Universiteit Leiden
Opleiding Informatica

Designing Ships using

Constrained Multi-Objective Efficient Global Optimization

Name:	Roy de Winter
Date:	31/05/2018
1st supervisor:	Thomas Bäck
2nd supervisor:	Kaifeng Yang

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

A modern ship design process is subject to a wide variety of constraints such as safety constraints, regulations, and physical constraints. Traditionally, ship designs are optimized in an iterative design process. However, this approach is very time consuming and is likely to get stuck in local optima. Not only does this optimization problem have complex constraints, it also consists of multiple objectives like resistance, stability, and cost.

This constraint multi-objective optimization problem can be dealt with much more efficiently than through the traditional approach. In this thesis, a global optimization algorithm is proposed that explores the design space with the help of integrated software tools that are capable of simultaneous evaluation of the ship objectives and constraints. The optimization algorithm proposed uses the $\mathcal{S}$-Metric-Selection-based Efficient Global Optimization (SMS-EGO) in combination with constraint handling techniques from an algorithm called Self-Adjusting Constrained Optimization by Radial Basis Function Approximation (SACOBRA). Since the evaluation of these ship designs is expensive in terms of computational effort, it is crucial for the algorithm to find feasible near-optimal solutions in as few evaluations as possible.

In this thesis, it is shown that the proposed Constrained Efficient Global Optimization (CEGO) algorithm can significantly improve ship designs by automatic optimization using a small evaluation budget.

Acknowledgements

I would like to thank Bas van Stein, Thomas Bäck, Matthys Dijkman and Thijs Muller for supervising me during this research. Furthermore, I would like to thank C-Job Naval Architects for the pleasant internship and for giving me the opportunity to do research on all kinds of optimization techniques at their Research and Development department.

Contents

–A ship in the harbor is safe, but that is not what ships are built for.

John A. Shedd

1

Introduction

The European Commission (EC) defined a goal to achieve a 60% reduction of greenhouse gas (GHG) emission by 2050 in the transport sector [15]. To achieve this goal, every transport sector will have to make smarter and better decisions. The International Maritime Organization (IMO) responsible for regulating shipping independently announced that by 2025 all new ships will have to be 30% more energy efficient than those built in 2014 [35]. Furthermore in a recent press briefing the IMO announced that the shipping industry should reduce the total annual GHG emission by 50% by 2050 compared to 2008 [18]. To achieve the goals set by the EC and the IMO, the new ships that are currently being engineered will have to be optimized for minimum environmental impact. Of course, the environmental impact is not the only objective to consider while optimizing a ship. The ship owners also want their ship to be operationally efficient and to have the lowest operational expenses possible. Additionally, safety and comfort of crew and/or passengers should meet the criteria given by the regulating authorities.

To achieve an optimal solution where all stakeholders are satisfied, typically different experts work together to optimize the ship. These experts traditionally optimize the ship using heuristic methods learned over the course of years, derived from knowledge gained through a process of trial and error. In this optimization process, naval architects traditionally use the design spiral [13] (Figure 1.0.1). As prescribed in the theory about the design spiral, the components of a ship are optimized step by step, iteratively.

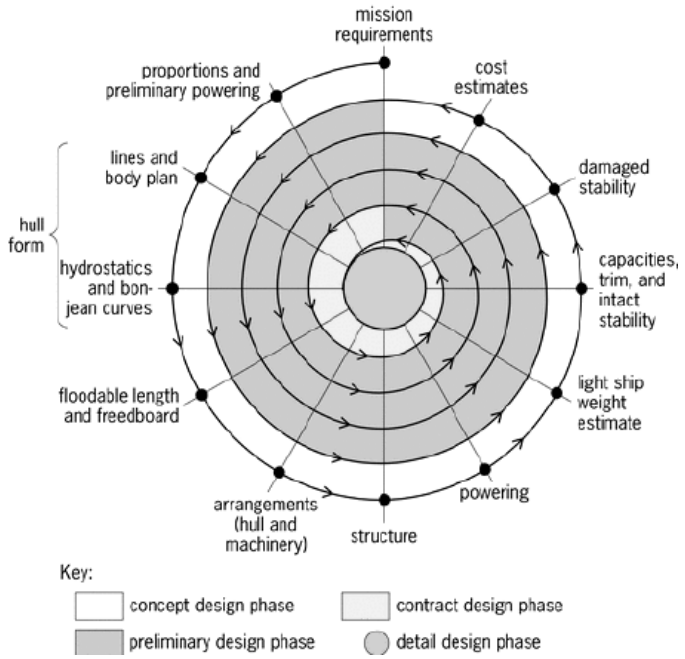

Figure 1.0.1: Classical ship design spiral.

This traditional, expert driven, iterative approach used to design a ship can cause the design process to get stuck in local optima. This is the case because, for a single naval architect or a group of experts, it is impossible to consider the whole design space and all the relationships and dependencies between the variables, constraints and objectives [26].

To make better design decisions in the future, the ship optimization processes such as proposed by Papanikolau [26] could be used. This integrated design approach brings together all key aspects of a design task at the same time.

1.1 Research Question

In this thesis, it is shown that an integrated design approach in combination with the proposed optimization algorithm results in significantly improved ship design solutions. This is shown by giving answer to the following research question and sub-questions:

How can state of the art optimization algorithms efficiently support multidisciplinary ship design applications?

The answer to this question is not a simple one and starts with a better understanding of state of the art optimization algorithms, multidisciplinary ship design applications, and efficient multi-objective problem solving.

Currently there already exist several algorithms that show great results on artificially designed constrained multi-objective optimization problems. To find out whether they are suitable for solving ship design applications the following sub-question is defined: **Which already existing state of the art optimization algorithms and theories could be used to solve multidisciplinary ship design optimization applications?**

The algorithm should be able to optimize multidisciplinary ship design optimization applications. Here, the term *multidisciplinary* refers to different ship design problems. Some optimization problems will have for example six decision variables, four objectives and fifteen inequality constraints, while other optimization problems might have equality constraints or only two objectives. To see how the developed algorithm deals with the multiple constraints and objectives, the following sub-question is defined: **How flexible are the state of the art optimization algorithms in terms of dealing with multiple constraints and multiple objectives?**

Furthermore, the algorithm has to be efficient, because optimizing a ship design can be very time consuming. Consider for example the case where we want to minimize the resistance. To minimize the resistance of a ship design, multiple Computational Fluid Dynamics (CFD) simulations [36] will have to be executed for different design variations. Only one CFD of one design variation can take up to several hours of computation time. Therefor, the next sub-question to be answered is: **In terms of function evaluations and solution quality, how efficient is the proposed optimization algorithm compared to existing approaches?**

1.2 Outline

The next chapters describe the research that has been conducted to answer the questions. In Chapter 2 the related research and relevant algorithms are discussed. The formulation of the problem statement is given in Chapter 3. In Chapter 4 the proposed algorithm is discussed in detail. Next, it is shown empirically in Chapter 5 that the proposed algorithm is efficient and is able to find a good approximation of the Pareto frontier using a limited evaluation budget. Finally, the results are discussed and conclusions are drawn in Chapter 6.

–If I have seen further it is by standing on the shoulders of Giants.

Isaac Newton

2

Earlier Research

Already quite some work has been done in the domains of constraint problem solving, multi-objective problem solving, and ship design optimization. Therefore, in this chapter the most relevant approaches and algorithms to our problem are discussed. This is organized by first discussing the constraint handling, then multi-objective optimization, and finally some earlier research in ship design optimization is addressed.

2.1 Constraint Handling

In the literature, several approaches for constraint handling in evolutionary algorithms have been presented so far [1, 4, 5, 7, 8]. The most prominent constraint handling techniques are the following:

1. Add a penalty to the objective functions for infeasible solutions [7],
2. Preferring feasible solutions above infeasible solutions in selection [8],
3. Multi-objective optimization with additional objectives for the constraint functions [5],
4. Repair algorithms that repair proposed infeasible solutions during the search [4],
5. Machine learning/surrogate models to model the constraint functions [1].

2.1.1 Penalty Functions

Adding a penalty to the objective function for infeasible solutions is a frequently used approach to handle constraints [7]. The idea behind this procedure is that infeasible solutions get a worse objective function value compared to the feasible solutions. This way the constrained optimization problem becomes an unconstrained one by adding the constraint violations to the objective function as a penalty term.

The main drawback of this approach is that this penalty function should include every constraint. Together with the penalty function the right balance between the fitness function and penalty function should be found. To find this right balance often new balance parameters need to be optimized. In order to find this right balance, typically a lot of trail and error is needed for effectively minimizing the penalty term to find good, feasible solutions.

2.1.2 Feasible Solution Preference

Feasible solution preference methods always prefer feasible solutions to infeasible solutions. This typically causes that during search, too little information from infeasible solutions is used. Deb [8] improved this method by introducing a diversity mechanism where an infeasible solution is sometimes ranked according to its fitness value among the feasible ones. This often shows good results but typically a large number of function evaluations are needed to find a feasible solution and even more to find a good feasible solution.

2.1.3 Objectives for Constraints

The general idea of optimizing an objective function that is subject to a number of constraints is: deal with constraints as hard objectives and deal with objectives as soft objectives. The hard objectives imply the following: if the constraints are not satisfied, the solution is not feasible and the objective value is irrelevant. One way to make sure that the constraints become satisfied in the optimization process is by treating the constraints as objectives that need to be optimized as well [5]. The objective and constraints (represented by additional objective functions) become equally important this way. The chosen optimization algorithm would now optimize the constraints and objectives simultaneously. A disadvantage of this approach is that the optimization focus lies to much on the constraints, and the problem gets much more complex and thereby may slow down the efficiency of the algorithm significantly.

2.1.4 Repair Infeasible Solutions

Another approach to deal with constraints is repairing infeasible solutions [4]. For some known linear boundaries of the decision variables this can be very

easy by simply truncating the decision space, but for non-linear real-valued functions of the decision variables it can be very hard. Therefore, it typically takes a large number of function evaluations before an infeasible solution is transformed into a feasible one.

2.1.5 Model Assisted Optimization

In the last decade, different models have been used to model the constraint functions: Poloczek and Kramer for example used Support Vector Machines as a classifier to predict the feasibility of solutions [29]. Powell invented a direct search method, which models the objective and constraints using linear approximations, better known as COBYLA [31], and recently Bagheri et al. improved the Constrained Optimization by Radial Basis function Approximation (COBRA) of Regis [32], by making the parameters of COBRA Self Adjusting (SACOBRA) [1]. Especialy the efficient SACOBRA solver that makes use of the Radial Basis Function (RBF) interpolation to model the constraints is very interesting since it is able to find high-quality results using only few function evaluations, is precise, is fast and does not need additional parameters. The only drawback known from SACOBRA and model assisted optimization is that it is not very suitable for solving highly multimodal functions in few function evaluations.

2.2 Multi-Objective Optimization

Simultaneously, optimizing objectives becomes extra challenging when the objectives are conflicting. To deal with this problem the classical approach would simply be: combine the objectives into one function which typically gives rise to one optimal point. However, we want the whole Pareto optimal set and not just one solution. A variety of approaches for finding this set, also called the Pareto frontier, has been proposed by different researchers, and is a very active field of research (e.g. see [6, 10, 40]). Fully automated methods which do not need any interference of a human decision maker during the optimization process, can generally be divided into three categories:

1. Enumerative
2. Deterministic
3. Stochastic

2.2.1 Enumerative Methods

The enumerative method is the most simple one [16]. This method simply evaluates every possible solution in a finite design space. This method is not very efficient since the number of possible solutions in a design space can be very large. On the other hand, if capacity and time is available to enumerate

the whole design space, this is the best solution since it can be guaranteed that all the optimal solutions will be found.

2.2.2 Deterministic Methods

Deterministic methods are similar to the enumerative methods but they try to include domain knowledge in such a manner that not the whole design space needs to be evaluated [9]. Examples of a few deterministic strategies are: greedy, hill-climbing, branch & bound, and depth and breadth-first search. The problem remains that the search space is in most cases still to big to evaluate all the solutions proposed by the deterministic methods.

2.2.3 Stochastic Methods

Stochastic optimization techniques are used as an alternative for enumerative and deterministic strategies and are able to deal with complex search spaces and multiple objectives. Stochastic search techniques typically have some randomness involved which causes that the obtained solutions are typically different for every run of a stochastic optimization technique. This also means that most of the stochastic algorithms cannot guarantee to find the global optimal solution. Examples of stochastic search strategies and algorithms are: Non-dominated Sorting Genetic Algorithm version II [11], Stength Pareto Evolutionary Algorithm version 2 [21], $\mathcal{S}$-Metric Selection Multi-Objective Optimization [30], and the Multi-Objective Genetic Algorithm [41].

Non-dominated Sorting Genetic Algorithm, version II

NSGA-II [11] is a classic multi-objective optimization algorithm. NSGA-II uses a non-dominated sorting-based selection operator. This operator creates a mating pool by combining the parent and child population to select the best N solutions for the next generation. This selection operator makes sure that the mating pool is well spread, the solutions in the pool have a high fitness, and the emphasis lies on the non-dominated solutions. Finally crossover is used to generate the new individuals from the selected parents.

Strength Pareto Evolutionary Algorithm 2

SPEA2 [21] is the second version of an evolutionary algorithm that optimizes the Pareto frontier. This algorithm uses a fine-grained fitness assignment strategy that is based on how many individuals each individual dominates and is dominated by. Furthermore, a nearest neighbor density estimation technique is incorporated which takes care of a more precise guidance of the search process. The algorithm also makes sure that the boundaries are guaranteed by truncation of the solutions that fall outside of the boundary.

$\mathcal{S}$-Metric Selection Multi-Objective Optimization

SMS-EGO [30] is an efficient multi-objective optimization algorithm that uses a Design and Analysis of Computer Experiments (DACE) to train Kriging [20] surrogate models in order to efficiently optimize the objective functions. This model assisted optimization technique utilizes the assumption that close solutions are more likely to have similar objective values. Furthermore, SMS-EGO uses the $\mathcal{S}$-metric or (hyper)volume contribution [2] to optimize the (hyper)volume between the current Pareto frontier and a reference point. This optimization algorithm, however, does not offer a constraint handling technique and is again not very suitible for solving highly multimodal functions.

Multi-Objective Genetic Algorithm

MOGA is based on the first version of the SPEA algorithm [41], where the fitness value is again based on the number of dominated individuals. The selection of the parents is done by tournament selection and the children are generated by single-point crossover. Furthermore, the children have a chance to get mutated by the so called creep mutation operator. This algorithm is currently used in the widely used ship design software NAPA[1].

2.3 Ship Design Applications

As mentioned before in the introduction (Chapter 1), ships are traditionally optimized step by step using the design spiral (Figure 1.0.1). Therefore, most earlier research to ship designs focuses on only one part of the ship. Consider for example the research on hull form optimization for reducing the resistance [3, 37], or the research on scantling optimization of the midship section [33], or the research on the dredging operations [39]. All of those typically focus on only one part of the ship. A combination of different scientific studies is made by Papanikolaou in [26]. In this paper all components of a ship are put together and optimized simultaneously.

A practical implementation of Papanikolaou's method is developed by C-Job Naval Architects[2] under the name of Accelerated Concept Design (ACD) and can be found in Figure 2.3.1. The figure contains eight crucial components that need to be optimized and or should be designed in such a manner that they meet criteria given by the stakeholders.

[1] NAPA Oy, Release 2017.3-3 (2018), NAPA software, `http://www.NAPA.fi/`.
[2] C-Job Naval Architects, (2018), `https://c-job.com/`.

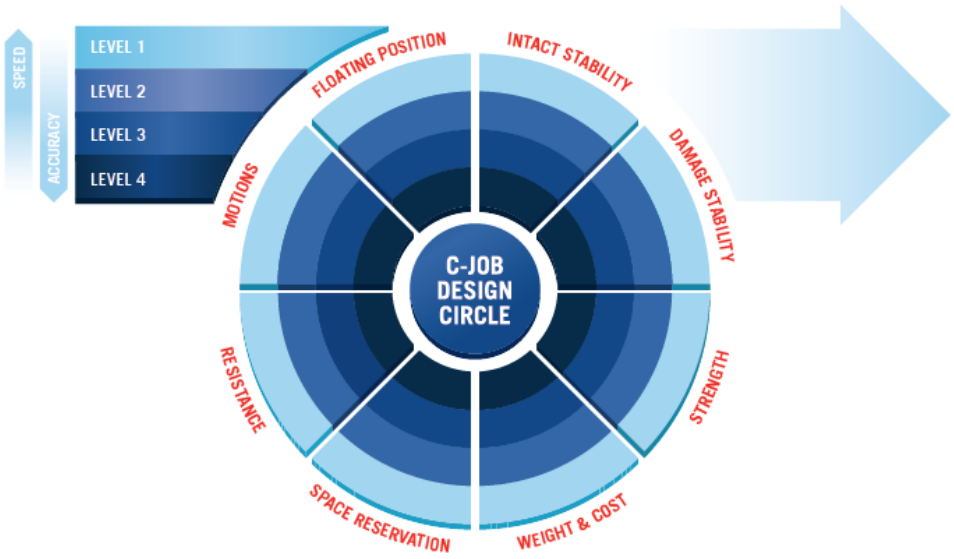

Figure 2.3.1: C-Job Naval Architects Design Circle. Four different levels of accuracy, with eight different design aspects to consider in the optimization process.

2.3.1 Accelerated Concept Design

The ACD tool consists of several software components that are capable of simultaneous evaluation of the ship objectives and constraints of a design variation in an automated manner. This way, different design variations can be computed sequentially without the interference of an engineer. The objectives and constraints can be computed using different levels of precision. The level of precision is typically set according to a client needs. The higher the accuracy level, the more calculation time will be required to compute the objective and or constraint. In the experiments conducted in this thesis, the level of accuracy is set to level 3 except for weight and cost. The accuracy of weight and cost is set to level 2.

A general overview of the objectives and constraints of a typical ship can be found in the following descriptions:

Floating Position The floating position of a ship is checked to see if the ship is in constant balance in still water and if the ship does not heel or trim too much in a stable state. The maximum values for these constraints are set by regulation authorities. Together with the floating position, the maximum draught is checked to see if it meets the stakeholders demands. Typically the maximum draught is set to a number of meters so it can still moor in a port or sail through a sea lock, river or canal.

Intact Stability When a ship is in normal operation configuration the ship should meet various stability criteria. The ship, for instance, should

remain stable under all, even extreme loading conditions. Additionally, the ship should also be capable of sailing with heavy weather conditions without capsizing. Of course a transatlantic passenger ship has more strict regulations compared to a simple bulk carrier that only sails inland. Therefore, different regulations are defined per ship type.

Damage Stability After a collision, a ship should not immediately sink or capsize. It should be capable of handling some damage to some selected spaces/rooms of the ship. The guidelines for the locations of the damage is defined by the IMO and other regulating authorities. Of course the regulations differ again per type of ship.

Strength Typically when cargo gets loaded to a ship, the ship bends a bit in the middle. This is the case because when a ship is fully loaded, the water gives an upward pressure to the ship while the gravity on the cargo now pushes the ship downwards in the middle, this phenomenon is better known as sagging ((1) in Figure 2.3.2). The inverse of sagging is named hogging ((2) in Figure 2.3.2) and mainly occurs when the ship is empty. How much the ship is allowed to sag and hog is again defined by the regulating authorities. The ship should be made strong enough so it does not exceed the maximum stress limit.

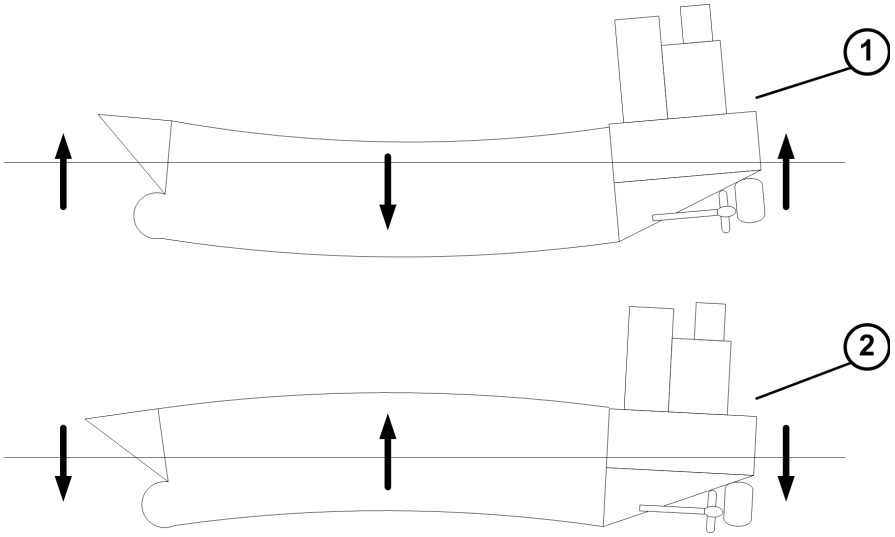

Figure 2.3.2: Ship hull is sagging in (1) and hogging in (2). Note that the bending is exaggerated.

Weight & Cost For a ship yard the weight goes hand in hand with the cost of a ship. This is the case because the required amount of of steel that needs to be bought and put together gives a rough indication of how much the ship will cost to build. Therefore, the weight of a ship

is typically an objective that is to be minimized, because ship owners want their ship to be cheap to build.

Space Reservation A space reservation should be made in such a manner that the fuel tank, engine, pump, crew accommodation, steering hut, and other equipment all fit in the hull or on deck. This space reservation typically results in a lot of constraints and often changes a bit when more details of the ship gets defined. A schematic space reservation can be found in Figure 2.3.3.

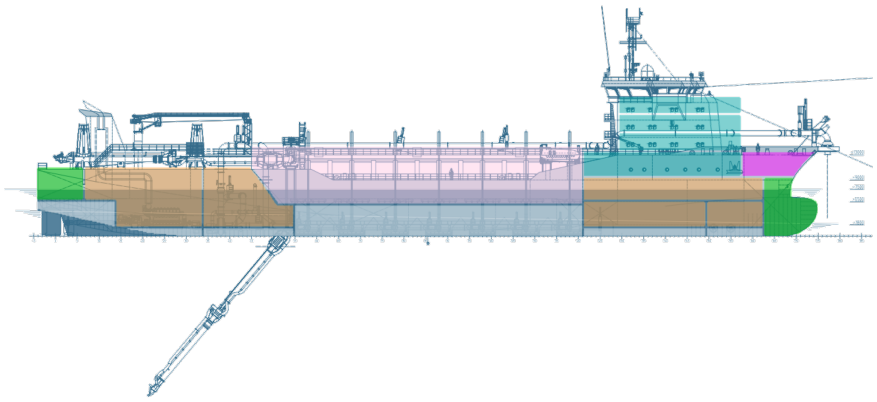

Figure 2.3.3: Space reservation of a ship. Where every color defines a different space.

Resistance Resistance is another objective that typically needs to be minimized for a ship to be as operationally efficient as possible. This can be achieved by changing the shape of the hull. Typically a long and slender ship will lead to less hull resistance compared to a shorter wider ship. Resistance is usually estimated using CFD simulations which can be very expensive in terms of computation time.

Motions Motion deals with how the ship reacts to waves and wind. It should not roll, pitch, yaw, heave, sway or surge (see Figure 2.3.4) too much after a single or a sequential number of waves. Passengers, for example, get seasick when there is too much heave/pitch and equipment can break when there is too much rolling back and forth after a number of sequential waves.

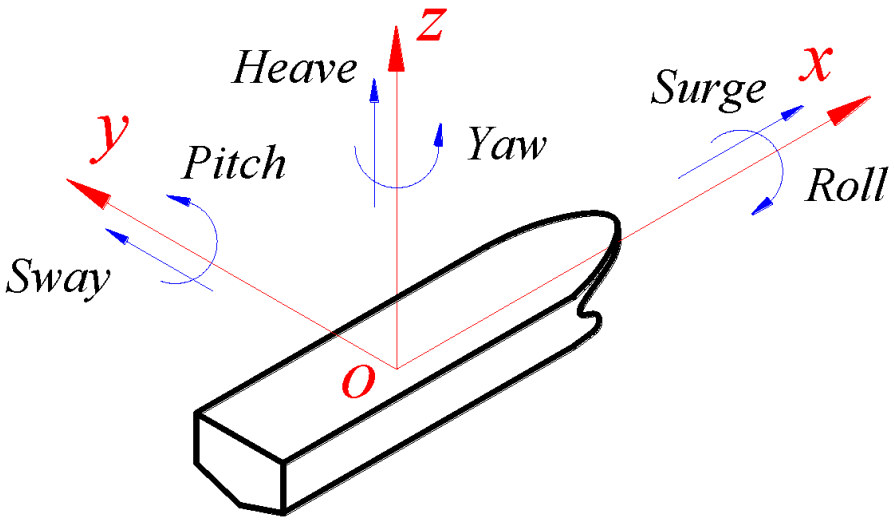

Figure 2.3.4: Roll, pitch, yaw, heave, sway, surge of a ship.

–The biggest room in the world is room for improvement.

Helmut Schmidt

3

Ship Design Optimization Problem

Ship design optimization is a complex process requiring the successful coordination many disciplines [27]. This results from the fact that a ship consist of a wide variety of systems and components, for example cargo storage and handling, energy generation and ship propulsion, accommodation of crew/passengers and ship navigation. All these components need to be optimized in such a manner that all the design stakeholders are satisfied, and the right balance is found between the conflicting objectives. Before we can optimize all components, we first have to describe the problem in more detail. In this chapter, the ship design stages and the optimization problem are defined. For illustration purposes a case study of a ship design optimization problem is presented.

3.1 Design Stages

Every ship starts with an initial idea from a client. After this idea has been described in a design specification, the ship design process can begin. The ship design process can roughly be divided into four design stages: Concept-, basic-, functional- and detail design. In these four stages an idea of the client is transformed into a realistic and complete design a shipyard can build. The different stages are further described in the following subsections and are visually presented in Figure 3.1.1.

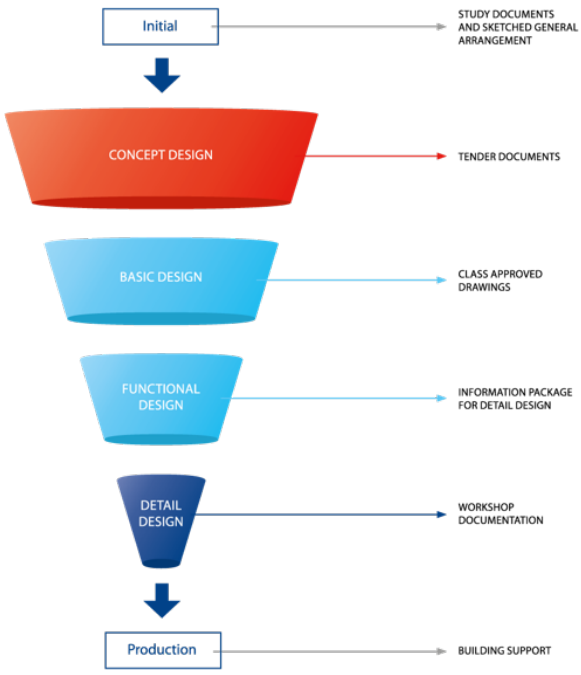

Figure 3.1.1: Four ship design stages: Concept-, basic-, functional- and detail design.

3.1.1 Initial Idea

In the initial idea, the future ship owner defines the purpose and the physical constraints of the ship. Different purposes of ships could for example be: dredging, fossil fuel drilling, urban transport, cargo transport or sailing for pleasure. Besides the purpose, the initial idea also contains the information about whether the ship should be sea worthy or that it should only be capable of sailing inland.

Depending on the purpose of the ship, different regulations are defined by the regulating authorities. These regulation constraints are from now on referenced to as domain constraints. Besides these regulations, a typical ship also has some preference or physical constraints like: The ship should fit trough the Suez Canal, the ship should be able to moor in the harbour of Rotterdam, or the ship should be able to sail underneath the Tower Bridge. These rough constraints typically can be transformed into relatively simple box constraints for the design, such as a maximum/minimum draught, air draught, width, length and deck hight.

Finally the future ship owner will probably also have some preferences such as CO2 neutrality, a small building, and minimum operational costs which are to be translated into either constraints or objectives.

3.1.2 Concept Design

The concept design phase is the crucial part of the ship design process. In the concept design phase the naval architects translate the initial idea into the concept design of the ship. In the concept design, the following components get defined, computed, and parameterized: the general arrangement, first estimations regarding stability, strength, and the main cross section.

These components will define the ship's future performance, safety and cost. In this stage of the design process, all different components need to be optimized and designed in such a manner that they meet all regulations and safety criteria. This is not trivial for the following three reasons: 1) The objectives are typically conflicting. 2) Computing the constraints and objectives is very time consuming due to the required simulation time. 3) Only little parallelism is possible due to a typically limited number of commercial licences available to the ship design company. A more mathematical representation of the optimization problem can be found in the problem in description Section 3.2.

After the concept design phase, a ship yard can make an estimation of how long it will take, and how expensive it will be to build the ship. A benefit of the concept design phase of C-Job Naval Architects is that the owner of the concept design can go to different ship yards to get the best offer.

3.1.3 Basic Design

This phase yields two things: The drawings of the design and a class approval which proves that the ship meets the latest regulations and criteria.

In the basic design stage only a few last minute, very small design changes take place. These small design changes lead to a ship design that meets all regulations and safety criteria. To meet the regulations and safety criteria, the strength, damage stability, and intact stability should for example be within regulated bounds.

3.1.4 Functional Design

This phase is a phase in between the detail and the basic design. Some shipyards in Asia and Southern Europe do not have enough capacity to go from basic to detail design in one step. Therefore, some guidance to the shipyard can be provided in the shape of a functional design phase. This phase yields a information package to support the next phase.

3.1.5 Detail Design

In this phase, multiple block sections of the ship are created and drawn. These workshop drawings are created so that the shipyard can follow the instructions to create and put together all the parts of the final ship.

3.2 Problem Description

Every ship design optimization problem depends on the decision variables, constraints and the objectives of the problem. The constraints can often be seen as hard objective functions, which need to be satisfied before the minimization of the remaining, soft objective functions takes place [14]. A general definition of a Multi-Objective Problem is given in Definition 1, the definition of Pareto optimality is given in Definition 2, the definition of Pareto dominance is given in Definition 3, and Pareto optimal sets are defined in Definition 4, all from [6].

Definition 1.
(**Multi-Objective Problem** [6]) : A general **Multi-Objective Problem** is defined as minimizing $F(\vec{x}) = (f_1(\vec{x}), \ldots, f_k(\vec{x}))$ subject to $g_i(\vec{x}) \leq 0, i = \{1, \ldots, m\}$ and $\vec{x} \in \Omega$. **Multi-Objective problem** solutions minimize the components of the vector $F(\vec{x})$ to find a set of Pareto optimal solutions, where $\vec{x}$ is a n-dimensional decision variable vector $\vec{x} = (x_1, \ldots, x_n)$ from some search space Ω. It is noted that $g_i(\vec{x}) \leq 0$ represent constraints that must be fulfilled while minimizing $F(\vec{x})$ and Ω contains all possible $\vec{x}$ that can be used to satisfy an evaluation of $F(\vec{x})$.

Definition 2.
(**Pareto Optimality** [6]) : A feasible solution $\vec{x} \in \Omega$ is said to be Pareto Optimal with respect to Ω if and only if there is no $\vec{x_2}$ for which $\vec{v} = F(\vec{x_2}) = [f_1(\vec{x_2}), \ldots, f_k(\vec{x_2})]$ dominates $\vec{u} = F(\vec{x}) = [f_1(\vec{x}), \ldots, f_k(\vec{x})]$.

Definition 3.
(**Pareto Dominance** [6]) : A vector $\vec{u} = [u_1, \ldots, u_k]$ is said to **dominate** another vector $\vec{v} = [v_1, \ldots, v_k]$ if and only if $\vec{u}$ is partially less than $\vec{v}$, i.e., $\forall i \in \{1, \ldots, k\}, u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i$. The following symbol is used to indicate Pareto Dominance: $\preceq$.

Definition 4.
(**Pareto Optimal Set** [6]) : For a given Multi-Objective Problem, $F(\vec{x})$, the **Pareto Optimal Set**, $\mathcal{P}^*$, is defined as:

$$\mathcal{P}^* := \{\vec{x} \in \Omega \mid \nexists \vec{y} \in \Omega, F(\vec{y}) \preceq F(\vec{x})\} \tag{3.2.1}$$

To get some more insight in the symbols, variables, constraints, objectives, a real world application, a dredger from C-Job Naval Architects, the largest independent ship design and engineering company in the Netherlands, is given in detail in the following subsections.

3.2.1 Decision Variables

The decision variables of a ship design problem are the numerical quantities for which values can be varied in the optimization process [6]. These quantities are denoted as x_j, where $j = 1, \ldots, n$, where x_j represents one descision variable. The vector $\vec{x}$ is then represented by:

$$\vec{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \tag{3.2.2}$$

All the possible combinations in between a predefined lower and upper bound of $\vec{x}$ together is called the search space denoted by Ω. Of course due to the constraints, not every combination will lead to a feasible solution.

The dredger (Figure 3.2.1) has the following decision variables: $\Delta_{breadth}$, Δ_{length}, foreship length, hopper length extension, hopper breadth, hopper height. Here Δ means a change opposed to the original design.

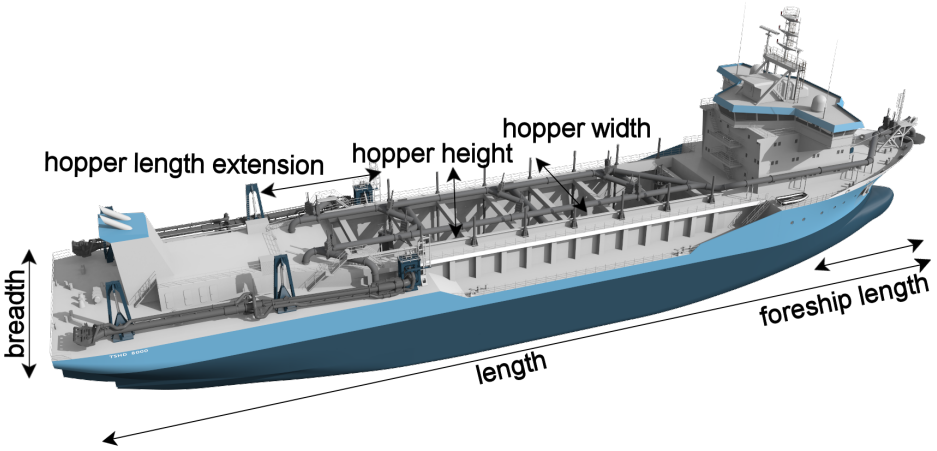

Figure 3.2.1: Trailer Suction Hopper Dredger designed by C-Job Naval Architects, with the design variables annotated.

The overall length and breadth of the hull can be transformed with the help of Free Form Deformation (FFD) [34]. For this transformation a box is drawn around the hull. Any point on the box can be moved in all directions and the parent surface that is inside this box will be transformed accordingly. This FFD can be achieved by changing the $\Delta_{breadth}$ and Δ_{length} parameter which then applies the FFD on the original concept design.

The part of the ship from the most forward bulkhead to the front is called the foreship. The location of this last bulkhead can be changed by varying the *foreship length* decision variable.

The cargo space, where the dredged material is dumped in, is called the hopper. Changes can be made to the *height*, the *breadth*, and to the *length*

extension of the hopper.

The optimization problem therefore has in total 6 design variables which is hard to visualize. To still be able to visualize the 6 dimensions, a parallel coordinate plot can be made. An example of such a plot is shown in Figure 3.2.2.

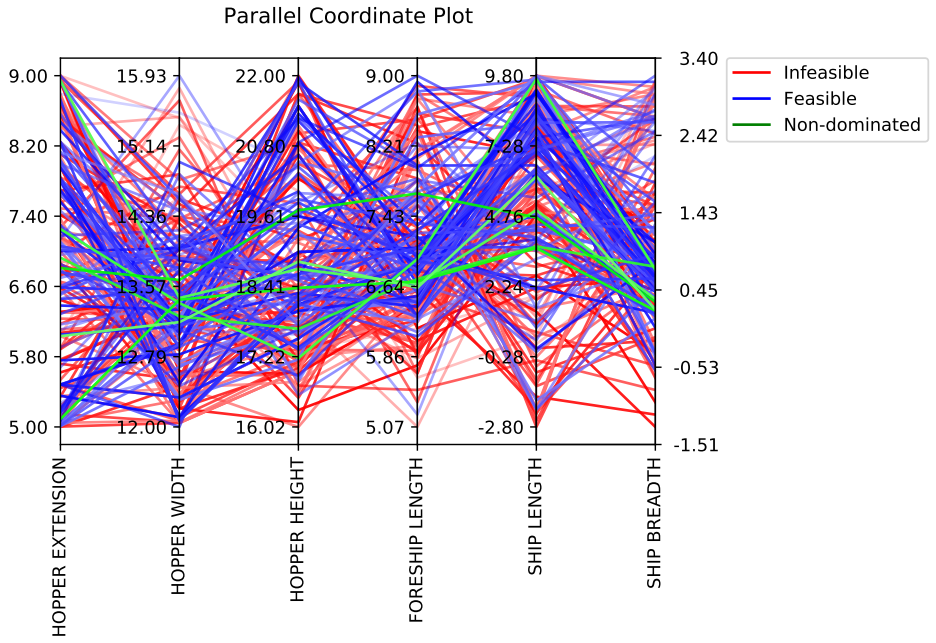

Figure 3.2.2: Parallel coordinate plot with infeasible solutions (red), feasible solutions (blue), Non-dominanated solutions (green), every vertical axes represents a decision variable.

3.2.2 Constraints

The constraints can be expressed in terms of function inequalities, where one function inequality represents one of the m constraints. All inequality constraints can then be expressed the following way:

$$g_i(\vec{x}) \leq c_i \quad \vee \quad g_i(\vec{x}) \geq c_i \quad \forall\, i \in \{1, \ldots, m_1\} \tag{3.2.3}$$

Where g_i is a, generally non-linear, real-valued function of the decision variables $\vec{x}$ and c_i is a constant value. Some special cases of constraints are equality constraints. These equality constraints are transformed into two approximate inequality constraints:

$$g_i(\vec{x}) \leq c_i + \epsilon \quad \wedge \quad g_i(\vec{x}) \geq c_i - \epsilon \quad \forall\, i \in \{1, \ldots, m_2\} \tag{3.2.4}$$

Where $\epsilon = 0.0001$ and therefore so small that in practice it can be neglected. Finally, all function inequalities are transformed without loss of generality in such a manner that we always get the function inequalities that represents all the constraints as presented in Equation 3.2.5.

$$g_j(\vec{x}) \leq 0 \quad \forall\, j \in \{1, \ldots, m\} \tag{3.2.5}$$

Note that it is assumed that there is at least one solution for $\vec{x} \in \Omega$ that satisfies all the constraints. If there is no such solution the optimization process is doomed to fail.

The constraints of a ship design optimization problem can be divided into two categories: domain constraints and practical constraints. Domain constraints are the constraints given by the regulating authorities while the practical constraints are typically physical constraints given by the client or imposed by physics. In the dredger case, the constraints mainly make sure that everything fits in the hull and that the safety constraints are taken into account.

For the practical constraints, the space reservation for: payload, fuel tank, engine, pump, and accommodation are checked to see if the design variation at least meets the minimum space required.

The domain constraints: steel arrangement, hull formation, double bottom check, location of foremost bulkhead, intact stability, draft when fully loaded, trim, and heel are checked to see if the ship meets the recommended stability criteria, and to see if it at least meets the prescribed rules from the regulating authorities.

In total, after all constraints are transformed, the dredger case has sixteen constraints, which are computed by subtracting the required minimum value from the actual value of the design variation. When all values are negative the ship is feasible.

3.2.3 Objectives

The objectives of a ship design optimization problem are typically conflicting and non-commensurable. As a consequence there is usually not one unique, perfect solution but a set of alternative, so called non-dominated solutions. This non-dominated solution set contains good compromises between the objective functions $f_j(\vec{x}), j = 1, \ldots, k$. Together they form the vector of functions to be minimized:

$$F(\vec{x}) = [f_1(\vec{x}), \ldots, f_k(\vec{x})] \tag{3.2.6}$$

The non-dominated set of solutions together form the Pareto frontier, where Pareto optimality is defined by Definition 2 and Pareto dominance is defined by Definition 3.

Note that all the objective functions are transformed to minimization problems. A transformation of a maximization problem into a minimization problem can easily be achieved by simply multiplying the objective function by minus one.

The dredger case has two objectives: maximizing the performance and minimizing the building costs. This can be achieved by minimizing the hull resistance and steel weight. This sounds trivial, but the objectives are a classical example of conflicting ones. A long and slender ship will lead to less hull resistance and a higher steel weight, while a wide shorter ship will have a higher hull resistance and a lower steel weight. These design variations of the ship, if non-dominated by other solutions, are then design variations in the Pareto Optimal set as described in Definition 4.

The resistance of the design variation can be estimated with a CFD simulation. There are different types of CFD simulation methods. In the concept phase of the dredger, a relatively simple potential flow solver [36] is used. This approach does not take everything into account but it is very suitable for comparing the resistance between different design variations.

An indication of the steel weight in the concept phase is calculated by first creating the main frame scantlings for either port or starbord and then mirroring it (example in Figure 3.2.3). The main frame is made strong enough such that it does not exceed the maximum stress limit by changing the scantlings (mainly plate thickness and profile heights). This way the maximum bending moments can never be exceeded. The surface of the scantlings multiplied by the length of the ship can then be used as an estimation of the steel weight.

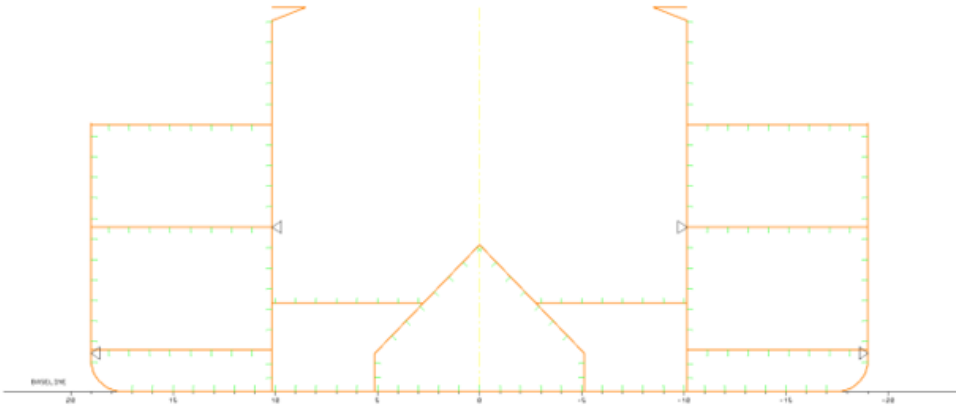

Figure 3.2.3: Schematic cross-section view of main frame scantlings.

–The inspiration you seek is already within you. Be silent and listen.

Jalāl ad-Dīn Muhammad Rūmī

4

Proposed Solution

In this chapter, a new *Constrained Efficient Global Optimization*(CEGO) algorithm is proposed, combining the strengths of both the $\mathcal{S}$-metric multi-objective optimization techniques from *SMS-EGO* and the constraint handling techniques from *SACOBRA*. This combination of the two algorithms can then be used to efficiently find an approximation of the feasible Pareto frontier of constrained multi-objective problems.

4.1 CEGO

The proposed algorithm consist of several crucial components: the initial sampling, the training of surrogate models for both objectives and constraints, defining the Pareto frontier, defining the $\mathcal{S}$-Metric Selection criterion, optimizing the infill criterion subject to the constraints, the evaluation of the found local optima, and the adjustment of the allowed constraint violations. A general overview of the algorithm is given in the CEGO psuedocode that can be found in Algorithm 1. Additionally, a flowchart of the algorithm is shown in Figure 4.1.1. Lastly the individual components of the algorithm are described in detail in the **description** following.

Algorithm 1 Pseudocode CEGO.

```
1:  par := InitialSample
2:  obj, con := Evaluate(par)
3:  eps := 0.01
4:  while eval < maxEval do
5:      objectiveModels := Kriging(par, obj)
6:      constraintModels := cubicRBF(Par, Con, EPS)
7:      paretoFront := paretoFrontFeasible(Obj, Con)
8:      criterion := SMetricSelectionCriterion(objModel, paretoFront, ref)
9:      x := findLocalOptima(objectiveModels, constraintModels, criterion, eps)
10:     obj[evall], con[evall] := Evaluate(x)
11:     eps = adjustMargins(con[evall], eps)
12:     evall := evall + 1
```

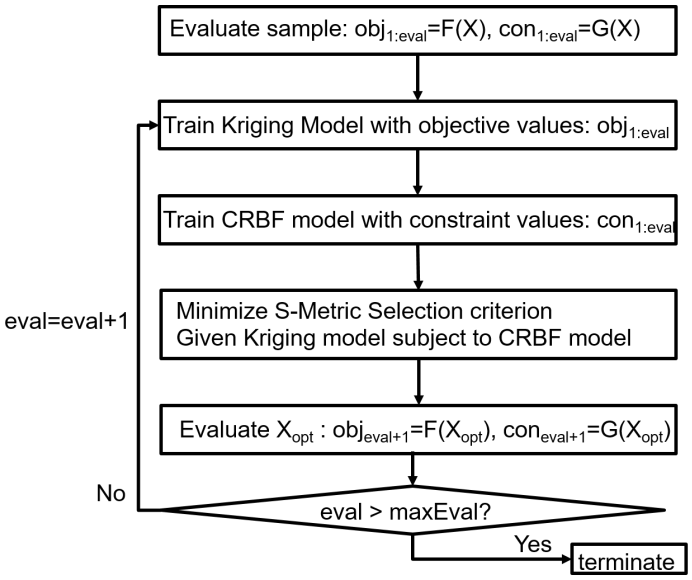

Figure 4.1.1: CEGO flowchart.

Initial Sampling The proposed algorithm starts with an initial sampling of the n decision variables using *Latin Hypercube Sampling*(LHS) [24]. The generated samples by LSH are as uncorrelated as possible. This way the average amount of information that can be gained by computing the objectives and constraints given by the LHS is maximized. The recommended size of the LHS is $11 \cdot n - 1$ [22] but for problems with a large number of parameters it can be smaller. Note that the inital size of the LHS must at least be bigger than $n + 1$ to be able to build the surrogate models [1]. The LSH samples then get evaluated for all objectives (Eq. 3.2.6) and constraints.

Objective Models The objective values are used to train the objective surrogate models. The objective surrogate models used are *Kriging* [23] (often also called Gaussian Process Regression models). For every objective function a separate Kriging model is fitted. Kriging treats every unknown objective function f as the combination of a centered Gaussian Process $\epsilon(x)$ of zero mean with an unknown constant trend μ. The advantage of using Kriging is that in addition to the predicted mean $y(x)$, the predicted uncertainty, called the Kriging variance $\sigma(x)$, is provided. The Kriging variance can be exploited in the optimization procedure (see description **Criterion for optimization**).

In Figure 4.1.2, 4.1.3, 4.1.4, and 4.1.5 a Kriging training process is presented for the $y = x^2$ function. As can be seen from the figures, the more points we use to train the Kriging model, the smaller the variance gets and the more precise the approximation of the kriging model becomes.

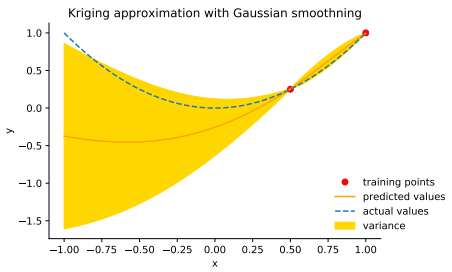

Figure 4.1.2: Kriging with Gaussian smoothing function approximation of $y = x^2$. Training points used: $x = 0.5$, and $x = 1$.

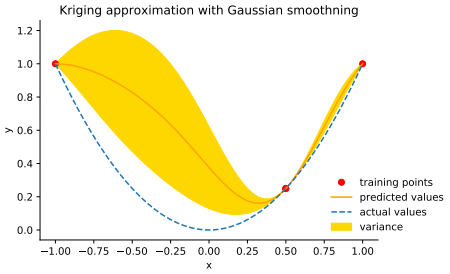

Figure 4.1.3: Kriging with Gaussian smoothing function approximation of $y = x^2$. Training points used: $x = -1$, $x = 0.5$, and $x = 1$.

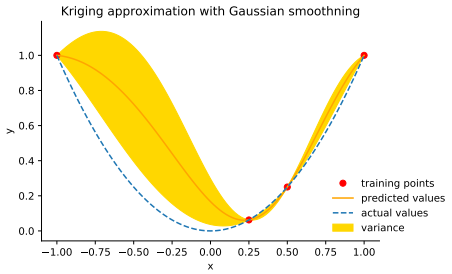

Figure 4.1.4: Kriging with Gaussian smoothing function approximation of $y = x^2$. Training points used: $x = -1$, $x = 0.25$ $x = 0.5$, and $x = 1$.

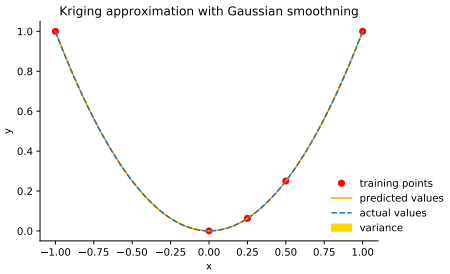

Figure 4.1.5: Kriging with Gaussian smoothing function approximation of $y = x^2$. Training points used: $x = -1$, $x = 0$, $x = 0.25$, $x = 0.5$, and $x = 1$.

Constraint Models The corresponding constraint values are used to train the constraint surrogate models. For the constraint surrogate models, *Cubic Radial Basis Functions* (CRBF) are used. Usually, to model a function with an CRBF model in a proper way, additional parameters need to be tuned manually. In CEGO this is taken care of by using the self adjusting parameter control proposed by Bagheri & co [1]. The self adjusting parameter control is used to fit a CRBF model for every constraint function. The steps taken to model the constraint functions are:

1. Rescale the search space to an interval of $[-1, 1]$,
2. Rescale the constraint functions so that they are equally important,
3. Define the distance requirement factor (DRC) that defines how close the solutions are allowed to be to each other, and alter it at every iteration,
4. Adjust the margin (ϵ) of allowed violation of the CRBF model at every iteration (see description **Adjust margins**).

In the first few iterations, the CRBF model might not fit the constraint function very well. Therefore, a violation of the constraints is allowed. The magnitude of the allowed violation decreases as more feasible solutions are found.

Now, lets assume that the objective function $y = x^2$ is subject to the constraint function $g = -x^3 - \frac{1}{4}$ where g should be smaller then or equal to 0 to be feasible. The constraint now limits the available ranges of values that the Kriging model is able to choose for x. In Figure 4.1.6, 4.1.7, 4.1.8, and 4.1.9 the available choices in each step that can be made by the Kriging model is visualized.

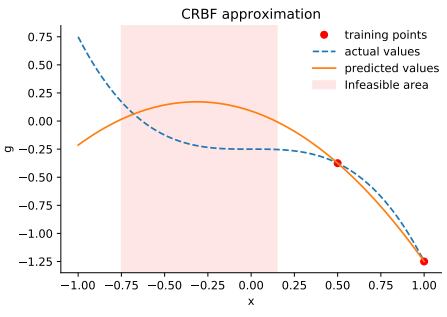

Figure 4.1.6: CRBF function approximation of $y = -x^3 - \frac{1}{4}$. Training points used: $x = 0.5$, and $x = 1$.

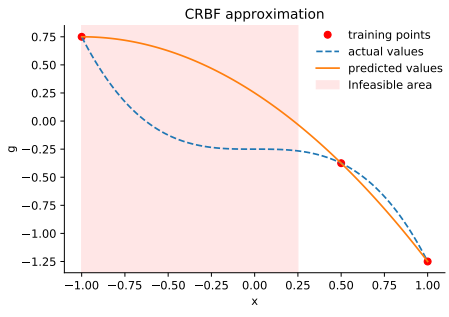

Figure 4.1.7: CRBF function approximation of $y = -x^3 - \frac{1}{4}$. Training points used: $x = -1$, $x = 0.5$, and $x = 1$.

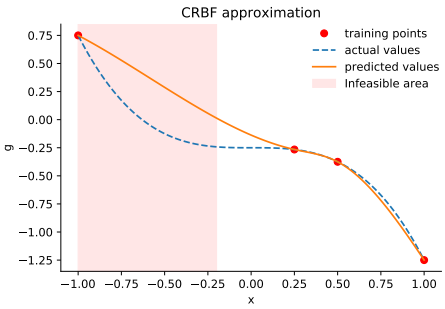

Figure 4.1.8: CRBF function approximation of $y = -x^3 - \frac{1}{4}$. Training points used: $x = -1$, $x = 0.25$ $x = 0.5$, and $x = 1$.

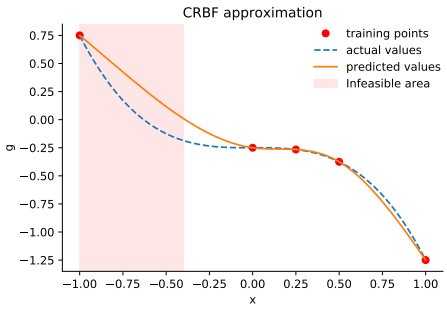

Figure 4.1.9: CRBF function approximation of $y = -x^3 - \frac{1}{4}$. Training points used: $x = -1$, $x = 0$, $x = 0.25$, $x = 0.5$, and $x = 1$.

Pareto Frontier Feasible In the previous example $y = x^2$, there is only one objective y. When more objectives and more constraints are present we have to optimize the objectives simultaneously. We do this by searching for non dominated feasible solutions. To do this, the determination of the current feasible Pareto frontier is important. The feasible Pareto frontier (denoted as $\vec{\Lambda}$) is defined by the set of feasible solutions which are also Pareto dominant according to Definition 3.

Criterion for Optimization In the criterion for optimization, CEGO uses the idea of Emmerich et al. to use $\mathcal{S}$-metric or (hyper)volume contribution [2] of a potential solution $\hat{y}_{pot}$ to the current Pareto frontier approximation. CEGO uses the $\mathcal{S}$-metric extended as an infill criterion [30]. The infill criterion function computes for a given input vector $\vec{x}$, the expected improvement $\vec{x}$ contributes to the current Pareto frontier approximation. The predicted objective scores $\hat{y}$ are predicted with the Kriging models together with their estimated uncertainties $\hat{s}$. If the 95% lower confidence bound $\hat{y}_{pot} = \hat{y} - \alpha \cdot \hat{s}$ is ϵ-dominant we compute the additional (hyper)volume it adds to the Pareto frontier. ϵ-dominance is applied to support a good distribution over the Pareto frontier. The values of $\vec{\epsilon}$ are set every iteration:

$$\vec{\epsilon} = \frac{max(\vec{\Lambda}) - min(\vec{\Lambda})}{1 + |\vec{\Lambda}| - \frac{1}{2^k} \cdot (maxEval - eval)}. \tag{4.1.1}$$

Here $max(\vec{\Lambda})$ is the maximum value per objective on the Pareto frontier, $min(\vec{\Lambda})$ is the minimum value per objective on the Pareto frontier, k is the number of objectives, *maxEval* the maximum number of allowed iterations, and *eval* the number of evaluations executed so far. The final (hyper)volume multiplied by minus one that $\hat{y}_{pot}$ adds to the Pareto frontier is the score the $\mathcal{S}$-metric criterion will return. If $\hat{y}_{pot}$ is not

expected to be ϵ-dominant, the infill criterion will return a penalty value p as presented in the following Equation:

$$p = \sum_{y^i \in \vec{\Lambda}} -1 + \prod_{j=1}^{m} 1 + \hat{y}_{pot,j} - y_j^i \tag{4.1.2}$$

3 hypothetical cases of the $\mathcal{S}$-Metric Selection infill criterion in a two dimensional objective space is shown in Figure 4.1.10. In this figure the blue triangles represent the non dominated solutions found in previous iterations, the black star represents the reference point, the grey area represents the the current (hyper)volume, and the area between the dark grey line and the grey area represents the ϵ area.

Suppose case 1, where we find the solution with the predicted objective scores $(1.65, 1.65)$. This point would be dominated by the previously found non-dominant solutions. Therefore, the infill criterion will return the penalty score: 0.3225. This score is computed with Equation 4.1.2.

Suppose case 2, where we find a solution with the predicted objective scores $(1.75, 1.05)$ which lies in the epsilon dominated area. This point does not significantly contribute anything to the Pareto frontier so the criterion will return 0.

Suppose case 3, where we find a solution with the predicted objective scores $(1.2, 1.2)$. This potential solution is not dominated by the earlier found solutions and does not lie in the epsilon dominated area. Therefore, the (hyper)volume that this point adds to the current Pareto frontier is calculated. This is equal to the surface of the green area in Figure 4.1.10. The score the infill criterion will return in in this case -0.27 which is equal to the (hyper)volume multiplied by minus one.

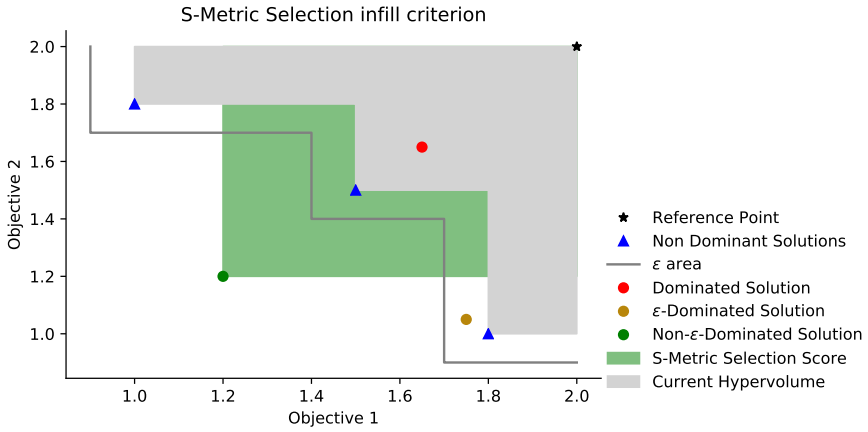

Figure 4.1.10: $\mathcal{S}$-Metric Selection infill criterion

Find Local Optima The infill criterion is optimized in such a manner that it searches for new optimal solutions that do not violate any of the constraints. This can be done with an optimization algorithm that is capable of dealing with: multiple decision variables ($\vec{x}$), single objective problem ($\mathcal{S}$-metric criterion), and multiple constraints (CRBF functions). In this research the Constrained Optimization by Linear Approximation (COBYLA) algorithm [31] is used to optimize the objectives subject to the constraints.

The problem that COBYLA can solve is setup the following way: The decision variables $\vec{x}$ can be changed within the predefined bounds. The objective is the $\mathcal{S}$-metric criterion that predicts the additional (hyper)volume given $\vec{x}$. The constraints can be interpolated with the CRBF models given $\vec{x}$ and finally a the DRC constraint is added that makes sure the solutions that get evaluated are not to close to the previously found solutions.

COBYLA can deal wich such problems and therefore is able to optimize the infill criterion under the condition that the constraints are satisfied. The vector $\vec{x}$ that is expected to be feasible and expected to contribute the most to the Pareto frontier approximation is then proposed as new solution. If in the current design space no feasible solution can be found, the vector $\vec{x}$ with the smallest expected constraint violation is chosen.

Evaluate The minimum found by COBYLA is proposed as a new solution, which is evaluated with the actual evaluation functions that are being optimized. This evaluation of $\vec{x}$ gives new objective values and constraint values that can be added to the population. The surrogate models are re-trained and the next iteration starts. This optimization process goes on until the evaluation budget is exhausted. The evaluation budget can be determined by the user. For simple problems the budget could be small but it is recommended to make the budget larger for complex problems with a lot of parameters. A guideline to take in mind is: $maxEval = 40 \cdot n$.

Adjust margins In the experiments reported in this thesis, the ϵ-value used for the constrained CRBF models starts at 0.01. When $\lfloor 2 \cdot \sqrt{n} \rfloor$ feasible solutions are found sequentially, ϵ decreases by 50%. This decrease of allowed violation can be justified by the fact that $\lfloor 2 \cdot \sqrt{n} \rfloor$ consecutive feasible solutions are found and therefore the CRBF models with the allowed violation have a better understanding of the constraint boundaries so the allowed violation should decrease. Furthermore by decreasing the allowed violation of the CRBF functions the CRBF functions can be more exploited so solutions closer to the constrained boundaries can be found.

Alternatively, ϵ increases by 100% when $\lfloor 2 \cdot \sqrt{n} \rfloor$ infeasible solutions are found. This increase of the allowed violation can be explained by the idea that the allowed violation of the CRBF is probably too tight and the CRBF models are not yet capable of modelling the constraint functions good enough yet. By increasing the allowed constraint violation margin of the CRBF models the constrained area can be explored more thoroughly so that the fit of the CRBF models can increase. Of course it does not make sense to explore the already known heavily violated constrained area, therefore the maximum of ϵ is set to 0.02 in the experiments reported in this thesis.

–Failure is just practice for success.

Christopher Hitchens

5

Experimental Evaluation

To evaluate the performance of the proposed algorithm, three different experiments are set up. In the first experimental setup, seven artificially designed problems are optimized. In the second experimental setup, seven Real World Like Problems (RWLP) are optimized. Finally, in the third experimental setup, the dredger ship design problem as described in the problem definition (Chapter 3) is optimized. All the optimization problems are optimized using the proposed algorithm CEGO, and compared with the already widely used algorithms NSGA-II, SPEA2, and MOGA.

5.1 Experimental Setup

For all experiments the number of allowed function evaluations is set to 200. The number of allowed function evaluations is limited because the algorithms should be able to find an approximation of the Pareto frontier in as little function evaluations as possible. If the algorithm is not capable of finding an approximation of the Pareto frontier in a few function evaluations the algorithm would not be suitable for real world problems, because the function evaluations of real world problems can be very expensive.

Furthermore, each algorithm is executed between 5 and 100 times per problem, depending on the time-complexity of the algorithm and the problem.

The criterion that is used to evaluate the performance is the (hyper)volume (HV) between a fixed reference point and the Pareto frontier. The reference

point values are set in such a manner that they represent the maximum values of interest for the objective functions. In Figure 5.1.1 an example of a reference point, the Pareto frontier and the volume of an example frontier is visualized. The algorithm with the highest HV between the fixed reference point and the Pareto frontier approximation can be defined as the algorithm that performed the best on the problem.

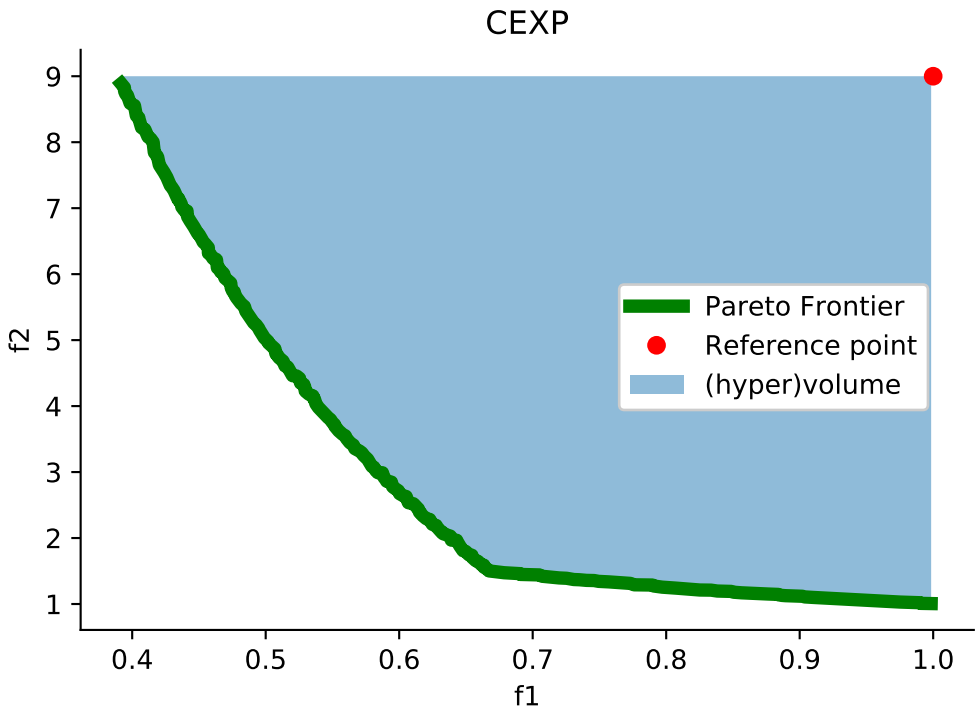

Figure 5.1.1: Pareto frontier, reference point and (hyper)volume visualized for the CEXP function.

The HV already gives a good indication of which algorithm gives the best result, but what is also of interest is the spread of the obtained solutions on the Pareto frontier approximation. To examine the spread of the obtained solutions the Pareto frontier is visualized.

In addition to the HV and the spread the convergence rate of the different algorithms can also be an interesting measure to look at. This is the case because if we have expensive evaluation functions the number of allowed function evaluations is typically limited. This means that an algorithm with a high convergence rate is more interesting compared to a not so slow converging algorithm.

5.1.1 Artificially Designed Functions

Inspired by previous studies on multi-objective optimization algorithms, seven widely used artificially designed functions are selected to experiment with: BNH [6], C3-DTLZ4 [38], OSY [12], SRN [12], TNK [12], CEXP [9] and CTP1 [9]. In Table 5.1.1 the number of objectives (k), number of variables (n), number of constraints (m), Lower Bound (LB), Upper Bound (UB) of the variables and the reference point (ref) are given for each function. To get some insight into the severity of the constraints, the percentage of feasible solutions (F(%)) is approximated by the evaluation of 1 million random samples.

Table 5.1.1: Constrained artificially designed test problems.

Problem	k	n	m	LB	UB	ref	F (%)
BNH	2	2	2	[0,0]	[5,3]	[140,50]	96.92
CEXP	2	2	2	[0.1,0]	[1,5]	[1,9]	57.14
C3-DTLZ4	2	6	2	[0,0,0,0,0,0]	[1,1,1,1,1,1]	[3,3]	22.22
SRN	2	2	2	[-20,-20]	[20,20]	[301,72]	16.18
TNK	2	2	2	[1e-5,1e-5]	[π,π]	[2,2]	5.05
OSY	2	6	6	[0,0,1,0,1,0]	[10,10,5,6,5,10]	[0,386]	2.78
CTP1	2	2	2	[0,0]	[1,1]	[1,2]	92.67

5.1.2 Real World Like Problems

The RWLP are real world like functions which are believed to be difficult because they have many complex constraints [19]. The following seven RWLP have been used in the experiments: Two-Bar Truss Design problem (TBTD) [17], Welded Beam problem (WB) [17], Disc Brake Design problem (DBD) [17], Speed Reducer Design problem (SRD) [25], Ship Parametric Design problem (SPD) [28], Car Side Impact problem (CSI) [19], and the Water Problem (WP) [19]. In Table 5.1.2, again the number of objectives (k), number of variables (n), number of constraints (m), Lower Bound (LB), Upper Bound (UB) of the variables and the reference point (ref) are given for each function. Note that if a function was to be maximized it is transformed into a minimization problem.

5.1.3 Optimizing a Dredger

Finally, the dredger case as described in the problem definition (Chapter 3) is optimized. The limits used for the dredger parameters are: delta breadth $\in [-1.6, 3.4]$, delta length $\in [-2.8, 9.8]$, foreship length $\in [16, 22]$, hopper length extension $\in [5, 9]$, hopper breadth $\in [5, 9]$, and hopper height $\in [12, 16]$.

Table 5.1.2: Constrained Real World Like Problems.

Problem	k	n	m	LB	UB	ref	F(%)
TBTD	2	3	2	[1, 0.0005, 0.0005]	[3, 0.05, 0.05]	[0.1, 100 000]	19.46
WB	2	4	5	[0.125, 0.1, 0.1, 0.125]	[5, 10, 10, 5]	[350, 0.1]	35.28
DBD	2	4	5	[55, 75, 1 000, 2]	[80, 110, 3 000, 20]	[5, 50]	28.55
SRD	2	7	11	[2.6, 0.7, 17, 7.3, 7.3, 2.9, 5]	[3.6, 0.8, 28, 8.3, 8.3, 3.9, 5.5]	[7 000, 1 700]	96.92
SPD	3	6	9	[150, 25, 12, 8, 14, 0.63]	[274.32, 32.31, 22, 11.71, 18, 0.75]	[16, 19 000, −260 000]	3.27
CSI	3	7	10	[0.5, 0.45, 0.5, 0.5, 0.875, 0.4, 0.4]	[1.5, 1.35, 1.5, 1.5, 2.625, 1.2, 1.2]	[42, 4.5, 13]	18.17
WP	5	3	7	[0.01, 0.01, 0.01]	[0.45, 0.1, 0.1]	[83 000, 1 350, 2.85, 15 989 825, 25 000]	92.06

The reference point is set to $[5\,000, 2]$. This is the case because we are only interested in design variations with a smaller resistance coefficient than 2, and design variations with a smaller steel weight than 5 000 tonnes. Furthermore, based on 200 random samples, approximately 24% of of the design space is feasible. The original dredger designed by human experts has an approximated steel weight of 2 039 tonnes and an estimated resistance coefficient of 1.08.

5.2 Algorithm Comparison

The theoretical test functions, the RWLP and the dredger problem are optimized with CEGO, NSGAII, SPEA2 and MOGA. The mean HV, the spread and the convergence rate are presented in the following subsections.

5.2.1 Hypervolume

As shown in Table 5.2.1, CEGO outperforms NSGA-II, SPEA2 and MOGA in terms of the HV measure for the final Pareto frontier with the limited

budget of 200 function evaluations for all test functions. In addition to a better performance on the HV measure, the standard deviation (std.) is also smaller for the CEGO algorithm compared to the other algorithms. This indicates that the CEGO algorithm does not only show a better performance but it is also more robust compared to the other algorithms.

Table 5.2.1: Mean (hyper)volume of the Pareto frontier to the reference point of each test problem. Bold face denotes the best result (according to Welchs t-test with significance level of 5%, the means are significantly different.).

Problem	Measure	NSGA-II	SPEA2	MOGA	CEGO
BNH	HV	5 187	5 137	4 993	**5 254**
	std.	19.41	20.60	59.96	7.633
CEXP	HV	3.414	3.141	2.950	**3.788**
	std.	0.086	0.145	0.185	0.006
C3-DTLZ4	HV	5.198	5.058	4.662	**6.098**
	std.	0.202	0.178	0.230	0.224
SRN	HV	58 179	48 780	51 863	**62 562**
	std.	1 536	4 237	18 392	7.493
TNK	HV	7.247	6.449	6.074	**8.058**
	std.	0.323	0.514	0.545	0.0003
OSY	HV	36 643	21 672	47 128	**100 375**
	std.	17 756	15 495	17 218	54.21
CTP1	HV	1.248	1.221	0.661	**1.303**
	std.	0.016	0.018	0.106	0.0004
TBTD	HV	7 868	7 060	608.8	**8 805**
	std.	470.9	674.3	1 826	8.155
WB	HV	34.07	33.67	33.93	**34.52**
	std.	0.189	0.386	0.252	0.058
DBD	HV	219.4	214.6	221.4	**227.9**
	std.	2.449	3.357	1.607	0.498
SRD	HV	1.991e+6	1.497e+6	1.662e+6	**4.157e+6**
	std.	8.965e+5	9.096e+5	1.690e+6	4 228
SPD	HV	2.454e+10	2.087e+10	1.942e+10	**3.240e+10**
	std.	3.134e+9	3.657e+9	3.749e+9	1.266e+9
CSI	HV	15.34	13.95	17.13	**23.21**
	std.	1.095	0.969	1.450	0.940
WP	HV	1.280e+19	1.131e+19	1.268e+19	**1.573e+19**
	std.	5.266e+17	6.095e+17	6.779e+17	7.340e+16

5.2.2 Spread

In the Figures 5.2.1, 5.2.2, 5.2.3, 5.2.4, 5.2.5 and 5.2.6 the non-dominated solutions of six typical test functions are visualized. From these figures it can

clearly be seen that the approximation of the Pareto frontier and the spread of the CEGO algorithm is better, compared to the other algorithms. Note that WP has five objectives and that five dimensions are hard to visualize, therefore the obtained Pareto frontier approximation obtained by the algorithms is visualized as a parallel coordinate plot with the five function values on the y-axes. Since the WP is a minimization problem we are interested in as low as possible values on every axes.

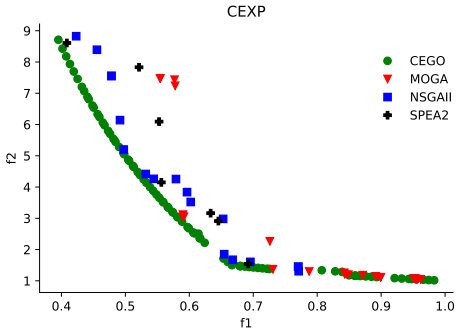

Figure 5.2.1: Pareto frontier obtained by the four algorithms on CEXP problem.

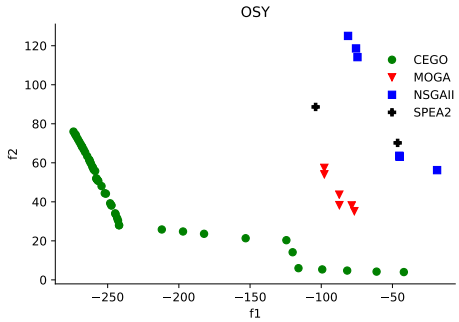

Figure 5.2.2: Pareto frontier obtained by the four algorithms on OSY problem.

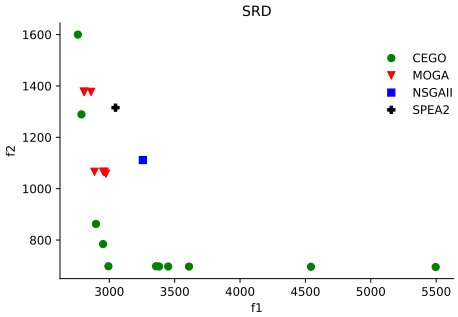

Figure 5.2.3: Pareto frontier obtained by the four algorithms on SRD problem.

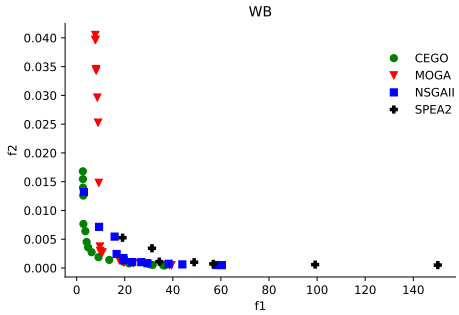

Figure 5.2.4: Pareto frontier obtained by the four algorithms on WB problem.

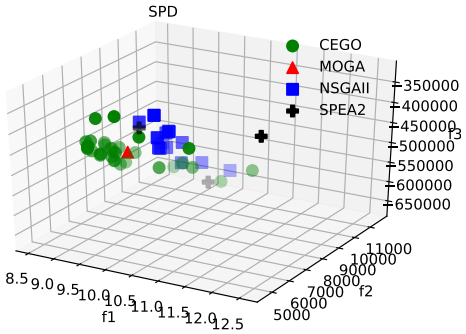

Figure 5.2.5: Pareto frontier obtained by the four algorithms on SPD problem.

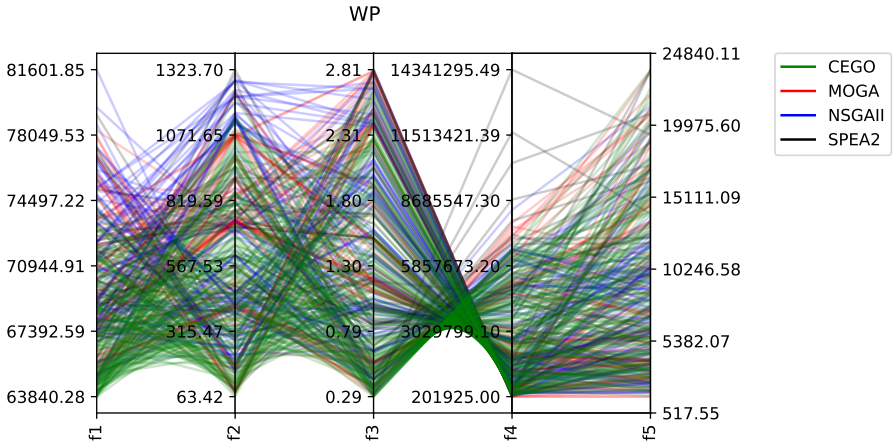

Figure 5.2.6: Pareto frontier obtained by the four algorithms on WP problem.

5.2.3 Convergence Rate

In the Figures 5.2.7, 5.2.8, 5.2.9, 5.2.10, 5.2.11 and 5.2.12 the convergence rate of a typical run of the four algorithms is visualized for six different test problems. In all the figures it can clearly be seen that after the initial 30 LHS samples, CEGO converges faster compared to MOGA, NSGAII and SPEA2. Note that in Figure 5.2.10 the HV-axis starts at ~ 29, this is the case because the first three iterations are not taken into consideration for aesthetic reasons.

Because the CEGO algorithm converges so fast on for example the CEXP and OSY problem (Figures 5.2.7, and 5.2.8), the evaluation budget for these problems could have been set to an even smaller number. On the other hand, CEGO did not fully converge yet on for example the WP problem (Figure 5.2.12). This gives an indication that a better set of solutions exists which could have been found if the evaluation budget would have been larger.

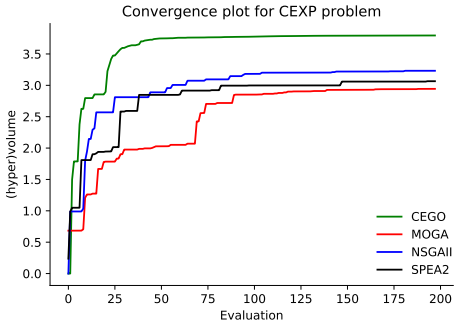

Figure 5.2.7: Convergence plot for the four algorithms on CEXP problem.

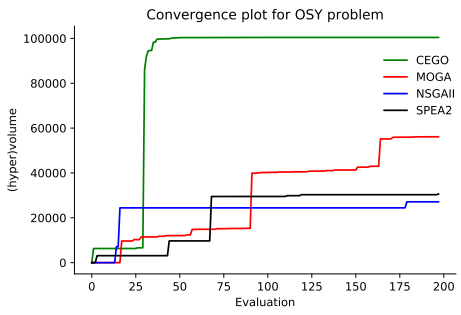

Figure 5.2.8: Convergence plot for the four algorithms on OSY problem.

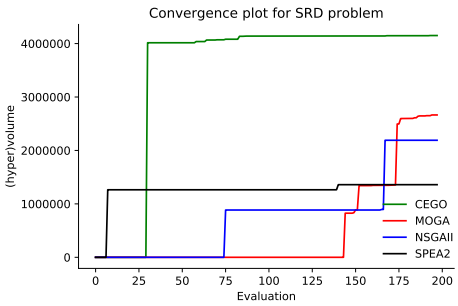

Figure 5.2.9: Convergence plot for the four algorithms on SRD problem.

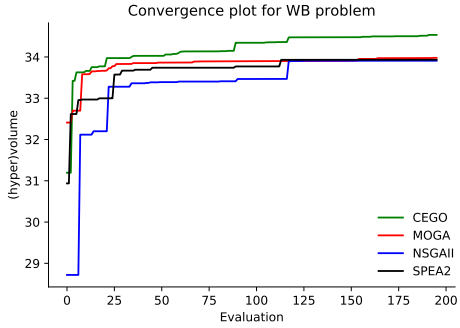

Figure 5.2.10: Convergence plot for the four algorithms on WB problem.

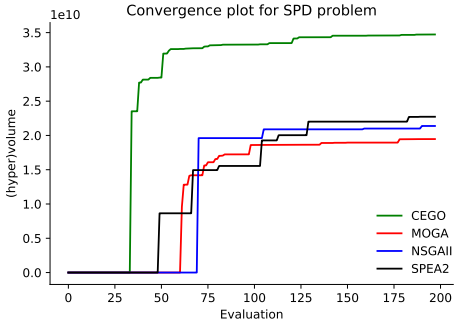

Figure 5.2.11: Convergence plot for the four algorithms on SPD problem.

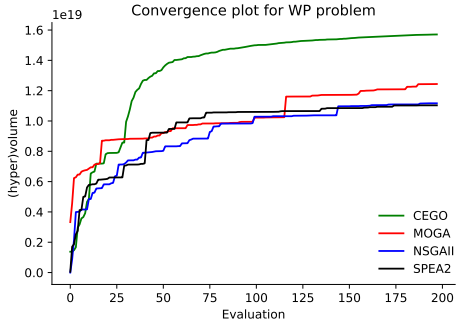

Figure 5.2.12: Convergence plot for the four algorithms on WP problem.

5.3 Dredger Optimization Results

In the dredger case, each algorithm was run 5 times, the mean HV for NSGAII, SPEA2, MOGA and CEGO were respectively 3 529, 3 579, 3 602 and **3 819**. Again, CEGO outperforms the other algorithms in terms of HV. Additionally the standard deviation of the obtained HV based on the 5 independent runs also shows that CEGO is more robust. The standard deviation obtained from NSGAII, SPEA2, MOGA and CEGO runs were 148.2, 93.35, 143.5 and 3.3 respectively.

A visualization of the obtained Pareto frontier of one of the runs for the four algorithms is visualized in Figure 5.3.1. CEGO found ten ships that dominate all those found by the other algorithms. A visualization of the convergence rate of the four algorithms is shown in Figure 5.3.2. This figure shows that none of the algorithms converged yet. It might therefore be wise to: 1) let the algorithm resume by incrementing the evaluation budget, 2) restart the algorithm with Pareto dominant solutions in the initial sample.

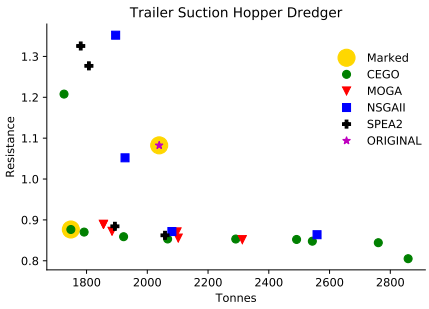

Figure 5.3.1: Original design and Pareto frontier obtained by CEGO, MOGA, NSGAII and SPEA2 on the dredger optimization problem.

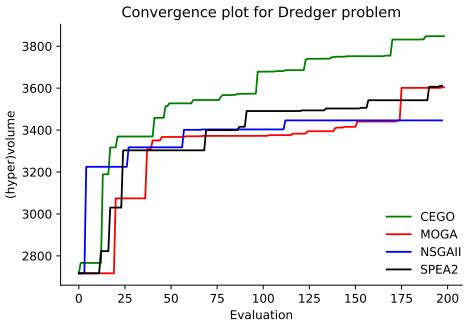

Figure 5.3.2: Convergence plot for CEGO, MOGA, NSGAII and SPEA2 on the dredger optimization problem.

The designs obtained by the algorithm would be hard to come up with for a naval architect. On the other hand, the solutions can easily be reverse engineered so that the naval architect can learn from it. Furthermore, these design variations can be used as a starting point for a naval architect to continue from. This way, more domain knowledge and intelligent corrections can be made to the design variation to turn the design into an actual concept design. For the purpose of learning from the algorithm, two dredgers that are marked in Figure 5.3.1 are graphically presented in Figure 5.3.3 and Figure 5.3.4.

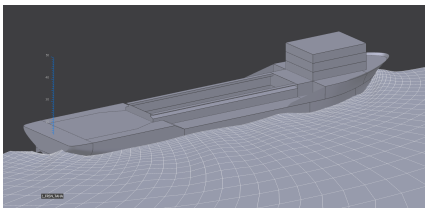

Figure 5.3.3: Original Dredger design by human experts.

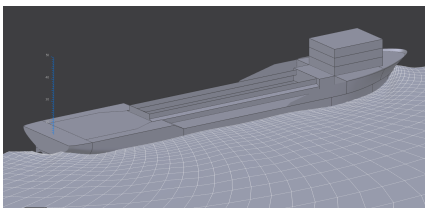

Figure 5.3.4: Dredger design optimized with CEGO.

As can be seen from the figures, the dredger found by CEGO is longer compared to the original design. Typically when a ship is longer, more steel is needed to fulfill the strength requirements of a ship. In this case the design variation found by CEGO actually has less steel weight. This is caused by a longer hopper which contributes to the overall strength of the ship. The extra strength results in less required steel in the plates and profiles which is needed to meet regulations concerning sagging and hogging.

Because the ship is longer and lighter, the ship has fewer draught and has more floating capacity compared to the original. Due to less draught and the longer design, the ship makes less waves and waves with a different wave pattern which results in less resistance.

–The key is not to prioritize what's on your schedule, but to schedule your priorities.

Stephen Covey

6

Conclusion and Future Work

In this thesis the Constrained Efficient Global Optimization (CEGO) algorithm is proposed and it is shown that CEGO is efficient in finding a Pareto frontier approximation using limited evaluation budgets for both Real-World Like Problems and artificially designed test functions. This algorithm has been shown to work in multidisciplinary ship design applications since a trailer suction hopper dredger was optimized using CEGO. When CEGO terminated after 200 function evaluations, ten unique non-dominated dredger designs were found. Compared to the original dredger design, the most efficient solution found in terms of resistance shows a 20% smaller resistance factor, and the most efficient solution found in terms of steel weight shows a 30% smaller steel weight. CEGO also outperforms state-of-the-art alternatives like NSGA-II, SPEA2, and MOGA on all of the fourteen test problems used in the experimental setup. The proposed CEGO algorithm in combination with an integrated design approach shows great potential and can be used to design ships that are more energy efficient while maintaining or even improving all other objectives.

6.1 Future Work

For future work, it would be interesting to define more independent parameters that influence the performance of the ship to obtain better solutions.

From an algorithm point of view, the proposed algorithm could be improved by taking the CRBF constraint surrogate models into account when

defining a new infill-criterion instead of using them as a constraint when minimizing the $\mathcal{S}$-metric infill-criterion. It would also be beneficial to parallelize the CEGO algorithm such that multiple evaluations can be run at the same time. Lastly it would be interesting to do more research to highly multimodal function optimization or severely constrained optimization problems. This would be interesting because surrogate models are not yet capable of modelling the highly multimodal functions in only a few function evaluations, and because still a lot of 'luck' is needed to find a feasible solution in extremely constrained design spaces.

Bibliography

[1] Samineh Bagheri, Wolfgang Konen, Michael Emmerich, and Thomas Bäck. Self-adjusting parameter control for surrogate-assisted constrained optimization under limited budgets. *Applied Soft Computing*, 61:377–393, 2017.

[2] Nicola Beume, Boris Naujoks, and Michael Emmerich. Sms-emoa: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669, 2007.

[3] Emilio Fortunato Campana, Giampaolo Liuzzi, Stefano Lucidi, Daniele Peri, Veronica Piccialli, and Antonio Pinto. New global optimization methods for ship design problems. *Optimization and Engineering*, 10 (4):533, 2009.

[4] Piya Chootinan and Anthony Chen. Constraint handling in genetic algorithms using a gradient-based repair method. *Computers & operations research*, 33(8):2263–2281, 2006.

[5] CA Coello Coello and E Mezura Montes. Use of dominance-based tournament selection to handle constraints in genetic algorithms. *Intelligent Engineering Systems through Artificial Neural Networks*, 11:177–182, 2001.

[6] Carlos A Coello Coello, Gary B Lamont, David A Van Veldhuizen, et al. *Evolutionary algorithms for solving multi-objective problems*, volume 5. Springer, 2007.

[7] Carlos A Coello Coello. Constraint-handling techniques used with evolutionary algorithms. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, pages 563–587. ACM, 2016.

[8] Kalyanmoy Deb. An efficient constraint handling method for genetic algorithms. *Computer methods in applied mechanics and engineering*, 186(2-4):311–338, 2000.

[9] Kalyanmoy Deb. *Multi-objective optimization using evolutionary algorithms*, volume 16. John Wiley & Sons, 2001.

[10] Kalyanmoy Deb. *Multi-Objective Optimization*, pages 273–316. Springer US, Boston, MA, 2005. ISBN 978-0-387-28356-2. doi: 10.1007/0-387-28356-0_10. URL https://doi.org/10.1007/0-387-28356-0_10.

[11] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and Tanaka Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *International Conference on Parallel Problem Solving From Nature*, pages 849–858. Springer, 2000.

[12] Kalyanmoy Deb, Amrit Pratap, and T Meyarivan. Constrained test problems for multi-objective evolutionary optimization. In *International conference on evolutionary multi-criterion optimization*, pages 284–298. Springer, 2001.

[13] J. HARVEY EVANS. Basic design concepts. *Journal of the American Society for Naval Engineers*, 71(4):671–678, 1959. ISSN 1559-3584. doi: 10.1111/j.1559-3584.1959.tb01836.x. URL http://dx.doi.org/10.1111/j.1559-3584.1959.tb01836.x.

[14] Carlos M Fonseca and Peter J Fleming. Multiobjective optimization and multiple constraint handling with evolutionary algorithms. i. a unified formulation. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 28(1):26–37, 1998.

[15] European Comission. Directorate-General for Mobility and Transport. *White Paper on Transport: Roadmap to a Single European Transport Area: Towards a Competitive and Resource-efficient Transport System*. Publications Office of the European Union, 2011.

[16] David E Goldberg and John H Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):5, 1988.

[17] Wenyin Gong, Zhihua Cai, and Li Zhu. An efficient multiobjective differential evolution algorithm for engineering design. *Structural and Multidisciplinary Optimization*, 38(2):137–157, 2009.

[18] International Maritime Organization. UN body adopts climate change strategy for shipping. http://www.imo.org/en/MediaCentre/PressBriefings/Pages/06GHGinitialstrategy.aspx, 2018. online; accessed 18-April-2018.

[19] Himanshu Jain and Kalyanmoy Deb. An evolutionary many-objective optimization algorithm using reference-point based nondominated sort-

ing approach, part ii: Handling constraints and extending to an adaptive approach. *IEEE Trans. Evolutionary Computation*, 18(4):602–622, 2014.

[20] Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

[21] Mifa Kim, Tomoyuki Hiroyasu, Mitsunori Miki, and Shinya Watanabe. Spea2+: Improving the performance of the strength pareto evolutionary algorithm 2. In *International Conference on Parallel Problem Solving from Nature*, pages 742–751. Springer, 2004.

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

[23] Daniel G. Krige. A Statistical Approach to Some Basic Mine Valuation Problems on the Witwatersrand. *Journal of the Chemical, Metallurgical and Mining Society of South Africa*, 52(6):119–139, December 1951.

[24] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979. ISSN 00401706.

[25] Seyedali Mirjalili, Pradeep Jangir, and Shahrzad Saremi. Multi-objective ant lion optimizer: a multi-objective optimization algorithm for solving engineering problems. *Applied Intelligence*, 46(1):79–95, 2017.

[26] A Papanikolaou, S Harries, M Wilken, and G Zaraphonitis. Integrated design and multiobjective optimization approach to ship design. In *Proceedings of international conference on computer application in shipbuilding*, volume 3, 2011.

[27] Apostolos Papanikolaou. Holistic ship design optimization. *Computer-Aided Design*, 42(11):1028–1044, 2010.

[28] Michael G Parsons and Randall L Scott. Formulation of multicriterion design optimization problems for solution with scalar numerical optimization methods. *Journal of Ship Research*, 48(1):61–76, 2004.

[29] Jendrik Poloczek and Oliver Kramer. Local svm constraint surrogate models for self-adaptive evolution strategies. In *Annual Conference on Artificial Intelligence*, pages 164–175. Springer, 2013.

[30] Wolfgang Ponweiser, Tobias Wagner, Dirk Biermann, and Markus Vincze. Multiobjective optimization on a limited budget of evaluations using model-assisted $\mathcal{S}$-metric selection. In *International Conference on Parallel Problem Solving from Nature*, pages 784–794. Springer, 2008.

[31] Michael JD Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. In *Advances in optimization and numerical analysis*, pages 51–67. Springer, 1994.

[32] Rommel G Regis. Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points. *Engineering Optimization*, 46(2):218–243, 2014.

[33] Thomas Richir, Jean-David Caprace, Nicolas Losseau, Maud Bay, Michael G Parsons, Samuel Patay, and Philippe Rigo. Multicriterion scantling optimization of the midship section of a passenger vessel considering iacs requirements. In *The 10th International Symposium on Practical Design of Ships and Other Floating Structures (PRADS)*, 2007.

[34] Thomas W Sederberg and Scott R Parry. Free-form deformation of solid geometric models. *ACM SIGGRAPH computer graphics*, 20(4): 151–160, 1986.

[35] TWP Smith, JP Jalkanen, BA Anderson, JJ Corbett, J Faber, S Hanayama, E O'Keeffe, S Parker, L Johanasson, L Aldous, et al. Third imo ghg study. 2015.

[36] Yusuke Tahara, F Stern, and Y Himeno. Computational fluid dynamics–based optimization of a surface combatant. *Journal of ship Research*, 48(4):273–287, 2004.

[37] Yusuke Tahara, Satoshi Tohyama, and Tokihiro Katsui. Cfd-based multi-objective optimization method for ship design. *International journal for numerical methods in fluids*, 52(5):499–527, 2006.

[38] Ryoji Tanabe and Akira Oyama. A note on constrained multi-objective optimization benchmark problems. In *Evolutionary Computation (CEC), 2017 IEEE Congress on*, pages 1127–1134. IEEE, 2017.

[39] Guolei Tang, Wenyuan Wang, Zijian Guo, Xuhui Yu, and Bingchang Wang. Simulation-based optimization for generating the dimensions of a dredged coastal entrance channel. *Simulation*, 90(9):1059–1070, 2014.

[40] Aimin Zhou, Bo-Yang Qu, Hui Li, Shi-Zheng Zhao, Ponnuthurai Nagaratnam Suganthan, and Qingfu Zhang. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1(1):32–49, 2011.

[41] Eckart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, 3(4):257–271, 1999.