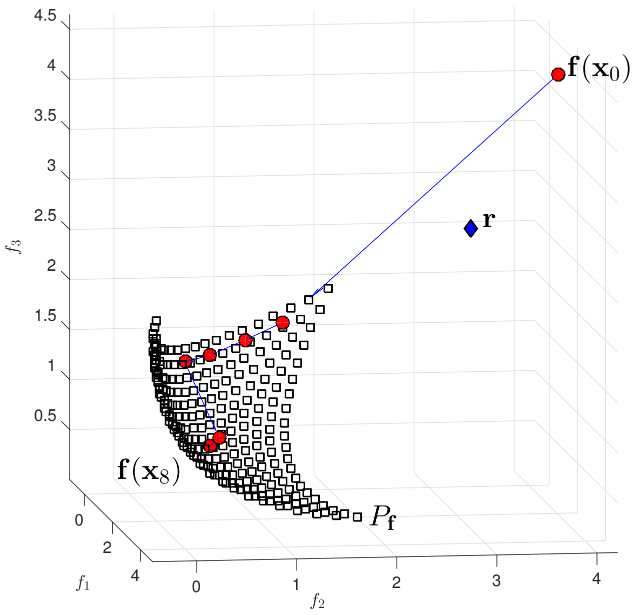
Algorithm 4 shows the pseudo code of the HVDS as standalone algorithm for one-element archives which puts together the above discussion. Figure 3.7 shows some exemplary iterations of HVDS in all three regions of MOP (3.13).

Algorithm 4 HVDS as standalone algorithm for one element archives

```
Require: $\mathbf{x}_0$: starting point, $a, b$: values for region assignment; $\mathbf{r}$: reference point
Ensure: : sequence $\{\mathbf{x}_i\}$ of candidate solutions
  $i := 0$
  repeat
      Compute $\eta(\mathbf{x}_i)$ as in Eq. (2.39)
      if $\eta(\mathbf{x}_i) \geq b$ then                                    ▷ $\mathbf{x}_i \in I$
          Compute $\nu_I$ (e.g. as in Eq. (2.37) for $k = 2$, otherwise solve Problem (2.41))
          Compute $t_I \in \mathbb{R}_+$
          $\mathbf{x}_{i+1} := \mathbf{x}_i + t_I \nu_I$
      else if $\eta(\mathbf{x}_i) \in (b, a)$ then                           ▷ $\mathbf{x}_i \in II$
          $\mathbf{d}_{II} := \mathbf{f}(\mathbf{x}_i) - \mathbf{r}$
          $\nu_{II} := J(\mathbf{x}_i)^+ \mathbf{d}_{II}$
          Compute $t_{II} \in \mathbb{R}_+$
          $\mathbf{x}_{i+1} := \mathbf{x}_i + t_{II} \nu_{II}$
      else                                                                   ▷ $\mathbf{x}_i \in III$
          Compute the KKT weight $\alpha$ as in Eq. (2.39)
          Compute $Q_2$ as in Eq. (3.19)
          Compute $\lambda^*$ by solving Problem (3.18)
          $\mathbf{d}_{III} := Q_2 \lambda^*$
          Compute $t_{III} \in \mathbb{R}_+$
          $\mathbf{x}_{i+1} := \mathbf{x}_i + t_{III} \nu_{III}$                 ▷ compute new iterate
      end if
      $i := i + 1$
  until $t_{III} = 0$ or a maximum number of iterations is reached
```



(a) Decision variable space



(b) Objective space

Figure 3.7: HVDS as standalone algorithm for a given point $\mathbf{x}_0$ assigned for local search through each region division.

General Archives

In the following, we describe the adaption of the HVDS for archives with arbitrary sizes. In the case of Region I the strategy to move points toward the solution set remains the same , however, movements for Regions II and III has to be adapted to the context.

Local search in Region I If a point $\mathbf{x} \in A$ is chosen for local search that is according to the region division 'far away' from the Pareto front, an improvement toward the solution set is desired regardless of the location of the other elements of A. Hence, we recommend to proceed as the one element archive by choosing a descent direction from Equation (2.37) in the case of $k = 2$ or proceed as in [24] for $k > 2$.

Local search in Region II If a point $\mathbf{x}_i \in A$ is located in the Region II it is necessary to consider its neighboring solutions in A since the chosen movement in objective space should ideally increase the contribution of the newly found solutions $\mathbf{x}_{new}$ computed from $\mathbf{x}$ without decreasing the contribution of the other elements in A. To perform this movement, we have to look for the k neighboring entries of $\mathbf{x}$ in A in order to set a new reference point. For $k = 2$ the adaption comes straightforward since the neighborhood from a point $\mathbf{x}$ is given by only two points for non-extreme points (where the archive entries are sorted according to one objective). For the extreme points we use the updated reference point given by the SMS-EMOA. Figure 3.8 shows the bi-objective case for setting the reference point $\mathbf{r}_{\mathbf{x}_i}$ for a non-extreme point.
For $k > 2$, the problem becomes more complicated since a neighborhood is not well defined (in the case of $k = 2$ it is enough only ordering the points in descending order to define the neighbors, however, for $k > 2$ it is not the case). To overcome this problem, we propose to choose such points that minimize the difference between each objective of $\mathbf{f}(\mathbf{x})$ from all non-dominated points in the population. Once having these points, we are now in the position to construct a new $\mathbf{r}_{\mathbf{x}}$ and use Equation (3.16) for obtaining the improvement direction. Algorithm 5 shows the pseudo code to obtain the reference point.

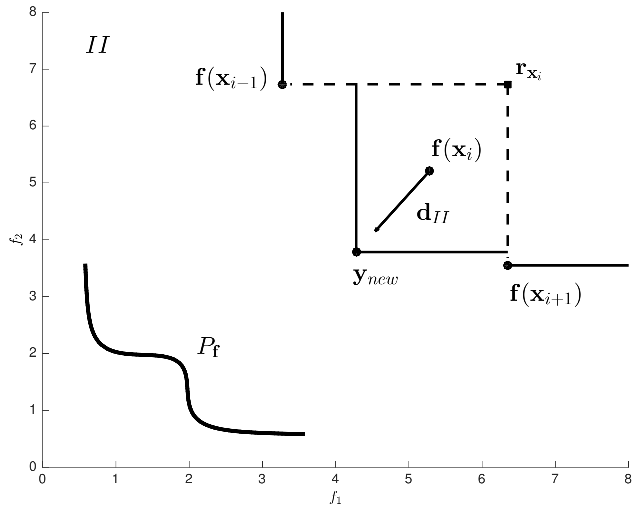

II
$\mathbf{f}(\mathbf{x}_{i-1})$
$\mathbf{r}_{\mathbf{x}_i}$
$\mathbf{f}(\mathbf{x}_i)$
$\mathbf{d}_{II}$
$\mathbf{y}_{new}$
$\mathbf{f}(\mathbf{x}_{i+1})$
$P_{\mathbf{f}}$
f_1
f_2

Figure 3.8: Local search in Region II for multiple archive entries.

Algorithm 5 Set reference point for Region II

```
Require: Point $\mathbf{x}$ from $A$ in Region $II$ and the set of non-dominated points $M$ from
  the current population.
Ensure: Reference point $\mathbf{r_x}$
  for $i = 1, \dots, k$ do
    Set $r_{\mathbf{x},i} = 0$
    Set $D_i := \infty$
    for all points $\mathbf{m}$ in $M$ do
      $aux := f_i(\mathbf{m}) - f_i(\mathbf{x})$
      if $f_i(\mathbf{m}) > f_i(\mathbf{x})$ and $aux < D_i$ then
        $D_i := aux$
        $r_{\mathbf{x},i} := f_i(\mathbf{m})$
      end if
    end for
  end for
```

Local search in Region III For Region III, we proceed to use the new reference point as stated for Region II. To be more precise, we propose to use the reference point $\mathbf{r_x}$ for intermediate points (i.e., $i \in \{2, \ldots, l-1\}$ where l is the number of elements assigned for local search in the archive A) and the original point $\mathbf{r}$ for the extreme archive entries (i.e., $i \in \{1, l\}$). In this case, we follow the same process as the one element archive using the generated reference point.

3.3 Integrating HVDS into SMS-EMOA

In the following, we present the integration of the HVDS for $k > 2$ into an EMOA. As base algorithm we have chosen to take SMS-EMOA since it is a state-of-the-art evolutionary algorithm that aims for optimal hypervolume approximations of the Pareto front. HVDS is integrated into SMS-EMOA as additional generational operator with a certain probability p_{HVDS} for application (for an empirical evaluation of the values of p_{HVDS}, a, and b we refer to the next section). As a brief discussion, the bi-objective HVDS was only integrated at the beginning of the search. Now, the algorithm will work as a generational operator which improves elements overall the optimization process. To be more precise, the new hybrid algorithm SMS-EMOA-HVDS starts by initializing a population P of μ elements at random from the domain S. Once having the initialized population the process begins by generating only one offspring $\mathbf{x}$ by genetic operators (crossover and mutation). HVDS is integrated as an additional generational operator for the produced new elements in each generation. That is, after producing a new element the algorithm selects a value $p \in [0, 1]$ uniformly at random. If p is less or equal than the probability p_{HVDS} the local search procedure will be applied, otherwise the new candidate solution is added to P. It is important to mention that the probability p_{HVDS} balances the emphasis between global and local search during the whole run of the algorithm. The aim to apply HVDS is to push the entire population toward the solution set during the algorithm process. In this case the produced element can be everywhere within the objective space, however, thanks to the region division we can make the decision of which LS strategy should be chosen in order to improve the hypervolume. Each time the HVDS strategy is called, it executes only one iteration step in our current implementation. Further, we use the dynamic reference point described in Algorithm 5. Algorithm 6 shows the pseudo code of the resulting algorithm SMS-EMOA-HVDS.

Algorithm 6 SMS-EMOA-HVDS

```
Require: MOP, archive size $\mu$, values $a, b$ for region division, probability $p_{HVDS}$ to
  apply the local search operator
Ensure: An approximation $P$ of the Pareto given MOP.
  Initialize a population $P \subset Q$ with $\mu$ elements at random
  repeat
      generate offspring $x \in Q$ from $P$ by variation
      choose $p \in [0, 1]$ uniformly at random
      if $p \leq p_{HVDS}$ then                                  ▷ Apply HVDS
          choose the generated offspring $x$ for local search
          $\tilde{x}$ :=HVDS($x$, $a$, $b$, $R$)
      else
          $\tilde{x} := x$
      end if
      $P := P \cup \{\tilde{x}\}$
      build ranking $S_1, \ldots, S_h$ from $P$
      compute the hypervolume contribution for each $x \in S_h$
      denote by $x^*$ the element with the least hypervolume contribution
      $P := P \setminus \{x^*\}$
  until stopping criterion fulfilled
  return $P$
```

3.4 Test Problems

The proposed hybrid algorithm is going to be tested over well-established two-objective (2D) and three-objective (3D) test function problems. For $k = 2$ (2D), we consider the DTLZ and the shifted and unconstrained ZDT (see [105]) test suites. In addition, two problems with a convex (MOP1) and convex-concave (MOP-DENT) Pareto fronts are included to the set of test functions. For $k = 3$ (3D), we select the DTLZ test suite and a convex Pareto front problem to see the applicability of the HVDS and a problem with convex Pareto front (MOP3). The previous test function problems were selected in order to see the behavior of the memetic algorithm within certain characteristics. For instance unimodal (DTLZ2, ZDT1, ZDT2, ZDT3, MOP1, MOP3 and MOP-DENT) and multimodal (DTLZ1, DTLZ3, DTLZ7, and ZDT6) test functions. Also different shapes of the Pareto fronts are considered such as linear (DTLZ1), convex (ZDT1, ZDT2, ZDT6, MOP1 and MOP3), concave (DTLZ2 and DTLZ3), disconnected (ZDT3 and DTLZ7), and convex-concave (MOP-DENT) Pareto fronts (see Appendix A for the definition of the selected MOPs).

3.5 Parameter setting

Over this section, we define the process to set the design parameters of Algorithm 6. In Table 3.1 the parameters are explained in more detail.

Table 3.1: Design parameters that are required for the realization of the HVDS.

Parameter	Description
a	Parameter to set the region division $a \in (0, 1)$.
b	Parameter to set the region division $b \in (0, 1) \wedge a < b$.
p_{HVDS}	Probability to apply the HVDS as an operator.

The design parameter configuration task is a time-consuming process which needs a high computational effort, however, it is necessary to find the best design parameters for an algorithm. In this case, we desire to find such parameters which work properly for the memetic algorithm. To perform this task, we use an adaption of the sequential parameter optimization (SPO) technique for tunning stochastic optimization algorithms proposed in [106] and extended to multi-objective optimization problems in [107, 108]. The SPO is adapted in the following to optimized the design parameters explained in Table 3.1.

3.5.1 Experimental Construction

Unlike the works proposed in [107, 108], we aim to find the best design parameters for our local search technique HVDS instead of finding the design parameters of the multi-objective evolutionary algorithm. Tables 3.2 and 3.3 show the setups for the experiments in 2D and 3D respectively.

Table 3.2: Setup for experiments in 2D with the SMS-EMOA-HVDS.

Name	Description
Problems	DTLZ1, DTLZ2, DTLZ3, MOP1, MOP-DENT, ZDT1, ZDT2, ZDT3, ZDT6.
HVDS parameters	a, b, and p_{HVDS}.
Stopping criterion	30,000 function evaluations.
Algorithm initialization	Uniform random.
Performance indicator	Hypervolume.

Table 3.3: Setup for experiments in 3D with the SMS-EMOA-HVDS.

Name	Description
Problems	DTLZ1, DTLZ2, DTLZ3, DTLZ7, and MOP3.
HVDS parameters	a, b, and p_{HVDS}.
Stopping criterion	50,000 function evaluations.
Algorithm initialization	Uniform random.
Performance indicator	Hypervolume.

Additionally, in Table 3.4 we determine the region of interest (ROI) of each design parameter and the discretization of every region to avoid wrong configurations.

Table 3.4: Setup for experiments in 3D with the SMS-EMOA-HVDS.

Parameter	ROI	Discretization
a	$[0.05, 0.2]$	$\{0.05, 0.1, 0.15, 0.2\}$
b	$[0.2, 0.9]$	$\{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$
p_{HVDS}	$[0.01, 0.2]$	$\{0.005, 0.01, 0.05, 0.1, 0.2\}$

We perform 20 independent experiments for every test function problem and the reference point is problem dependent, nevertheless in most of the cases, we take

$(11, 11)^T$ for 2D and $(11, 11, 11)^T$ for 3D. In total, we have to perform 3,200 experiments for each test function problem. Table 3.5 includes the configuration of the design parameters of the SMS-EMOA.

Table 3.5: Configuration values for SMS-EMOA, where DI means "distribution index", SBX stands for simulated binary crossover and PM for polynomial mutation.

Description	Parameter	2D values	3D values
Crossover Operator	SBX	$DI = 20.0$, $p = 0.9$	$DI = 20.0$, $p = 0.9$
Mutation Operator	PM	$DI = 20.0$, $p = 1/n$	$DI = 20.0$, $p = 1/n$
Population Size	μ	100	300

In order to decide which of the configuration is the best, we use boxplots in three different stages of the evolution process to see the performance of each configuration against the SMS-EMOA without local search. Then, we select the ones which obtain a better value than SMS-EMOA. Finally, we select the configuration which outperforms SMS-EMOA and behaves well in the different test function problems.

3.6 Numerical Results

The algorithm was applied 20 times with population sizes 100 (2D) and 300 (3D), respectively. The value of the hypervolume indicator (HV) of the final result is used as basis of comparison to respective runs of the original SMS-EMOA. Based on extensive systematic investigations performed in the previous section, the settings $a = 0.2, b = 0.4$ (2D) resp. $a = 0.1$ (3D) together with $p_{HVDS} = 0.005$ turned out to be most stable over all settings while in general algorithm behavior is not extremely sensitive regarding the respective settings of a and b. A dynamic reference point was used for Region II (see Algorithm 5). Tables 3.6 and 3.7 provide the detailed algorithm settings along with the experimental results and thereby allows for the assessment of the effect of the HVDS operator.

For all test functions, the superior algorithm regarding the average HV is highlighted in gray indicating that the sophisticated local search strategy indeed has a positive effect on algorithm performance. The corresponding ordering regarding me-

Table 3.6: HV results of SMS-EMOA with HVDS as operator (using the same number of function evaluations). The values are obtained over 20 independent test runs and for $k = 2, 3$.

				SMS-EMOA-HVDSMO		
Problem	# Func	n	Average	Deviation	Median	Covered HV
DTLZ1 (2D) M	30K	15	118.25014	3.65605	119.61943	97.82947 %
DTLZ2 (2D) U	10K	30	120.20997	0.00055	120.21009	99.99944 %
DTLZ3 (2D) M	30K	15	101.84429	23.49979	105.40498	84.72152 %
MOP1 (2D) U	10K	2	3251.81142	16.40663	3253.60993	99.81105 %
MOP-DENT (2D) U	10K	2	17.34873	0.00220	17.34852	99.88260 %
ZDT1 (2D) U	10K	30	109.53759	0.17000	109.60691	99.88643 %
ZDT2 (2D) U	10K	30	108.61966	1.95078	109.22509	99.35129 %
ZDT3 (2D) U	10K	30	116.73398	1.46083	117.67874	99.11522 %
ZDT6 (2D) M	10K	15	104.81097	0.97200	104.78108	98.40024 %
MOP3 (3D) U	10K	15	21.14882	0.03884	21.16804	99.99858 %
DTLZ1 (3D) M	50K	15	1330.93727	0.07304	1330.97379	99.99723 %
DTLZ2 (3D) U	10K	30	1330.41646	0.00501	1330.41637	99.99920 %
DTLZ3 (3D) M	50K	15	1326.62757	5.85269	1330.17426	99.71441 %
DTLZ7 (3D) M	30K	15	964.42986	57.13546	992.99807	97.10511 %

Table 3.7: HV results of SMS-EMOA without HVDS as operator (using the same number of function evaluations). The values are obtained over 20 independent test runs and for $k = 2, 3$.

			SMS-EMOA			
Problem	# Func	n	Average	Deviation	Median	Covered HV
DTLZ1 (2D) M	30K	15	117.40941	5.64543	119.24526	97.13393 %
DTLZ2 (2D) U	10K	30	120.20604	0.00130	120.20637	99.99617 %
DTLZ3 (2D) M	30K	15	105.68880	9.48479	107.17194	87.91967 %
MOP1 (2D) U	10K	2	3248.72641	22.57145	3253.23842	99.71636 %
MOP-DENT (2D) U	10K	2	17.34866	0.00270	17.34891	99.88220 %
ZDT1 (2D) U	10K	30	109.26564	0.63385	109.57209	99.63844 %
ZDT2 (2D) U	10K	30	106.42534	3.45765	109.13577	97.34421 %
ZDT3 (2D) U	10K	30	117.05177	1.27276	117.69188	99.38512 %
ZDT6 (2D) M	10K	15	97.60659	1.49610	97.41107	91.63651 %
MOP3 (3D) U	10K	15	21.10150	0.04299	21.09881	99.77483 %
DTLZ1 (3D) M	50K	15	1330.93127	0.07912	1330.97013	99.99678 %
DTLZ2 (3D) U	10K	30	1330.41083	0.00288	1330.41086	99.99878 %
DTLZ3 (3D) M	50K	15	1325.70533	8.01075	1330.19390	99.64509 %
DTLZ7 (3D) M	30K	15	993.01561	0.02094	993.01893	99.98331 %

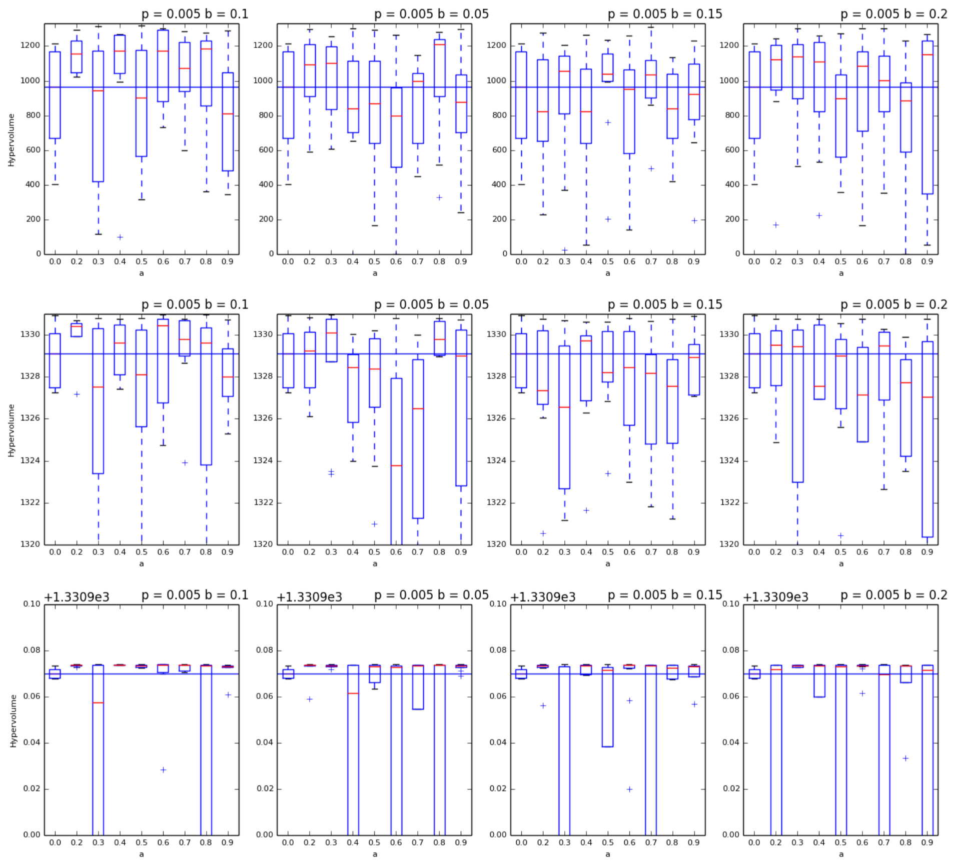


Figure 3.9: Example of the set of boxplots using the hypervolume indicator in three different stages for the 3-objective: DTLZ1.

dian performance only changes slightly. Moreover, on all considered problems the percentage of the covered optimal hypervolume is very high while keeping in mind that the number of function evaluations has not been specifically tuned to gain optimal hypervolume. The optimal HV values for all the problems have been computed by numerically solving the following $(\mu \cdot n)$-dimensional scalar optimization problem

$$\max_{v^{(1)},\dots,v^{(\mu)} \in \mathbb{R}^n} \mathcal{H}(v^{(1)}, \dots, v^{(\mu)}; r), \tag{3.26}$$

where μ denotes the population size and $\mathcal{H} : \mathbb{R}^{n \cdot \mu} \to \mathbb{R}$ the objective of the population based hypervolume indicator.

Robustness and statistical significance of the results can be assessed by observing the algorithms' behavior along the whole run in Figures 3.10 to 3.16 and section 3.6. Also SMS-EMOA-HVDS often shows faster convergence in earlier phases of the optimization. The respective boxes range from the lower up to the upper quartile while the median is higlighted by a horizontal line inside the box. Points are plotted individually as outliers if they are larger than box boundaries plus resp. minus 1.5 times the interquartile range. The plotted whisker terminates at the adjacent point, which is the most extreme point that is not an outlier. Results with (LS) and without local search (WLS) are compared while the horizontal line visualizes the median HV of the original SMS-EMOA runs.

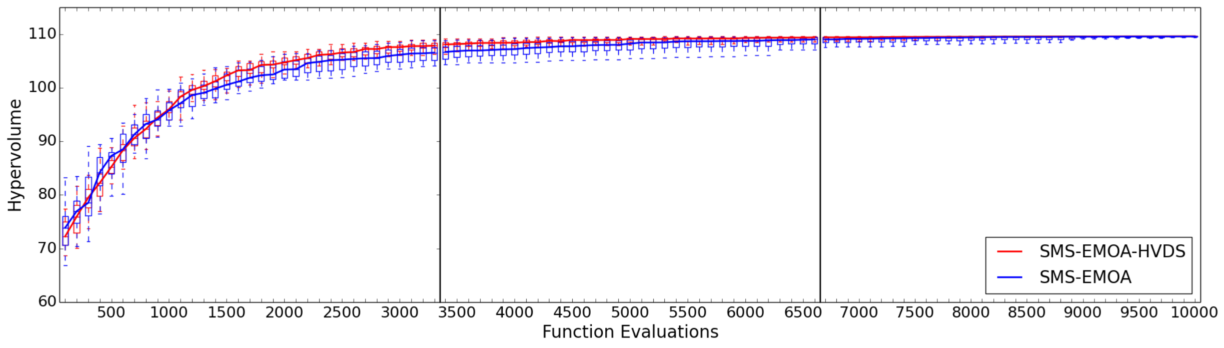


Figure 3.10: Comparison using boxplots for ZDT1 (2D) during 30,000 function evaluations (★).

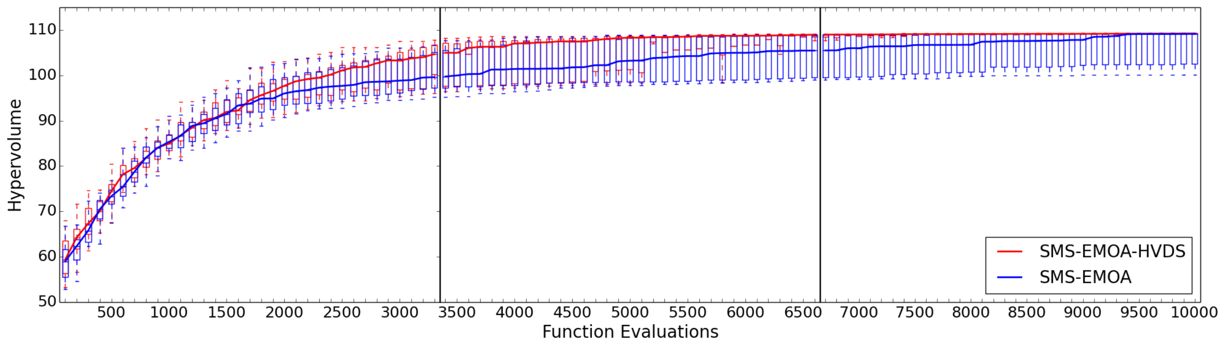


Figure 3.11: Comparison using boxplots for ZDT2 (2D) during 30,000 function evaluations (★).

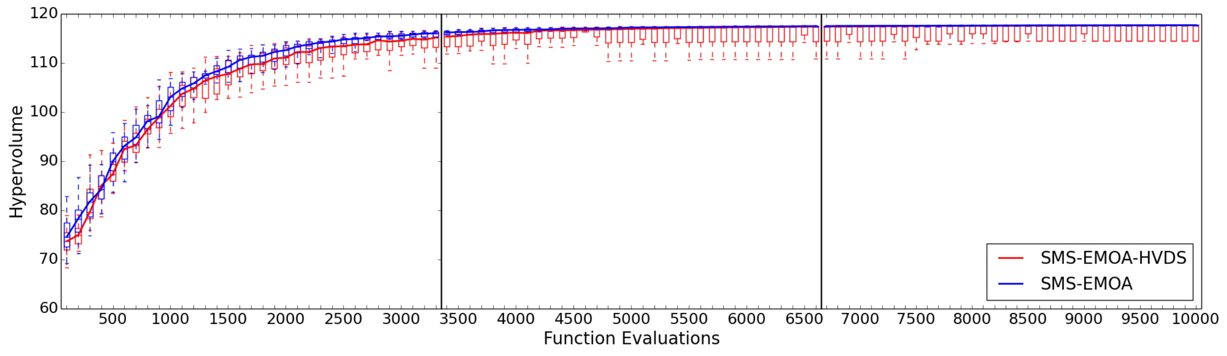


Figure 3.12: Comparison using boxplots for ZDT3 (2D) during 30,000 function evaluations.

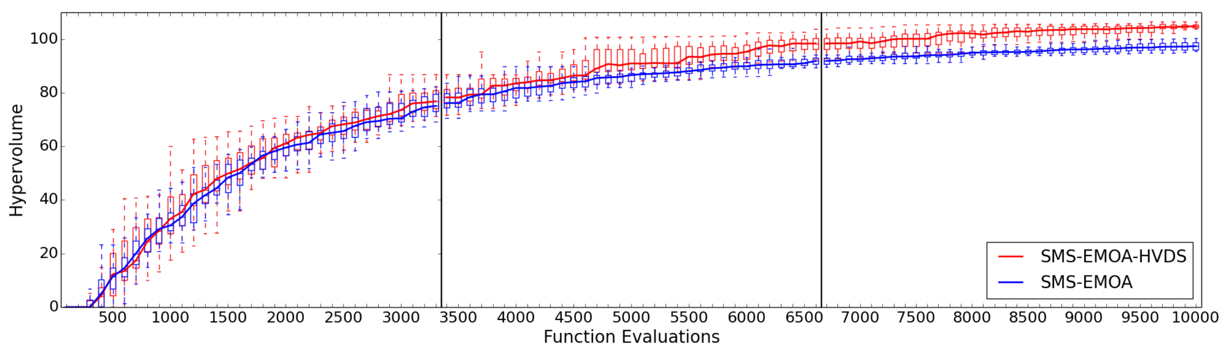


Figure 3.13: Comparison using boxplots for ZDT6 (2D) during 30,000 function evaluations (★).

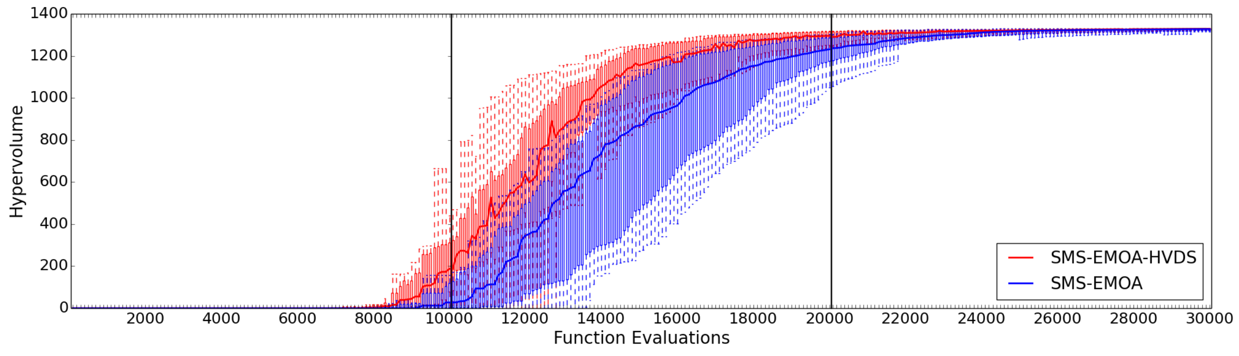


Figure 3.14: Comparison using boxplots for DTLZ1 (3D) during 50,000 function evaluations (★).

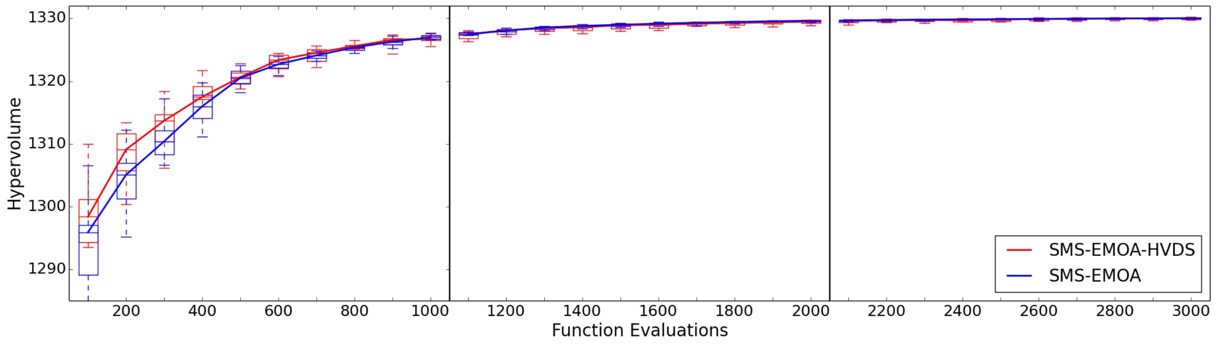


Figure 3.15: Comparison using boxplots for DTLZ2 (3D) during 50,000 function evaluations (★).

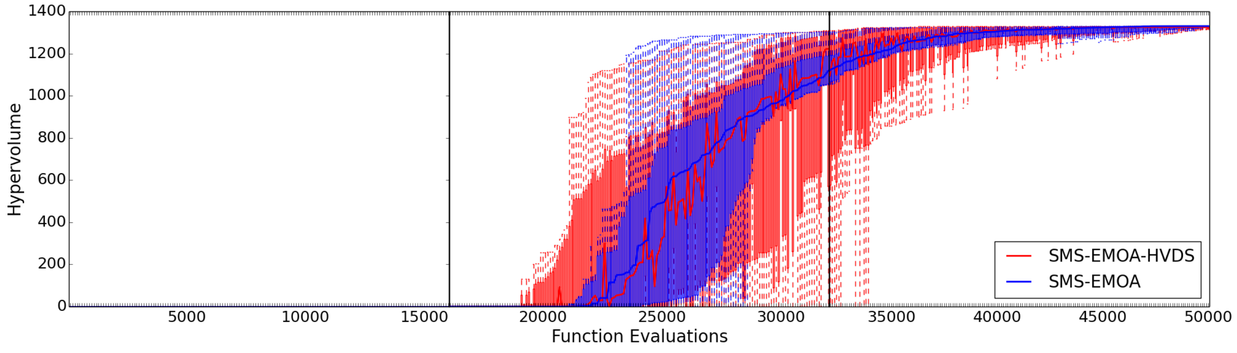


Figure 3.16: Comparison using boxplots for DTLZ3 (3D) during 50,000 function evaluations.

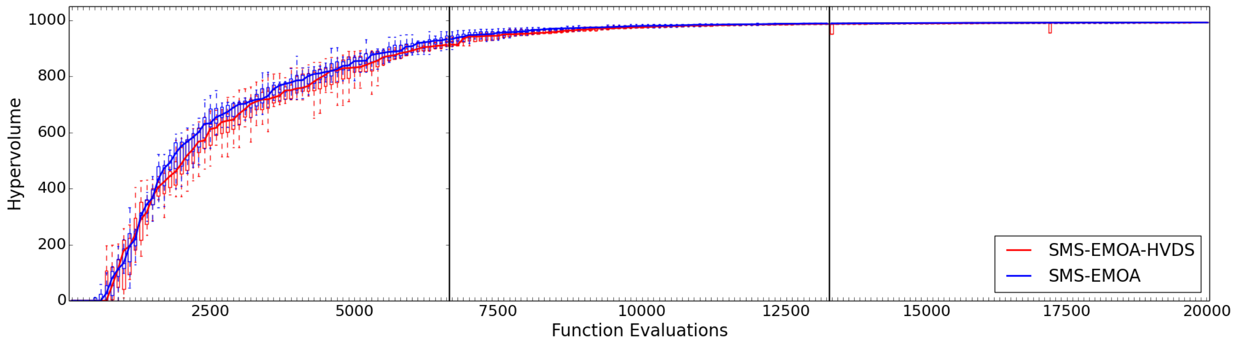


Figure 3.17: Comparison using boxplots for DTLZ7 (3D) during 50,000 function evaluations.

Figures marked with (★) reflect a statistically significant result based on the Wilcoxon Rank test using a significance level $\alpha = 0.05$. It becomes obvious that in 3D the SMS-EMOA supplemented with the HVDS strategy outperforms the original version for almost all test functions. In the case of ZDT3 and DTLZ7, we detect that the performed movements in Region III has no positive effect for certain regions within a disconnected Pareto front. The extremes of every disconnected segment of the Pareto front could, in this case, produce dominated points, since we performed a search over the orthogonal vector to the linearization of the Pareto front. Then, this linearization could point to a dominated region which cause that the generated point does not improve the hypervolume value. For DTLZ3, we observe that since the problem is multimodal movement in Region III could occur over local fronts which cause no improvement to the hypervolume. In 2D, results are comparable for all test functions and superiority of the local search variant is given for DTLZ3, ZDT1, ZDT2 and ZDT6. However, also due to the sophisticated strategy in Region III which searches *along* the Pareto front, the algorithm in general faces special challenges for disconnected fronts.

4 | The Newton Method for Hypervolume Indicator

Multi-objective optimization problems are predominantly solved by population based metaheuristics. Based on stochastic search operators they perform robustly, however, also tend to lack precision in the final stage of the approximation as well as convergence guarantees. In the last decade, the hypervolume indicator has been one of the most widely used tool to measure the performance in evolutionary multi-objective optimization. This indicator has also been integrated into some derivative free evolutionary methods that work to maximize the hypervolume value (see [57, 84, 99, 109, 110]). While stochastic methods that aim for maximizing the hypervolume value have received considerable attention, research on using deterministic and derivative based methods is still in its infancy. The use of gradients in multi-objective optimization has long been considered as a remedy to imprecision [24, 26, 95, 111]. However, the initially proposed methods were only able to improve the closeness to the Pareto front, but not the diversity of approximations. In order to include diversity in gradient based search for Pareto fronts Schütze et al. proposed directed search methods (see [28, 85]), which allow not only to move towards the Pareto front based on local gradient information, but also along the Pareto front in order to increase diversity. Among these contributions to the state-of-the-art, it is not clearly specified how improvements can (or should) be measured. The integration of an indicator can steer the local search in better directions and also be more effective for improving points.

As already mentioned, Emmerich et al. proposed to use the hypervolume gradient defined on an entire population in order to find diverse set-approximations to the

Pareto front (see [35, 36]). These first results showed that gradient based methods can locally improve accuracy of hypervolume maximal sets, however, specialized step size control strategies has to be coupled with the method to work properly.

In this chapter, we present a full study of the behavior of population based techniques that aim for maximizing the hypervolume value, more specifically the produced result by following the hypervolume gradient flow. Under this flow, populations evolve toward a final state (population) whose hypervolume indicator is locally maximal. Some insights obtained on selected test functions explain to a certain extent observations made in previous studies and give some possible insights into the application of mathematical programming techniques to this problem. Next, the hypervolume Hessian matrix is stated and we present two ways to compute it by using finite differences and its exact computation. Based on this, we propose the population based Hypervolume Newton Method (HNM) for hypervolume maximization. We first address unconstrained MOPs and make further on first attempts for the treatment of constrained problems by including inequality and equality constrained MOPs. Fast population-based convergence can be observed towards optimal populations, however, the results indicate that the success depends crucially on the choice of the initial population. The resulting method may even converge quadratically, however, this property is—as for all Newton methods—only of local nature. Also, we propose a hybrid of the HNM and an evolutionary algorithm in order to obtain a fast and reliable algorithm for the numerical treatment of such problems. Finally, to show the strengths of both methods the HNM standalone algorithm and the novel memetic approach, we present numerical results and comparisons on several benchmark problems.

4.1 The Hypervolume Gradient Flow

Here, we start by investigating the flow that is induced by the hypervolume gradient. The aim of this study is to understand the success of failure of using the hypervolume gradient direction over different MOPs. Recall from Chapter 2 that we can state the hypervolume function $\mathcal{H}$ set-based (i.e., the variables of $\mathcal{H}$ are individuals of a given approximation set) as in Equation (2.53). Then, the hypervolume flow of a given starting vector $\mathbf{X}_0$ can be stated as the following initial value problem:

$$\begin{aligned} &\mathbf{X}(0) = \mathbf{X}_0 \in \mathbb{R}^n \text{ with } |\mathbf{X}_0| = \mu \\ &\dot{\mathbf{X}}(t) = \nabla\mathcal{H}(\mathbf{X}(t)), \quad t > 0. \end{aligned} \tag{4.1}$$

By following this flow, we obtain for every initial vector $\mathbf{X}_0$ a final vector whose hypervolume is locally optimal (this is due to the fact that the hypervolume gradient is zero at every end point of Problem (4.1)). Thus, the geometry of such solution curves are of particular interest for the understanding of the success or failure of the related hypervolume gradient-based optimization technique.

For our purpose, we consider the next four bi-objective optimization problems:

1. The generalized Schaffler problems (MOP-GSP) by Emmerich and Deutz ([112]):

$$\begin{aligned} &f_1, f_2 : \mathbb{R}^n \to \mathbb{R} \\ &f_1(\mathbf{x}) = \frac{(\sum_{i=1}^n x_i^2)^\alpha}{(n^\alpha)} \\ &f_2(\mathbf{x}) = \frac{(\sum_{i=1}^n (1 - x_i)^2)^\alpha}{(n^\alpha)} \end{aligned} . \tag{4.2}$$

   The Pareto set of this problem family is given by the line segment connecting the points $(0, 0)^T$ and $(1, 1)^T$ (including the end points). The Pareto front is concave for $\alpha < 0.5$, linear for $\alpha = 0.5$, and convex for $\alpha > 0.5$. For $\alpha = 0.5$ the hypervolume indicator maximum is known to be an evenly spaced point set on the efficient set and Pareto front [112].

2. The next problem is MOP2 ([16]):

$$\begin{aligned} &f_1, f_2 : \mathbb{R}^2 \to \mathbb{R} \\ &f_1(\mathbf{x}) = (x_1 - 1)^4 + (x_2 - 1)^2. \\ &f_2(\mathbf{x}) = (x_1 + 1)^2 + (x_2 + 1)^2 \end{aligned} \tag{4.3}$$

   The Pareto set is a curve connecting the points $\mathbf{m}_1 = (1,1)^T$ and $\mathbf{m}_2 = (-1,-1)^T$ and its Pareto front is convex.

3. We consider also the problem MOP1 stated in (2.13) variant of the above model. In this case the Pareto set is a line segment connecting $\mathbf{m}_1$ and $\mathbf{m}_2$, and the Pareto front is convex.

4. Finally, we will consider the MOP-DENT ([113]):

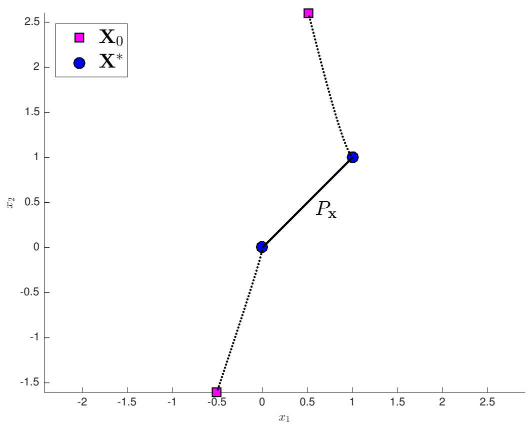
$$\begin{aligned} &f_1, f_2 : \mathbb{R}^2 \to \mathbb{R} \\ &f_1(\mathbf{x}) = \frac{1}{2} \cdot (\sqrt{(1 + (x_1 + x_2)^2)} + \sqrt{(1 + (x_1 - x_2)^2)} + x_1 - x_2) + \lambda \cdot e^{(-1 \cdot (x_1 - x_2)^2)} \\ &f_2(\mathbf{x}) = \frac{1}{2} \cdot (\sqrt{(1 + (x_1 + x_2)^2)} + \sqrt{(1 + (x_1 - x_2)^2)} - x_1 + x_2) + \lambda \cdot e^{(-1 \cdot (x_1 - x_2)^2)}, \\ &\qquad \text{where } \lambda = 0.85 \end{aligned} \tag{4.4}$$

   For the domain $S = [-4,4]^2$ the Pareto front is the line segment connecting the points $(4,-4)^T$ and $(-4,4)^T$ and its Pareto front is convex-concave.

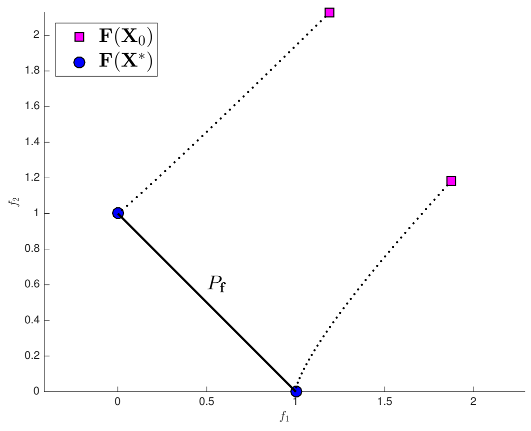
Example 1. To examine the hypervolume flow, we first consider MOP-GSP for $\alpha = 0.5$ (see Figure 4.1). The optimal archive in this case are two Pareto optimal solutions near to the end points of the solution set. Starting with the 2-element archive

$$\mathbf{X}_0 = \{\mathbf{x}_1 = (-0.5, -2.6)^T, \mathbf{x}_2 = (0.5, 2.6)^T\}, \tag{4.5}$$

we can observe that both individuals $\mathbf{x}_1$ and $\mathbf{x}_2$ directly move toward the nearest optimal solution. The flow hence yields the desired behavior.



(a) Decision Variable Space



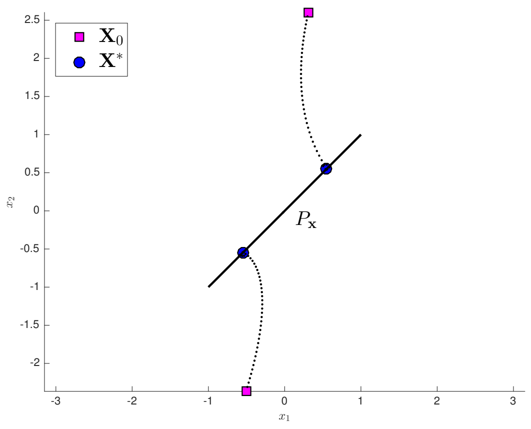
(b) Objective Space

Figure 4.1: Hypervolume flow on a 2-element population on MOP-GSP for $\alpha = 0.5$.

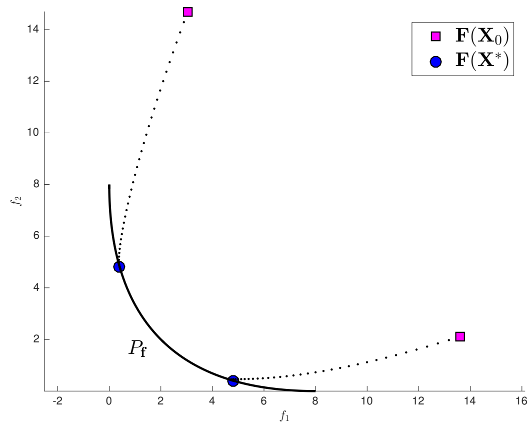
Example 2. Next, we consider MOP1 (see Figure 4.2). Similar to Example 1 the optimal archive are two Pareto optimal solutions, in this case the optimal points are not near to the end points of the solution set. Starting with the 2-element archive

$$\mathbf{X}_0 = \{\mathbf{x}_1 = (-0.5, -2.3)^T, \mathbf{x}_2 = (0.3, 2.6)^T\}, \tag{4.6}$$

we can observe that both individuals$\mathbf{x}_1$ and $\mathbf{x}_2$ directly move toward the nearest optimal solution. The flow hence yields again the desired behavior.



(a) Decision Variable Space



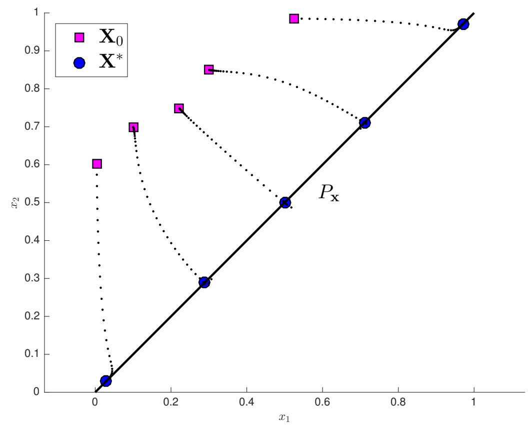
(b) Objective Space

Figure 4.2: Hypervolume flow on a 2-element population on MOP1.

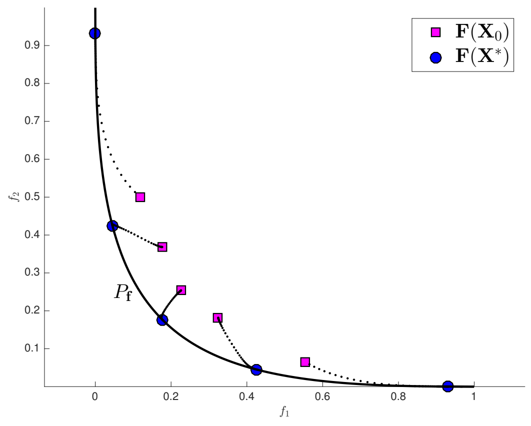
Example 3. We consider a 5-element population on MOP-GSP for $\alpha = 1.25$ (see Figure 4.3). Similarly to the above two examples, all individuals of the initial population

$$\mathbf{X}_0 = \{\mathbf{x}_1 = (0.0, -0.6)^T, \mathbf{x}_2 = (0.1, 0.7)^T, \mathbf{x}_3 = (0.2, 0.7)^T, \mathbf{x}_4 = (0.3, 0.8)^T, \mathbf{x}_5 = (0.5, 1.0)^T\}, \tag{4.7}$$

perform a more or less direct movement (both in decision and objective space) toward the optimal 5-element hypervolume archive for the reference point $\mathbf{r} = (10, 10)^T$.



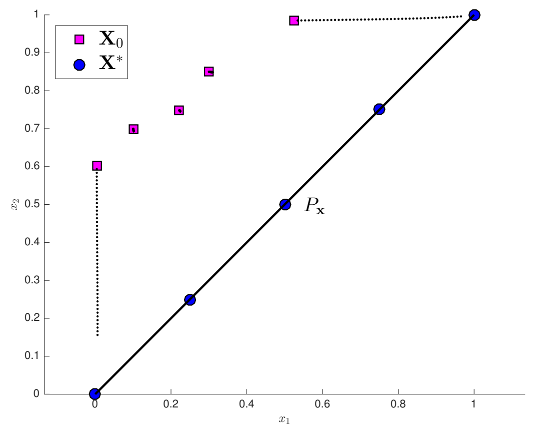
(a) Decision Variable Space



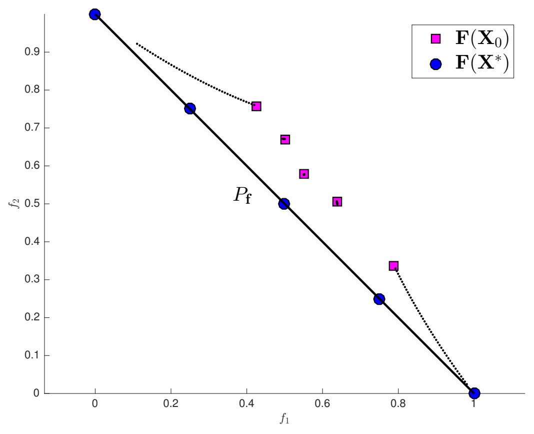
(b) Objective Space

Figure 4.3: Hypervolume flow on a 5-element population on MOP-GSP for $\alpha = 1.25$.

Example 4. Next, we consider the same initial population as before, but consider MOP-GSP with $\alpha = 0.5$ and $\alpha = 0.25$ (i.e., only the value of α in the model has been changed, the Pareto front is now linear and then concave, respectively). Instead of a convergent behavior we see in Figures 4.4 and 4.5 that the extreme solutions $\mathbf{x}_1$ and $\mathbf{x}_5$ (to be more precise, the solutions such that the images $\mathbf{f}(x_i)$ are minimal according to f_1 and f_2, respectively) perform a movement toward the extreme points of the Pareto set/front. The other solutions also perform a movement toward the Pareto set, however, it is apparent that this is done with a much slower 'speed'.

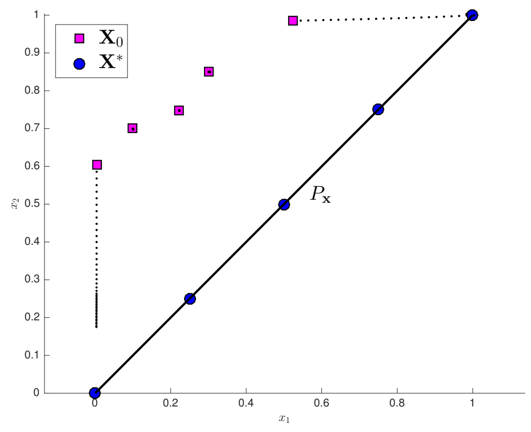


(a) Decision Variable Space

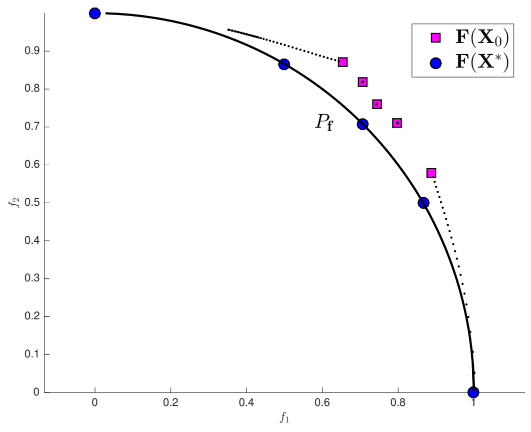


(b) Objective Space

Figure 4.4: Hypervolume flow on a 5-element population on MOP-GSP for $\alpha = 0.5$.



(a) Decision Variable Space



(b) Objective Space

Figure 4.5: Hypervolume flow on a 5-element population on MOP-GSP for $\alpha = 0.25$.

The reason for this 'creepiness' is certainly the huge difference in the norms of the sub-gradients. A discretization of (4.1) via the Euler method leads from a given population $\mathbf{X}_i$ to the new one

$$\mathbf{X}_{i+1} = \mathbf{X}_i + t_i \nabla \mathcal{H}(\mathbf{X}_i), \tag{4.8}$$

where $t_i > 0$ is the chosen step size. For every individual $\mathbf{x}_j \in \mathbf{X}_i$ it thus holds

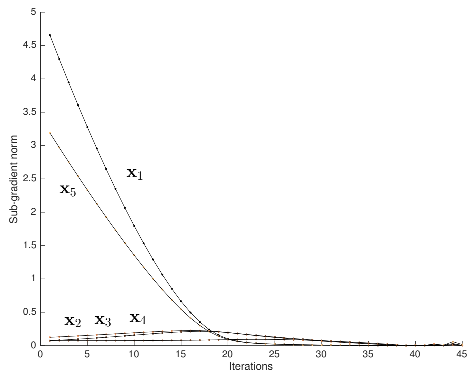
$$\mathbf{x}_j^{(new)} = \mathbf{x}_j + t_i \mathbf{g}_j, \tag{4.9}$$

where $\mathbf{x}_j^{(new)}$ denotes the new individual and $\mathbf{g}_j \in \mathbb{R}^n$ the sub-gradient of $\nabla\mathcal{H}$ and $\mathbf{x}_i$.

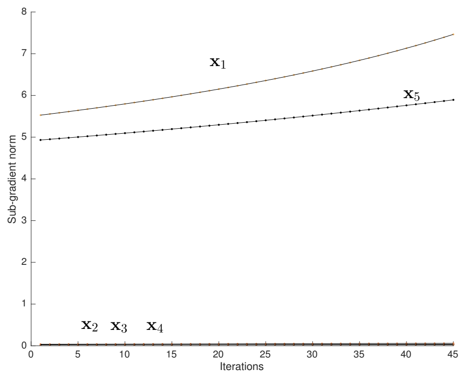
For the difference of the two individuals it holds hence

$$\|\mathbf{x}_j^{(new)} - \mathbf{x}_j\| = t_i \|\mathbf{g}_j\|. \tag{4.10}$$

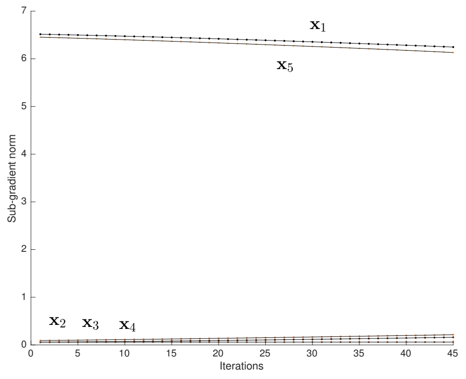
Since t_i is equal for all sub-gradients, the speed of the movement is hence entirely determined by the norm of the sub-gradients. Figure 4.6 shows the evolution of the norms of the sub-gradients for the populations considered in Examples 3 and 4. In both cases, there is a significant difference in the norms, namely the norms for the extreme solutions are much higher than the norms of the other individuals. While this difference reduces for the first problem until all norms are in the same range, this does not hold for Example 4. Instead, this difference is nearly stable for the considered sequence.



(a) Decision Variable Space



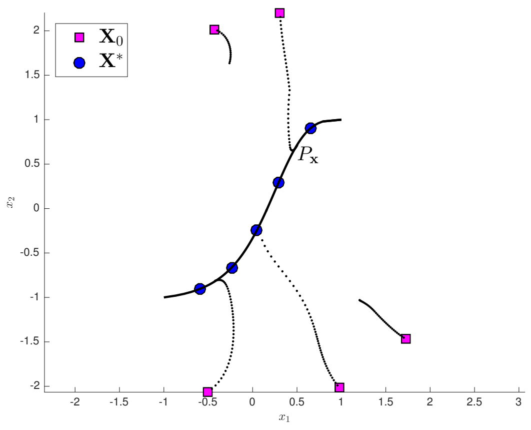
(b) Objective Space



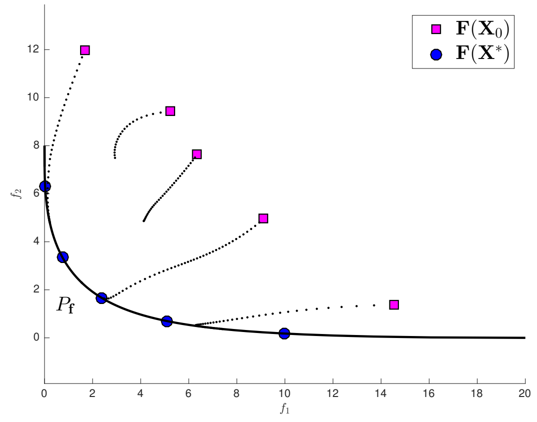
(c) Objective Space

Figure 4.6: Behavior of the norms of the sub-gradients of the populations considered in Figures 4.3 to 4.5.

Example 5. Figures 4.7 and 4.8 show two examples of 5-element populations where domination occurs (using MOP2 and MOP-DENT). To be more precise, it happens along the gradient flow that solutions in the population start to dominate others. Strictly dominated solutions have a hypervolume contribution that is constantly zero in some epsilon environment and their sub-gradients are zero vectors which causes the points that correspond to these sub-gradients to become stationary. The search continues with the remaining non-dominated solutions. In both cases, the iterations converge toward the optimal hypervolume archive, albeit for 3 elements.

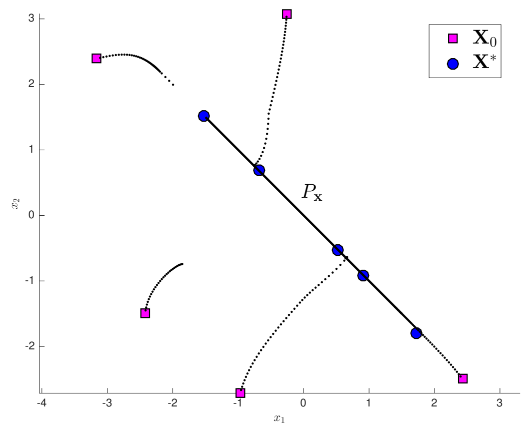


(a) Decision Variable Space

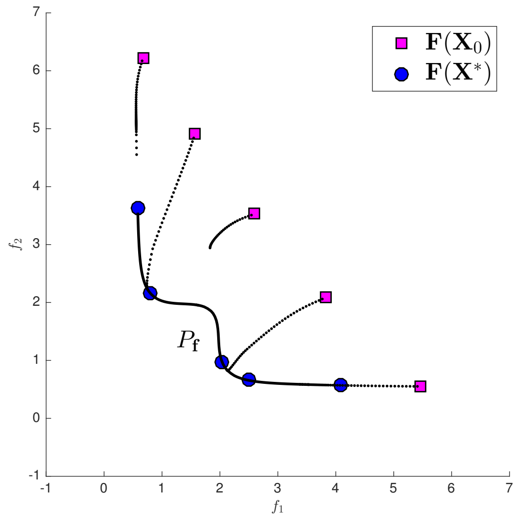


(b) Objective Space

Figure 4.7: Hypervolume flow on a 5-element population on MOP2.



(a) Decision Variable Space



(b) Objective Space

Figure 4.8: Hypervolume flow on a 5-element population on MOP-DENT.

As we mention in Chapter 2 a study on the use of the steepest ascent method for the numerical realization of (4.1) can be found in [35]. Linear set-wise convergence has been achieved in case of linear Pareto fronts, however, also the loss of certain elements during the search as a result of domination has been reported. The above observed creepiness may be an explanation for this phenomenon.

4.2 The Hypervolume Hessian

In this section, we start by approximating the Hessian matrix using finite differences. Then, we derive the mathematical expression of the Hessian matrix of the hypervolume indicator for the general *multi-objective* optimization scenario. Finally, the Hessian matrix for *bi-objective* cases is presented by its mathematical formulation.

4.2.1 Approximating the Hypervolume Hessian

Before, we derive the mathematical expression for computing the Hypervolume Hessian matrix, we use the following finite difference approximation to compute each member of this matrix:

$$\nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X})_{i,j} = \frac{\mathcal{H}_{\mathbf{F}}(\mathbf{X}+h\mathbf{e}_i+h\mathbf{e}_j)-\mathcal{H}_{\mathbf{F}}(\mathbf{X}+h\mathbf{e}_i-h\mathbf{e}_j)-\mathcal{H}_{\mathbf{F}}(\mathbf{X}-h\mathbf{e}_i+h\mathbf{e}_j)-\mathcal{H}_{\mathbf{F}}(\mathbf{X}-h\mathbf{e}_i-h\mathbf{e}_j)}{4h^2} \tag{4.11}$$

where i and j represent the entry of the Hessian matrix $\nabla^2\mathcal{H}_{\mathbf{F}}(\mathbf{X}) \in \mathbb{R}^{\mu\cdot n\times\mu\cdot n}$ and $\mathbf{e}_i$ and $\mathbf{e}_j$ represent the ith-column and the jth-column respectively of the identity matrix $\mathbf{e} \in \mathbb{R}^{\mu\cdot n\times\mu\cdot n}$. We set $h = 0.001$ for all our computations.

4.2.2 Analytic Formulation of the Hypervolume Hessian

Here, the analytic formulation of the Hypervolume Hessian matrix is presented. For conciseness, matrix calculus notations are used in the following derivation, which helps to understand the structure of the Hessian matrix. The hypervolume Hessian matrix is the "Jacobian" of the hypervolume gradient defined as follows:

$$
\begin{aligned}
\nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) &= \frac{\partial}{\partial \mathbf{X}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{X}}\right) \qquad (4.12)\\
&= \left(\underbrace{\frac{\partial}{\partial \mathbf{X}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(1)}}\right)}_{\mu\cdot n\times n}, \ldots, \frac{\partial}{\partial \mathbf{X}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(\mu)}}\right)\right)\\
&= \begin{pmatrix}
\frac{\partial}{\partial \mathbf{x}^{(1)}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(1)}}\right) & \cdots & \frac{\partial}{\partial \mathbf{x}^{(1)}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(\mu)}}\right)\\
\vdots & \ddots & \vdots\\
\frac{\partial}{\partial \mathbf{x}^{(\mu)}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(1)}}\right) & \cdots & \frac{\partial}{\partial \mathbf{x}^{(\mu)}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(\mu)}}\right)
\end{pmatrix},
\end{aligned}
$$

where each *sub-gradient* is differentiated with respect to $\mathbf{X}$. This results in μ^2 block partitions $(n \times n)$ of the Hessian matrix. The (i,j)-block matrix can be further expressed as follows:

$$
\begin{aligned}
&\frac{\partial}{\partial \mathbf{x}^{(i)}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(j)}}\right) = \frac{\partial}{\partial \mathbf{x}^{(i)}}\left(\frac{\partial \mathbf{y}^{(j)}}{\partial \mathbf{x}^{(j)}}\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{y}^{(j)}}\right)\\
&= \sum_{z=1}^{k}\frac{\partial}{\partial \mathbf{x}^{(i)}}\left(\frac{\partial f_z(\mathbf{x}^{(j)})}{\partial \mathbf{x}^{(j)}}\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_z(\mathbf{x}^{(j)})}\right)\\
&= \underbrace{\sum_{z=1}^{k}\frac{\partial}{\partial \mathbf{x}^{(i)}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_z(\mathbf{x}^{(j)})}\right)\nabla f_z(\mathbf{x}^{(j)})^{\top}}_{\mathbf{A}_{ij}}\\
&+ \underbrace{\sum_{z=1}^{k}\frac{\partial^2 f_z(\mathbf{x}^{(j)})}{\partial \mathbf{x}^{(i)}\partial \mathbf{x}^{(j)}}\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_z(\mathbf{x}^{(j)})}}_{\mathbf{B}_{ij}}.
\end{aligned}
$$

According to the differentiation above, each (i,j)-block matrix is a combination of two components: $\mathbf{A}_{ij}$ and $\mathbf{B}_{ij}$. Note that matrix $\mathbf{A}_{ij}$, $\frac{\partial}{\partial \mathbf{x}^{(i)}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_z(\mathbf{x}^{(j)})}\right)$ is a column vector of size n and stands for the sub-gradient of $\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_z(\mathbf{x}^{(j)})}$ at $\mathbf{x}^{(j)}$. In the following, we abbreviate $f_z(\mathbf{x}^{(i)})$ as $f_z^{(i)}$ and its gradient $\nabla f_z(\mathbf{x}^{(i)})$ as $\nabla f_z^{(i)}$.

4.2.3 The Hypervolume Hessian for $k = 2$

In the following, we define how to compute the two components $\mathbf{A}_{ij}$ and $\mathbf{B}_{ij}$ for $k = 2$.

First Component Matrix: $\mathbf{A}_{ij}$

Matrix $\mathbf{A}_{ij}$ has size $n \times n$ and can be expressed as a sum of outer products:

$$\mathbf{A}_{ij} = \sum_{z=1}^{k} \frac{\partial}{\partial \mathbf{x}^{(i)}} \left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_z^{(j)}} \right) \nabla f_z^{(j)\top}. \tag{4.13}$$

In the following lemma, a detailed expression of $\mathbf{A}_{ij}$ is given for the bi-objective case ($k = 2$). Without loss of generality, we assume that the objective vectors (and corresponding decision vectors) are arranged according to the ascending order of the first objective values.

Lemma 12. *Let $k = 2$ and all vectors $\mathbf{x}^{(i)}$, $i = 1 \ldots, \mu$, be mutually non-dominated, the first component $\mathbf{A}_{ij}$ is non-zero only if the block matrix is located on the main diagonal ($i = j$) or the first diagonal above/below the main diagonal ($|i - j| = 1$), and it can be written as:*

$$\mathbf{A}_{ij} = \begin{cases} \nabla f_2^{(j)} \nabla f_1^{(j)\top} + \nabla f_1^{(j)} \nabla f_2^{(j)\top} & \textit{if } i = j \\ -\nabla f_1^{(j+1)} \nabla f_2^{(j)\top} & \textit{if } i = j+1 \\ -\nabla f_2^{(j-1)} \nabla f_1^{(j)\top} & \textit{if } i = j-1 \\ \mathbf{0} & \textit{otherwise.} \end{cases} \tag{4.14}$$

Proof. Assume a fixed reference point $\mathbf{r} = (r_1, r_2)^\top$. To simplify the formulation, we denote $f_1^{(\mu+1)} := r_1$ and $f_2^{(0)} := r_2$. The partial derivative of the hypervolume indicator w.r.t. the objective value is derived in [36], which corresponds to the length of the steps of the attainment curve:

$$\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_1^{(j)}} = f_2^{(j)} - f_2^{(j-1)}, \; \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_2^{(j)}} = f_1^{(j)} - f_1^{(j+1)}. \tag{4.15}$$

It is clear that $\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_1^{(j)}}$ is a function of only $\mathbf{x}^{(j)}$ and $\mathbf{x}^{(j-1)}$ (similar argument holds for $\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_2^{(j)}}$). The gradient of the partial derivatives can be given, for example: $\frac{\partial}{\partial \mathbf{x}^{(j)}} \left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_z^{(j)}} \right) = \nabla f_2^{(j)}$. Such a gradient is nonzero for at least one objective function, when $i = j$, $i = j+1$ or $i = j-1$. By substituting the required gradients into Equation (4.13), the expression of $\mathbf{A}_{ij}$ can be obtained. □

Second Component Matrix: $\mathbf{B}_{ij}$

$\mathbf{B}_{ij}$ is a weighted sum of second order derivatives of the objective functions, where the weights are partial derivatives of the hypervolume indicator at each objective value (cf. Equation (2.55)). Note that the second order derivative $\frac{\partial^2 f_z^{(j)}}{\partial \mathbf{x}^{(i)} \partial \mathbf{x}^{(j)}}$ is not zero if and only if $i = j$:

$$\mathbf{H}_z^{(j)} := \frac{\partial^2 f_z^{(j)}}{\partial {\mathbf{x}^{(j)}}^2}$$

is the Hessian matrix of objective function f_z at point $\mathbf{x}^{(j)}$. Consequently, matrix $\mathbf{B}_{ij}$ can be written as:

$$\mathbf{B}_{ij} = \begin{cases} \sum_{z=1}^{k} \frac{\partial \mathcal{H}_\mathbf{F}(\mathbf{X})}{\partial f_z^{(j)}} \mathbf{H}_z^{(j)} & \text{if } i = j \\ \mathbf{0} & \text{if } i \neq j. \end{cases} \tag{4.16}$$

Note that $\frac{\partial \mathcal{H}_\mathbf{F}(\mathbf{X})}{\partial f_z^{(j)}}$ can be obtained from Equation (4.15).

4.3 The Hypervolume Newton Method

In this section, we state the population based Newton method for hypervolume maximization. This can be done, since we have now the gradient and Hessian of the hypervolume indicator. For this, we will first consider the unconstrained case and later on discuss first attempts to treat constrained problems.

4.3.1 Unconstrained Case

Given an unconstrained MOP and a population of μ individuals, the Newton step (or Newton function) is defined as follows:

$$\begin{aligned} \mathbf{N} :& \mathbb{R}^{\mu \cdot n} \to \mathbb{R}^{\mu \cdot n} \\ \mathbf{N}(\mathbf{X}) :&= \mathbf{X} - \nabla^2 \mathcal{H}_F(\mathbf{X})^{-1} \nabla \mathcal{H}_F(\mathbf{X}). \end{aligned} \tag{4.17}$$

The Newton direction for the entire population is given by

$$\nu_\mathbf{N} := -\nabla^2 \mathcal{H}_F(\mathbf{X})^{-1} \nabla \mathcal{H}_F(\mathbf{X}) \in \mathbb{R}^{\mu \cdot n}, \tag{4.18}$$

and the according directions for each individual $\mathbf{x}^{(i)}$ of $\mathbf{X}$ are denoted by $\nu_{\mathbf{N}}^{(i)} \in \mathbb{R}^n$, $i = 1, \ldots, \mu$.

Since the hypervolume indicator sub-gradient (Equation 2.55) for the strictly dominated sub-vector $\mathbf{x}^{(i)}$ of $\mathbf{X}$ is zero also its corresponding Newton direction will be zero. Consequently, such a point will remain stationary when applying the set-based Newton method. In the following, we restrict the approximation set $\mathbf{X}$ to the set of mutually non-dominated elements. For this purpose, we define the set $\tilde{\mathbf{X}}$ as the subset of $\mathbf{X}$ that contains all its non-dominated elements. The Hypervolume Newton Method (HNM) is thus defined as

$$\begin{aligned} \mathbf{X}_0 &\in \mathbb{R}^{\mu \cdot n} \\ \mathbf{X}_{i+1} &= \mathbf{N}(\tilde{\mathbf{X}}_i), \text{ for } \quad i = 0, 1, 2, \ldots \ . \end{aligned} \tag{4.19}$$

The pseudo code for HNM is shown in Algorithm 7. For the step size control we suggest to choose the initial step $\tilde{t}_0 = 1$ together with quadratic backtracking to satisfy the Wolfe Conditions [12] on the hypervolume. If automatic differentiation [12] is used to evaluate the (exact) gradient and the Hessian matrix at the iterate $\mathbf{X}_i$, the cost for each Newton step is given by $5\mu + (4 + 6n)\mu$ function evaluations.

Algorithm 7 $HNM(\mathbf{X}_0, I_{max}, tol_x)$

```
Require: An initial point $\mathbf{X}_0 \in \mathbb{R}^{\mu \cdot n}$, maximal number of iterations $I_{max}$, tolerance $tol_x \in \mathbf{R}_+$.
Ensure: The best found Newton iteration $\mathbf{X}_{(N)}$ according to $\mathcal{H}$.
1: for $i = 0$ to $I_{max}$ do
2:     Compute $\tilde{\mathbf{X}}_i$, $\nabla \mathcal{H}_{\mathbf{F}}(\tilde{\mathbf{X}}_i)$, $\nabla^2 \mathcal{H}_{\mathbf{F}}(\tilde{\mathbf{X}}_i)$
3:     Compute step size $t_i \in \mathbb{R}_+$
4:     $\mathbf{X}_{i+1} := \tilde{\mathbf{X}}_i - t_i \nabla^2 \mathcal{H}_F(\tilde{\mathbf{X}}_i)^{-1} \nabla \mathcal{H}_F(\tilde{\mathbf{X}}_i)$
5:     if $||\nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}_{i+1})|| < tol_x$ then
6:         return $\mathbf{X}_{(N)} := \mathbf{X}_{i+1}$
7:     end if
8: end for
9: return $\mathbf{X}_{(N)} := \mathbf{X}_{i+1}$
```

Example 6. In order to demonstrate the performance of the HNM, we consider the MOP1 (stated in (2.13)) where we choose as reference point $\mathbf{r} = (20, 20)^T$.

(a) We choose $\mu = 5$ and the initial population $\mathbf{X}_0$ as

$$\begin{aligned} \mathbf{X}_0 &= \{\mathbf{x}_0^{(1)}, \mathbf{x}_0^{(2)}, \mathbf{x}_0^{(3)}, \mathbf{x}_0^{(4)}, \mathbf{x}_0^{(5)}\} \\ &= \left\{ \begin{pmatrix} 0 \\ -2 \end{pmatrix}, \begin{pmatrix} 0.5 \\ -1.5 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} 1.5 \\ -0.5 \end{pmatrix}, \begin{pmatrix} 2 \\ 2 \end{pmatrix} \right\}. \end{aligned} \tag{4.20}$$

Figure 4.9 shows the performance of HNM both in decision and objective space. As it can be seen, the iterations quickly approach the optimal solution for $\mu = 5$ and a given reference point. This observation is confirmed in Table 4.1, where the hypervolume values, the norm of the gradients, and the error—measured in terms of the Hausdorff distance [6] of $\mathbf{X}_i$ and the optimal solution—are displayed for each iteration. The values indicate quadratic convergence.

(b) Next, we consider the same setting but using as initial population

$$\begin{aligned} \mathbf{X}_0 &= \{\mathbf{x}_0^{(1)}, \mathbf{x}_0^{(2)}, \mathbf{x}_0^{(3)}, \mathbf{x}_0^{(4)}, \mathbf{x}_0^{(5)}\} \\ &= \left\{ \begin{pmatrix} -0.12 \\ -1.57 \end{pmatrix}, \begin{pmatrix} 0.48 \\ -1.24 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} 1.32 \\ -0.26 \end{pmatrix}, \begin{pmatrix} 1.89 \\ -0.11 \end{pmatrix} \right\}. \end{aligned} \tag{4.21}$$

Figure 4.10 and Table 4.2 show the numerical results of HNM. In step 2, $\mathbf{x}_2^{(1)}$ gets dominated by $\mathbf{x}_2^{(3)}$. The iteration thus continues with $\tilde{\mathbf{X}}_2 \in \mathbb{R}^{4 \cdot 2}$, i.e., with a set of 4 2-dimensional vectors. HNM converges (again quadratically) toward the optimal hypervolume population, albeit for population size $\mu = 4$.

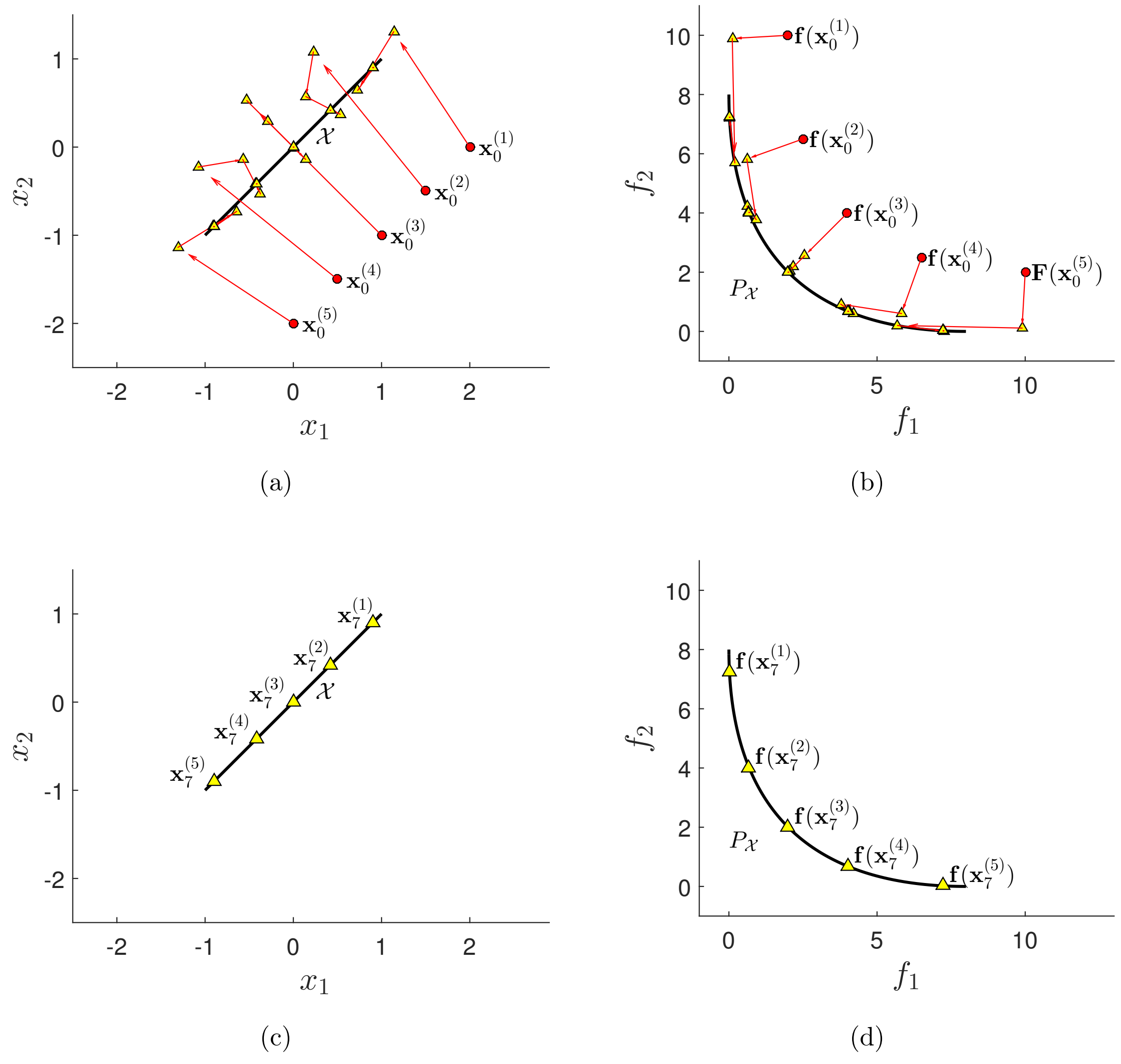


Figure 4.9: Numerical result of HNM on MOP1 using (4.20) as initial population. Above: the iterations in decision and objective space. Below: the optimal solution and its image for $\mu = 5$ and $\mathbf{r}$.

Table 4.1: Hypervolume values, error, and hypervolume gradients for application of HNM on MOP1 using (4.20) (compare to Figure 4.9).

Iter	$\mathcal{H}$	**Error (HNM)**	$\|\|\nabla\mathcal{H}_{\mathbf{F}}\|\|$
0	306.500000000000	76.569536787325	48.826222462934
1	369.562245664015	13.507291123309	21.062836757428
2	379.065240846390	4.004295940935	13.997313947531
3	382.734095975048	0.335440812277	2.879962272550
4	383.068028174229	0.001508613095	0.199992009700
5	383.069536709027	0.000000078297	0.001265863184
6	383.069536787325	0.000000000000	0.000000101250
7	383.069536787325	0.000000000000	0.000000000000

Table 4.2: Hypervolume values, error, and hypervolume gradients for application of HNM on MOP1 using (4.21) for $\mathbf{X}_0$ (compare to Figure 4.10).

Iter	$\mathcal{H}$	**Error (HNM)**	$\|\|\nabla\mathcal{H}_{\mathbf{F}}\|\|$
$\mu = 5$			
0	321.548361341053	61.521175446272	52.900629002755
1	376.616155150423	6.453381636902	14.685530752574
$\mu = 4$			
2	373.544616989121	9.524919798204	2.013226644050
3	380.698229481420	2.371307305905	0.110423150875
4	380.698531798899	2.371004988425	0.000213267251
5	380.698531800725	2.371004986600	0.000000002021
6	380.698531800725	2.371004986600	0.000000000000
7	380.698531800725	2.371004986600	0.000000000000

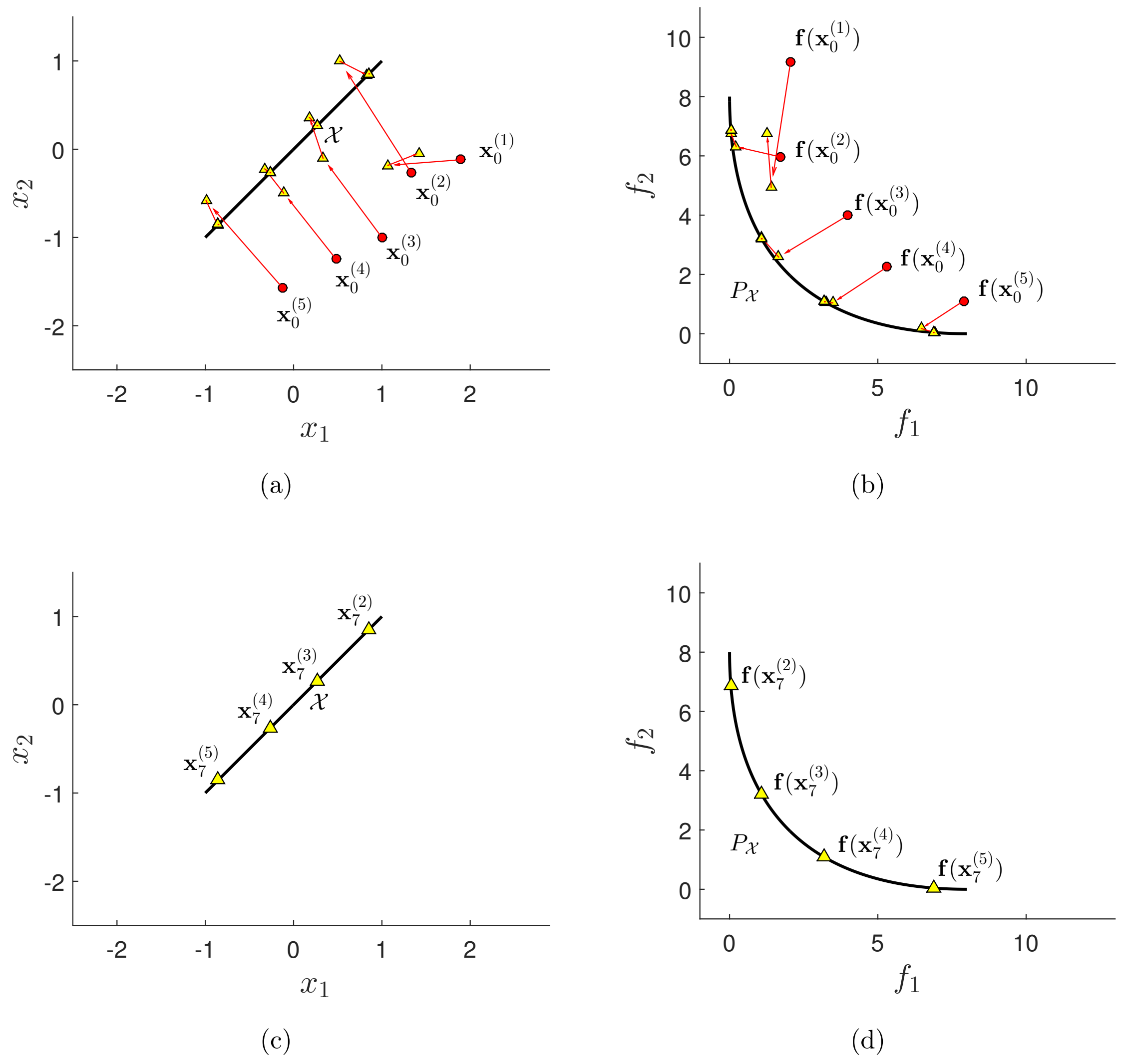


Figure 4.10: Numerical result of HNM on MOP1 using (4.21) as initial population. Above: the iterations in decision and objective space. Below: the optimal solution and its image for $\mu = 4$ and $\mathbf{r}$.

4.3.2 Inequality Constraint Case

Next, we consider inequality constrained MOPs of the form

$$\begin{aligned} &\min_{\mathbf{x}\in\mathbb{R}^n} \mathbf{f}(\mathbf{x}), \\ &\text{s.t } g_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, p. \end{aligned} \tag{4.22}$$

For the treatment of such problems, we propose to utilize a penalization approach that transforms the original (constrained) MOP into an auxiliary unconstrained one. In the current context, the related unconstrained problem reads as:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \mathbf{f}(\mathbf{x}) + P(\mathbf{x})\mathbf{c}, \tag{4.23}$$

where $\mathbf{c} = (c, .., c)^\top \in \mathbb{R}^m$, $c > 0$ a given (large) constant, and

$$p(\mathbf{x}) := \sum_{i=1}^{p} \max(0, g_i(\mathbf{x}))^2 \tag{4.24}$$

the penalization function. To solve inequality constrained MOPs of the form (4.22), HNM is thus applied on the unconstrained problem (4.23) as described above. To avoid convergence toward spurious solutions, the value of c cannot remain fixed during the computations. Instead, we need a sequence $c_u > 0$ with $\lim_{i\to\infty} c_u = \infty$. In our computations, we have chosen $c_0 := 10$ and in each Newton step the value is increased by a factor of 10, i.e., $c_u := 10^{u+1}$.

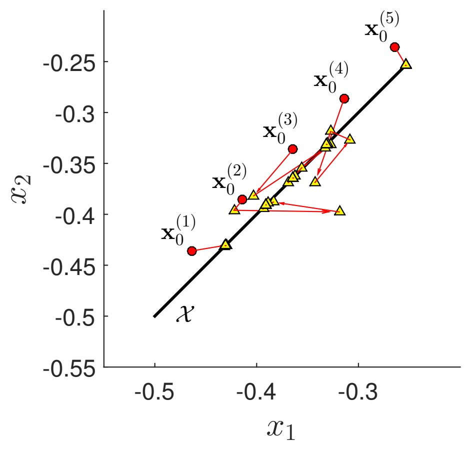
Example 7. We reconsider again MOP1 but additionally impose the following box constraint

$$x_i \in [-0.5, -0.25], \quad i = 1, 2. \tag{4.25}$$

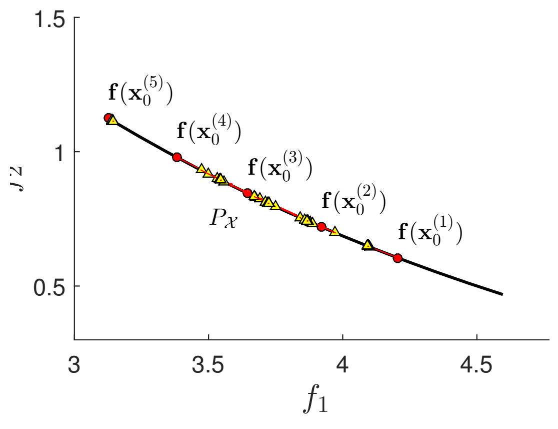
We have chosen $\mathbf{r} = (20, 20)^T$ and $\mu = 5$ as before, and the initial population as

$$\begin{aligned} \mathbf{X}_0 &= \{\mathbf{x}_0^{(1)}, \mathbf{x}_0^{(2)}, \mathbf{x}_0^{(3)}, \mathbf{x}_0^{(4)}, \mathbf{x}_0^{(5)}\} \\ &= \left\{ \begin{pmatrix} -0.46 \\ -0.43 \end{pmatrix}, \begin{pmatrix} -0.41 \\ -0.38 \end{pmatrix}, \begin{pmatrix} -0.36 \\ -0.33 \end{pmatrix}, \begin{pmatrix} -0.31 \\ -0.28 \end{pmatrix}, \begin{pmatrix} -0.26 \\ -0.23 \end{pmatrix} \right\}. \end{aligned} \tag{4.26}$$

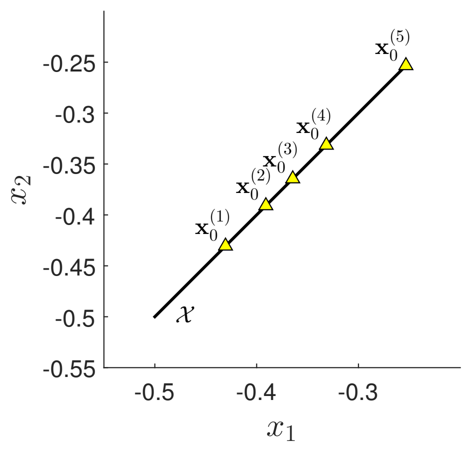
Figure 4.11 shows a numerical result of HNM for this setting. Table 4.3 indicates that the iterations converge, but that the convergence speed is only linearly which is due to the penalization approach.



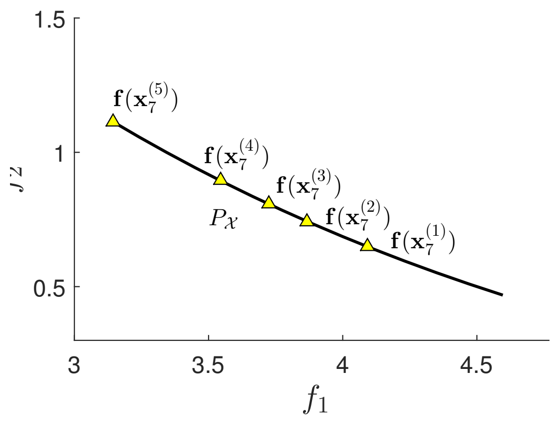
(a)



(b)



(c)



(d)

Figure 4.11: Numerical result of HNM on MOP1 together with the box constraints (4.25) using (4.26) as initial population. Above: the iterations in decision and objective space. Below: the optimal solution and its image for $\mu = 5$ and $\mathbf{r}$.

Table 4.3: Hypervolume values, error, and hypervolume gradients for the application of HNM on MOP1 with the box constraint (4.25) using (4.26) for $\mathbf{X}_0$ (compare to Figure 4.11).

Iter	$\mathcal{H}$	**Error (HNM)**	$\|\|\nabla\mathcal{H}_{\mathbf{F}}\|\|$
1	325.901249212877	0.020663552304	1.212734648416
2	325.920707182928	0.001205582253	0.051243303186
3	325.921835652869	0.000077112312	0.004631644572
4	325.921906470408	0.000006294772	0.000422880099
5	325.921912146183	0.000000618997	0.000039564993
6	325.921912707233	0.000000057948	0.000003700931
7	325.921912759765	0.000000005415	0.000000346099

4.3.3 Equality Constraint Case

In the case of having an equality constrained MOP of the following form:

$$\begin{aligned} &\min_{\mathbf{x}\in\mathbb{R}^n} \mathbf{f}(\mathbf{x}), \\ &\text{s.t } h_j(\mathbf{x}) = 0, \quad j = 1, \ldots, q, \end{aligned} \tag{4.27}$$

we can redefine the previous problem by using the $\mu \cdot n$-vector notation for a given MOP as follows:

$$\begin{aligned} &\min_{\mathbf{X}\subset\mathbb{R}^n} \mathbf{F}(\mathbf{X}). \\ &\text{s.t } \mathbf{h}(\mathbf{X}) = 0 \end{aligned} \tag{4.28}$$

Then, related SOP according to the hypervolume indicator is given by

$$\begin{aligned} &\max_{\mathbf{X}\subset\mathbb{R}^n} \mathcal{H}_{\mathbf{F}}(\mathbf{X}), \\ &\text{s.t } \mathbf{h}(\mathbf{X}) = \mathbf{0} \end{aligned} \tag{4.29}$$

where $|\mathbf{X}| = \mu$ and $\mathbf{X} = \left(\mathbf{x}^{(1)^T}, \ldots, \mathbf{x}^{(\mu)^T}\right)^T$.

The constraints $\mathbf{h}(\mathbf{X})$ are considered as follows:

$$\mathbf{h} : \mathbb{R}^{\mu \cdot n} \rightarrow \mathbb{R}^{\mu \cdot q}$$

$$\mathbf{h}(\mathbf{X}) = \begin{pmatrix} h_1(\mathbf{X}) \\ \vdots \\ h_q(\mathbf{X}) \end{pmatrix} = \begin{pmatrix} h_1(\mathbf{x}^{(1)}) \\ \vdots \\ h_q(\mathbf{x}^{(1)}) \\ \vdots \\ \vdots \\ h_1(\mathbf{x}^{(\mu)}) \\ \vdots \\ h_p(\mathbf{x}^{(\mu)}) \end{pmatrix} = \begin{pmatrix} \bar{h}_{1,1}(\mathbf{X}) \\ \vdots \\ \bar{h}_{1,p}(\mathbf{X}) \\ \vdots \\ \vdots \\ \bar{h}_{\mu,1}(\mathbf{X}) \\ \vdots \\ \bar{h}_{\mu,q}(\mathbf{X}) \end{pmatrix}, \tag{4.30}$$

where $\bar{h}_{i,j} : \mathbb{R}^{\mu \cdot n} \rightarrow \mathbb{R}$.

To tackle an equality constrained MOP, we propose to use the Lagrange multipliers and state the problem using the Karush-Kuhn-Tucker equations of (4.29) which read as follows

$$\begin{aligned} \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}) + \sum_{i=1}^{\mu \cdot q} \lambda_i \nabla \bar{h}_i(\mathbf{X}) = 0 \\ \mathbf{h}(\mathbf{X}) = 0 \end{aligned}. \tag{4.31}$$

The KKT equations leads to the root finding problem $G : \mathbb{R}^{\mu \cdot (n+q)} \rightarrow \mathbb{R}^{\mu \cdot (n+q)}$

$$G(\mathbf{X}, \lambda) = \begin{pmatrix} \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}) + \sum_{i=1}^{\mu \cdot q} \lambda_i \nabla \bar{h}_i(\mathbf{X}) \\ \mathbf{h}(\mathbf{X}) \end{pmatrix} = 0. \tag{4.32}$$

The derivative of G is given by

$$DG(\mathbf{X}, \lambda) = \begin{pmatrix} \nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) + \sum_{i=1}^{\mu \cdot q} \lambda_i \nabla^2 \bar{h}_i(\mathbf{X}) & \mathbf{H}(\mathbf{X})^T \\ \mathbf{H}(\mathbf{X}) & 0 \end{pmatrix} \in \mathbb{R}^{\mu \cdot (n+q) \times \mu \cdot (n+q)}, \tag{4.33}$$

where

$$\mathbf{H}(\mathbf{X}) = \begin{pmatrix} \nabla \bar{h}_{1,1}(\mathbf{X})^T \\ \vdots \\ \nabla \bar{h}_{\mu,q}(\mathbf{X})^T \end{pmatrix}. \tag{4.34}$$

Then, the Newton method for $G(\mathbf{X}, \lambda) = 0$ is given by $N_G : \mathbb{R}^{\mu\cdot(n+q)} \rightarrow \mathbb{R}^{\mu\cdot(n+q)}$ where

$$N_G(\mathbf{X}, \lambda) = (\mathbf{X}, \lambda) - DG(\mathbf{X}, \lambda)^{-1} G(\mathbf{X}, \lambda). \tag{4.35}$$

In the following, we present some example of the performance of the hypervolume Newton method for equality constrained MOPs (HNMEC).

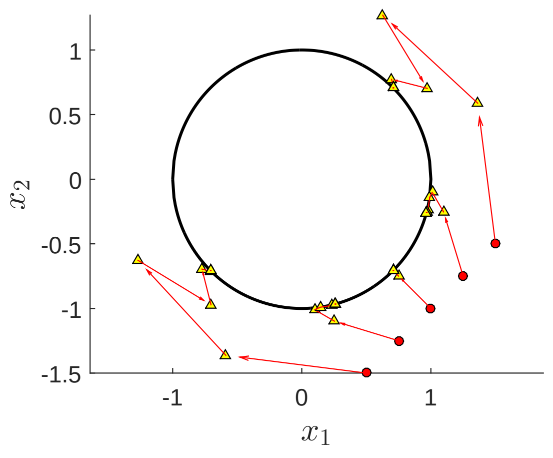
Example 8. By considering MOP1 but additionally we integrate the following equality constrained

$$h(\mathbf{x}) = x_1^2 + x_2^2 - 1 = 0. \tag{4.36}$$

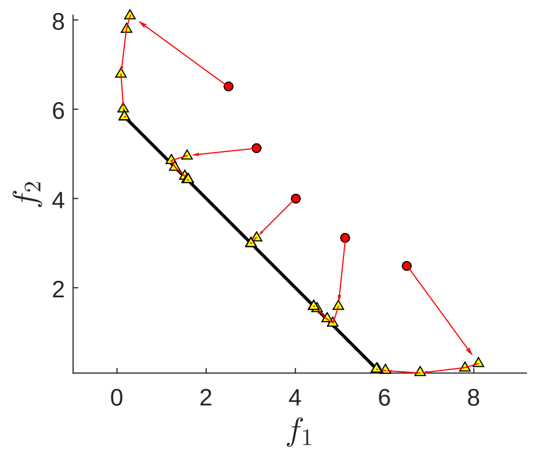
For this problem, we choose $\mathbf{r} = (20, 20)^T$, $\mu = 5$, and the initial population from an infeasible region as

$$\begin{aligned} \mathbf{X}_0 =& \{\mathbf{x}_0^{(1)}, \mathbf{x}_0^{(2)}, \mathbf{x}_0^{(3)}, \mathbf{x}_0^{(4)}, \mathbf{x}_0^{(5)}\} \\ =& \left\{ \begin{pmatrix} 0.5 \\ -1.5 \end{pmatrix}, \begin{pmatrix} 0.75 \\ -1.25 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} 1.25 \\ -0.75 \end{pmatrix}, \begin{pmatrix} 1.5 \\ -0.5 \end{pmatrix} \right\}, \end{aligned} \tag{4.37}$$

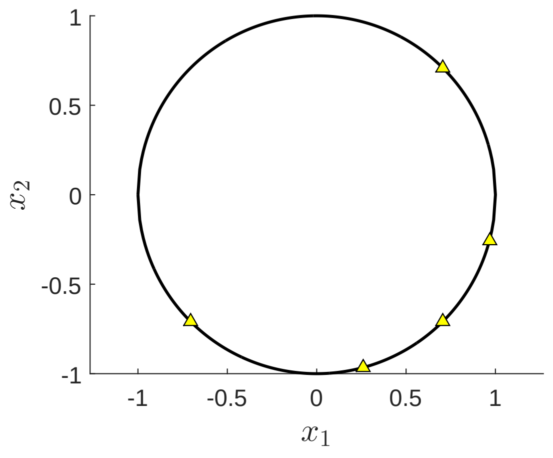
as the initial population $\mathbf{X}_0$. Figure 4.12 shows the performance of HNM both in decision and objective space. We notice that the method quickly approach to the optimal solution for $\mu = 5$ and the given reference point. Table 4.4 includes the error—measured in terms of the Hausdorff distance of $\mathbf{X}_i$ and the optimal solution—and the norm of the equality constraints for each iteration.



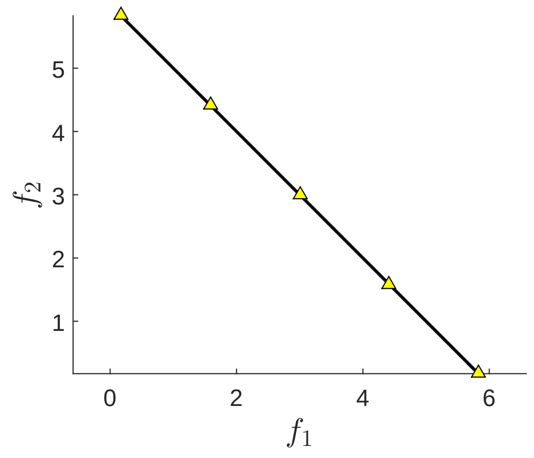
(a)



(b)



(c)



(d)

Figure 4.12: Numerical result of HNM on MOP1 together with the equality constraint 4.36 using 4.37 as an starting point. Above: the iterations in decision and objective space. Below: the optimal solution and its image for $\mu = 5$ and $\mathbf{r}$.

Table 4.4: Iteration, error values, and the norm of the equality constraints for the application of HNMEC on MOP1 with the equality constraint of Equation (4.36) using (4.37) for an starting point (compare to Figure 4.12).

Iteration	Error	$\|\|h(\mathbf{X})\|\|$
0	1.4442252032	2.8339460122
1	0.6663222123	1.7481771055
2	0.5674712326	1.4205720349
3	0.2648013579	0.6206272816
4	0.0671014459	0.1157213015
5	0.0053679372	0.0069536261
6	0.0000346286	0.0000411696
7	0.0000000013	0.0000000017
8	0.0000000000	0.0000000000

Example 9. Another example to demonstrate the performance of the HNMEC, it is considering the following equality constrained MOP constructed by using the ZDT1:

$$\begin{aligned} f_1(\mathbf{x}) &= x_1 \\ f_2(\mathbf{x}) &= g(\mathbf{x})(2 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})}) \\ g(\mathbf{x}) &= 1 + \frac{9}{n-1}\sum_{i=1}^{n} x_i^2 \\ s.t.& \\ h(\mathbf{x}) &= (x_1 - 0.5)^2 + (x_2 - 0.4)^2 - 0.25 = 0. \end{aligned} \tag{ECMOP1}$$

In this case, we choose $\mathbf{r} = (11, 11)^T$ as the reference point and

$$\begin{aligned} \mathbf{X}_0 &= \left\{\mathbf{x}_0^{(1)}, \mathbf{x}_0^{(2)}, \mathbf{x}_0^{(3)}\right\} \\ &= \left\{ \begin{pmatrix} 0.007 \\ 0.316 \end{pmatrix}, \begin{pmatrix} 0.078 \\ 0.130 \end{pmatrix}, \begin{pmatrix} 0.7074 \\ -0.05498 \end{pmatrix} \right\} \end{aligned} \tag{4.38}$$

as the initial population $\mathbf{X}_0$. Figure 4.13 shows the performance of HNM both in decision and objective space. We notice that the method quickly approaches to the optimal solution for $\mu = 3$ and the given reference point. Table 4.5 includes the error—measured in terms of the Hausdorff distance of $\mathbf{X}_i$ and the optimal solution—and the norm of the equality constraints for each iteration.

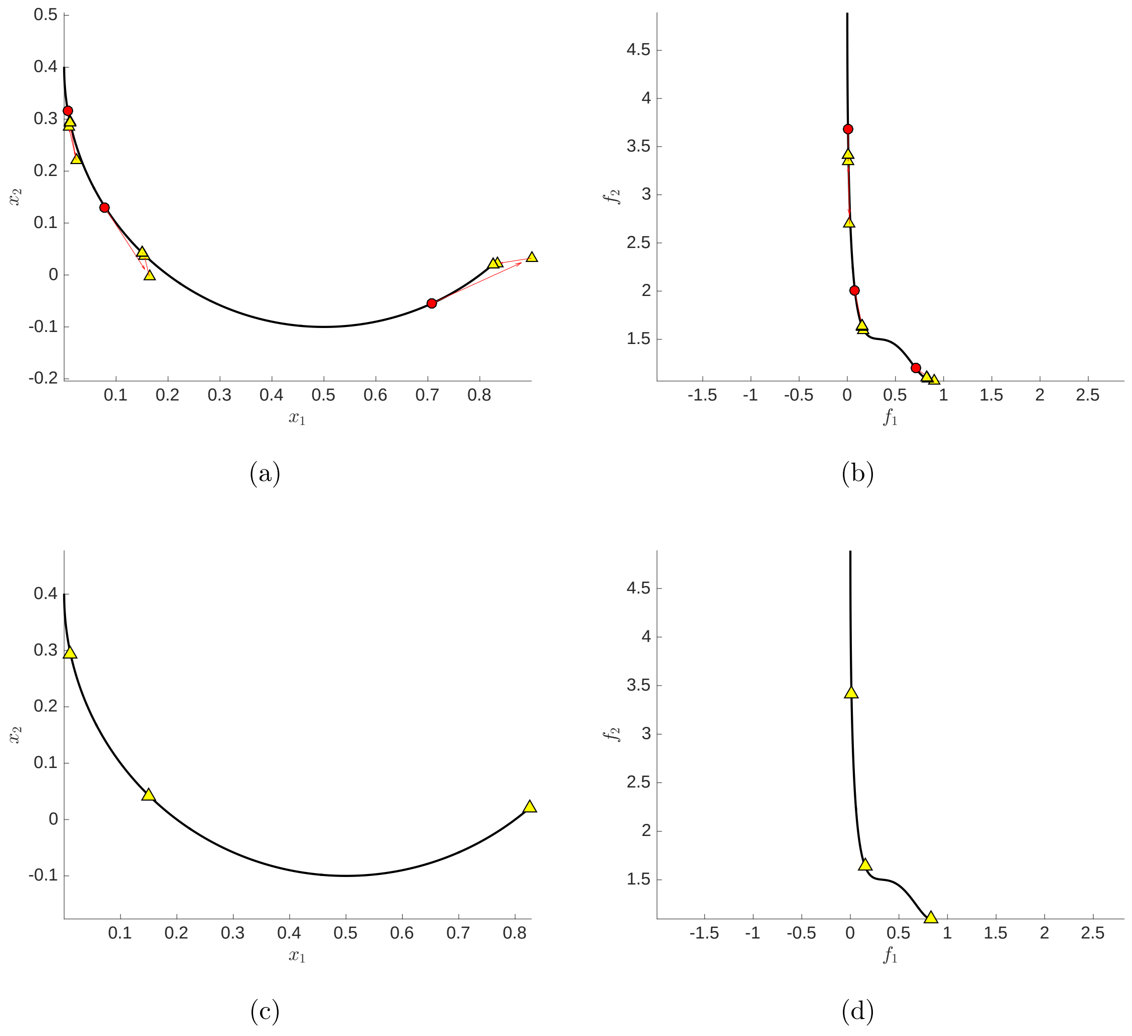


Figure 4.13: Numerical result of HNM on ZDT1 together with an equality constraint using (4.38) as an starting point. Above: the iterations in decision and objective space. Below: the optimal solution and its image for $\mu = 3$ and $\mathbf{r}$.

Example 10. Finally, in order to see the performance of the HNMEC, when the dimension of the decision space grows ($n = 3$), we introduced the next equality

Table 4.5: Iteration, error values, and the norm of the equality constraints for the application of HNMEC on ZDT1 with an equality constraint using (4.38) for an starting point (compare to Figure 4.13).

Iteration	Error	$\|\|h(\mathbf{X})\|\|$
0	0.1406683928	0.0009898211
1	0.0744211059	0.0517750630
2	0.0089787918	0.0064097784
3	0.0001241347	0.0001138679
4	0.0000004208	0.0000000222
5	0.0000000000	0.0000000000

constrained MOP:

$$\begin{aligned} f_j(\mathbf{x}) &= \sum_{i=1}^{n}(x_i - a_i^j)^2 + (x_j - a_j^j)^4 \\ s.t. & \\ h(\mathbf{x}) &= 0.5x_1 - x_2 = 0. \end{aligned} \tag{ECMOP2}$$

For the latter problem, we choose $n = 3$, $\mathbf{r} = (30, 30)^T$ as the reference point and

$$\begin{aligned} \mathbf{X}_0 &= \left\{\mathbf{x}_0^{(1)}, \mathbf{x}_0^{(2)}, \mathbf{x}_0^{(3)}\right\} \\ &= \left\{(-0.8727, -0.4364, -0.3273)^T, (0.4364, 0.2182, 0.3)^T, (1.473, 0.7364, 0.709)^T\right\}, \end{aligned} \tag{4.39}$$

as the initial population $\mathbf{X}_0$. Figure 4.14 shows the performance of HNM both in decision and objective space.Once again the method quickly approach to the optimal solution for $\mu = 3$. Table 4.6 includes the error—measured in terms of the Hausdorff distance of $\mathbf{X}_i$ and the optimal solution— and the norm of the equality constraints for each iteration.

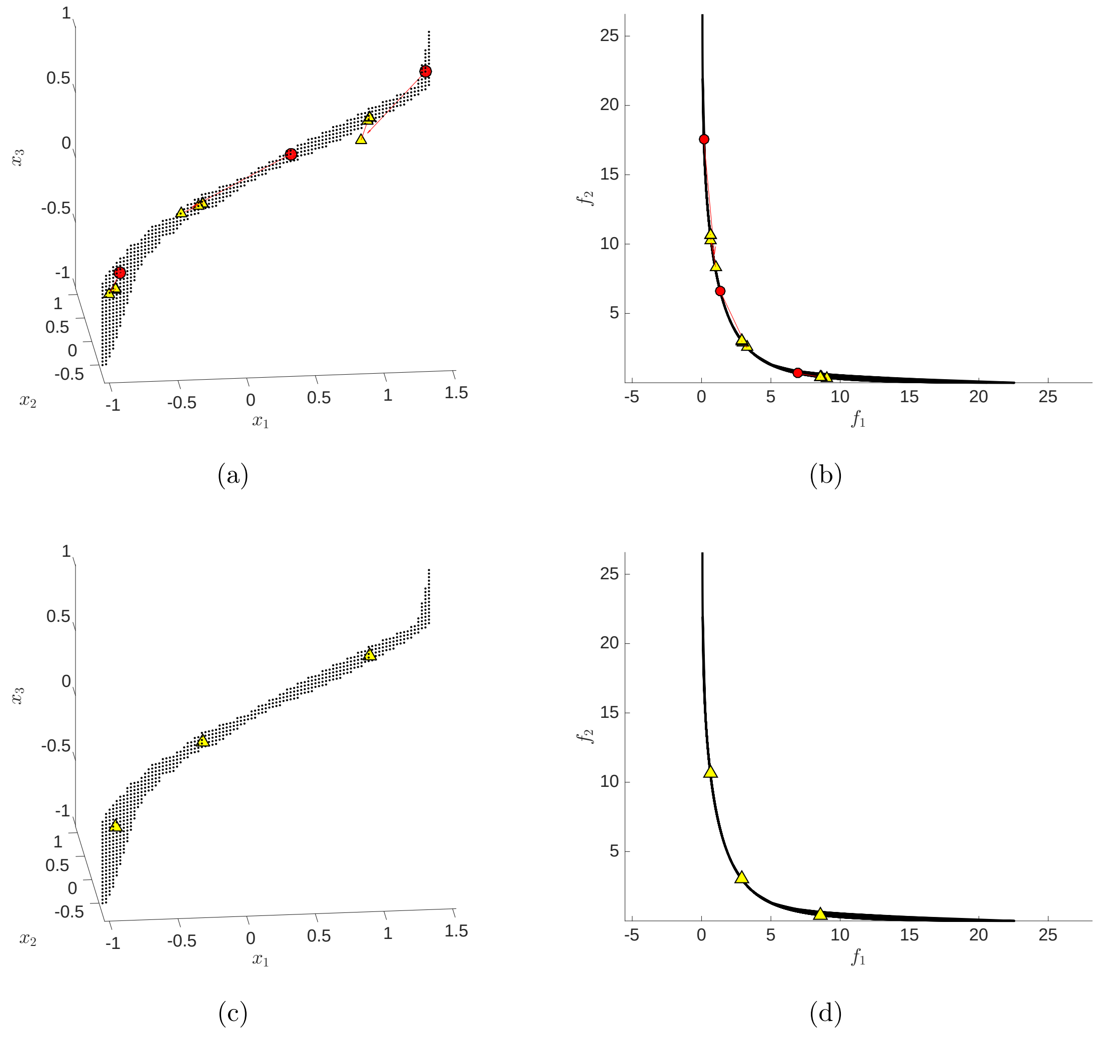


Figure 4.14: Numerical result of HNM on problem ECMOP2 using 4.39 as an starting point. Above: the iterations in decision and objective space. Below: the optimal solution and its image for $\mu = 3$ and $\mathbf{r}$.

4.3.4 Investigation of the Hypervolume Newton Method for Equality Constraints

A point $\mathbf{X}^* \subset \mathbb{R}^n$ is optimal for Equation (4.29) if and only if there is a $\lambda^* \in \mathbb{R}^{\mu \cdot p}$ such that

$$\mathbf{h}(\mathbf{X}^*) = \mathbf{0} \tag{4.40}$$

Table 4.6: Iteration, error values, and the norm of the equality constraints for the application of HNMEC on problem ECMOP2 using (4.39) for an starting point (compare to Figure 4.14).

Iteration	Error	$\|\|h(\mathbf{X})\|\|$
0	0.7915615680	0.0001118034
1	0.1892124982	0.0000000000
2	0.0392555569	0.0000000000
3	0.0009810937	0.0000000000
4	0.0000009242	0.0000000000
5	0.0000000000	0.0000000000
6	0.0000000000	0.0000000000

and

$$\nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}^*) + \nabla \mathbf{h}(\mathbf{X}^*)\lambda^* = \mathbf{0}. \tag{4.41}$$

We refer to $\mathbf{h}(\mathbf{X}^*) = \mathbf{0}$ as the primal feasibility equations where the equality constraints are linear and $\nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}^*) + \nabla \mathbf{h}(\mathbf{X}^*)\lambda^* = \mathbf{0}$ as the dual feasibility equation.

The Newton step

In order to define the Newton step, we use a second order approximation. To derive the Newton step $\nu_{\mathbf{N}}$ for the equality constrained problem stated in Equation (4.29) at the feasible point $\mathbf{X}$, we replace the objective with its second order Taylor approximation near $\mathbf{X}$

$$\begin{aligned} \max\ & \tilde{\mathcal{H}_{\mathbf{F}}}(\mathbf{X}+\nu_{\mathbf{N}}) = \mathcal{H}_{\mathbf{F}}(\mathbf{X}) + \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X})^T \nu_{\mathbf{N}} + \frac{1}{2}{\nu_{\mathbf{N}}}^T \nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X})\nu_{\mathbf{N}}. \\ & s.t. \\ & \quad \mathbf{h}(\mathbf{X}+\nu_{\mathbf{N}}) = \mathbf{0}. \end{aligned} \tag{4.42}$$

Equation (4.42) is a convex quadratic maximization problem with equality constraints, then the Newton step $\nu_{\mathbf{N}}$ is characterized by

$$\begin{pmatrix} \nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) & \nabla \mathbf{h}(\mathbf{X})^T \\ \nabla \mathbf{h}(\mathbf{X}) & \mathbf{0} \end{pmatrix} \begin{pmatrix} \nu_{\mathbf{N}} \\ \lambda \end{pmatrix} = \begin{pmatrix} -\nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}) \\ \mathbf{0} \end{pmatrix}, \tag{4.43}$$

where λ is the associated dual variable for the quadratic problem.

The Newton Decrement

The Newton decrement for the problem stated in Equation (4.29) denoted by $\psi(\mathbf{X})$ is given by

$$\psi(\mathbf{X}) = ({\nu_{\mathbf{N}}}^T \nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) \nu_{\mathbf{N}})^{1/2}, \tag{4.44}$$

where $\psi(\mathbf{X})$ is the norm of the Newton step, in the norm determined by the Hypervolume Hessian matrix.

Let

$$\tilde{\mathcal{H}_{\mathbf{F}}}(\mathbf{X} + \nu) = \mathcal{H}_{\mathbf{F}}(\mathbf{X}) + \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X})^T \nu + \frac{1}{2} \underbrace{\nu^T \nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) \nu}_{\psi(\mathbf{X})^2} \tag{4.45}$$

be the second-order Taylor approximation of $\mathcal{H}_F$ at $\mathbf{X}$. The difference between $\mathcal{H}_F(\mathbf{X})$ and the minimum of the second-order model satisfies

$$\mathcal{H}_F(\mathbf{X}) - \inf \{ \tilde{\mathcal{H}_{\mathbf{F}}}(\mathbf{X} + \nu) \mid \mathbf{h}(\mathbf{X} + \nu) = \mathbf{0} \} = \psi(\mathbf{X})^2/2. \tag{4.46}$$

This means that $\psi(\mathbf{X})^2/2$ gives an estimate of $\mathcal{H}_{\mathbf{F}}(\mathbf{X}) - p^*$ based on the quadratic model at $\mathbf{X}$ and also that $\psi(\mathbf{X})$ serves as the basis of a good stopping criterion. The Newton decrement comes up in the line search as well, since the directional derivative of $\mathcal{H}_{\mathbf{F}}$ in the direction $\nu_{\mathbf{N}}$ is

$$\frac{d}{dt} \tilde{\mathcal{H}_{\mathbf{F}}}(\mathbf{X} + t\nu_{\mathbf{N}}) \Big|_{t=0} = \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X})^T \nu_{\mathbf{N}} = -\psi(\mathbf{X})^2. \tag{4.47}$$

Feasible ascent direction

Supposing that $\mathbf{h}(\mathbf{X}) = \mathbf{0}$. $\nu \in \mathbb{R}^{\mu \cdot n}$ is a feasible direction if $\nabla \mathbf{h}(\mathbf{X}) \nu = \mathbf{0}$. In this case. every point of the form $\mathbf{X} + \nu$ is also feasible, i.e., $\mathbf{h}(\mathbf{X} + \nu) = \mathbf{0}$. ν is a ascent direction for $\mathcal{H}_{\mathbf{F}}$ at $\mathbf{X}$, if for a small $t > 0$, $\mathcal{H}_{\mathbf{F}}(\mathbf{X} + t\nu) > \mathcal{H}_{\mathbf{F}}(\mathbf{X})$. The Newton step is always a feasible descent direction (except when $\mathbf{X}$) is optimal, since $\nu_{\mathbf{N}} = \mathbf{0}$). The second set of equations from Equation (4.43) that define $\nu_{\mathbf{N}}$ are $\nabla \mathbf{h}(\mathbf{X}) \nu_{\mathbf{N}} = \mathbf{0}$, which shows it i a feasible direction; that it is a ascent direction follows form Equation (4.47).

Infeasible start Newton method

Here, we generalize the Hypervolume Newton Method for equality constraints where initial points are not feasible, i.e., for points $\mathbf{X}$ such that $\mathbf{h}(\mathbf{X}) \neq \mathbf{0}$.

First, we recall the optimality conditions for the equality constrained maximization problem stated in Equations (4.40) and (4.41).

Let $\mathbf{X}$ denote the current point, which we do not assume to be feasible. The goal is to find a step ν' so that $\mathbf{X} + \nu'$ satisfies at least approximately the optimality conditions, i.e., $\mathbf{X} + \nu' \approx \mathbf{X}^*$. To do this, we substitute $\mathbf{X} + \nu'$ for $\mathbf{X}^*$ and λ' for λ^* in Equations (4.40) and (4.41), and use the first-order approximation

$$\nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X} + \nu') \approx \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}) + \nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) \nu' \tag{4.48}$$

for the gradient to obtain

$$\mathbf{h}(\mathbf{X} + \nu') = \mathbf{0}, \text{ and } \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}) + \nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) \nu' + \nabla \mathbf{h}(\mathbf{X})^T \lambda' = 0. \tag{4.49}$$

This is a set of linear equations for ν' and λ',

$$\begin{pmatrix} \nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) & \nabla \mathbf{h}(\mathbf{X})^T \\ \nabla \mathbf{h}(\mathbf{X}) & \mathbf{0} \end{pmatrix} \begin{pmatrix} \nu_{\mathbf{N}} \\ \lambda' \end{pmatrix} = - \begin{pmatrix} \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}) \\ \mathbf{h}(\mathbf{X}) \end{pmatrix}. \tag{4.50}$$

The equations are similar to Equation (4.43) but slightly different in the second block of the right-hand-side that contains $\mathbf{h}(\mathbf{X})$, which is the residual vector for the linear equality constraints. In the case that $\mathbf{X}$ is feasible, the residual vanishes, and Equation (4.50) is reduced to Equation (4.43) that define the standard Newton step at a feasible point $\mathbf{X}$.

An interpretation (in terms of a primal-dual method for equality constraints) of Equation (4.50) is given in the following: by a primal-dual, we mean one in which we update both the primal variable $\mathbf{X}$, and the dual variable λ, in order to approximately satisfy the optimality conditions. We express the optimality conditions as $r(\mathbf{X}^*, \lambda^*) = \mathbf{0}$, where $r : \mathbf{R}^{\mu \cdot n} \times \mathbf{R}^{\mu \cdot q} \to \mathbf{R}^{\mu \cdot n} \times \mathbf{R}^{\mu \cdot q}$ is defined as

$$r(\mathbf{X}, \lambda) = (r_{dual}(\mathbf{X}, \lambda), r_{pri}(\mathbf{X}, \lambda)). \tag{4.51}$$

where

$$r_{dual}(\mathbf{X}, \lambda) = \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}) + \nabla \mathbf{h}(\mathbf{X})^T \lambda \tag{4.52}$$

and

$$r_{pri}(\mathbf{X}, \lambda) = \mathbf{h}(\mathbf{X}) \tag{4.53}$$

are the dual residual and primal residual, respectively. The first-order Taylor approximation of r, near our current estimate $\mathbf{Y}$, is

$$r(\mathbf{Y} + \mathbf{z}) \approx \tilde{r}(\mathbf{Y} + \mathbf{z}) = r(\mathbf{Y}) + Dr(\mathbf{Y})\mathbf{z}, \tag{4.54}$$

where $Dr(\mathbf{Y}) \in \mathbb{R}^{(\mu(n+q))\times(\mu(n+q))}$ is the derivative of r, evaluated at $\mathbf{Y}$. We define the primal-dual Newton step $\nu_{\mathbf{Y}_{pd}}$ as the step $\mathbf{z}$ for which the Taylor approximation $\tilde{r}(\mathbf{Y} + \mathbf{z})$ vanishes, i.e.,

$$Dr(\mathbf{Y})\nu_{\mathbf{Y}_{pd}} = -r(\mathbf{Y}). \tag{4.55}$$

Note that we consider both $\mathbf{X}$ and λ as variables: $\nu_{\mathbf{Y}_{pd}} = (\nu_{\mathbf{X}_{pd}}, \nu_{\lambda_{pd}})$ gives both a primal and dual step.

Evaluating the derivative of r, we can express Equation (4.55) as

$$\begin{pmatrix} \nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) & \nabla \mathbf{h}(\mathbf{X})^T \\ \nabla \mathbf{h}(\mathbf{X}) & \mathbf{0} \end{pmatrix} \begin{pmatrix} \nu_{\mathbf{X}_{pd}} \\ \nu_{\lambda_{pd}} \end{pmatrix} = -\begin{pmatrix} r_{dual} \\ r_{pri} \end{pmatrix} = -\begin{pmatrix} \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}) + \nabla \mathbf{h}(\mathbf{X})^T \lambda \\ \mathbf{h}(\mathbf{X}) \end{pmatrix}. \tag{4.56}$$

Writing $\lambda + \nu_{\lambda_{pd}}$ as λ^+, we can express this as

$$\begin{pmatrix} \nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) & \nabla \mathbf{h}(\mathbf{X})^T \\ \nabla \mathbf{h}(\mathbf{X}) & \mathbf{0} \end{pmatrix} \begin{pmatrix} \nu_{\mathbf{X}_{pd}} \\ \lambda^+ \end{pmatrix} = -\begin{pmatrix} \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}) \\ \mathbf{h}(\mathbf{X}) \end{pmatrix}, \tag{4.57}$$

which is exactly the same according Equation (4.50). The solutions of Equations (4.50), (4.56) and (4.57) are therefore related as

$$\nu_{\mathbf{N}} = \nu_{\mathbf{X}_{pd}} \text{ and } \lambda' = \lambda^+ = \lambda + \nu_{\lambda_{pd}}. \tag{4.58}$$

This shows that the infeasible Newton step is the same as the primal part of the primal-dual step, and the associated dual vector λ' is the updated primal-dual variable $\lambda^+ = \lambda + \nu_{\lambda_{pd}}$.

The Newton direction, at an infeasible point, is not necessarily a descent direction for $\mathcal{H}_{\mathbf{F}}$. From Equation (4.50), we note that

$$\begin{aligned} \frac{d}{dt}\tilde{\mathcal{H}_{\mathbf{F}}}(\mathbf{X}+t\nu')\Big|_{t=0} &= \nabla\mathcal{H}_{\mathbf{F}}(\mathbf{X})^T\nu' \\ &= -\nu'^T(\nabla^2\mathcal{H}_{\mathbf{F}}(\mathbf{X})\nu' + \nabla\mathbf{h}(\mathbf{X})^T\lambda) \\ &= -\nu'^T\nabla^2\mathcal{H}_{\mathbf{F}}(\mathbf{X})\nu' + \nabla\mathbf{h}(\mathbf{X})^T\lambda, \end{aligned} \tag{4.59}$$

which is not necessarily negative (unless, of course, $\mathbf{X}$ is feasible, i.e., $\mathbf{h}(\mathbf{X}) = \mathbf{0}$). The primal-dual interpretation, however, shows that the norm of the residual decreases in the Newton direction, i.e.,

$$\frac{d}{dt}||r(\mathbf{Y}+t\nu_{\mathbf{Y}_{pd}})||_2^2\Big|_{t=0} = 2r(\mathbf{Y})^T Dr(\mathbf{Y})\nu_{\mathbf{Y}_{pd}} = -2r(\mathbf{Y})^T r(\mathbf{Y}). \tag{4.60}$$

Taking the derivative of the square, we obtain

$$\frac{d}{dt}||r(\mathbf{Y}+t\nu_{\mathbf{Y}_{pd}})||_2\Big|_{t=0} = -||r(\mathbf{Y})||_2. \tag{4.61}$$

This allows us to use $||r||_2$ to measure the progress of an infeasible solution by using the Newton method.

4.4 The SMS-EMOA-HNM

By the above discussion, we can expect linear convergence for inequality constrained problems and even quadratic convergence for unconstrained problems. The convergence, however, is only of local nature as it is the case for all Newton methods. Thus, a carefully chosen starting population is required or else the Newton iteration may fail to converge. We propose thus in the next step to hybridize the HNM with an evolutionary algorithm in order to obtain a fast and reliable memetic strategy. As an evolutionary algorithm we will use the SMS-EMOA which is a state-of-the-art EMOA that aims for hypervolume maximization ([57]). Crucial for the effective realization of the hybrid is to decide when to switch from the evolutionary strategy to HNM. We suggest to apply the Newton method once (i) the current population of SMS-EMOA only consists of mutually non-dominated elements and (ii) the difference of the hypervolume values of two consecutive populations is less than a given threshold.

More precisely, the hybrid method we propose here, SMS-EMOA-HNM, works as follows (compare to Algorithm 8): First, an initial population with μ elements is chosen at random. In each iteration step (Steps 4 to 11), a new offspring is generated and added to the current archive. Then, the new population is categorized into s sub-fronts, $G_1, \ldots, G_s$, according to the domination grade. Next, the hypervolume contribution is computed for each individual of the worst front G_s, and the individual with the least contribution is deleted from the archive (denoted as updated archive). This iteration is continued until the population consists of only one sub-front and the difference of the hypervolume value of the current and the updated archive is less or equal to a threshold tol_{HNM}. Finally, HNM is performed on the best-found population and the improved result is returned by the algorithm.

Algorithm 8 SMS-EMOA-HNM

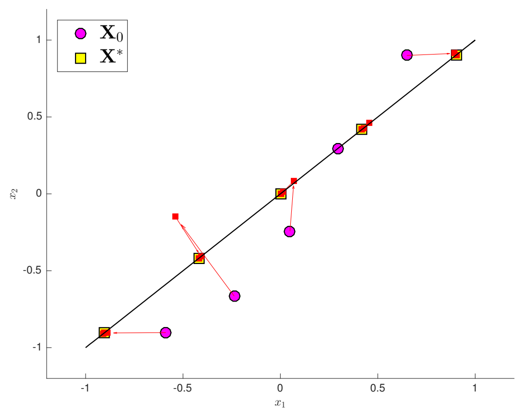
```
Require: A reference point $\mathbf{r} \in \mathbb{R}^m$, a given tolerance $tol_{HNM} \in \mathbb{R}_+$ for SMS-EMOA,
    and a maximal number of iterations $I_{max}$ and a tolerance $tol_x \in \mathbb{R}_+$ for HNM.
Ensure: A final population $X_{(F)}$..
 1: Initialize a population $X$ with $\mu$ elements at random.
 2: Set $\mathcal{H}_u := \mathcal{H}(\mathbf{f}(X))$
 3: do
 4:     Set $\mathcal{H}_c := \mathcal{H}_u$                                  ▷ HV value of current archive
 5:     Generate an offspring $\mathbf{x}' \in \mathbf{S}$ from $X$.
 6:     Set $X := X \cup \{\mathbf{x}'\}$.
 7:     Build a ranking $X = G_1 \cup G_2 \cup \ldots \cup G_s$ according the grade of dominance,
    where $G_s$ denotes the worst sub-front.
 8:     Compute the hypervolume contribution for each $\mathbf{x} \in G_s$.
 9:     Denote by $\mathbf{x}^*$ the element with least hypervolume contribution in $G_s$.
10:     Set $X := X \setminus \{\mathbf{x}^*\}$
11:     Set $\mathcal{H}_u := \mathcal{H}(\mathbf{f}(X))$                     ▷ HV value of updated archive
12: while $|\mathcal{H}_u - \mathcal{H}_c| > tol_{HNM}$ or $s > 1$
13: Construct a $\mu \cdot n$-vector $\mathbf{X}$ from $X$.
14: Set $\tilde{\mathbf{X}} := HNM(\mathbf{X}, I_{max}, tol_x)$.
15: return $X_{(F)}$ constructed from $\tilde{\mathbf{X}}$
```

4.5 Numerical results

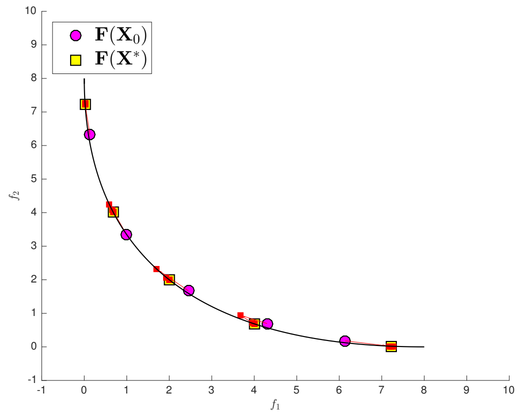
Here, we present some numerical results regarding the use of the proposed hypervolume Newton method (HNM). First, a comparison against the gradient-ascent hypervolume method (GAH) is included by using the approximation of the Hessian to compute the hypervolume Newton method (HNMA). Next, we compare the computational time in order to calculate both the HNMA and the HNM. Finally, numerical results are shown regarding the integration of the HNM into the SMS-EMOA.

4.5.1 HNMA vs GAH

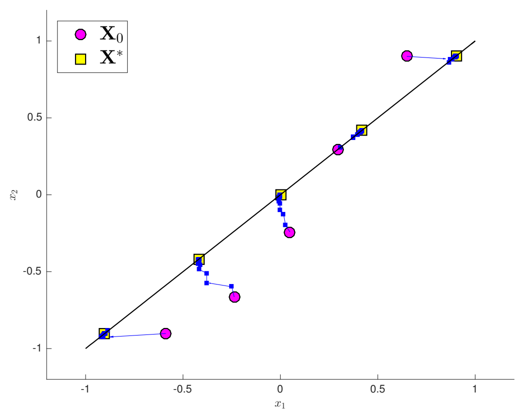
The first attempt to use the hypervolume Newton method works by computing the approximation of the hypervolume Hessian matrix as stated in Equation (4.11). Here, we first present the performance of the HNMA as a standalone algorithm, and further on the HNMA is compared against the GAH [35]. For this purpose, we take MOP1 and MOP2, then a population of 5 elements is assigned for approaching to the solution set. Figures 4.15 and 4.16 show the comparison of the performance of the HNMA against the GAH in both decision variable and objective space. The graphical comparison shows the speed of the HNMA to get approached to the best hypervolume approximation for 5 elements in both examples. One of the main advantages of the HNMA over the GAH is the choice of a correct step size, while in GAH has to be adapted iteration by iteration in the HNMA we set $t_i = 1$ for every iteration of the method.



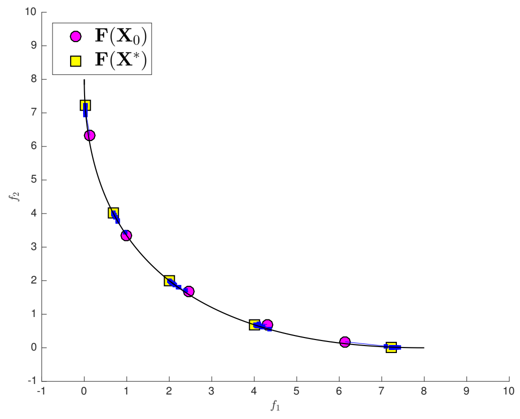
(a)



(b)



(c)



(d)

Figure 4.15: Application of the hypervolume Newton method (HNMA) (a) and (b), against the application of the gradient-ascent hypervolume method on a 5-element population on MOP1 in both decision variable (left) and objective (right) space. The starting population for both methods is $\mathbf{X}_0 = \{\mathbf{x}_1 = (-0.5, -0.9)^T, \mathbf{x}_2 = (-0.2, -0.6)^T, \mathbf{x}_3 = (0.0, -0.2)^T, \mathbf{x}_4 = (0.3, 0.3)^T, \mathbf{x}_5 = (0.6, 0.9)^T\}$.

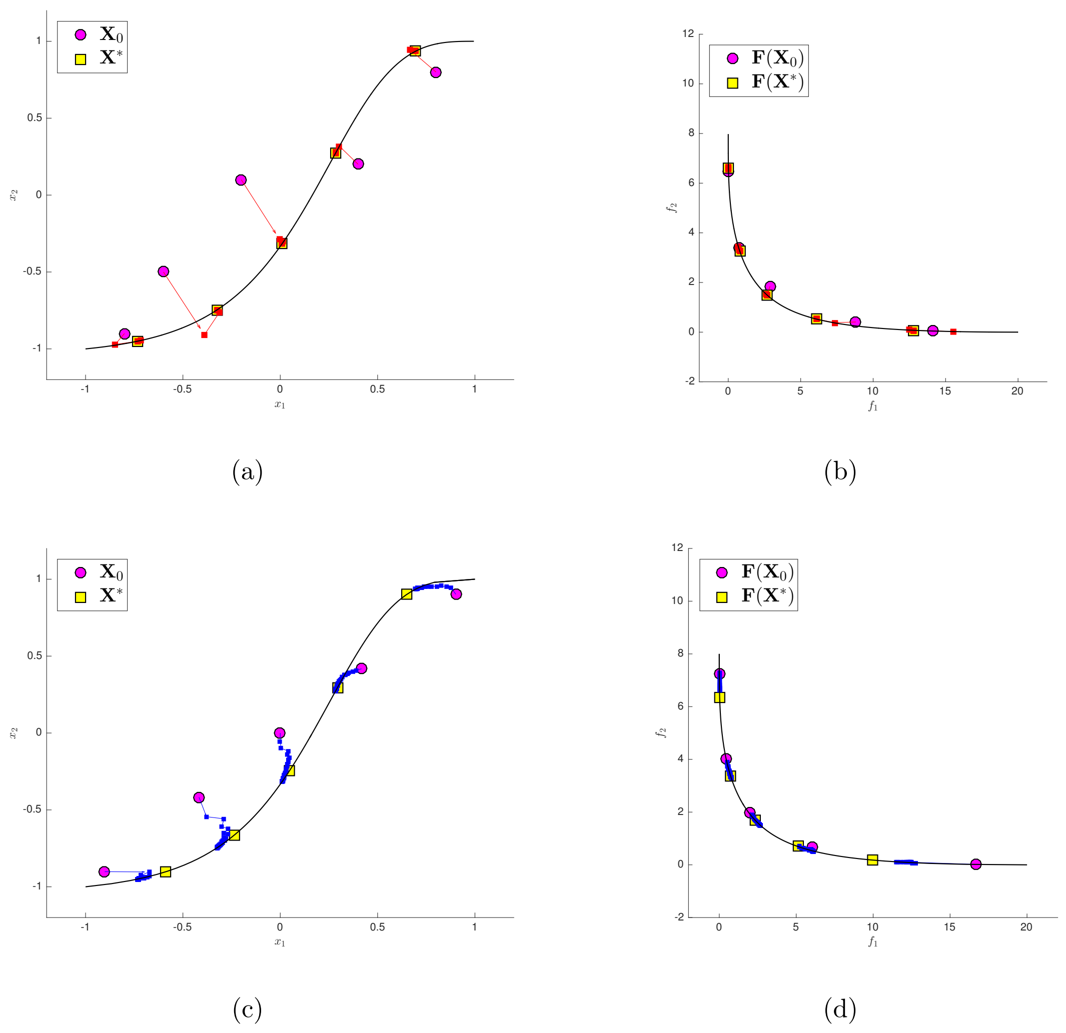


Figure 4.16: Application of the hypervolume Newton method (HNMA) (a) and (b), against the application of the gradient-ascent hypervolume method on a 5-element population on MOP2 in both decision variable (left) and objective (right) space. The starting population for both methods is $\mathbf{X}_0 = \{\mathbf{x}_1 = (0.9, 0.9)^T, \mathbf{x}_2 = (0.4, 0.4)^T, \mathbf{x}_3 = (0.0, 0.0)^T, \mathbf{x}_4 = (-0.4, -0.4)^T, \mathbf{x}_5 = (-0.9, -0.9)^T\}$.

The errors are demonstrated in Tables 4.7 and 4.8, respectively. In both tables, a comparison is made to the GAH yielding a different convergence rate. In this case, we see empirically that the use of HNMA even using an approximation to compute the Hessian matrix can yield quadratic convergence.

Table 4.7: Errors of the iterations obtained by the HNMA and the GAH on MOP1 (refer to Figure 4.15).

Iteration	Error HNMA	Error GAH
0	6.3019173538212	6.3019173538212
1	0.3822821410060	1.1256506467367
2	0.0036900401779	0.6345590706760
3	0.0000002871523	0.4247608492216
4	0.0000000000001	0.3043120967485
5	0.0	0.2237221272231
⋮	⋮	⋮
48	0.0	0.0000046743939
49	0.0	0.0000036350798
50	0.0	0.0000028266895

Table 4.8: Errors of the iterations obtained by the HNMA and the GAH on MOP2 (refer to Figure 4.16).

Iteration	Error HNMA	Error GAH
0	1.548406126519012	1.5484061265190
1	0.462396554222039	0.5448676870680
2	0.000049417729375	0.2211324621996
3	0.000000013384221	0.0937296798889
4	0.000000000000001	0.0400550587705
5	0.0	0.0177895807373
⋮	⋮	⋮
48	0.0	0.0000000000013
49	0.0	0.0000000000007
50	0.0	0.0000000000002

4.5.2 HNMA vs HNM

In this part, we investigate the HNMA and the hypervolume Newton method using the mathematical formulation of the Hessian matrix instead of an approximation. One natural question is how the performance of the HNM is against the version where the Hessian is approximated via finite differences (HNMA). Performance-wise we have not observed a noticeable difference in our computations. This changes, however, when considering the computational times for both methods due to the bandwidth structure of the Hessian. As an example, Table 4.9 shows the CPU times of the HNM and HNMA on MOP1 for two different population sizes. The results show average values over 20 different randomly chosen initial populations with seven iterations each. As anticipated, the CPU times are significantly less when computing the exact Hessian, for increasing values of μ and n. In our example, the CPU times differ by 2 orders of magnitude for $\mu = 100$ where we only have $n = 2$ decision variables.

Table 4.9: CPU times (in seconds) of HNMA and HNM for different population sizes using 7 iterations over 20 experiments.

μ	Description	**Time (HNMA)**	**Time (HNM)**
20	Mean	52.1808	**4.2234**
	Std	0.5563	0.0274
100	Mean	3710.7411	**13.7274**
	Std	22.1876	0.1342

4.5.3 SMS-EMOA-HNM vs SMS-EMOA vs IBEA

Next, we evaluate the potential of the novel memetic strategy. As test problems we selected nine bi-objective optimization problems (since the computation of the exact Hessian matrix is only stated for $k = 2$) three unconstrained (MOP1, MOP-GSP ($\lambda = 1.5$), Fonseca1) and six box-constrained MOPs (ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, DTLZ1, DTLZ2, DTLZ3, DTLZ4, DTLZ7, MOP-GSP ($\lambda = 0.5$)) that have different properties such as the shapes of the Pareto front and the modalities of the objective functions (see Appendix A for the definition of all considered MOPs). The dimensions of the decision spaces vary from 2 to 30. For comparison we choose to use SMS-EMOA, the base algorithm of our memetic strategy, and IBEA ([114]).

As design parameters we have chosen $tol_{HNM} = 1 \times 10^{-4}$, $I_{max} = 6$, $tol_x = 1 \times 10^{-10}$, and $\mu = 50$. As reference point we have chosen $\mathbf{r} = (11, 11)^T$ for all problems and have set a budget of 50,000 function evaluations. All experiments have been repeated for 20 independent runs. Figures 4.17, 4.18, and 4.19 show the convergence plots for all three algorithms. The black lines represent the means for every 100 function evaluations of SMS-EMOA-HNM, the red dotted lines the respective means for the competitor, and the magenta dots the maximal hypervolume values for each problem. We can observe that the new hybrid reaches the maximal hypervolume values in all cases while the two other algorithms get stuck in some cases. Further, even if all algorithms converge toward the maximum, the convergence is significantly faster for SMS-EMOA-HNM in these cases. Table 4.10 shows the hypervolume values for all three methods for a budget of 30,000 function evaluations except for the ZDT problems where the budget is set to 48,000 function evaluations. In all cases, SMS-EMOA-HNM achieves the best values.

The above mentioned superiority does not only hold when considering the hypervolume value but also the approximation quality of the Pareto set which we consider next. Figure 4.20 shows a numerical result of SMS-EMOA-HNM and its base MOEA on the Lame Super Sphere problem with $n = 2$ for a budget of 20,000 function evaluations. The Pareto set is the line segment connecting the points $(0, 0)^T$ and $(1, 1)^T$. Apparently, the approximation quality of SMS-EMOA-HNM is much better than the one from SMS-EMOA. Remarkably, the approximation quality of the latter can not reach the other one even if the function evaluation budget is increased. This observation gets confirmed in Table 4.11 where the approximation qualities (measured via the averaged Hausdorff distance Δ_2 [6]) of the final populations for all problems and algorithms are displayed.
Concluding, one can say that SMS-EMOA-HNM significantly outperforms the two other MOEAs on our chosen benchmark functions.

Concluding, one can say that SMS-EMOA-HNM significantly outperforms the two other MOEAs on our chosen benchmark functions. Finally, even considering the

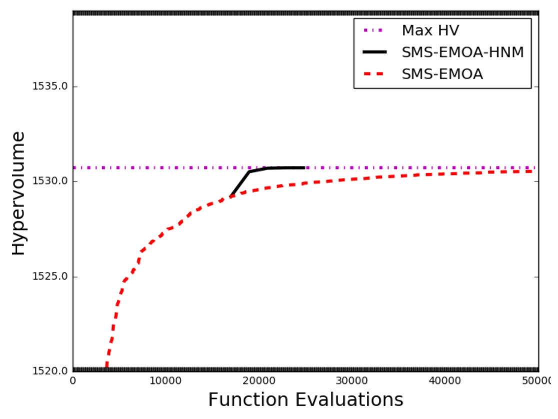
Table 4.10: Statistical results according hypervolume for IBEA, SMS-EMOA, and SMS-EMOA-HNM for a budget of 30,000 function evaluations.

Problem	**Desc.**	**IBEA**	**SMS-EMOA**	**SMS+HNM**
MOP 2	Mean	1491.79259	1529.57219	**1530.51847**
	Std	9.07020	0.45207	0.35653
Fonseca 1	Mean	15.05659	15.06154	**15.06166**
	Std	0.00046	0.00016	0.00006
LSS	Mean	15.69047	15.94703	**15.94728**
$\alpha = 1.5$	Std	0.08453	0.00006	0.00005
ZDT1M	Mean	109.59640	109.65720	**109.65725**
	Std	0.03508	0.00013	0.00006
ZDT2M	Mean	106.51610	106.51728	**106.51762**
	Std	0.00016	0.00024	0.00011
ZDT3M	Mean	116.59622	116.81255	**117.77585**
	Std	1.11096	0.80569	0.00425
ZDT4M	Mean	109.00213	109.13230	**109.55300**
	Std	0.98721	0.10023	0.011003
ZDT6M	Mean	103.23684	**103.88335**	103.84285
	Std	0.89765	0.38369	0.56432
DTLZ1	Mean	120.74389	120.87222	**120.87234**
	Std	0.14745	0.00019	0.00017
DTLZ2	Mean	120.20350	120.20744	**120.20866**
	Std	0.00037	0.00008	0.00005
DTLZ3	Mean	120.19543	**120.20411**	120.20043
	Std	0.00532	0.00216	0.00422
DTLZ4	Mean	116.02312	**116.63485**	114.23112
	Std	5.73222	4.99513	6.21521
DTLZ7	Mean	92.02387	92.31212	**92.70048**
	Std	5.11123	4.85353	4.764850
LSS	Mean	118.48256	118.47451	**118.49017**
$\alpha = 0.5$	Std	0.00014	0.00681	0.00409

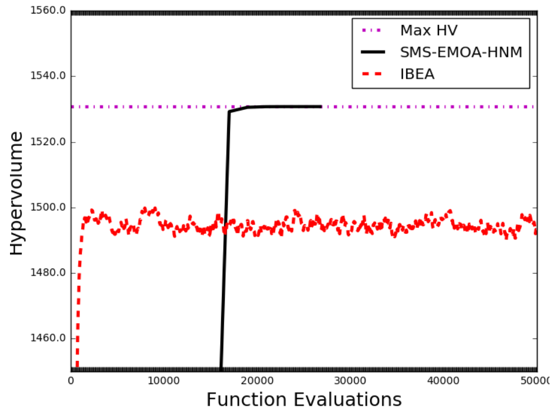
computational time expended to compute the local search, our algorithm use less time in most of the cases than the SMS-EMOA (See Table 4.12). For this comparison, we use the implementation provided by jMetal [115].

Table 4.11: Δ_2 values between the final approximations and the Pareto sets for the three different algorithms. The values are taken from 20 independent runs.

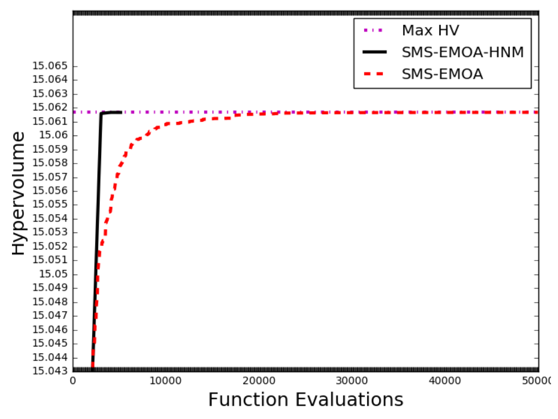
Problem	**Desc.**	**IBEA**	**SMS-EMOA**	**SMS+HNM**
MOP 2	Mean	0.03498	0.03907	**0.00274**
	Std	0.00047	0.00252	0.00081
Fonseca 1	Mean	0.28842	**0.01714**	0.05033
	Std	0.01285	0.00281	0.00693
LSS	Mean	0.86807	0.11214	**0.06331**
$\alpha = 1.5$	Std	0.01165	0.00346	0.00911
ZDT1M	Mean	0.72564	0.55342	**0.09236**
	Std	0.00984	0.00342	0.00023
ZDT2M	Mean	0.53948	0.23453	**0.12893**
	Std	0.00748	0.00231	0.00199
ZDT3M	Mean	0.61321	0.42948	**0.09787**
	Std	0.00564	0.00234	0.00432
ZDT4M	Mean	0.48587	0.36395	**0.31012**
	Std	0.08549	0.02184	0.02342
ZDT6M	Mean	0.74125	**5.41245**	5.98743
	Std	0.16352	0.09854	0.062145
DTLZ1	Mean	0.44647	0.04733	**0.01037**
	Std	0.00186	0.00832	0.00781
DTLZ2	Mean	0.18434	0.02867	**0.00478**
	Std	0.00234	0.00032	0.00147
DTLZ3	Mean	0.55874	**0.07548**	0.09874
	Std	0.21487	0.04784	0.24154
DTLZ4	Mean	0.98745	**0.54127**	0.84741
	Std	0.35241	0.24517	0.45147
DTLZ7	Mean	0.74859	0.42151	**0.40714**
	Std	0.34182	0.01987	0.00987
LSS	Mean	0.20047	0.20815	**0.20270**
$\alpha = 0.5$	Std	0.00221	0.00012	0.01106



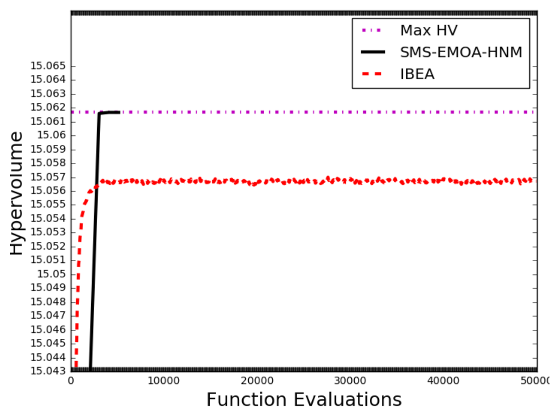
(a) SMS-EMOA MOP2



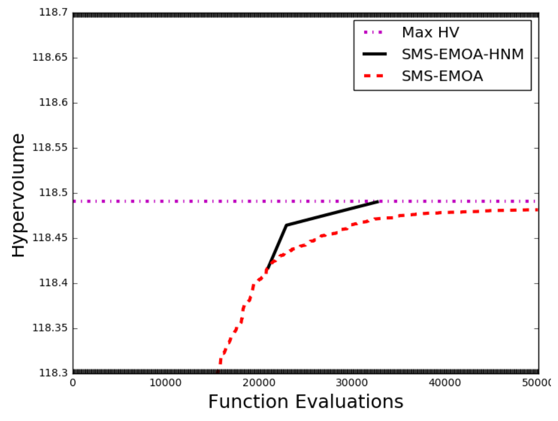
(b) IBEA MOP2



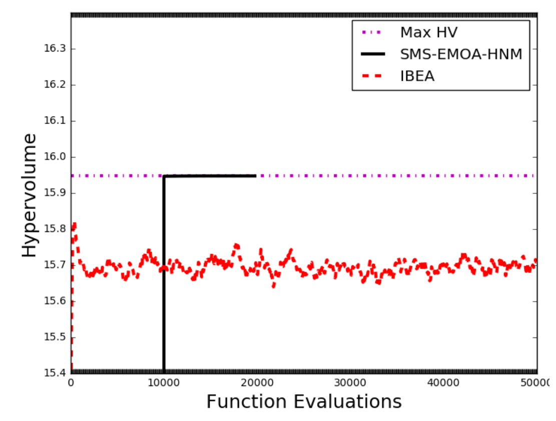
(c) SMS-EMOA Fonseca



(d) IBEA Fonseca

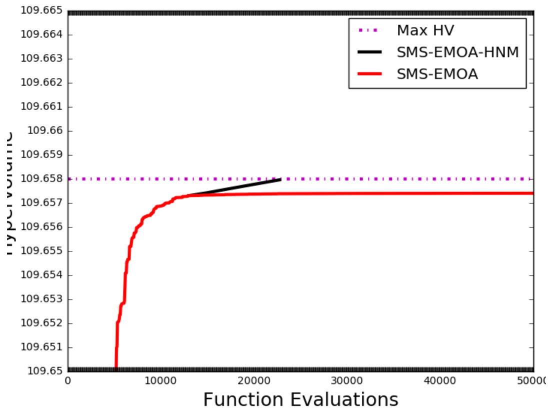


(e) SMS-EMOA LSS $\alpha = 1.5$

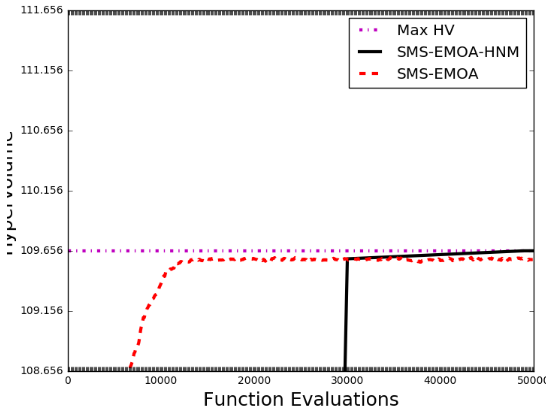


(f) IBEA LSS $\alpha = 1.5$

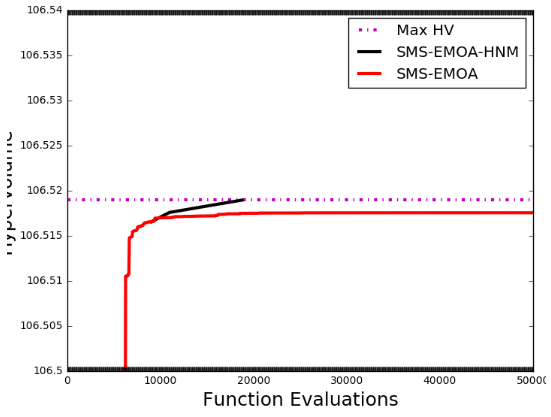
Figure 4.17: Convergence plot comparisons over the whole execution of SMS-EMOA, IBEA, and SMS-EMOA-HNM for the unconstrained problem cases.



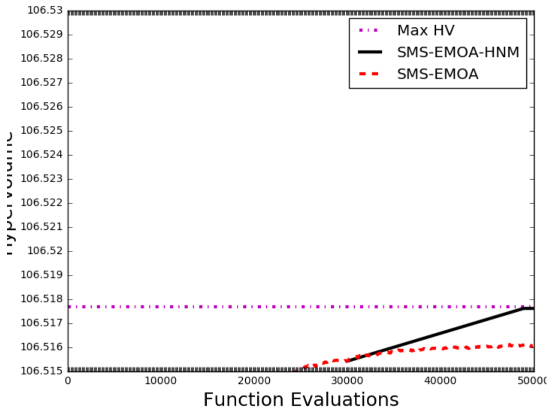
(a) SMS-EMOA ZDT1M



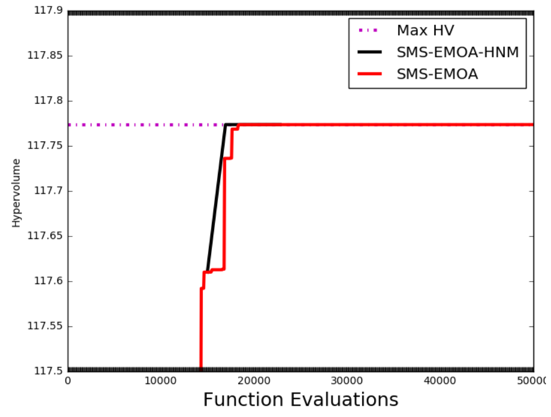
(b) IBEA ZDT1M



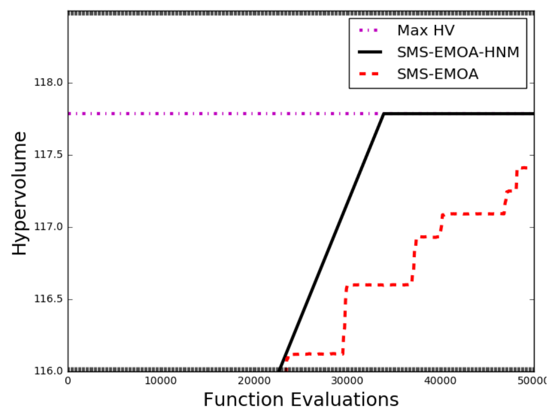
(c) SMS-EMOA ZDT2M



(d) IBEA ZDT2M

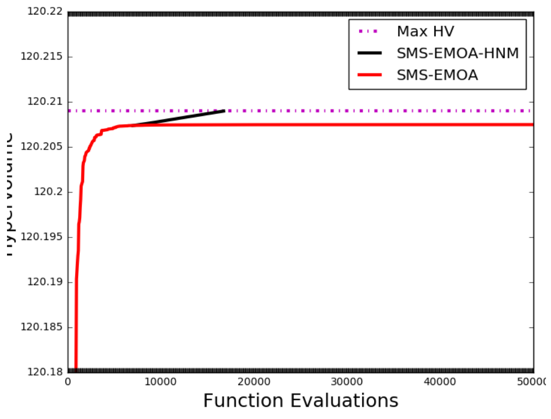


(e) SMS-EMOA ZDT3M

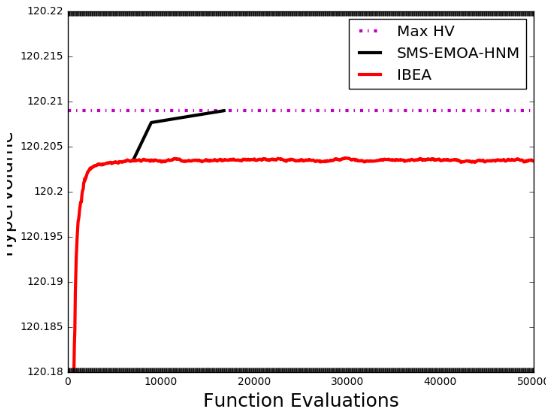


(f) IBEA ZDT3M

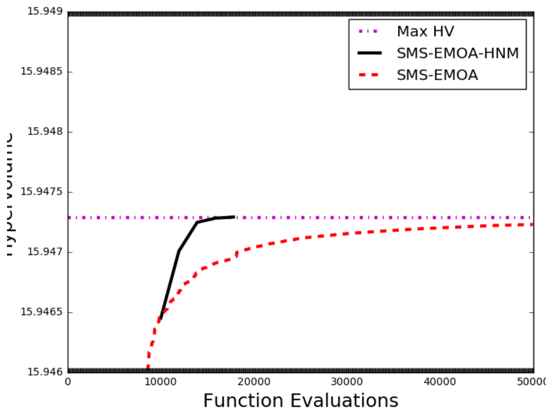
Figure 4.18: Convergence plot comparisons over the whole execution of SMS-EMOA, IBEA, and SMS-EMOA-HNM for the box-constrained problem cases (Part 1).



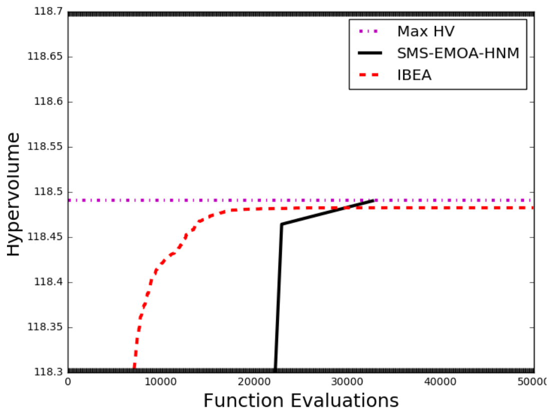
(a) SMS-EMOA DTLZ2



(b) IBEA DTLZ2

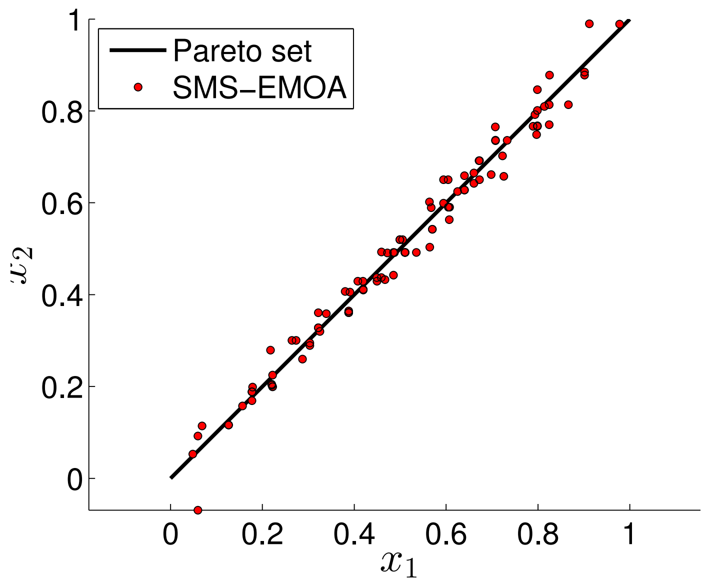


(c) SMS-EMOA LSS $\alpha = 0.5$

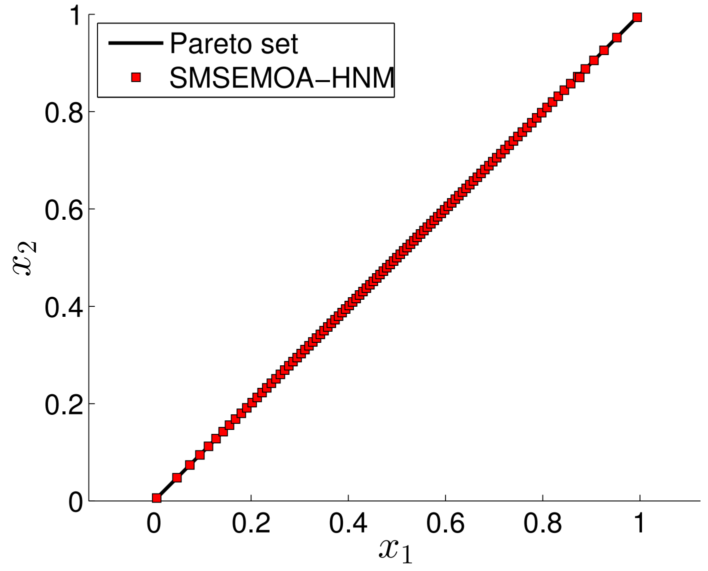


(d) IBEA LSS $\alpha = 0.5$r

Figure 4.19: Convergence plot comparisons over the whole execution of SMS-EMOA, IBEA, and SMS-EMOA-HNM for the box-constrained problem cases (Part 2).



(a) SMS-EMOA



(b) SMS-EMOA-HNM

Figure 4.20: Final approximation in Parameter space obtained by SMS-EMOA (above) and SMS-EMOA-HNM (below) in the Lame Super Sphere problem with $n = 2$ and $\mu = 100$ after 20,000 function evaluations.

Table 4.12: Statistical results according time (seconds) for SMS-EMOA, and SMS-EMOA-HNM during a function evaluations (FE) budget used for applying the HNM.

Problem	**#FE**	**Desc.**	**SMS-EMOA**	**SMS+HNM**
MOP 2	7800	Mean	3.84589	**2.35028**
		Std	0.95642	0.04619
Fonseca 1	7800	Mean	1.25218	**0.64951**
		Std	1.57494	0.00523
LSS	11700	Mean	4.48897	**3.69927**
$\alpha = 1.5$		Std	1.93809	0.07537
ZDT1M	9750	Mean	5.08166	**3.12735**
		Std	0.79936	0.04271
ZDT2M	9750	Mean	4.97982	**3.75375**
		Std	2.34125	0.08197
ZDT3M	7800	Mean	**5.13236**	5.42000
		Std	2.29418	0.05670
ZDT4M	7800	Mean	**3.914806**	5.30775
		Std	2.23733	0.23887
ZDT6M	7800	Mean	5.70897	**5.40971**
		Std	1.79426	0.08283
DTLZ1	7800	Mean	3.83354	**2.93848**
		Std	0.96666	0.07112
DTLZ2	9750	Mean	8.08016	**3.11715**
		Std	3.70127	0.05568
DTLZ3	7800	Mean	5.11303	**3.11460**
		Std	2.61293	0.07341
DTLZ4	7800	Mean	6.36661	**2.90622**
		Std	2.36567	0.05611
DTLZ7	7800	Mean	8.54426	**2.92156**
		Std	2.26612	0.06659
LSS	7800	Mean	5.25721	**4.63070**
$\alpha = 0.5$		Std	1.30050	0.12503

5 | On the Treatment of Parameter dependent Multi-Objective Optimization Problems

Nowadays, in many real world applications, we have in addition to the multiple objectives to consider the influence of external parameters that can affect the solution set. For instance, consider the fabrication of an embedded system, which is designed to report the activity of a volcano. The fabrication aims to reduce the energy consumption and also to increase the accuracy of the measures to report, however, both objectives are in conflict. In other words for a more accurate measure more is the energy consumption. Since the embedded system has to work inside a volcano, the temperature of the environment will change over a certain range causing that the system consumes more energy to maintain stable its own temperature, in order to its components could work properly. The previous example fits perfectly into the context of a PMOP where an external parameter (temperature) cannot be neglected nor optimized but it has to be considered over the whole optimization process.

As we mentioned in Chapter 2, there is a set of evolutionary techniques that have been developed for the treatment of a given PMOP ([60, 60, 61, 61–70]). Examples of them include several strategies in order to (i) detect a change in the external parameter and (ii) then to use a mechanism to response to that change. Most of the evolutionary techniques consider to solve a PMOP as a MOP with a certain value for the external parameter. However, to solve a PMOP as a whole is not yet well explored.

In this chapter, we first present the simple neighbourhood search (SNS) in the context of PMOPs. The algorithm is based on the observations made in a study [58] over the behavior of stochastic local search (SLS) for PMOPs contained in Chapter 2. Comparisons show the advantage of considering neighbourhood information to produce points in order to converge faster toward the solution set. Next to this, we present an evolutionary framework to tackle PMOPs in a non-sequential way. This framework aims to find an approximation of the solution set by considering a discretization of the λ-space. Then, based on this framework, an evolutionary approach based on decomposition (PMOEA/D) is presented for the treatment of PMOPs. The algorithm is able to compute the family of Pareto sets and fronts of a given PMOP in one single run. PMOEA/D incorporate results and knowledge from the SLS and SNS. We include numerical results and comparisons against modified versions of MOEA/D adapted to the context of PMOPs.

5.1 Simple Neighborhood Search

In this section, we present the simple neighborhood search in the context of PMOPs. Our principal objective is to investigate the influence of SNS within set based methods. In order to prevent interferences with other effects we have thus to omit all other operators (as, e.g., crossover). The Simple Neighborhood Search for PMOPs takes this into consideration: initially, a generation $A_0 \subset \mathbb{R}^{n+l}$ is chosen at random, where Λ is discretized into $\tilde{\Lambda} = \{\lambda_1, \ldots, \lambda_s\}$. In the iteration process, for every element $(a_{\mathbf{x}}, a_\lambda) \in A_i$, a new element $(b_{\mathbf{x}}, b_\lambda)$ is chosen via SLS, where b_λ has to take one of the values of $\tilde{\Lambda}$. The given archive A_i and the set of newly created solutions B_i are the basis for the sequences of candidate solutions A_i^l, $l = 1, \ldots, s$, and the new archive A_{i+1}: for A_i^l the non-dominated solutions from $A_i \cup B_i$ with λ-value λ_j are taken, and A_{i+1} is the union of these sets (plus the respective λ values). Algorithm 9 shows the pseudo code of SNS. Hereby, $nondom(A)$ denotes the non-dominated elements of a set A, $\pi(A, \lambda_i) := \{a \ : \ (a, \lambda_i) \in A\}$ denotes the $\mathbf{x}$-values of the elements of A with λ-value λ_i, and $(A, \lambda) := \{(a, \lambda) \ : \ a \in A\}$.

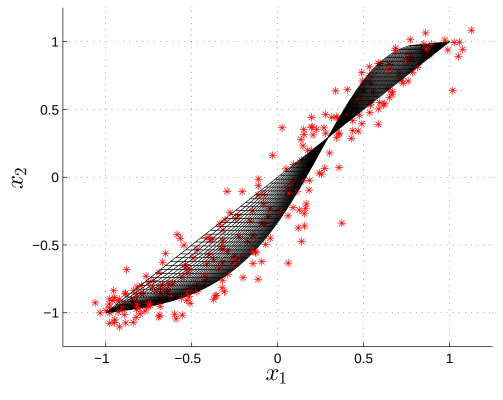
For sake of a small comparison we also investigate here the global counterpart of SNS, the Simple Global Search (GS), where all points are chosen uniformly at ran-

Algorithm 9 SNS for PMOPs

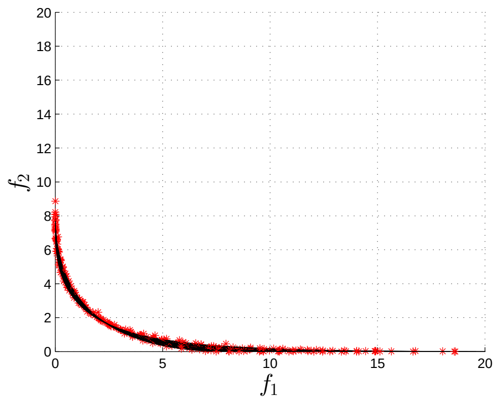
```
Require: Neighborhood $N_i(\mathbf{x}, \lambda)$ of a given point $(\mathbf{x}, \lambda)$ in iteration $i$.
Ensure: Sequence $A_i^l$ of candidate solutions for $F_{\lambda_l}$, $l = 1, \dots, s$
 1: Generate $A_0 \subset \mathbb{R}^{n+l}$ at random
 2: for $i = 0, 1, 2, \dots$ do
 3:     $B_i^l := \emptyset$
 4:     for all $(a_x, a_\lambda) \in A_i$ do
 5:         choose $(b_x, b_\lambda) \in N_i(a_x, a_\lambda)$
 6:         $B_i := B_i \cup (b_x, b_\lambda)$
 7:     end for
 8:     $A_{i+1}^l := nondom(\pi(A_i \cup B_i, \lambda_l))$, $l = 1, \dots, s$
 9:     $A_{i+1} := \bigcup_{l=1}^s (A_{i+1}^l, \lambda_l)$
10: end for
```

dom from the entire domain. That is, GS can be viewed as an application of SNS where the neighborhood N_i in Line 5 of Algorithm 1 is chosen as the entire domain. In order to reduce the overall number of candidate solutions we have not stored all non-dominated solutions but have used $ArchiveUpdateTight2$ ([116]) to update the archives A_i^m. The archiver $ArchiveUpdateTight2$ aims, roughly speaking, for gap free ϵ-Pareto sets. In our computations, we have used $\epsilon = (0.05, 0.05)^T$. Further, in each computation we have used 10 equally spaced divisions in λ-space and have generated one random element for the initial archive A_0 (i.e., $|A_0| = 10$).

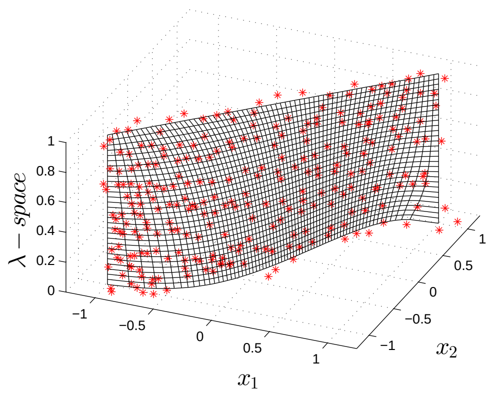
Figure 5.1 shows some numerical results by solving PMOP (2.17) in two different angles. In this example we have used a budget of 3,000 function evaluations (FEs) for SNS. Figures 5.1, 5.2, and 5.3 show the respective result for GS for a budget of 3,000 and 10,000 FEs. As domain we have chosen $S = [-10, 10]^2$. The superiority of SNS can be detected visually since those final archives are evenly spread around the solution sets. Compared to this, the result of GS lacks both in spread and convergence, though more than 3 times the number of FEs has been spent to get this result. This observation is confirmed by the values in Tables 5.1 and 5.2 where we show the distances (measured in terms of Δ_2 [6]) between the outcome sets and the union of the 10 Pareto fronts (i.e., our discretized set of interest).



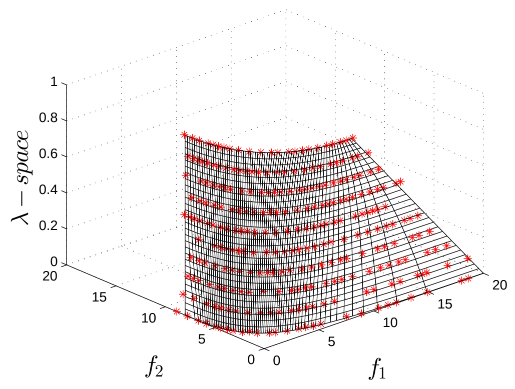
(a) SNS, decision space



(b) SNS, objective space

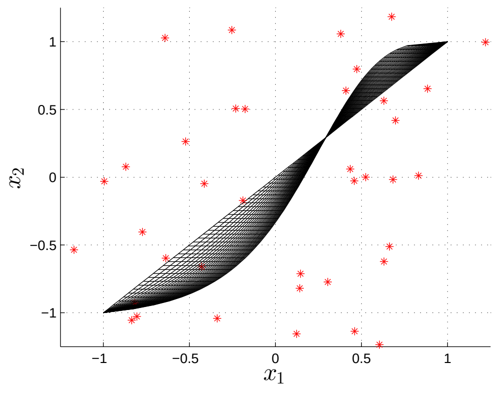


(c) SNS, decision space

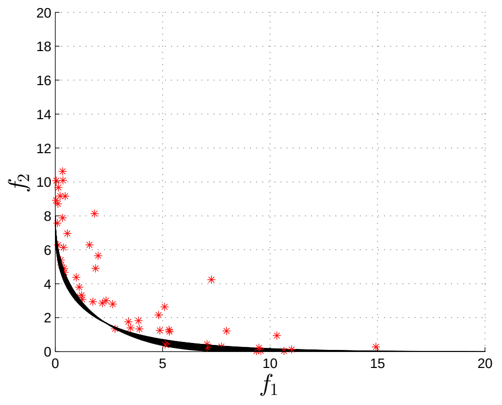


(d) SNS, objective space

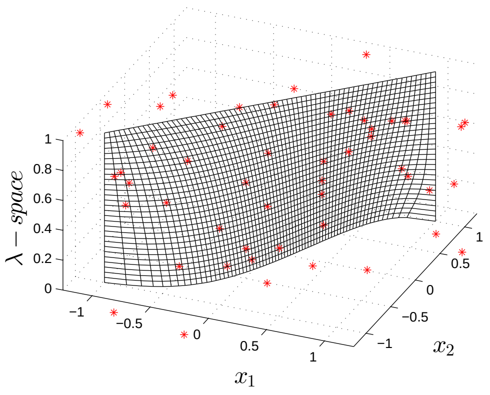
Figure 5.1: Numerical result of SNS with a budget of 3,000 FEs for PMOP (2.17).



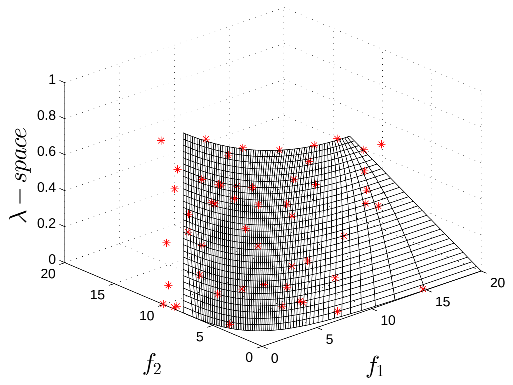
(a) GS 3,000, decision space



(b) GS 3,000, objective space

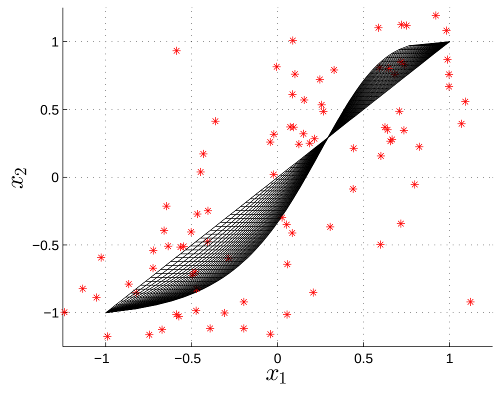


(c) GS 3,000, decision space

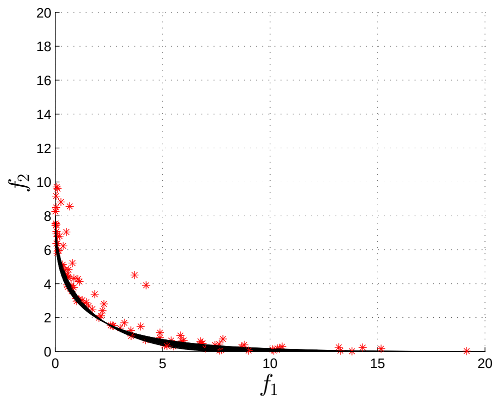


(d) GS 3,000, objective space

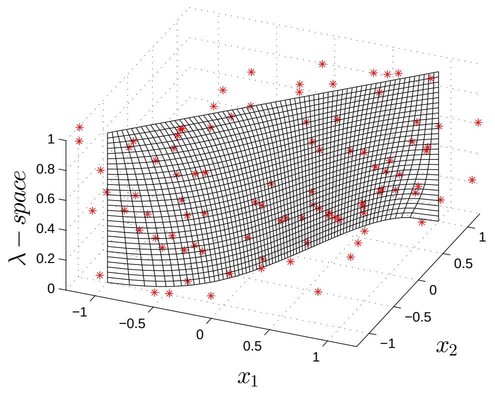
Figure 5.2: Numerical result of GS with a budget of 3,000 FEs for PMOP (2.17).



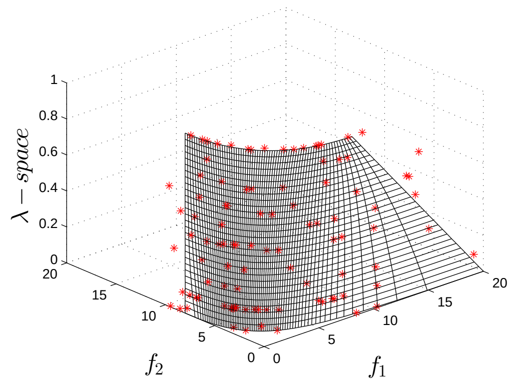
(a) GS 10,000, decision space



(b) GS 10,000, objective space



(c) GS 10,000, decision space



(d) GS 10,000, objective space

Figure 5.3: Numerical result of GS with a budget of 10,000 FEs for PMOP (2.17).

Table 5.1: Comparative results for PMOP(2.17) using Δ_2 value between the final archives and the union of the 10 Pareto fronts for both SNS and GS for a budget of 3,000 FEs. Shown are worst, average, and best value over 20 independent runs.

Algorithm	PMOP1					
Problem	SNS			GS		
λ-value/Description	Worst	Average	Best	Worst	Average	Best
$\lambda = 0.00$	1.934	**1.494**	1.101	25.342	17.220,	14.425
$\lambda = 0.11$	2.434	**1.977**	1.103	14.349	7.349	7.023
$\lambda = 0.22$	2.453	**2.126**	1.132	24.342	22.440	20.213
$\lambda = 0.33$	1.743	**1.526**	1.341	6.342	5.826	4.342
$\lambda = 0.44$	1.234	**1.032**	0.532	11.433	9.356	9.003
$\lambda = 0.55$	1.673	**1.402**	0.643	7.322	5.963	4.095
$\lambda = 0.66$	1.563	**1.220**	0.992	6.424	3.226	2.657
$\lambda = 0.77$	0.984	**0.550**	0.314	3.221	2.217	2.043
$\lambda = 0.88$	0.546	**0.381**	0.249	9.534	6.241	5.141
$\lambda = 1.00$	0.986	**0.385**	0.148	5.213	3.320	2.134

Next we consider a second PMOP which is again a convex homotopy of two MOPs. In this case, one of the Pareto fronts is convex while the other one is concave.

$$\begin{aligned} \mathbf{f}_\lambda &: \mathbb{R}^2 \to \mathbb{R}^2 \\ \mathbf{f}_\lambda(\mathbf{x}) &:= (1-\lambda)\mathbf{f}_1(\mathbf{x}) + \lambda \mathbf{f}_2(\mathbf{x}), \end{aligned} \tag{5.1}$$

where $\lambda \in [0,1]$, $a_1 = 0$, $a_2 = 1$ and $\mathbf{f}_1, \mathbf{f}_2 : \mathbb{R}^2 \to \mathbb{R}^2$,

$$\mathbf{f}_1(\mathbf{x}) = \begin{pmatrix} (x_1^2 + x_2^2)^{0.125} \\ ((x_1 - 0.5)^2 + (x_2 - 0.5)^2)^{0.25} \end{pmatrix},$$

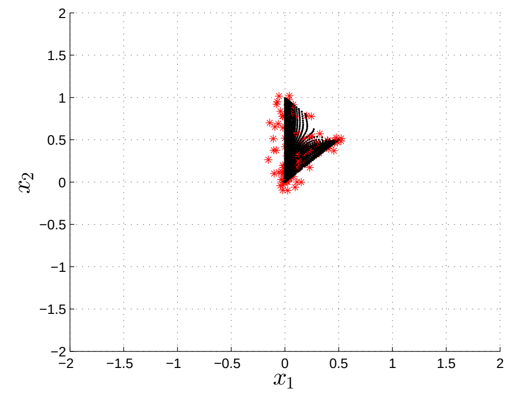
$$\mathbf{f}_2(\mathbf{x}) = \begin{pmatrix} x_1^2 + x_2^2 \\ (x_1 - a_1)^2 + (x_2 - a_2)^2 \end{pmatrix}.$$

Figures 5.4, 5.5, and 5.6 show some exemplary numerical results of SNS (5,000 FEs) and GS (5,000 and 10,000 FEs) using $S = [-10, 10]^2$. Again, SNS, though less FEs were used, is superior with respect to spread and convergence according the in-

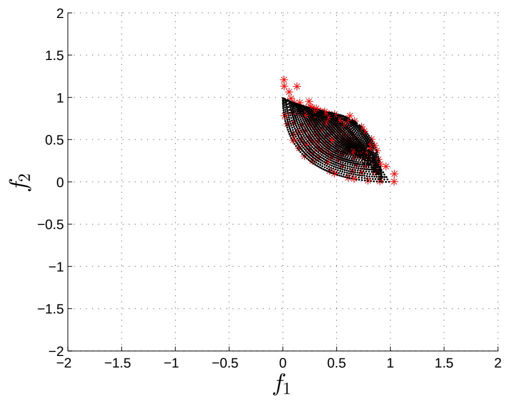
Table 5.2: Comparative results for PMOPs (5.1) using Δ_2 value between the final archives and the union of the 10 Pareto fronts for both SNS and GS for a budget of 5,000 FEs. Shown are worst, average, and best value over 20 independent runs.

Algorithm	PMOP2					
Problem	SNS			GS		
λ-value/Description	Worst	Average	Best	Worst	Average	Best
$\lambda = 0.00$	0.632	**0.365**	0.242	0.992	0.692	0.432
$\lambda = 0.11$	0.567	**0.221**	0.201	0.832	0.620	0.597
$\lambda = 0.22$	0.453	**0.136**	0.103	0.567	0.454	0.353
$\lambda = 0.33$	0.578	**0.103**	0.098	0.693	0.496	0.394
$\lambda = 0.44$	0.375	**0.049**	0.019	0.700	0.684	0.592
$\lambda = 0.55$	0.198	**0.047**	0.043	0.422	0.329	0.239
$\lambda = 0.66$	0.297	**0.069**	0.034	0.596	0.552	0.539
$\lambda = 0.77$	0.246	**0.056**	0.022	0.834	0.821	0.739
$\lambda = 0.88$	0.186	**0.077**	0.062	0.532	0.446	0.422
$\lambda = 1.00$	0.123	**0.045**	0.032	0.423	0.351	0.311

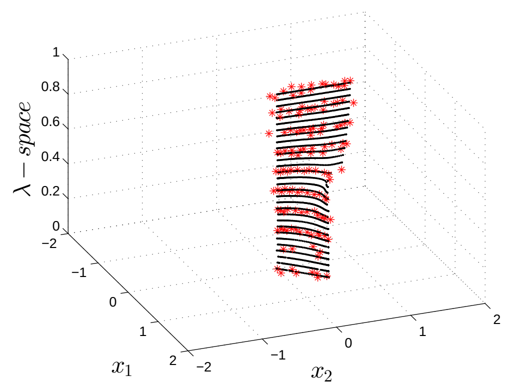
dicator Δ_2 in Table 5.1.



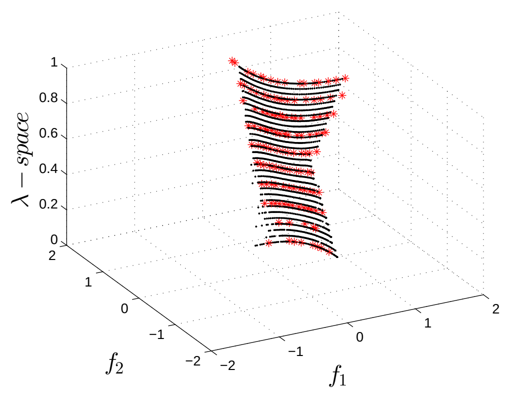
(a) SNS, decision space



(b) SNS, objective space

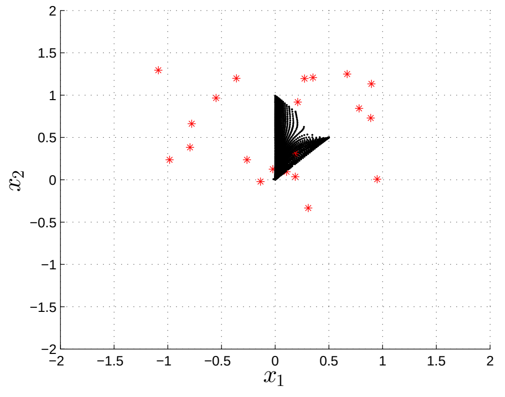


(c) SNS, decision space

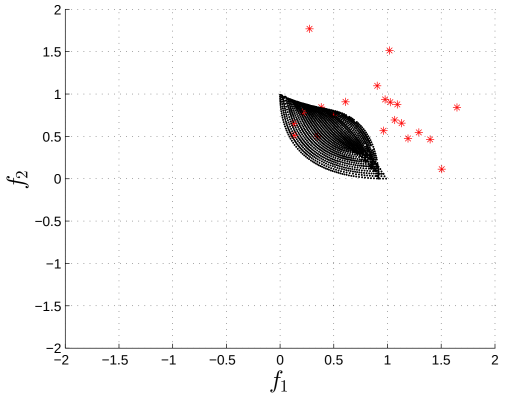


(d) SNS, objective space

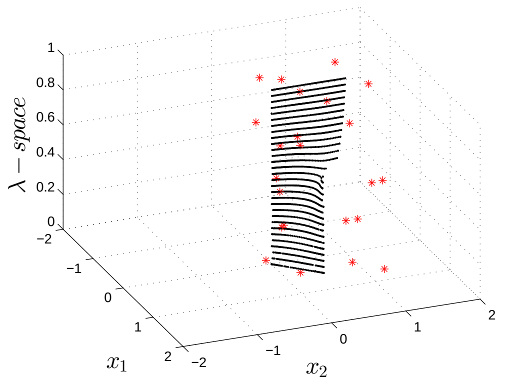
Figure 5.4: Numerical results of SNS with a budget of 5,000 FEs for PMOP (5.1).



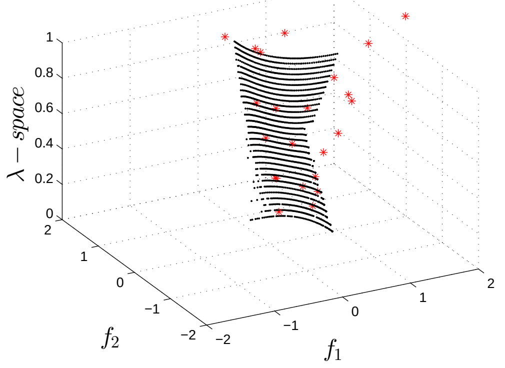
(a) GS 5,000, decision space



(b) GS 5,000, objective space

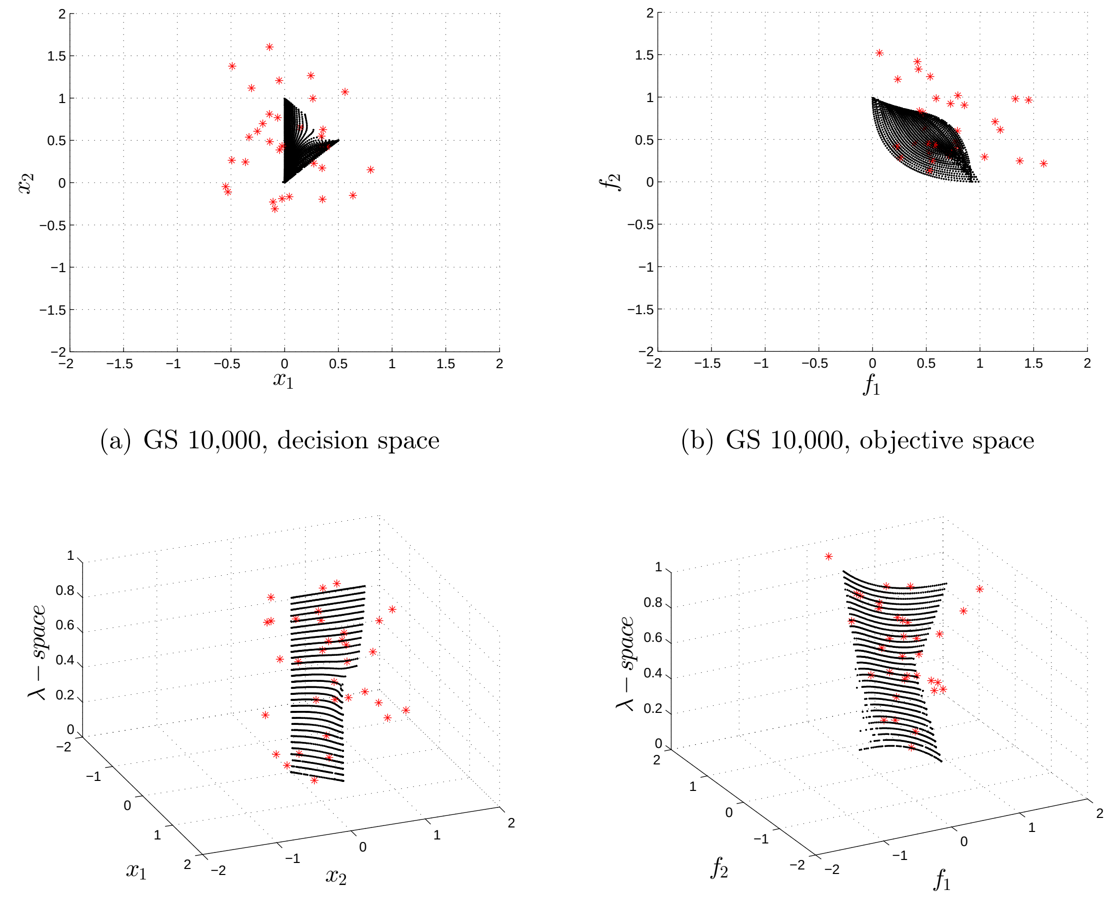


(c) GS 5,000, decision space



(d) GS 5,000, objective space

Figure 5.5: Numerical results of GS with a budget of 5,000 FEs for PMOP (5.1).



(a) GS 10,000, decision space

(b) GS 10,000, objective space

(c) GS 10,000, decision space

(d) GS 10,000, objective space

Figure 5.6: Numerical results of GS (10,000 FEs) for PMOP (5.1).

From [58], we know that both the movement toward as well as along the set of interest (in objective space) are inherent in SLS. Further, there is also a kind of 'opening' of the search in objective space which allows to find in principle all regions of the solution set Some first tests on a simple set based neighborhood search, called SNS, confirmed these statements on two PMOPs. Thus, the discussions indicate that the problem to find an approximation of the entire solution set of a PMOP is a well-conditioned problem for set based probabilistic algorithms such as evolutionary algorithms (EAs).

5.2 A General Framework for the Non-Sequential Treatment of PMOPs

In the following section, we present an evolutionary algorithm for the treatment of PMOPs. The algorithm is called PMOEA/D which is an approach based on decomposition. The most frequently used symbols of this section can be found in Table 5.3. We assume problems with only one single external parameter, i.e., $l = 1$.

Table 5.3: Symbols.

n	number of decision variables
k	number of objectives
μ_s	number of individuals per slice
s	number of slices in the approximation
μ	number of individuals in the whole population
W	set of weight vectors
l	number of external parameters
$\Lambda = \{\lambda_1, \ldots, \lambda_s\}$	external parameter space discretization
$P = \{(\mathbf{x}^{(1)}, \lambda_{j1}), \ldots, (\mathbf{x}^{(\mu)}, \lambda_{j\mu}))\}$	population with μ individuals $(\mathbf{x}^{(i)}, \lambda_{j\mu}) \in \mathbb{R}^{(n+l)}$

In order to improve the understanding of the proposed approach, we present, a general scheme to tackle a given PMOP in a non-sequential way. Algorithm 10 describes, in a straightforward manner, the process to find an approximation of the family of solution sets of a PMOP as a whole. The algorithm considers a discretization of the external parameter space $\Lambda = \{\lambda_1, \dots, \lambda_s\}$ where λ_{ji} is the corresponding scalar value to the λ_j-value within the discretization of Λ with $j = 1, ..., s$, s is the number of values in the discretization of Λ and μ is the total number of individuals in a population P_t. As a first step an initialization process is performed to create a population $P_0 = \{(\mathbf{x}^{(1)}, \lambda_{j1}), \dots, (\mathbf{x}^{(\mu)}, \lambda_{j\mu})\}$ of μ individuals. Then, we select a MOEA and after a configuration process we start by producing a new generation P_{t+1}, such a process is repeated until certain criterion is not fulfilled.

After using the selected MOEA to produce new populations, a projection operation is computed, in order to obtain the elements associated to each value λ_j of the discretization in Λ-space. Finally, a whole approximation of the family of solution sets is produced.

Algorithm 10 General scheme to tackle a given PMOP in a non-sequential way

```
Ensure: A final approximation $P^{(final)}$ of the family of Pareto sets and fronts of a given PMOP.
 1: Set $\Lambda := \{\lambda_1, \dots, \lambda_s\}$.
 2: Set $t := 0$.
 3: Initialize a population $P_t = \{(\mathbf{x}^{(1)}, \lambda_{j1}), \dots, (\mathbf{x}^{(\mu)}, \lambda_{j\mu})\} \subset \mathbb{R}^{n+l}$.
 4: Configure MOEA
 5: while stopping criterion is not fulfilled do
 6:     Compute $P_{t+1}$ from $P_t$ using a MOEA for one generation.
 7:     $t := t + 1$
 8: end while
 9: for $j = 1, \dots, s$ do
10:     $P_j^{(final)} := \prod(P_t, j)$
11: end for
12: return $P^{(final)} := \{P_1^{(final)}, ..., P_s^{(final)}\}$
```

5.3 An Evolutionary Approach for the Non-Sequential Treatment of PMOPs

We describe in Algorithm 10 a general framework for finding an approximation of the solution set of a given PMOP by avoiding sequential optimization of each λ_j in Λ. This framework is still undefined in many senses, for instance a description of the MOEA, also it is missing to explain a mechanism to balance the number of individuals of each slice, and finally, a more detail explanation of the generation of P_{t+1}. Among all the algorithms within the state of the art MOEA/D ([56]) is one of the most widely used approaches to compute a good approximation of the solution set of a given MOP. The algorithm decomposes a multi-objective optimization problem into a finite number of scalar optimization sub-problems and then it optimizes them concurrently. The algorithm requires an initial population P_0, a set of weight vectors W, and a given utility function. So, every time the algorithm produces a trial solution, the utility function and the associated weight vector decide if the new solution replaces a solution into the current P_t or not. This process is repeated until a stopping criterion is reached.

In the following, we propose an adaption of the MOEA/D algorithm to tackle PMOPs in a non-sequential way. Our algorithm decomposes a PMOP in a finite number of sub-problems. The number of sub-problems is given by the number of individuals in the population μ. The algorithm requires a set W_λ of μ well-spread weight vectors $\mathbf{w}^{(i)} \in \mathbb{R}^{(k+l)}$, over all the slices λ_j for $j = 1, ..., s$ and also a given utility function g_T. Then, let $\mathbf{w}^{(1)}, \dots, \mathbf{w}^{(\mu)}$ be the weight vectors that belong to W_λ, an utility function g_T and $Z = \{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(s)}\}$ a set of s reference points in $\mathbb{R}^{(k)}$ a PMOP can be decomposed into μ scalar optimization sub-problems. In order to make the decomposition process, we take the Tchebycheff method adapted to the context of PMOPs as an utility function g_T where the i-th sub-problem for a certain λ_j is defined as follows:

$$g_T((\mathbf{x}, \lambda_j)|\mathbf{w}^{(i)}, \mathbf{z}^{(j)}) = \max_{1 \le r \le k} \{\mathbf{w}_r^{(i)} |f_r(\mathbf{x}, \lambda_j) - z^{(j)}|\}, \tag{5.2}$$

where $\mathbf{w}^{(i)} = (w_1^{(i)}, \dots, w_k^{(i)})^T$. In order to tackle a given PMOP as a whole the new algorithm minimizes all the μ sub-problems in one single run. The following subsections will explain the different parts of the algorithm.

5.3.1 Reference Points Set

The reference point set Z contains the best found values for each objective function f_r for $r = 1, ..., k$ (considering minimization) according to every slice λ_j for $j = 1, \ldots, s$. Algorithm 11 shows the process to construct Z.

Algorithm 11 Generate_$Z(P, s, k)$

```
Require: A population $\mathbf{P}$, a number of slices $s$, a number of objectives $k$
Ensure: A set of best found values $Z$ for each slice $\lambda_j$ for $j = 1, ..., s$.
1: for $j = 1$ to $s$ do
2:     for $r = 1$ to $k$ do
3:         Set $\mathbf{z}_r^{(j)} = \arg\min_{(\mathbf{x},\lambda)\in P} f_r(\mathbf{x}, \lambda)$ s.t $\lambda = \lambda_j$.
4:     end for
5: end for
6: Return $Z$.
```

5.3.2 Weight Vectors Set

The set of well-spread weight vectors can be seen as a surface of weight vectors, however, for our purpose we treat individually every weight vector $\mathbf{w}_i$ for $i = 1, ..., \mu$. Figure 5.7 shows how the surface is. Every slice λ_j for $j = 1, ..., s$ contains a set of μ_s weight vectors generated by a certain method. In our case, we use a low discrepancy sequence based on lattices proposed in ([117]). The main reason to use this method is that the complexity is far lower than the one given by the uniform design method proposed in ([118]). Algorithm 12 shows away to construct W_λ.

5.3.3 Types and Construction of the Neighborhoods

For our algorithm, we determine two kinds of neighborhoods: one global and one local. The global neighborhood contains elements from the complete surface W_λ depicted in Figure 5.8(a), while the local neighborhood depicted in 5.8(b) only includes

Algorithm 12 Generate_$W_\lambda(\mu_s, s)$

```
Require: A number of individuals per slice $\mu_s$, number of slices $s$
Ensure: Surface $W_\lambda$.
1: Set $W_\lambda := \{\emptyset\}$
2: for $j = 1$ to $s$ do
3:     Generate $\mu_s$ well spread weight vectors $\mathbf{v}_1, \ldots, \mathbf{v}_{\mu_s}$ in $\mathbb{R}^k$.
4:     for $p = 1$ to $\mu_s$ do
5:         Set $\mathbf{w} := (\mathbf{v}_p^T, \lambda_j)^T$
6:         Set $W_\lambda := W_\lambda \cup \{\mathbf{w}\}$
7:     end for
8: end for
9: Return $W_\lambda$.
```

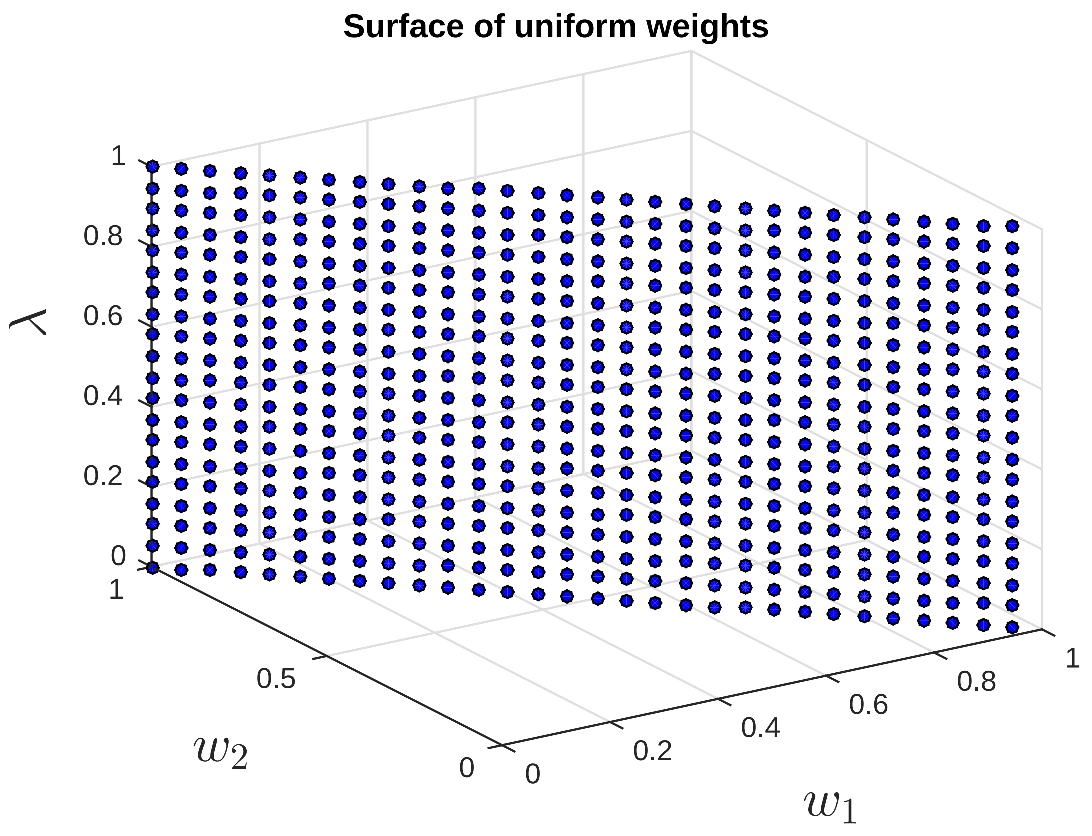
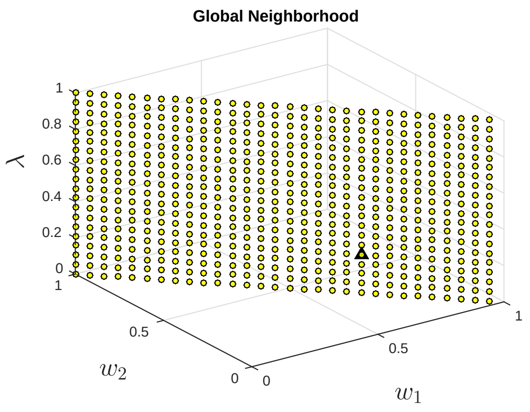
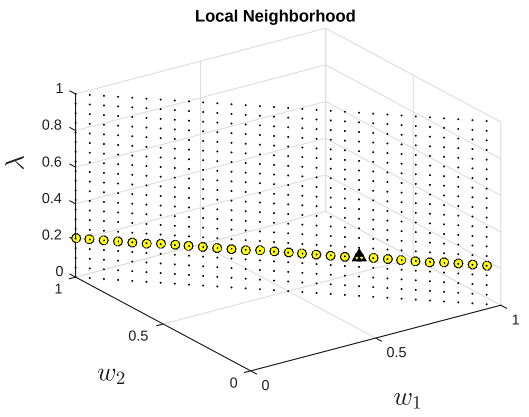


Figure 5.7: Surface of weight vectors W_λ for $k = 2$.

elements with the same λ value. In order to assign the global neighborhood and the local neighborhood for a $\mathbf{w}^{(i)}$ weight vector, we need to compute the Euclidean distances between each weight vector in $W_\lambda \setminus \{\mathbf{w}^{(i)}\}$ and select the T closest weight vectors (in our computations, we set T equal to 10 as in the original MOEA/D for more details see [56]). For each $i = 1, \ldots, \mu$, set $GN(i) = \{ig_1, \ldots, ig_T\}$, where $\mathbf{w}^{(ig_1)}, \ldots, \mathbf{w}^{(ig_T)}$ are the closest weight vectors in the global neighbourhood to $\mathbf{w}^{(i)}$ and set $LN(i) = \{il_1, \ldots, il_T\}$, where $\mathbf{w}^{(il_1)}, \ldots, \mathbf{w}^{(il_T)}$ are the closest weight vectors in the local neighbourhood to $\mathbf{w}^{(i)}$.



(a) Global Neighborhood



(b) Local Neighborhood

Figure 5.8: The yellow points represent the considered neighbourhood for a given $\mathbf{w}^{(i)}$ (black triangle) in both cases globally (left) and locally (right).

5.3.4 Generation of a New Element in the Population

For creating a new element, we first select a $i-th$ sub-problem, then randomly select one index p from $GN(i)$ and generate a new solution $\mathbf{y}$ from $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(p)}$ by using genetic operators. For this case, we use 'rand/1/bin' differential evolution crossover operator with $p = 0.9$, $CR = 1.0$, $F = 0.5$. Since we use a global neighborhood for producing a new element $\mathbf{y}$, we have the risk to jump to another slice, in order to avoid this situation, we set the λ value of $\mathbf{y}$ equal to the λ value of $\mathbf{x}^{(i)}$. Then, we produce a new element $(\mathbf{y}, \lambda_{ji})$.

5.3.5 Mutation Operator

The neighborhood $\mathcal{N}(\mathbf{x}, \lambda)$ is represented by the following expression with center $(\mathbf{x}, \lambda)$ and two radius $r_{\mathbf{x}} > 0$ in $\mathbf{x}$-space and $r_\lambda > 0$ in λ-space,

$$\mathcal{N}(\mathbf{x}, \lambda) := \{\mathbf{x}' \in \mathbb{R}^n | \; ||\mathbf{x} - \mathbf{x}'|| < r_{\mathbf{x}} \wedge \lambda' \in \mathbb{R}^l | \; ||\lambda - \lambda'|| < r_\lambda\}. \tag{5.3}$$

In order to perform mutation over the produced solution $(\mathbf{y}, \lambda_{ji})$ to produce $(\tilde{\mathbf{y}}, \lambda_{ji})$, we use the simple neighborhood search (SNS) for PMOPs proposed in ([58]). The SNS produces new individuals based on Equation (5.3). In order to set r_x and r_λ, we select randomly an index p from the global neighborhood of the $i-th$ element to compute the Euclidean distance between $\mathbf{x}^{(p)}$ and $\mathbf{x}^{(i)}$ for setting r_x. In this first approach, we set $r_\lambda = 0$ to avoid that the number of elements per slice will change. Equation (5.3) proposed a continuous neighborhood in Λ-space, however, the values over Λ are discrete. The probability used for this operator is $pr = 1/\mu$. In the case that one looks for another alternative to perform the mutation operator, one can use the polynomial mutation adapted to the context.

5.3.6 Update a Sub-problem

For updating the $i-th$ sub-problem, we only use the elements from the local neighborhood $LN(i)$. For each index $u \in LN(i)$ if

$$g_T((\tilde{\mathbf{y}}, \lambda_{ji})|\mathbf{w}^{(i)}, \mathbf{z}^{(j)}) \leq g_T((\mathbf{x}^{(u)}, \lambda_j)|\mathbf{w}^{(i)}, \mathbf{z}^{(j)}), \tag{5.4}$$

then set $\mathbf{x}^{(m)} = \tilde{\mathbf{y}}$ and $\mathbf{f}^{(m)} = \mathbf{f}_{\lambda^{(i)}}(\tilde{\mathbf{y}})$.

5.3.7 Update of Z

In the case that the produced $(\tilde{\mathbf{y}}, \lambda_{ji})$ individual contains a better objective value than the best found value according each λ_j in Λ, we have to update z^j if

$$z_r^{(j)} > f_r((\tilde{\mathbf{y}}, \lambda_{ji})), \tag{5.5}$$

then set $z_r^{(j)} = f_r(\tilde{\mathbf{y}}, \lambda_{ji})$ for $r = 1, ..., k$. In the case that the condition is not fulfilled $z_r^{(j)}$ remains the same.

5.3.8 The Algorithm

Algorithm 13 unifies every step of the PMOEA/D. It starts by generating a surface of weight vectors $W_\lambda$and a set of reference points Z. The surface will represent every slice λ_j that belongs to the discretization of the external parameter Λ. Then, we construct the local and global neighborhoods for each $i-th$ weight in the surface W_λ. Both neighbors according to the $i-th$ weight will help us to generate and replace elements into the population every iteration step. Once the initial population is computed the algorithm starts the decomposition process. Every $i-th$ weight in the surface will represent a sub-problem which has to be solve concurrently with the others. The generation process will compute a new element by using the whole information of the current population. Next, by using an utility function we update the sub-problem. Algorithm 13 takes as a basis the framework stated in Algorithm 10. Step 4 of Algorithm 10 comprises steps 3-7 of Algorithm 13 which is the configuration process of the PMOEA/D. The generation of new populations (steps 5-8) of Algorithm 10 is done in steps 8-16. The remaining steps remain the same.

Algorithm 13 PMOEA/D

```
Require: A PMOP, number of individuals per slice $\mu$, number of slices $s$, number of neighbors $T$, and $iter_{max}$ maximum number function evaluations.
Ensure: Final approximation $P^{(final)}$.
 1: Set $\Lambda := \{\lambda_1, \dots, \lambda_s\}$.
 2: Set $t := 0$.
 3: Set $\mu_s = \mu/s$ as the number of individuals in each slice.
 4: Generate a surface $W_\lambda$ of $\mu$ weight vectors as Algorithm (12).
 5: For each index $i = 1, ..., \mu$, set its global neighborhood $GN(i)$ and its local neighborhood $LN(i)$.
 6: Initialize a population $P_t = \{(\mathbf{x}^{(1)}, \lambda_{j1}), \dots, (\mathbf{x}^{(\mu)}, \lambda_{j\mu})\} \subset \mathbb{R}^{n+l}$.
 7: Build the reference points set $\mathbf{Z}$ as Algorithm (11).
 8: for $t = 1$ to $iter_{max}$ do
 9:     for $i = 1$ to $\mu$ do
10:         Randomly select one index $p$ from $GN(i)$ and generate a new solution $(\mathbf{y}, \lambda_{ji})$ from $(\mathbf{x}^{(i)}, \lambda_{ji})$ and $(\mathbf{x}^{(p)}, \lambda_{ji})$ by using genetic operators (DE-crossover 'rand/1/bin').
11:         Apply SNS operator on $(\mathbf{y}, \lambda_{ji})$ to produce $(\tilde{\mathbf{y}}, \lambda_{ji})$.
12:         Evaluate $(\tilde{\mathbf{y}}, \lambda_{ji})$ as $\mathbf{f}_{\lambda^{(i)}}(\tilde{\mathbf{y}}, \lambda_{ji})$.
13:         Update $\mathbf{Z}$ with $\mathbf{f}_{\lambda^{(i)}}(\tilde{\mathbf{y}}, \lambda_{ji})$ using Equation (5.5).
14:         For a random index $u \in LN(i)$, if Equation (5.4), then set $x^{(u)} = \tilde{\mathbf{y}}$ and $\mathbf{f}_{\lambda^{(u)}} = \mathbf{f}_\lambda(\tilde{\mathbf{y}}, \lambda_{ji})$.
15:     end for
16: end for
17: for $j = 1, \dots, s$ do
18:     $P_j^{(final)} := \prod(P_t, j)$
19: end for
20: return $P^{(final)} := \{P_1^{(final)}, ..., P_s^{(final)}\}$
```

The missing step to define is the projection process in the step 18 which is reduce to select those elements with the same λ-value. Since the population is already arranged by slices, we only have to take those elements per each λ_j.

Table 5.4: Test function problems 1.

Parameter dependent multi-objective optimization problems
FDA1 (Dynamic PS, static PF, and convex PFs)
$f_1(\mathbf{x}_I) = x_1$
$g(\mathbf{x}_{II}) = 1 + \sum_{x_i \in \mathbf{x}_{II}} (x_i - G(\lambda))^2$
$h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}}$
$G(\lambda) = \sin(0.5\pi\lambda)$
$\mathbf{x}_I = (x_1) \in [0,1]$, $\mathbf{x}_{II} = (x_2, ..., x_n) \in [-1,1]$, and $\lambda \in [0,2]$
FDA2 (Static PS, Dynamic PF, and convex to nonconvex PFs)
$f_1(\mathbf{x}_I) = x_1$
$g(\mathbf{x}_{II}) = 1 + \sum_{x_i \in \mathbf{x}_{II}} x_i^2$
$h(\mathbf{x}_{III}, f_1, g) = 1 - \left(\frac{f_1}{g}\right)^{\left(H(\lambda) + \sum_{x_i \in \mathbf{x}_{II}} (x_i - H(\lambda))^2\right)}$
$H(\lambda) = 0.75 + 0.7 \sin(0.5\pi\lambda)$
$\mathbf{x}_I = (x_1) \in [0,1]$, $\mathbf{x}_{II}, \mathbf{x}_{III} \in [-1,1]$, and $\lambda \in [0,3]$

5.4 Experiments and results

In order to test the proposed algorithm we use the work presented in [119]. The authors proposed four different types of a PMOPs based on the changes on the Pareto set and front according the effect that produces a modification of the value of an external parameter. For experimental purposes the authors also include five different PMOPs. Tables 5.4 and 5.5 describe the formulation of each of the 5 test problems.

FDA1, FDA2, and FDA3 are bi-objective optimization problems and it second objective is defined as $f_2(\mathbf{x}_I, \mathbf{x}_{II}) = g * h$. Figure 5.9 depicts the family of Pareto fronts of FDA1 to FDA3 in two different angles. In the case of FDA4 and FDA5, we use $k = 3$.

Table 5.5: Test function problems 2.

Parameter dependent multi-objective optimization problems

FDA3 (Dynamic PS, Dynamic PF, and nonconvex PFs)

$f_1(\mathbf{x}_I) = \sum_{x_i \in \mathbf{x}_i} x_i^{D(t)}$

$g(\mathbf{x}_{II}) = 1 + G(\lambda) + \sum_{x_i \in \mathbf{x}_{II}} (x_i - G(\lambda))^2$

$h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}}$

$G(\lambda) = |\sin(0.5\pi\lambda)|$

$D(\lambda) = 10^{2\sin(0.5\pi\lambda)}$

$\mathbf{x}_I \in [0,1]$, $\mathbf{x}_{II} \in [-1,1]$, and $\lambda \in [0,2]$

FDA4 (Dynamic PS, Static PF, and nonconvex PFs)

$f_1(\mathbf{x}) = (1 + g(\mathbf{x}_{II})) \prod_{i=1}^{k-1} \cos\left(\frac{x_i\pi}{2}\right)$

$f_m(\mathbf{x}) = (1 + g(\mathbf{x}_{II})) \prod_{i=1}^{k-m} \cos\left(\frac{x_i\pi}{2}\right) \sin\left(\frac{x_{k-m+1}\pi}{2}\right)$, $m = 2 : k-1$

$f_k(\mathbf{x}) = (1 + g(\mathbf{x}_{II})) \sin\left(\frac{x_1\pi}{2}\right)$

$g(\mathbf{x}_{II}) = \sum_{x_i \in \mathbf{x}_{II}} (x_i - G(\lambda))^2$

$G(\lambda) = |\sin(0.5\pi\lambda)|$

$\mathbf{x}_{II} = (x_k, ..., x_n)$, $x_i \in [0,1]$ and $\lambda \in [0,2]$

FDA5 (Dynamic PS, Dynamic PF, and nonconvex PFs)

$f_1(\mathbf{x}) = (1 + g(\mathbf{x}_{II})) \prod_{i=1}^{k-1} \cos\left(\frac{y_i\pi}{2}\right)$

$f_m(\mathbf{x}) = (1 + g(\mathbf{x}_{II})) \prod_{i=1}^{k-m} \cos\left(\frac{y_i\pi}{2}\right) \sin\left(\frac{y_{k-m+1}\pi}{2}\right)$, $m = 2 : k-1$

$f_k(\mathbf{x}) = (1 + g(\mathbf{x}_{II})) \sin\left(\frac{y_1\pi}{2}\right)$

$g(\mathbf{x}_{II}) = G(\lambda) + \sum_{x_i \in \mathbf{x}_{II}} (x_i - G(\lambda))^2$

$y_i = x_i^{D(t)}$ for $i = 1, ..., (k-1)$

$G(\lambda) = |\sin(0.5\pi\lambda)|$, $D(\lambda) = 1 + 100\sin^4(0.5\pi\lambda)$

$\mathbf{x}_{II} = (x_k, ..., x_n)$, $x_i \in [0,1]$ and $\lambda \in [0,2]$

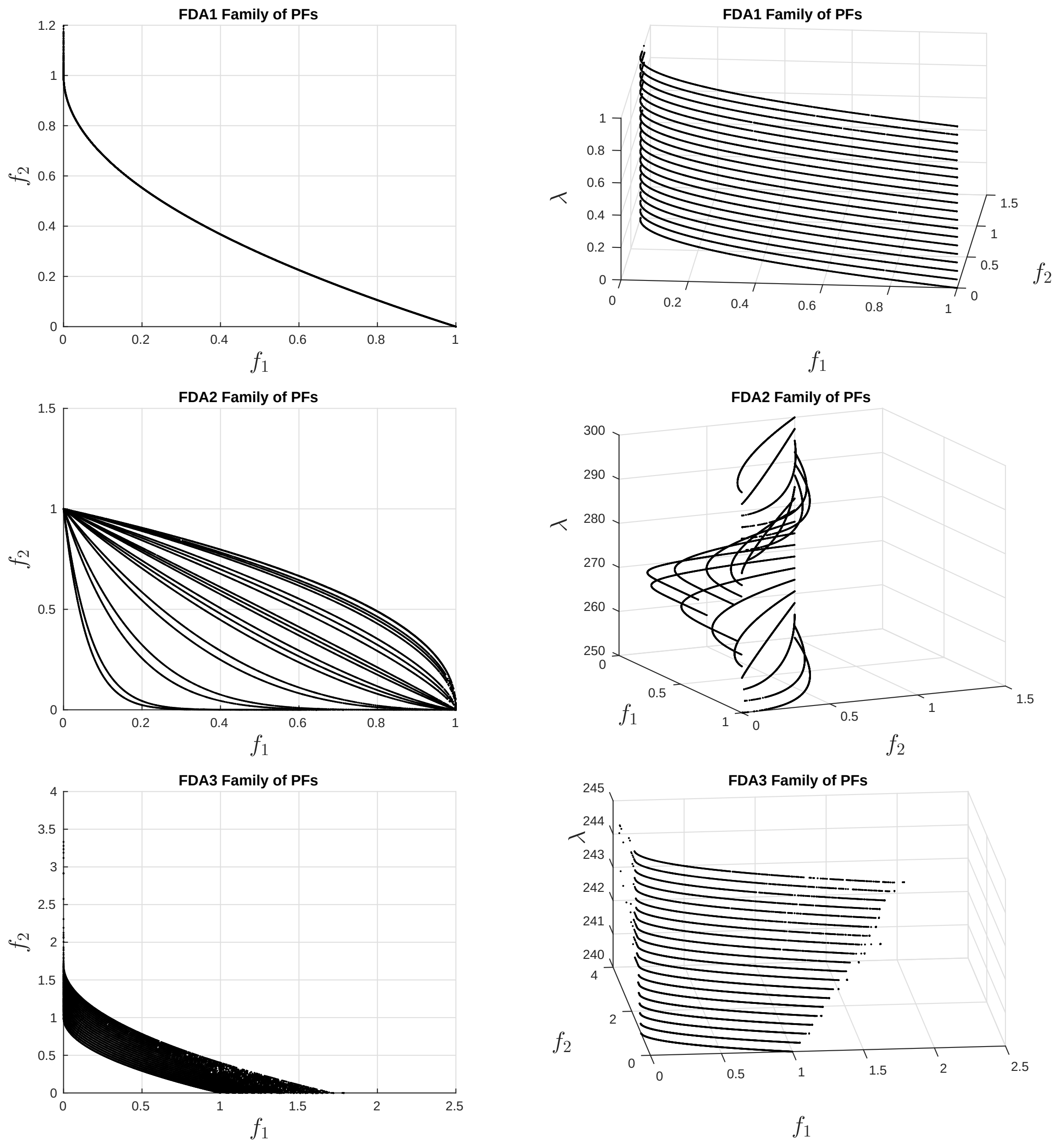


Figure 5.9: The images refers to the real family of Pareto fronts in 2D and 3D for FDA1 (first row), FDA2 (second row), and FDA3 (third row).

Table 5.6: Test function problems.

Parameter dependent multi-objective optimization problems
FDA1 (Dynamic PS, static PF, and convex PFs)
$n = 20$, $\lvert\mathbf{x}_I\rvert = 1$, $\lvert\mathbf{x}_{II}\rvert = 19$, $k = 2$
FDA2 (Static PS, Dynamic PF, and convex to nonconvex PFs)
$n = 21$, $\lvert\mathbf{x}_I\rvert = 1$, $\lvert\mathbf{x}_{II}\rvert = 10$, $\lvert\mathbf{x}_{III}\rvert = 10$, $k = 2$
FDA3 (Dynamic PS, Dynamic PF, and nonconvex PFs)
$n = 20$, $\lvert\mathbf{x}_I\rvert = 5$, $\lvert\mathbf{x}_{II}\rvert = 15$, $k = 2$
FDA4 (Dynamic PS, Static PF, and nonconvex PFs)
$n = 12$, $\lvert\mathbf{x}_I\rvert = 2$, $\lvert\mathbf{x}_{II}\rvert = 10$, $k = 3$
FDA5 (Dynamic PS, Dynamic PF, and nonconvex PFs)
$n = 12$, $\lvert\mathbf{x}_I\rvert = 2$, $\lvert\mathbf{x}_{II}\rvert = 10$, $k = 3$

For the comparison, we use the averaged Hausdorff distance (AHD) for PMOPs between $\mathbf{f}_\lambda(P_{\mathbf{x}_\lambda})$ and $\mathbf{f}_\lambda(A)$ defined as

$$\Delta_p(\mathbf{f}_\lambda(P_{\mathbf{x}_\lambda}), \mathbf{f}_\lambda(A)) := \left(\frac{1}{s} \sum_{i=1}^{s} \Delta_q(\mathbf{f}_\lambda(\pi(A)), \mathbf{f}_\lambda(P_{\mathbf{x}_\lambda}))^p \right)^{1/p}, \tag{5.6}$$

where A represents the approximation or final population P given by an algorithm, s is the number of slices, the function π yield the non-dominated individuals from A and

$$\Delta_q(\mathbf{f}_\lambda(\pi(A)), \mathbf{f}_\lambda(P_{\mathbf{x}_\lambda})) = \max\left(IGD_q(\mathbf{f}_\lambda(\pi(A)), \mathbf{f}_\lambda(P_{\mathbf{x}_\lambda})), GD_q(\mathbf{f}_\lambda(\pi(A)), \mathbf{f}_\lambda(P_{\mathbf{x}_\lambda}))\right). \tag{5.7}$$

The q value represents the natural parameter for computing the AHD, while the new parameter q controls the effect of each slice in the approximation. The configuration of each of the problems is included in Table 5.6. Tables 5.7, and 5.8 shows the resulting Δ_p (adapted to the context of PMOPs) values for the final approximation P^{final} produced by:

- PMOEA/D: our algorithm.

- MOEAD-1250: standard MOEA/D with a budget of N/ξ function evaluations for each slice, where N is the number of function evaluations used for PMOP-EMOA to get the entire family of Pareto fronts, and ξ is the number of slices. For this experiment $N = 25000$ and $\xi = 20$ resulting in a budget of 1250 functions evaluations per each slice.

- MOEAD-25000: standard MOEA/D with a budget of N function evaluations for each slice, where N is the number of function evaluations used for PMOP-EMOA to get the entire family of Pareto fronts. For this experiment $N = 25000$.

The references of the family of Pareto fronts are computed by an adaption of the NBI method (see [20]) for the context of PMOPs. The comparison tables, also contains the α-value for the Wilcoxon rank sum test. The α-value is obtained comparing the results of PMOEA/D against MOEAD-1250 and MOEAD-25000. Problems marked with (*) reflect a statistically significant result based on the Wilcoxon Rank test using a significance level $\alpha = 0.05$.

Table 5.7: Comparison of PMOEA/DA against MOEAD-1250 and MOEAD-25000 over five test functions. The indicator values are averaged over 20 independent runs with $p = 2$ and $q = 2$.

	$\Delta_p(\mathbf{f}_\lambda(P_{\mathbf{x}_\lambda}), \mathbf{f}_\lambda(P^{final}))$					
	PMOEA/D		MOEAD-1250		MOEAD-25000	
Problem	Mean	Std	Mean	Std	Mean	Std
FDA1 (*)	0.040048	0.001029	0.071082	0.011025	0.045591	0.000074
α-value			0.00000006		0.00000006	
FDA2 (*)	0.041030	0.003328	0.278690	0.036303	0.036819	0.001216
α-value			0.00000007		0.00001807	
FDA3 (*)	0.090148	0.015753	0.240333	0.066283	0.146737	0.043556
α-value			0.00000006		0.00019970	
FDA4 (*)	0.083534	0.013532	0.412342	0.082342	0.099232	0.059283
α-value			0.00000008		0.00539222	
FDA5 (*)	0.073452	0.032345	0.324223	0.102483	0.062452	0.034958
α-value			0.00000002		0.00993422	

Table 5.8: Comparison of PMOEA/D against MOEAD-1250 and MOEAD-25000 over five test functions. The indicator values are averaged over 20 independent runs with $p = 2$ and $q = \infty$.

	$\Delta_p(\mathbf{f}_\lambda(P_{\mathbf{x}_\lambda}), \mathbf{f}_\lambda(P^{final}))$					
	PMOEA/D		MOEAD-1250		MOEAD-25000	
Problem	Mean	Std	Mean	Std	Mean	Std
FDA1	0.202191	0.004335	0.197318	0.022580	0.227145	0.000261
α-value			0.12643061		0.00000006	
FDA2 (*)	0.116906	0.009268	0.534042	0.057807	0.117264	0.002353
α-value			0.00000006		0.031497988	
FDA3 (*)	0.920543	0.017751	0.939615	0.042377	0.934756	0.006885
α-value			0.04643094		0.00000006	
FDA4 (*)	0.624223	0.023242	0.883431	0.111344	0.735743	0.032958
α-value			0.00424526		0.00352226	
FDA5 (*)	0.912322	0.023984	0.935323	0.094232	0.952342	0.064837
α-value			0.00999426		0.00653226	

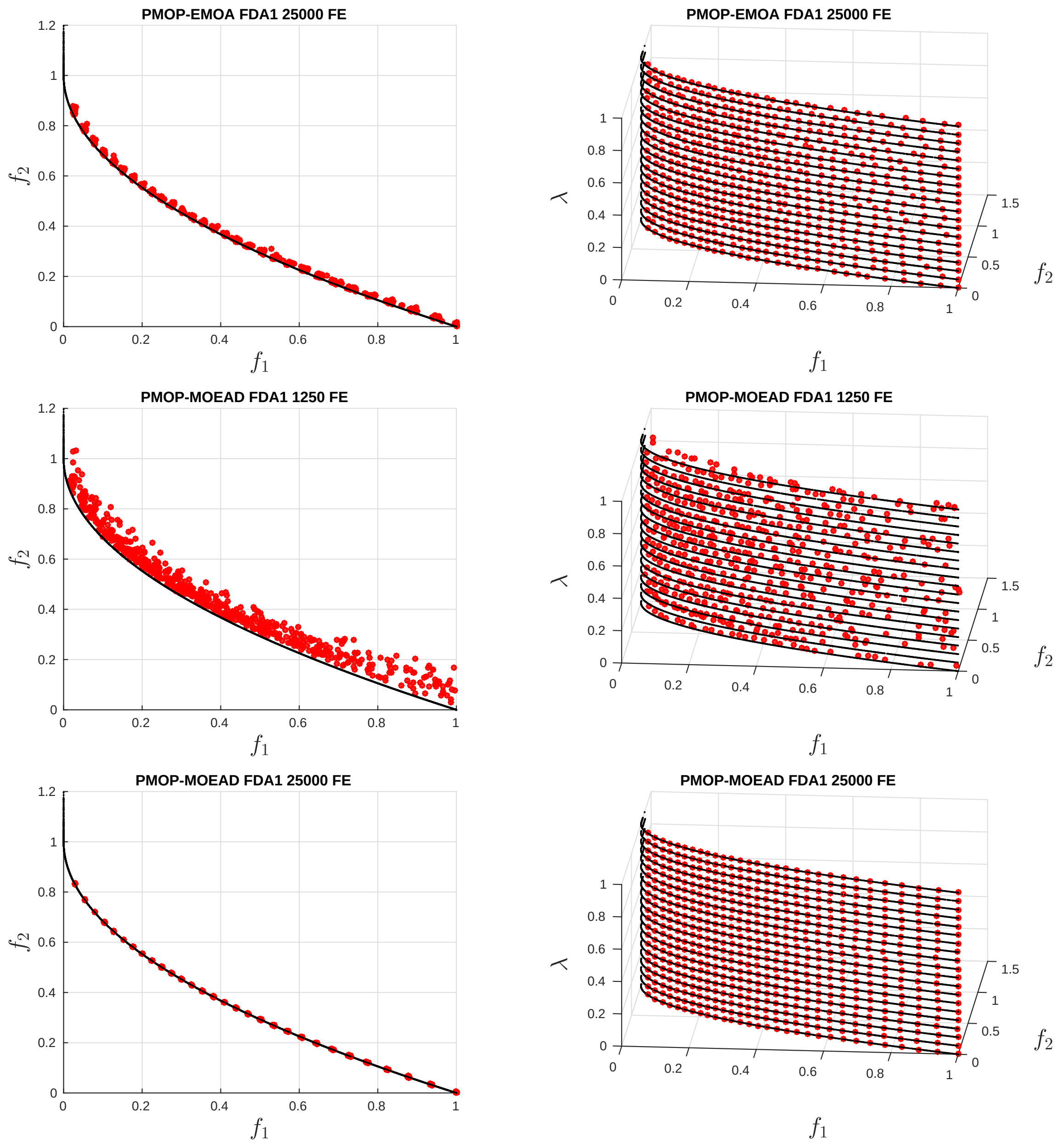


Figure 5.10: Results for PMOEA/D (first row), MOEAD-1250 (second row), and MOEAD-25000 (third row) for FDA1.

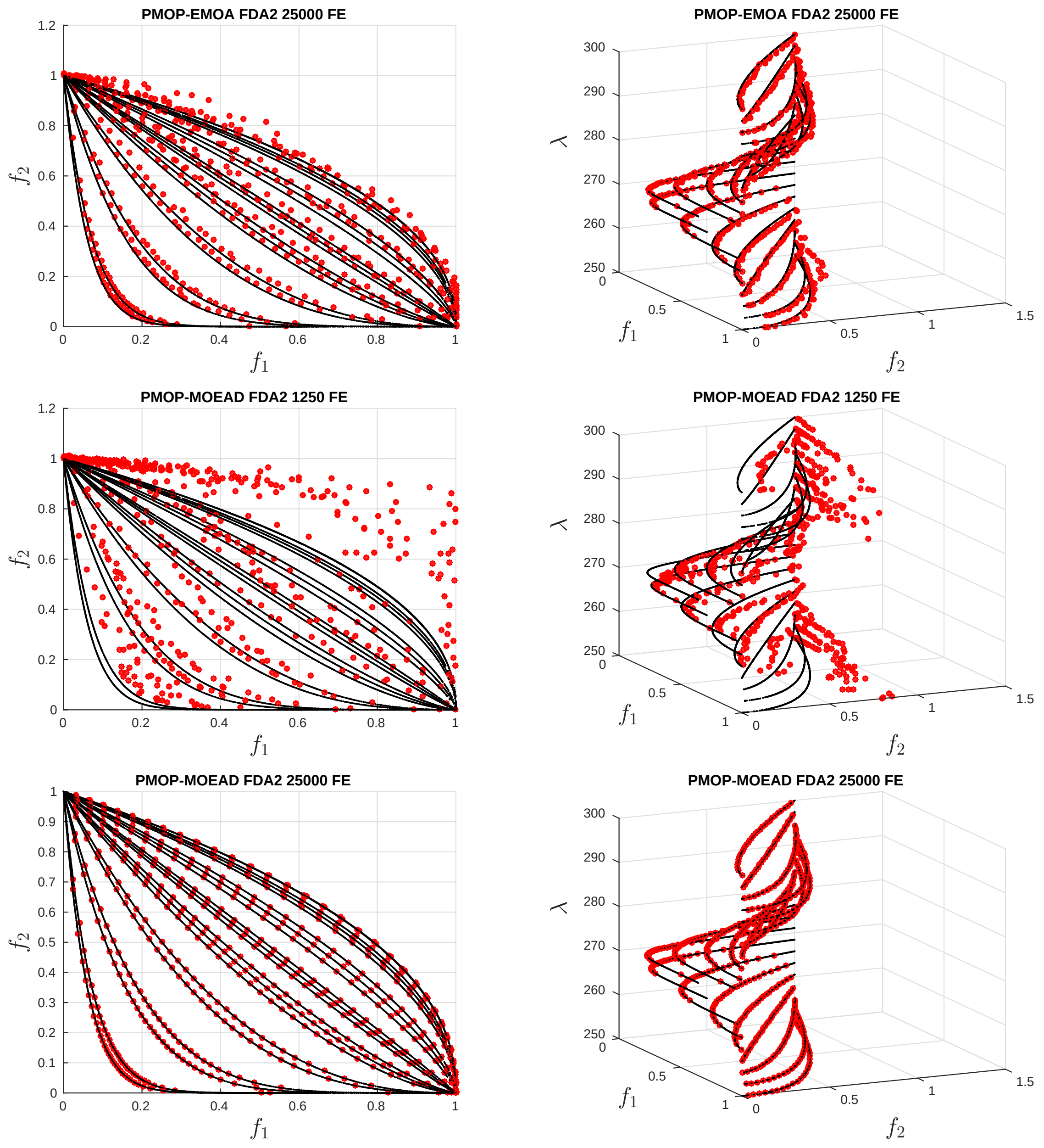


Figure 5.11: Results for PMOEA/D (first row), MOEAD-1250 (second row), and MOEAD-25000 (third row) for FDA2.

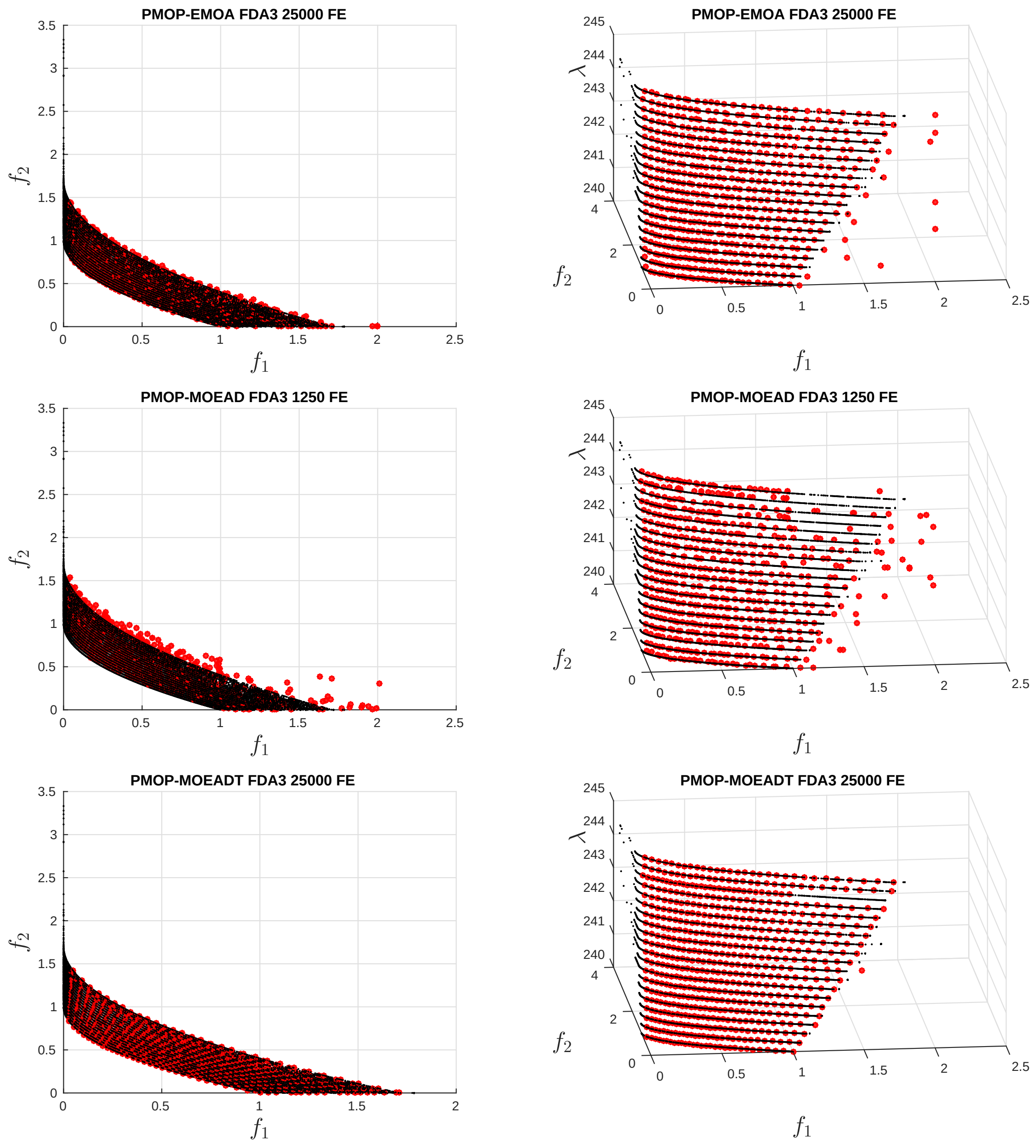


Figure 5.12: Results for PMOEA/D (first row), MOEAD-1250 (second row), and MOEAD-25000 (third row) for FDA3.

6 | Conclusions and Future Work

Here, we state the conclusions of the work done within this thesis project. Further, some paths for possible future work are detected in order to continue this work.

- We presented a new region division for MOPs with $k > 2$ objectives. The new region division procedure allows us to divide the whole objective space for MOPs with $k > 2$ objectives and it also locates the points assigned for local search over this region division. We determined three different distance regions ('far away', 'in between', and 'near'). Therefore, given an unconstrained MOP with all its objectives differentiable, we are able to locate a given point $\mathbf{x} \in \mathbb{R}^n$ within the proposed three distance regions.

- We presented a new local search strategy called HVDS for $k > 2$ objectives. This local search procedure aim to maximize the hypervolume. The strategy includes the proposed region division of the objective space in order to perform a specific local movement in every region according to the location of a given solution or solutions assigned for local search. Then, the HVDS can move points over each of the distance region aiming for maximizing the hypervolume indicator.

- The HVDS was included into the SMS-EMOA as a generational operator with a certain probability. The resulted memetic strategy outperformed the basic version of SMS-EMOA for problems with $k = 2, 3$ objectives. The SMS-EMOA-HVDS showed to have better results in eleven of fourteen test problems. Hence, the proposed memetic algorithm improve results obtained by state-of-the-art algorithms.

- We observed that the SMS-EMOA-HVDS works properly for problems with convex, concave, and convex-concave Pareto fronts. However, in the case of disconnected Pareto fronts we lost against the SMS-EMOA without local search. Then, it is needed to analyse more deeply why the local search procedure cannot outperform the state-of-the-art algorithms.

- In the case of the hypervolume gradient flow, we detected the presence of a certain 'creepiness' in the flow coming from the fact that the norms of the sub-gradients are in different ranges. This behavior was detected over several test problems. Such a creeping behavior led in some cases to domination of points in the population. Thus, it is not sufficient for an initial population that its individuals are mutually non-dominated.

- We proposed the hypervolume Newton method (HNM) for hypervolume maximization for the treatment of sufficiently smooth continuous MOPs. In order to present this method, we first used an approximation of the hypervolume Hessian matrix. Then, we derived analytically the hypervolume Hessian matrix for a given MOP and population. This matrix is fully expressed for the case of two objectives. Thus, the HNM can be applied over population for a given bi-objective optimization problem.

- The HNM was compared as standalone algorithm against a first order technique base on the hypervolume gradient the gradient ascent hypervolume method (GAH). Results showed that the HNM reaches the best hypervolume value faster than GAH for a given population with a certain number of elements. We observed empirically quadratic converge rates of the HNM. Then, the HNM can be the perfect choice to be coupled with an evolutionary technique to designed a novel memetic algorithm.

- We designed a novel memetic hypervolume based evolutionary algorithm, the SMS-EMOA-HNM. This algorithm outperformed the basic version of SMS-EMOA. Different than the SMS-EMOA-HVDS, here we integrated the local search strategy as a refinement operator. In other words, we applied the HNM on the best-found population. The results showed that we reached higher hypervolume values than the basic version of the SMS-EMOA using less function evaluations. Hence, we have successfully adapted the HNM into an evolutionary algorithm that aims for maximizing the hypervolume indicator.

- We developed memetic algorithms based on local searches strategies that aims for improving the hypervolume value. The algorithms shows to have better numerical results than the state of the art algorithms. We concluded that the integration of local search strategies improve the performance of a given evolutionary approach. This improvement can be done over the optimization process or as a refinement process.

- We developed a simple neighborhood search algorithm in the context of PMOPs. Such an algorithm shows that both movements toward and along the set of interest are inherent in stochastic local search. Hence, to approximate the entire solution set of a given PMOP is well-conditioned for set based stochastic algorithms.

- We presented a general scheme to tackle a given PMOP in a non-sequential way. The algorithm starts from the discretization of the external parameter space. Then, it requires an adaption of a MOEA to produce the whole approximation of the solution set. Thus, we conclude that it is possible to compute the whole approximation of the solution set of a PMOP in one single run of an evolutionary approaches.

- We have seen that decomposition helps to tackle a PMOP as a whole, in this case, we presented the PMOEA/D for generating a whole approximation of a given PMOP. The algorithm produced competitive an well-distributed approximation of the family of Pareto sets and fronts.

- The exploitation of the knowledge produced by the SLS and SNS leaded us to design an algorithm able to compute an approximation of the family of Pareto sets and fronts. The interaction between elements from different λ-values help to push the whole approximation slices toward the solution set.

6.1 Future Work

We detected some possible paths for future work:

- In the case of the hypervolume Newton method new strategies to manage constraints are desired. Also, a more detailed analysis of the properties of the HNM would be helpful.

- Quasi-Newton methods also can be adapted to replace the computation of the Hessian matrix in order to reduce computational effort an save some function evaluations.

- For the memetic strategy, we detected that it is also possible to integrated the HNM as an evolutionary operator and not only as a refinement strategy.

- An extension of the proposed Newton framework for other performance indicators such as Δ_p, $R2$, among others is also desired.

- The development of an evolutionary strategy that aim to solve equality constrained MOPs is desired, in order to couple with the HNMEQ. For instance, the adaption of ELSA to the context of MOPs.

- In the case the PMOEA/D, more comparison are needed, for instance convergence graphs to see the evolution of the algorithm.

- Finally, the design of a memetic algorithm for the treatment of PMOPs is of our interest. Since, we have both an evolutionary technique (PMOEA/D) and a local search strategy (λ-DS) for the context of PMOPs.

A | Test Problems

In the following, we present the definition of the test problems used throughout this thesis work.

Table A.1: Test problems (part 1).

Fonseca1 (Concave Pareto front) $S = [-5,5]^2$ $f_1(\mathbf{x}) = 1 - \exp^{(-(x_1-1)^2-(x_2+1)^2)}$ $f_2(\mathbf{x}) = 1 - \exp^{(-(x_1+1)^2-(x_2-1)^2)}$
Modified ZDT1 (Convex Pareto front) $S = [0,1] \times [-1,1]^{n-1}$ $f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})(2 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})})$ $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=1}^{n} x_i^2$
Modified ZDT2 (Concave Pareto front) $S = [0,1] \times [-1,1]^{n-1}$ $f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})(2 - (f_1(\mathbf{x})/g(\mathbf{x}))^2)$ $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=1}^{n} x_i^2$
Modified ZDT3 (Disconnected Pareto front) $S = [0,1] \times [-1,1]^{n-1}$ $f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})(2 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})}) - (f_1(\mathbf{x})/g(\mathbf{x}))\sin(10\pi f_1(\mathbf{x})))$ $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=1}^{n} x_i^2$

Table A.2: Test problems (part 2).

Modified ZDT6 (Concave Pareto front) $S = [0,1] \times [-1,1]^{n-1}$ $f_1(\mathbf{x}) = 1 - \exp -4x_1$ $f_2(\mathbf{x}) = g(\mathbf{x})(2 - (f_1(\mathbf{x})/g(\mathbf{x}))^2)$ $g(\mathbf{x}) = 1 + \frac{9}{n-1}\sum_{i=1}^{n} x_i^2$
DTLZ 1 (Linear Pareto front) $S = [0,1]^n$ $f_1(\mathbf{x}) = (1 + g(\mathbf{x}))0.5\prod_{i=1}^{k-1} x_i$ $f_{m=2:k-1}(\mathbf{x}) = (1 + g(\mathbf{x}))0.5(\prod_{i=1}^{k-m} x_i)(1 - x_{k-m+1})$ $f_k(\mathbf{x}) = (1 + g(\mathbf{x}))0.5(1 - x_1)$ $g(\mathbf{x}) = 100(n + \sum_{i=1}^{n}((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5))))$
DTLZ 2 (Concave Pareto front) $S = [0,1]^n$ $f_1(\mathbf{x}) = (1 + g(\mathbf{x}))\prod_{i=1}^{k-1} \cos(x_i\frac{\pi}{2})$ $f_{m=2:k-1}(\mathbf{x}) = (1 + g(\mathbf{x}))(\prod_{i=1}^{k-m} \cos(x_i\frac{\pi}{2}))\sin(x_{k-m+1}\frac{\pi}{2})$ $f_k(\mathbf{x}) = (1 + g(\mathbf{x}))\sin(x_1\frac{\pi}{2})$ $g(\mathbf{x}) = \sum_{i=1}^{n}(x_i - 0.5)^2$
DTLZ 3 (Concave Pareto front) $S = [0,1]^n$ $f_1(\mathbf{x}) = (1 + g(\mathbf{x}))\prod_{i=1}^{k-1} cos(x_i\frac{\pi}{2})$ $f_{m=2:k-1}(\mathbf{x}) = (1 + g(\mathbf{x}))(\prod_{i=1}^{k-m} \cos(x_i\frac{\pi}{2}))\sin(x_{k-m+1}\frac{\pi}{2})$ $f_k(\mathbf{x}) = (1 + g(\mathbf{x}))sin(x_1\frac{\pi}{2})$ $g(\mathbf{x}) = 100(n + \sum_{i=1}^{n}((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5))))$
DTLZ 7 (Disconnected Pareto front) $S = [0,1]^n$ $f_1(\mathbf{x}) = x_1$ $f_{m=2:k-1}(\mathbf{x}) = x_m$ $f_k(\mathbf{x}) = (1 + g(\mathbf{x}))(k - \sum_{i=1}^{k-1}(\frac{f_i}{1+g(\mathbf{x})}(1 + \sin(3\pi f_i))))$ $g(\mathbf{x}) = 1 + \frac{9}{\|x_k\|}\sum_{i=1}^{n} x_i$

Bibliography

[1] M. Brown and R. E. Smith. Directed Multi-Objective Optimisation. *International Journal of Computers, Systems and Signals*, 6(1):3–17, 2005.

[2] E. Zitzler and L. Thiele. Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study. In *Conference on Parallel Problem Solving from Nature (PPSN V)*, pages 292–301, Berlin, 1998. Springer.

[3] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge. Comparing Images Using the Hausdorff Distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993.

[4] D. A. Van Veldhuizen and G. B. Lamont. On measuring multiobjective evolutionary algorithm performance. In *2000 Congress on Evolutionary Computation*, volume 1, pages 204–211, Piscataway, New Jersey, July 2000. IEEE Service Center.

[5] C. A. Coello Coello and N. Cruz Cortés. Solving Multiobjective Optimization Problems using an Artificial Immune System. *Genetic Programming and Evolvable Machines*, 6(2):163–190, June 2005.

[6] O. Schütze, X. Esquivel, A. Lara, and C. A. Coello Coello. Using the averaged Hausdorff distance as a performance measure in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 16(4):504–522, 2012.

[7] Chi-Keong Goh and Kay Chen Tan. *Dynamic Evolutionary Multi-objective Optimization*, pages 125–152. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[8] Carlo Raquel and Xin Yao. *Dynamic Multi-objective Optimization: A Survey of the State-of-the-Art*, pages 85–106. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[9] Radhia Azzouz, Slim Bechikh, and Lamjed Ben Said. *Dynamic Multi-objective Optimization Using Evolutionary Algorithms: A Survey*, pages 31–70. Springer International Publishing, Cham, 2017.

[10] W. Karush. *Minima of Functions of Several Variables with Inequalities as Side Conditions*, pages 217–245. Springer Basel, Basel, 2014.

[11] H. W. Kuhn and A. W. Tucker. Nonlinear Programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, Berkeley, Calif., 1951. University of California Press.

[12] J. Nocedal and S. Wright. *Numerical Optimization.* Springer Series in Operations Research and Financial Engineering. Springer, 2006.

[13] V. Pareto. *Cours D'Economie Politique, volume I and II.* Librairie Droz, 1964.

[14] F. Y. Edgeworth. *Mathematical Psychics: An essay on the application of mathematics to the moral sciences*, volume 10. Kegan Paul, 1881.

[15] M. Köppen and K. Yoshida. Substitute distance assignment in NSGA-II for handling many-objective optimization problems. In S. Obayashi et al., editor, *Evolutionary Multi-Criterion Optimization, 4th International Conference (EMO 2007)*, pages 727–741, Matshushima, Japan, March 2007. Springer. Lecture Notes in Computer Science Vol. 4403.

[16] O. Schütze. *Set Oriented Methods for Global Optimization.* Dissertation, Universität Paderborn, December 2004.

[17] L.A. Zadeh. Optimality and Non-Scalar-Valued Performance Criteria. *IEEE Transactions on Automatic Control*, 8:59–60, 1963.

[18] S. Gass and T. Saaty. The Computational Algorithm for the Parametric Objective Function. *Naval Research Logistics Quarterly*, 2(1):39–45, 1955.

[19] V. J. Bowman. On the Relationship of the Tchebycheff Norm and the Efficient Frontier of Multiple-Criteria Objectives. In *Multiple Criteria Decision Making*, volume 130 of *Lecture Notes in Economics and Mathematical Systems*, pages 76–86. Springer Berlin Heidelberg, 1976.

[20] I. Das and J. Dennis. Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems. *SIAM Journal on Optimization*, 8(3):631–657, 1998.

[21] Y. Y. Haimes, D. A. Wismer, and L. S. Lasdon. On a Bicriterion Formulation of the Problems of Integrated System Identification and System Optimization. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-1(3):296–297, July 1971.

[22] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems.* Springer, 2nd edition edition, 2007.

[23] K. M. Miettinen. *Nonlinear Multiobjective Optimization.* Kluwer Academic Publishers, 1999.

[24] S. Schäffler, R. Schultz, and K. Weinzierl. A Stochastic Method for the Solution of Unconstrained Vector Optimization Problems. *Journal of Optimization Theory and Applications*, 114(1):209–222, 2002.

[25] M. Dellnitz, O. Schütze, and T. Hestermeyer. Covering Pareto Sets by Multilevel Subdivision Techniques. *Journal of Optimization Theory and Applications*, 124(1):113–136, 2005.

[26] J. Fliege and B. Svaiter. Steepest descent Methods for Multicriteria Optimization. *Mathematical Methods of Operations Research*, 51(3):479–494, 2000.

[27] P. Bosman and D. de Jong. Exploiting gradient information in numerical multi–objective evolutionary optimization. In *Proceedings of the 2005 conference on Genetic and evolutionary computation*, GECCO '05, pages 755–762, New York, NY, USA, 2005. ACM.

[28] O. Schütze, A. Martín, A. Lara, S. Alvarado, E. Salinas, and C. A. C. Coello. The directed search method for multi-objective memetic algorithms. *Computational Optimization and Applications*, 63(2):305–332, 2016.

[29] C. Hillermeier. *Nonlinear Multiobjective Optimization: A Generalized Homotopy Approach*. Springer, 2001. ISBN-13:9783764364984.

[30] O. Schütze, A. Dell'Aere, and M. Dellnitz. On Continuation Methods for the Numerical Treatment of Multi-Objective Optimization Problems. In Jürgen Branke, Kalyanmoy Deb, Kaisa Miettinen, and Ralph E. Steuer, editors, *Practical Approaches to Multi-Objective Optimization*, number 04461 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2005. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

[31] A. Martín and O. Schütze. *A New Predictor Corrector Variant for Unconstrained Bi-objective Optimization Problems*, pages 165–179. Springer International Publishing, Cham, 2014.

[32] A. Martín and O. Schütze. Pareto Tracer: A Predictor-Corrector Method for Multi-objective Optimization Problems. *Engineering Optimization*, 2017.

[33] Benjamin Martin, Alexandre Goldsztejn, Laurent Granvilliers, and Christophe Jermann. On continuation methods for non-linear bi-objective optimization: towards a certified interval-based approach. *Journal of Global Optimization*, 64(1):3–16, Jan 2016.

[34] Benjamin Martin, Alexandre Goldsztejn, Laurent Granvilliers, and Christophe Jermann. Certified parallelotope continuation for one-manifolds. *SIAM Journal on Numerical Analysis*, 51(6):3373–3401, 2013.

[35] M. Emmerich, A. Deutz, and N. Beume. Gradient-Based/Evolutionary Relay Hybrid for Computing Pareto Front Approximations Maximizing the S-Metric. In Thomas Bartz-Beielstein et al., editors, *Hybrid Metaheuristics*, volume 4771 of *Lecture Notes in Computer Science*, pages 140–156. Springer Berlin Heidelberg, 2007.

[36] M. Emmerich and A. Deutz. Time Complexity and Zeros of the Hypervolume Indicator Gradient Field. In Oliver Schütze et al., editors, *EVOLVE - A Bridge*

between Probability, Set Oriented Numerics, and Evolutionary Computation III, volume 500 of *Studies in Computational Intelligence*, pages 169–193. Springer International Publishing, 2014.

[37] H. Wang, Y. Ren, A. Deutz, and M. Emmerich. On Steering Dominated Points in Hypervolume Indicator Gradient Ascent for Bi-Objective Optimization. In Oliver Schütze, Leonardo Trujillo, Pierrick Legrand, and Yazmin Maldonado, editors, *NEO 2015: Results of the Numerical and Evolutionary Optimization Workshop NEO 2015 held at September 23-25 2015 in Tijuana, Mexico*, pages 175–203. Springer International Publishing, Cham, 2017.

[38] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial intelligence through simulated evolution.* Wiley, Chichester, WS, UK, 1966.

[39] H. P. Schwefel. *Kybernetische Evolution als Strategie der Experimentellen Forschung in der Strömungstechnik.* Dissertation, Technical University of Berlin, 1965.

[40] I. Rechenberg. Cybernetic solution path of an experimental problem. *Journal of Computer and Communications*, 1965.

[41] R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*, pages 39–43. IEEE, 1995.

[42] R. Storn and K. Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, 1997.

[43] J. Jahn. Multiobjective search algorithm with subdivision technique. *Computational Optimization and Applications*, 35(2):161–175, 2006.

[44] M. Dellnitz, O. Schütze, and S. Sertl. Finding zeros by multilevel subdivision techniques. *IMA Journal of Numerical Analysis*, 22(2):167–185, 2002.

[45] M. Dellnitz and O. Junge. An adaptive subdivision technique for the approximation of attractors and invariant measures. *Computing and Visualization in Science*, 1(2):63–68, 1998.

[46] M. Dellnitz and A. Hohmann. A subdivision algorithm for the computation of unstable manifolds and global attractors. *Numerische Mathematik*, 75(3):293–317, 1997.

[47] M. Dellnitz and O. Junge. Set oriented numerical methods for dynamical systems. *Handbook of Dynamical Systems*, 2:221–264, 2002.

[48] P. Deuflhard, M. Dellnitz, O. Junge, and C. Schütte. Computation of essential molecular dynamics by subdivision techniques. In *Computational molecular dynamics: challenges, methods, ideas*, pages 98–115. Springer, 1999.

[49] C. S. Hsu. Cell-to-cell mapping: A method of global analysis for nonlinear systems. *Applied mathematical sciences*, 1987.

[50] C. Hernández, O. Schütze, and J. Q. Sun. Computing the Set of Approximate Solutions of a Multi-Objective Optimization Problem by Means of Cell Mapping Techniques. In *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation IV*, Advances in Intelligent Systems and Computing. Springer, 2013.

[51] C. Hernández, Y. Narajani, Y. Sardahi, W. Liang, and O. Schütze. Simple cell mapping Method for Multiobjective Optimal PID Control Design. *International Journal of Dynamics and Control*, 2013.

[52] C. A. Coello Coello. Evolutionary Multi-Objective Optimization: A Historical View of the Field. *Computational Intelligence Magazine, IEEE*, 1(1):28–36, 2006.

[53] A. Abraham, L. Jain, and R. Goldberg. *Evolutionary Multiobjective Optimization-Theoretical Advances and Applications*. Springer, 2005.

[54] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons, Inc., New York, NY, USA, 2001.

[55] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[56] Q. Zhang and H. Li. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[57] N. Beume, B. Naujoks, and M. Emmerich. An EMO Algorithm Using the Hypervolume Measure as Selection Criterion. *In Coello, C. A. C.,Aguirre, A. H., and Zitzler, E., editors, EMO*, pages 62–76, 2005.

[58] Víctor Adrián Sosa Hernández, Oliver Schütze, Heike Trautmann, and Günter Rudolph. On the behavior of stochastic local search within parameter dependent mops. In António Gaspar-Cunha, Carlos Henggeler Antunes, and Carlos Coello Coello, editors, *Evolutionary Multi-Criterion Optimization: 8th International Conference, EMO 2015, Guimarães, Portugal, March 29 –April 1, 2015. Proceedings, Part II*, pages 126–140, Cham, 2015. Springer International Publishing.

[59] Victor Adrian Sosa Hernandez. On the Numerical Treatment of Parametric Multi-Objective Optimization Problems and Memetic Evolutionary Algorithms. Msc thesis, CINVESTAV-IPN, December 2013.

[60] C. K. Goh and K. C. Tan. A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 13(1):103–127, Feb 2009.

[61] S. Jiang and S. Yang. A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 21(1):65–82, Feb 2017.

[62] Mardé Helbig and Andries P. Engelbrecht. *Dynamic Multi-Objective Optimization Using PSO*, pages 147–188. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[63] M. G. Martínez-Peñaloza and E. Mezura-Montes. Immune generalized differential evolution for dynamic multiobjective optimization problems. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 1918–1925, May 2015.

[64] Xiaodong Li, J. Branke, and M. Kirley. On performance metrics and particle swarm methods for dynamic multiobjective optimization problems. In *2007 IEEE Congress on Evolutionary Computation*, pages 576–583, Sept 2007.

[65] Mario Cámara, Julio Ortega, and Francisco de Toro. *Approaching Dynamic Multi-Objective Optimization Problems by Using Parallel Evolutionary Algorithms*, pages 63–86. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[66] Z. Peng, J. Zheng, and J. Zou. A population diversity maintaining strategy based on dynamic environment evolutionary model for dynamic multiobjective optimization. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 274–281, July 2014.

[67] Victoria S Aragón, Susana Cecilia Esquivel, and Carlos Coello Coello. Evolutionary multiobjetive optimization in non-stationary environments. *Journal of Computer Science & Technology*, 5, 2005.

[68] A. Zhou, Y. Jin, and Q. Zhang. A population prediction strategy for evolutionary dynamic multiobjective optimization. *IEEE Transactions on Cybernetics*, 44(1):40–53, Jan 2014.

[69] Aimin Zhou, Yaochu Jin, Qingfu Zhang, Bernhard Sendhoff, and Edward Tsang. Prediction-based population re-initialization for evolutionary dynamic multi-objective optimization. In Shigeru Obayashi, Kalyanmoy Deb, Carlo Poloni, Tomoyuki Hiroyasu, and Tadahiko Murata, editors, *Evolutionary Multi-Criterion Optimization*, volume 4403 of *Lecture Notes in Computer Science*, pages 832–846. Springer Berlin Heidelberg, 2007.

[70] Y. Jin, C. Yang, J. Ding, and T. Chai. Reference point based prediction for evolutionary dynamic multiobjective optimization. In *2016 IEEE Congress on Evolutionary Computation (CEC)*, pages 3769–3776, July 2016.

[71] Kalyanmoy Deb, Udaya Bhaskara Rao N., and S. Karthik. *Dynamic Multi-objective Optimization and Decision-Making Using Modified NSGA-II: A Case Study on Hydro-thermal Power Scheduling*, pages 803–817. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[72] A. Isaacs, V. Puttige, T. Ray, W. Smith, and S. Anavatti. Development of a memetic algorithm for dynamic multi-objective optimization and its applications for online neural network modeling of uavs. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 548–554, June 2008.

[73] C. Xiao, D. Soetanto, K. Muttaqi, and M. Zhang. A dynamic evolutionary strategy for time ahead energy storage management in microgrid. In *2016 IEEE International Conference on Power System Technology (POWERCON)*, pages 1–6, Sept 2016.

[74] J. Tang, S. Alam, C. Lokan, and H. A. Abbass. A multi-objective evolutionary method for dynamic airspace re-sectorization using sectors clipping and similarities. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8, June 2012.

[75] H. Kaji, K. Ikeda, and H. Kita. Acceleration of parametric multi-objective optimization by an initialization technique for multi-objective evolutionary algorithms. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pages 2291–2297, June 2008.

[76] F. V. C. Martins, E. G. Carrano, E. F. Wanner, R. H. C. Takahashi, and G. R. Mateus. A dynamic multiobjective hybrid approach for designing wireless sensor networks. In *2009 IEEE Congress on Evolutionary Computation*, pages 1145–1152, May 2009.

[77] M. B. Abello, L. T. Bui, and Z. Michalewicz. An adaptive approach for solving dynamic scheduling with time-varying number of tasks - part II. In *2011 IEEE Congress of Evolutionary Computation (CEC)*, pages 1711–1718, June 2011.

[78] Anke K Hutzschenreuter, Peter A. N. Bosman, and Han La Poutrú. Evolutionary multiobjective optimization for dynamic hospital resource management. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 320–334. Springer, 2009.

[79] P. P. Y. Wu, D. Campbell, and T. Merz. Multi-objective four-dimensional vehicle motion planning in large dynamic environments. *IEEE Transactions*

on Systems, Man, and Cybernetics, Part B (Cybernetics), 41(3):621–634, June 2011.

[80] André R. da Cruz, Rodrigo T. N. Cardoso, and Ricardo H. C. Takahashi. *Multiobjective Dynamic Optimization of Vaccination Campaigns Using Convex Quadratic Approximation Local Search*, pages 404–417. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[81] M. A. Montes de Oca, C. Cotta, and F. Neri. Local Search. In *Handbook of Memetic Algorithms*, volume 379 of *Studies in Computational Intelligence*, pages 29–41. Springer Berlin, Heidelberg, 2012.

[82] P. Moscato. On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. Technical Report 826, California Institute of Technology, 1989.

[83] R. Dawkins. *The Selfish Gene*. Popular Science, 1989.

[84] P. Koch, O. Krämer, G. Rudolph, and N. Beume. On the Hybridization of SMS-EMOA and Local Search for Continuous Multiobjective Optimization. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, GECCO '09, pages 603–610, New York, NY, USA, 2009. ACM.

[85] A. Lara, G. Sánchez, C. A. C. Coello, and O. Schütze. HCS: A New Local Search Strategy for Memetic Multiobjective Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation*, 14(1):112–132, Feb 2010.

[86] N. Beume. S-metric calculation by considering dominated hypervolume as Klee's measure problem. *Evolutionary Computation*, 17(4):477–492, 2009.

[87] J. Knowles and D. Corne. Properties of an Adaptive Archiving Algorithm for Storing Nondominated Vectors. *IEEE Transactions on Evolutionary Computation*, 7(2):100–116, 2003.

[88] A. Auger, J. Bader, D. Brockhoff, and E. Zitzler. Theory of the Hypervolume Indicator: Optimal μ-distributions and the Choice of the Reference Point. In *Proceedings of the Tenth ACM SIGEVO Workshop on Foundations of Genetic Algorithms*, FOGA '09, pages 87–102, New York, NY, USA, 2009. ACM.

[89] V. A. Sosa Hernández, O. Schütze, G. Rudolph, and H. Trautmann. The Directed Search Method for Pareto Front Approximations with Maximum Dominated Hypervolume. In M. Emmerich et al., editor, *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation IV*, volume 227 of *Advances in Intelligent Systems and Computing*, pages 189–205. Springer International Publishing, 2013.

[90] A. Lara, C. A. Coello Coello, and O. Schütze. A Painless Gradient-assisted Multi-objective Memetic Mechanism for Solving Continuous Bi-objective Optimization Problems. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8. IEEE, IEEE Press, 2010.

[91] H. Ishibuchi, T. Yoshida, and T. Murata. Balance between Genetic Search and Local Search in Memetic Algorithms for Multiobjective Permutation Flowshop Scheduling. *IEEE Transactions on Evolutionary Computation*, 7(2):204–223, 2003.

[92] J. Knowles and D. Corne. Memetic Algorithms for Multiobjective Optimization: Issues, Methods and Prospects. In William E. Hart, N. Krasnogor, and J. E. Smith, editors, *Recent Advances in Memetic Algorithms*, pages 313–352. Springer. Studies in Fuzziness and Soft Computing, Vol. 166, 2005.

[93] P. Shukla. On Gradient Based Local Search Methods in Unconstrained Evolutionary Multi-objective Optimization. In S. Obayashi et al., editor, *EMO 2007*, pages 96–110, 2007.

[94] M. Vasile and F. Zuiani. Multi-agent collaborative search: An agent-based memetic multi-objective optimization algorithm applied to space trajectory design. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 225(11):1211–1227, 2011.

[95] P. A. N. Bosman. On Gradients and Hybrid Evolutionary Algorithms for Real-Valued multiobjective optimization. *IEEE Trans. Evolutionary Computation*, 16(1):51–69, 2012.

[96] F. Neri, C. Cotta, and P. Moscato. *Handbook of Memetic Algorithms*, volume 379. Springer, 2012.

[97] J. D. Knowles. *Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization*. PhD thesis, University of Reading, UK, 2002. (Section 4.3.4 "S metric Archiving").

[98] J. D. Knowles and D. Corne. Properties of an Adaptive Archiving Algorithm for Storing Nondominated Vectors. *IEEE Transactions on Evolutionary Computation*, 7(2):100–116, 2003.

[99] E. Zitzler and S. Künzli. Indicator-Based Selection in Multiobjective Search. In X. Yao et al., editors, *Proc. Int'l Conf. on Parallel problem Solving from Nature (PPSN VIII)*, pages 832–842, Berlin Heidelberg, 2004. Springer.

[100] J. Knowles and D. Corne. On metrics for comparing nondominated sets. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 1, pages 711–716. IEEE, 2002.

[101] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *Evolutionary Computation, IEEE Transactions on*, 7(2):117–132, 2003.

[102] K. Bringmann and T. Friedrich. Tight Bounds for the Approximation Ratio of the Hypervolume Indicator. In *Proceedings of the 11th International Conference Parallel Problem Solving From Nature (PPSN)*, volume 6238 of *Lecture Notes in Computer Science*, pages 607–616, Krakow, Poland, September 2010. Springer.

[103] K. Bringmann and T. Friedrich. The Maximum Hypervolume Set Yields Near-optimal Approximation. In *Proceedings of the ACM Genetic and Evolutionary Computation Conference (GECCO)*, pages 511–518. ACM Press, 2010.

[104] J. Fliege, L. M. Graña Drummond, and B. F. Svaiter. Newton's Method for Multiobjective Optimization. *SIAM J. on Optimization*, 20:602–626, 2009.

[105] P. K. Shukla. Gradient Based Stochastic Mutation Operators in Evolutionary Multi-objective Optimization. In *Adaptive and Natural Computing Algorithms: 8th International Conference, ICANNGA 2007, Warsaw, Poland, April 11-14, 2007, Proceedings, Part I*, pages 58–66, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[106] T. Bartz-Beielstein. *Experimental Research in Evolutionary Computation: The New Experimentalism (Natural Computing Series).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[107] B. Naujoks, D. Quagliarella, and T. Bartz-Beielstein. Sequential parameter optimisation of evolutionary algorithms for airfoil design. Technical report, University of Las Palmas de Gran Canaria, 2006.

[108] S. Wessing and B. Naujoks. Sequential parameter optimization for multi-objective problems. In *IEEE Congress on Evolutionary Computation*, pages 1–8, July 2010.

[109] C. Igel, N. Hansen, and S. Roth. Covariance Matrix Adaptation for Multi-objective Optimization. *Evolutionary Computation*, 15(1):1–28, 2007.

[110] J. Bader and E. Zitzler. HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization. *Evolutionary Computation*, 19(1):45–76, 2011.

[111] K. Deb, S. Lele, and R. Datta. A hybrid evolutionary multi-objective and SQP based procedure for constrained optimization. In *International Symposium on Intelligence Computation and Applications*, pages 36–45. Springer, 2007.

[112] M. Emmerich and A. Deutz. A Family of Test Problems with Pareto-fronts of Variable Curvature Based on Super-spheres. In *MCDM 2006*, Chania, Greece, June 19-23 2006.

[113] K. Witting. *Numerical Algorithms for the Treatment of Parametric Optimization Problems and Applications.* PhD thesis, University of Paderborn, 2012.

[114] E. Zitzler and S. Künzli. Indicator-based selection in multiobjective search. In *Parallel Problem Solving from Nature-PPSN VIII*, pages 832–842. Springer, 2004.

[115] Juan J Durillo, Antonio J Nebro, and Enrique Alba. The jMetal framework for multi-objective optimization: Design and architecture. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8. IEEE, 2010.

[116] Oliver Schütze, Marco Laumanns, Emilia Tantar, Carlos Coello, and E. Talbi. Computing gap free Pareto front approximations with stochastic search algorithms. *Evolutionary Computation*, 18(1):65–96, 2010.

[117] Saúl Zapotecas Martínez, Víctor A. Sosa Hernández, Hernán Aguirre, Kiyoshi Tanaka, and Carlos A. Coello Coello. *Using a Family of Curves to Approximate the Pareto Front of a Multi-Objective Optimization Problem*, pages 682–691. Springer International Publishing, Cham, 2014.

[118] Kai-Tai Fang, Dennis K. J. Lin, Peter Winker, and Yong Zhang. Uniform design: theory and application. *Technometrics*, 42(3):237–248, 2000.

[119] Marco Farina, Kalyanmoy Deb, and Paolo Amato. Dynamic Multiobjective Optimization Problems: Test Cases, Approximations, and Applications. *IEEE Transactions on Evolutionary Computation*, 8(5):425–442, 2004.