UNIVERSITY OF SOUTHAMPTON

Improvement Criteria for Constraint Handling and Multiobjective Optimization

by

James M. Parr

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the
Faculty of Engineering and the Environment
Computational Engineering and Design Group

February 9, 2013

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND THE ENVIRONMENT
COMPUTATIONAL ENGINEERING AND DESIGN GROUP

Doctor of Philosophy

Improvement Criteria for Constraint Handling and Multiobjective Optimization

by James M. Parr

In engineering design, it is common to predict performance based on complex computer codes with long run times. These expensive evaluations can make automated and wide ranging design optimization a difficult task. This becomes even more challenging in the presence of constraints or conflicting objectives.

When the design process involves expensive analysis, surrogate (response surface or meta) models can be adapted in different ways to efficiently converge towards global solutions. A popular approach involves constructing a surrogate based on some initial sample evaluated using the expensive analysis. Next, some statistical improvement criterion is searched inexpensively to find model update points that offer some design improvement or model refinement. These update points are evaluated, added to the set of initial designs and the process is repeated with the aim of converging towards the global optimum.

In constrained problems, the improvement criterion is required to update the surrogate models in regions that offer both objective and constraint improvement whilst converging toward the best feasible optimum. In multiobjective problems, the aim is to update the surrogates in such a way that the evaluated points converge towards a spaced out set of Pareto solutions.

This thesis investigates efficient improvement criteria to address both of these situations. This leads to the development of an improvement criterion that better balances improvement of the objective and all the constraint approximations. A goal-based approach is also developed suitable for expensive multiobjective problems. In all cases, improvement criteria are encouraged to select multiple updates, enabling designs to be evaluated in parallel, further accelerating the optimization process.

Contents

List of Figures

List of Tables

Declaration of Authorship

I, James Michael Parr, declare that the thesis entitled 'Improvement Criteria for Constraint Handling and Multiobjective Optimization' and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I contributed myself;
- parts of this work have been published as: Parr et al. [2010, 2012a,b,c].

Signed:....................................... Date:...................................

Acknowledgements

"...have you thought about doing a PhD?"

This simple question began my journey over three years ago. Along the way I have been influenced by countless people and received support, assistance and inspiration in many forms.

First and foremost, I would like to acknowledge my mum, dad, brother and Emily Devine for all their support and encouragement. Without these special people, my journey would not have been possible.

Carren Holden, András Sóbester, Ivan Voutchkov, David Toal, Neil Bressloff, Ros Mizen and many other academics and staff at Airbus and University of Southampton should be acknowledged for providing both assistance and inspiration throughout my PhD.

Thanks to all my friends, both new and old. Gary Battams, Andy Cook, Stuart Jinks, Julie Cheung, Matt Palmer, Angus Webb, Sanjay Pant, Steven Johnston, Neil O'Brian, Marc Bolinches, Mario Ferraro, Chris Brooks, Praveen Thokala, Athanasios Makrodimopoulos, and Giles Endicott all deserve a mention for the coffee, cake and conversation we shared.

I would like to give a special thank you to Alex Forrester for all his support, kind words and for believing in my abilities. Special thanks also go to Jay Meas, Steve Powell and Caleb Sawade for making my journey an enjoyable and unforgettable one.

Finally I would like to acknowledge Andy Keane for all his influence, sharing knowledge, and for asking the question that started the journey.

Nomenclature

α	Exploration control parameter
δ	Trust region radius or inequality constraint control parameter
ϵ	Expected feasibility control parameter
θ	Hyperparameter governing rate of correlation
λ	Lagrangian multiplier or Kriging regression constant
μ	Mean
$\hat{\mu}$	Kriging mean
$\hat{\mu}_r$	Regressing Kriging mean
σ^2	Variance
$\hat{\sigma}^2$	Kriging variance
$\boldsymbol{\Phi}$	Probability distribution function
$\boldsymbol{\phi}$	Probability density function
$\boldsymbol{\Psi}$	Correlation matrix
$\boldsymbol{\psi}$	Correlation vector
τ	Equality constraint control parameter
a	Slack variable
d	Number of design variables or step length
$f(\boldsymbol{x})$	Objective function at $\boldsymbol{x}$
$\hat{f}(\boldsymbol{x})$	Objective function prediction at $\boldsymbol{x}$
F	Feasibility function
g_{max}	Maximum constraint value evaluated
g_{min}	Minimum constraint value evaluated
g_{limit}	Inequality constraint limit

$g(\boldsymbol{x})$	Inequality constraint function at $\boldsymbol{x}$
$\hat{g}(\boldsymbol{x})$	Inequality constraint function prediction at $\boldsymbol{x}$
$G(\boldsymbol{x})$	Inequality constraint random variable
h_{limit}	Equality constraint limit
$h(\boldsymbol{x})$	Equality constraint function at $\boldsymbol{x}$
H	Hypervolume
$H(\boldsymbol{x})$	Equality constraint random variable or hypervolume
I	Improvement function
I_H	Hypervolume-based improvement function
I_M	Maximin improvement function
$\boldsymbol{I}$	Identity matrix
k	Number of Pareto solutions
m	Number of objectives
n	Number of evaluated point
p	Hyperparameter governing smoothness
P	Probability or penalty function
q	Number of update stages
s	Trust region step
$\hat{s}(\boldsymbol{x})$	Estimated error in Kriging prediction at $\boldsymbol{x}$
$\boldsymbol{x}$	Design variable
$\boldsymbol{x}^*$	Optimum design variable
$\hat{y}$	Kriging prediction
$\hat{y}_r$	Regressing Kriging prediction
y_{max}	Maximum objective value evaluated
y_{mean}	Mean of all objective values evaluated
y_{min}	Minimum objective value evaluated
$y_{minfeas}$	Minimum feasible objective value evaluated
y^*	Optimum objective function value
$\boldsymbol{y}^*$	Pareto set
$y(\boldsymbol{x})$	Objective function at $\boldsymbol{x}$
$Y(\boldsymbol{x})$	Objective random variable

Acronyms and Abbreviations

BFGS	Broyden, Fletcher, Goldfrab and Shanno algorithm
CDF	Cumulative distribution function
CFD	Computational fluid dynamics
CSM	Computational structural mechanics
DHC	Dynamic hill climbing
DOE	Design of experiments
DSO	Design search and optimization
EGO	Efficient global optimization
FEM	Finite element method
GA	Genetic algorithm
MOEA	Multiobjective evolutionary algorithm
MOGA	Multiobjective genetic algorithm
NSGA	Nondominated sorting genetic algorithm
PDF	Probability density function
SPEA	Strength Pareto evolutionary algorithm
SQP	Sequential quadratic programming

Chapter 1

Introduction

1.1 Design Search and Optimization

Predicting performance is fundamental to every engineering design problem. Unlike other forms of design, engineering design builds upon the understanding and evaluation of underlying physical processes. Over the course of history, designers have seen radical developments in the ability to predict performance, inevitably leading towards more accurate and, in many cases, more expensive forms of analysis. Nonetheless, successful engineering design cannot be achieved through accurate analysis alone but rather from an amalgamation of analysis and methodical design decisions. By exploiting analysis, *design search and optimization* (DSO) aims to aid the decision making process.

In most engineering design problems, the designer wishes to make improvements to an initial or existing design. Whether this improvement is achieved through small design changes or a radical redesign, the designer will need to perform some sort of design search to identify the best designs. In such a scenario, the designer is faced with a number of decisions. What designs should be evaluated to see an improved performance? What designs are likely to ensure design constraints are met? What forms of analysis should be used, at what fidelity and at what cost? Making these decisions is far from straightforward.

Naturally, early forms of design search favoured simple trial and error approaches. Although structured and controlled experiments were used as early as the 18th century[1], optimization algorithms using simple heuristics only began to develop on the arrival of the digital computer in the 1960s. Since then, the field of DSO has continued to grow and has led to different genres of optimization, all with many classes of problem. Structural optimization alone has large areas of research in different classes including topology optimization, shape optimization and structural sizing.

[1]A famous study being the treatment of scurvy by James Lind (1716-94).

Most early optimizers can be categorised as either local or global. A local search aims to travel 'downhill', using information based on gradients or heuristics. Although local searches can be very efficient, they are vulnerable to local optima and depend heavily on initial designs. A global search aims to explore the design space more thoroughly, visiting many local optima before converging towards a global solution. For larger problems, better designs are often found this way but require many evaluations to reach optimal designs. Both local and global methods typically concentrated on applications involving analytical expressions that are cheap to evaluate. Although these methods still have their uses in engineering design today, they tend to be less useful when used directly to search design problems that involve expensive analysis. Over time, just as analysis has developed, so have different approaches to DSO.

1.2 Dealing with Expensive Analysis

Driven by continual improvements in computational methods, aircraft design is one area that has seen radical changes in analysis since the 1960s. It is now commonplace to use tools such as computational fluid dynamics (CFD) and computation structural mechanics (CSM) to evaluate aircraft performance. These tools enable the designer to find new designs without the development time, cost and restrictions of experimental testing. Even so, tough competition and ever-rising fuel prices still makes time and cost efficient aircraft design a very difficult task. With industry being concerned with the computational efficiency of DSO, developing efficient approaches to optimization has become paramount.

In aircraft design, using high fidelity tools to evaluate aerodynamic performance often requires the order of hours or even days. A typical design search requires many design evaluations, resulting in many days or months before an optimal design is found. Searching for a globally optimal solution on a problem with many design variables quickly becomes intractable when using traditional methods.

Perhaps the most obvious solution is to reduce the cost of the analysis. This is often achieved using lower fidelity analysis at early stages in the design process. High fidelity analysis is then employed as the design progresses towards more detailed stages, where fewer designs are explored. Although this is a common approach, better designs are often found when introducing high fidelity analysis as early on as possible in the design process [Giunta, 1997]. Another approach to reduce costs is to lower the total number of evaluations needed to converge towards optimal designs. For an experienced designer, it may be obvious what sort of optimizer is the most efficient. Some aerodynamic problems can be solved quite efficiently using local methods even when analysis is expensive [Jameson, 2004]. For problems that are less familiar or where the designer has little knowledge of the tools being used, the analysis is treated as a black-box. Although many optimizers are developed with efficiency in mind, selecting the correct tool for the job can be difficult when there is little knowledge of the problem being tackled.

Surrogate models are frequently used to deal with expensive problems. Based on a small sample of designs evaluated using the expensive analysis, surrogates essentially aim to fill in the gaps, replacing the expensive analysis with an approximation that is cheap to evaluate. Traditional searches can then be used to search the surrogate inexpensively to find new designs. Since the surrogate acts as a replacement to the expensive analysis, they are particularly useful for dealing with analysis that is treated as a black-box.

Consider Figure 1.1 as the output of some expensive black-box computer analysis[2]. This represents some design objective or performance metric, such as the drag on a wing, when two design variables that define the geometry are changed. The designer is unaware of what this design space looks like but wishes to find the global optimum, the combination of design variables that give the wing with the least drag. As the simulations are expensive, it is important to do this in the least number of function evaluations possible. Directly employing a local search may be misled by any of the local optima and directly running a global search would require too many expensive evaluations. The designer could use multiple restarts of a local search to increase the chances of finding the global optimum, but for expensive problems this can also become impractical.

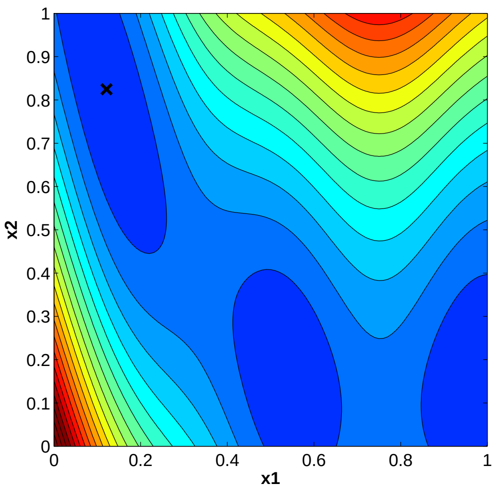


FIGURE 1.1: Output of an expensive black-box analysis with × - global optimum.

Now consider a cheap approximation of this expensive objective. Figure 1.2(a) is a surrogate based on only 15 expensive evaluations. Since the surrogate is only an approximation, the optimum according to the surrogate is not guaranteed to correspond to the optimum of the expensive analysis. It is clear that in this case the surrogate does not model the expensive analysis as well as hoped. Perhaps this is expected with so few samples, nonetheless, the global optimum is unlikely to be found by simply searching the

[2]This is in fact a modified version of the Branin test function found in Forrester et al. [2008].

surrogate. The surrogate could be improved using a larger sample but exploiting such a surrogate is still unlikely to exactly approximate the region of the global optimum. Instead the model can be updated using some sort of adaptive sampling. By using a statistical based improvement criterion, model updates can be selected methodically to offer improved performance or model refinement. This finds the global optimum much more efficiently. Figure 1.2(b) adopts such a strategy using ten additional model updates and the global optimum is located with very few evaluations of the expensive analysis.

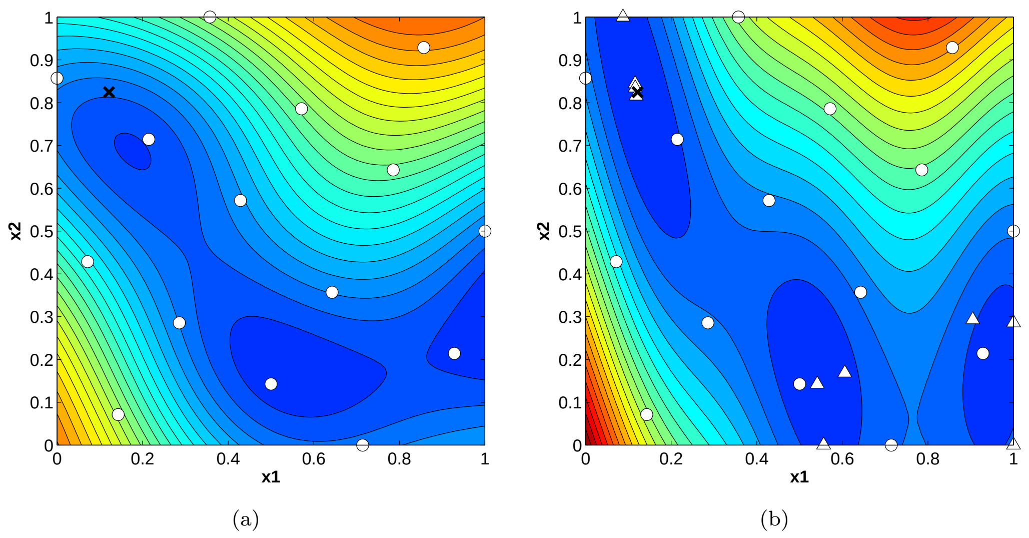


FIGURE 1.2: Objective approximation with $\times$ - global optimum. Surrogate model after (a) $\bigcirc$ – 15 expensive evaluations, (b) $\triangle$ – 10 additional updates.

The efficiency and reliability of surrogate-based optimization depends heavily on the type of improvement criterion used. Conventionally, these model update points are added sequentially, rebuilding a new surrogate each time the analysis is evaluated. Provided resources exist to evaluate several designs in parallel, updating the surrogate using multiple updates offers a promising avenue to further accelerate convergence towards global optima. This idea is exploited in many of the methods investigated within this thesis.

1.3 Research Objectives

In the area of surrogate-based optimization, the majority of literature concentrates on unconstrained problems. In reality, design problems are commonly subject to a number of design constraints. This is perhaps almost always the case in engineering design since a designer will be required to meet conflicting performance constraints, safety requirements or even manufacturing restrictions. Keeping this in mind, there has been relatively little research which adapts improvement criteria to be suitable for dealing with design constraints. This work aims to contribute in the field of surrogate-based optimization with focus on efficient improvement criteria for constrained problems.

Furthermore, most engineering design problems are rarely characterised by a single objective function. An aircraft wing design that seeks only optimum aerodynamic performance, i.e. low drag, irrespective of the structure is likely to be a structurally complex and heavy design. A better wing design is much more likely to be characterised by seeking trade-offs between good aerodynamic performance and low structural weight. Even if a single performance goal is sought, it is likely the designer wishes to explore the trade-offs between performance and cost. After all, a good design is often a profitable one. This work therefore also aims to develop efficient improvement criteria for surrogate-based multiobjective optimization.

The reader should note that methods developed within this thesis are intended for applications in aerospace design. Different methods are tested with an emphasised on real engineering design problems rather than benchmarking performance on a large library of artificial test functions.

1.4 Thesis Overview

This thesis first introduces some existing approaches in design search and optimization. This is by no means exhaustive but provides a background for the subsequent work. Chapter 2 includes an overview of local and global optimization such as early gradient-based methods, pattern searches and evolutionary methods. Further emphasis is made on multiobjective and constrained optimization, areas of research of particular interest in this thesis. Using approximations in optimization is introduced with further detail on global optimization using surrogates. A brief overview of initial sampling, Kriging and improvement criteria forms the basis for surrogate-based global optimization of unconstrained, single objective problems.

In Chapter 3, surrogate-based optimization is extended to deal with constrained problems. Three basic methods are reviewed, comparing simple penalty approaches and a probabilistic approach to constraint handling. This sets a benchmark on some test problems that are investigated further in Chapter 4. Here, the probabilistic approach is enhanced, aiming to better model constraint boundaries. Chapter 4 also investigates the use of multiobjective optimization to better balance exploration and exploitation of the objective and all constraint approximations. These enhanced approaches offer improvements in both optimization efficiency and reliability when compared to the simple approaches tested in Chapter 3.

Chapter 5 concentrates on accelerating surrogate-based constrained optimization further by evaluating multiple updates in parallel. Some existing approaches for selecting multiple updates are modified to handle constraints and compared with selecting multiple updates based on Pareto optimal solutions. These different approaches are implemented with the aim of halving the optimization cost associated with evaluating single designs sequentially. These multiple update improvement criteria are used to optimize test problems introduced in Chapter 3 and a real engineering design example, minimizing the structural weight of a transonic passenger aircraft wing.

In Chapter 6 we move away from constrained optimization and concentrate on surrogate-based multiobjective optimization. In a similar manner to Chapter 3, this aims to benchmark the current state of research and tests some existing methods for efficient multiobjective optimization on a number of test problems, including real engineering design examples. Chapter 7 extends the multiobjective improvement criteria tested in Chapter 6 to select multiple updates. This includes a novel approach based on selecting goal points to explicitly improve on interesting regions of a Pareto set. This chapter demonstrates significant savings in the optimization cost when evaluating multiple updates in parallel.

Finally, in Chapter 8 a selection of methods are used to optimize an expensive wing design problem. Here we use constrained surrogate-based optimization for wingbox sizing and seek trade-offs between the wingbox structural weight and the wing drag using surrogate-based multiobjective optimization. This demonstrates the use of different improvement criteria developed within this thesis on a constrained multiobjective design problem with objectives from two disciplines.

Chapter 2

Some Existing Approaches to DSO

2.1 Local Optimization

Pioneers of early optimization techniques were often concerned with making improvements to existing designs rather than creating an optimal design from scratch. This process involves making small perturbations of design variables in an attempt to drive the design in the direction of improved performance, known as hill climbing. Gradient-based methods use gradient information to direct the search up or downhill. Formally, for an unconstrained minimization of $f(\boldsymbol{x})$, with an initial guess $\boldsymbol{x}_0$, a gradient-based search aims to compute the next iteration $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + d\Delta\boldsymbol{x}$, where d is a step length and $\Delta\boldsymbol{x}$ is the direction of steepest descent.

In its simplest form, gradients can be exploited to determine the direction of steepest decent. Moving an optimal step length in this direction and repeating the process will inevitably lead to some design improvement. To improve the speed of convergence, some methods exploit gradient information by assuming that the objective function in the region close to an optimum can be approximated by a quadratic form. This can be achieved quite simply using Newton's method, which begins by approximating the objective function at $\boldsymbol{x}_i + d$ using a Taylor series expansion about $\boldsymbol{x}_i$

$$f(\boldsymbol{x}_i + d) = f(\boldsymbol{x}_i) + d\nabla f(\boldsymbol{x}_i). \tag{2.1}$$

Rather than finding the root of a function, Newton's method is used to find a point of zero gradient. We therefore seek a step d that satisfies the condition $\nabla f(\boldsymbol{x}_i + d) = 0$ and we differentiate to get

$$\nabla f(\boldsymbol{x}_i) + d\nabla^2 f(\boldsymbol{x}_i) = 0. \tag{2.2}$$

If the quadratic approximation holds, the point of zero gradient is also the function optimum. Since this is unlikely to correspond to the exact function minimum, this condition is instead used to compute the next step in the optimization. The next iteration $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + d$ can be expressed as

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \boldsymbol{H}^{-1}\nabla f(\boldsymbol{x}_i), \tag{2.3}$$

which is expressed in terms of the Hessian, $\boldsymbol{H}$, the matrix of second order partial derivatives.

Newton's method assumes the Hessian is known or easy to compute but in reality this is rarely the case and instead quasi-Newton methods use approximations of the inverse Hessian to calculate the next step [Rao, 1996]. A popular quasi-Newton algorithm is the BFGS method named after its inventors Broyden, Fletcher, Goldfarb and Shanno. See Nocedal and Wright [2006] for a full discussion and implementation of the BFGS algorithm and similar quasi-Newton methods.

In cases where gradient information is not available, inaccurate or difficult to calculate, pattern searches, also known as direct methods or zeroth order methods, can be used instead. These derivative free algorithms are based on selecting a trial point within the vicinity of the initial or current design. If performance is improved, the search steps in the direction of this trial point. If no improvement is found, a trial point in a different direction is selected. If a number of trial points all fail to offer any improvement on the current design, a new step length is chosen and the process is repeated. Commonly used algorithms include Hooke and Jeeves [1960], the simplex algorithm [Nelder and Mead, 1965] and the multidirectional search of Torczon [1997].

Both gradient-based methods and pattern searches are popular for noise-free optimization since, in low dimensions, they use relatively few function evaluations to converge to an optimum of high precision. However, methods based on gradients naturally converge into the closest basin of attraction and pattern searches have no way of accepting a poor trial point even if it may offer long term improvement to the direction of the search. This causes both gradient-based methods and pattern searches to be depended on the location of the initial design and can only guarantee convergence towards a local optimum as illustrated in Figure 2.1. Gradient-based methods are also highly dependent on the method used to estimate local gradients and sensitive to any noise and uncertainty in the objective.

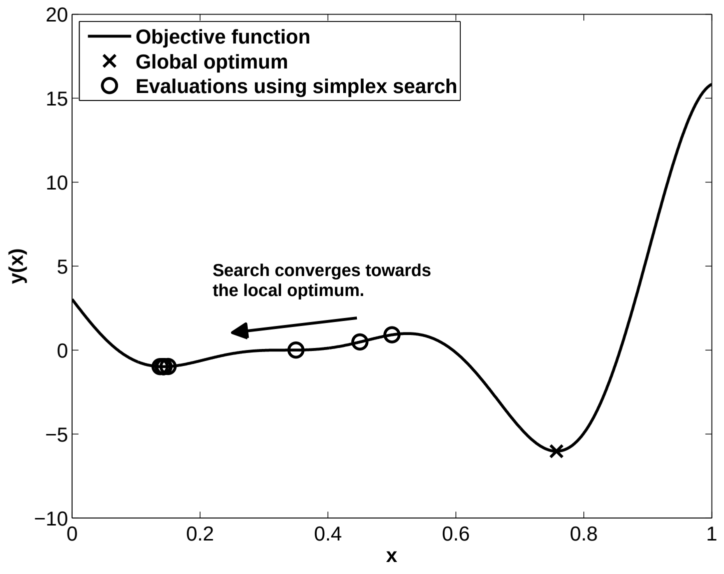


FIGURE 2.1: An example of the simplex search converging towards the local minimum of the Forrester function [Forrester et al., 2008].

2.2 Global Optimization using Evolutionary Algorithms

A popular approach to encourage a global search is to use locally converging methods from a number of different starting locations. This approach attempts to explore the design space more thoroughly and assumes one of the local optima identified is also the global optimum. This may work well in practice provided a large number of starting points are used but the expense of such an approach can quickly become impractical and is not guaranteed to identify the global optimum in highly multimodal problems.

Evolutionary algorithms attempt to redeem the shortcomings of gradient-based methods and pattern searches. Some common themes for inspiration include Darwin's theory of evolution, annealing in metallurgy and natural social behaviour of animals or insects. Although the composition of these different algorithms can be very diverse, they share some fundamental features. Firstly, these methods are all stochastic in the sense that they make use of random sequences. Repeating the search a number of times may achieve a broadly repeatable result but the path the search takes to get there is likely to be very different. Secondly, these algorithms tend not to be concerned with making improvements directly to an initial or existing design. They instead usually rely on an initial population of designs which evolve in some sense, directing the search towards the global optimum. All these methods also allow for the incorporation of poor designs, allowing the search to benefit from designs that do not offer objective improvement but instead help guide the search. Provided a sufficient number of design can be directly evaluated, these features allow the search to explore globally and enables the global optimum to be identified even on the most complex and multimodal problems[1].

[1]See for example Keane's bump function, http://www.soton.ac.uk/∼ajk/bump.html.

Coello Coello et al. [2007] provides heuristics for different evolutionary algorithms including simulated annealing, tabu search, ant systems and particle swarms. Here, two evolutionary algorithms that are used throughout this thesis are discussed further.

2.2.1 Genetic Algorithm

Genetic algorithms (GAs) are arguably the most popular of the evolutionary algorithms and have received copious attention for several decades since their introduction by Holland [1962]. They are based on Darwin's theory of evolution where several processes are applied to a population of individuals to mimic the progressions of natural selection. To aid in the evolutionary process, design variables in real numbers, known as the phenotype, are mapped to a binary (or other) encoded equivalent, the genotype. This allows each individual in the population to be manipulated easily, creating new designs when converted back into real design variables.

The binary encoded genotype is for a phenotype $(0.1328, 0.1641)$ where $x_1, x_2 \in [0, 1]$ is pictured in Figure 2.2.

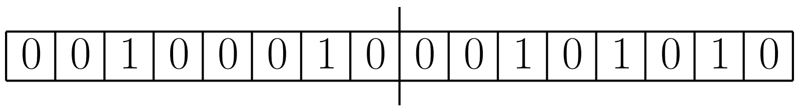


FIGURE 2.2: Binary encoding of a design using eight binary digits for each variable.

A key process of the genetic algorithm is the breeding between pairs of designs to spawn the next generation. Individuals of the population are selected to be parents based on a random procedure with some bias towards individuals with a higher fitness (lower objective value if minimizing). This makes it more likely that the strongest individuals are allowed to breed and pass on beneficial properties to the next generation. Although pressure is applied to use parents of greater fitness, poor designs with a lower fitness value are not neglected but do gradually die out. This encourages exploration of the design space during early generations and converges towards the best designs as generations pass.

The breeding process is performed via crossover of the two encoded genotypes, exchanging a random length of bits between them. Encouraging selection of parents with high fitness and using crossover alone is often enough to generate a working genetic algorithm but such a basic search may get stuck in local regions if some members of the population are over dominant. One way to escape local attractions and encourage a higher level of global exploration is to introduce mutation, where-by after crossover there is a probability that a genotype will get mutated. This mutation step can simply consist of switching a single digit in the genotype. Another mechanism useful in genetic algorithms is the idea of elitism. This involves directly carrying on the individual(s) with the greatest fitness from one generation to the next without applying crossover or mutation. This ensures the information of the best designs is not lost and further encourages an improvement of average fitness from one generation to the next.

Including all these processes, the design of a GA can be defined by five parameters; population size, number of generations, probability of crossover, probability of mutation and the level of elitism. The choice of these parameters will depend heavily on the problem at hand, however, it is common for trade-offs between cost and accuracy to be made. For example, a large population size will help ensure global exploration of the design space but will require a very large number of function evaluations. Reducing the population size but increasing the probability of mutation may achieve a similar level of exploration by increasing diversity with fewer function evaluations but it is unlikely the global optimum will be found to an equivalent accuracy. It is often necessary to carefully tune these parameters to gain the best performance, see Keane [1995].

Although the correct choice of GA parameters will achieve a reliable search on even the most complex problems, the nature of using binary encoded genotype can restrict the level of exploitation. This makes it difficult for the GA to converge towards an optimum with high accuracy. A common solution is to hybridize the GA with a locally converging search. An overview of different kinds of hybridization is given by Gudla and Ganguli [2005]. In post-hybridization a global and local search is used in two phases. First the GA is used to locate the global region and then a local search is used to converge to an optimum value within this region. This is a common approach and GAs have been combined with both pattern searches [Chelouah and Siarry, 2003], and gradient-based methods [Gudla and Ganguli, 2005]. Both the traditional GA and in its hybridized form have been used consistently throughout literature. Examples of a GA being applied in aerodynamic design can be found in Sóbester and Keane [2002], Vicini and Quagliarella [1998] and in structural design Erbatur et al. [2000] and Keane [1994].

2.2.2 Dynamic Hill Climbing

A less popular approach to global optimization is dynamic hill climbing (DHC). Introduce by Yuret and Maza [1993], DHC takes ideas from both genetic algorithms and classic hill climbing techniques. The algorithm begins by selecting a random individual which is mutated to expand the population to have $2d + 1$ members, where d is the number of design variables. The mutation simply consists of adding a scalar value to one of the design coordinates. Each member of the population is evaluated and the member with the highest fitness is selected as the parent for the next population. If a parent gets selected twice in a row, its age increases and if a mutated individual gets selected, its age is inherited from its parent. As the algorithm progresses, the magnitude of the scalar applied in the mutation step is reduced with increasing age. Once the age is beyond a certain limit, the mutation size is sufficiently small to converge upon an accurate solution. This strategy is a simple and an effective hill climber, but is only guaranteed to converge towards local optima.

To escape local regions, two important processes are added to the algorithm. Firstly a dynamic coordinate frame is used, which allows the search to choose a different coordinate frame if the search gets stuck. By recording previous moves, the Gram-Schmidt orthogonalization is used to find a new direction which is applied in the next mutation step. Secondly, once a certain age is reached the optimum found is recorded and the search is restarted. The location for this restart is decided based on the result from all previously found optima. To ensure global exploration, the hill climber is used to maximize diversity in the recorded optima and to select a new starting point. This does not require further function evaluations since a measure of diversity is used as the function fitness, which is based on the previous optimum points. This DHC process is repeated until a specified number of function evaluations is reached.

Yuret and Maza [1993] demonstrated superior performance of the DHC method compared to a basic genetic algorithm on several test problems. DHC can be applied to complex problems with a high number of dimensions and has been used by Toal et al. [2008] and Keane [2006] to tune hyperparameters in surrogate modelling.

2.3 Multiobjective Optimization

It is common for engineering design problems to have a number of conflicting objectives. In aerospace design, systems are often required to have a high performance whilst being light weight, strong and robust. Even in problems with a single performance goal, it is likely the designer is required to also design for minimum cost.

With these multiple goals in mind it is clear that selecting an optimal design is not possible unless some form of weighting is assigned to each objective. In some problems it may be possible to predefine weightings on each goal, in such cases the multiobjective problem can be transformed into a single aggregated objective function and solved using traditional optimization methods. In most cases the assignment of weightings becomes subjective and radical designs may be missed. Without prior knowledge of which weighting produces the most desirable design, it is necessary to form a set of solutions which are all optimal in some sense. This set of solutions is referred to as the Pareto set.

If a set of designs is evaluated, the designs which are Pareto optimal are all solutions that are nondominated by the performance of all other solutions in all the goals. The identified nondominated solutions make up the Pareto set and the full family of all possible Pareto optimal designs make up the Pareto front [Fonseca and Fleming, 1995].

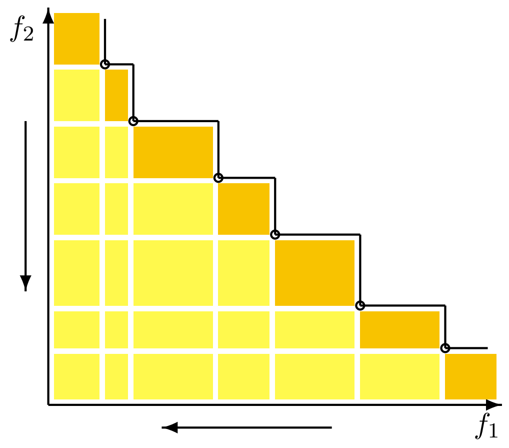


FIGURE 2.3: A Pareto set of six nondominated points. The solid line is the Pareto front. 🟧 – Regions that augment the Pareto set. 🟨 – Regions that dominate the Pareto set.

Figure 2.3 presents an example of two objectives presented as a Pareto front. In this figure it is obvious to see any improvement in the objective f_1 will be detrimental to the performance of f_2 and vice versa. If an additional Pareto optimal solution is added which lies within the yellow region, the new point is nondominated and will replace one or more of the current solutions in the Pareto set. New points lying in the orange region augment the set, adding points to the current Pareto set.

Rather than converging to a single solution, the goal of multiobjective optimization is to construct a well balanced and spaced out Pareto set. Trying to achieve a number of spread out Pareto optimal solutions requires additional computational effort. Furthermore, the concept of local and global convergence still applies to the search of a Pareto set and the mapping between design variables and the different objective spaces is usually highly nonlinear. This requires all objective functions to have a large and even spread of solutions to ensure a global Pareto set is found.

As already mentioned, a simple way to construct a Pareto set is to combine the objectives into an single aggregate function using weightings. This aggregate function can then be treated as a single objective optimization problem to find a single nondominated solution. By cycling through different combination of weights it is possible to identify a set of Pareto solutions [Vicini and Quagliarella, 1998]. For some problems the Pareto front may be disconnected or nonconvex and a simple linear sum of weights will not identify the full Pareto set. Alternative aggregate functions can be used to alleviate some of these difficulties but since the shape of the Pareto front is not known prior to its formulation, it is unclear what form the aggregate function should take. Simple aggregation functions also require scaling of each objective function and without prior knowledge of the full problem, choosing a suitable scaling adds further difficulties.

2.3.1 Multiobjective Evolutionary Algorithms

One way to overcome the limitations of directly solving single aggregate functions is to formulate the Pareto set through a multiobjective evolutionary algorithm (MOEA). Early algorithms retained the idea of using a single aggregated function for assigning fitness, see for example Schaffer [1985] and more recently Zhang and Li [2007]. Arguably, the most successful algorithms are those that base fitness assignment on Pareto optimality and not individual objective performance [Coello Coello, 2006]. The basic idea here is to rank the solutions according to their dominance over all other solutions in the population. The first set of nondominated solutions will be assigned rank 1 and then removed from the population. The next set of nondominated solutions are identified and assigned rank 2 then removed and so on, until the whole population is suitably ranked, Figure 2.4. This idea was originally adopted by Fonseca and Fleming [1993] in their multiobjective genetic algorithm (MOGA) and soon after in the nondominated sorting genetic algorithm (NSGA) proposed by Srinivas and Deb [1995].

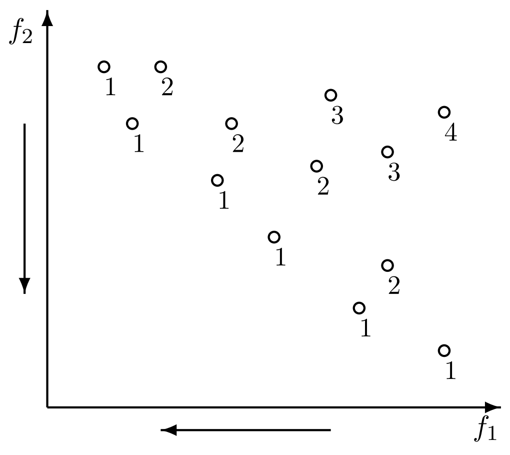


Figure 2.4: Nondominance ranking.

Further development in this field was seen after the introduction of the strength Pareto evolutionary algorithm (SPEA) proposed by Zitzler and Thiele [1999]. The main improvement made to previous algorithms was the storage of nondominated solutions from previous populations. This can be considered a similar property to elitism in traditional evolutionary algorithms. These improvements were quickly adopted by a number of other authors, see Coello Coello [2006]. Among this second generation of algorithms was the much improved NSGA-II [Deb et al., 2002]. This method does not use an external storage of previous nondominated solutions but instead uses an elitist approach by combining the best parents with the best offspring, also known as $(\mu + \lambda)$-selection.

A comprehensive survey of popular multiobjective evolutionary algorithms is presented by Ghosh and Dehuri [2004]. Despite extensive research, using MOEA's directly to solve multiobjective problems requires a very large number of function evaluations to converge to a global Pareto set. If a single objective algorithm runs in time $\mathcal{O}(GN)$ where G is the number of generations and N is the population size, it is common for a multiobjective optimization algorithm to run in time $\mathcal{O}(GMN^2)$, where M is the number of objectives [Deb et al., 2002].

2.4 Constrained Optimization

It is perhaps almost always the case that design optimization problems are subject to a number of constraints. It is important to effectively deal with these constraints during DSO to ensure a feasible optimum design is found. Formally, a single objective constrained problem involves the minimization (or maximization) of a function $f(\boldsymbol{x})$ subject to multiple inequality constraints $g_i(\boldsymbol{x}) \geq 0$ and equality constraints $h_j(\boldsymbol{x}) = 0$. Since any constraint adds complexity to the problem, where possible the designer should aim to reduce the number of constraints. In some cases this may be achieved through problem transformation [Keane and Nair, 2005]. A typical example of problem transformation occurs in optimizing a wing for minimum drag and for a fixed lift. A direct approach is to allow the angle of attack to vary as a design variable and set fixed lift as an equality constraint. Since the angle of attack directly controls the amount of lift, a better approach is to eliminate the angle of attack as a design variable and instead evaluate the wing design at different angles of attack. The correct angle of attack and corresponding drag value for a given lift can then be identified from the linear lift-curve slope. This reduces the design space and eliminates the lift equality constraint.

A typical design constraint in structural optimization involves ensuring the maximum allowable stress is below the material yield stress. For a full analysis this requires a constraint to be imposed for every element in the finite element mesh, resulting in hundreds if not thousands of constraints. If the problem is well understood, a certain dimension, typically thickness, may be directly related to the level of stress in the model and it may be possible to transform the problem in a similar manner to the wing design example. For more complex models this is unlikely to be the case and the designer may reduce the number of constraints through some form of constraint aggregation or constraint lumping [Martins et al., 2004].

2.4.1 Lagrangian Multipliers

With some prior knowledge of the design problem, the designer may wish to eliminate any constraints that are inactive. A constraint that is inactive does not influence the final feasible solution and should be omitted to reduce any computational burden of added constraints. In practice it may not be immediately obvious which constraints are active and which are inactive. Lagrangian multipliers are able to handle constraints by indentifying which constraints are active at the local optimum. Considering only equality constraints, $h_j(\boldsymbol{x}) = 0$, the original constrained problem is replaced with the Lagrangian, $f(\boldsymbol{x}) - \sum \lambda_j h_j(\boldsymbol{x})$. The minimum is found by finding a solution where the gradients of the Lagrangian all go to zero at the same time as the constraints. For inequality constraints, $g_i(\boldsymbol{x}) \geq 0$, it is necessary to introduce slack variables, a_i and the Lagrangian becomes $f(\boldsymbol{x}) - \sum \lambda_i \left[g_i(\boldsymbol{x}) - a_i^2\right]$, which essentially treats all inequality constraints as equality constraints. At the minimum for each constraint either the Lagrangian multiplier λ_i or the slack variable a_i will be equal to zero. A slack variable equal to zero suggests the inequality has been reduced to an equality constraint and a Lagrangian multiplier equal to zero indicates the constraint is inactive, see Keane and Nair [2005] for examples. A popular approach that makes use of Lagrangian multipliers is sequential quadratic programming (SQP), see Boggs and Tolle [1995].

If a constraint is active and the solution lies on the constraint boundary, this is referred to as being 'tight' [Sasena, 2002]. Figure 2.5 illustrates cases where the constraint is both active and tight and active but not tight. Lagrangian multipliers deals with constraints that are active and tight and ignores constraints that are not tight. Using a local search with Lagrangian multipliers will work well in Figure 2.5(a) provided the search is initiated in a the region of the local optimum and will happily omit the constraint in Figure 2.5(b). It becomes less obvious which constraints should be omitted when searching for a global optimum and ignoring the constraint in Figure 2.5(b) will lead to an infeasible solution.

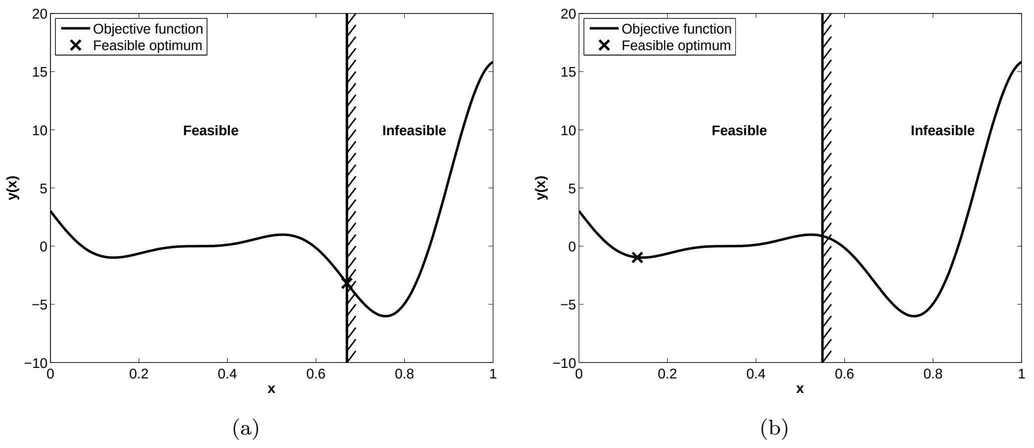


FIGURE 2.5: Active and tight constraints × - minimum feasible solution. (a) Active and tight. (b) Active but not tight.

2.4.2 Penalty Methods

Using Lagrangian multipliers assumes the gradient or Hessian of the Lagrangian can be computed. In cases where gradients are unavailable or perhaps expensive to compute, it becomes necessary to employ heuristic approaches. Methods that can be applied to all problems and integrated into any local or global search are those methods based on penalty functions.

The simplest penalty method, known as a one pass external penalty function, simply adds a large constant to the objective value whenever a constraint is violated, $f_p(\boldsymbol{x}) = f(\boldsymbol{x}) + P$. This transforms the constrained problem into an unconstrained one with a clear distinction between feasible and infeasible designs. Minimizing this modified version of the objective function will lead to a solution in the strictly feasible design space. Due to the simplicity of penalty methods and applicability to any constrained problem, it is not surprising that they are a popular approach and commonly used in industry.

Using a one pass penalty method can cause problems for local searches since a solution starting in the infeasible region is not guaranteed to descend towards feasible solutions. This is less of a problem for evolutionary algorithms but the one pass penalty also leads to a discontinuity along the constraint boundary. This steep cliff can make it difficult to converge towards a tight solution that lies along the constraint boundary. Perhaps another important consideration is that it does not take into consideration the degree of constraint violation or the total number of violated constraints. This can be an important consideration in engineering design since a designer may be satisfied with a small degree of constraint violation if it results in a significant improvement in the objective. An intuitive approach to handling the constraints is to scale the penalty on the severity of constraint violation. One possible alternative is basing the penalty on the number of constraints that are violated but this may lead to more discontinuities in the landscape and remain difficult to search. More complex forms of scaling can be used to ensure the search is directed towards feasible regions but these methods lead to a number of penalty parameters that need to be fixed.

2.4.3 Filter Methods

A penalty parameter that is too small may falsely identify feasible regions and a penalty that is too large can flatten the features of the objective function. In engineering design, prior knowledge of the problem is rarely available and selecting the best penalty parameters becomes very difficult to achieve. Fletcher and Leyffer [2002] overcome the difficulties associated with choosing effective penalty parameters by introducing the concept of a filter.

The filter treats the minimization of the objective and satisfaction of constraints as separate aims. For a single inequality constraint $g(\boldsymbol{x})$ the constrained problem can be redefined with two objectives where we wish to minimize $f(\boldsymbol{x})$ and also minimize some constraint violation function $c\,(g(\boldsymbol{x}))$. This is analogous to multiobjective optimization and it is convenient to make use of the concept of domination. If f^i and c^i denote values of $f(\boldsymbol{x})$ and $c\left(g(\boldsymbol{x})\right)$ evaluated at $\boldsymbol{x}^i$ then the pair dominate another pair f^j and c^j if $f^i > f^j$ and $c^i > c^j$. A filter is simply a set of solutions that dominate all other solutions and acts as a way to accept or reject a new step. Filter methods are often employed when using SQP and a new step is excepted if a new trial point dominates any points in the filter. This new nondominated point remains in the filter and any dominated points are removed. Each new point excepted either improves the objective function or improves constraint satisfaction.

Audet and Dennis [2004] extend filter methods for derivative free optimization. A similar approach is adopted, replacing SQP with a generalized pattern search. Brekelmans et al. [2005] also combine the filter method with derivative free optimization, using the trust region approach to guide the search towards local feasible solutions.

2.4.4 Using Multiobjective Optimization to Handle Constraints

Penalty methods are also a popular approach in evolutionary optimization but suffer from the same problems as traditional methods when selecting reliable penalty parameters [Coello Coello, 2002]. Deb [2000] considers the superiority of feasible solutions to develop a penalty approach without the need to select penalty parameters. In a similar manner to the filter methods, other approaches avoid the selection of penalty parameters by treating the satisfaction of constraints as a separate goal. By treating the constrained problem explicitly as a multiobjective one, solutions are found that simultaneously satisfy constraints and offer objective improvement. This allows feasible solutions to be approached in a different manner and from different directions which would otherwise be limited by a penalty aggregated design space.

A taxonomy of different approaches is provided by Mezura-Montes and Coello Coello [2008], categorizing constrained optimization via multiobjective optimization as two groups. The first group transforms the constrained optimization into a biobjective problem, one for the objective and the other for a sum of constraint violations. An approach taken by Surry and Radcliffe [1997] forms a biobjective problem using the unconstrained objective function and an objective based on constraint violation ranking. Solutions can then be selected based on the level of constraint violation or objective fitness. The drawback of this method is that it introduces extra parameters to make this choice. More popular approaches use an individual objective for each constraint [Coello Coello, 2000a]. Here each population is evolved with the goal of finding a feasible solution with respect to a single constraint. The populations are then combined to find a solution that satisfies all the constraints but introducing separate populations for each constraint can become difficult to handle as the number of constraints increase. Angantyr et al. [2003] use the idea of Pareto ranking to rank solutions depending on their feasibility. The solution is also given a rank depending on its objective function value and a new objective fitness is assigned given by a weighted sum of these rankings. A number of other approaches exist which use multiobjective approaches in different ways, see Coello Coello [2000b], Ray et al. [2000] and Oyama et al. [2005]. Since it is very difficult for one approach to perform well on all problems, Mallipeddi and Suganthan [2010] conducted a study to measure the performance of an approach that uses a combination of superiority of feasible solutions, dynamic penalty functions, the ϵ-constraint method and multiobjective constraint handling.

2.5 Using Approximations in Optimization

Many approaches in optimization take advantage of approximations. We have already touched on Newton's method that approximates local regions of an objective function using a Taylor series expansion. This approach uses a quadratic model of the objective to find suitable search directions and then takes a step in this direction. This idea is fundamental in many gradient-based searches, improving convergence but does not adequately accelerate the optimization process when dealing with expensive analysis. In such cases, more general approximation methods have been developed with the aim of reducing the number of direct evaluations of the expensive analysis. These methods build an approximation $\hat{f}(\boldsymbol{x})$ of the expensive analysis $f(\boldsymbol{x})$ that is cheaper to evaluate and subsequently used in the optimization process. This type of approximation is commonly referred to as a surrogate, meta or response surface model and can be used in conjunction with a variety of local and global search algorithms.

2.5.1 Trust Regions

The trust region approach first replaces the objective function $f(\boldsymbol{x})$ with a surrogate $\hat{f}(\boldsymbol{x})$. This may take the form of a second order polynomial similar to Newton's method but a key advantage of the trust region method is the ability to incorporate more general approximations. Unlike many local search methods, the chosen step length d is limited to some trust region radius δ_i. This radius defines the region of the surrogate that can be trusted and adaptively increases or decreases as the search progresses (depending on how well the surrogate $\hat{f}(\boldsymbol{x})$ predicts the true objective $f(\boldsymbol{x})$). To find a suitable step $\boldsymbol{s}_i$, the trust region approach minimizes the sub problem $\hat{f}(\boldsymbol{x}_i + \boldsymbol{s}_i)$ where $\|\boldsymbol{s}_i\| \leq \delta_i$. Note the minimization of the surrogate rather than the original objective function. This can be performed cheaply using any existing optimization routine such as BFGS for example. Once a step $\boldsymbol{s}_i$ is found, $f(\boldsymbol{x}_i + \boldsymbol{s}_i)$ is evaluated. Depending on how well the surrogate has performed a decision is made to either except the step or to compute a new step based on a new trust region radius. If the surrogate performs poorly, the trust region radius may be reduced. If the surrogate performs well, leading to a reduced objective value, the step is accepted. If the step is accepted the surrogate is rebuilt based on the new iterate and the size of the trust region is either kept the same or increased.

The trust region approach can be extended to constrained problems building a surrogate for the objective and each constraint. A simple approach minimizes $\hat{f}(\boldsymbol{x}_i + \boldsymbol{s}_i)$ subject to the inequality constraints $\hat{g}(\boldsymbol{x}_i + \boldsymbol{s}_i)$ and equality constraints $\hat{h}(\boldsymbol{x}_i + \boldsymbol{s}_i)$. This can be solved readily using a constrained optimizer such as SQP. Fletcher and Leyffer [2002] combined trust regions with the filter method to better handle constraints.

By finding $\boldsymbol{s}_i$ based on minimising the surrogate model, the number of function evaluations using the original objective can be reduced. This can offer time savings over traditional methods if the time taken to evaluate an objective function is significant. As already mentioned, another advantage of the trust region method is the ability to incorporate more general approximations. This can be achieved without sacrificing convergence properties as long as the surrogate matches the value and the gradient of the objective function at $\boldsymbol{x}_i$ [Alexandrov et al., 1998]. This makes it possible to replace local quadratic models with global approximation techniques including higher order polynomials, radial basis functions and Gaussian process based models such as Kriging.

2.5.2 Surrogate-Assisted Evolutionary Algorithms

Evolutionary algorithms are reliable for global optimization but they are methods that require a very large number of function evaluations. This characteristic is not a problem for objective functions that are inexpensive to compute but becomes an issue when using expensive analysis. In such cases, evaluating the objective function a large number of times is simply too time consuming for evolutionary algorithms to be used directly.

An intuitive approach to incorporate surrogates in evolutionary algorithms is to directly replace the expensive analysis using a surrogate model built using a limited number of designs evaluated using the expensive analysis. The original EA can then be used cheaply by evaluating the surrogate rather than directly evaluating the expensive analysis. The performance of such an approach depends heavily on the accuracy of the surrogate being used. Ratle [1998] uses Kriging interpolation in this way to reduce the expense of a GA. The surrogate is updated after a number of generations to help improve the accuracy of the Kriging prediction. Using a limited number of function evaluations reduces the cost of the optimization but it becomes very difficult to build surrogates that are globally accurate enough to converge towards reliable solutions. It is possible to improve the accuracy of the surrogate by using more training data but this can quickly become impractical. In fact, for high dimensional problems the curse of dimensionality can make the cost of building a globally accurate surrogate more time consuming than applying an EA directly.

To overcome these difficulties, it is common to use the surrogate to assist the evolutionary process rather than guide it directly. This is achieved by using evaluations of the surrogate together with evaluations of the original objective function. Jin et al. [2002] introduced the concept of evolution control to help prevent the search from being misled by poor surrogate models. Evolution control evaluates a certain number of individuals in each generation using the original objective function. If an entire generation is evaluated using the original objective, this becomes a controlled generation. An empirical criterion is used to decide the frequency of evolution control and evaluated individuals are used to update a neural-network based surrogate model. Ong et al. [2004] investigate different approaches to circumvent the curse of dimensionality when building global surrogates. One approach uses local surrogates to assist a hybrid GA. Here, every individual is treated as a starting point for a locally converging search, performed using a trust region framework. The results of each local search are evaluated using the expensive analysis and provided they are better than the starting point, they are added to the population and the usual genetic operators applied. An alternative approach uses coevolution, decomposing the problem into subsets. Each subset is treated as different species that can be evolved cooperatively or in competition with one another. By dividing design variables among different species, the dimensionality of each subset and each surrogate is therefore reduced.

Emmerich [2005] pre-screens each population using surrogates based on Gaussian processes. Only the most promising individuals are then evaluated using the original objective function. In this study the most promising individuals are those that offer greatest fitness and/or offer improvement to the surrogate model. These individuals can be identified by making use of model error estimates available when using Gaussian processes. This pre-screening procedure can be generalised to constrained and multiobjective problems and is analogous to the improvement criteria considered in the next section.

2.6 Efficient Global Optimization

In lieu of dealing with expensive analysis, we have visited the trust region approach that can be combined with both local and global surrogates to converge toward optimal solutions. However, for consistency conditions to be met, gradient information is required. For global optimization and in the absence of any gradient information, a surrogate-assisted EA may be better suited. Rather than applying an EA directly, using surrogates can significantly reduce the number of expensive evaluations required but this often remains impractical when using time consuming analysis that is in the order of hours or days.

Here we visit an alternative approach that uses statistical methods to sequentially update global surrogates and efficiently converge towards optimal solutions. A basic strategy is outlined in Figure 2.6. This begins prior to having any knowledge of the design space by making use of an initial sample. The surrogate model is then built based on evaluations of this initial sample using the expensive analysis. Next, the surrogate is searched inexpensively to find an update point that offers some improvement over all previously sampled points. This update point is evaluated using the expensive analysis and the surrogate is rebuilt based on the new set of evaluated designs. This process is repeated until a time limit, evaluation budget or model accuracy is reached.

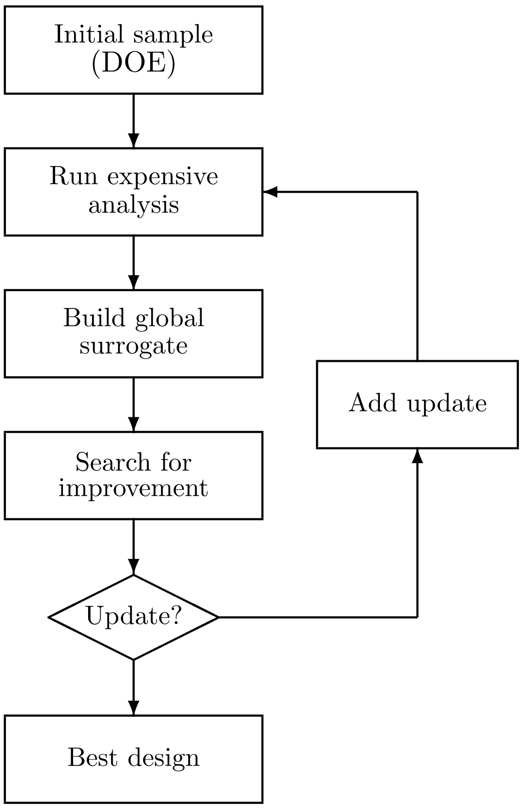


FIGURE 2.6: Basic strategy for optimization using global surrogates.

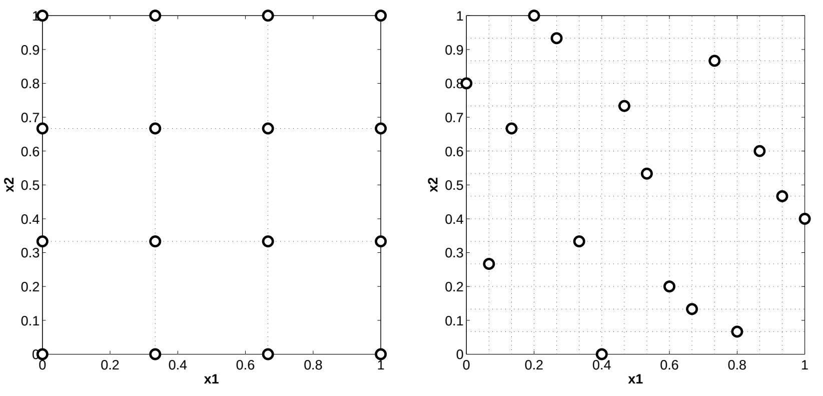
To distinguish this approach from the other methods discussed, we refer to this as surrogate-based rather than surrogate-assisted optimization. The efficiency and reliability of surrogate-based optimization relies heavily on the choice of initial sample, the choice of surrogate and the improvement criterion (also known as the infill sampling criterion) used to select new designs. The remainder of this chapter discusses these key elements in more detail.

2.6.1 Initial Sampling

It is common in surrogate-based optimization to base an initial sample on some design of experiments (DOE) technique. The correct choice of initial sample will depend on the type of problem being tackled, the choice of surrogate model being used and the total budget of evaluations. Perhaps most significantly, the size of the initial sample will depend on the dimensionality of the design space being modelled. Say for example, we are happy with a surrogate that approximates a one dimensional function using ten sample locations, to achieve the same sample density on a four dimensional problem would require an initial sample based on 10,000 evaluations. This type of sampling plan is regarded as a 10^4 full factorial and is clearly impractical when using expensive analysis.

When using surrogates in optimization, this curse of dimensionality makes it imperative that a designer aims to reduce the number of design variables being optimized. Pre-screening the design problem and removing any inactive design variables will significantly reduce the cost of building suitable surrogate models. Here we assume that any pre-screening has been performed and the designer is left with design variables that all have a noticeable effect on the objective.

Full factorial sampling plans result in uniform sampling with good spatial dispersion, giving a clear indication of any correlation between design variables and their response over the whole design space. A major drawback is that this requires sampling plans of distinct sizes, often leading to an impractical number of evaluations. One way to provide a more flexible sample is to stratify the design space in all dimensions. This essentially splits the design space into a set of rows and columns, each containing a given number of samples. This allows each dimension to be sampled at more levels without a significant increase in the number of sample points. This is illustrated in Figure 2.7 where the 4^2 full factorial samples each design variable on four levels. In contrast, each sample point in the random Latin hypercube sampling plan has a unique projection, sampling each design variable on 16 levels.



(a) (b)

FIGURE 2.7: Sampling plans based on 16 samples. (a) 4^2 full factorial. (b) A random Latin hypercube.

Any combination of points in a random Latin hypercube will result in unique projections but is not guaranteed to give a sampling plan with a good spatial dispersion. A bad Latin hypercube design may fail to capture important features in the design space and lead to surrogate models with poor global accuracy. One way to improve the space filling properties of a Latin hypercube is by optimizing the Morris and Mitchell criterion [Morris and Mitchell, 1995]. This essentially maximizes the minimum distance between each pair of points, see Figure 2.8. Although many other types of sampling plans exist, here we limit ourselves to using Latin hypercubes. They are favoured in this thesis as they can be computed relatively quickly and are independent of the surrogate model being used. They are also one of the most flexible methods and can be based on any number of sample points, useful for designing initial samples around a limited budget of design evaluations. For more detail on Latin hypercube sampling see Mackay et al. [1979] and for a comprehensive survey of different sampling plans suitable for computer experiments see Simpson et al. [2001].

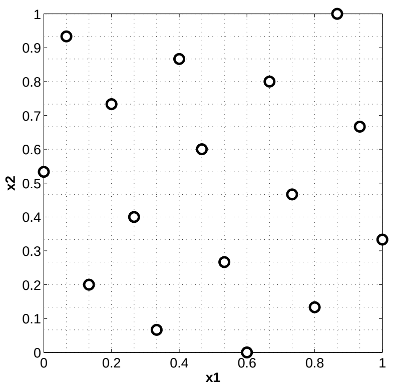


FIGURE 2.8: Optimized Latin hypercube.

2.6.2 Overview of Kriging

Once an initial sample is selected and designs evaluated, the next step is to build a surrogate model. A wide choice of surrogate models is available to the designer, accompanied by a large field of literature. An overview of the most popular methods used in surrogate-based optimization is provided by Jones [2001] and more recently by Forrester and Keane [2009]. Some of the more popular methods include polynomial models, radial basis functions and support vector regression, each method having their own merits and weaknesses. Generally speaking, polynomial models offer simplicity when modelling low dimensional problems but their weakness lies with high dimensional problems and they offer limited flexibility when dealing with multimodal landscapes. These problems are better dealt with using radial basis functions which have a level of flexibility that can be easily controlled. Radial basis functions are fundamental to other methods such as neural networks and Kriging, both offering increased flexibility but at the cost of increased complexity and higher model training costs. Here we limit ourselves to a more detailed discussion of Kriging.

Kriging is one method frequently covered in the literature and has been applied to a variety of engineering design problems, including aerodynamic, [Keane, 2002, Yoneta et al., 2010], structural [Booker et al., 1999, Sakata et al., 2003], multidisciplinary [Simpson, 2001] and multiobjective design problems [Keane, 2006, Jeong and Obayashi, 2005].

The origins of Kriging lies in the field of geophysics [Krige, 1951] and was introduced into the field of engineering by Sacks et al. [1989]. Kriging is a Gaussian process based model and has been popularised because of its great flexibility combined with the ability to make estimates of model uncertainty, a characteristic useful in choosing model update points.

The following overview of Kriging is based on the introduction provided by Jones [2001] but other publications useful to the interested reader include Forrester et al. [2008, 2006] and of course Sacks et al. [1989].

To begin let the function prediction, $\hat{f}(\boldsymbol{x})$, be used as a surrogate to the expensive function $f(\boldsymbol{x})$. This surrogate is built using a set of inputs $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)}$ and known outputs $\boldsymbol{y} = \{y^{(1)}, y^{(2)}, \ldots, y^{(n)}\}^{\mathrm{T}}$ found by evaluating the true function $f(\boldsymbol{x})$.

Before demonstrating how this prediction is achieved using Kriging it is necessary to view the function y as a realization of a Gaussian process. The Gaussian process is specified by a mean and covariance function and is essentially a collection of random variables each with a normal distribution. The value of y at $\boldsymbol{x}$ can be considered as a realization of a random variable $Y(\boldsymbol{x})$. This introduces the idea of uncertainty representing the fact that we do not know the true output at most sets of inputs. Different interpretations of Kriging make different assumptions about the Gaussian process. Here we assume use a mean function that is constant in the manner as Ordinary Kriging.

Assuming that the function we wish to approximate is smooth and continuous, two points $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$ are close if $||\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}||$ is small and then $Y(\boldsymbol{x}^{(i)})$ and $Y(\boldsymbol{x}^{(j)})$ are likely to be highly correlated. As $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$ move apart the opposite is true. This is encapsulated in the Kriging correlation function,

$$\mathrm{Corr}\left[Y(\boldsymbol{x}^{(i)}), Y(\boldsymbol{x}^{(j)})\right] = \exp\left(\sum_{l=1}^{d} \theta_l |\boldsymbol{x}_l^{(i)} - \boldsymbol{x}_l^{(j)}|^{p_l}\right). \tag{2.4}$$

Here we use a power exponential correlation function but many alternatives exists, see Santner et al. [2003]. Note that the correlation between $Y(\boldsymbol{x}^{(i)})$ and $Y(\boldsymbol{x}^{(j)})$ depends only on the sample locations and the parameters θ_l and p_l. If $\boldsymbol{x}^{(i)} = \boldsymbol{x}^{(j)}$ then the correlation is 1 and converges to zero as $||\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}|| \rightarrow \infty$. The parameters θ_l and p_l are known as the hyperparameters, which must be estimated.

From this we can construct a correlation matrix for all the observed data,

$$\boldsymbol{\Psi} = \begin{pmatrix} \mathrm{Corr}\left[Y(\boldsymbol{x}^{(1)}), Y(\boldsymbol{x}^{(1)})\right] & \ldots & \mathrm{Corr}\left[Y(\boldsymbol{x}^{(1)}), Y(\boldsymbol{x}^{(n)})\right] \\ \vdots & \ddots & \vdots \\ \mathrm{Corr}\left[Y(\boldsymbol{x}^{(n)}), Y(\boldsymbol{x}^{(1)})\right] & \ldots & \mathrm{Corr}\left[Y(\boldsymbol{x}^{(n)}), Y(\boldsymbol{x}^{(n)})\right] \end{pmatrix}, \tag{2.5}$$

and the covariance matrix

$$\mathrm{Cov}\left(\boldsymbol{Y}, \boldsymbol{Y}\right) = \sigma^2 \boldsymbol{\Psi}. \tag{2.6}$$

The correlations described is our matrix $\boldsymbol{\Psi}$ depends on the distance between sample points and the hyperparameters $\boldsymbol{\theta} = \{\theta_1, \theta_2, \ldots, \theta_d\}^{\mathrm{T}}$ and $\boldsymbol{p} = \{p_1, p_2, \ldots, p_d\}^{\mathrm{T}}$. Broadly speaking, the values of $\boldsymbol{p}$ control the smoothness of correlation and $\boldsymbol{\theta}$ controls the extent of influence each variable has on the correlation. These parameters can be estimated using the maximum likelihood estimation. For a Gaussian process the likelihood can be written as

$$L(\mu, \sigma^2, \boldsymbol{\theta}, \boldsymbol{p}) = \frac{1}{(2\pi\sigma^2)^{\frac{n}{2}} |\boldsymbol{\Psi}|^{\frac{1}{2}}} \exp\left[-\frac{(\boldsymbol{y} - \mathbf{1}\mu)^{\mathrm{T}} \boldsymbol{\Psi}^{-1} (\boldsymbol{y} - \mathbf{1}\mu)}{2\sigma^2}\right]. \tag{2.7}$$

Maximizing the likelihood identifies parameters which model the function's behaviour consistently with the data seen. In practice it is simpler to maximize the natural log of the likelihood function. Ignoring constant terms, this may be written as

$$\ln(L) = -\frac{n}{2}\ln(\sigma^2) - \frac{1}{2}\ln(|\boldsymbol{\Psi}|) - \frac{(\boldsymbol{y} - \mathbf{1}\mu)^{\mathrm{T}} \boldsymbol{\Psi}^{-1} (\boldsymbol{y} - \mathbf{1}\mu)}{2\sigma^2}. \tag{2.8}$$

Expressions for the optimal values of μ and σ^2 can be found by taking the derivatives of equation (2.8) and setting to zero, yielding

$$\hat{\mu} = \frac{\mathbf{1}^{\mathrm{T}} \boldsymbol{\Psi}^{-1} \boldsymbol{y}}{\mathbf{1}^{\mathrm{T}} \boldsymbol{\Psi}^{-1} \mathbf{1}}, \tag{2.9}$$

$$\hat{\sigma}^2 = \frac{(\boldsymbol{y} - \mathbf{1}\hat{\mu})^{\mathrm{T}} \boldsymbol{\Psi}^{-1} (\boldsymbol{y} - \mathbf{1}\hat{\mu})}{n}. \tag{2.10}$$

Substituting equations (2.9) and (2.10) into equation (2.8) gives the concentrated ln-likelihood function,

$$\ln(L) \approx -\frac{n}{2}\ln(\hat{\sigma}^2) - \frac{1}{2}\ln(|\boldsymbol{\Psi}|). \tag{2.11}$$

The value of this function depends only on $\boldsymbol{\Psi}$, and hence, on the hyperparameters $\boldsymbol{\theta}$ and $\boldsymbol{p}$. Finding optimal values for $\boldsymbol{\theta}$ and $\boldsymbol{p}$ is not simple and cannot be achieved through differentiation. This problem requires a direct numerical global optimization and needs treating with care if good results are to be obtained.

In order to make a prediction $\hat{y}$ at some new point $\boldsymbol{x}$, it is also necessary to maximize the likelihood of this prediction. Suppose $\hat{y}$ is an estimated function value, the quality of this estimate can be evaluated by adding $\hat{y}$ to the observed data, $\tilde{\boldsymbol{y}} = \{\boldsymbol{y}, \hat{y}\}^{\mathrm{T}}$, and computing the augmented likelihood function.

The augmented correlation matrix is given as

$$\tilde{\boldsymbol{\Psi}} = \begin{pmatrix} \boldsymbol{\Psi} & \boldsymbol{\psi} \\ \boldsymbol{\psi}^{\mathrm{T}} & 1 \end{pmatrix}, \tag{2.12}$$

where $\boldsymbol{\psi}$ is the vector of correlation between the observed data and the prediction

$$\boldsymbol{\psi} = \begin{pmatrix} \mathrm{Corr}\left[Y(\boldsymbol{x}^{(1)}), Y(\boldsymbol{x})\right] \\ \vdots \\ \mathrm{Corr}\left[Y(\boldsymbol{x}^{(n)}), Y(\boldsymbol{x})\right] \end{pmatrix}. \tag{2.13}$$

Using the optimum parameter values obtained, the augmented likelihood reflects how consistent the estimate is with the observed pattern of variation. The best value for this estimate is therefore the value of $\hat{y}$ that maximizes the augmented likelihood function. Recalling that $\boldsymbol{\psi}$ contains our new sample point $\boldsymbol{x}$, the Kriging predictor can be expressed as

$$\hat{y}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{\psi}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}(\boldsymbol{y} - \mathbf{1}\hat{\mu}). \tag{2.14}$$

This predictor is for an interpolating model. In the presence of noise this procedure can be filtered by including a regression constant λ. The regression constant λ is added to the leading diagonal of $\boldsymbol{\Psi}$ producing $\boldsymbol{\Psi} + \lambda\boldsymbol{I}$, where $\boldsymbol{I}$ is an identity matrix. Using the same derivation as before, the regressing Kriging prediction is given by

$$\hat{y}_r(\boldsymbol{x}) = \hat{\mu}_r + \boldsymbol{\psi}^{\mathrm{T}}(\boldsymbol{\Psi} + \lambda\boldsymbol{I})^{-1}(\boldsymbol{y} - \mathbf{1}\hat{\mu}_r), \tag{2.15}$$

where

$$\hat{\mu}_r = \frac{\mathbf{1}^{\mathrm{T}}(\boldsymbol{\Psi} + \lambda\boldsymbol{I})^{-1}\boldsymbol{y}}{\mathbf{1}^{\mathrm{T}}(\boldsymbol{\Psi} + \lambda\boldsymbol{I})^{-1}\mathbf{1}}. \tag{2.16}$$

$$\hat{\sigma}_r^2 = \frac{(\boldsymbol{y} - \mathbf{1}\hat{\mu}_r)^{\mathrm{T}}(\boldsymbol{\Psi} + \lambda \boldsymbol{I})^{-1}(\boldsymbol{y} - \mathbf{1}\hat{\mu}_r)}{n}. \tag{2.17}$$

A suitable regression constant λ can be found in the same manner as the other model parameters, using maximum likelihood estimation.

2.6.3 Improvement Criteria

A key process in surrogate-based optimization is the selection of model update points for model refinement. These updates can be selected in a number of ways. The most straightforward improvement criterion considers the addition of an update point at the current optimum predicted by the surrogate model. Figure 2.9 shows an example of updating a surrogate based on the Kriging prediction. This is pure exploitation of the approximation and with further updates can quickly converge upon an optimal solution. Unless the surrogate model is globally accurate, this method is not guaranteed to converge towards the global optimum. In Figure 2.9(b) we see that adding an update at the minimum of the Kriging prediction converges towards the local optimum.

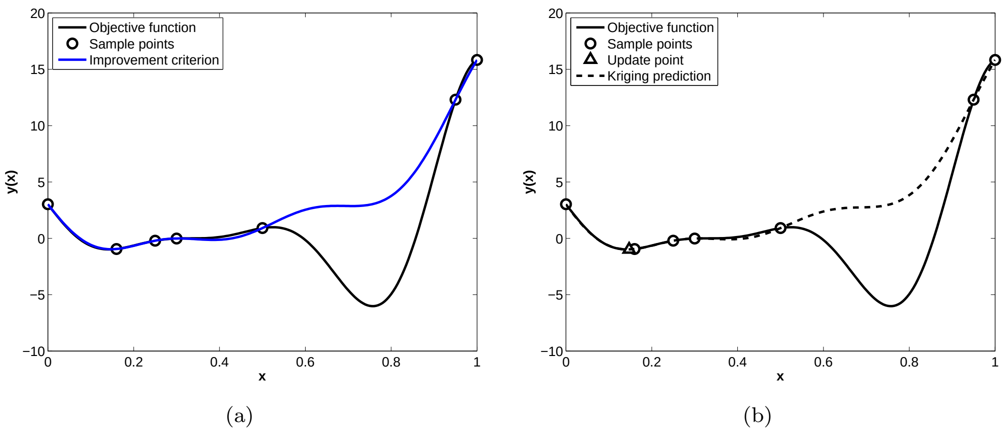


Figure 2.9: Kriging surrogate of the Forrester function. (a) Kriging prediction as an improvement criterion. (b) Updated Kriging prediction.

More reliable methods attempt to combine elements of both objective improvement and global exploration of the design space. The improvement criterion should therefore aim to strike a balance between exploitation and exploration of the objective approximation. Using surrogate models based on Kriging allows the prediction of the mean response (i.e. the Kriging prediction) $\hat{y}$ at $\boldsymbol{x}$ but also the variance in this prediction. Assuming an interpolating model, this is estimated as (see Sacks et al. [1989]),

$$\hat{s}^2(\boldsymbol{x}) = \hat{\sigma}^2 \left[1 - \boldsymbol{\psi}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}\boldsymbol{\psi} + \frac{(1 - \mathbf{1}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}\boldsymbol{\psi})^2}{\mathbf{1}^{\mathrm{T}}\boldsymbol{\Psi}^{-1}\mathbf{1}} \right]. \tag{2.18}$$

It is intuitive that the predicted variance will be equal to zero at previously sampled points[2] and larger in regions that are poorly sampled. This feature of Gaussian process based models is useful in the selection of model update points, where the improvement criteria can account for those areas with a high predicted variance. Figure 2.10 demonstrates the Kriging prediction and the 95% confidence intervals for this prediction.

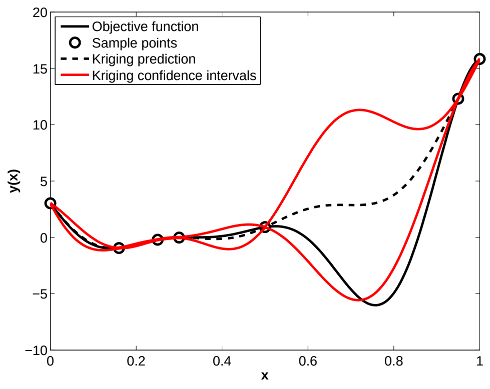


FIGURE 2.10: Kriging prediction of the Forrester function and the predicted error in the prediction.

Sasena et al. [2002] and Jones [2001] both provide comparisons of different improvement criteria. This includes popular approaches used in engineering design and others previously used in the geology literature, such as those used by Watson and Barnes [1995]. It is clear from these two studies that the most efficient approaches utilize this predicted variance. Other attempts at developing efficient improvement criteria, for example Martin and Simpson [2002] and Regis and Shoemaker [2005], offer little in the way of improvement over the methods suggested by Jones [2001].

Sampling in regions of high variance will lead to pure exploration of the surrogate model, simply filling in the gaps between previously sampled points. Although this can be expected to converge towards the global optimum, this may take a long time to do so. Instead a popular approach is to search the surrogate for maximum probability of improvement, an idea originally introduced by Kushner [1964]. This is given by integrating over the probability density function (PDF) below the minimum point sampled so far, y_{min}. Making use of the fact that $y(\boldsymbol{x})$ is treated as the realization of a random function and is normally distributed with $\mathcal{N}\left(\hat{y}, \hat{s}^2\right)$, dropping the dependencies on $\boldsymbol{x}$ for notation simplicity, the probability of improvement can thus be expressed as,

$$P[y < y_{min}] = \int_{-\infty}^{y_{min}} \boldsymbol{\phi}\left(y\right) \mathrm{d}y, \tag{2.19}$$

where $\boldsymbol{\phi}\left(y\right)$ is the PDF representing uncertainty about $y(\boldsymbol{x})$ given by,

[2]This requires re-interpolation of $\hat{\sigma}_r^2$ when using regressing Kriging, see Forrester et al. [2008].

$$\boldsymbol{\phi}(y) = \frac{1}{\hat{s}\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{y-\hat{y}}{\hat{s}}\right)^2\right]. \tag{2.20}$$

The probability of improvement can be expressed analytically as

$$P[y < y_{min}] = \boldsymbol{\Phi}\left(\frac{y_{min} - \hat{y}}{\hat{s}}\right), \tag{2.21}$$

where $\boldsymbol{\Phi}(.)$ is the cumulative distribution function (CDF) for a standard normal distribution.

The probability of improvement will return a value between zero and one, allowing regions of improvement to be identified. Figure 2.11 updates the surrogate at the point of maximum probability of improvement. Potential improvement is indicated in the unsampled region around the global optimum but in this case, the probability of improvement is largest in the region of the local optimum. Although this does not identify the global optimum after a single update, after further updates the probability of improvement will reduce in increasingly sampled regions and the search is forced to explore, eventually converging on the global optimum.

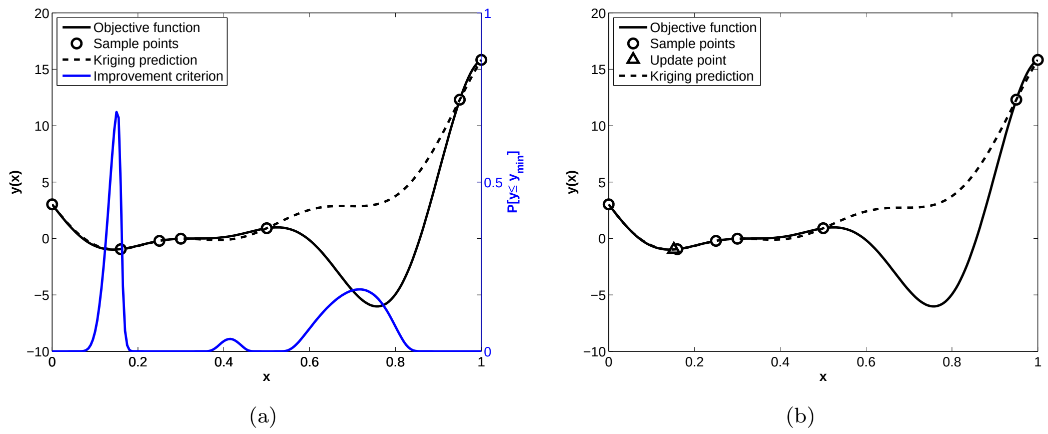


FIGURE 2.11: Kriging surrogate of the Forrester function. (a) Probability of improvement as an improvement criterion. (b) Updated Kriging prediction.

When using the probability of improvement, it is possible that different regions of the design space will indicate an equivalent improvement. This can lead to several plateaus in the landscape of the improvement criterion, making it difficult to pin point the best location for the next update. A better metric considers the magnitude of the improvement. This can be achieved by defining the improvement as $I(\boldsymbol{x}) = y_{min} - y$, and computing the expected improvement given as

$$E[I(\boldsymbol{x})] = \int_{-\infty}^{y_{min}} (y_{min} - y)\,\boldsymbol{\phi}(y)\,\mathrm{d}y. \tag{2.22}$$

This can be expressed analytically as

$$E[I(\boldsymbol{x})] = (y_{min} - \hat{y})\boldsymbol{\Phi}\left(\frac{y_{min} - \hat{y}}{\hat{s}}\right) + \hat{s}\boldsymbol{\phi}\left(\frac{y_{min} - \hat{y}}{\hat{s}}\right). \tag{2.23}$$

where $\boldsymbol{\phi}(.)$ and $\boldsymbol{\Phi}(.)$ are the PDF and CDF for a standard normal distribution.

Figure 2.12 computes the next update point based on the maximum expected improvement. This better balances the exploitation and exploration of the Kriging prediction and the expected improvement is larger in the region of the global optimum. Maximizing the expected improvement has been used consistently throughout the literature and is fundamental in the popular efficient global optimization (EGO) algorithm [Jones et al., 1998].

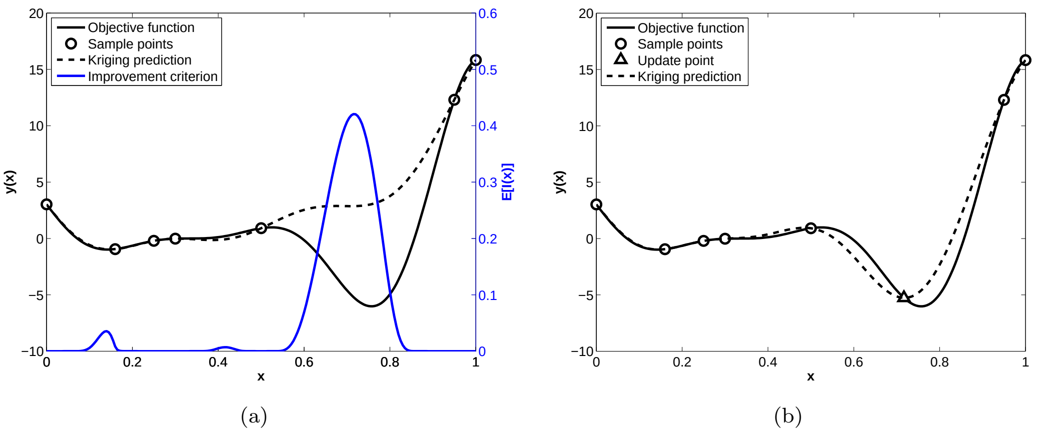


FIGURE 2.12: Kriging surrogate of the Forrester function. (a) Expected improvement as an improvement criterion. (b) Updated Kriging prediction.

Under certain conditions this concept will guarantee global convergence since an unsampled point will always indicate a predicted variance greater than zero. This results in a positive value of expected improvement at all unsampled points even if the function is poorly modelled. For global optimization, using the expected improvement is advantageous since it is likely to be larger at under sampled areas near to the global optimum. Vazquez and Bect [2010] provide properties of convergence whilst Bull [2011] provides some theoretical results for convergence rates of EGO. While it is reasonable to suggest that theoretical properties of algorithms based on Gaussian processes are less well understood than traditional methods, they perform well in practice and very useful in many engineering design problems.

Chapter 3

Surrogate-Based Constrained Optimization

There has been extensive research into the use of different surrogates and efficient improvement criteria over the last two decades but there has been relatively little emphasis on dealing with constraints considering how often they arise in practical problems. This chapter reviews some basic improvement criteria that can be used to handle constraints in surrogate-based optimization, identifying strengths and weaknesses of these basic approaches.

3.1 Handling Constraints

Dealing with constraints can significantly increase the complexity and the expense of optimization depending of their severity and the number of constraints that need to be satisfied. The designer should always aim to remove any inactive constraints and reduce the number of constraints by construction where possible. If the constraints are cheap to evaluate directly, it is possible to formulate an improvement criterion based only on feasible solutions and the global optimum can be found at little extra cost. If the constraints are expensive to evaluate they also need to be approximated using surrogates and selecting suitable update points becomes less straightforward. In this study this is assumed to be the case and Kriging is used to model the objective and each individual constraint. The improvement criterion is now required to evaluate the objective and all the constraint approximations to find updates that refine the surrogate models in regions that converge towards a feasible optimum. It is possible to find update points by searching the objective and constraint surrogates inexpensively using some constrained optimization algorithm. This approach is taken by Sasena [2002] in his implementation of SuperEGO. An alternative is to modify the improvement criterion to include information about both objective improvement and constraint feasibility. This transforms the search for an update point into an unconstrained one and the improvement criterion can be optimized using any global search. When transforming the problem it is important the improvement criterion aims to strike a balance between exploitation and exploration of the objective and all the constraint approximations for good feasible design to be found.

In this thesis the improvement criteria are modified to deal with only inequality constraints. Figure 3.1 shows an example of the Forrester function subject to an inequality constraint. For the constraint to be satisfied the value of the constraint has to be greater than or equal to the constraint limit, leading to distinct feasible and infeasible regions. It is possible to use the methods discussed in this chapter to also handle equality constraints by treating each equality as two tightly bound inequality constraints, a common approach in engineering design [Sasena, 2002].

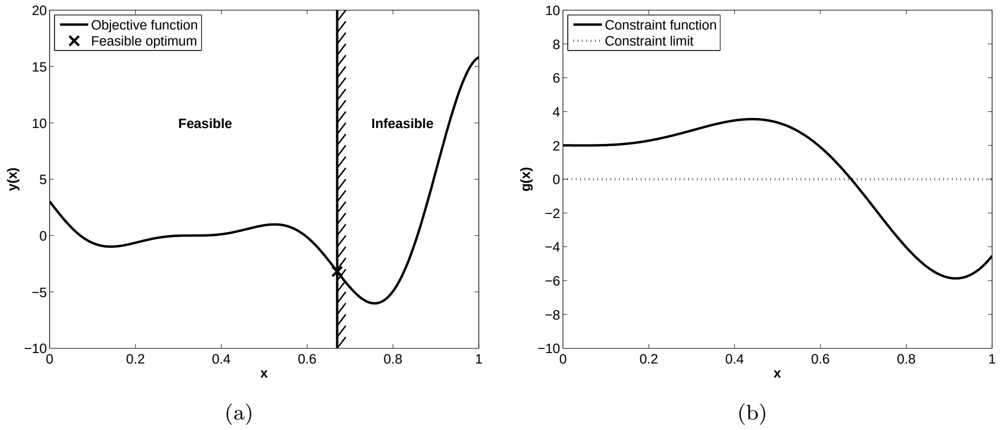


FIGURE 3.1: Forrester function subject to an inequality constraint. (a) Feasible and infeasible regions when $g(\boldsymbol{x}) \geq 0$. (b) Constraint function and constraint limit.

3.1.1 Function Prediction with One Pass Penalty

The most straightforward improvement criterion considers the addition of an update point at the current optimum predicted by the surrogate model. This was demonstrated in the previous chapter on a simple unconstrained problem, see Figure 2.9. This is pure exploitation of the model prediction and with further updates can quickly converge upon an optimal solution. This method is modified to deal with constraints by transforming the constrained problem into an unconstrained one. This is achieved most simply by the addition of a one pass penalty function to the Kriging prediction,

$$\hat{y}(\boldsymbol{x})_P = \hat{y}(\boldsymbol{x}) + P.$$

This penalty is included if any one of the constraint surrogates predict a violation. Selecting updates based on this modified improvement criteria is illustrated in Figure 3.2(d).

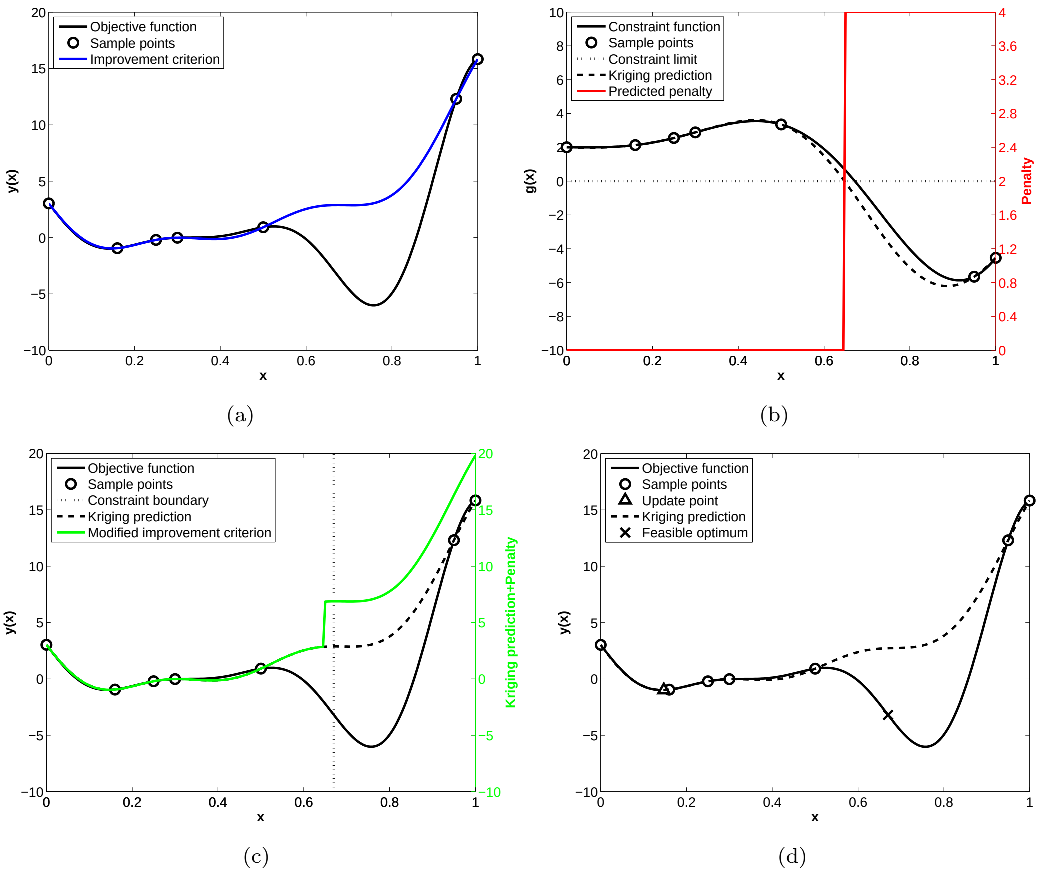


FIGURE 3.2: Constrained Forrester function. (a) Kriging prediction of the objective. (b) Kriging prediction of the constraint and the predicted penalty. (c) Modified improvement criterion. (d) Updated objective prediction.

This creates a landscape with a steep cliff marking the edge of the feasible design space as predicted by the constraint surrogate. On simple problems this method will perform well, however, with no element of exploration of poorly modelled regions, details of the design space can be missed. As seen already, the lack of exploration of the design space makes it unlikely that we will converge towards a global optimum, whilst the abrupt cliff caused by the penalty approach also makes this method vulnerable to deceptive or poorly modelled constraint functions. Figure 3.2(c) illustrates the uncertainty in the constraint prediction leading to a penalty that falsely predicts the constraint boundary. For a reduced notation, this improvement criterion is denoted as YP for the remainder of this chapter.

3.1.2 Expected Improvement with One Pass Penalty

Using Kriging, or an alternative Gaussian process based model, permits the estimation of model uncertainty. This feature is useful for the selection of update points where the improvement criterion can account for those areas of the model with high uncertainty, thus adding an element of exploration. For constrained problems, the improvement criterion should seek improvement over the current feasible minimum giving the expected improvement as,

$$E[I(\boldsymbol{x})] = (y_{minfeas} - \hat{y})\boldsymbol{\Phi}\left(\frac{y_{minfeas} - \hat{y}}{\hat{s}}\right) + \hat{s}\boldsymbol{\phi}\left(\frac{y_{minfeas} - \hat{y}}{\hat{s}}\right), \tag{3.1}$$

where $y_{minfeas}$ is the minimum feasible point sampled so far. For a poor initial sample or if the feasible region is small, a feasible point may not have been sampled. In such cases, the point that is closest to being feasible is used.

In order to deal with constraints, the search for maximum expected improvement can again be modified into an unconstrained problem via a one pass penalty function,

$$E[I(\boldsymbol{x})]_P = E[I(\boldsymbol{x})] \times P.$$

Here the penalty is equal to one for feasible solutions and zero if any constraint is violated. Selecting an update based on this modified improvement criterion is illustrated in Figure 3.3.

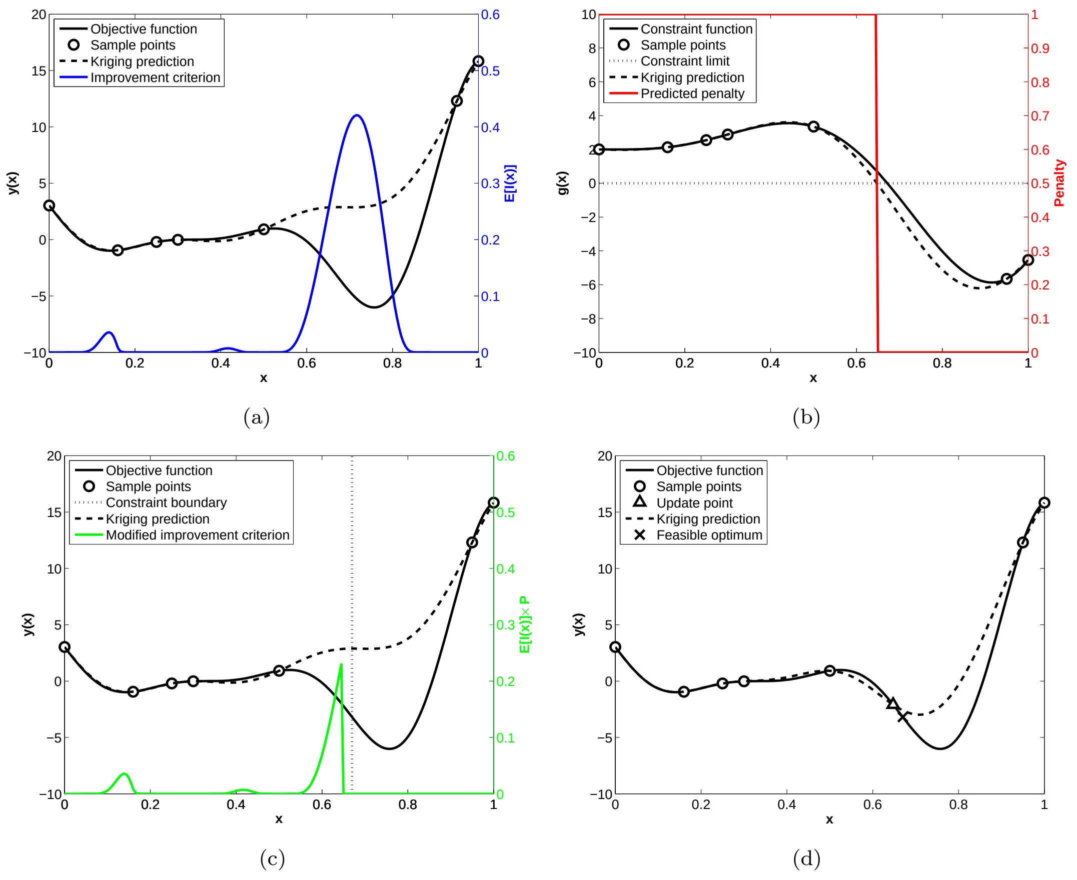


FIGURE 3.3: Constrained Forrester function. (a) Kriging prediction of the objective and the expected improvement. (b) Kriging prediction of the constraint and the predicted penalty. (c) Modified improvement criteria. (d) Updated objective prediction.

Utilizing expected improvement will encourage exploration of the predicted feasible regions of the objective. This works well in this problem and with further updates can be expected to converge towards an accurate feasible optimum. One concern with this approach is that the edges of the feasible regions are again defined by a sheer cliff, being deceptive when the constraints are poorly modelled, illustrated in Figure 3.3(c). For the remainder of this chapter this improvement criterion is denoted as EIP.

3.1.3 Expected Improvement with Probability of Feasibility

Using a one pass penalty function will limit updates to objective improvement only in the strictly feasible space. Although this is sensible for refinement of the objective approximation, this may curb progression of constraint approximations. This can become a problem, especially in cases where the constraints are poorly modelled. As already noted, an inaccurate constraint model can cause the penalty function to be missed or wrongly applied. One way to better handle the constraints is suggested by Schonlau [1997]. Assuming independence between the objective and constraint functions[1], this approach combines the expected improvement of the objective function and the probability of feasibility calculated from the constraint functions as a simple product,

$$E[I(\boldsymbol{x})]_{con} = E[I(\boldsymbol{x})] \times P[g \leq g_{limit}]. \tag{3.2}$$

The probability of feasibility $P[g \leq g_{limit}]$ is calculated in the same manner as probability of improvement but instead identifies regions of feasibility, where g_{limit} is the constraint limit. Maximizing the probability of feasibility can also be used to find an initial feasible point as pointed out by Sasena [2002]. If the constraint value has to be less than or equal to a constraint limit, the probability of feasibility for a single constraint is given as

$$P[g \leq g_{limit}] = \int_{-\infty}^{g_{limit}} \phi\,(g)\,\mathrm{d}g. \tag{3.3}$$

Alternatively, for a constraint value greater or equal to a constraint limit the probability of feasibility is expressed as

$$P[g \geq g_{limit}] = \int_{g_{limit}}^{\infty} \phi\,(g)\,\mathrm{d}g, \tag{3.4}$$

The probability of feasibility can be expressed analytically as

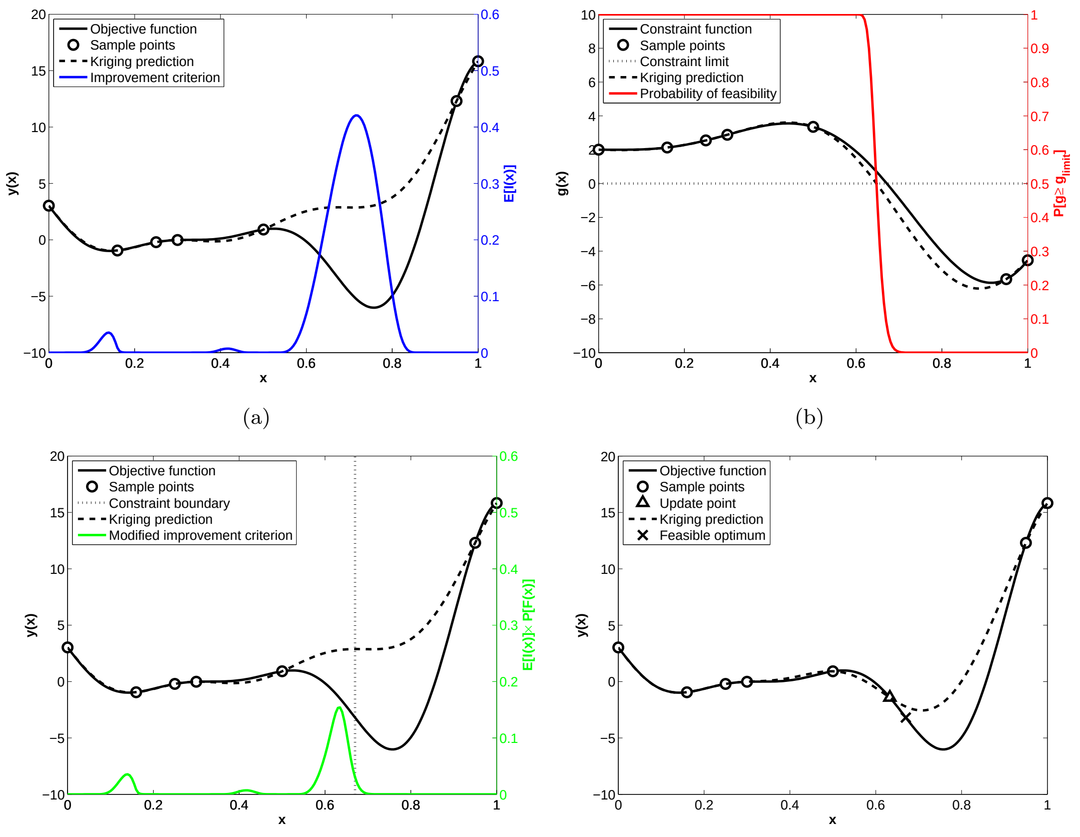
[1]This is a modelling assumption.

$$P[g \leq g_{limit}] = \mathbf{\Phi}\left(\frac{g_{limit} - \hat{g}}{\hat{s}}\right), \tag{3.5}$$

or

$$P[g \geq g_{limit}] = \mathbf{\Phi}\left(\frac{\hat{g} - g_{limit}}{\hat{s}}\right). \tag{3.6}$$

Combining the expected improvement and probability of feasibility in this way will gradually drive the improvement criteria towards zero in the transition between feasible and infeasible regions, adaptively softening the sheer cliff landscape produced by a one pass penalty function. This is illustrated in Figure 3.4(c) where there is no longer a sheer cliff in the landscape of the modified improvement criterion.



(a) (b)

(c) (d)

FIGURE 3.4: Constrained Forrester function. (a) Kriging prediction of the objective and the expected improvement. (b) Kriging prediction of the constraint and the probability of feasibility. (c) Modified improvement criteria. (d) Updated objective prediction.

In cases where more than one constraint is applied, the total probability of feasibility is a product of all individual constraint probabilities of feasibility. Forrester et al. [2008] refer to this method as constrained expected improvement. When dealing with complicated constraints or deceptive constraint functions, this improvement criterion helps to balance exploration and exploitation without the use of arbitrary penalty functions. Throughout this thesis this improvement criterion is denoted as EIPF.

3.2 Model Fitting and Search Algorithms

Surrogate-based optimization requires several sub optimizations at various stages. Firstly, an initial sample is required that adequately represents the design space. As already discussed it is common to use design of experiments and some search algorithm to optimize these designs. For the 2D test problems used here, random Latin hypercubes are used to encourage a diverse set of initial samples. For larger dimensional problems, such as the aircraft wing design problem in Section 3.6, a set of optimized Latin hypercube designs are used. It is then necessary to fit a surrogate model and find new updates to be evaluated. Both processes can be considered as sub problems that require further searches to be run.

3.2.1 Hyperparameter Tuning

Once an initial sample is selected and all designs evaluated, the next step is to fit the surrogate models. Fitting a Kriging model requires sufficient hyperparameter tuning. Different tuning strategies have been examined by Toal et al. [2008], concluding that a so called 'light tune' strategy is adequate up to 12 variables, provided enough true simulations are performed. The light tune consists of a hybrid search using a 1,000 evaluation GA and a 1,000 evaluation DHC to maximize the concentrated likelihood function. The light tune strategy has been adopted for hyperparameter tuning for all objective surrogates and all constraint surrogates studied here. The hyperparameters are tuned after the initial sample stage and retuned after every update.

3.2.2 Finding Updates

To find update points of interest it is necessary to search an improvement criterion. Some improvement criteria, such as expected improvement, can be highly multimodal and searching for the optimum update can, in itself, be a tough optimization problem. Fortunately the improvement criteria is usually cheap to compute and to ensure the best location for the next update is consistently found, a heavy hybrid search is used, consisting of a 10,000 evaluation GA and a 5,000 evaluation DHC. This search minimizes YP or maximizes EIP and EIPF to find the next update point.

3.3 Comparison Metrics

The performance of the improvement criteria can be assessed in a number of ways. Here this performance is measured using the true sample data, the known outputs $\boldsymbol{y} = \{y^{(1)}, y^{(2)}, \ldots, y^{(n)}\}^{\mathrm{T}}$ and the chosen set of inputs $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(n)}$, where n includes both the initial sample and all evaluated update points. If the true optimum is known, an intuitive method of comparison is to find the error between the true feasible optimum, y^* at $\boldsymbol{x}^*$, and the best feasible point sampled so far, y^{best} at $\boldsymbol{x}^{best}$. Assuming we are searching for a minimum, the error in the objective space is simply $y^{best} - y^*$.

The performance of each improvement criterion will depend on the placement of points in the initial sample. In some cases, by pure luck, this sample will include a point close to the global optimum, accelerating the search. For the artificial test problems, tests are repeated for a set of 100 random Latin hypercube initial samples to iron out any dependency on the initial sample. As the wing design problem is more time consuming, this is repeated on 50 different optimized Latin hypercube initial samples.

Given a limited evaluation budget the designer will also be concerned about the efficiency and reliability of each method. After repeating the search on a number of different initial samples, the performance of each improvement criterion can be represented in terms of a probability. This probability characterizes the consistency of each method in achieving an optimal solution to a prescribed accuracy. For example, an improvement criterion that finds an optimal solution within an accuracy of 0.01 of the true optimum, 90 times out of 100, will have a probability of 90% at this accuracy. This metric is computed after each update so the most efficient and reliable improvement criterion can be identified. By decreasing the required accuracy, this metric will demonstrate both the efficiency and reliability of each method in finding the region of the global optimum. Increasing the accuracy will demonstrate which methods can identify the exact global optimum.

To check statistical significance between the different methods, a statistical resampling method is used to compute a better representation of the average performance and 95% confidence intervals. Here, bootstrapping is used [Chernick, 2008].

3.4 Artificial Test Problems

Throughout this thesis, different improvement criteria are tested on three artificial test problems. In all cases the optimization strategy follows Figure 2.6. The process begins with the generation of an initial sample, produced using a random Latin hypercube. Based on this initial sample, the next step is to fit the Kriging models, tuning the hyperparemeters for the objective function and the constraint function(s). The chosen improvement criterion is then searched to find the first update point. The selected point is evaluated and the models are retuned after adding the new point to the set of samples. This process is repeated for a number of updates until the total number of evaluations reaches a specified budget. The size of this initial sample and the total number of updates has been catered individually for each problem. Note that all design variables are normalized to $[0, 1]$ to avoid any scaling issues when building and searching surrogate models.

The first two test problems minimize a modified version of the Branin function, found in Forrester et al. [2008]. This modification adapts the original Branin function to include one global optimum, rather than three optima of equal value. The modified Branin function is given by

$$f(\boldsymbol{x}) = \left(a_2 - \frac{5.1}{4\pi^2} + \frac{5}{\pi}a_1 - 6\right)^2 + 10\left[\left(1 - \frac{1}{8\pi}\right)\cos a_1 + 1\right] + \frac{5a_1 + 25}{15}, \tag{3.7}$$

where $a_1 = 15x_1 - 5$, $a_2 = 15x_2$ and $x_1, x_2 \in [0, 1]$. Test problem 1 is concerned with minimizing the Branin function subject to a simple inequality constraint,

$$g(\boldsymbol{x}) = x_1 x_2 \geq 0.2, \tag{3.8}$$

resulting in a global feasible optimum $y^* = 5.576$. In test problem 2 the complexity of the constraint in increased. The constraint function is a version of the Gomez#3 function [Sasena et al., 2002], with an additional sinewave to increase modality,

$$g(\boldsymbol{x}) = \left(4 - 2.1b_1^2 + \frac{1}{3}b_1^4\right)b_1^2 + b_1 b_2 + \left(-4 + 4b_2^2\right)b_2^2 + 3\sin\left[6\left(1 - b_1\right)\right] + 3\sin\left[6\left(1 - b_2\right)\right] \geq 6, \tag{3.9}$$

where $b_1 = 2x_1 - 1$ and $b_2 = 2x_2 - 1$, resulting in the global feasible optimum $y^* = 12.001$. The bounded feasible space and the location of the global optimum for both of these problems are presented in Figure 3.5.

To investigate the performance of these methods when dealing with multiple constraints, a third test problem is introduced. This deceptive test function, originally used by Sasena [2002], has two active and one inactive constraint. Here the inactive constraint is ignored and the minimization of the constrained problem is defined as,

$$f(\boldsymbol{x}) = -(x_1 - 1)^2 - (x_2 - 0.5)^2, \tag{3.10}$$

subject to,

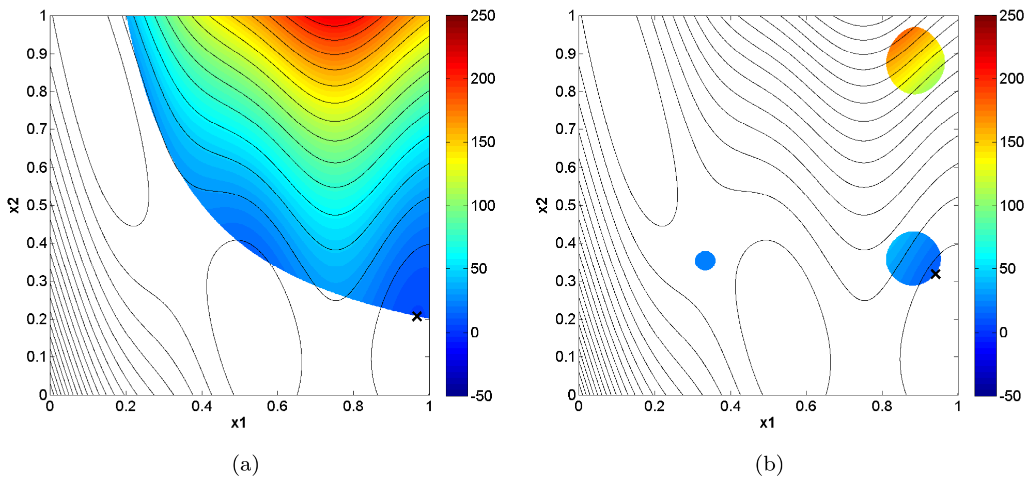


FIGURE 3.5: Bounded feasible space with $\times$ - global optimum. (a) Test problem 1 $g(\boldsymbol{x}) \geq 0.2$ (b) Test problem 2 $g(\boldsymbol{x}) \geq 6$.

$$g_1(\boldsymbol{x}) = (x_1 - 0.5)^2 - (x_2 - 0.5)^2 \leq 0.2, \tag{3.11}$$

$$g_2(\boldsymbol{x}) = \left((x_1 - 3)^2 + (x_2 + 2)^2\right) e^{(-x_2^7)} \leq 12, \tag{3.12}$$

where $x_1, x_2 \in [0, 1]$ giving a global feasible optimum $y^* = -0.7483$.

The bounded feasible space and the location of the global optimum for this problem are presented in Figure 3.6.

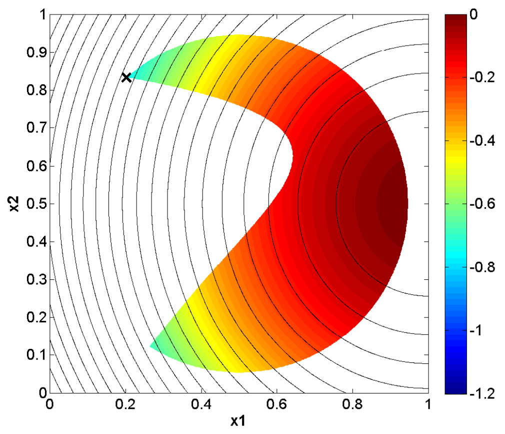


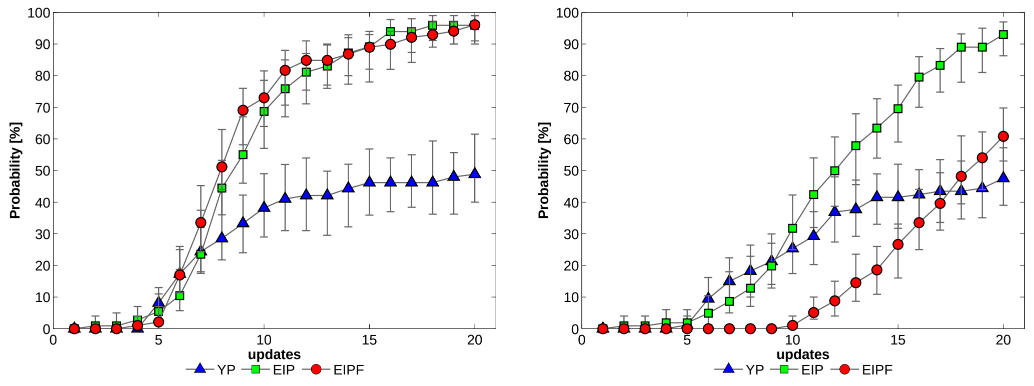
FIGURE 3.6: Bounded feasible space and $\times$ - global optimum for test problem 3 $g_1(\boldsymbol{x}) \leq 0.2$ and $g_2(\boldsymbol{x}) \leq 12$.

3.5 Results and Discussion

Test problem 1 uses an initial sample of eight points and 20 further updates whilst the more complex test problem 2 and test problem 3 use an initial sample with ten points and 30 further updates.

Figure 3.7 shows the results obtained when the different improvement criteria are applied to test problem 1. As expected, due to the lack of exploration of the objective space, YP performs poorly when compared to the other two approaches. YP tends to converge towards the closest basin of attraction which incidentally, by luck, happens to corresponds to the global optimum 50% of the time. Perhaps it is worth noting here that although this method is less reliable, when it does find the global optimum it is capable of converging towards accurate solutions, as shown by Figure 3.7(b).

Both methods based on the expected improvement reliably find the region of the global optimum as shown by Figure 3.7(a). However, as the accuracy of the desired solution is increased EIP becomes more efficient and a more reliable approach. Sasena [2002] goes some way to explain why this is often the case when comparing the probabilistic and penalty based methods. One concern is that combining the expected improvement with the probability of feasibility impacts the value of the improvement criterion too strongly at the constraint boundary. Since this is where the optimum lies, this can prevent convergence towards accurate solutions. We can see from these results that using a probabilistic approach offers little benefits over the penalty function when dealing with a single simple constraint.



(a) (b)

Figure 3.7: Test problem 1 mean probability and 95% confidence intervals of the best feasible point being within (a) 0.1 of the true optimum value and (b) 0.01 of the true optimum value.

In test problem 2 the complexity of the constraint is increased and both YP and EIP struggle to consistently identify the region of the global optimum, Figure 3.8(a). It is clear that the search is now hindered by the constraint handling method and a probabilistic approach better manages the constraint by relaxing the constraint boundaries. This allows progression of both the objective and constraint approximations. This works well for this problem since in the early stages of the search the feasible regions are poorly modelled. The poor performance of EIP clearly suggests that the one pass penalty function is a poor method when dealing with complex constraints. What allowed us to find accurate solutions in test problem 1 now prevents us from finding the global optimum altogether. However, all these methods struggle to consistently find accurate solutions.

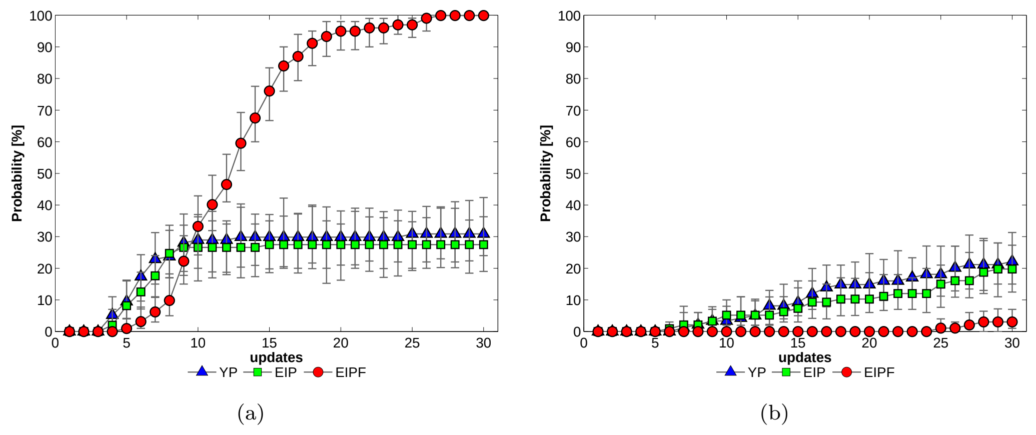


FIGURE 3.8: Test problem 2 mean probability and 95% confidence intervals of the best feasible point being within (a) 0.1 of the true optimum value and (b) 0.01 of the true optimum value.

It is evident that although EIPF consistently finds the region of the global optimum, converging towards accurate solutions remains difficult.

Figure 3.9 shows results for test problem 3. The global optimum of this problem is identified consistently by all three methods, Figure 3.9(a). This is found with very few updates since the objective and constraint functions can be easily modelled: pure exploitation of the objective combined with a penalty approach works well. The difficulty here lies with converging towards more accurate solutions since the optimum lies on the boundary of two constraints. Although these constraints are relatively simple to model they have to be very accurate at the boundaries for accurate solutions to be identified. Figure 3.9(b) shows that all methods struggle to identify accurate solutions. In cases where the constraint boundaries are difficult to approximate, adding updates in the 'just' infeasible region may help to build better approximations.

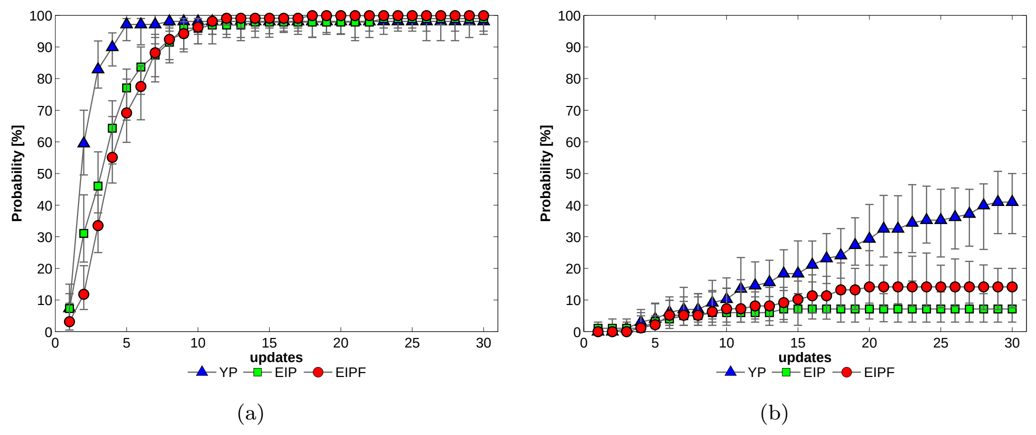


FIGURE 3.9: Test problem 3 mean probability and 95% confidence intervals of the best feasible point being within (a) 0.01 of the true optimum value and (b) 0.001 of the true optimum value.

Table 3.1 summarises the results in Figure 3.7(b), Figure 3.8(b) and Figure 3.9(b), illustrating the best improvement criterion when finding the exact global optimum. This highlights the poor reliability of all three improvement criteria on test problem 2 and test problem 3.

Table 3.1: Summary of mean results for test problem 1, 2 and 3. Mean probability of the best feasible point being within 0.01 on test problem 1 and 2 and 0.001 on test problem 3.

	Test problem 1		Test problem 2		Test problem 3	
	No. of updates		No. of updates		No. of updates	
	10	20	15	30	15	30
YP	25%	48%	**8%**	**22%**	**18%**	**41%**
EIP	**32%**	**93%**	7%	20%	7%	7%
EIPF	1%	61%	0%	3%	10%	14%

3.6 Aircraft Wing Design Problem

Next we turn to a real design problem that is dominated by constraints. Wing design for transonic civil aircraft is a very complex task. It is common for such tasks to incorporate aspects of strength, fuel capacity, operating costs and so on. Usually this process is dominated by complicated simulations using expensive design tools. In early design stages it is common for engineers to use less sophisticated tools in favour of reduced analysis cost and time. In such cases, empirical codes are used which make no attempt at solving the flow conditions over the wing or detailed structure analysis but give rapid estimates of likely drag and strength values.

In this study the aircraft performance is computed using a light release of a former Airbus conceptual design tool[2]. The optimization problem is simplified to a single low wing drag objective. Here this is defined as the wing drag area D/q also known as the drag per unit dynamic pressure. To retain some of the design complexity, limits are placed on key geometry and constraints exist on the wing weight, fuel tank volume, pitch-up margin and undercarriage bay length. Within the design tool, a wing weight and structural sizing analysis is performed in an attempt to size structural components sufficient for the required loading. Table 3.2 shows design parameter values, and limits, for a transonic civil transport wing with 11 design variables. The table also includes the low drag objective and constraint function values calculated using the design tool.

Full runs using the design tool are used for the construction of five surrogate models, one for the low drag objective and one for each constraint. The initial sample consists

[2]This version of the Tadpole concept design tool [Cousin and Metcalfe, 1990] was developed by Airbus for research at the University of Southampton.

TABLE 3.2: Best known design variables, limits, constraint values, and objective value.

Lower limit	Best design	Upper limit	Quantity
100	155	250	Wing area, m^2
6	11.4	12	Aspect ratio
0.2	0.446	0.45	Kink position
25	28.5	45	Sweep angle, deg
0.4	0.526	0.7	Inboard taper ratio
0.2	0.201	0.6	Outboard taper ratio
0.1	0.160	0.18	Root t/c
0.06	0.107	0.14	Kink t/c
0.06	0.060	0.14	Tip t/c
4.0	5.0	5.0	Tip washout, deg
0.65	0.82	0.84	Kink washout fraction
—	13,707	13,761	Wing weight, kg
40.0	40.02	—	Wing volume, m^3
—	5.392	5.4	Pitch-up margin
2.5	2.508	—	Undercarriage bay length, m
—	2.757	—	D/q, m^2

of 60 points and 90 further points are selected via an improvement criterion. In total, the design tool is evaluated 150 times. As before the performance of each strategy is compared using data collected over a number of different initial samples.

This is a real engineering problem and the exact location of the true optimum is unknown. This makes comparison of the best found location and the true optimum location an unreasonable metric for comparison. The error in the objective space can still act as a good comparison but in this case, the true optimum y^* is replaced with the best known feasible solution. The best feasible solution known is D/q = 2.7568 m^2, at a point where all four constraints are active, see Table 3.2.

Figure 3.10 illustrates the reliability of each method when applied to this aircraft wing design problem. When finding a solution within 0.05 of the best known solution, all methods find reliable solutions before the evaluation budget is exhausted. When finding solutions within a tighter accuracy of 0.01, only EIPF performs reliably. The poor performance of YP and EIP clearly demonstrates that finding accurate solutions is difficult to achieve when using simple penalty functions. This may be due to the fact that in many constrained optimization problems the global solution often lies close to, or on one or more constraint boundaries. In regions of multiple active constraints, locating a feasible point can be very difficult to achieve. In the case of the one pass penalty function, such regions are too easily discarded if any one of the constraints is deceptive or poorly modelled. Using EIPF clearly copes much better since the constraint boundaries have been relaxed and the impact of a poorly modelled constraint is not as severe. Table 3.3 highlights the dominant performance in 3.10(b).

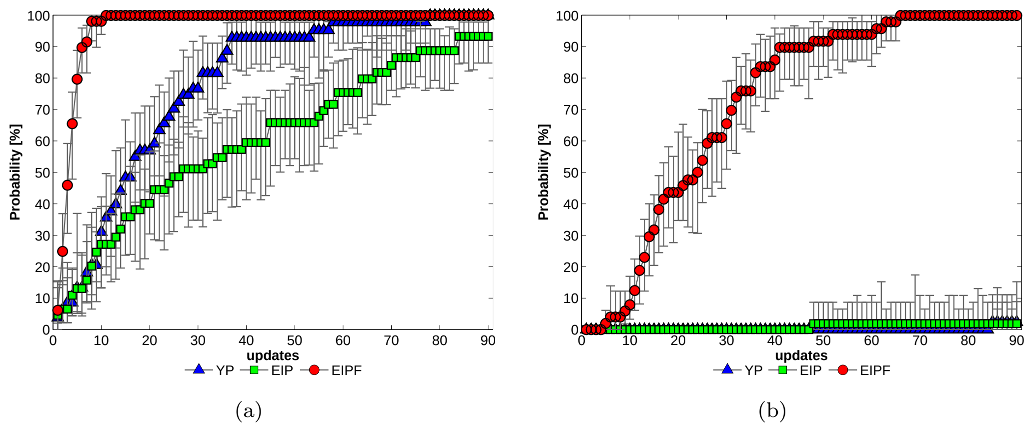


FIGURE 3.10: Aircraft wing design problem mean probability and 95% confidence intervals of the minimum feasible D/q being within (a) 0.05 of the best known solution and (b) 0.01 of the best known solution.

TABLE 3.3: Summary of mean results for the aircraft wing design problem. Mean probability of finding D/q within 0.01 of the best known solution.

	No. of updates	
	45	90
YP	0%	2%
EIP	0%	2%
EIPF	**89%**	**100%**

3.7 Conclusions

Many engineering design problems require a large number of time consuming, high fidelity computer simulations. In aid of reducing the total number of expensive evaluations, this chapter reviews the performance of three different improvement criteria suitable for surrogate-based constrained optimization.

From the results gained on the artificial test problems, it appears that the correct choice of improvement criteria is dependent on the problem being tackled. Exploiting the surrogate prediction can work well on simple unimodal problems but is not guaranteed to find global solutions when the objective is multimodal. Although less efficient on very simple problems, using expected improvement generalizes better when the objective becomes more complex. This can be modified to deal with constraints using either a penalty of probabilistic method. A penalty will work well if constraints are simple to approximate but can fail to find global solutions when the constraints become complex. Instead, using the probability of feasibility relaxes constraint boundaries and is capable of finding global solutions even on very complex problems but may prevent convergence towards more accurate solutions.

In the presence of multiple constraints in the real aircraft design problem, the probabilistic approach significantly outperformed the other two methods. Although EIPF appears superior, further improvements may still be made as solutions may be discarded if a single constraint gives a probability of feasibility close or equal to zero. In certain problems it may be better to search directly for solutions which lay on constraint boundaries. This is investigated further in the next chapter.

Choosing the correct improvement criteria for the task in hand may not always be obvious without prior knowledge of the complexity of the objective and each constraint. It is clear that in simply constrained problems, YP can perform well. If the objective is complex but constraints are simple, EIP is likely to be the best method whilst if both the objective and constraints are complex, EIPF is favoured. One way to achieve a more generalized approach is to combine the advantages of the different methods discussed. This can be achieved by hybridizing the methods. For example, it is possible to begin the search using EIPF and towards the end of the search switch to YP. This encourages exploration of the design space and relaxes the constraint boundaries early on in the search and exploits the predictions once the region of the global feasible optimum is identified. To explore this method further we would have to decide which methods to hybridize and at what point in the search to make a switch between methods. This is unlikely to generalize well and instead we investigate alternative approaches in the next chapter.

Chapter 4

Enhancing Improvement Criteria for Constraint Handling

Chapter 3 considers some basic approaches to constraint handling in surrogate-based optimization. The findings discussed support the use of expected improvement to help explore the objective function as opposed to purely exploiting the surrogate prediction. Two approaches to constraint handling were also tested. This compared the addition of a one pass penalty and multiplication by the probability of feasibility. Both approaches transform the constrained problem into an unconstrained one, leading to an aggregated search space. This can become difficult to search when dealing with complex or multiple constraints, which may lead to the selection of poor update points. Furthermore, all update points are chosen on the basis that they are likely to be feasible and also add value to the objective surrogate, showing no attempt at directly improving the constraint surrogate. Since badly modelled constraints will mislead the chosen update points, in certain cases it is better to update with the intention of explicitly improving the constraint surrogate(s). This chapter investigates approaches to better handle constraints with the intention of improving the efficiency and reliability of surrogate-based constrained optimization.

4.1 Enhanced Improvement Criteria

In most engineering design problems constraints tend to drive optimal designs towards limits on performance and geometry. Therefore, the global optimum tends to be a tight solution, lying on the boundary of many active constraints. When using surrogates, locating a tight solution relies on an accurate representation of the constraint boundaries. This can be difficult to achieve especially in the presence of complex, multiple constraints or when the constraints are more complex or harder to model than the objective. In these cases it becomes sensible to only construct surrogates for active constraints (if possible, all inactive ones are ignored) and attempt to reduce the predicted error explicitly along the constraint boundaries. To this end, different improvement criteria can be used that encourage updates to be placed in close proximity to the constraint boundaries, better modelling these tight regions. As long as the global region of the optimum has been identified, large errors in the constraint approximation away from the constraint boundary become unimportant.

Several authors have investigated improvement criteria that seek a desired response rather than finding a global optimum. Ranjan and Bingham [2008] provide an extension to the expected improvement for contour estimation, aiming to balance exploitation of the region surrounding a contour and exploration of the design space in areas of uncertainty. In a similar manner, Picheny et al. [2010] describe an improvement criterion based on the integrated mean squared error that encourages updates in the vicinity of a boundary and Bichon et al. [2008] introduces a formulation of the expected feasibility, providing an indication of how well a response is expected to satisfy an equality constraint.

Here we investigate similar formulations to enhance improvement criteria, encouraging updates to be selected along the constraint boundaries whilst efficiently converging towards the feasible global optimum. Although we stick to dealing with inequality constraints, these enhanced approaches can also be used to deal with equality constraints as demonstrated in Parr et al. [2012a].

4.1.1 Enhancing Probability of Feasibility

The probability of feasibility identifies regions of feasibility computed as a probability of the constraint prediction being less (or greater) than a constraint limit. This can perform well when dealing with inequality constraints but is less useful when trying to satisfy an equality. In traditional optimization, active inequality constraints are often treated as equality constraints. Keeping this in mind, sampling can be encouraged in the region of the constraint boundary by integrating the PDF over a reduced interval that surrounds the constraint limit. The enhanced probability of feasibility for an inequality treated as an equality constraint $h(\boldsymbol{x}) = h_{limit}$ becomes

$$P[h_{limit} - \tau \leq h \leq h_{limit} + \tau]_{\tau} = \int_{h_{limit}-\tau}^{h_{limit}+\tau} \boldsymbol{\phi}\,(h)\,\mathrm{d}h, \tag{4.1}$$

where τ reflects some required tolerance associated with meeting the constraint.

Treating the inequality as an equality in this way falsely suggests a region of feasibility below the constraint limit. This may be useful to help model the constraint boundary from both the feasible and infeasible directions and may offer advantages if the designer is happy with small constraint violations. Here the inequality constraint is assumed to be a hard constraint and the enhanced probability of feasibility for $g(\boldsymbol{x}) \leq g_{limit}$ becomes,

$$P[g \leq g_{limit}]_\delta = \int_{g_{limit}-\delta}^{g_{limit}} \boldsymbol{\phi}\,(g)\,\mathrm{d}g, \tag{4.2}$$

and for an inequality constraint $g(\boldsymbol{x}) \geq g_{limit}$ the enhanced probability of feasibility is expressed as

$$P[g \geq g_{limit}]_\delta = \int_{g_{limit}}^{g_{limit}+\delta} \boldsymbol{\phi}\,(g)\,\mathrm{d}g, \tag{4.3}$$

where δ defines the width of the integral.

This encourages designs to be selected on the feasible side of the constraint limit and retains the smoothing property associated with the probability of feasibility along the constraint boundary. This approach also gives the user some flexibility between exploiting constraint boundaries and exploring feasible regions. The extent of exploitation depends on the user defined value of δ.

Figure 4.1 illustrates the influence of the control parameter δ on a simple product constraint from test problem 1. It shows a low value of δ resulting in exploitation of the constraint boundary prediction. At a higher value of δ, the enhanced probability of feasibility is clearly more conservative and generalised towards the standard probability of feasibility. When dealing with inequality constraints in this study, δ takes a value of 5% of the full range of constraint values, as suggested by Picheny et al. [2010]. Since the range of true constraint values is unlikely to be known, δ is taken as 5% of the known range of outputs, simply given by the difference between g_{max} and g_{min}, the maximum and minimum constraint values sampled.

For an inequality constraint $g(\boldsymbol{x}) \leq g_{limit}$, the enhanced probability of feasibility is expressed analytically as,

$$P[g \leq g_{limit}]_\delta = \boldsymbol{\Phi}\left(\frac{g_{limit} - \hat{g}}{\hat{s}}\right) - \boldsymbol{\Phi}\left(\frac{g_{limit} - \delta - \hat{g}}{\hat{s}}\right), \tag{4.4}$$

and for $g(\boldsymbol{x}) \geq g_{limit}$,

$$P[g \geq g_{limit}]_\delta = \boldsymbol{\Phi}\left(\frac{g_{limit} + \delta - \hat{g}}{\hat{s}}\right) - \boldsymbol{\Phi}\left(\frac{g_{limit} - \hat{g}}{\hat{s}}\right), \tag{4.5}$$

where $\delta = 0.05\,|g_{max} - g_{min}|$.

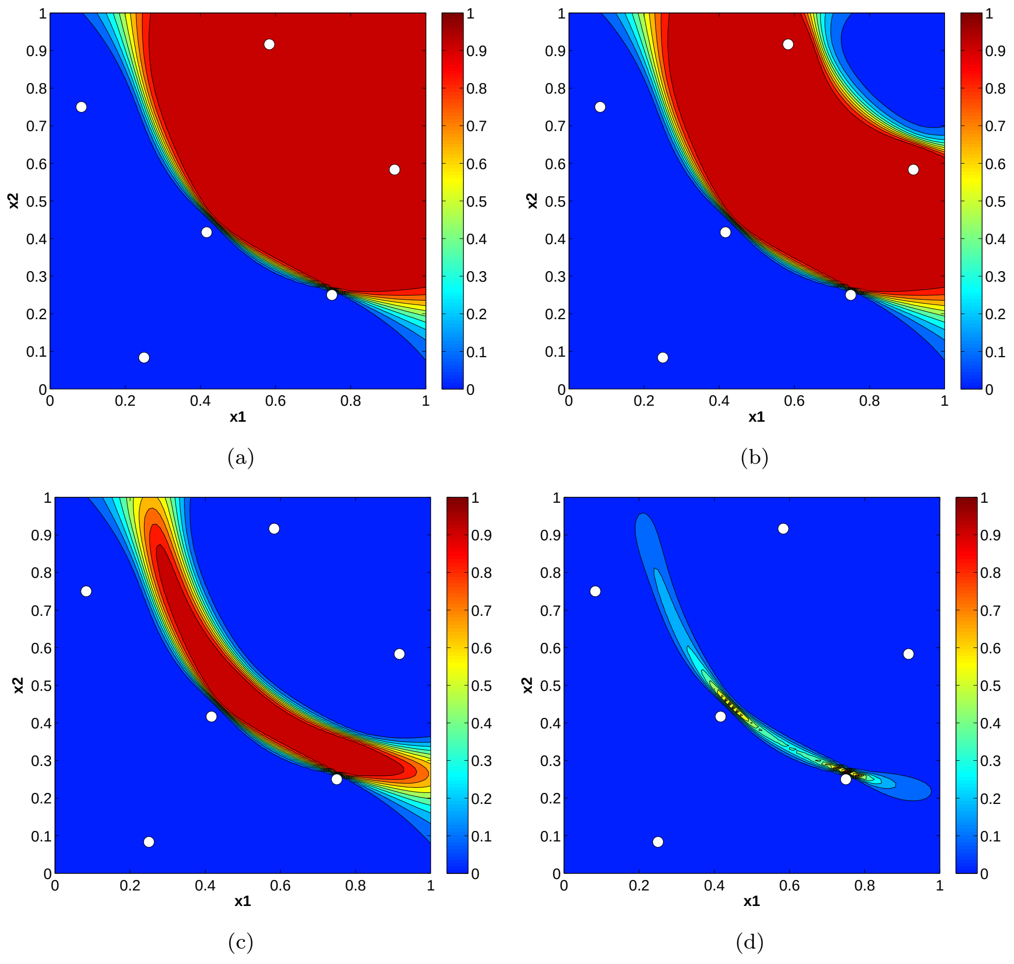


FIGURE 4.1: Product inequality constraint $g(\boldsymbol{x}) \geq 0.2$, ○ – sample data. Enhanced probability of feasibility when (a) $\delta = 1$, (b) $\delta = 0.4$, (c) $\delta = 0.1$ and (d) $\delta = 0.01$.

This constraint handling approach can be combined with the expected improvement in the same manner as the standard probability of feasibility, denoted as EIPF_δ.

4.1.2 Expected Feasibility

Ranjan and Bingham [2008] and Bichon et al. [2008] consider the extension of expected improvement to encourage updates in the region of a contour or a constraint boundary. Ranjan and Bingham [2008] consider the improvement function $I(\boldsymbol{x}) = \epsilon - [h_{limit} - h]^2$ whilst Bichon et al. [2008] consider the improvement as a measure of feasibility given as $F(\boldsymbol{x}) = \epsilon - |h_{limit} - h|$. Both of these approaches provide an indication of how well a response is expected to satisfy an equality constraint. Here a similar approach is taken for inequality constraints and the expected feasibility for an inequality constraint $g(\boldsymbol{x}) \leq g_{limit}$ is given as,

$$E[F(\boldsymbol{x})] = \int_{g_{limit}-\epsilon}^{g_{limit}} \left[1 - \frac{(g_{limit} - g)}{\epsilon}\right] \boldsymbol{\phi}\,(g)\,\mathrm{d}g \tag{4.6}$$

and for an equality constraint $g(\boldsymbol{x}) \geq g_{limit}$ the expected feasibility is expressed as

$$E[F(\boldsymbol{x})] = \int_{g_{limit}}^{g_{limit}+\epsilon} \left[1 - \frac{(g - g_{limit})}{\epsilon}\right] \boldsymbol{\phi}\,(g)\,\mathrm{d}g, \tag{4.7}$$

where $\epsilon = \frac{g_{max}}{\hat{s}_{max}}\hat{s}$.

Unlike the probability and enhanced probability of feasibility, this metric will be larger closer to the constraint boundary. Setting ϵ to the maximum constraint output g_{max} will scale the expected feasibility between zero and one with a maximum feasibility expected along the constraint boundary, Figure 4.2(a). This gives a metric that exploits the constraint boundary without the need to reduce the region of integration. Both Ranjan and Bingham [2008] and Bichon et al. [2008] define ϵ as a factor of the standard deviation of the prediction, $\epsilon = \alpha\hat{s}$. This encourages an extra degree of exploration determined by the value of α.

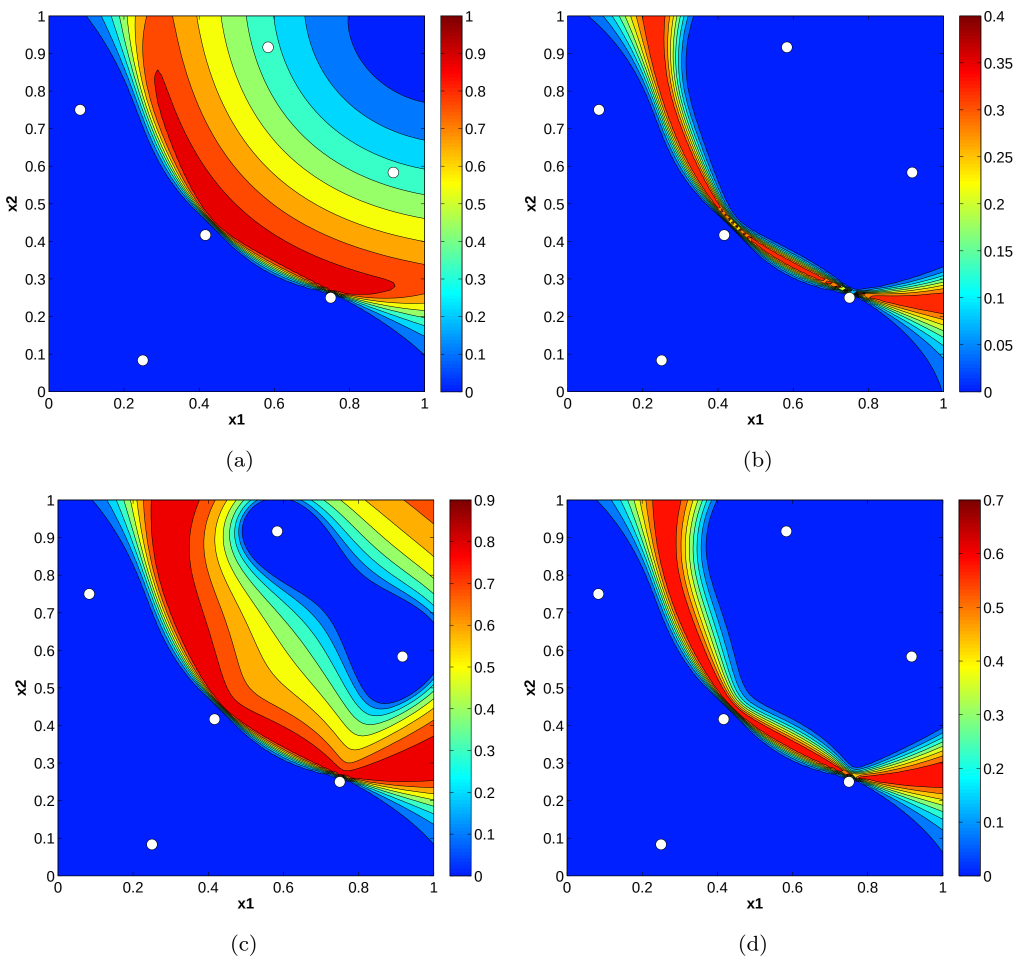


FIGURE 4.2: Product inequality constraint with $g(\boldsymbol{x}) \geq 0.2$, ○ – sample data. The expected feasibility when (a) $\epsilon = g_{max}$, (b) $\epsilon = 2\hat{s}$, (c) $\epsilon = 10\hat{s}$ and (d) $\epsilon = \frac{g_{max}}{\hat{s}_{max}}\hat{s}$.

Figures 4.2(b) and 4.2(c) illustrates the expected feasibility for different values of α on the simple product constraint. A small value of α results in a very narrow region of integration whilst a large value encourages wider ranging exploration. Basing the measure of feasibility on the predicted variance has an intuitive appeal but the correct choice of α is not obvious and largely problem dependent. To overcome this issue, it is necessary to sensibly scale the value of predicted variance. Here, $\epsilon = \frac{g_{max}}{\hat{s}_{max}}\hat{s}$ where g_{max} is the maximum constraint output and $\hat{s}_{max}$ is the maximum standard deviation. This gives a maximum measure of feasibility when $g \to g_{limit}$ and $\hat{s} \to \hat{s}_{max}$. Since g_{max} is unknown it is taken as the maximum sampled constraint output whilst $\hat{s}_{max}$ requires an additional search to find the maximum standard deviation.

For an inequality constraint $g(\boldsymbol{x}) \leq g_{limit}$, the expected feasibility when $\hat{s} > 0$ is expressed analytically as,

$$\begin{aligned} E[F(\boldsymbol{x})] = &\left[1 - \frac{(g_{limit} - \hat{g})}{\epsilon}\right]\left[\boldsymbol{\Phi}\left(\frac{g_{limit} - \hat{g}}{\hat{s}}\right) - \boldsymbol{\Phi}\left(\frac{g_{limit} - \epsilon - \hat{g}}{\hat{s}}\right)\right] \\ &-\frac{\hat{s}}{\epsilon}\left[\boldsymbol{\phi}\left(\frac{g_{limit} - \hat{g}}{\hat{s}}\right) - \boldsymbol{\phi}\left(\frac{g_{limit} - \epsilon - \hat{g}}{\hat{s}}\right)\right], \end{aligned} \tag{4.8}$$

and for $g(\boldsymbol{x}) \geq g_{limit}$,

$$\begin{aligned} E[F(\boldsymbol{x})] = &\left[1 - \frac{(\hat{g} - g_{limit})}{\epsilon}\right]\left[\boldsymbol{\Phi}\left(\frac{g_{limit} + \epsilon - \hat{g}}{\hat{s}}\right) - \boldsymbol{\Phi}\left(\frac{g_{limit} - \hat{g}}{\hat{s}}\right)\right] \\ &+\frac{\hat{s}}{\epsilon}\left[\boldsymbol{\phi}\left(\frac{g_{limit} + \epsilon - \hat{g}}{\hat{s}}\right) - \boldsymbol{\phi}\left(\frac{g_{limit} - \hat{g}}{\hat{s}}\right)\right], \end{aligned} \tag{4.9}$$

where $\epsilon = \frac{g_{max}}{\hat{s}_{max}}\hat{s}$.

Since this formulation of expected feasibility is scaled between zero and one, this metric can be combined with the expected improvement in the same manner as the probability of feasibility, denoted here as EIEF. Note that this improvement criterion does not require any user defined parameters when dealing with inequality constraints and is independent of the tolerance τ when dealing with equalities, however, this is at the expense of an additional search to find $\hat{s}_{max}$.

4.2 Using Multiobjective Optimization to Handle Constraints

Another enhancement considered in this chapter is the use of multiobjective optimization to handle constraints. So far, in both the penalty and probabilistic approaches considered for constraint handling, the constrained problem is transformed into an unconstrained one. This manipulation of the constrained problem may result in some misrepresentation, a concern considered by Audet et al. [2000] and more recently by Regis [2011]. Furthermore, using improvement criteria such as expected improvement tends to lead to multimodal landscapes, which in themselves can be difficult to search. When the problem is transformed using the penalty or probabilistic approach, the search space becomes riddled with cliffs, further increasing the difficulty of the search. This becomes even more of an issue when dealing with complex or multiple constraint functions. This aggregation of the search space can be avoided by treating objective improvement and constraint satisfaction as separate goals. This concept is explored in traditional optimization using filters [Fletcher and Leyffer, 2002, Audet et al., 2000] and in evolutionary algorithms using sub populations for each constraint [Coello Coello, 2000a] or using nondominance to rank constraint violation [Angantyr et al., 2003, Ray et al., 2000, Oyama et al., 2005]. Here multiobjective optimization is used to formulate a set of potential update points that are all Pareto optimal and offer a trade-off between objective improvement and constraint satisfaction.

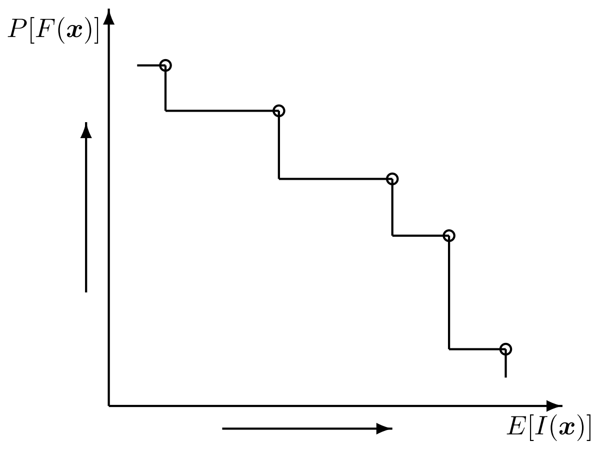


FIGURE 4.3: Pareto front when maximizing both $E[I(\boldsymbol{x})]$ and $P[F(\boldsymbol{x})]$: ○ – Pareto set.

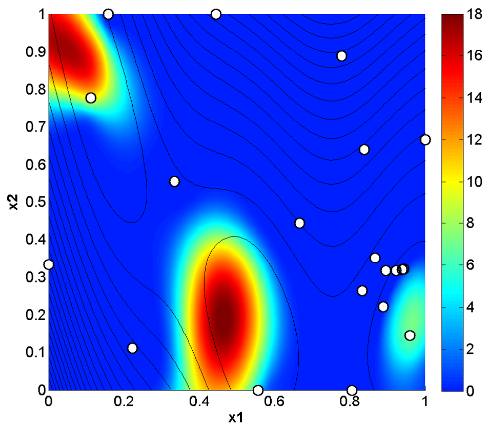
Figure 4.3 shows an example of two goals presented as a Pareto front. In this figure it is obvious that any increase in expected improvement of the objective will be detrimental to the probability of feasibility of the constraint. Conversely, any increase in probability of feasibility leads to a smaller expected improvement.

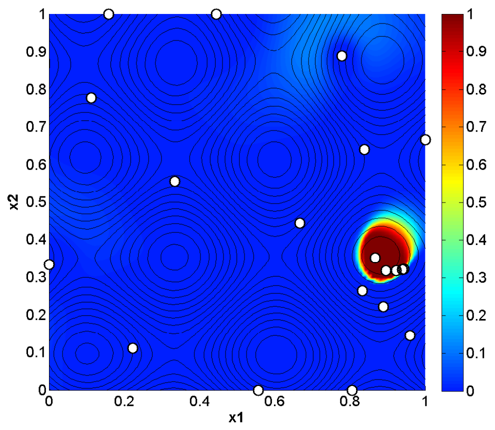
Constructing a Pareto set gives the designer a flexible choice of update points. Since every point in the Pareto set will lead to objective or constraint improvement, it is intuitive to select more than one point from the Pareto set. This allows multiple designs to be selected that can be evaluated in parallel, further accelerating the optimization process. This is a very useful feature of this approach and is investigated further in the next chapter. To remain consistent with selecting update points one at a time, only single updates are considered in this chapter. Since we have a set of potential updates, it is therefore necessary to select one. Here we choose the individual point from the Pareto set with the maximum product of $E[I(\boldsymbol{x})]$ and $P[F(\boldsymbol{x})]$. When making use of multiobjective optimization this method is denoted as EIvsPF. An important note here is that in the single objective approach, EIPF can be considered as an unknown weighted sum of $E[I(\boldsymbol{x})]$ and $P[F(\boldsymbol{x})]$. Intuitively, by searching EIPF directly as a single objective, the solution will be nondominated and expected to lie somewhere on the Pareto front. Typically you would expect this solution to correspond to the optimum point found using EIvsPF when the point from the Pareto set with maximum product of $E[I(\boldsymbol{x})]$ and $P[F(\boldsymbol{x})]$ is selected. This is usually the case but when combining the two goals as a single objective, the aggregation can cause the search space to become severe and complex and in certain cases using a multiobjective approach can achieve better solutions. This is a commonly observed phenomenon [Waldock and Corne, 2011, Corne et al., 2003] and several authors have even reformulated single objective problems so they can be solved using multiobjective methods, see Jensen [2003], Knowles et al. [2001] and Neumann and Wegener [2008].

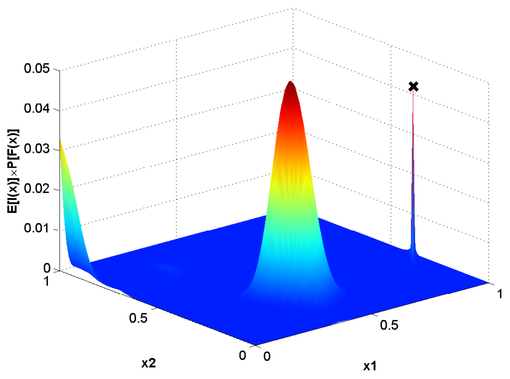
To clarify this point Figure 4.4 shows a comparison of the two approaches when finding update points on test problem 2. This figure is drawn after a number of updates so the feasible regions and areas of good objective values have already been identified. As hinted earlier the search space for the expected improvement and probability of feasibility can be far from simple and when combined as a single objective, $E[I(\boldsymbol{x})] \times P[F(\boldsymbol{x})]$, the search space is multimodal with several local optima. Here, the global maximum of $E[I(\boldsymbol{x})] \times P[F(\boldsymbol{x})]$ is very difficult to find, situated on top of a small peak with steep edges, Figure 4.4(c). The single objective approach uses a hybrid search using a GA and followed by a DHC. The multiobjective method uses a standard implementation of NSGA-II[1]. Both these algorithms are run with a population size of 100 with 10,000 total evaluations. Figure 4.4(d) shows the best solution identified using each method. The single objective search EIPF does not find the global solution. Perhaps this is as expected since the global optimum lies in a very severe location. Using EIvsPF however, approaches the problem in a different manner, searching from different directions which are otherwise limited by the complex $E[I(\boldsymbol{x})] \times P[F(\boldsymbol{x})]$ design space. NSGA-II finds a set of solutions clustered in different areas of the design space and selecting the solution with the largest product of $E[I(\boldsymbol{x})]$ and $P[F(\boldsymbol{x})]$ corresponds to the global solution.

It is important to note that when using EIvsPF, further constraints will add additional probability of feasibility objectives, increasing the dimensionality of the Pareto front. Multiobjective problems with more than four objectives are classified as manyobjective problems and are inherently difficult to solve [Khare et al., 2003, Praditwong and Yao, 2007, Schütze et al., 2010]. In problems with more than a few constraints, the probability of feasibility of each constraint can be multiplied together. This reduces the problem to two objectives, one for the objective improvement and the other for the satisfaction of

[1]Find this implementation at http://www.mathworks.com/matlabcentral/fileexchange/10429.


(a)


(b)


(c)

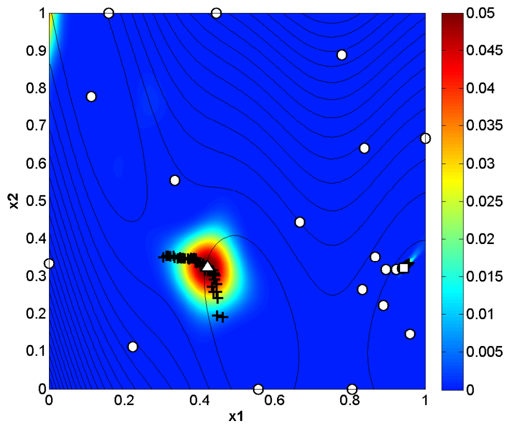

(d)

FIGURE 4.4: Comparison of searching improvement criteria on test problem 2. Contours show the true functions and ○ – is the sample data. (a) $E[I(x)]$ search space. (b) $P[F(x)]$ search space. (c) $E[I(x)] \times P[F(x)]$ search space, × – global optimum. (d) △ – EIPF the single objective search and □ – EIvsPF the multiobjective search, + – Pareto set.

all constraints. This is used on the aircraft wing design problem since treating the four constraints individually would lead to a manyobjective problem.

4.3 Results and Discussion

The enhancements just discussed are now applied to the four problems investigated in the previous chapter. For each constraint handling method discussed, $P[F(\boldsymbol{x})]$, $P[F(\boldsymbol{x})]_\delta$ and $E[F(\boldsymbol{x})]$, we compare a single objective and multiobjective approach. These methods include EIPF investigated in Chapter 3, the multiobjective alternative EIvsPF and all enhanced variations, EIPF$_\delta$, EIvsPF$_\delta$, EIEF and EIvsEF. These abbreviations are listed in Table 4.1 for reference. The single objective improvement criteria EIPF, EIPF$_\delta$ and EIEF are searched using a 10,000 iteration GA followed by a 5,000 iteration DHC, as in Chapter 3. The multiobjective methods, EIvsPF, EIvsPF$_\delta$ and EIvsEF are searched using NSGA-II with a population of 200 and 100 generations, producing a highly populated Pareto set where a single individual is selected per update. For the aircraft wing design problem, this is increased to 200 generations to deal with the larger number of design variables.

TABLE 4.1: List of improvement criteria.

Improvement criteria	Constraint handling	Search method
EIPF	Probability of feasibility	GA+DHC
EIvsPF	Probability of feasibility	NSGA-II
EIPFδ	Enhanced probability of feasibility	GA+DHC
EIvsPFδ	Enhanced probability of feasibility	NSGA-II
EIEF	Expected feasibility	GA+DHC
EIvsEF	Expected feasibility	NSGA-II

4.3.1 Artificial Test Problems

Figure 4.5 shows the results obtained when the different improvement criteria are applied to test problem 1. All these methods tested are able to find the region of the global optimum, with little statistical significance between each approach. When finding the global solution to a greater accuracy, the enhanced methods outperform the standard probability of feasibility. The multiobjective search improves performance when using expected feasibility but has little impact when using the enhanced probability of feasibility and in fact degrades performance when using the standard formulation. Although the enhanced approaches have helped to improve the performance of the constraint handling, for this simply constrained problem using a simple penalty function still performs better than the enhanced methods tested, see Figure 3.7.

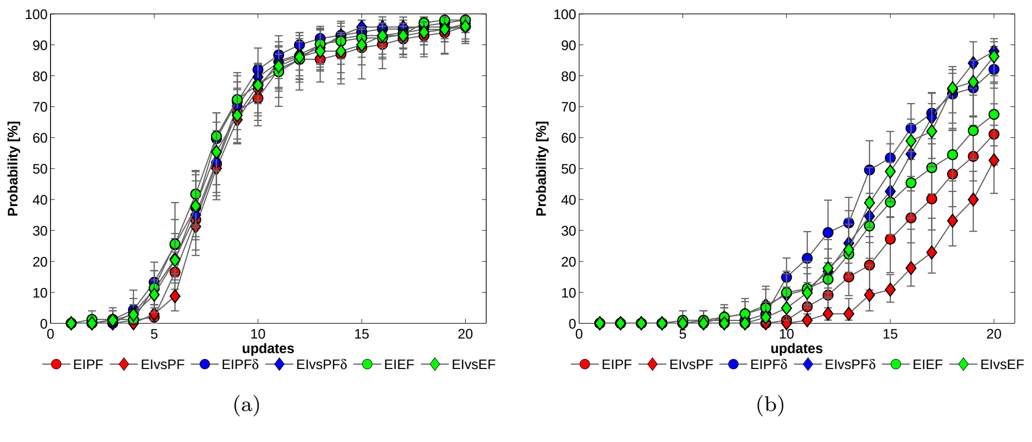
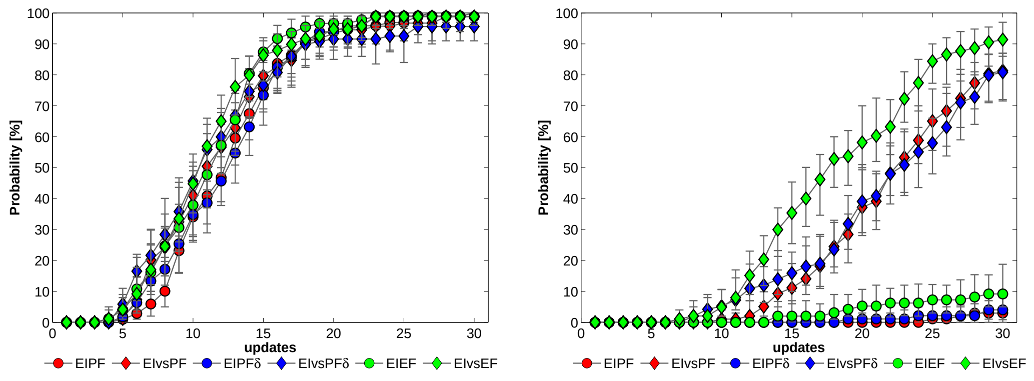


FIGURE 4.5: Test problem 1 mean probability and 95% confidence intervals of the best feasible point being within (a) 0.1 of the true optimum value and (b) 0.01 of the true optimum value.



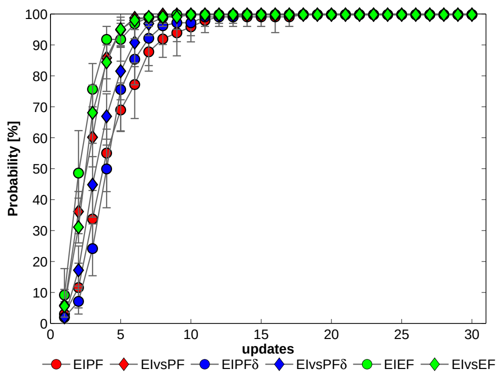
(a) (b)

FIGURE 4.6: Test problem 2 mean probability and 95% confidence intervals of the best feasible point being within (a) 0.1 of the true optimum value and (b) 0.01 of the true optimum value.

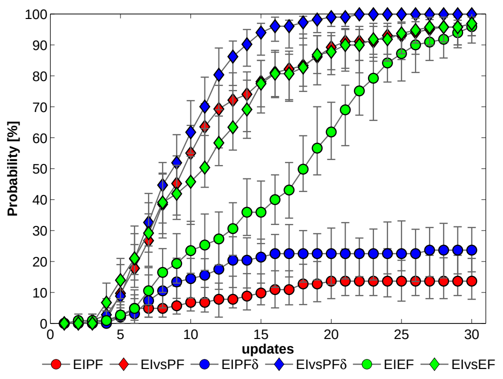
For test problem 2, all improvement criteria are able to identify a solution within the global optimum. In Chapter 3 we noted that all methods tested struggled to consistently find accurate solutions. In this problem combining the improvement of the objective and feasibility of the constraint as a single objective results in a very difficult landscape to search. Better updates are found by treating the objective and constraint separately. This is highlighted in Figure 4.6(b) where it is clear that employing a multiobjective search offers a significant enhancement over the single objective approach. All methods that use a multiobjective search have a much better performance. Here, EIvsEF is the most efficient and the most reliable improvement criterion tested.

Figure 4.7 shows results for test problem 3. As in Chapter 3, all methods consistently identified the region of the global optimum. Although there is little difference between the performance of each method, EIvsPF, EIEF and EIvsEF are slightly more efficient. The methods tested in Chapter 3 struggled to consistently identify accurate solutions due to the narrow region of feasibility at the global optimum. This is also the case for EIPF$_\delta$ but it does offer a small improvement over the standard approach. In this problem, accurately predicting the constraint boundaries in the region of the global optimum is paramount. Using the expected feasibility encourages points to be placed on the constraint boundary, offering improved performance. Further performance gains are achieved by using a multiobjective search. This approach finds better updates converging towards more accurate solutions. This is due to the multiobjective approach avoiding the aggregation of the design space in a narrow region dominated by constraints. Here, EIvsPF$_\delta$ is the most efficient and the most reliable improvement criterion tested.

Table 4.2 summarises the results in Figure 4.5(b), Figure 4.6(b) and Figure 4.7(b). From this summary it is evident that a better performance is achieved when using the enhanced improvement criteria but no one approach performs the best for all problems.



(a)



(b)

FIGURE 4.7: Test problem 3 mean probability and 95% confidence intervals of the best feasible point being within (a) 0.01 of the true optimum value and (b) 0.001 of the true optimum value.

TABLE 4.2: Summary of mean results for test problem 1, 2 and 3. Mean probability of the best feasible point being within 0.01 on test problem 1 and 2 and 0.001 on test problem 3.

	Test problem 1		Test problem 2		Test problem 3	
	No. of updates		No. of updates		No. of updates	
	10	20	15	30	15	30
EIPF	1%	61%	0%	3%	10%	14%
EIPFδ	**15%**	82%	0%	4%	21%	24%
EIEF	10%	68%	2%	9%	36%	96%
EIvsPF	1%	53%	11%	81%	77%	97%
EIvsPFδ	9%	**88%**	16%	81%	**94%**	**100%**
EIvsEF	5%	86%	**35%**	**91%**	78%	97%

4.3.2 Aircraft Wing Design Problem

Figure 4.8 shows the results of the enhanced methods on the aircraft wing design problem. All methods rapidly identify feasible solutions within 0.05 of the best known solution. These methods offer improved performance when compared to the penalty approaches investigated in Chapter 3 but do not offer any improvements over the standard constrained improvement criterion EIPF. This is highlighted in Table 4.3, a surprising result since the optimal solution lies on the boundary of four active constraints. In this case the enhanced probability of feasibility and expected feasibility exploit the constraint boundaries too strongly, restricting the development of the constraint surrogates early on in the search. This may be mitigated using a larger initial sample or reducing the amount the constraints are exploited as the search progresses (i.e. reduce δ as we add more updates). Here it is evident that using the multiobjective approach has a very poor performance when compared to the single objective search. This poor performance is due to a lower density of solutions in the design space. At lower dimensions, the number of solutions in the region of optimal updates is fairly dense due to a relatively large population of Pareto solutions. With increasing dimensions, the density of Pareto solutions in the design space can quickly become sparse. This is less of a problem for the single objective approach as it aims to converge towards a single solution rather than a Pareto set of solutions.

Dealing with a larger number of design variables is currently a major weakness of this multiobjective approach. This may be alleviated by using a search with a larger population as in [Parr et al., 2012b] but this can quickly become impractical and much more time consuming than using a single objective search. An alternative is to concentrate the multiobjective search on particular regions on the Pareto front, in turn focusing the search on a particular region of the design space. As we will see in the next chapter, good update points tend to lie on knee points of the Pareto front. By explicitly searching for these regions the cost of applying a multiobjective search at higher dimensions can be significantly reduced [Deb and Gupta, 2011]. Achieving similar performance gains as seen in the other test problems and at a reasonable cost, requires further investigation into the type of multiobjective search used, an area of future work suggested in the final chapter.

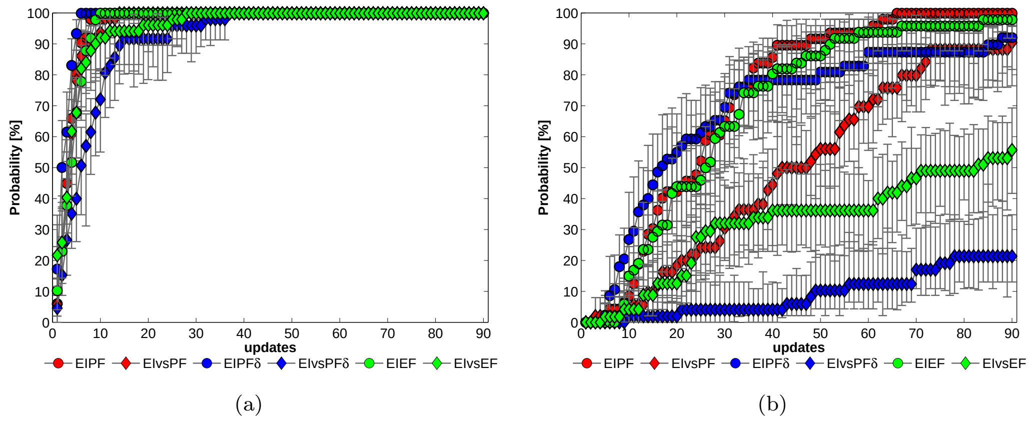


FIGURE 4.8: Aircraft wing design problem mean probability and 95% confidence intervals of the minimum feasible D/q being within (a) 0.05 of the best known solution and (b) 0.01 of the best known solution.

4.4 Conclusions

This chapter attempts to improve the performance of the improvement criteria investigated in Chapter 3. Enhanced performance is achieved in two main ways. Firstly, updates are encouraged to lie along the constraint boundaries, better modelling tight regions that constrain the global optimum. This is achieved through the formulation of the enhanced probability of feasibility and expected feasibility. A second enhancement

Table 4.3: Summary of mean results for the aircraft wing design problem. Mean probability of finding D/q within 0.01 of the best known solution.

	No. of updates	
	45	90
EIPF	**89%**	**100%**
EIPFδ	78%	92%
EIEF	84%	98%
EIvsPF	50%	90%
EIvsPFδ	6%	21%
EIvsEF	36%	56%

uses a multiobjective search to better balance the improvement of the objective and constraint feasibility. This avoids any aggregation of the design space and allows the search for new update points to take different directions to a single objective approach. This identifies better updates leading to better performance on the artificial test problems. A major weakness of the multiobjective search is that it is more vulnerable to the curse of dimensionality, performing poorly when applied to the 11 dimension aircraft wing design problem. For good results to be achieved, the multiobjective search requires a larger density of Pareto solutions in the region of good updates, requiring many more evaluations of the improvement criteria when compared to the single objective approach. The poor performance of the multiobjective search at higher dimensions may also be improved by focusing the search on promising regions of the Pareto front rather than searching for a full Pareto set of solutions.

Although the enhanced methods introduced here can offer improved performance, in many cases they should be used with caution. Although these methods generalise well to problems with complex constraints, a simple penalty function may prove to be the most efficient if constraints are simple. It should also be noted that encouraging updates to lie on the constraint boundaries may hinder the performance of the search if the global solution is not tight. In many cases the designer may be aware of which constraints are active and result in a tight solution. If uncertain, the designer should use the standard probability of feasibility or simple penalty functions instead.

Chapter 5

Parallel Evaluations in Surrogate-Based Constrained Optimization

Chapter 3 considers surrogate-based constrained optimization using a conventional two stage approach. The surrogate is first built based on a sample of evaluated points followed by a search of an improvement criterion to find update points that offer model improvement. This process is repeated until a time limit, evaluation budget or model accuracy is reached. Rather than finding single updates sequentially, distributing the evaluation of multiple update points on several processers is an attractive concept. This will have a clear benefit on the rate that surrogate models are developed, further accelerating convergence towards optimal solutions.

5.1 Multiple Update Strategies

The idea of multiple updates is far from new, having been explored over a decade ago by Schonlau [1997]. With the availability of parallel computing becoming commonplace, formulation of improvement criteria to select multiple updates has received further attention in recent years [Henkenjohann and Kunert, 2007, Ponweiser et al., 2008b, Loeppky et al., 2010, Viana and Haftka, 2010b]. Recently Ginsbourger et al. [2010] provide an extension to Schonlau's early work by providing a sound analytical expression for finding multiple updates based on the expected improvement. Unfortunately this relies on multivariate integrals and the expressions become mathematically challenging beyond selecting two update points. Viana and Haftka [2010a] alleviate this problem by selecting target values and computing the probability of improving upon these targets, assuming the probability of improvement at a point is independent from all other points. By selecting multiple targets, different interpretations of the probability of improvement can be search simultaneously. This is a simple approach to finding multiple updates but requires a suitable selection of target values and looses any benefits associated with employing expected improvement.

Although some progress has been made towards developing improvement criteria suitable for selecting multiple updates, little effort has been made to apply these algorithms to constrained problems. Here a new method is introduced that addresses this issue and is compared against existing methods with added constraint handling. As mentioned in the previous chapter, this alternative method is based on selecting multiple updates from a Pareto set of solutions.

5.1.1 Sequential Designs in Stages

Conventionally, expected improvement describes a sequential process, searching for one update point at a time. This can be very time consuming and cost savings may be made by sampling multiple points at once rather than sampling at multiple stages. Ideally, when selecting q updates, the new designs, $\boldsymbol{x}^{(n+1)}, ..., \boldsymbol{x}^{(n+q)}$, are found by maximizing $E[I(\boldsymbol{x}^{(n+1)}, ..., \boldsymbol{x}^{(n+q)})]$. Unfortunately this involves multiple integrals with q-dimensional Gaussian distribution functions. Although an analytical expression for the so called 'multipoints expected improvement' when $q = 2$ is provided by Ginsbourger et al. [2010], solving this problem numerically when $q > 2$ requires expensive statistical estimate techniques such as Monte-Carlo simulations. Schonlau simplifies this approach by computing the multipoint expected improvement sequentially. The so called 'sequential design in stages' approach finds q updates using an adapted expected improvement algorithm. The expected improvement at the $(n + i)^{th}$ step when $\hat{s}^{(n+i)} > 0$ is given as

$$\begin{aligned} E^{(n+i)}[I(\boldsymbol{x})] =& \hat{s}^{(n+i)} \frac{\left(y_{minfeas}^{(n)} - \hat{y}^{(n)}\right)}{\hat{s}^{(n)}} \boldsymbol{\Phi} \left(\frac{y_{minfeas}^{(n)} - \hat{y}^{(n)}}{\hat{s}^{(n)}} \right) \\ &+ \hat{s}^{(n+i)} \boldsymbol{\phi} \left(\frac{y_{minfeas}^{(n)} - \hat{y}^{(n)}}{\hat{s}^{(n)}} \right), \end{aligned} \tag{5.1}$$

where $i = 1, ..., q$. In this formulation $\hat{s}^{(n+i)}$ is the estimated error of prediction at $x^{(n+i)}$. This estimate depends only on $\boldsymbol{x}^{(n+i)}$ and the updated correlation parameters, not the unknown response $y^{(n+i)}$. This permits multiple regions for update points to be identified before any expensive evaluations are made. Schonlau suggested that it is important to retain $\hat{s}^{(n)}$ in the formulation otherwise the certainty of the difference $y_{minfeas}^{(n)} - \hat{y}^{(n)}$ is falsely predicted.

As with the single update approach, this method is repeated for a number of stages until a total evaluation budget or time limit is reached. In cases where designs are time consuming to evaluate, sequential design in stages offers a major advantage since the evaluation of multiple designs at each stage lends itself to parallelization.

5.1.2 Kriging Believer and Constant Liar

To overcome the limitations of calculating analytical solutions for multiple updates, Ginsbourger et al. [2010], introduced two strategies to approximate the multipoints expected improvement when $q > 2$, referred to as the 'Kriging believer' and the 'constant liar'. Rather than simultaneously searching for multiple updates, both these methods find a set of solutions sequentially in a similar manner to Schonlau's sequential design in stages. Whilst the original sequential design in stages simply updates the estimated error in the model, the Kriging believer and constant liar also update the mean prediction. This requires the updated response $y^{(n+i)}$ which is unknown. In order to find a set of update points without evaluating the true response, the Kriging believer substitutes the unknown response with a dummy value given by the current Kriging prediction. This enables a set of updates to be identified, again in a sequential process.

Believing the surrogate model for several iterations comes with an element of risk and the search can be easily misled. This may be a concern if the Kriging prediction overshoots the true function, potentially leading to a cluster of points in optimistic regions. Perhaps a more logical approach is to encourage the search towards exploitation or exploration. This can be achieved by forcing a lie for the unknown responses. In the case of the constant liar, this dummy value or lie is fixed by the user. Ginsbourger et al. considered three values for the lie, y_{min}, y_{max} and y_{mean}. The most exploitative of the three, y_{min}, was demonstrated to be a promising strategy when predicting multiple updates on the Branin-Hoo test function.

5.1.3 Expected Improvement with Cluster Analysis

Rather than adopting a sequential approach to finding multiple updates, the history of a single search can be examined for multiple local minima. Since the expected improvement metric is typically highly multimodal, selecting a single update from q local basins of attraction offers an intuitive approach for choosing multiple updates. This is an approach taken by Sóbester et al. [2004] and Toal [2009]. Here, a heavy hybrid search consisting of a GA and a DHC is used to search the improvement criterion. The full history of the search is grouped into q clusters in the design vector space using k-means clustering [Anderberg, 1973]. The designs that offer greatest improvement from each cluster are then selected for the set of multiple updates. Unlike the other methods discussed, this approach only requires a single search to identify multiple update points.

This may offer an advantage if searching the improvement criterion multiple times at each update stage becomes expensive. An obvious drawback of this method is that there is no guarantee that q good updates will be identified from q local basins of attraction and may lead to some updates being located well away from areas of interest.

5.1.4 Handling Constraints

A probabilistic approach is adopted here for the modification of the multiple update improvement criteria. We assume that the designer is unaware of the activity of any constraints and use the standard probability of feasibility for constraint handling. When using the expected improvement with cluster analysis, this is a straightforward modification and denoted here as EIPFq. The modified version of Schonlau's sequential design in stages, denoted as EIPFs, does not permit the constraint approximation to be updated when sequentially finding q updates. Therefore the constraint approximation and the probability of feasibility remain unaltered during each update stage. This is also the case in the modified Kriging believer and constant liar methods denoted as EIPFkb and EIPFcl. It is possible to update both the mean prediction of the objective and constraint approximations sequentially during each stage with dummy values evaluated from the current Kriging predictions but updating constraints with a poor prediction may falsely identify feasible solutions.

5.2 Expected Improvement versus Probability of Feasibility

As pointed out in Chapter 4, using multiobjective optimization to find a Pareto set of potential updates lends itself to the selection of multiple update points. It is possible to select multiple updates in a number of ways. The designer may make a weighted judgment depending on the complexity of the objective and constraint functions. For example, this may lead to the selection of points at the end points of the Pareto set. Here, the selection of q updates is performed in a two step process. First the Pareto set is grouped into q clusters in the design space using k-means clustering. Next, the individual with the highest product of expected improvement and probability of feasibility is selected from each cluster. This gives q update points that tend to be diverse when clusters are sparse and exploit interesting regions when clusters are dense. It is also possible to cluster the designs in the objective space. Although this is not investigated here this is likely to offer advantages when dealing with a large number of design variables.

Figure 5.1 provides some insight into this approach for selecting multiple updates on test problem 2. Due to the small values of both expected improvement and probability of feasibility at optimal update locations, the Pareto set is presented using natural logarithms. Interestingly, selected updates gravitate towards the knee points of the

Pareto front. At these locations a small gain in the expected improvement results in a 'more' significant sacrifice in the probability of feasibility and vice versa.

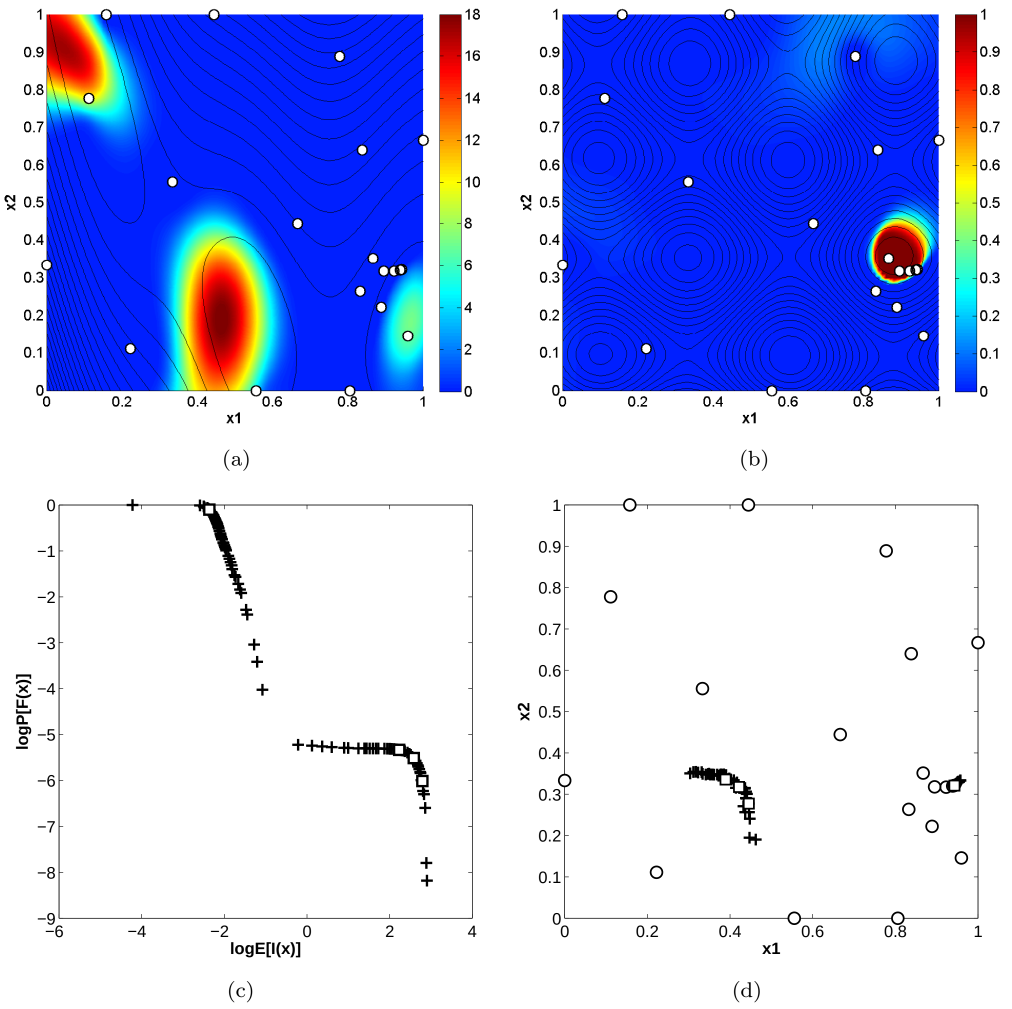


FIGURE 5.1: Expected improvement and probability of feasibility example for test problem 2. (a) $E[I(\boldsymbol{x})]$. (b) $P[F(\boldsymbol{x})]$. (c) $E[I(\boldsymbol{x})]\mathrm{vs}P[F(\boldsymbol{x})]$ $+$ – Pareto set in objective space. (d) $E[I(\boldsymbol{x})]\mathrm{vs}P[F(\boldsymbol{x})]$ $+$ – Pareto set in design space including ○ – previous samples and □ – selected updates with $q = 4$.

This approach offers a better balance between selecting updates that improve both the objective and constraint surrogates and identifies multiple update points using a single multiobjective search. Weaknesses of using a multiobjective search was discussed in Chapter 4 and the current implementation remains suitable only on problems with a relatively low number of design variables.

5.3 Results and Discussion

We now present some results on the three test problems introduced in Chapter 3. Test problem 1 uses an initial sample of eight points and selects four updates ($q = 4$) in ten update stages. Test problem 2 and test problem 3 use an initial sample of ten points, $q = 4$ and 15 update stages. Assuming resources exists (four separate machines) that allow new update points to be evaluated in parallel, the time spent evaluating new designs (referred to as the evaluation wall clock time) equates to half of the total evaluation time required using single updates in Chapter 3 and Chapter 4. Although the evaluation wall clock time is halved, the total CPU time (or total number of evaluations) is doubled. A dashed line is included that represents the equivalent CPU time required when using methods discussed in previous chapters.

In Figure 5.2 it is clear that EIPFs, EIPFkb and EIPFcl all converge towards the region of the global optimum very efficiently. When compared to using single updates, Figure 3.7 and 4.5, this is achieved in an equivalent CPU time and 25% of the evaluation wall clock time. Here, EIvsPF is slightly less efficient and EIPFq performs poorly compared to the other multiple update strategies. Other than EIPFq, all methods are able to find accurate solutions reliably after ten update stages and more efficiently than using single updates.

Similar results are presented in Figure 5.3 for test problem 2. Here all methods converge towards the region of the global optimum quickly with EIPFkb being the most efficient. When finding more accurate solutions, EIvsPF is the only approach that performs reliably due to its ability to better handle the complex constraint. Selecting multiple updates from a Pareto set of solutions performs well, retaining the desirable properties of this multiobjective approach.

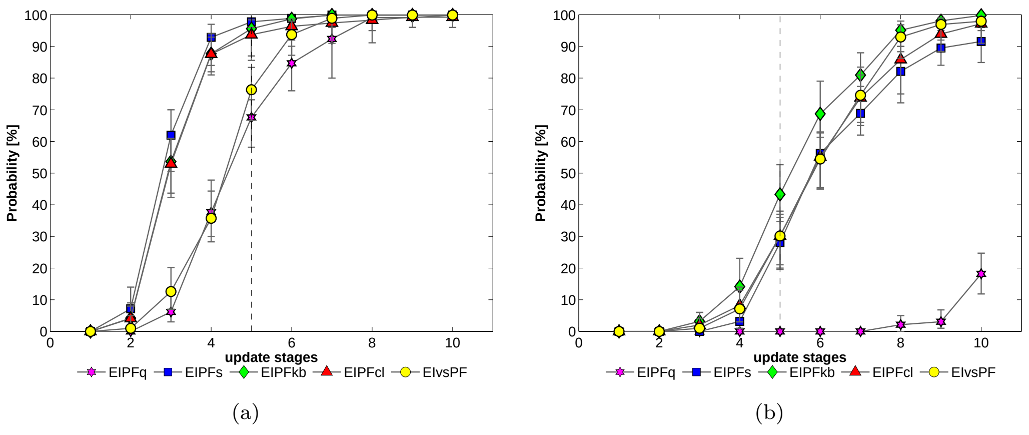


FIGURE 5.2: Test problem 1 mean probability and 95% confidence intervals of the best feasible point being within (a) 0.1 of the true optimum value and (b) 0.01 of the true optimum value.

Figure 5.4 illustrates results obtained on test problem 3. EIPFkb, EIPFcl and EIvsPF all perform very efficiently when finding solutions within 0.01 of the global optimum. Both EIPFq and EIPFs are less reliable on this test problem. When finding more accurate solutions, EIvsPF is again the most reliable approach but is only slightly more efficient than using single updates in Chapter 4. In fact, using EIvsPF with multiple updates is

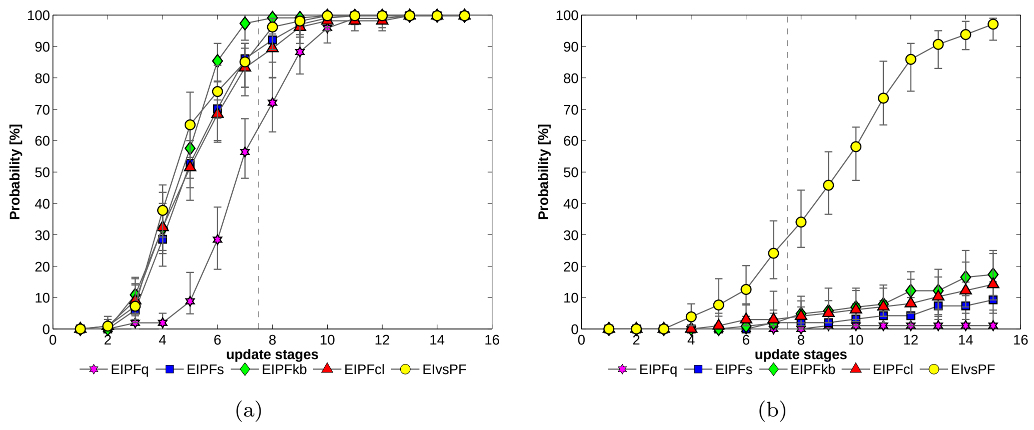


FIGURE 5.3: Test problem 2 mean probability and 95% confidence intervals of the best feasible point being within (a) 0.1 of the true optimum value and (b) 0.01 of the true optimum value.

less efficient that using EIvsPF$_\delta$ with single updates, see Figure 4.7. This highlights that on some problems, adding update points one at a time using a more suitable improvement criterion can outperform a less sophisticated approach with multiple updates.

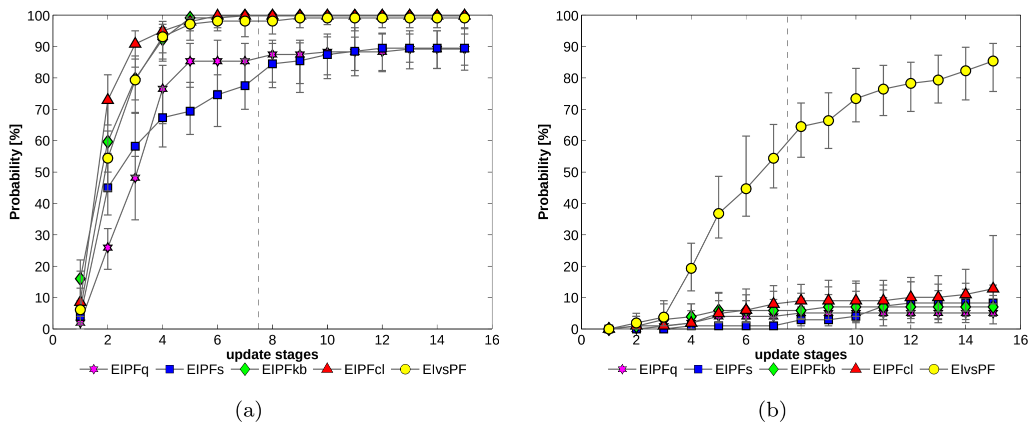


FIGURE 5.4: Test problem 3 mean probability and 95% confidence intervals of the best feasible point being within (a) 0.01 of the true optimum value and (b) 0.001 of the true optimum value.

Table 5.1 summarises the results in Figure 5.2(b), Figure 5.3(b) and Figure 5.4(b). Out of the multiple update improvement criteria, EIvsPF offers the best performance on test problem 2 and test problem 3 and very competitive on test problem 1.

Table 5.1: Summary of mean results for test problem 1, 2 and 3. Mean probability of the best feasible point being within 0.01 on test problem 1 and 2 and 0.001 on test problem 3.

	Test problem 1		Test problem 2		Test problem 3	
	No. of update stages		No. of update stages		No. of update stages	
	5	10	8	15	8	15
EIPFq	0%	18%	0%	1%	5%	5%
EIPFs	27%	92%	2%	9%	3%	8%
EIPFkb	**43%**	**100%**	5%	17%	6%	7%
EIPFcl	30%	97%	4%	15%	9%	13%
EIvsPF	31%	98%	**34%**	**97%**	**65%**	**86%**

5.4 Wingbox Structure Design Problem

Next we demonstrate the performance of these improvement criteria on a realistic engineering design problem. We aim to minimize the structural weight of a transonic wing based on a 150 seat passenger aircraft. Minimizing aircraft structural weight plays a key role in cost effective aircraft design. Essentially, any reduction in structural weight allows an increase in payload or aircraft range, directly impacting the costs of operating airlines. In Chapter 3 we introduced an empirical-based analysis tool used in conceptual wing design, relying on statistical data from previously designed aircraft. In this chapter we use physics-based analysis tools to directly measure the performance of each evaluated design.

For a fixed planform, the optimization problem involves minimizing the structural weight of a simplified wingbox structure, Figure 5.5. The wingbox structure comprises of two spars, ribs and upper and lower surface skins. The design variables to be optimized are the spar, rib and skin thickness as well as the rib pitch (the distance between each rib). This results in a modest four variable design problem with constraints placed on the maximum allowable stress (stress safety factor), the wingbox tip displacement and structural buckling.

Aerodynamic loads are computed for the wing at a cruise speed of Mach 0.8 using a CFD tool based on full potential (FP) flow theory with a viscous drag correction. An overview of aerodynamic analysis and the FP method used is provided in Appendix C. These aerodynamic loads are applied to the wingbox structure simulating steady level flight at an altitude of 11,000 m. For each wingbox design, structural stresses and displacements are computed using a linear elastic stress analysis based on the finite element method. An eigenvalue buckling prediction is also performed to compute the buckling criteria. More information on this structural analysis is presented in Appendix D and a detailed

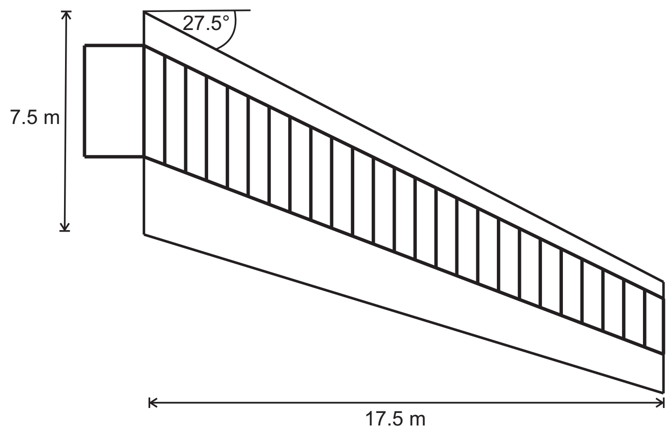


Figure 5.5: Wing planform and wingbox section.

description of the wingbox model, including geometry and mesh generation is provided in Appendix E.

Full runs using the analysis tools are used for the construction of four surrogate models, one for the wingbox weight objective and one for each constraint. The initial sample consists of 20 points spanning the four design variables generated by optimizing a Latin hypercube. A multiple update improvement criterion is used to select four updates for each of 20 update stages, totalling a further 80 evaluations. As in previous test cases, the performance of each strategy is computed using data collected from different initial samples. Due to higher computational demands of this design problem, a smaller set of 30 initial samples is used.

Since this test case is simulating a real engineering design problem, the exact location of the true optimum is unknown. The true optimum is replaced with the best found solution over all the runs performed. This is a value of 11,798 kg identified using the EIvsPF improvement criterion.

The wing planform and the best found wingbox design variables are listed in Table 5.2. Figure 5.6 shows the optimization history for this particular case as a parallel axis plot. This is a simple line plot mapping out the normalized design variables, objective and constraint values of the 100 designs evaluated. A major challenge in this problem is to find feasible designs. In this example, only 11 wing designs out of the 100 evaluated are feasible and it is clear that the optimization drives the design variables towards both geometry and structural constraints. In particular, the skin thickness is driven to a maximum value and the rib pitch is low to resist buckling. A thicker skin and more ribs results in a heavier design but in this case, this is necessary to satisfy the buckling constraint. The spar thickness is high to resist tip deflection and the rib thickness is low to reduce the structural weight.

Figure 5.7 illustrates the reliability of each method when applied to this wingbox design problem. Due to the poor performance of EIPFq in previous problems, this improvement

TABLE 5.2: Design variables, constraint values, and objective value.

Lower limit	Best design	Upper limit	Quantity
—	7.5	—	Root chord, m
—	17.5	—	Span, m
—	0.5	—	Taper ratio
—	25	—	Quarter chord sweep angle, deg
0.010	0.035	0.050	Spar thickness, m
0.010	0.016	0.050	Rib thickness, m
0.010	0.030	0.030	Skin thickness, m
0.020	0.040	0.200	Rib pitch, m
1.50	1.77	—	Stress safety factor
—	1.49	1.50	Tip displacement, m
1.00	1.01	—	Buckling eigenvalue
—	11,798	—	Wingbox weight, kg

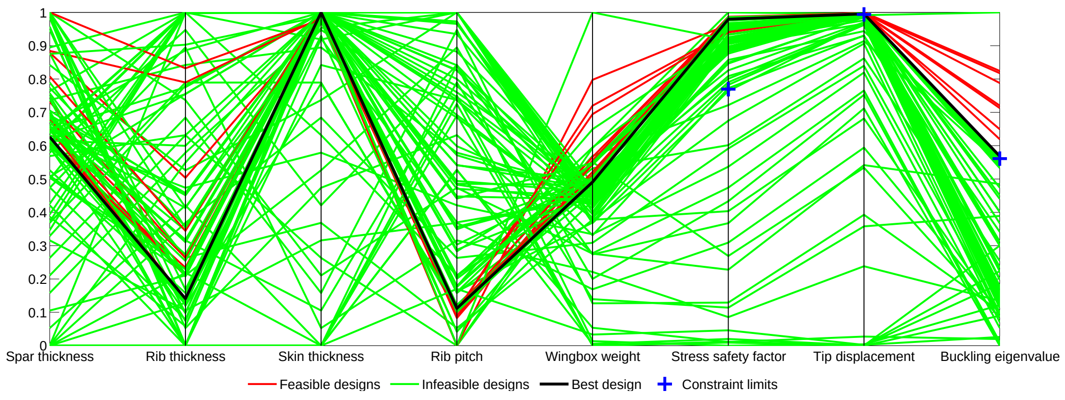
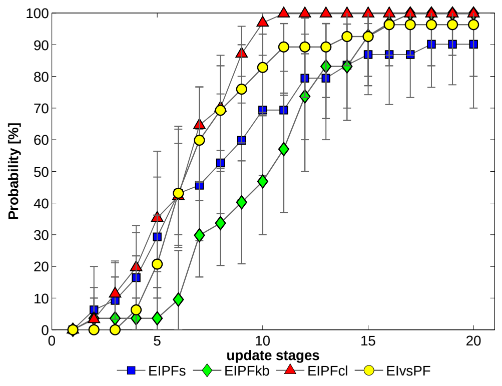
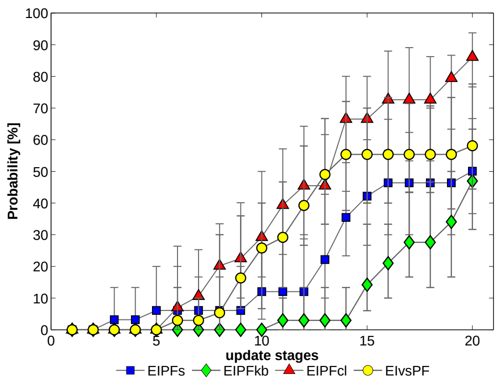


FIGURE 5.6: Parallel axis plot of the optimization history when finding the best known wingbox design with a structural weight of 11,798 kg.

criterion is not considered in this problem. Figure 5.7(a) identifies which methods are capable of consistently finding a feasible wingbox weight within 500 kg of the best known design. In this problem it is clear EIPFcl is the most efficient and identifies the solution on every occasion. The performance of EIPFkb and EIvsPF are comparable with little statistical significance between them but these methods are significantly less reliable and less efficient when identifying the optimum value within 100 kg. For this design problem, using EIvsPF does not perform as well as hoped (although it does find the best design) and EIPFcl is the most reliable and efficient approach, highlighted in 5.3. Here, EIPFcl performs well as it tends to exploit regions of good designs, working well in this problem as the objective and constraint functions are relatively well behaved and simple to model (both the stress and displacement constraints happen to be unimodal in this case). EIPFcl tends to identify more feasible designs than the other methods tested, resulting in better designs. If we consider planform changes or different loading conditions, the constraint functions may be less well behaved and EIPFcl may no longer be the best improvement criteria on this problem.



(a)



(b)

FIGURE 5.7: Wingbox design problem mean probability and 95% confidence intervals of the best feasible design within (a) 500 kg of the best known value and (b) 100 kg of the best known value.

TABLE 5.3: Summary of mean results for the wingbox design problem. Mean probability of finding the best feasible design within 100 kg of the best known solution.

	No. of update stages	
	10	20
EIPFs	13%	51%
EIPFkb	0%	47%
EIPFcl	**31%**	**87%**
EIvsPF	28%	61%

5.4.1 A Note on Computational Cost

An important consideration in this test problem is not only the accuracy and efficiency of each method but also the cost of searching each improvement criterion. In most problems tested so far, the cost of searching the improvement criteria to find updates is considered insignificant when compared to the cost of evaluating expensive analysis codes. In reality, the cost of searching for updates themselves may become significant, especially when searching for multiple updates. Since EIPFs, EIPFkb and EIPFcl all find multiple updates sequentially, using q single objective searches, this may be considered more expensive than using a single multiobjective search to identify multiple updates. This is shown in Table 5.4 where we give the average CPU cost associated with finding multiple updates on this wingbox design problem. This assumes each single objective search is run using a 10,000 iteration GA followed by a 5,000 iteration DHC. The multiobjective search is run using NSGA-II with a population of 200 for 100 generations.

TABLE 5.4: Improvement criteria computational cost.

Improvement criterion	$q = 1$	$q = 2$	$q = 3$	$q = 4$
EIPFs	40 sec	80 sec	121 sec	162 sec
EIPFkb	39 sec	79 sec	119 sec	159 sec
EIPFcl	40 sec	79 sec	120 sec	158 sec
EIvsPF	72 sec	72 sec	72 sec	72 sec

Clearly, when searching for a single update, running a single objective search is less time consuming than a multiobjective search. The difference in CPU cost becomes more significant when selecting more updates and running a single multiobjective search is less expensive. In this particular problem when $q = 4$, running multiple single objective searches is double the expense of a single multiobjective search. This makes little difference to the complete optimization (roughly 30 minutes), however, in Chapter 8 we use EIPFkb and EIvsPF to optimize the wingbox weight of many different wing designs and we see that the extra cost of employing EIPFkb over a large number of runs can become significant.

5.5 Conclusions

This chapter investigates different improvement criteria suitable for selecting multiple updates to be evaluated in parallel. Existing multiple update strategies, including Schonlau's sequential design in stages and Ginsbourger's Kriging believer and constant liar, are extended to handle constrained problems using the probability of feasibility introduced in Chapter 3. Two alternative improvement criteria that select multiple updates based on a single search are also tested. The first of these methods uses clustering to select multiple updates from a full history of a global search. The other uses a multiobjective approach introduced in Chapter 4, selecting multiple updates from a Pareto set of solutions.

On the three artificial test problems, EIPFkb, EIPFcl and EIvsPF prove to be the most promising methods and are capable of identifying the region of the global optimum very efficiently. Based on evaluating four updates in parallel, these methods find the region of the global optimum with a significant reduction in the total evaluation wall clock time, when compared to using single updates. Significant time saving can also be made when finding more accurate solutions, but at the expense of a significant increase in CPU cost. In some cases, the requirement for additional computational resources may not justify the use of multiple updates. This is likely to be the case if there is a budget on the total CPU time used. We also found that in test problem 3, using multiple updates offered very little gain over using the single update methods investigated in previous chapters.

This is an important consideration and the designer may indeed find adequate results using single updates even if resources for parallel updates exist, however, this is less likely to be the case when tackling larger problems.

The importance of selecting the correct improvement criteria is highlighted on the wing-box design problem where EIPFcl proved to be the most efficient and reliable method. Although the EIvsPF is expected to generalise well to different design problems with a modest number of design variables, this method remains relatively undeveloped and further improvements may yet be made. This may include some of the enhancements discussed in Chapter 4 but may also concentrate on different strategies for selecting multiple updates from the Pareto front.

Chapter 6

Surrogate-Based Multiobjective Optimization

In the last three chapters we have investigated different improvement criteria suitable for handling design constraint in surrogate-based optimization. In many real design problems the designer is also concerned about handling multiple, conflicting objectives and seeks a set of optimal designs that all offer some trade-off between the different goals. In aid of extending surrogate-based methods further, this chapter reviews some useful improvement criteria that can be used to handle multiple objectives in surrogate-based optimization.

6.1 Dealing with Multiple Objectives

A number of authors have used surrogate models to aid multiobjective optimization in a variety of ways, see Santana-Quintero et al. [2010] and Knowles and Nakayama [2008]. Different levels of approximation can be related to the associated cost of the true function evaluation [Jin, 2005]. For example, if the run time of an individual simulation is only a few seconds, it may still be considered expensive since directly solving the problem using evolutionary algorithms will typically need tens of thousands of function evaluations. Conversely, the cost of updating and retuning a number of global surrogates may be equally great. For function evaluations with this associated cost a common approach is to periodically use surrogates to assist evolutionary algorithms as a way to screen out poor individuals in the population [Ray et al., 2009, Emmerich, 2005].

In problems that incur function evaluation times in the order of hours, or sometimes days, the fitting costs of surrogate models is small compared to the overall cost of a large number of function evaluations. For function evaluations with this magnitude of cost, it is necessary to use surrogate models as a replacement to the true simulations for further function evaluations. A simple approach is to replace the true objective functions using the surrogate models and proceed using a multiobjective evolutionary algorithm. This is the approach taken by Voutchkov and Keane [2006] where NSGA-II is used to construct a Pareto set of solutions and updated using one or more points from the current set of solutions. This method converges towards an accurate Pareto set in very few true function evaluations but the update strategy is equivalent to updating using the predicted optimum in the single objective case and does not guaranteed convergence towards the global Pareto set.

To encourage convergence towards a global set of solutions, Jeong and Obayashi [2005] compute the expected improvement of each individual objective. This results in multiple improvement criteria that can be searched inexpensively using a multiobjective algorithm to obtain nondominated solutions in terms of the expected improvements. Using one or more of these solutions as update points is expected to offer improvement to one or more of the objectives. As long as these updates improve on the current points in the Pareto set, a new design will be nondominated offering refinement of the Pareto front.

Voutchkov and Keane [2010] share further experience in tackling multiobjective problems and examine six different update strategies for finding good quality global Pareto sets. One strategy includes the selection of random updates to help escape local Pareto fronts whilst other strategies are based on the expected improvement and variance in the Kriging prediction of each objective. The remaining strategies use search algorithms in various ways to find suitable update points. In summary, the correct choice of update strategy was found to be highly problem dependent and spreading the evaluation budget over a combination or two or more update strategies is recommended. Other methods include those introduced by Knowles [2006] and Zhang et al. [2009] which also incorporate the expected improvement criteria. These methods are based on traditional weighted aggregation approaches which require normalization of the objectives being dealt with.

Rather than searching for a trade-off between improvement in each objective using a multiobjective search or weighted aggregation, an alternative is to seek designs that explicitly offer improvement over the current Pareto set. In the following sections we introduce current state of the art approaches that combine the improvements of each objective as a single improvement criterion. These methods are considered as the most efficient (in terms of the total number of function evaluations) and analogous to EGO.

6.2 Improvement Criteria for Multiobjective Optimization

Following the same surrogate-based approach introduced in Chapter 2, the first stage makes use of an initial sample of evaluated designs (since we have multiple objectives this may involve some multidisciplinary analysis) to build a Kriging approximation of each objective. Next the surrogates are updated with new designs found by searching some multiobjective improvement criterion. This improvement criterion should aim to update the surrogates in regions that refine the Pareto set. As before, it is important to retain a balance between exploration of the design space and exploitation of the Kriging predictions. Although the following improvement criteria can be generalized for many objectives, we limit our discussion to problems with two objectives. In this chapter we also limit the discussion to single updates. Multiple updates are discussed in more detail in the next chapter.

6.2.1 Multiobjective Probability of Improvement

One way to achieve a balance between exploiting the surrogate to improve on the current Pareto set and exploring the design space is by maximizing the multiobjective probability of improvement. For multiple objectives, the probability of a new design being an improvement is simply $P[y_1 \leq \boldsymbol{y}_1^* \cup y_2 \leq \boldsymbol{y}_2^* \cup, ..., \cup y_m \leq \boldsymbol{y}_m^*]$, where $\boldsymbol{y}_i^*$ is a set of Pareto solutions and m is the number of objectives. This is given by integrating the volume under the multivariate PDF. Since the goal is to converge towards a global set of Pareto solutions, the limits of integration can be set to consider points that dominate or augment the current Pareto set. These regions are visualized for two objectives ($m = 2$) in Figure 6.1.

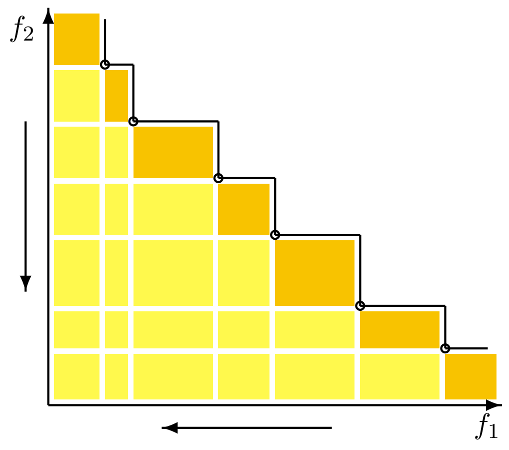


FIGURE 6.1: A Pareto set with two objectives and regions of integration. 🟧 – Regions that augment the Pareto set. 🟨 – Regions that dominate the Pareto set.

Keane [2006] derives both the augmenting and dominating multiobjective probability of improvement for two objectives. To encourage the development of well populated Pareto set and to help eliminate any gaps, regions that augment the set are considered as an improvement in this study. Generalizing for two objectives the set of Pareto solutions is,

$$\boldsymbol{y}^*_{1,2} = \left[(y_1^1, y_2^1), ..., (y_1^k, y_2^k)\right], \tag{6.1}$$

where k is the number of Pareto solutions. Integrating over the regions that augment the current set, the multiobjective probability of improvement can be expressed as

$$\begin{aligned} P[y_1 \leq \boldsymbol{y}_1^* \cup y_2 \leq \boldsymbol{y}_2^*] = &\int_{-\infty}^{y_1^1} \int_{-\infty}^{\infty} \boldsymbol{\phi}\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1 \\ &+ \sum_{i=1}^{k-1} \int_{y_1^i}^{y_1^{i+1}} \int_{-\infty}^{y_2^i} \boldsymbol{\phi}\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1 \\ &+ \int_{y_1^k}^{\infty} \int_{-\infty}^{y_2^k} \boldsymbol{\phi}\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1. \end{aligned} \tag{6.2}$$

Despite multiobjective problems having inherently correlated (mostly conflicting) objectives, here the objectives are assumed to be independent which tends to be common practice [Wagner et al., 2010]. Recent work by Svenson [2011] demonstrates little or no gain when using dependence models. Assuming independence, the multivariate PDF is simply,

$$\boldsymbol{\phi}\left(y_1, ..., y_m\right) = \prod_{i=1}^{m} \boldsymbol{\phi}\left(y_i\right) \tag{6.3}$$

and for two objectives the multivariate PDF is given as

$$\boldsymbol{\phi}\left(y_1, y_2\right) = \frac{1}{\hat{s}_1\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{y_1 - \hat{y}_1}{\hat{s}_1}\right)^2\right] \frac{1}{\hat{s}_2\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{y_2 - \hat{y}_2}{\hat{s}_2}\right)^2\right]. \tag{6.4}$$

The multiobjective probability of improvement can therefore be expressed analytically as

$$\begin{aligned} P[y_1 \leq \boldsymbol{y}_1^* \cup y_2 \leq \boldsymbol{y}_2^*] = &\, \boldsymbol{\Phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right) \\ &+ \sum_{i=1}^{k-1} \left\{ \left[\boldsymbol{\Phi}\left(\frac{y_1^{i+1} - \hat{y}_1}{\hat{s}_1}\right) - \boldsymbol{\Phi}\left(\frac{y_1^i - \hat{y}_1}{\hat{s}_1}\right)\right] \right. \\ &\left. \times \boldsymbol{\Phi}\left(\frac{y_2^i - \hat{y}_2}{\hat{s}_2}\right) \right\} \\ &+ \boldsymbol{\Phi}\left(\frac{y_2^k - \hat{y}_2}{\hat{s}_2}\right)\left[1 - \boldsymbol{\Phi}\left(\frac{y_1^k - \hat{y}_1}{\hat{s}_1}\right)\right]. \end{aligned} \tag{6.5}$$

The multiobjective probability of improvement is a concept that can be easily understood when considering the individual regions below the Pareto front. Throughout the remainder of this thesis this method is denoted as PI. As in the single objective case, the probability of improvement returns values between zero and one, so a criterion that additionally expresses the magnitude of improvement is desirable. The extension of expected improvement to multiple objectives is less intuitive and a suitable measure of improvement is less obvious. This has lead to several interpretations of the multiobjective expected improvement.

6.2.2 Euclidean-Based Expected Improvement

Keane [2006] derives the expected improvement for two objectives, where it is expressed as the first moment of the multivariate PDF integral. This is the same integral considered in the derivation of the probability of improvement and the moment arm around the current Pareto set is taken as the Euclidean distance between the centroid of this integral $(\bar{y}_1, \bar{y}_2)$ and the closest point on the current Pareto set (y_1^*, y_2^*). This leads to the following definition of the multiobjective expected improvement,

$$E[I_E(\boldsymbol{x})] = P[y_1 \leq \boldsymbol{y}_1^* \cup y_2 \leq \boldsymbol{y}_2^*]\sqrt{(\bar{y}_1 - y_1^*)^2 + (\bar{y}_2 - y_2^*)^2}, \tag{6.6}$$

where

$$\begin{aligned}\bar{y}_1 P[y_1 \leq \boldsymbol{y}_1^* \cup y_2 \leq \boldsymbol{y}_2^*] = & \int_{-\infty}^{y_1^1}\int_{-\infty}^{\infty} y_1 \boldsymbol{\phi}\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1 \\ & + \sum_{i=1}^{k-1}\int_{y_1^i}^{y_1^{i+1}}\int_{-\infty}^{y_2^i} y_1 \boldsymbol{\phi}\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1 \\ & + \int_{y_1^k}^{\infty}\int_{-\infty}^{y_2^k} y_1 \boldsymbol{\phi}\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1,\end{aligned} \tag{6.7}$$

and

$$\begin{aligned}\bar{y}_2 P[y_1 \leq \boldsymbol{y}_1^* \cup y_2 \leq \boldsymbol{y}_2^*] = & \int_{-\infty}^{y_1^1}\int_{-\infty}^{\infty} y_2 \boldsymbol{\phi}\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1 \\ & + \sum_{i=1}^{k-1}\int_{y_1^i}^{y_1^{i+1}}\int_{-\infty}^{y_2^i} y_2 \boldsymbol{\phi}\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1 \\ & + \int_{y_1^k}^{\infty}\int_{-\infty}^{y_2^k} y_2 \boldsymbol{\phi}\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1.\end{aligned} \tag{6.8}$$

Analytical expressions for the Euclidean-based expected improvement can be found in Keane [2006]. Although an improvement function is not explicitly given, the Euclidean-based improvement can be considered equivalent to,

$$I_E = \min_{j=1,\ldots,k} \sqrt{\left(y_1 - y_1^j\right)^2 + \left(y_2 - y_2^j\right)^2}. \tag{6.9}$$

where $j = 1, \ldots, k$ and k is equal to the number of solutions in the current Pareto set $\boldsymbol{y}^*_{1,2}$.

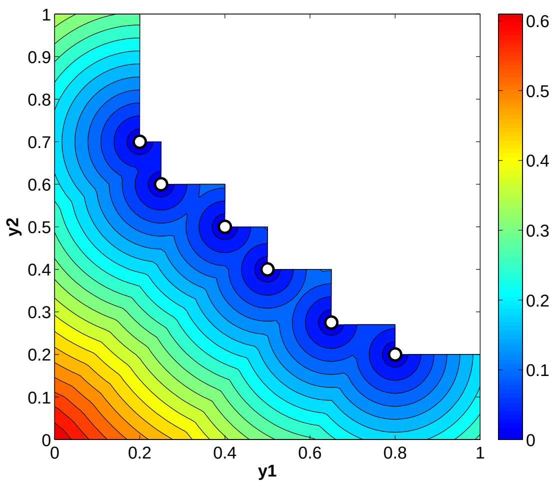


FIGURE 6.2: The Euclidean-based improvement.

Figure 6.2 illustrates the Euclidean-based improvement for an example Pareto set with two objectives. In this interpretation of the multiobjective expected improvement, the further the predicted update point lies from points in the current Pareto set, the further the centroid will lie from the front. This gives a larger expected improvement for points that are predicted to dominate the front the most. It also rewards points that lie in significant gaps in the Pareto front. Basing the improvement criterion on domination will drive both objectives to be improved irrespective of scaling but when calculating which Pareto set member lies closest to the front, scaling of the objectives becomes important. This improvement criterion is denoted as EIe for the remainder of this thesis.

6.2.3 Hypervolume-Based Expected Improvement

Since the main goal of multiobjective optimization is to converge towards a Pareto set, it is sensible to base the improvement criterion on some Pareto set quality indicator. Calculating the hypervolume is a popular method to assess the quality of a Pareto set [Deb, 2001, Coello Coello et al., 2007]. The hypervolume is the volume in the objective space dominated by the Pareto set, illustrated in Figure 6.3 for a Pareto set $\boldsymbol{y}^*_{1,2}$. A larger hypervolume suggests a greater convergence to the global set but also favours well spread Pareto sets that are highly populated.

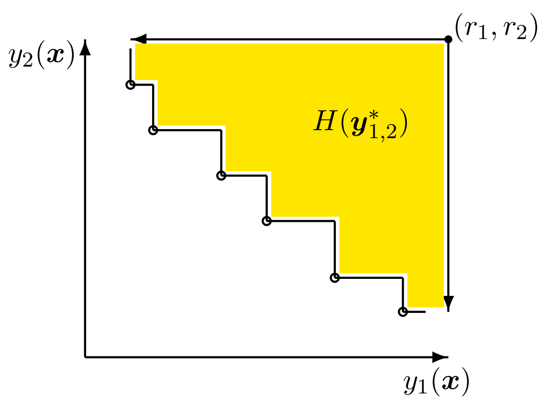


FIGURE 6.3: The hypervolume indicator. 🟨 – hypervolume given the Pareto set $\boldsymbol{y}^*_{1,2}$.

Several methods have generalised expected improvement to multiple objectives using the hypervolume indicator, see Ponweiser et al. [2008a] and M. Emmerich [2006]. For two objectives, the hypervolume-based expected improvement is defined as

$$E[I_H(\boldsymbol{x})] = \int_R I_H \phi\,(y_1, y_2)\,\mathrm{d}y_1 \mathrm{d}y_2. \tag{6.10}$$

where R is the region of integration and I_H is the improvement in hypervolume given as,

$$I_H = H\left(\boldsymbol{y}^*_{1,2} \cup \boldsymbol{y}_{1,2}\right) - H\left(\boldsymbol{y}^*_{1,2}\right). \tag{6.11}$$

Figure 6.4 illustrates the hypervolume-based improvement for an example Pareto set with two objectives. Clearly, a larger improvement is achieved in regions further from the Pareto set, encouraging new updates that are nondominated. One concern is that this improvement may not prioritize points at either end of the Pareto set, making it difficult to find a full spread of Pareto solutions. This is highlighted in Figure 6.4 where there is little improvement at each end of the Pareto set.

In recent work Emmerich et al. [2011] provide an analytical expression for the exact computation of the hypervolume-based expected improvement. The calculation of the hypervolume requires a suitable reference point (r_1, r_2) and is also sensitive to the relative scaling of the objectives. In comparison to the Euclidean-based improvement there is a computational burden associated with computing the hypervolume, becoming a practical concern depending on the problem being tackled. In this study the hypervolume-based expected improvement is denoted as EIh.

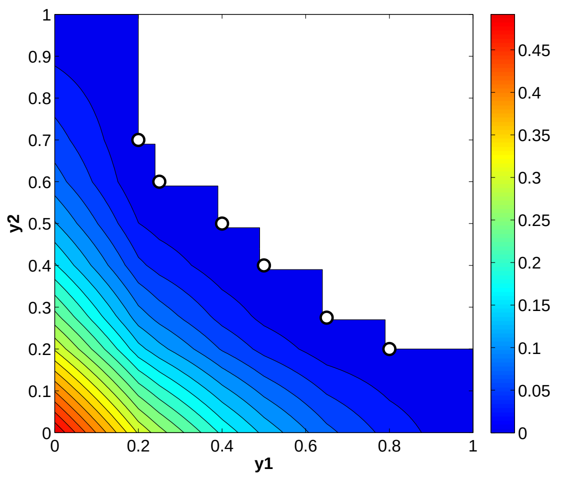


FIGURE 6.4: The hypervolume-based improvement.

6.2.4 Maximin Expected Improvement

Another interpretation of multiobjective expected improvement is the maximin expected improvement. This was first considered by Bautista [2009] and revisited by Svenson [2011] who provides an analytical expression for its computation. The maximin expected improvement is defined as

$$E[I_M(\boldsymbol{x})] = \int_R I_M \phi\,(y_1, y_2)\,\mathrm{d}y_1\mathrm{d}y_2, \tag{6.12}$$

which uses the maximin improvement function

$$I_M = -\max_{j=1,\ldots,k}(w_j)\,. \tag{6.13}$$

Here, w_j is a vector containing the minimum distances between (y_1, y_2) and the current Pareto set, given as

$$w_j = \min_{j=1,\ldots,k}\begin{bmatrix} (y_1 - y_1^j) \\ (y_2 - y_2^j) \end{bmatrix}. \tag{6.14}$$

Figure 6.5 illustrates the maximin improvement for an example Pareto set with two objectives.

This maximin improvement can be quickly computed and is directly related to Pareto dominance. This provides a simple indication of improvement where dominance is expected when $I_M > 0$. This is free from any external parameters and computing the expected improvement does not require any heavy computation. Since this is essentially

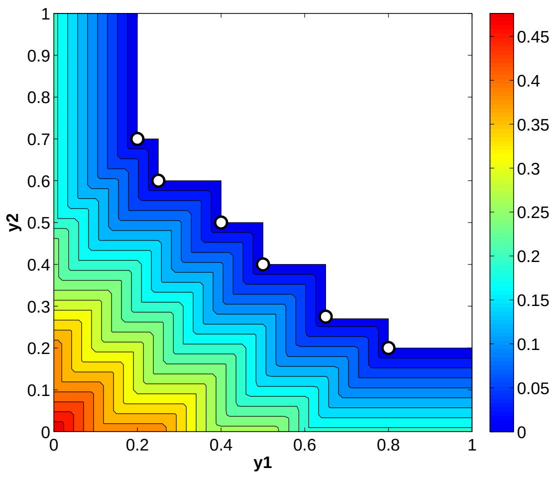


FIGURE 6.5: The maximin improvement.

a distance based metric, the relative scaling of objectives remains important. Here, the maximin expected improvement is denoted as EIm.

6.3 Scale Dependence

Scaling of the objectives is often unavoidable when considering improvement metrics based on multiple objectives. When dealing with real design problems, this necessary scaling is undesirable since the limits of the objective outputs are unknown. In such cases it is necessary to use some form of empirical scaling based on information gained from previous samples. Svenson [2011] considers scale invariance as an important property of improvement criteria. Scale dependence becomes more of an issue when dealing with constraints as objectives may be empirically scaled between all observed responses or all observed feasible responses. For some problems this scale dependence can make scale invariant methods advantageous.

In this study, all test problems are treated as real engineering design problems where the limits of outputs are unknown and scaling can only be based on previously observed samples. Scaling objectives in this way can poorly represent the limits of the design space. This may be mitigated by using a good initial sample but there is still a risk that the scaling misrepresents the problem if the sample space is highly multimodal. Nonetheless, this is a necessary step to extend many of these methods to deal with real engineering design problems. Here, all scale dependent methods are scaled between zero and one using the maximum and minimum output values observed so far.

The scale dependent methods tested include the Euclidean-based expected improvement, hypervolume-based expected improvement and the maximin expected improvement. For the hypervolume-based expected improvement the reference point on each iteration will be represented by the scaled values $[1, 1]$. The improvement based on the hypervolume will change as the reference point changes with its relative scaling. Having a high dependence on this reference point is one drawback of the hypervolume-based approach. When dealing with constraints, all objective values for these three methods are scaled between zero and one, regardless of feasibility.

6.4 Comparison Metrics

The performance of a multiobjective search can be measured in a number of ways. The goal is to find a global set of Pareto solutions but these should also be well populated, have a good spread and be well spaced out. Due to these various goals, it is not obvious how Pareto sets should be compared. There are a number of Pareto set quality indicators in the literature, all with their merits and disadvantages, see Deb [2001] or Coello Coello et al. [2007]. A Pareto set quality indictor already introduced is the hypervolume indicator which has been found to be a reliable base to measure the quality of Pareto sets considered in this study.

Although the problems tested are treated as expensive, for testing purposes they are in fact quick to compute. This enables the true Pareto set to be identified and a suitable reference point to compute the hypervolume can also be found. Here, the true Pareto set for each problem is computed based on a grid of 100×100 evaluated designs. A baseline hypervolume can then be computed for the true Pareto set. An improvement criterion with good performance will be indicated by a measured hypervolume that is close to the baseline hypervolume. Here, the difference between the measured and baseline hypervolume is referred to as the hypervolume error or H_{err}.

When comparing the performance based on the hypervolume, it is expected that this will favour EIh. To help avoid any bias between EIh and the hypervolume performance indictor, the performance is also measured based on another Pareto set quality indicator. In this study we use the ϵ-indicator introduced by Zitzler et al. [2003]. Given a Pareto set B that is an approximation of the Pareto set A, the ϵ-indicator or ϵ_{err} is simply the smallest value that must be added to all elements of A so that it is dominated by the approximation set B. This can be thought of the error or the distance between the approximation set and the true Pareto front. A good approximation of the true Pareto set is indicated by a small value given by the ϵ-indicator. Svenson documents an interesting relationship between EIm and the ϵ-indicator so a bias may be expected between this improvement criterion and the ϵ-indicator.

When comparing different methods, the designer will also be concerned about the reliability and efficiency of each method. Each improvement criterion is therefore tested 100 times, each time initiated with a different initial sample (a random Latin hypercube). As a result, the reliability of each approach can be represented in terms of a probability. This probability characterizes the consistency of each method in achieving a value, given by the Pareto set quality indicator, to a prescribed accuracy. For example, a Pareto set that gives a hypervolume error below 0.01, 90 times out of 100, will have a probability of 90% at this level. This performance indicator can be computed after every update so the most efficient improvement criteria can be identified. To check statistical significance between methods, bootstrapping is used to compute an accurate representation of the mean performance and 95% confidence intervals.

An example Pareto set for each improvement criterion is also illustrated. This provides a visual comparison of each method based on a single run initiated with an optimized Latin hypercube. These figures are referenced in the main text but found in Appendix A in favour of space.

6.5 Artificial Test Problems

We now introduce two further artificial test problems suitable for comparison in multiobjective optimization. As mentioned previously, these problems are treated as expensive but for testing purposes these are quick to compute. This allows an insight into the design space and a good approximation of the true Pareto set. To avoid any multidimensional scaling issues when constructing and searching the surrogates, in all cases the design space is scaled into the unit cube $[0, 1]^d$, where d is the number of design variables.

Test problem 4 is a modified version of the multimodal, multiobjective test problem found in Deb [1999]. Given here as,

$$\begin{aligned}
&\text{Minimize} \quad f_1(\boldsymbol{x}) = a_1, \\
&\text{Minimize} \quad f_2(\boldsymbol{x}) = \frac{2}{a_1} - \frac{1}{a_1}\exp\left[-\left(\frac{a_2 - 0.2}{0.06}\right)^2\right] - \frac{0.8}{a_1}\exp\left[-\left(\frac{a_2 - 0.6}{0.4}\right)^2\right],
\end{aligned} \tag{6.15}$$

where $a_1 = 0.9x_1 + 0.1$, $a_2 = 0.9x_2 + 0.1$ and $x_1, x_2 \in [0, 1]$.

This test problem has both a local and global front caused by the multimodal properties of f_2, see Figure 6.6. This is a challenging problem to optimize using surrogates and the improvement criteria are required to explore the design space sufficiently to ensure this global region is identified. The global Pareto set is illustrated in Figure 6.7. The dense region of points away from the global set is the local Pareto set.

Test problem 5 is based on the WSNL problem introduced in Williams et al. [2010]. This is modified to give the multiobjective problem,

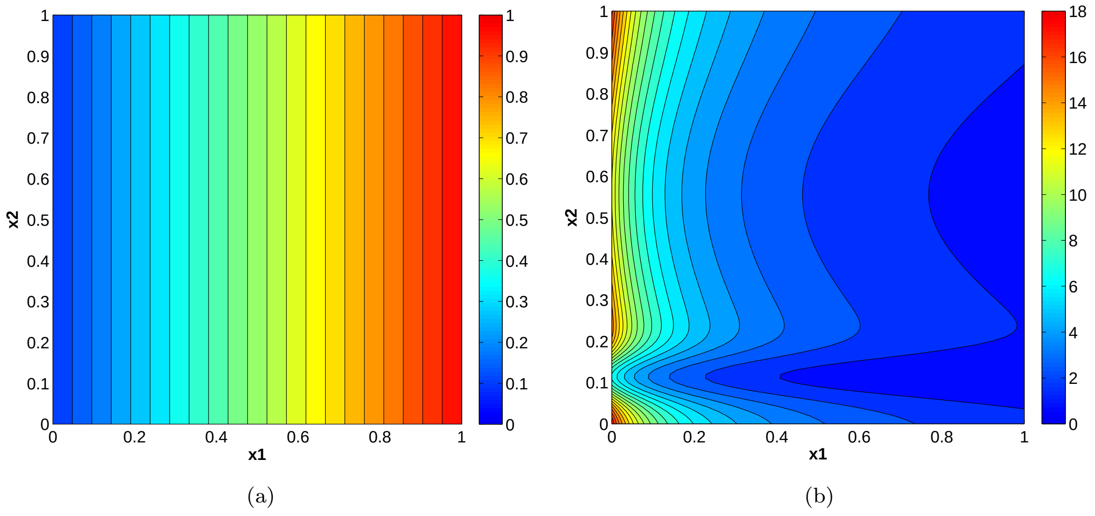


FIGURE 6.6: Test problem 4 design space for (a) $f_1(\boldsymbol{x})$ and (b) $f_2(\boldsymbol{x})$.

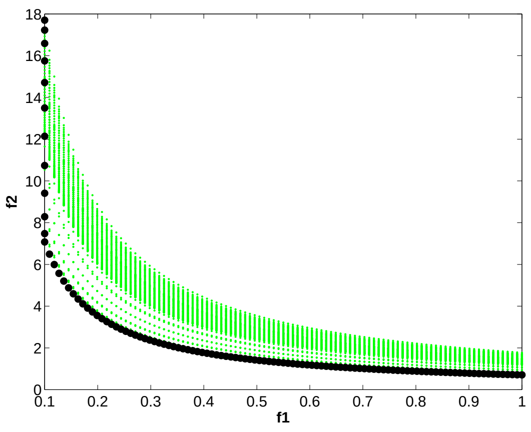


FIGURE 6.7: Test problem 4 100×100 grid of evaluated points with ● – global Pareto set.

$$
\begin{aligned}
\text{Minimize}\quad & f_1(\boldsymbol{x}) = \left(b_2 - \frac{5.1}{4\pi^2}b_1^2 + \frac{5}{\pi}b_1 - 6\right)^2 + 10\left[\left(1 - \frac{1}{8\pi}\right)\cos\left(b_1\right) + 1\right], \\
\text{Minimize}\quad & f_2(\boldsymbol{x}) = -\sqrt{\left(10.5 - b_1\right)\left(b_1 + 5.5\right)\left(b_2 + 0.5\right)} - \frac{1}{30}\left(b_2 - \frac{5.1}{4\pi^2}b_1^2 - 6\right)^2 \\
& \quad - \frac{1}{3}\left[\left(1 - \frac{1}{8\pi}\right)\cos\left(b_1\right) + 1\right],
\end{aligned}
\tag{6.16}
$$

where $b_1 = 15x_1 - 5$, $b_2 = 15x_2$ and $x_1, x_2 \in [0, 1]$.

This problem demonstrates a different challenge since the multimodality of f_1 makes it difficult to find a Pareto set with a full spread of points. Figure 6.8 and Figure 6.9 show the design space and the resulting Pareto set for this test problem.

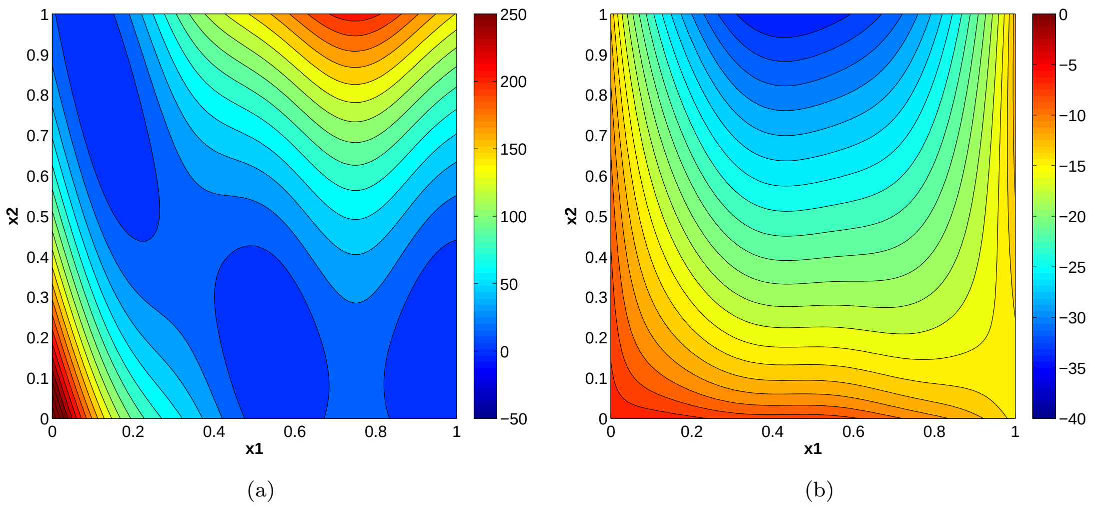


FIGURE 6.8: Test problem 5 design space for (a) $f_1(\boldsymbol{x})$ and (b) $f_2(\boldsymbol{x})$.

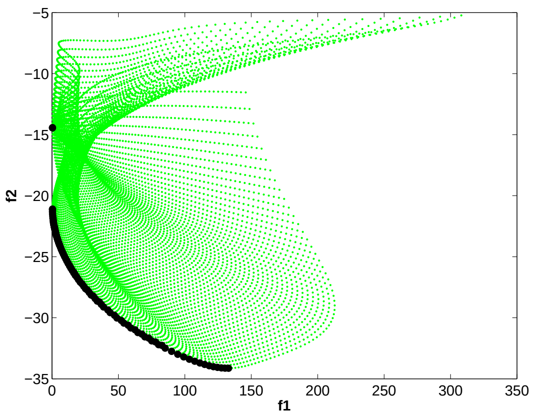


FIGURE 6.9: Test problem 5 100×100 grid of evaluated points with ● – global Pareto set.

6.6 Nowacki Beam Design Problem

The first of engineering test problems considers the design of a tip-loaded encastre cantilever beam with breadth b and height h. This design problem originally introduced by Nowacki [1980] has been previously studied as a multiobjective design problem by Keane [2006] and more recently by Svenson [2011]. The beam has a fixed length $l = 1.5$ m with a tip load $F = 5$ kN. The beam is made from a mild steel with yield stress $\sigma_Y = 240$ MPa, Young's modulus $E = 216.62$ GPa, Poisson ratio $v = 0.27$ and shear modulus calculated as $G = 86.65$ GPa. The multiobjective problem aims to minimize both the cross sectional area and the bending stress subject to a number of constraints, see Table 6.1.

The design space for this problem and the resulting Pareto set is pictured in Figure 6.10 and Figure 6.11, demonstrating a clear trade-off between minimum cross sectional area

TABLE 6.1: Design variables, objectives and constraints for Nowacki beam design problem.

Lower limit	Upper limit	Units	Type	Quantity
0.005	0.05	m	Variable	Breadth b
0.02	0.25	m	Variable	Height h
—	—	m^2	Objective	Cross-sectional area $A = bh$
—	240	MPa	Objective/constraint	Bending stress $\sigma_B = 6Fl/(bh^2)$
—	0.005	m	Constraint	Tip deflection $\delta = Fl^3/(3EI_Y)$, where $I_Y = bh^3/12$
—	10	—	Constraint	Height to breadth ratio.
—	120	MPa	Constraint	Shear stress $\tau = 3F/(2bh)$
10000	—	N	Constraint	Critical force for twist buckling, $F_{CRIT} = (4/l^2)\sqrt{GI_TEI_Z/(1-v^2)}$, where $I_T = (b^3h + bh^3)/12$ and $I_Z = b^3h/12$

and minimum bending stress. Since the design space is bounded by constraints, the improvement criteria are required to converge towards feasible designs.

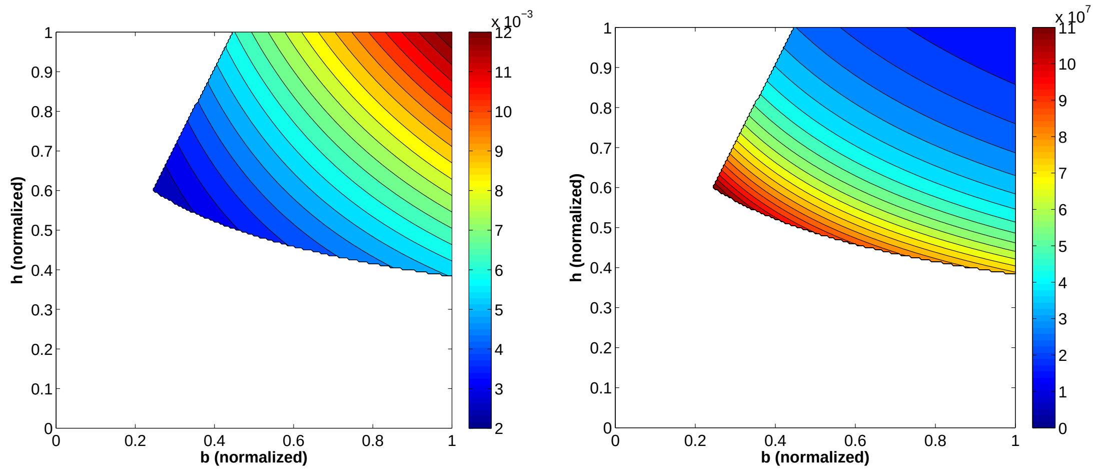


FIGURE 6.10: Nowacki beam design space for (a) cross-sectional area, m^2 and (b) bending stress, MPa.

Handling constraints adds further challenges to the optimization since the improvement criteria are required to balance exploration and exploitation of both the objectives and all the constraints surrogates. Here the constraints are assumed to be expensive to evaluate and also replaced with surrogates. The constraints are handled by modifying the multiobjective improvement criteria using the probability of feasibility. This can be multiplied with the multiobjective probability of improvement or the different formulations of multiobjective expected improvement in the same manner as the single objective case.

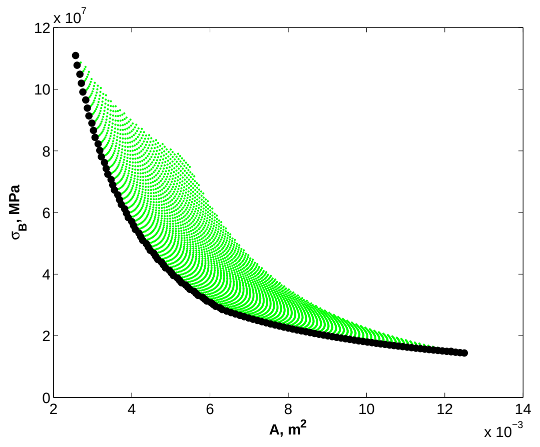


FIGURE 6.11: Nowacki beam design problem 100 × 100 grid of evaluated points with ● – global Pareto set.

6.7 Satellite Boom Design Problem

The final design problem is based on a simplified satellite boom structure pictured in Figure 6.12, originally introduced in Keane and Bright [1996].

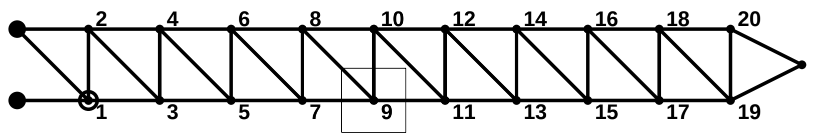


FIGURE 6.12: Baseline satellite boom geometry. ●- encastre nodes, **O** - excited node and □ - bounds on node 9.

The aim of this design problem is to maximize the vibration attenuation at the tip of the boom whilst minimizing the boom weight. This can be achieved by manipulating the geometry of the structure, controlled by the x- and y-coordinates of the nodes 1-20. Here, only node 9 is varied, resulting in two design variables. The analysis of the structure uses 42 Euler-Bernoulli beams, with three finite elements per beam and a mass per unit length of 2.74 kgm^{-1}. Figure 6.13 shows the band-averaged attenuation and boom weight for different coordinates of node 9, bounded by the box shown in Figure 6.12.

This clearly demonstrates a difficult real design problem where significant differences between global and local fronts exist. Figure 6.14 shows objective values evaluated using a grid of 100×100 points. For simplicity, rather than maximizing the attenuation, the negative attenuation in minimized. The Pareto points with minimum negative attenuation are particularly hard to find and correspond to the global maximum in Figure 6.13(a).

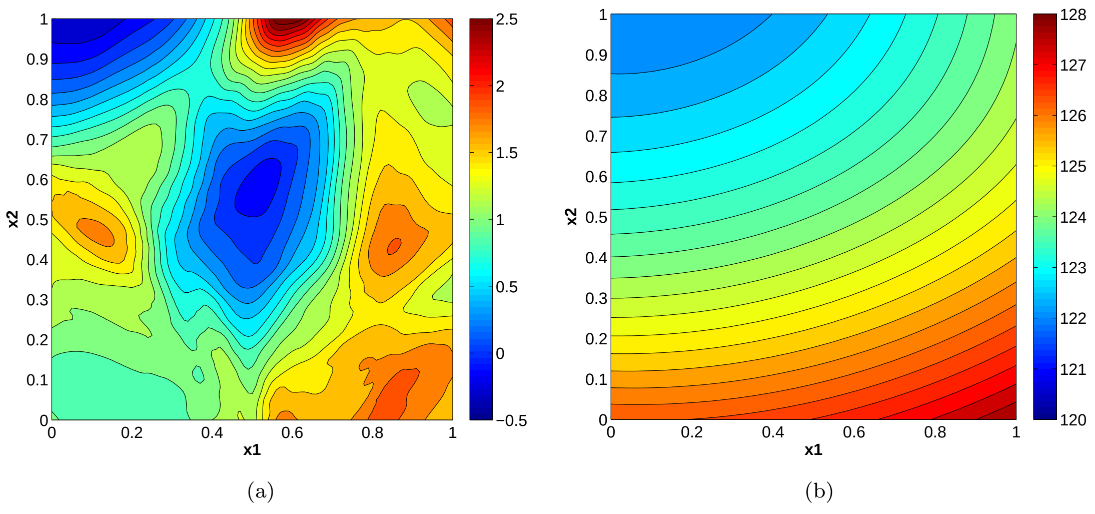


FIGURE 6.13: Satellite boom design space (a) band averaged attenuation and (b) boom weight, kg.

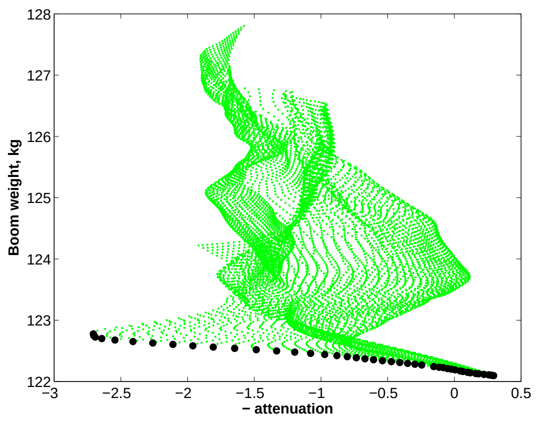


FIGURE 6.14: Satellite boom design problem 100×100 grid of evaluated points with ● – global Pareto set.

6.8 Results and Discussion

As pointed out in Chapter 3, some improvement criteria for single objective problems can be highly multimodal and difficult to search. Maximizing improvement criteria based on multiple objectives is no exception and often requires an exhaustive global search. To ensure the maximum improvement is found on every iteration, a heavy hybrid search is used. With the exception of the hypervolume-based expected improvement, all the improvement criteria are maximized using a 10,000 iteration GA followed by 5,000 iteration DHC. Due to the hypervolume-based expected improvement being more computationally expensive to search, the search effort is halved using a 5,000 iteration GA and 2,500 iteration DHC. Although the search effort is reduced, this is still substantial and yields a similar run time to the other searches on most problems, providing a fair comparison.

6.8.1 Artificial Test Problems

Test problem 4 uses an initial sample of 20 points and 20 further updates whilst the more complex test problem 5 uses an initial sample with 20 points and 40 further updates. Figure 6.15 shows the results obtained when the different improvement criteria are applied to test problem 4. When comparing the hypervolume error, Figure 6.15(a), all the improvement criteria tested share a similar performance. The performance of each method is supported when also comparing the ϵ-indicator, accept EIe which has a poorer performance.

Figure A.1 illustrates some example Pareto sets obtained when initiating the search using an optimized Latin hypercube. In this particular example, both PI and EIh struggle to converge towards the global Pareto set. PI tends to over exploit good designs and leads to the clustering of points along the Pareto front. Visually, EIe identifies the best Pareto set with a good spread of solutions that are well spaced out.

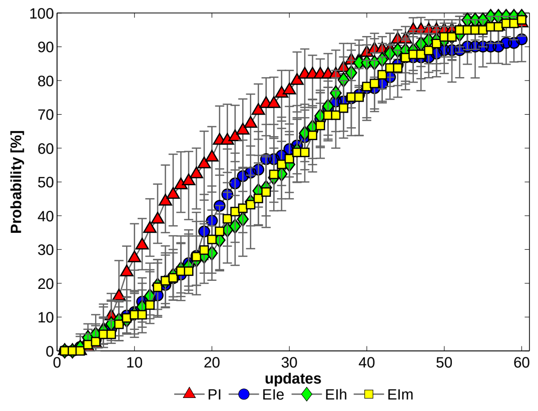
In test problem 5, Figure 6.16, the performance of each improvement criterion becomes more distinguished. Both the hypervolume error and the ϵ-indicator suggest EIh is the most efficient and reliable method tested. Although less efficient, EIm also proves to be a reliable improvement criterion. Both EIh and EIm are superior to PI and EIe which are unable to identify Pareto sets with a hypervolume error less than 0.02. This poor performance is supported by the ϵ-indicator and also documented by Svenson on an (almost) identical test problem. Whilst PI is expected to have a poorer performance, the reasons for EIe performing so poorly are uncertain. Wagner et al. [2010] and Svenson [2011] suggest that EIe is not monotonic with respect to Pareto dominance. This suggests that, in special cases, less dominating regions of the design space can offer a larger improvement. This theoretical property may account for the poor performance on some problems.

Figure A.4 shows some example Pareto sets for this problem. Visually, EIh and EIm both provide a good set of solutions and the poor performance of PI on this problem is clear. In this particular example EIe performs well but does not have a full set of global solutions.

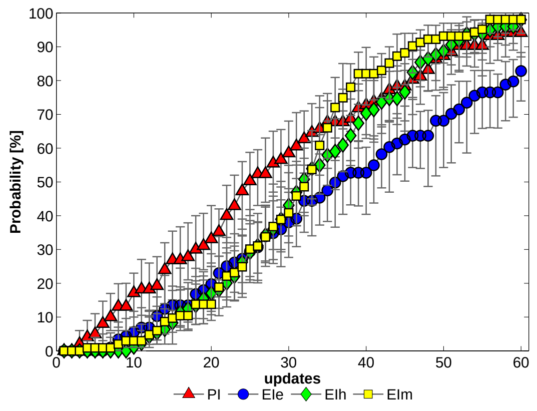
Table 6.2 provides a summary of the best improvement criteria investigated on test problem 4 and test problem 5 with EIhg and EImg being the most promising.

6.8.2 Nowacki Beam Design Problem

The Nowacki beam design problem uses an initial sample of 20 points with 40 further updates. Figure 6.17 shows the performance of each improvement criteria on this engineering design problem. Unlike test problem 5, EIe now performs very well, supported by

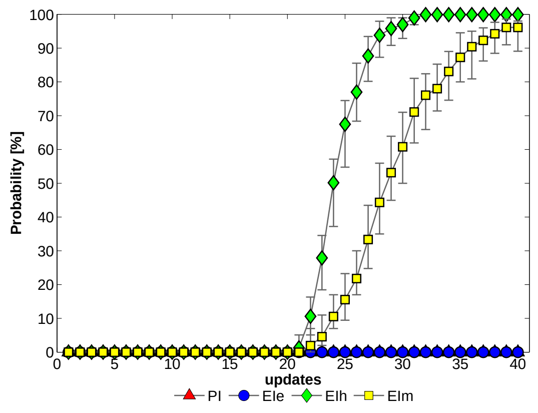


(a)

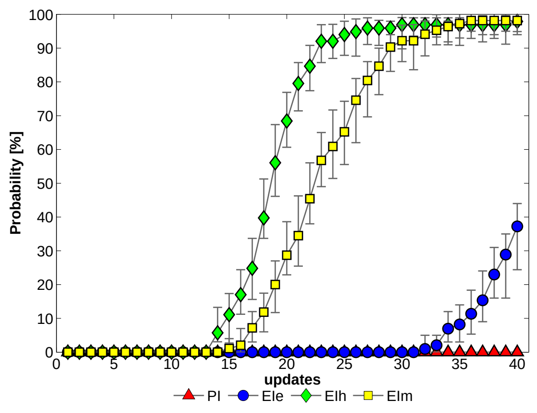


(b)

FIGURE 6.15: Test problem 4 mean probability and 95% confidence intervals of the (a) hypervolume error being less than 0.07 and (b) the ϵ-indicator less than 0.08.



(a)



(b)

FIGURE 6.16: Test problem 5 mean probability and 95% confidence intervals of the (a) hypervolume error being less than 0.02 and (b) the ϵ-indicator less than 0.04.

TABLE 6.2: Summary of mean results for test problem 4 and 5 with the hypervolume error, H_{err}, and ϵ-indicator, ϵ_{err}.

	Test problem 4				Test problem 5			
	$H_{0.07}$		$\epsilon_{0.08}$		$H_{0.02}$		$\epsilon_{0.04}$	
	No. of updates		No. of updates		No. of updates		No. of updates	
	30	60	30	60	20	40	20	40
PI	**77%**	97%	**58%**	94%	0%	0%	0%	0%
EIe	61%	93%	37%	81%	0%	0%	0%	35%
EIh	55%	**99%**	42%	**98%**	0%	**100%**	**68%**	**98%**
EIm	57%	98%	41%	**98%**	0%	65%	29%	**98%**

both quality indicators. The performance of PI remains poor whilst EIh and EIm share a very similar performance, although less reliable than EIe on this particular problem.

Figure A.7 illustrates some example Pareto sets for this problem. Both PI and EIm have over exploited regions close to the centre of the Pareto front. EIh provides a good set of optimal solutions but the Pareto set is poorly populated. EIe provides a set of solutions with a good spread and are well spaced. In this problem, solutions that lie along the Pareto front also lie on constraint boundaries. Better constraint handling is likely to result in a better convergence of all the improvement criteria tested.

Table 7.2 provides a summary for the results illustrated in Figure 6.17. In this problem EIe is clearly the most efficient and reliable method tested but performance enhancements can still be made.

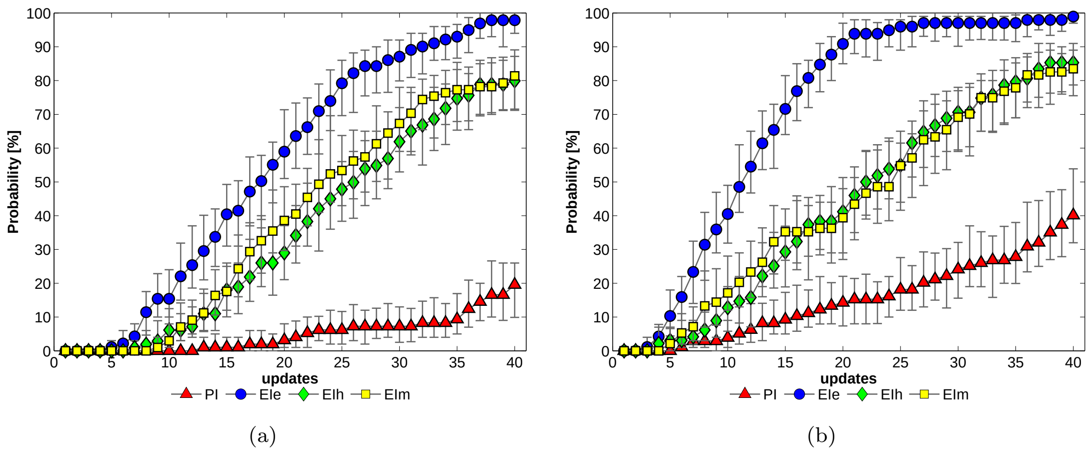


FIGURE 6.17: Nowacki beam problem mean probability and 95% confidence intervals of the (a) hypervolume error being less than 0.07 and (b) the ϵ-indicator less than 0.11.

TABLE 6.3: Summary of mean results for the Nowacki beam design problem with the hypervolume error, H_{err}, and ϵ-indicator, ϵ_{err}.

	$H_{0.07}$		$\epsilon_{0.11}$	
	No. of updates		No. of updates	
	20	40	20	40
PI	3%	19%	14%	41%
EIe	**57%**	**98%**	**91%**	**99%**
EIh	29%	80%	41%	85%
EIm	39%	81%	39%	83%

6.8.3 Satellite Boom Design Problem

The satellite boom design problem is optimized based on an initial sample of 20 points with 60 updates. In this problem, EIh again appears superior to the other methods tested and supported by both quality indicators, Figure 6.18. The performance of PI remains poor whilst the performance of EIe and EIm appear to be dependent on the quality indicator. This highlights the importance of choosing an appropriate comparison metric when comparing the performance of the improvement criteria.

Figure A.10 illustrates some example Pareto sets for this problem. All methods are capable of finding a global set of Pareto solutions but in this case, PI results in a poor spread of points. EIh and EIe both provide a visually good set of solutions but based on the performance indicators used, EIh offers the best performance and highlighted in Table 7.3.

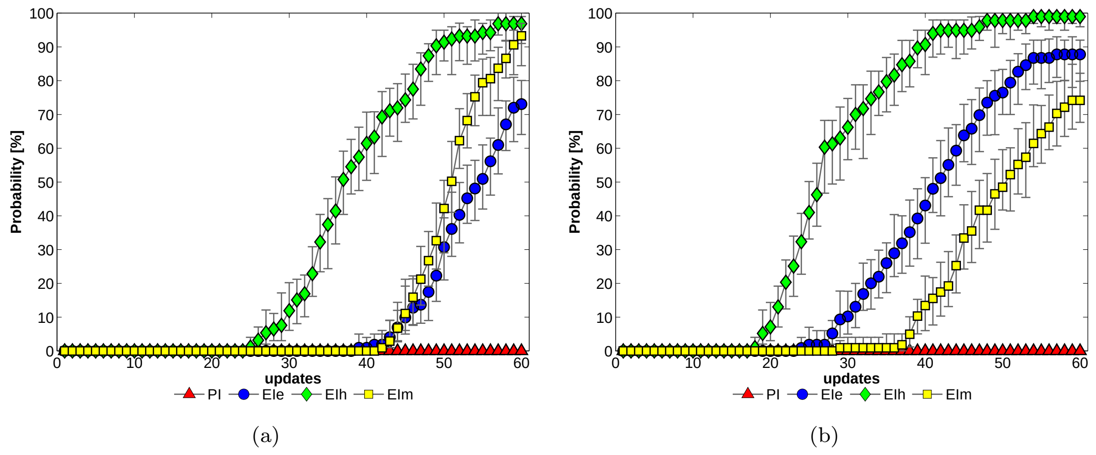


Figure 6.18: Satellite boom problem mean probability and 95% confidence intervals of the (a) hypervolume error being less than 0.008 and (b) the ϵ-indicator less than 0.04.

Table 6.4: Summary of mean results for the satellite boom design problem with the hypervolume error, H_{err}, and ϵ-indicator, ϵ_{err}.

	$H_{0.008}$		$\epsilon_{0.04}$	
	No. of updates		No. of updates	
	30	60	30	60
PI	0%	0%	0%	0%
EIe	0%	73%	10%	87%
EIh	**11%**	**97%**	**66%**	**99%**
EIm	0%	93%	1%	73%

6.9 Conclusions

In aid of reducing the total number of expensive evaluations in surrogate-based multiobjective optimization, this chapter reviews the performance of four existing multiobjective improvement criteria on a selection of test problems. In general, PI performs poorly and significant enhancements in both efficiency and reliability can be achieved by an improvement criterion based on expected improvement. Perhaps the best variant of expected improvement is unclear but EIh appears promising having a dominant performance on two of the four problems investigated. In cases where computing the expected improvement based on the hypervolume becomes expensive, EIm offers a suitable alternative having a reasonable performance on all four test problems. Although EIe exhibits some poor theoretical properties, this was the best improvement criterion tested on the Nowacki beam design problem but its inconsistent performance on the other problems may cause concern.

Although a constrained problem has been investigated in this chapter, little emphasis has been placed on constraint handling in multiobjective optimization. In the Nowacki beam design problem, all Pareto optimal solutions lie on the constraint boundaries. As demonstrated in previous chapters, the modification of the improvement criteria for constraint handling can have a big influence on the efficiency and reliability of the improvement criteria. These improvements are expected to translate to handling constraints in multiobjective optimization and an investigation into different constraint handling strategies is an area recommended for future work.

This study has been limited to design problems with only two variables and two objectives but all these methods can be extended to deal with larger problems. Formulating an analytical expression for the improvement criteria above two objectives is likely to be challenging and other methods may need to be adopted. Svenson [2011] use Monte-Carlo methods to extend the four improvement criteria discussed to deal with test problems with up to six design variables and six objectives.

Chapter 7

Parallel Evaluations in Surrogate-Based Multiobjective Optimization

Chapter 6 introduces some useful improvement criteria for surrogate-based multiobjective optimization. These improvement criteria have been formulated with single sequential updates in mind, limiting the rate of surrogate improvement. Clear benefits are associated with evaluating several designs in parallel, taking advantage of any additional computing resources. In this chapter we extend the existing improvement criteria discussed in Chapter 6 to select multiple updates using the Kriging believer strategy and introduce an alternative goal-based approach.

7.1 Multiple Update Strategies

The four multiobjective improvement criteria PI, EIe, EIh and EIm, discussed in Chapter 6 can be extended to select multiple updates in a number of ways. The fact that most improvement criteria tend to be multimodal means it is relatively easy to find multiple updates by utilizing several local optima. This approach is taken by Keane and Scanlan [2007], finding multiple updates based on the multiobjective probability of improvement. The authors exploited available computing resources and software licenses by evaluating ten designs in parallel at each update stage. Svenson [2011] proposes a batch hypervolume improvement function, essentially maximizing the expected improvement in the hypervolume based on multiple points. This however leads to a complex sub optimization problem with $q \times d$ dimensions.

In this study, multiple updates are selected using the Kriging believer strategy discussed in Chapter 5. This method substitutes a dummy value into the set of observed responses once the first update has been found. By doing so, the correlation matrix is updated and the predicted variance at the location of the dummy value goes to zero. Without evaluating the first update and without retuning, the improvement criteria can be searched again for a second update point. Since the predicted variance at the previous update is zero, maximum improvement will lie elsewhere in the design space. This is repeated until the required number of multiple updates is found.

Using the Kriging believer is easy to implement but relies on sequential searches on the surrogate to find multiple updates. This prevents the search for multiple updates themselves to be performed in parallel. As noted in Section 5.4.1, whilst the search for update points is cheap this is unlikely to raise concern but when dealing with problems with more than a few design variables or heavily populated Pareto sets, searching for multiple updates sequentially may become an issue. The four improvement criteria modified to select multiple updates using the Kriging believer are denoted as PIkb, EIekb, EIhkb and EImkb respectively. Next, an alternative goal-based approach is introduced that can be used to find multiple updates without the need for sequential searches.

7.2 Goal-Based Improvement

A key advantage of selecting multiple updates is that they allow different regions of the Pareto set to be developed simultaneously, accelerating convergence towards a global Pareto set. To achieve this, the goal-based approach considers improvement in a particular region of the Pareto front. The user first specifies a goal point $\boldsymbol{y}^g$, and the improvement criterion aims to improve upon this goal. This is a much simpler approach than considering the improvement associated with the entire Pareto set and lends itself to selecting multiple updates since the improvement over different goals can be evaluated independently and in parallel. Furthermore, selecting goal points provides an opportunity for design preferences to be incorporated into the search. This may occur when the designer is interested in a particular region of the Pareto front or if one region looks more promising than another. In this study it is assumed that no design preferences are available and the choice of goal points is automated.

When selecting goal points automatically, it is possible to select them based on a Pareto set quality indicator, the spacing of the Pareto set or based on under sampled regions of the design space. Whichever goal points are selected, it is intuitive to encourage updates to augment or dominate the current Pareto set.

Figure 7.1 illustrates a set of postulated goal points based on the spacing of existing solutions. Selecting one or more of these goals will encourage convergence to a well spaced set of Pareto solutions. To find the most suitable of these goals, they are ranked according to the area that augments the Pareto set,

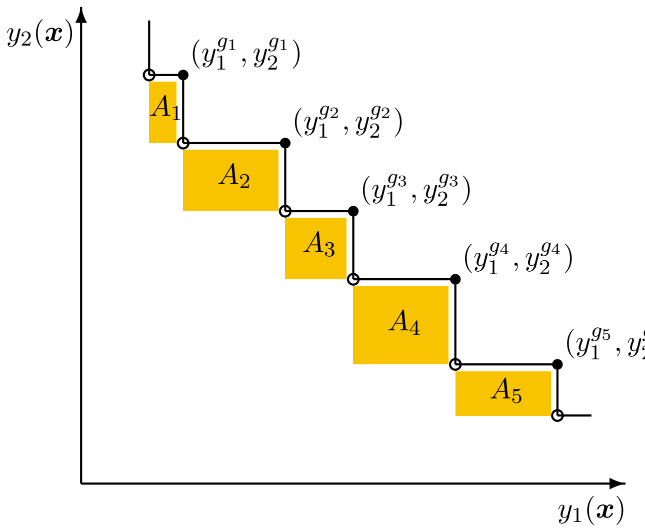


FIGURE 7.1: Computing goal points. ○– current Pareto set, ●– postulated goals 🟨 – augmented area.

$$A_i = \left(y_1^{i+1} - y_1^i\right)\left(y_2^i - y_2^{i+1}\right), \tag{7.1}$$

where $i = 1 \ldots (k-1)$.

This is simply a product of the distance between neighbouring solutions. This is quick to compute and ranks the goals regardless of scaling. In this example the goal points are ranked $(y_1^{g_4}, y_2^{g_4})$,$(y_1^{g_2}, y_2^{g_2})$, $(y_1^{g_5}, y_2^{g_5})$,$(y_1^{g_3}, y_2^{g_3})$ and $(y_1^{g_1}, y_2^{g_1})$ respectively. From these ranked solutions, it is then possible to select one or more of these goals depending on the required number of updates. If the required number of updates exceeds the number of postulated goals, it may be necessary to select additional goals. In this study, the Pareto solutions themselves are used as goals if additional goal points are required.

7.2.1 Goal-Based Probability of Improvement

The goal-based probability of improvement considers the integral under the multivariate PDF, with limits based on the region pictured in Figure 7.2. These limits bound the region of improvement made over the single goal point (y_1^g, y_2^g).

For two objectives, the goal-based probability of improvement is given by

$$P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = \int_{-\infty}^{y_1^g}\int_{-\infty}^{y_2^g} \phi\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1, \tag{7.2}$$

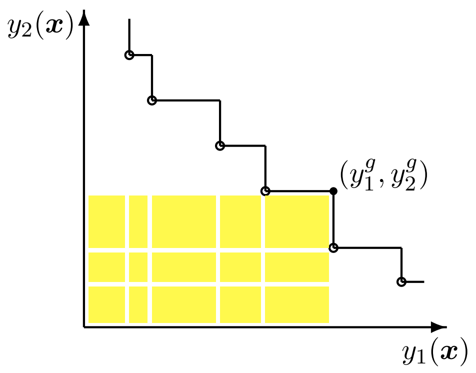


FIGURE 7.2: Region of goal-based improvement. 🟨 – Region that offers improvement over a single goal point where ○– current Pareto set and ●– goal point (y_1^g, y_2^g).

or given analytically as,

$$P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = \mathbf{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\mathbf{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right). \tag{7.3}$$

Defining the region of integration in this way achieves a simple formulation of the multi-objective probability of improvement that exploits a specific region of the Pareto front. Note that exploiting the Pareto set by using this goal-based improvement may hinder the addition of points at either end of the Pareto set. This can be easily dealt with by including an additional region that improves over the end points of the Pareto set. This is included when a goal point is chosen that lies next to an end point, shown in Figure 7.3. The goal-based probability of improvement for Figure 7.3(a) is given as

$$\begin{aligned} P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = & \int_{-\infty}^{y_1^g}\int_{-\infty}^{y_2^g} \boldsymbol{\phi}(y_1, y_2)\,\mathrm{d}y_2\mathrm{d}y_1 \\ & + \int_{-\infty}^{y_1^1}\int_{y_2^g}^{\infty} \boldsymbol{\phi}(y_1, y_2)\,\mathrm{d}y_2\mathrm{d}y_1, \end{aligned} \tag{7.4}$$

and for Figure 7.3(b)

$$\begin{aligned} P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = & \int_{-\infty}^{y_1^g}\int_{-\infty}^{y_2^g} \boldsymbol{\phi}(y_1, y_2)\,\mathrm{d}y_2\mathrm{d}y_1 \\ & + \int_{y_1^g}^{\infty}\int_{-\infty}^{y_2^e} \boldsymbol{\phi}(y_1, y_2)\,\mathrm{d}y_2\mathrm{d}y_1. \end{aligned} \tag{7.5}$$

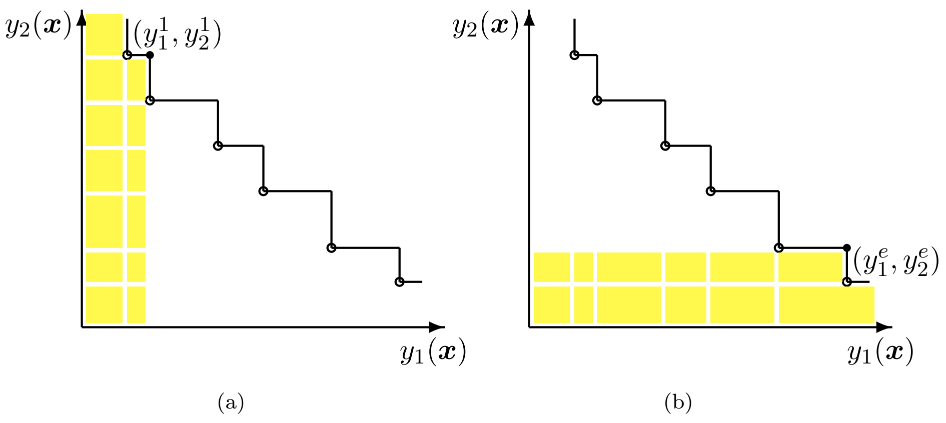


FIGURE 7.3: End point regions. 🟨 – Region that offers improvement over the goal point (y_1^g, y_2^g) which lies next to the end point (a) (y_1^1, y_2^1) and (b) (y_1^e, y_1^e).

Analytic expressions for these formulations are given in Appendix B. Adding updates based on this goal-based probability of improvement is well suited to selecting multiple updates as it is can be searched independently several times with different goal points. This allows different regions of the Pareto front to be exploited simultaneously. The goal-based probability of improvement is denoted as PIg.

7.2.2 Goal-Based Expected Improvement

This goal-based approach can be extending to the formulation of expected improvement considered next. It is possible to formulate the goal-based expected improvement in a number of ways. Here we limit ourselves to formulations that are considered analogous to the improvement criteria introduced in Chapter 6.

7.2.2.1 Euclidean-Based

In a similar manner to the goal-based probability of improvement, it is possible to extend the goal-based approach to the Euclidean-based expected improvement. By considering the improvement over a given point, the goal-based expected improvement can be expressed as

$$E[I_E(\boldsymbol{x})] = P[y_1 \leq y_1^g \cup y_2 \leq y_2^g]\sqrt{(\bar{y}_1 - y_1^g)^2 + (\bar{y}_2 - y_2^g)^2}, \tag{7.6}$$

where

$$\bar{y}_1 P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = \int_{-\infty}^{y_1^g}\int_{-\infty}^{y_2^g} y_1 \boldsymbol{\phi}\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1, \tag{7.7}$$

and

$$\bar{y}_2 P[y_1 \le y_1^g \cup y_2 \le y_2^g] = \int_{-\infty}^{y_1^g} \int_{-\infty}^{y_2^g} y_2 \phi\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1, \tag{7.8}$$

Here the goal-based improvement function is not given explicitly but can be considered equivalent to,

$$I_E = \sqrt{\left(y_1 - y_1^g\right)^2 + \left(y_2 - y_2^g\right)^2}. \tag{7.9}$$

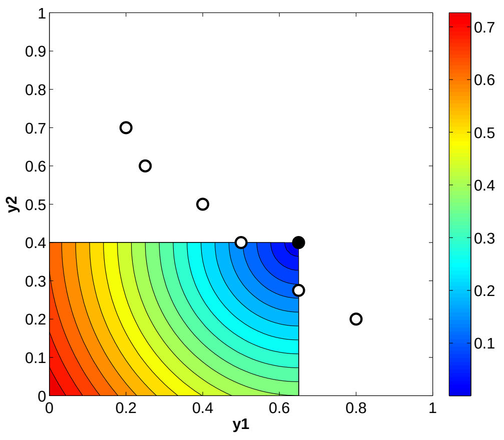


FIGURE 7.4: Euclidean goal-based improvement.

Figure 7.4 illustrates the Euclidean goal-based improvement for an example Pareto set. This rewards the most dominating regions but does not consider any improvement at either end of the Pareto set. End regions are included when the goal point lies close to the end point (y_1^1, y_2^1) where $P[y_1 \le y_1^g \cup y_2 \le y_2^g]$ is given by equation (7.4) and

$$\begin{aligned}
\bar{y}_1 P[y_1 \le y_1^g \cup y_2 \le y_2^g] &= \int_{-\infty}^{y_1^g} \int_{-\infty}^{y_2^g} y_1 \phi\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1 + \int_{-\infty}^{y_1^1} \int_{y_2^g}^{\infty} y_1 \phi\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1, \\
\bar{y}_2 P[y_1 \le y_1^g \cup y_2 \le y_2^g] &= \int_{-\infty}^{y_1^g} \int_{-\infty}^{y_2^g} y_2 \phi\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1 + \int_{-\infty}^{y_1^1} \int_{y_2^g}^{\infty} y_2 \phi\left(y_1, y_2\right) \mathrm{d}y_2 \mathrm{d}y_1.
\end{aligned} \tag{7.10}$$

When the goal point lies close to the end point (y_1^e, y_2^e), $P[y_1 \le y_1^g \cup y_2 \le y_2^g]$ is given by equation (7.5) and

$$\bar{y}_1 P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = \int_{-\infty}^{y_1^g} \int_{-\infty}^{y_2^g} y_1 \phi(y_1, y_2)\, \mathrm{d}y_2 \mathrm{d}y_1 + \int_{y_1^g}^{\infty} \int_{-\infty}^{y_2^e} y_1 \phi(y_1, y_2)\, \mathrm{d}y_2 \mathrm{d}y_1,$$
$$\bar{y}_2 P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = \int_{-\infty}^{y_1^g} \int_{-\infty}^{y_2^g} y_2 \phi(y_1, y_2)\, \mathrm{d}y_2 \mathrm{d}y_1 + \int_{y_1^g}^{\infty} \int_{-\infty}^{y_2^e} y_2 \phi(y_1, y_2)\, \mathrm{d}y_2 \mathrm{d}y_1. \tag{7.11}$$

Analytic expressions for these formulations are given in Appendix B. Figure 7.5 shows the Euclidean goal-based improvement for these end regions, encouraging the selection of new samples at either end of the Pareto set. Since this formulation still requires calculation of the Euclidean distance, relative scaling of the objectives remains important. The Euclidean goal-based expected improvement is termed as EIeg for the remainder of this thesis.

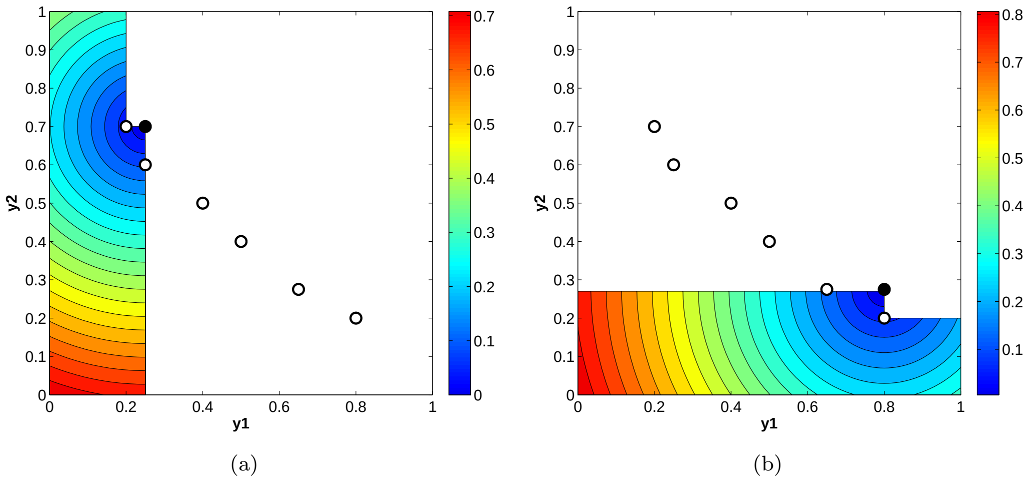


Figure 7.5: Euclidean goal-based improvement with end regions.

7.2.2.2 Hypervolume-Based

When considering the improvement over a goal point, the hypervolume-based improvement is heavily simplified. In this study this improvement is given quite simply by

$$I_H = (y_1^g - y_1)(y_2^g - y_2). \tag{7.12}$$

This can be viewed as the product of individual improvements in each objective. Figure 7.6 illustrates this improvement given a Pareto set and goal point. This improvement increases as (y_1, y_2) moves down and to the left of the goal point, thus favouring points that are the most dominating. This avoids any computational burden associated with computing the improvement in the hypervolume based on the entire Pareto set. Furthermore, this goal-based improvement is advantageous as it can be used regardless of objective scaling.

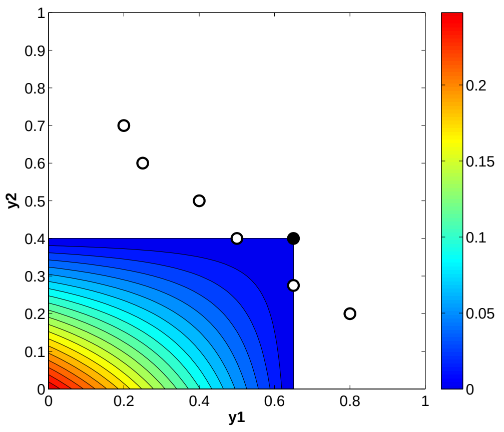


FIGURE 7.6: Goal-based improvement in hypervolume.

The goal-based expected improvement for two objectives is given as,

$$E[I_H(\boldsymbol{x})]_G = \int_{-\infty}^{y_1^g}\int_{-\infty}^{y_2^g} (y_1^g - y_1)(y_2^g - y_2)\boldsymbol{\phi}\,(y_1, y_2)\,\mathrm{d}y_2\mathrm{d}y_1, \tag{7.13}$$

or given analytically as

$$\begin{aligned} E[I_H(\boldsymbol{x})]_G = {} & y_1^g \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) y_2^g \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) \\ & - y_1^g \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right] \\ & - y_2^g \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right] \\ & + \left\{\left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right]\right. \\ & \times \left.\left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right]\right\}. \end{aligned} \tag{7.14}$$

Again it is important to consider the regions that offer improvement at either end of the Pareto set. If a goal is selected next to (y_1^1, y_2^1), the goal-based expected improvement is given as

$$E[I_H(\boldsymbol{x})]_G = \int_{-\infty}^{y_1^g}\int_{-\infty}^{y_2^g} (y_1^g - y_1)(y_2^{max} - y_2)\boldsymbol{\phi}\,(y_1, y_2)\,\mathrm{d}y_2\mathrm{d}y_1 \\ + \int_{-\infty}^{y_1^1}\int_{y_2^g}^{y_2^{max}} (y_1^g - y_1)(y_2^{max} - y_2)\boldsymbol{\phi}\,(y_1, y_2)\,\mathrm{d}y_2\mathrm{d}y_1, \tag{7.15}$$

where y_2^{max} is the maximum value samples so far. If a goal is selected next to (y_1^e, y_2^e),

$$E[I_H(\boldsymbol{x})]_G = \int_{-\infty}^{y_1^g}\int_{-\infty}^{y_2^g} (y_1^{max} - y_1)(y_2^g - y_2)\boldsymbol{\phi}\,(y_1, y_2)\,\mathrm{d}y_2\mathrm{d}y_1 \\ + \int_{y_1^g}^{y_1^{max}}\int_{-\infty}^{y_2^e} (y_1^{max} - y_1)(y_2^g - y_2)\boldsymbol{\phi}\,(y_1, y_2)\,\mathrm{d}y_2\mathrm{d}y_1, \tag{7.16}$$

where y_1^{max} is the maximum value sampled so far. Analytical expressions for equation (7.15) and equation (7.16) is provided in Appendix B.

Figure 7.7 illustrates the goal-based improvement when selecting goals close to these end points. Note that this formulation requires suitable values of y_1^{max} and y_2^{max} which depend on previously sampled points. The goal-based expected improvement in hypervolume is termed EIhg.

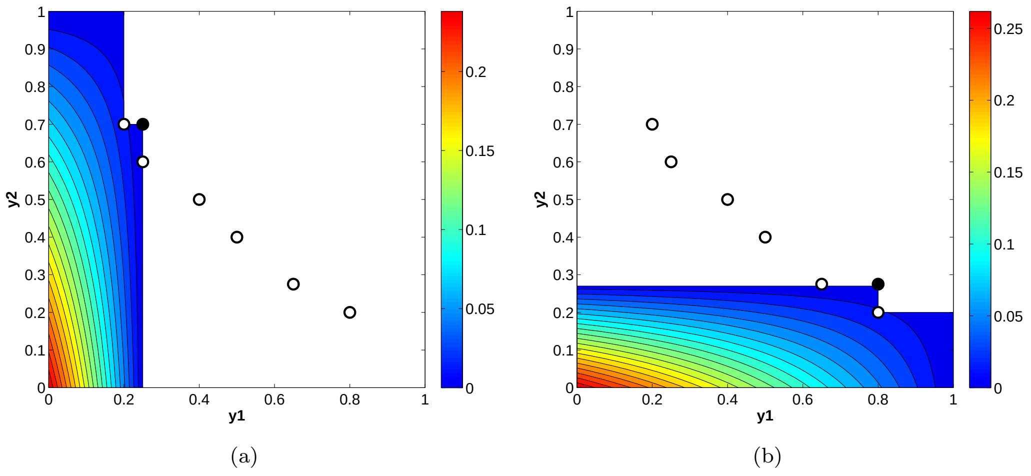


FIGURE 7.7: Goal-based improvement in hypervolume with end regions.

7.2.2.3 Maximin

The final goal-based improvement criterion considered the maximin improvement function. Figure 7.8 shows two regions that offer improvement over a goal point (y_1^g, y_2^g).

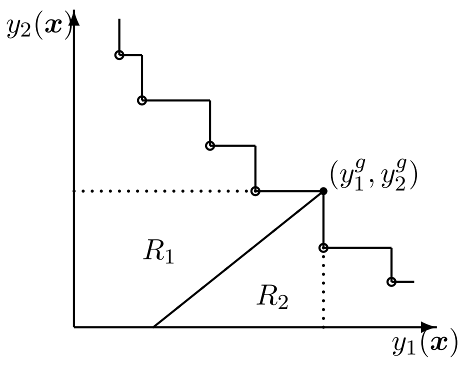


FIGURE 7.8: Regions of goal-based maximin improvement with a single goal point (y_1^g, y_2^g).

Based on the maximin improvement, equation (6.13), the improvement in R_1 is simply $y_1^g - y_1$ and in R_2 the improvement is given as $y_2^g - y_2$. This leads to the improvement shown in Figure 7.9.

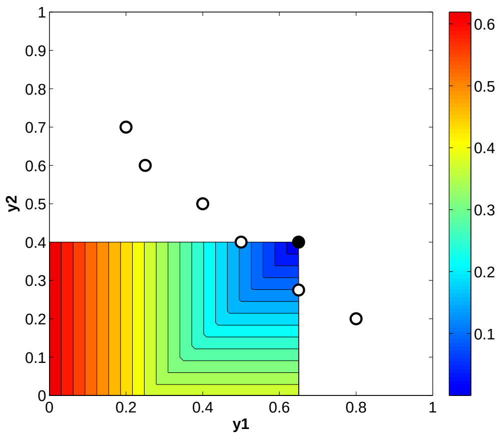


FIGURE 7.9: Goal-based maximin improvement.

Based on this improvement function, the goal-based expected improvement is given as

$$
\begin{aligned}
E[I_M(\boldsymbol{x})]_G = &\int_{-\infty}^{y_1^g} \int_{y_2^g - y_1^g + y_1}^{y_2^g} (y_1^g - y_1)\boldsymbol{\phi}\,(y_1, y_2)\,\mathrm{d}y_2 \mathrm{d}y_1 \\
&+ \int_{-\infty}^{y_2^g} \int_{y_1^g - y_2^g + y_2}^{y_1^g} (y_2^g - y_2)\boldsymbol{\phi}\,(y_1, y_2)\,\mathrm{d}y_1 \mathrm{d}y_2.
\end{aligned}
\tag{7.17}
$$

Following a similar approach to Svenson [2011], the goal-based maximin expected improvement can be expressed analytically as,

$$\begin{aligned}
E[I_M(\boldsymbol{x})]_G = {} & \hat{s}_1 \boldsymbol{\Phi}\left(\frac{\hat{y}_1 - y_1^g}{\hat{s}_1}\right) \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) \\
& + \hat{s}_2 \boldsymbol{\Phi}\left(\frac{\hat{y}_2 - y_2^g}{\hat{s}_2}\right) \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) \\
& + \boldsymbol{\Phi}\left(\frac{\hat{y}_2 - y_2^g}{\hat{s}_2}\right) \left[(y_1^g - \hat{y}_1) \boldsymbol{\Phi}\left(\frac{\hat{y}_1 - y_1^g}{\hat{s}_1}\right) + \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) \right] \\
& + \boldsymbol{\Phi}\left(\frac{\hat{y}_1 - y_1^g}{\hat{s}_1}\right) \left[(y_2^g - \hat{y}_2) \boldsymbol{\Phi}\left(\frac{\hat{y}_2 - y_2^g}{\hat{s}_2}\right) + \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) \right] \\
& + \sqrt{A} \frac{\hat{s}_1}{\hat{s}_2 \sqrt{2\pi}} \exp\left\{ -\frac{1}{2} \left[\frac{\hat{y}_1^2}{\hat{s}_1^2} + \frac{(y_1^g - y_2^g + \hat{y}_2)^2}{\hat{s}_2} \right] \right\} \exp\left(\frac{1}{2} A v_1^2\right) \boldsymbol{\Phi}\left(\frac{y_1^g - A v_1}{\sqrt{A}}\right) \\
& + \sqrt{A} \frac{\hat{s}_2}{\hat{s}_1 \sqrt{2\pi}} \exp\left\{ -\frac{1}{2} \left[\frac{\hat{y}_2^2}{\hat{s}_2^2} + \frac{(y_2^g - y_1^g + \hat{y}_1)^2}{\hat{s}_1} \right] \right\} \exp\left(\frac{1}{2} A v_2^2\right) \boldsymbol{\Phi}\left(\frac{y_2^g - A v_2}{\sqrt{A}}\right) \\
& + (y_1^g - \hat{y}_1) \int_0^{\boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)} \boldsymbol{\Phi}\left(\frac{y_1^g - y_2^g - \hat{y}_1 + \hat{y}_2 - \hat{s}_1 \boldsymbol{\Phi}(w)^{-1}}{\hat{s}_2}\right) \mathrm{d}w \\
& + (y_2^g - \hat{y}_2) \int_0^{\boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)} \boldsymbol{\Phi}\left(\frac{y_2^g - y_1^g - \hat{y}_2 + \hat{y}_1 - \hat{s}_2 \boldsymbol{\Phi}(w)^{-1}}{\hat{s}_1}\right) \mathrm{d}w.
\end{aligned} \tag{7.18}$$

where

$$A = \frac{\hat{s}_1^2 \hat{s}_2^2}{\hat{s}_1^2 + \hat{s}_2^2} \tag{7.19}$$

$$v_1 = \frac{\hat{y}_1}{\hat{s}_1^2} + \frac{y_1^g - y_2^g + \hat{y}_2}{\hat{s}_2} \tag{7.20}$$

and

$$v_2 = \frac{\hat{y}_2}{\hat{s}_2^2} + \frac{y_2^g - y_1^g + \hat{y}_1}{\hat{s}_1}. \tag{7.21}$$

Note that due to the non-rectangular regions of integration, the analytical expression is more complex and the last two terms must be computed using numerical quadrature. Since this is still based on the distance between a goal point and a new sample location, relative scaling of the objectives remains important.

As with the other goal-based methods it is important to consider regions close to the end points. If a goal is selected next to (y_1^1, y_2^1), the goal-based expected improvement is given as

$$\begin{aligned}E[I_M(\boldsymbol{x})]_G = &\int_{-\infty}^{y_1^g}\int_{y_2^g-y_1^g+y_1}^{y_2^g}(y_1^g-y_1)\phi\,(y_1,y_2)\,\mathrm{d}y_2\mathrm{d}y_1\\ &+\int_{-\infty}^{y_2^g}\int_{y_1^g-y_2^g+y_2}^{y_1^g}(y_2^g-y_2)\phi\,(y_1,y_2)\,\mathrm{d}y_1\mathrm{d}y_2\\ &+\int_{-\infty}^{y_1^1}\int_{y_2^g}^{\infty}(y_1^g-y_1)\phi\,(y_1,y_2)\,\mathrm{d}y_2\mathrm{d}y_1.\end{aligned}\tag{7.22}$$

If a goal is selected next to (y_1^e, y_2^e),

$$\begin{aligned}E[I_M(\boldsymbol{x})]_G = &\int_{-\infty}^{y_1^g}\int_{y_2^g-y_1^g+y_1}^{y_2^g}(y_1^g-y_1)\phi\,(y_1,y_2)\,\mathrm{d}y_2\mathrm{d}y_1\\ &+\int_{-\infty}^{y_2^g}\int_{y_1^g-y_2^g+y_2}^{y_1^g}(y_2^g-y_2)\phi\,(y_1,y_2)\,\mathrm{d}y_1\mathrm{d}y_2\\ &+\int_{-\infty}^{y_2^e}\int_{y_1^g}^{\infty}(y_2^g-y_2)\phi\,(y_1,y_2)\,\mathrm{d}y_1\mathrm{d}y_2.\end{aligned}\tag{7.23}$$

Figure 7.10 illustrates the maximin improvement when selecting end points and analytical expressions are provided in Appendix B. Here the goal-based maximin expected improvement is termed EImg.

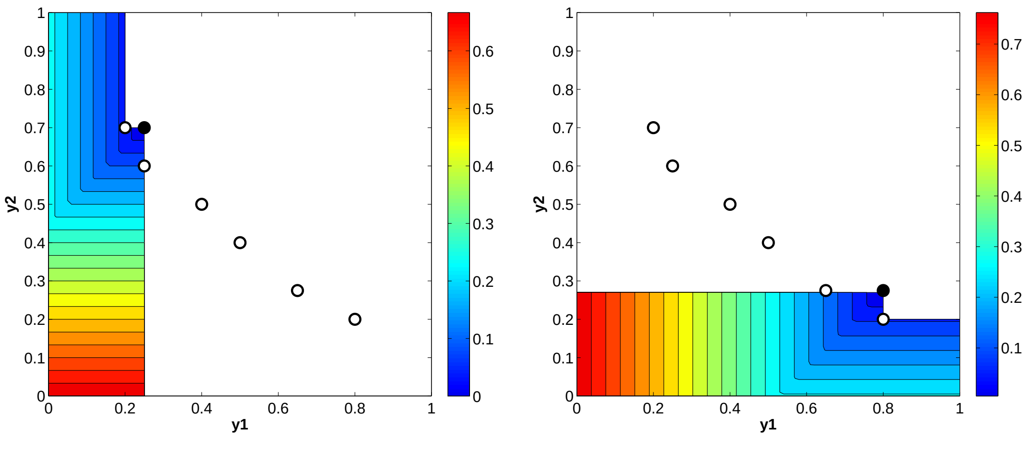


FIGURE 7.10: Goal-based maximin improvement with end regions.

7.3 Results and Discussion

Test problem 4 and the satellite boom design problem use an initial sample of 20 points and selects four updates ($q = 4$) at each of 30 update stages. Test problem 5 and the

Nowacki beam design problem use an initial sample of 20 points, $q = 4$ and 20 update stages. Assuming resources exists that allow new update points to be evaluated in parallel, the time spent evaluating new designs (again referred to as the evaluation wall clock time) equates to half of the total evaluation time required using single updates in Chapter 6. Although the evaluation walk clock time is halved, the total CPU cost (or total number of evaluations) is doubled. A dashed line is included that represents the equivalent CPU cost as the methods tested in Chapter 6.

The performance of each multiple update improvement criterion is assessed on the same hypervolume error and ϵ-indicator presented in the single update case. The reader should note that by using multiple updates, many more designs are evaluated. This causes the design space to become more populated, often resulting in more populated and better quality Pareto sets. The following results do not illustrate any increase in the Pareto set quality but instead provide a direct comparison to the quality achieved when using single updates. Some example Pareto sets are included in Appendix A, providing a visual comparison of each method.

7.3.1 Artificial Test Problems

In Figure 7.11 we show the efficiency and reliability of the multiple update improvement criteria when applied to test problem 4. When finding a hypervolume error less than 0.07, all methods based on the Kriging believer perform well. Although there is little statistical significance between methods, EImkb offers the best performance in Figure 7.11(a) and Figure 7.11(b). EImkb finds good quality Pareto sets equivalent to those found using single updates in less than a third of the total wall clock time, demonstrating significant improvements in efficiency. The example Pareto sets in Figure A.2 show similarities with those found using single updates. In this particular example, EIhkb achieves the best Pareto set visually and offers significant improvements over EIh when using single updates in Figure A.1.

In Figure 7.11(c) and Figure 7.11(d) we see that two of the goal-based methods match the performance of EImkb. The poor performance of the scale dependent goal-based methods, EIeg and EImg, suggest that the goal-based approach does not generalise well to all improvement criteria. Based on both the hypervolume error and ϵ-indictor, EIhg is the most efficient and reliable method tested, finding good quality Pareto sets with an equivalent CPU cost and 25% of the total wall clock time when compared with single updates.

The scale dependent goal-based improvement criteria continue to perform poorly in test problem 5, Figure 7.12. In this problem EIkb and EIekb also fail to find good quality Pareto sets reliably, reflecting the poor performance of these methods in Chapter 6. As in the single update case, EIhkb and EImkb are the most reliable methods when

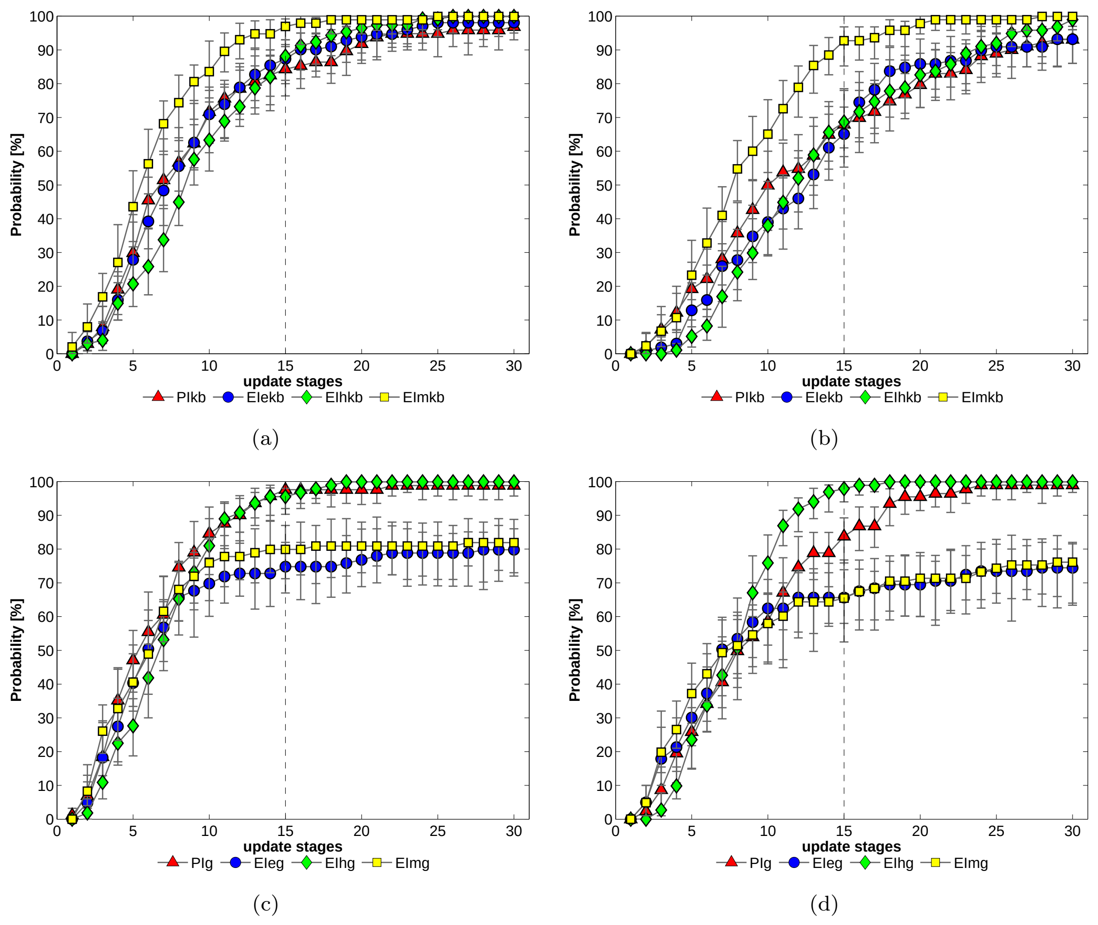


FIGURE 7.11: Test problem 4 mean probability and 95% confidence intervals of the hypervolume error being less than 0.07 and ϵ-indicator less than 0.08. (a) Hypervolume error when using Kriging believer, (b) ϵ-indicator when using Kriging believer, (c) hypervolume error when using goal-based improvement and (d) ϵ-indicator when using goal-based improvement.

selecting multiple updates using the Kriging believer. In this problem, EIhg proves to be the most promising method showing a significant improvement in efficiency over the other methods tested.

In both test problems tested here, an interesting observation is that the probability of improvement performs better when formulated based on goal points. Defining goal points introduces some forced exploration into the search and helps to reduce the clustering of update points. This is observed particularly well in Figure A.6. Although both PIkb and PIg are often inferior to the other methods tested, the results go some way to demonstrate that better updates can be selected by reducing the region of integration and explicitly improving on particular regions of the Pareto front.

Table 7.1 summarizes the performance of each improvement criterion on test problem 4 and test problem 5. EIhg performs very well in comparison to the other improvement criteria tested, especially at fewer update stages.

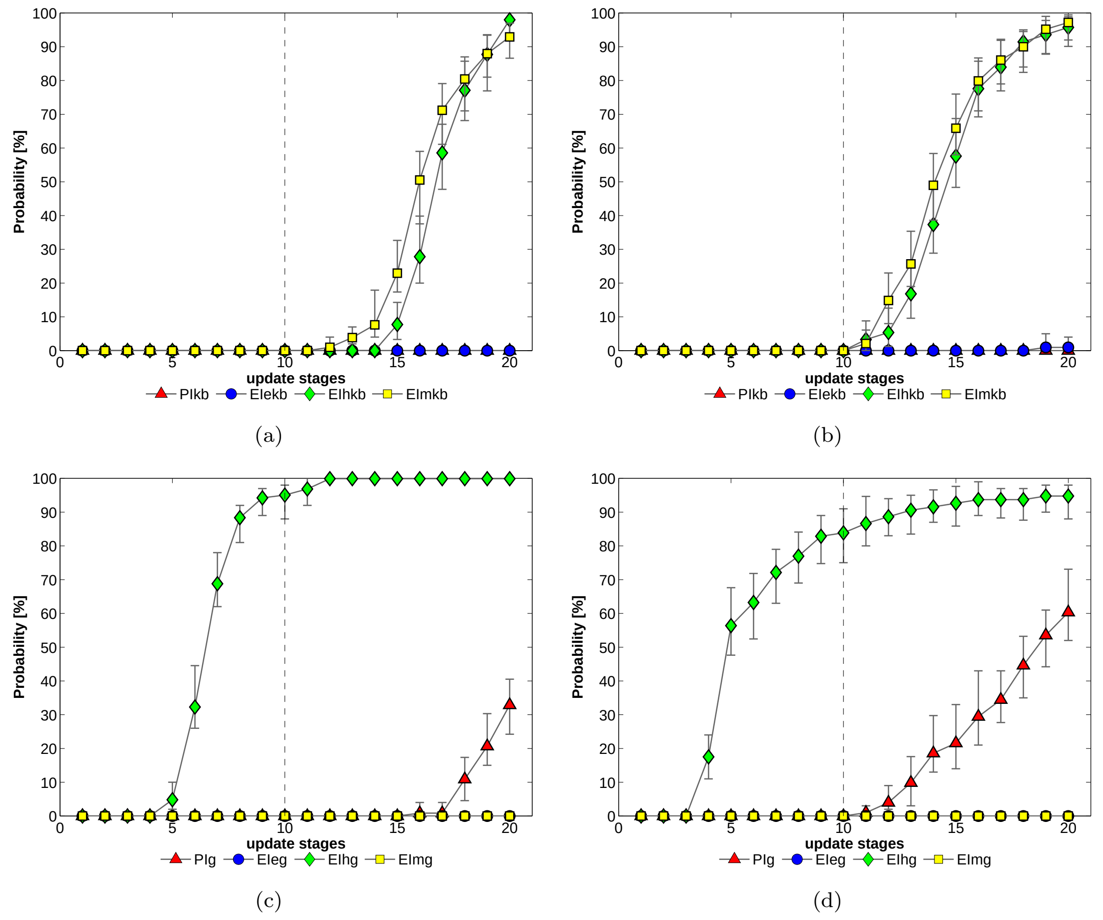


FIGURE 7.12: Test problem 5 mean probability and 95% confidence intervals of the hypervolume error less being than 0.02 and ϵ-indicator less than 0.04. (a) Hypervolume error when using Kriging believer, (b) ϵ-indicator when using Kriging believer, (c) hypervolume error when using goal-based improvement and (d) ϵ-indicator when using goal-based improvement.

7.3.2 Nowacki Beam Design Problem

In the Nowacki beam design problem, the most reliable and efficient method is again EIhg, highlighted in Table 7.2. This improvement criteria is able to find Pareto sets with an equivalent hypervolume error as Figure 6.17 in 25% of the total evaluation wall clock time and at the same CPU cost. This increase in efficiency is less dramatic when comparing the ϵ-indictor but still offers significant improvements over the other methods tested. Out of the remaining methods, EIekb, EIhkb, EIeg and EImg all offer a good performance. The improvement criteria using the Kriging believer are much less efficient when compared to EIhg and require further update stages to consistently find good quality Pareto sets. This is seen clearly when comparing the example Pareto sets in Figure A.8 and A.9.

TABLE 7.1: Summary of mean results for test problem 4 and 5 with the hypervolume error, H_{err}, and ϵ-indicator, ϵ_{err}.

	Test problem 4				Test problem 5			
	$H_{0.07}$		$\epsilon_{0.08}$		$H_{0.02}$		$\epsilon_{0.04}$	
	No. of update stages		No. of update stages		No. of update stages		No. of update stages	
	15	30	15	30	10	20	10	20
PIkb	85%	97%	68%	93%	0%	0%	0%	0%
EIekb	87%	98%	65%	93%	0%	0%	0%	1%
EIhkb	88%	**100%**	69%	99%	0%	98%	0%	96%
EImkb	97%	**100%**	92%	**100%**	0%	93%	0%	**97%**
PIg	**98%**	99%	84%	99%	0%	32%	0%	61%
EIeg	75%	80%	65%	74%	0%	0%	0%	0%
EIhg	96%	**100%**	**98%**	**100%**	**95%**	**100%**	**84%**	95%
EImg	80%	82%	64%	76%	0%	0%	0%	0%

TABLE 7.2: Summary of mean results for the Nowacki beam design problem with the hypervolume error, H_{err}, and ϵ-indicator, ϵ_{err}.

	$H_{0.07}$		$\epsilon_{0.11}$	
	No. of update stages		No. of update stages	
	10	20	10	20
PIkb	7%	23%	13%	42%
EIekb	22%	77%	59%	90%
EIhkb	28%	89%	39%	86%
EImkb	15%	60%	19%	58%
PIg	10%	21%	17%	27%
EIeg	79%	94%	69%	75%
EIhg	**98%**	**100%**	**85%**	**91%**
EImg	74%	89%	65%	74%

7.3.3 Satellite Boom Design Problem

Figure 7.14 compares the performance of each improvement criteria on the satellite boom design problem. Due to the nature of this problem having a distinct set of global solutions, the Pareto set becomes highly populated once the global region of the attenuation function is identified, see Figure 6.13(a). Due the expense of computing the hypervolume over the entire Pareto set, EIhkb becomes more computationally expensive to search as the Pareto set becomes more populated. In this particular problem, the expense of this

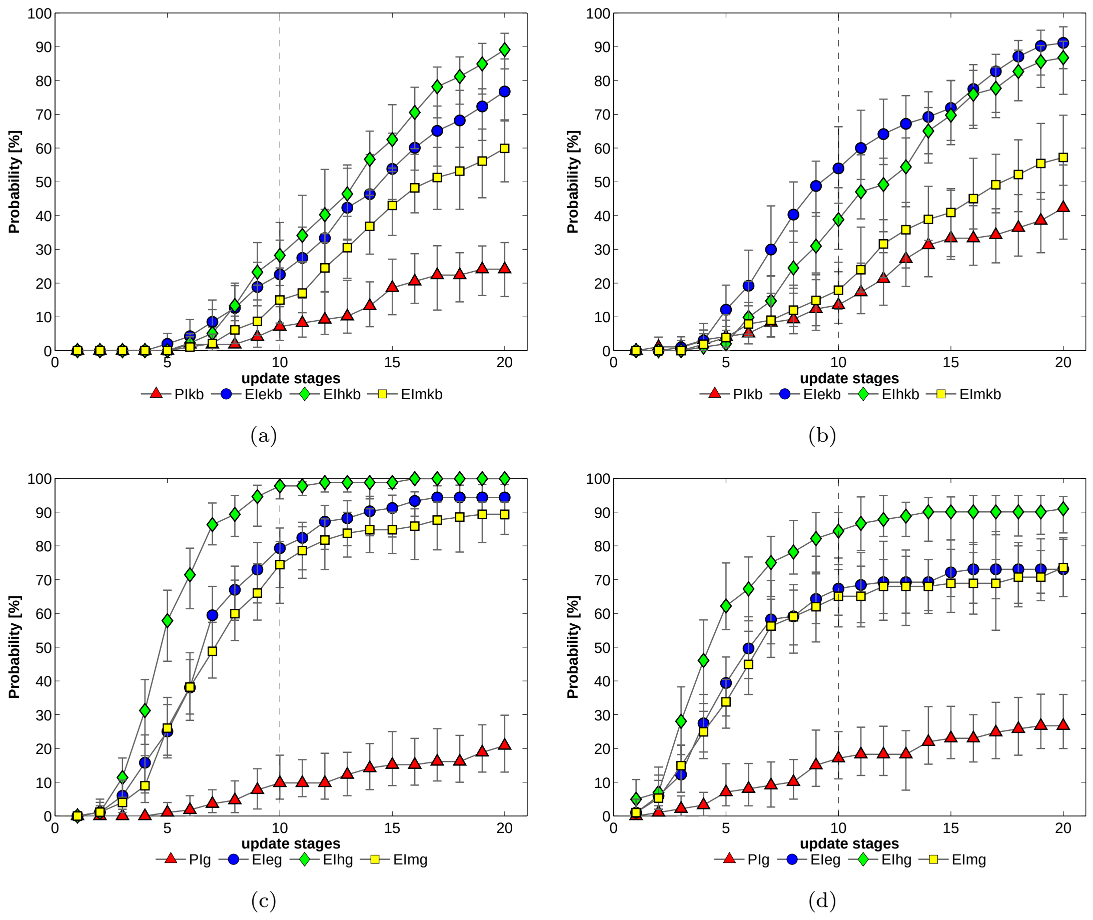


FIGURE 7.13: Nowacki beam design problem mean probability and 95% confidence intervals of the hypervolume error being less than 0.07 and ϵ-indicator less than 0.11. (a) Hypervolume error when using Kriging believer, (b) ϵ-indicator when using Kriging believer, (c) hypervolume error when using goal-based improvement and (d) ϵ-indicator when using goal-based improvement.

search is significant when compared to the other methods tested. To keep retain a similar overall run time achieved using the other improvement criteria, EIhkb is only tested up to 20 update stages. Due to the reduced number of update stages, EIhkb performs relatively poorly when finding a small hypervolume error, highlighting the drawback of this computational burden. Out of the remaining methods using the Kriging believer, EImkb offers the best performance. Out of all the methods tested, EIhg is again the most promising method, offering significant gains in performance and supported by both quality indicators, highlighted in Table 7.3. Figure A.11 and Figure A.12 show some example Pareto sets, illustrating a highly populated and well spread Pareto set when using EIhg.

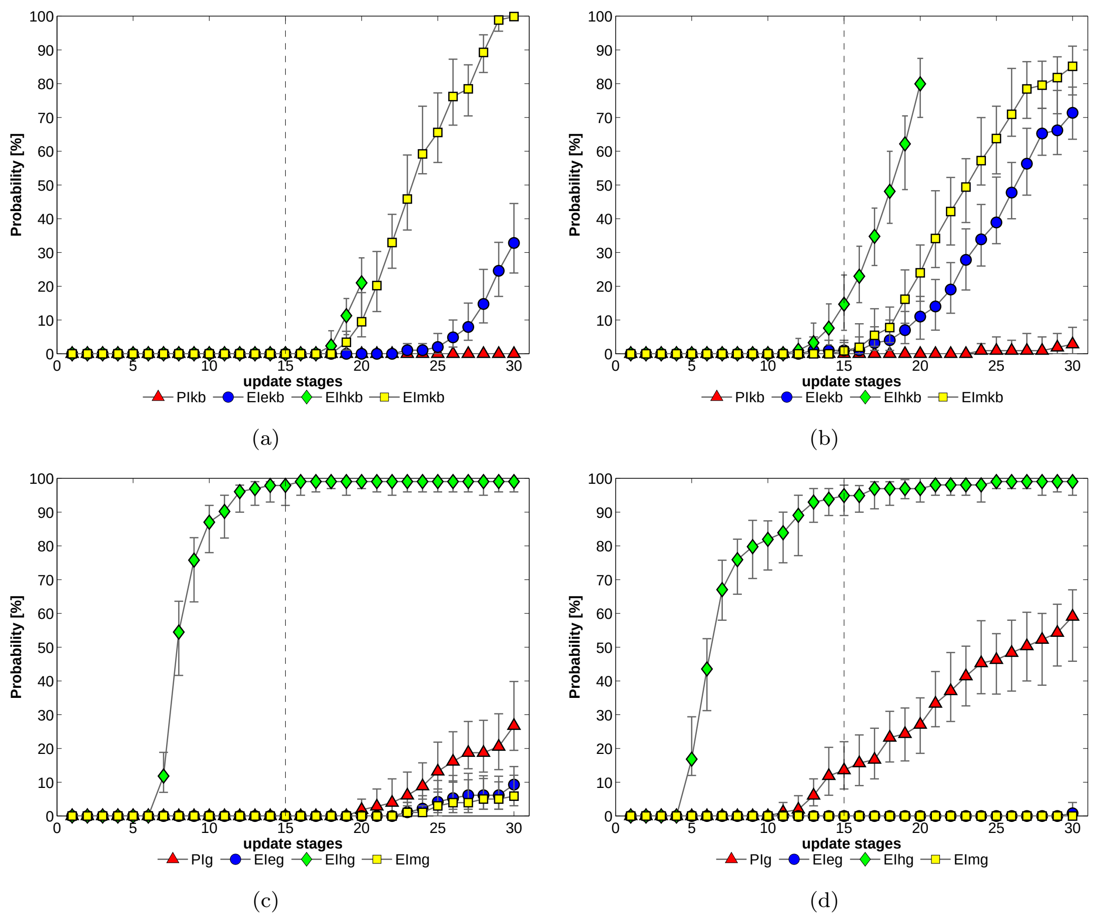


FIGURE 7.14: Satellite boom design problem mean probability and 95% confidence intervals of the hypervolume error being less than 0.008 and ϵ-indicator less than 0.04. (a) Hypervolume error when using Kriging believer, (b) ϵ-indicator when using Kriging believer, (c) hypervolume error when using goal-based improvement and (d) ϵ-indicator when using goal-based improvement.

7.4 A Note on Single Updates

Although the goal based approach has been developed to select multiple updates, it can also be used to select single updates. This is highlighted here to demonstrate the applicability of the goal-based approach even when resources to run multiple updates do not exist. To briefly demonstrate the selection of single updates, we compare the most promising method EIhg with the four improvement criteria tested in Chapter 5. When using EIhg, a single update is selected based on the postulated goal with maximum augmented area, equation (7.1). We limit the comparison to the hypervolume error given in Figure 7.15. These results show a competitive performance of EIhg when selecting single updates, offering performance gains on three out of the four test problems.

Table 7.3: Summary of mean results for the satellite boom design problem with the hypervolume error, H_{err}, and ϵ-indicator, ϵ_{err}.

	$H_{0.008}$		$\epsilon_{0.04}$	
	No. of update stages		No. of update stages	
	15	30	15	30
PIkb	0%	0%	0%	3%
EIekb	0%	33%	1%	73%
EIhkb	0%	—%	15%	—%
EImkb	0%	**100%**	1%	85%
PIg	0%	27%	14%	59%
EIeg	0%	9%	0%	1%
EIhg	**98%**	99%	**65%**	**99%**
EImg	0%	6%	0%	0%

7.5 Conclusions

This chapter extends four multiobjective improvement criteria to select multiple updates at each update stage. The improvement criteria are first modified based on the Kriging believer strategy introduced in Chapter 5. This is an effective approach to selecting multiple updates, identifying better Pareto sets whilst significantly reducing the evaluation wall clock time when compared to selecting single updates. EIhkb and EImkb appear to be the most promising methods when using the Kriging believer but in certain problems, adding multiple updates can quickly increase the computation burden of EIhkb. A goal-based strategy is also introduced that exploits particular regions of the Pareto set. Each of the four improvement criteria are formulated using some goal-based improvement and compared with those results obtained using the Kriging believer. Whilst the two scale dependent methods, EIeg and EImg, did not offer any improvements over the Kriging believer on three out of the four problems, the scale invariant methods, PIg and EIhg, offers some significant enhancements in performance. In particular, EIhg offers some advantageous properties being considerably more efficient and more reliable than the other improvement criteria tested on all four test problems. In certain cases, good quality Pareto sets are found at similar CPU cost and 25% of the evaluation wall clock time when compared to selecting single updates. This supports the use of multiple updates even if a budget on total CPU effort exists. EIhg has also been shown to generalize well when selecting single updates and compared directly with results obtained in Chapter 6 and furthermore, is the only improvement criteria based on expected improvement that is scale invariant.

Improvement criteria based on selecting goal points can be computed cheaply and do not

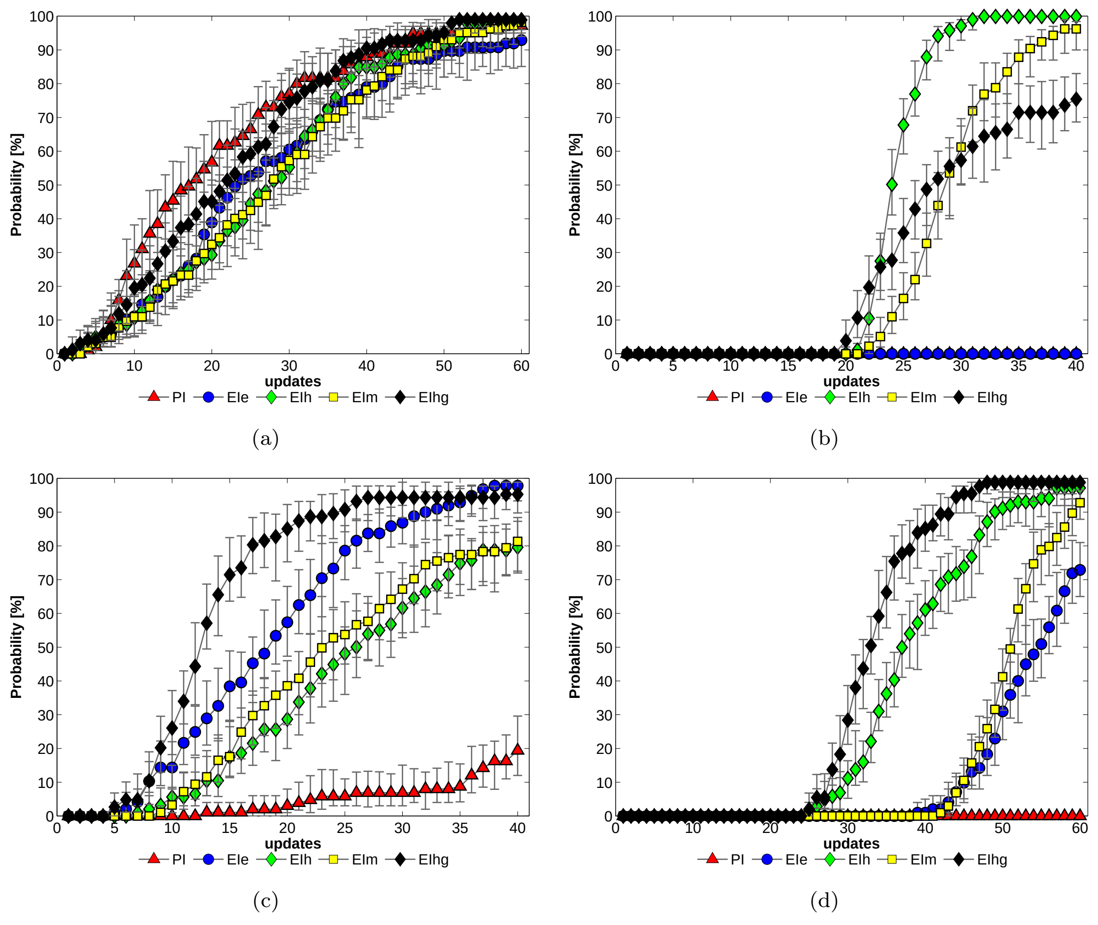


FIGURE 7.15: Mean probability and 95% confidence intervals of the hypervolume error in (a) test problem 4 (hypervolume error = 0.07), (b) test problem 5 (hypervolume error = 0.02), (c) Nowacki beam design problem (hypervolume error = 0.07) and (d) satellite boom design problem (hypervolume error = 0.008).

rely on sequential searches to indentify multiple updates. Here, goal-points are selected based on the spacing between Pareto solutions but many other approaches should be investigated. In particular, these methods can easily incorporate user preferences. This may be advantageous on problems where the designer has identified a region of the Pareto front that is more appealing. This is often the case when some trade-off is not modelled in the design problem, or when an already optimal baseline design exists. If a particular design looks promising, the designer may wish concentrate updates in this region of Pareto front. Incorporating user preferences will allow the multiobjective optimization to be accelerated further by converging towards designs of interest rather than refining the entire Pareto set. This is an area recommended for future work.

Chapter 8

Multiobjective Wing Design

Having investigated a number of improvement criteria suitable for surrogate-based optimization, this chapter aims to demonstrate their use on a typical aircraft design problem that uses expensive analysis. This makes use of multiobjective improvement criteria to optimize a transonic wing with two objectives and incorporates a constrained optimization for wingbox sizing. To achieve optimization efficiency, multiple updates are used to accelerating convergence towards optimal designs.

8.1 Wing Design Problem

At transonic speeds, the geometry of a low drag wing tends to be heavily swept with a high aspect ratio (long and slender). This is typically in tension with a light weight structure that favours a tapered wing with a large inboard section. Essentially, minimizing wing drag results in a heavy wing structure and minimizing weight will hinder aerodynamic performance. Considering both aerodynamic performance and structural weight simultaneously is essential when making decisions on the most cost effective aircraft designs. This naturally leads to a multidisciplinary problem with the aim of seeking trade-offs between aerodynamic performance and structural weight, a typical problem in aerospace design, see Keane and Scanlan [2007], Lam et al. [2009] and Vicini and Quagliarella [1998].

Here we investigate a wing design problem based on a 150 seat passenger aircraft flying at a cruise speed of Mach 0.8 and an altitude of 11,000 m. The wing is based on the ONERA-D airfoil section and is required to produce enough lift for a maximum cruise weight of 60,000 kg. The optimization aims to minimize the wing drag area D/q and the wingbox structural weight. To ensure the wing can carry enough fuel, a constraint also exists on the wing volume.

The wing is characterized by four design variables listed in Table 8.1. Each wing is first evaluated at several angles of incidence using the FP flow solver. Based on a linear increase in lift with increasing angle of incidence, wing drag and aerodynamic loads are computed at the angle of incidence needed to achieve the required lift (see Appendix C for more details on aerodynamic analysis and FP). Computing this drag objective for each wing design typically takes 30 minutes on a single processor.

TABLE 8.1: Wing design variables.

Lower limit	Baseline design	Upper limit	Quantity
6	7.5	9	Root chord, m
15	17.5	20	Span, m
0.2	0.5	0.8	Taper ratio
10	25	30	Quarter chord sweep angle, deg

Figure 8.1 illustrates the predicted pressure distribution over the upper surface of the baseline wing design. To achieve the required lift at cruise conditions, this wing has an angle of incidence of 2.75 degrees resulting in a wing drag area D/q= 1.988 m^2. The sharp changes in pressure illustrate the presence of shock waves, a typical feature of transonic flow and a large contributor to drag.

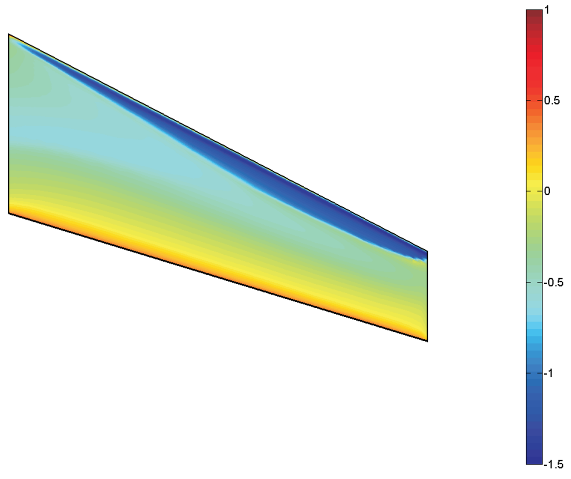


FIGURE 8.1: Upper surface pressure distribution for the baseline wing design.

Once the drag objective is computed, the resulting aerodynamic loads are used to size the wingbox structural components. In early stages of aircraft design, structural sizing is typically performed using some iterative dimensioning process to find a wingbox structure that withstands the aerodynamic loading. Instead of sizing structural components iteratively, we perform a sub-optimization to find a wingbox structure with minimum weight and that satisfies structural stress, displacement and buckling constraints. Since structural analysis is typically expensive, this sub-optimization is also performed using surrogates and this wingbox sizing is analogous to the wingbox structure design problem investigated in Chapter 5. Therefore, for each wing design evaluated during the multiobjective optimization, a full surrogate-based constrained optimization is required to suitably size the wingbox structure.

In this chapter, the limits placed on the wingbox geometry differ to those in Chapter 5. To help ensure a feasible wingbox structure is found for each wing, the upper limits of the spar, rib and skin thickness are increased. This makes the structural displacement and buckling constraints easier to satisfy, increasing feasible design space. The design variables and constraint limits used during the wingbox sizing are given in Table 8.2.

Table 8.2: Wingbox sizing design variables and constraint limits.

Lower limit	Upper limit	Quantity
0.010	0.080	Spar thickness, m
0.010	0.080	Rib thickness, m
0.010	0.060	Skin thickness, m
0.020	0.200	Rib pitch, m
1.50	—	Stress safety factor
—	1.50	Tip displacement, m
1.00	—	Buckling eigenvalue

The time taken to evaluate the wingbox weight depends on the choice of improvement criteria used during the sub-optimization. Here, the wingbox sizing makes use of several computers using the multiple update improvement criteria for constrained problems introduced in Chapter 5. Here we use either EIPFkb or EIvsPF when selecting multiple updates. Using an initial sample of 20 and 20 update stages each with four updates (100 total evaluations), a wingbox structure is sized in roughly 75 minutes using EIPFkb and 45 minutes when using EIvsPF (CPU costs extrapolated from Table 5.4 in Chapter 5). A comparison of EIPFkb and EIvsPF when optimizing the wingbox structure for the baseline wing is given in Table 8.3. This includes the optimized wingbox design variables, final wingbox weights and the resulting wing volume. In this particular example, EIPFkb converges towards a design with a smaller rib thickness, resulting in a slightly reduced wingbox weight and a small increase in wing volume.

To summarize the evaluation of a single wing, the wing drag and aerodynamic loads are first computed using FP based on a fixed aircraft cruise weight of 60,000 kg. Next, the wingbox is sized according to the aerodynamic loads using a surrogate-based constrained sub-optimization. Combining the overall evaluation time of the drag prediction and the wingbox sizing, evaluating a single wing using EIPFkb during the wingbox sizing takes 1.75 hours. This reduces to 1.25 hours when using EIvsPF during the wingbox sizing.

TABLE 8.3: Baseline wingbox design variables using two different improvement criteria.

EIPFkb	EIvsPF	Quantity
0.010	0.010	Spar thickness, m
0.015	0.017	Rib thickness, m
0.032	0.032	Skin thickness, m
0.041	0.040	Rib pitch, m
1.78	1.78	Stress safety factor
1.50	1.51	Tip displacement, m
1.10	1.14	Buckling eigenvalue
10,958	11,203	Wingbox weight, kg
35.77	35.68	Wing volume, m^3

Based on the wing planform and the final wingbox structure, the wing volume can be quickly computed. Since this depends on the computation of the wingbox structure, this constraint is also expensive to compute and modelled using a surrogate model. Figure 8.2 provides an overview for the constrained multiobjective optimization and Figure 8.3 illustrates the constrained sub-optimization used for the wingbox sizing.

8.2 Optimization Comparison

Due to the expense of this optimization problem, we limit our comparison to two different methods. Method 1 utilizes improvement criteria proposed elsewhere. This uses EIhkb for the multiobjective optimization and EIPFkb for the selection of multiple updates during the wingbox sizing. Method 2 uses new improvement criteria discussed within this thesis. This uses the goal-based improvement criteria, EIhg, for multiobjective optimization and EIvsPF for the wingbox sizing. In both methods, four designs are evaluated in parallel. Since four processors are required for the wingbox sizing, this optimization is based on 16 parallel processes. The optimization is initiated with an optimized Latin hypercube with 40 designs, plus the baseline design, and updated for 10 update stages. This results in a total of 81 evaluations using the drag prediction and wingbox sizing analysis (each of which requires 100 evaluations of the structural analysis).

An initial comparison of each method is based on the hypervolume indicator. This is used to illustrate the performance of each method after each update stage, allowing the most efficient method to be identified. The expense of each method is also an important consideration and illustrated to highlight some key advantages when using method 2. Finally, we provide a visual comparison of the final Pareto sets and briefly discuss some of the optimized designs.

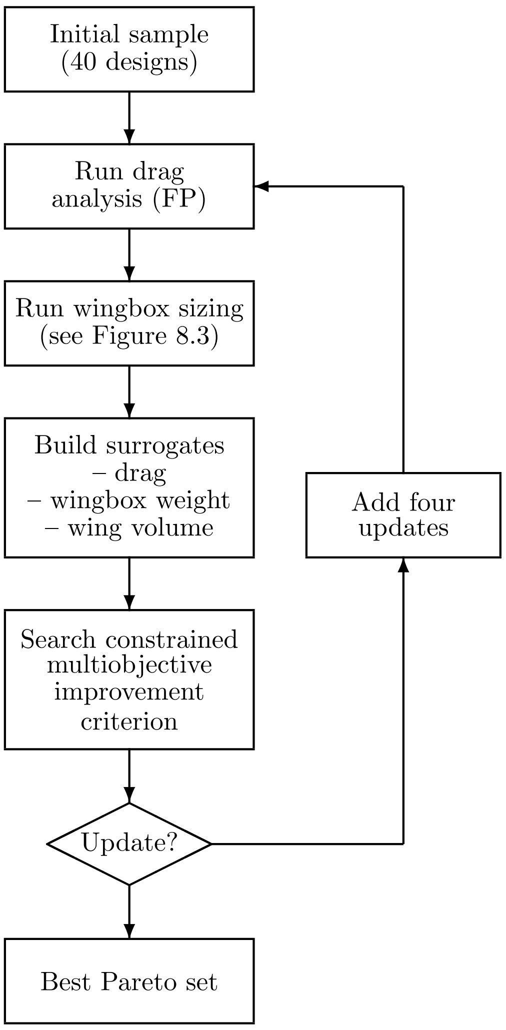


Figure 8.2: Constrained multiobjective wing design optimization.

The reader should note that in this test problem, averaging of results is intractable and each method has only been tested once. The following comparison is therefore unlikely to reflect the overall ability of each method but highlights key advantages and provides an example of how surrogate-based optimization can be used on a real engineering design problem.

8.2.1 Hypervolume Indicator

In a similar manner to previous chapters, the hypervolume indicator is used to measure the performance of each method. The hypervolume for each method is illustrated after each update stage in Figure 8.4. Before any updates are added, the hypervolume is based solely on the initial sample of evaluated designs. This suggests that, in terms of the hypervolume, there is little difference between using EIPFkb or EIvsPF during the wingbox sizing. This is better illustrated in Figure 8.6, showing equivalent Pareto sets

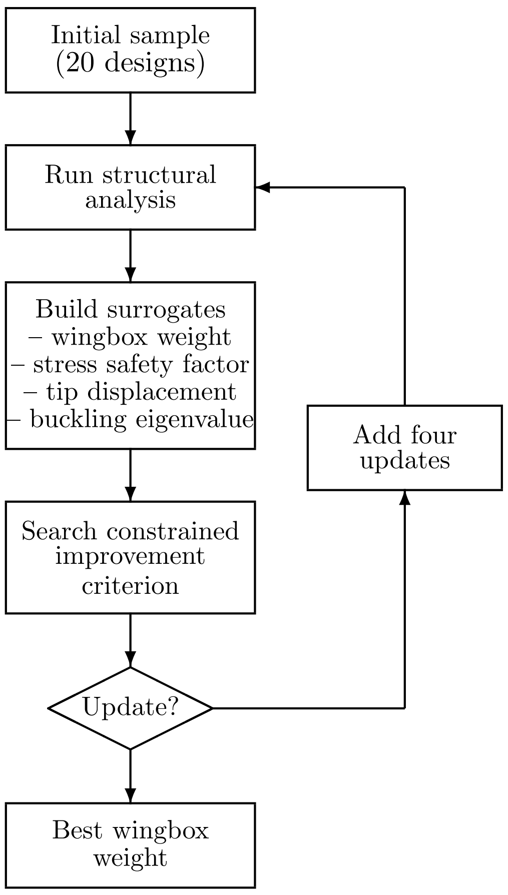


FIGURE 8.3: Wingbox sizing constrained optimization.

before any updates are added. Once updates are added, both methods select designs that contribute towards a better Pareto set and increase the hypervolume. In this problem, selecting new updates using the goal-based approach works well and method 2 rapidly identifies new designs that increase the hypervolume. This leads to a better Pareto set much earlier on in the optimization. Method 1 is clearly less efficient and requires many more updates to achieve a similar hypervolume. This reflects a similar comparison between the performance of EIhkb and EIhg given in Chapter 7.

8.2.2 Optimization Cost

Next we consider the optimization cost of each method. Since each method uses different improvement criteria to select new updates, each method has different computational demands. The most significant difference between each method exists during the wingbox sizing. Using EIvsPF in method 2 reduces the computational demands when searching for multiple updates. This makes evaluating the wingbox weight cheaper compared to EIPFkb and the savings become significant when sizing many wing designs. Over the

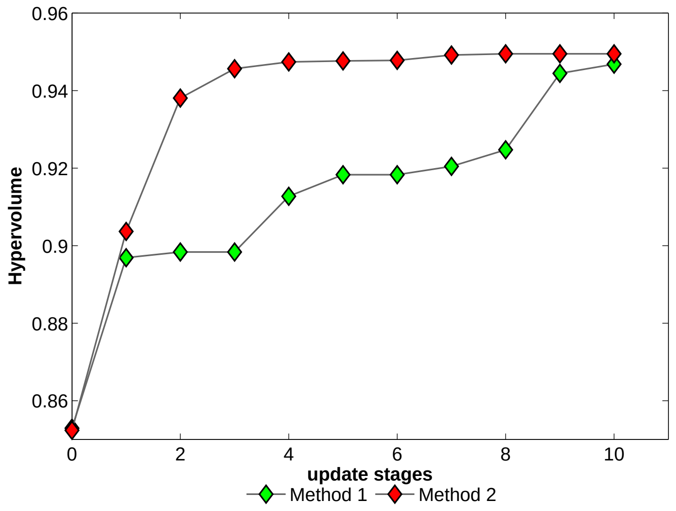


FIGURE 8.4: Comparison of method 1 and method 2 on the wing design problem: Hypervolume indicator after each update stage.

entire optimization (81 designs based on four parallel runs) a 30% saving in evaluation time is achieved using EIvsPF. Method 2 also reduces the computational burden (although insignificant in this case) when searching for multiple updates using EIhg. This is because multiple updates can be computed in parallel and the goal-based hypervolume is less computationally expensive to compute, see Section 7.2.2.2.

To illustrate differences between each method, an estimation of the optimization run time is given in Figure 8.5. This includes the wing evaluation time (including drag analysis and wingbox sizing), time spent searching the improvement criteria (EIhkb or EIhg) and the total tuning costs. It is clear that the tuning costs and searching the two improvement criteria is insignificant compared to the wing evaluations. This is typical of an expensive optimization problem and clearly warrants the use of surrogate models. Overall, the optimization using method 1 takes approximately 36 hours whilst method 2 takes 25 hours. These time savings are exaggerated when evaluating more designs in parallel and for further update stages, highlighting the benefits associated with employing the improvement criteria used in method 2.

The reader should note that both method 1 and method 2 offer significant time saving over a conventional approach by utilizing multiple updates. Although we do not provide a direct comparison between single and multiple updates on this wing design problem, the author suggests the entire optimization would take, at best, one week to converge towards an equivalent Pareto set when using single updates.

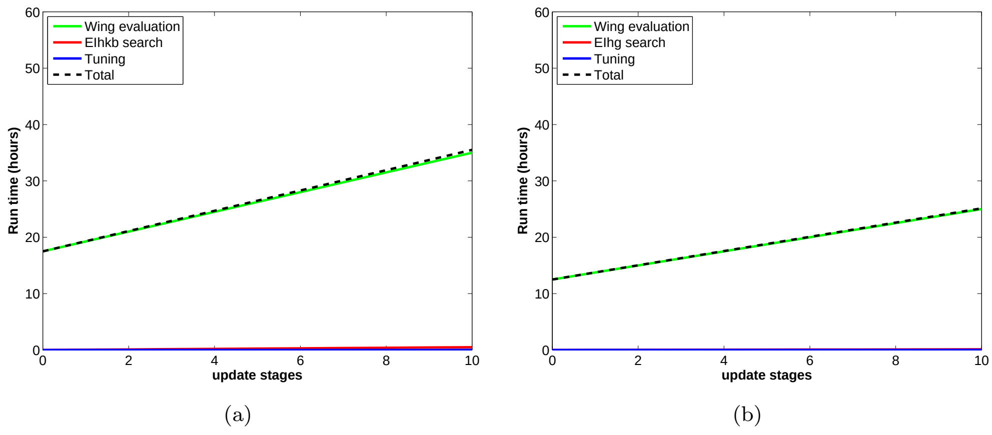


Figure 8.5: Comparison of method 1 and method 2 on the wing design problem: Optimization run time using (a) method 1 and (b) method 2.

8.2.3 Final Pareto Set

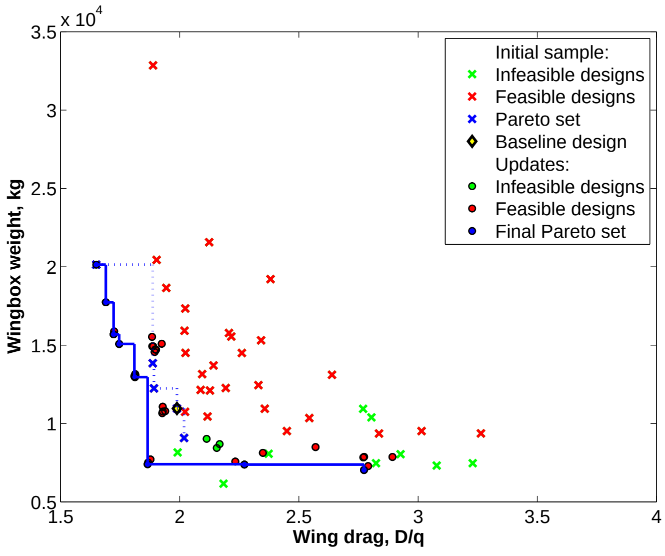
Our final comparison is of the final Pareto sets achieved after all ten update stages. For each method, Figure 8.6 illustrates all evaluated designs and the final Pareto sets. Both methods select new designs that are below and to the left of all feasible initial sample points, contributing towards a better Pareto set. Furthermore, designs have been identified that improve on the baseline design to simultaneously reduce the wing drag area and the wingbox weight.

The final hypervolume achieved by each method is very similar (0.947 and 0.950, respectively) but the appearance of each Pareto set is quite different. Method 2 finds a Pareto set that is more populated and has a larger spread of designs. This gives the designer a wide choice of designs, with four explicitly improving on the baseline design in both objectives. Method 1 finds a Pareto set with a poorer spread and is less well populated but benefits from a design that dominates others identified using method 2. A better comparison of the final Pareto sets is given in Figure 8.7.

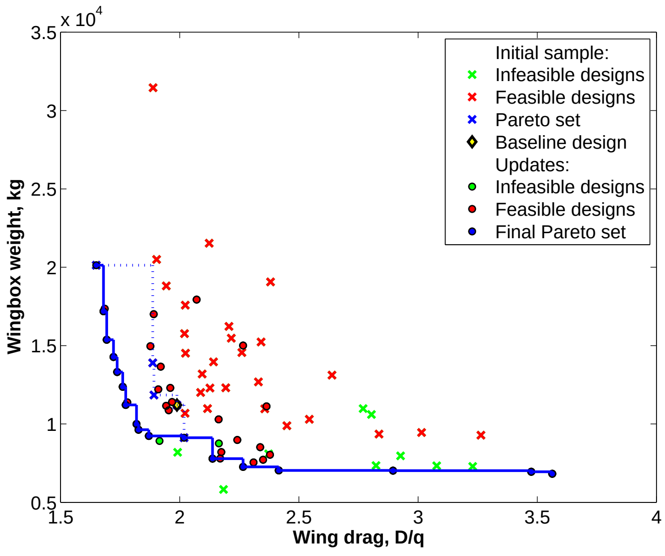
From a designer's point of view, method 1 is less efficient and computationally more expensive than method 2. On the other hand, method 1 finds a dominating design that method 2 does not. This particular design improves on the baseline design quite significantly and is a likely choice for further analysis.

8.3 Further Discussion

Since the final Pareto sets include a range of designs, it is interesting to look at some selected designs in more detail. Figure 8.8 illustrates the wing planform and wingbox design for the baseline design and three Pareto optimal wing designs. Design 1 is a low drag design, located at the far end of the Pareto set in Figure 8.6(b). Design 2



(a)



(b)

FIGURE 8.6: Comparison of method 1 and method 2 on the wing design problem: Evaluated designs and final Pareto set after 10 update stages using (a) method 1 and (b) method 2.

is a low wingbox weight design found on the far right and design 3 offers a reasonable trade-off between low drag and low wingbox weight. This is the design that dominates the baseline design in both objectives in Figure 8.6(a).

Table 8.4 examines the design variables, objective values and some derived parameters in more detail. As expected, a low drag wing favours a heavy sweep and large aspect ratio. This geometry helps to reduce shock wave and lift induced components of drag but leads to a large wing structure with a significant weight penalty. In contrast, design 2 achieves minimum wingbox weight by increasing the root section, reducing the aspect ratio and reducing the taper ratio. This helps to resist deformation of the wing and

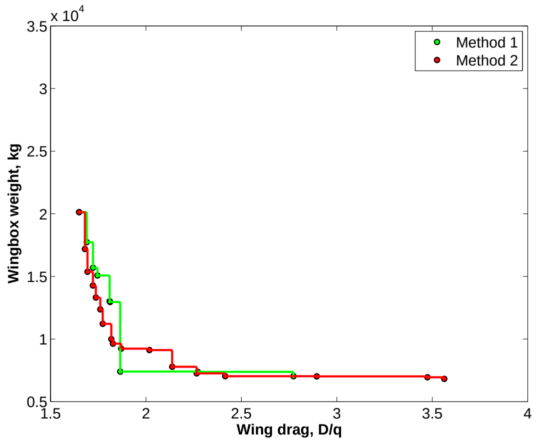


FIGURE 8.7: Final Pareto set comparison.

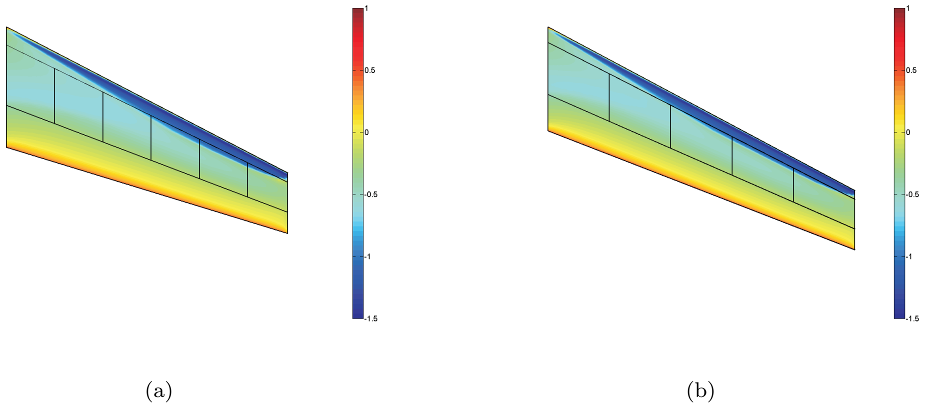
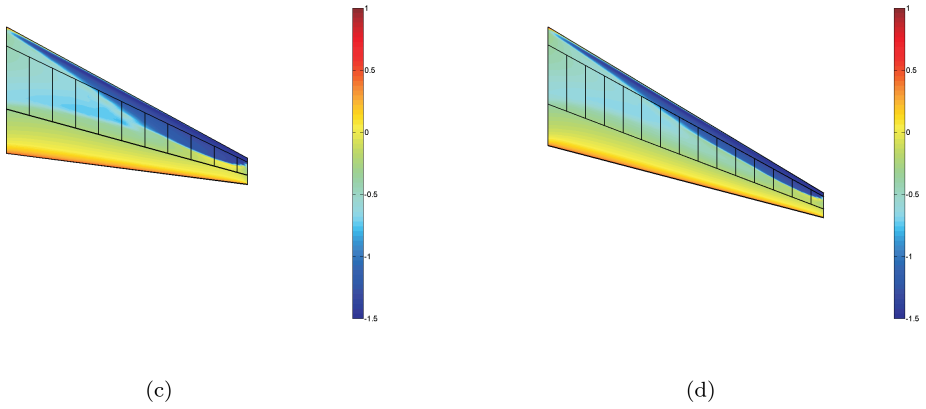



FIGURE 8.8: Planform and wingbox design with upper surface pressure distribution for the (a) baseline design, (b) design 1, (c) design 2 and (d) design 3.

reduces stresses at the wing root. As a consequence, a large angle of incidence is needed to achieve the required lift, resulting in strong shocks and high drag. Unsurprisingly, design 3 offers a sensible compromise between low drag and low weight. This wing is likely to be the most promising, sharing similarities with typical wings used on most transonic wide body airliners currently in operation. This design has a large sweep for low drag at transonic flight conditions and a large root cord and taper to better distribute the aerodynamic loading. When compared to the baseline design, a larger sweep helps to reduce the component of wave drag due to shocks whilst a balance of span, taper ratio and root chord reduce the wingbox weight and drive the design towards the wing volume constraint.

TABLE 8.4: Design variables and objective function values for the baseline wing and three Pareto optimal designs.

Baseline design	Design 1	Design 2	Design 3	Quantity
7.5	6.5	7.9	7.4	Root chord, m
17.5	19.1	15.0	17.2	Span, m
0.50	0.57	0.21	0.21	Taper ratio
25.0	26.4	23.7	27.3	Quarter chord sweep angle, deg
98.4	96.8	71.4	76.4	Wing area, m^2
6.93	8.32	7.15	8.61	Aspect ratio
2.75	2.63	3.60	3.21	Angle of incidence, deg
1.988	1.650	3.56	1.865	D/q, m^2
10,958	20,130	6,823	7,402	Wingbox weight, kg
35.8	27.6	24.4	24.3	Wing volume, m^3

In reality, the weight penalty associated with design 1 is likely to deem it impractical. Design 3 may also be problematic due to its high drag, penalizing aircraft range and operational costs. Although it is a good idea to find a broad range of designs, spanning the entire Pareto front, when evaluations are expensive searching for designs that are clearly impractical becomes an inefficient use of resources. It may be possible to narrow the design space or incorporate more constraints to avoid the inclusion of impractical designs, however, this is likely to limit the optimization to only small design changes and may exclude promising designs. In this particular problem, more practical designs are achievable by introducing additional wing design parameters. Introducing wing camber for example, is likely to reduce the wing drag and increase lift at smaller angles of attack. Similarly, introducing wing twist will allow better distribution of aerodynamic loads, helping to reduce wingbox weight.

More realistic results are also achievable through more detailed analysis. This may incorporate more sophisticated CFD codes or nonlinear structural analysis but also better treatment of aerodynamic loads and coupling effects. This usually requires an iterative process and should account of aeroelasticity. The reader is directed to Hürlimann [2010] for an example of more detailed analysis that can be incorporated in to a similar study performed here.

To help reduce the cost of multiobjective optimization further, it may be sensible to incorporate user preferences into the search. This can help to focus the search on regions of the Pareto front that are of particular interest, the region around a baseline design for example. This is becoming a popular approach and has been recently applied aerodynamic problems by Carrese et al. [2011, 2012]. Although not considered further in this thesis, this can be achieved quite simply using the goal-based approach by specifying goal points based on user preferences.

8.4 Conclusions

In this chapter we have demonstrated the use of surrogate-based multiobjective optimization on a wing design problem incorporating a surrogate-based constrained optimization for wingbox sizing. This compares the performance of two methods that make use of 16 parallel processes to accelerate convergence towards a set of optimal trade-offs. Method 1 is based on improvement criteria proposed elsewhere, using the hypervolume-based expected improvement for multiobjective optimization and expected improvement and probability of feasibility for wing box sizing. In method 1, both improvement criteria use the Kriging believer strategy to select multiple updates. Method 2 uses the goal-based expected improvement for multiobjective optimization and a multiobjective search to select multiple updates during the wingbox sizing. Method 2 is based on improvement criteria introduced within this thesis.

Both methods converge towards a Pareto set of designs within a reasonable time frame, offering significant improvements over an existing baseline design. When basing the performance of each method on the hypervolume, method 2 is clearly more efficient, supporting the use of the goal-based improvement criterion to select multiple updates. By using a multiobjective search to select multiple updates during the wingbox sizing, method 2 also reduces the overall run time by 30%. When comparing the final Pareto sets, method 2 offers a wide choice of designs but method 1 benefits from a single dominating design, unidentified by method 2.

A number of simplifications in the analysis of each wing design has led to some optimal designs being impractical in some sense. More realistic designs require a more detailed analysis incorporating aeroelastic effects, and iterative calculations to compute the required lift, aircraft range and fuel payload. Nonetheless, the methods tested here have demonstrated their practical use on an aircraft design problem incorporating expensive analysis.

Chapter 9

Conclusions and Future Work

In engineering design, the designer is often faced with a number of decisions when seeking new and improved designs. When the design process involves time consuming simulations, surrogate models can be used to emulate the expensive analysis and used for automated and wide ranging design search and optimization. By reducing the number of designs evaluated using the expensive analysis, surrogate-based optimization offers a realistic alternative to direct optimization using traditional methods. The efficiency and reliability of surrogate-based methods depends heavily on the choice of improvement criterion used to select model update points, the new designs to be evaluated. The research within this thesis has focused on developing efficient improvement criteria for constrained and multiobjective problems, leading to several contributions to the field of surrogate-based optimization. These contributions include:

- Enhanced improvement criteria for constrained problems have been investigated, demonstrating an increase in efficiency when selecting updates along constraint boundaries.
- Dealing with constraints in surrogate-based optimization using a multiobjective search has demonstrated improvements in reliability and efficiency and well suited for selecting multiple updates to further accelerate convergence towards global solutions.
- Multiobjective improvement criteria have been extended to select multiple updates, further accelerating convergence towards Pareto optimal solutions.
- A novel goal-based approach to surrogate-based multiobjective optimization has been introduced, suitable for selecting both single and multiple updates, offering improved performance over the other approaches tested.

In Chapter 3 different improvement criteria useful for handling inequality constraints in surrogate-based optimization were reviewed and tested on different test problems.

Exploiting the surrogate prediction works well when optimizing simple, unimodal problems but is not guaranteed to find global solutions when the objective is more complex and multimodal. Although less efficient on very simple problems, expected improvement performs better when optimizing more complex objective functions. This is combined with a penalty or probabilistic approach for constraint handling. A simple penalty performs well when the constraints are simple to approximate but fails to find the best feasible design when the constraints become more complex. Using the probability of feasibility relaxes constraint boundaries, identifying the region of the global optimum on even very complex and multimodal problems but prevents convergence to the most accurate solutions. Although the correct choice of improvement criterion suitable for constraint handling appears problem dependent, combining expected improvement of the objective and probability of feasibility of the constraints generalized well to an aircraft wing design problem, providing motivation in Chapter 4 to develop probabilistic approaches further.

The performance of the improvement criteria suitable for constrained optimization was enhanced in two main ways. Firstly, the enhanced probability of feasibility and expected feasibility were formulated to encourage updates to be selected along constraint boundaries. Handling constraints based on the enhanced improvement criteria better models likely regions of optimal designs but is only suitable when global solutions are tight. The second enhancement uses a multiobjective search to better balance the improvement of the objective and feasibility of the constraints. This approaches the problem differently by avoiding an aggregated search space. This identified better updates leading to significant improvements in performance when dealing with complex or multiple constraints.

Using a multiobjective search leads to a Pareto set of solutions that are all optimal in some sense. In the case of constrained optimization, this leads to a trade-off between solutions with a high predicted objective improvement but low constraint feasibility and vice versa. This lends itself to the selection of multiple updates that can be evaluated in parallel, making use of any additional computing resources available. This is investigated further in Chapter 5 and compared with other multiple update improvement criteria. Assuming four designs can be evaluated in parallel, it was found that a significant reduction in the overall wall clock time can be achieved when selecting multiple updates. These improvement criteria were further tested on a wingbox structure problem, evaluating designs based on a finite element method.

In Chapter 6 we moved away from constrained optimization and investigated improvement criteria suitable for multiobjective problems. This included the multiobjective probability of improvement and three variations of multiobjective expected improvement. In general, the probability of improvement performed poorly and clear benefits are associated with formulating multiobjective expected improvement. The best formulation of expected improvement is unclear but basing the improvement on the change in hypervolume offered significant enhancements in performance on two of the four problems tested but with the drawback of a higher computational burden. Chapter 7 extended the multiobjective improvement criteria to select multiple updates. In addition to extending existing methods, Chapter 7 also introduced a new goal-based approach suitable for selecting multiple updates in multiobjective optimization. This goal-based approach was applied to existing improvement criteria, offering very promising results when formulating the expected improvement using the goal-based improvement in hypervolume. Although this goal-based approach was developed with the intention of selecting multiple updates, it has also been used successfully to select single updates, offering a competitive performance with other methods investigated in Chapter 6.

The applicability of the surrogate-based methods to an expensive design problem was demonstrated in Chapter 8 on a constrained multiobjective wing design problem. Performance enhancements associated with the handling of constraints using a multiobjective search and dealing with multiple objectives using a goal-based approach was demonstrated and compared with another method using improvement criteria available outside of this thesis. Both methods utilize the selection of multiple updates, taking advantage of 16 parallel processes. The best quality Pareto set was found using methods developed during this research and demonstrates a reduced computational burden.

From this research a number of areas have been identified suitable for further research. In Chapter 3 we pointed out the potential to hybridize different improvement criteria. This may provide particularly useful in constrained problems where the development of constraint surrogates is vital for good results to be achieved. This may involve using some of the enhanced improvement criteria discussed in Chapter 4 to exploit constraint boundaries towards the end of an optimization. Another avenue for potential research in this area is to utilize the conditional likelihood discussed in Jones [2001] and Forrester et al. [2008] for exploiting constraint boundaries.

Chapter 4 discussed some current weaknesses when dealing with constraints using a multiobjective search. Although this approach provided some significant performance gains when dealing with problems with only two design variables, the cost of finding good updates based on a Pareto set of solutions proved impractical on the 11 dimensional aircraft design problem. Further development of this method should concentrate on testing different multiobjective algorithms suitable for larger dimensions. Further enhancements may also be achieved by investigating other approaches for selecting a set of multiple updates from a Pareto set of solutions. This may involve explicitly searching for knee points, reducing the computational burden of searching multiple objectives. Coevolution may also provide an alternative approach to finding updates that offer both objective improvement and constraint feasibility and has frequently been used as an alternative to multiobjective optimization in the evolutionary computing literature [Coello Coello et al., 2007].

Many of the improvement criteria investigated in this thesis that are suitable for constraint handling in single objective problems can also be used to handle constraints when dealing with multiple objectives. It is expected that some of the performance enhancements found in Chapter 4 will translate to multiobjective problems when the Pareto set lies on constraint boundaries. It is also possible to find a Pareto set of solutions based on maximizing a multiobjective improvement criteria and maximizing the probability of feasibility providing an alternative approach to find multiple updates in constrained multiobjective problems.

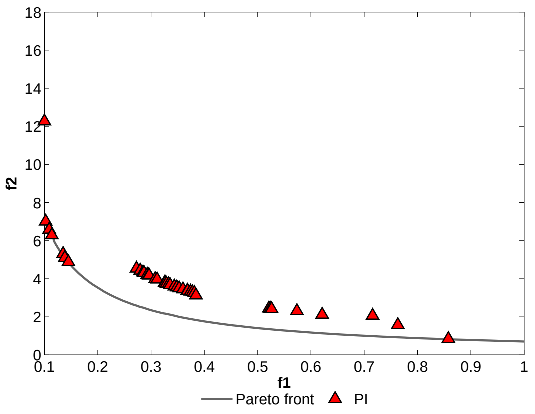
We have demonstrated that formulating goal-based improvement can reduce the complexity of multiobjective improvement criteria and lends itself to the selection of multiple updates. Many potential formulations for goal-based improvement exist and the choice of suitable goal points is endless, providing a large scope for further development. The selection of goal points may be combined with the idea of goal-seeking and the conditional likelihood to locally refine Pareto sets or to incorporate user preferences.

Although problems have not been tackled beyond two objectives, it is possible to extend the goal-based improvement criteria to tackle larger problems. Difficulties are likely to occur when extending these methods to deal with manyobjective problems but efficiently solving three and four objective problems is thought to be achievable.

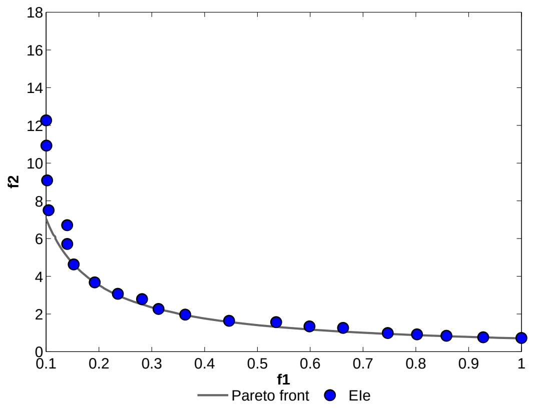
Throughout this thesis, multiple updates have been selected based on the ability to evaluate four designs in parallel. Choosing four updates is somewhat arbitrary, chosen here to match available resources on a typical modern desktop machine. In reality designers are likely to have many processors available and will have to make a decision on how many designs should be evaluated. Further research into the optimum number of updates for different test problems would provide some valuable information to designers that wish to better manage their computational resources.

Appendix A

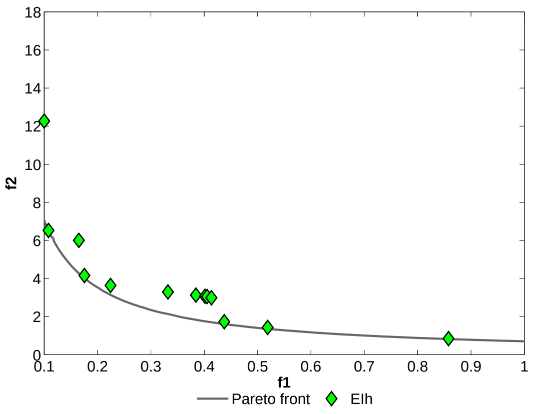
Example Pareto Sets



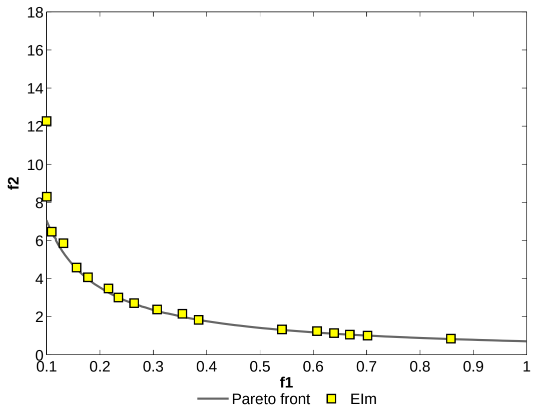
(a)



(b)

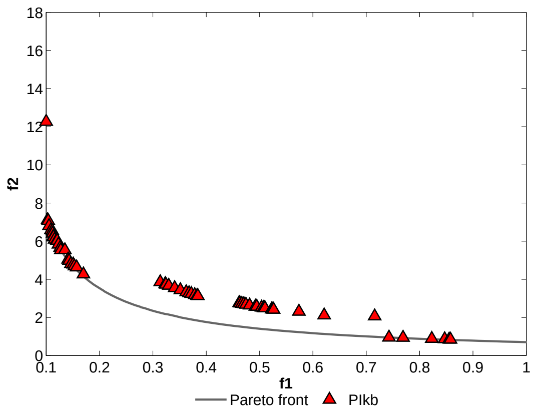


(c)

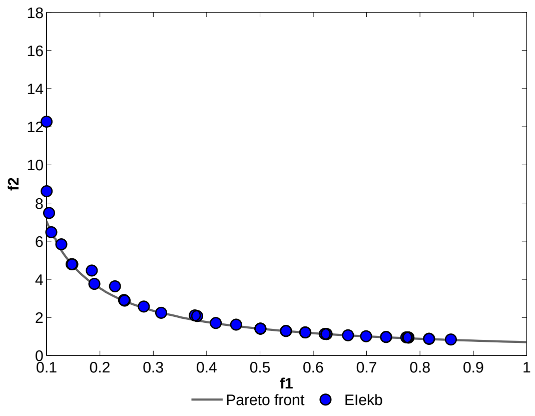


(d)

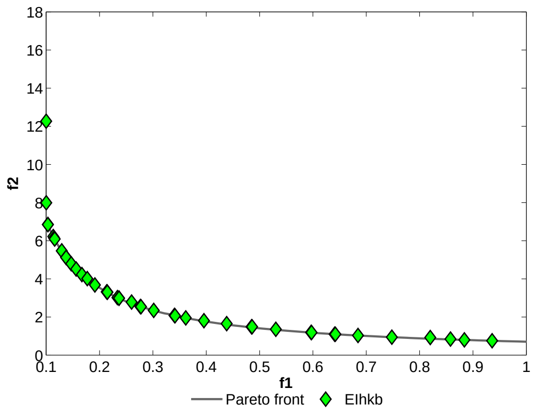
FIGURE A.1: Example Pareto sets for test problem 4 after 60 updates using (a) multi-objective probability of improvement, (b) Euclidean-based expected improvement, (c) hypervolume-based expected improvement and (d) maximin expected improvement.



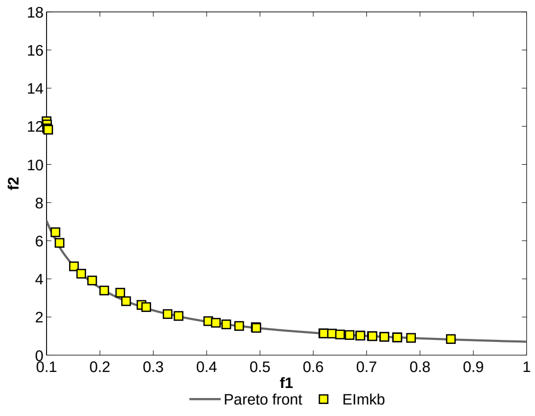
(a)



(b)



(c)



(d)

FIGURE A.2: Example Pareto sets for test problem 4 after 30 update stages using the Kriging believer strategy. (a) Multiobjective probability of improvement, (b) Euclidean-based expected improvement, (c) hypervolume-based expected improvement and (d) maximin expected improvement.

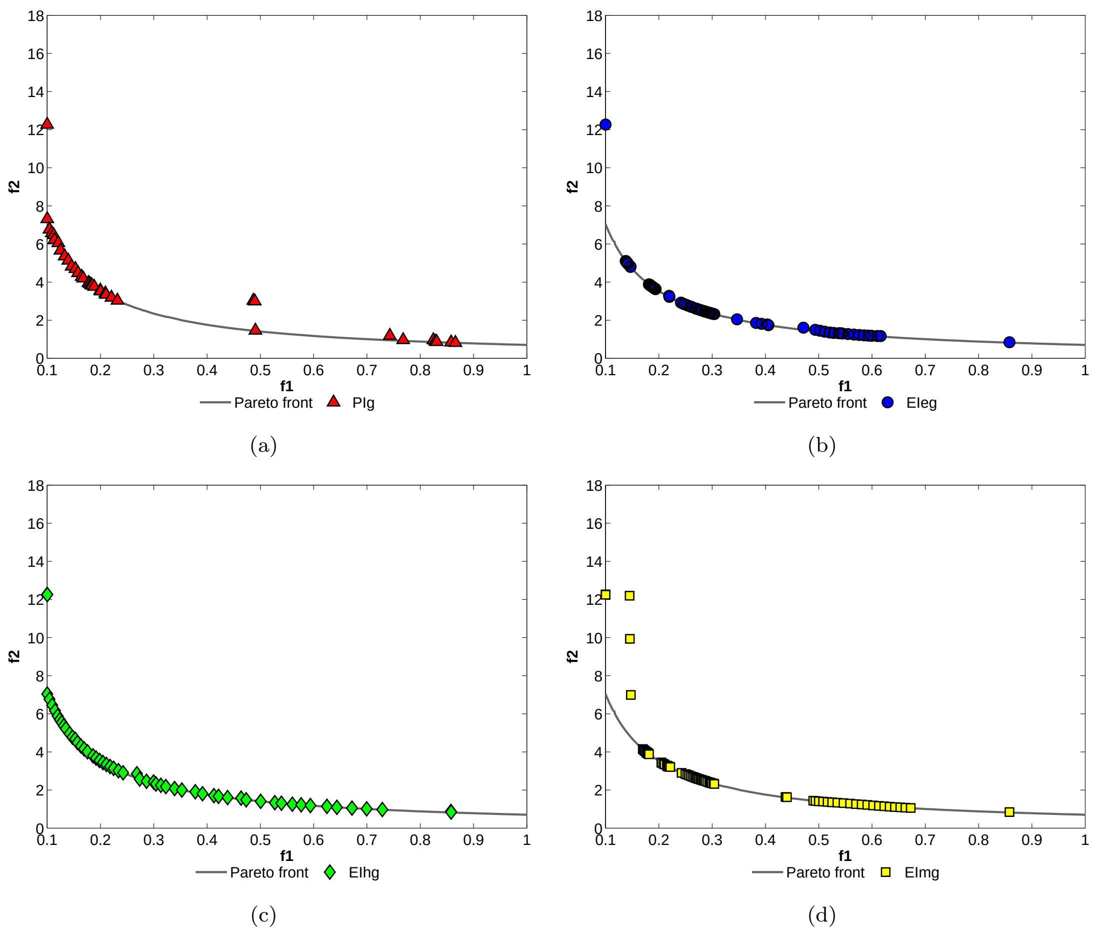
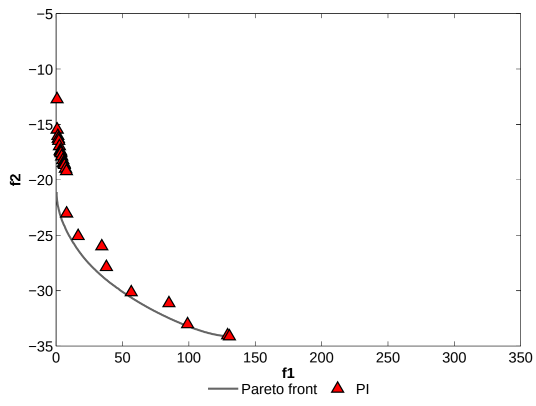
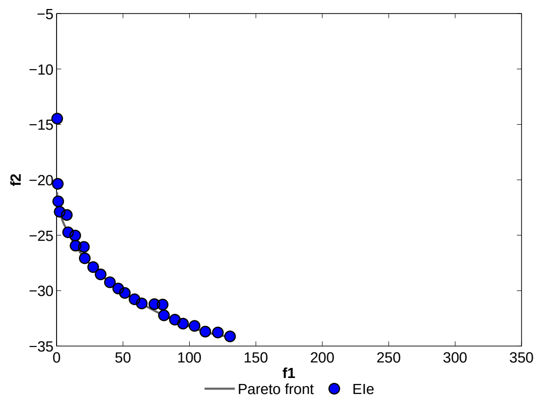


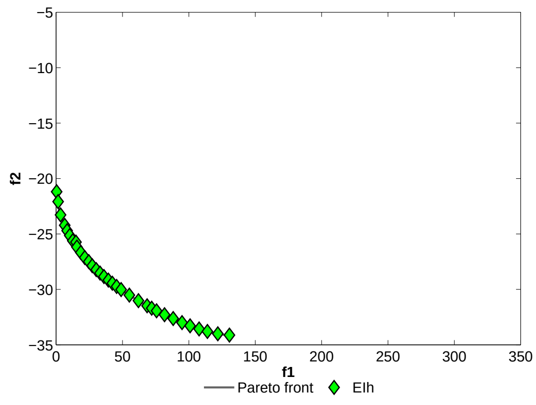
FIGURE A.3: Example Pareto sets for test problem 4 after 30 update stages using (a) goal-based multiobjective probability of improvement, (b) Euclidean goal-based expected improvement, (c) goal-based expected improvement in hypervolume and (d) goal-based maximin expected improvement.



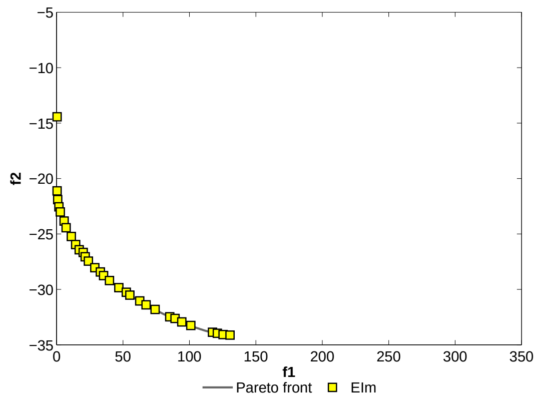
(a)



(b)



(c)



(d)

FIGURE A.4: Example Pareto sets for test problem 5 after 40 updates using (a) multi-objective probability of improvement, (b) Euclidean-based expected improvement, (c) hypervolume-based expected improvement and (d) maximin expected improvement.

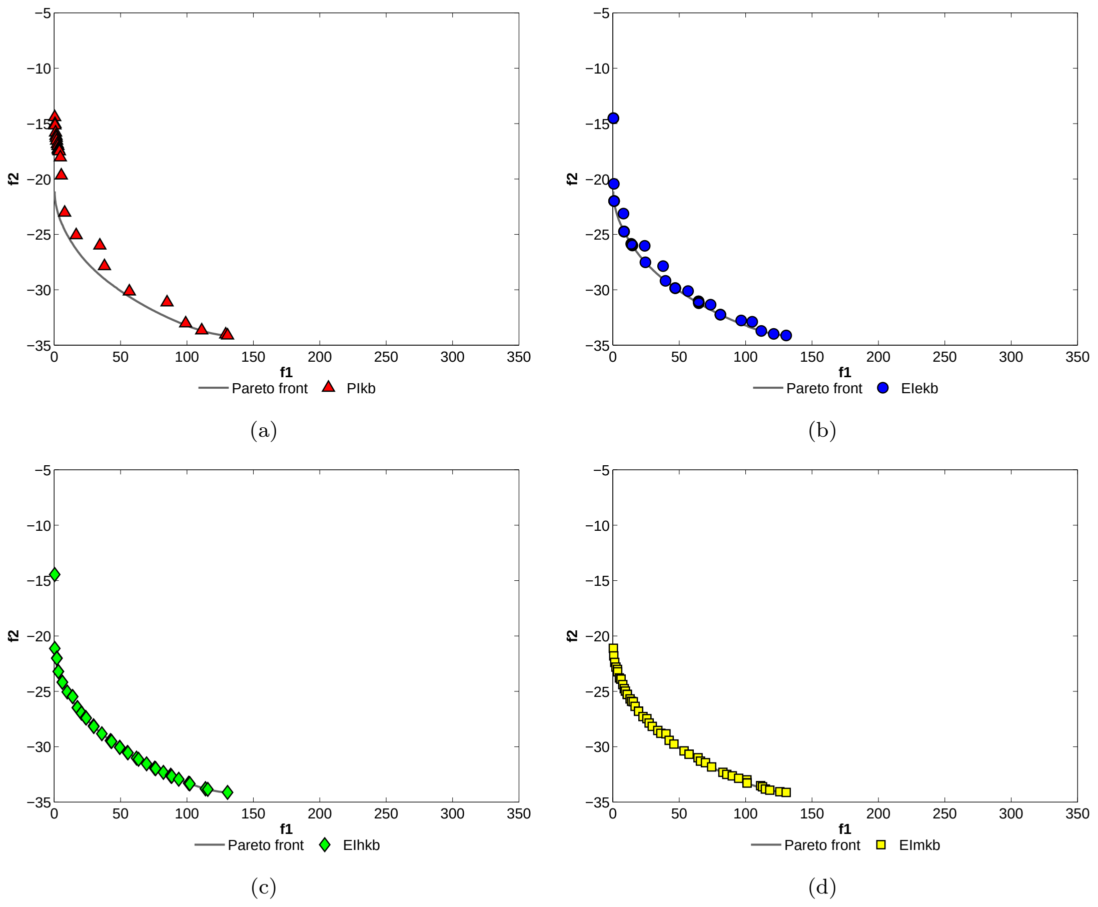


FIGURE A.5: Example Pareto sets for test problem 5 after 20 update stages using the Kriging believer strategy. (a) Multiobjective probability of improvement, (b) Euclidean-based expected improvement, (c) hypervolume-based expected improvement and (d) maximin expected improvement.

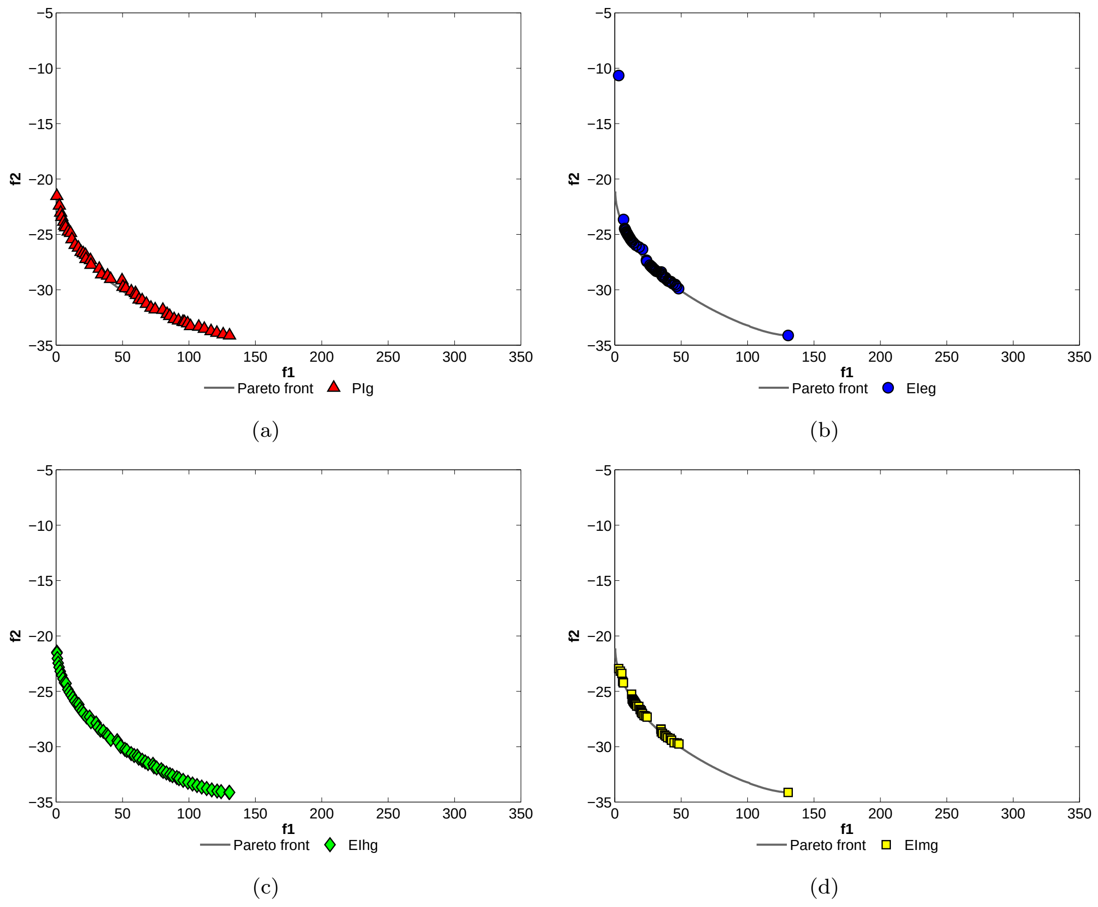
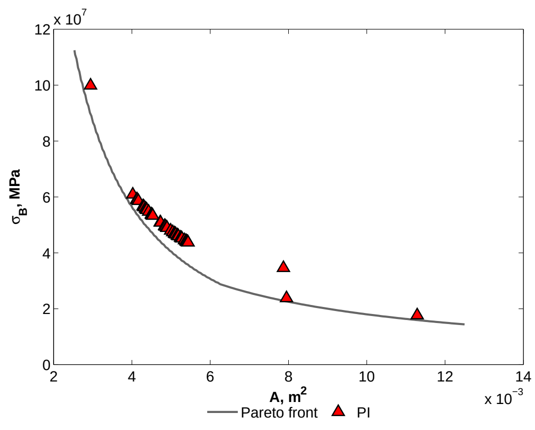
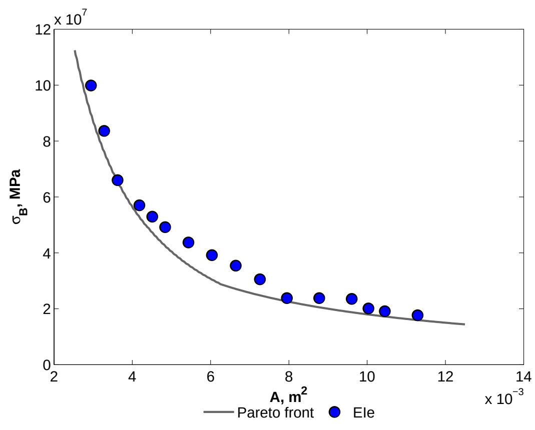


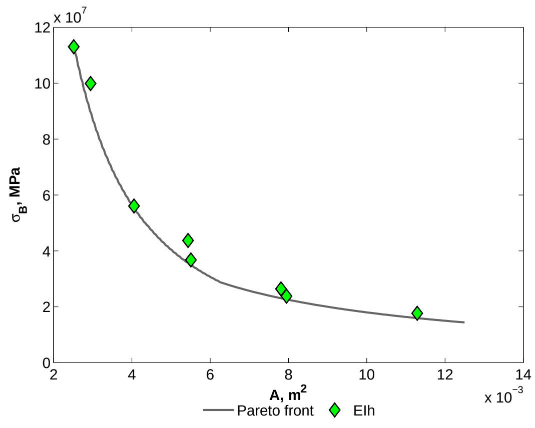
FIGURE A.6: Example Pareto sets for test problem 5 after 20 update stages using (a) goal-based multiobjective probability of improvement, (b) Euclidean goal-based expected improvement, (c) goal-based expected improvement in hypervolume and (d) goal-based maximin expected improvement.



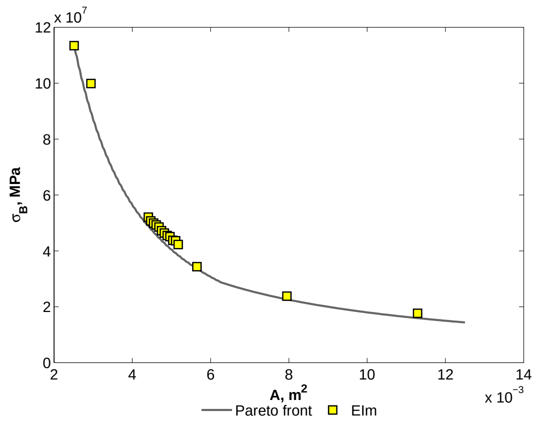
(a)



(b)

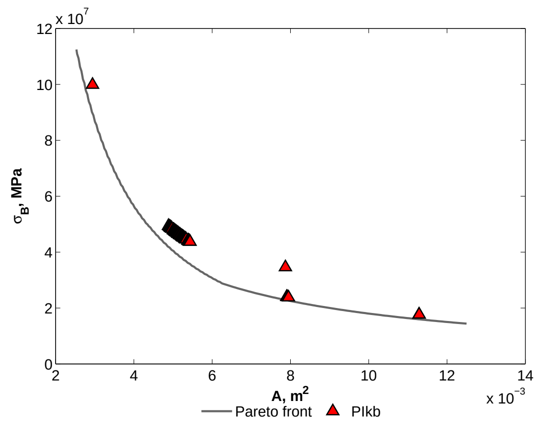


(c)

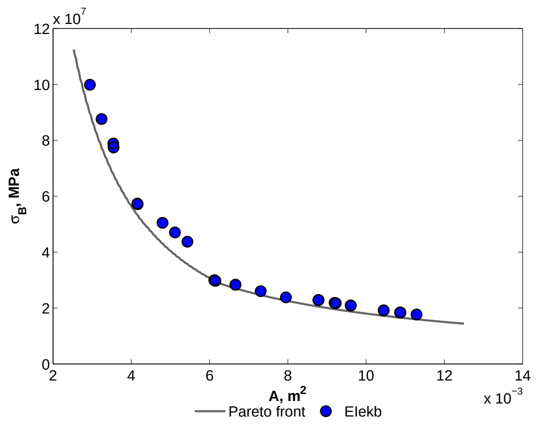


(d)

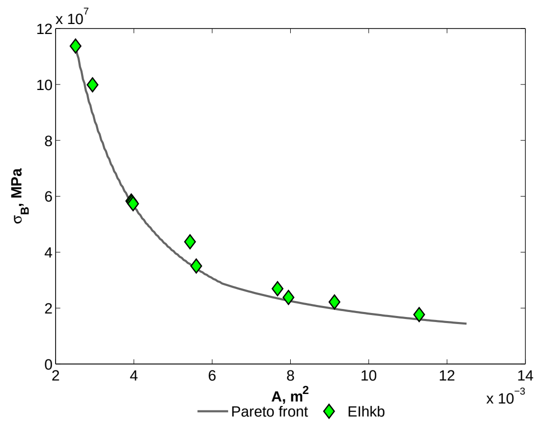
FIGURE A.7: Example Pareto sets for the Nowacki beam design problem after 40 updates using (a) multiobjective probability of improvement, (b) Euclidean-based expected improvement, (c) hypervolume-based expected improvement and (d) maximin expected improvement.



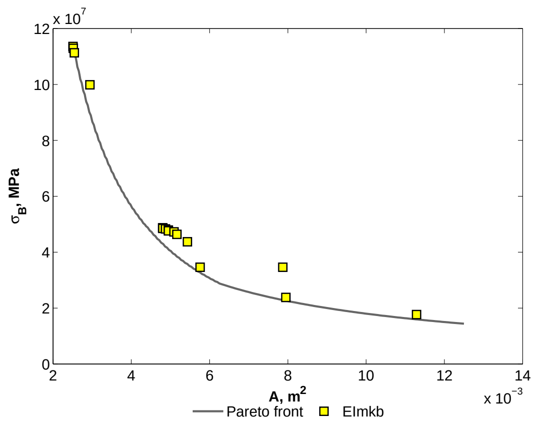
(a)



(b)

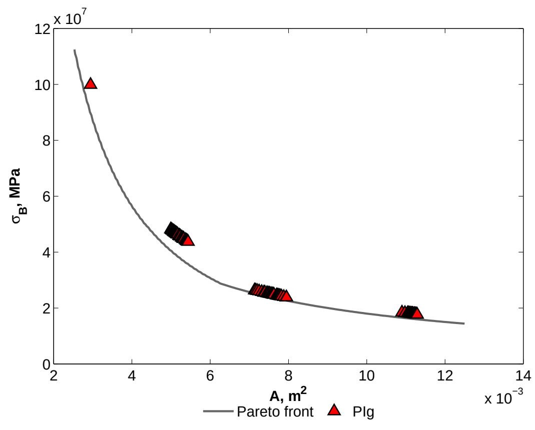


(c)

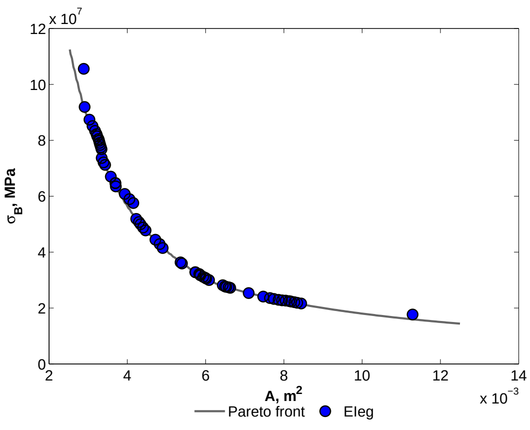


(d)

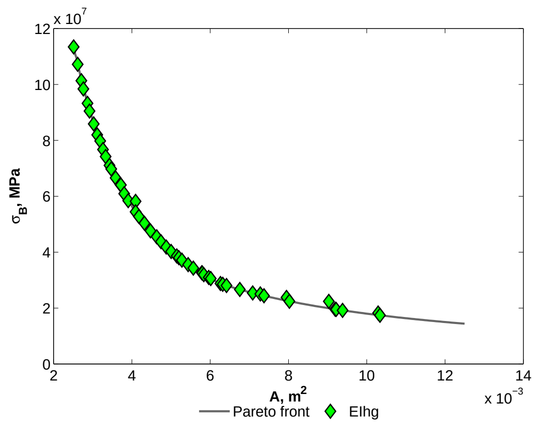
FIGURE A.8: Example Pareto sets for the Nowacki beam design problem after 20 update stages using the Kriging beliver strategy. (a) Multiobjective probability of improvement, (b) Euclidean-based expected improvement, (c) hypervolume-based expected improvement and (d) maximin expected improvement.



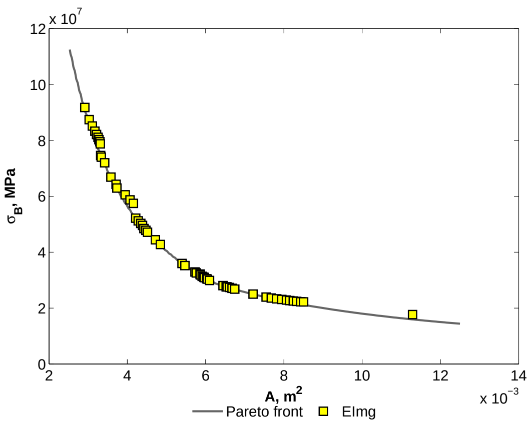
(a)



(b)

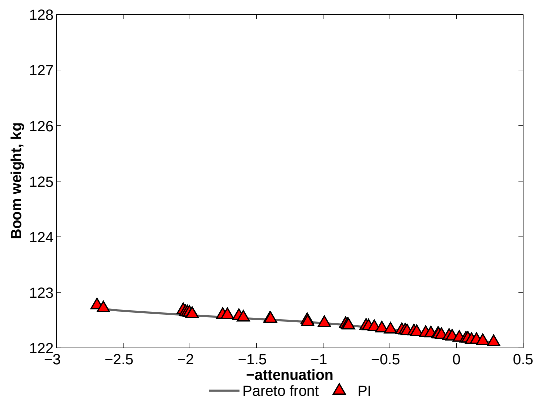


(c)

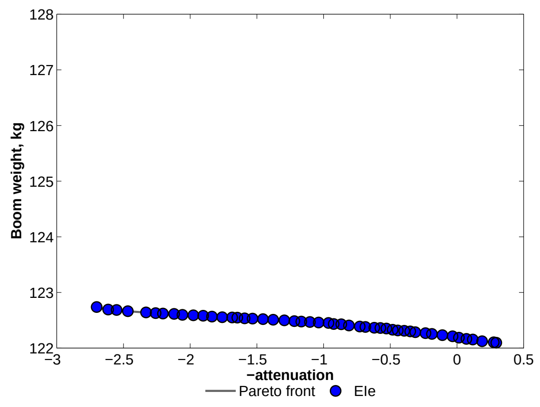


(d)

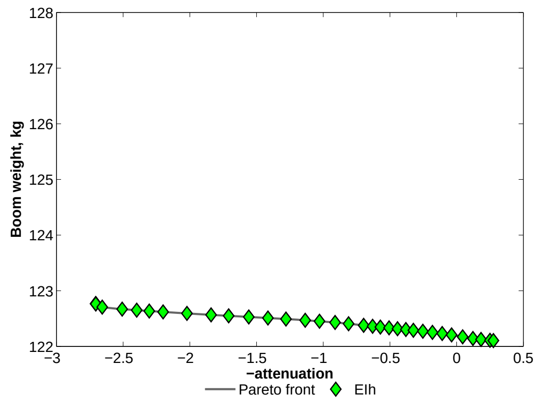
FIGURE A.9: Example Pareto sets for the Nowacki beam design problem after 20 update stages using (a) goal-based multiobjective probability of improvement, (b) Euclidean goal-based expected improvement, (c) goal-based expected improvement in hypervolume and (d) goal-based maximin expected improvement.



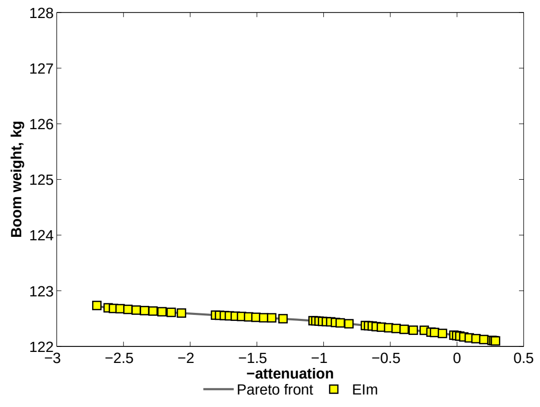
(a)



(b)

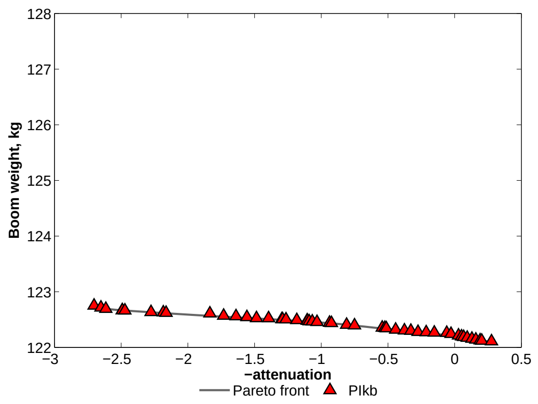


(c)

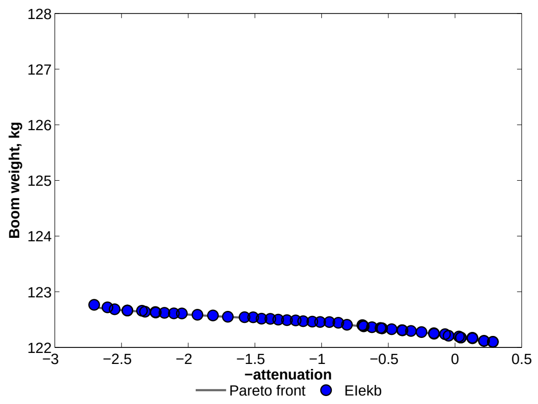


(d)

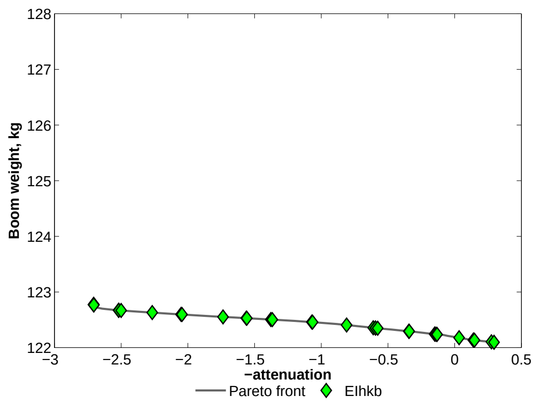
FIGURE A.10: Example Pareto sets for the satellite boom design problem after 60 updates using (a) multiobjective probability of improvement, (b) Euclidean-based expected improvement, (c) hypervolume-based expected improvement and (d) maximin expected improvement.



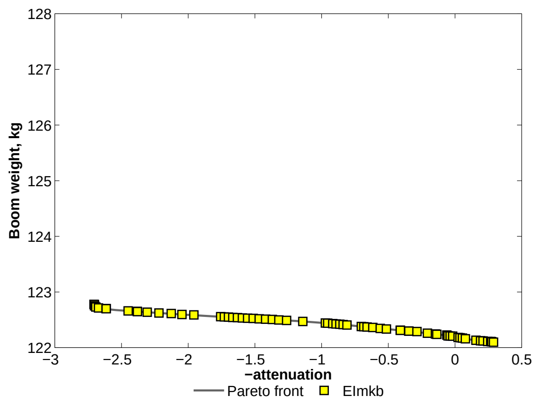
(a)



(b)

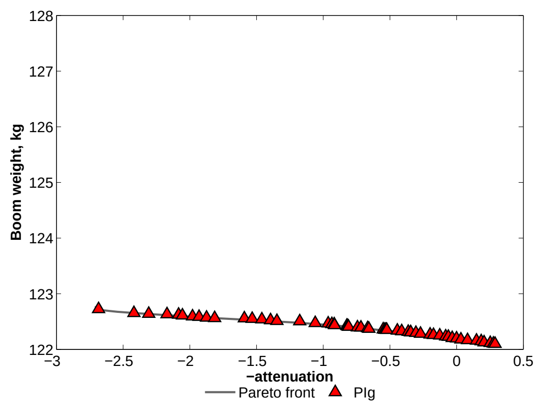


(c)

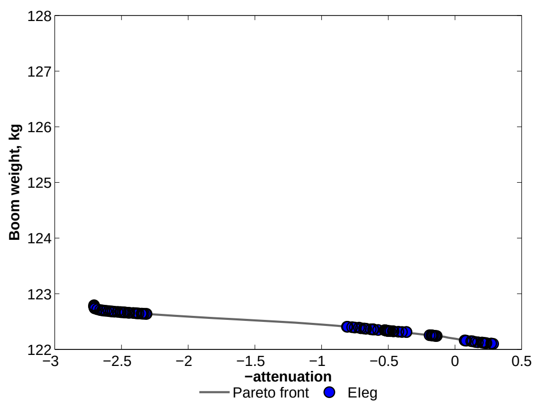


(d)

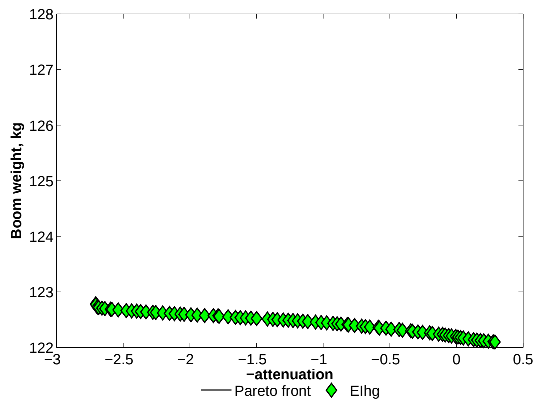
FIGURE A.11: Example Pareto sets for the satellite boom design problem after 30 update stages using the Kriging believer strategy. (a) Multiobjective probability of improvement, (b) Euclidean-based expected improvement, (c) hypervolume-based expected improvement and (d) maximin expected improvement.



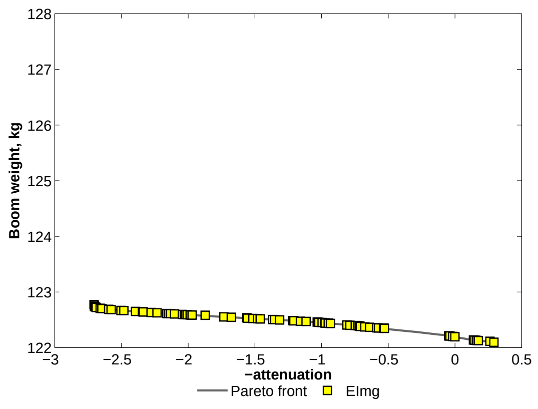
(a)



(b)



(c)



(d)

FIGURE A.12: Example Pareto sets for the satellite boom design problem after 30 update stages using (a) goal-based multiobjective probability of improvement, (b) Euclidean goal-based expected improvement, (c) goal-based expected improvement in hypervolume and (d) goal-based maximin expected improvement.

Appendix B

Analytical Expressions for Goal-Based Improvement Criteria

For a goal point next to an end point (y_1^1, y_2^1) the goal-based probability of improvement can be expressed as,

$$\begin{aligned} P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = &\boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) \\ &+ \boldsymbol{\Phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right)\left[1 - \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right], \end{aligned} \tag{B.1}$$

and for a goal point next to an end point (y_1^e, y_2^e),

$$\begin{aligned} P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = &\boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) \\ &+ \boldsymbol{\Phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right)\left[1 - \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right]. \end{aligned} \tag{B.2}$$

B.0.1 Euclidean-Based

The Euclidean goal-based expected improvement is given as,

$$E[I_E(\boldsymbol{x})]_G = P[y_1 \leq y_1^g \cup y_2 \leq y_2^g]\sqrt{(\bar{y}_1 - y_1^g)^2 + (\bar{y}_2 - y_2^g)^2}. \tag{B.3}$$

For a goal point next to an end point (y_1^1, y_2^1)

$$\begin{aligned}\bar{y}_1 P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = &\boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right] \\ &+ \left[1 - \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right]\left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right)\right]\end{aligned} \tag{B.4}$$

and

$$\begin{aligned}\bar{y}_2 P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = &\boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right] \\ &+ \boldsymbol{\Phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right)\left[\hat{y}_2 - \hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) + \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right].\end{aligned} \tag{B.5}$$

For a goal point next to an end point (y_1^e, y_2^e)

$$\begin{aligned}\bar{y}_1 P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = &\boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right] \\ &+ \boldsymbol{\Phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right)\left[\hat{y}_1 - \hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) + \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right]\end{aligned} \tag{B.6}$$

and

$$\begin{aligned}\bar{y}_2 P[y_1 \leq y_1^g \cup y_2 \leq y_2^g] = &\boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right] \\ &+ \left[1 - \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right]\left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right)\right].\end{aligned} \tag{B.7}$$

B.0.2 Hypervolume-Based

For a goal point next to an end point (y_1^1, y_2^1) the goal-based expected improvement in hypervolume can be expressed as,

$$
\begin{aligned}
E[I_H(\boldsymbol{x})]_G = {} & y_1^g \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) y_2^{max} \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) \\
& - y_1^g \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) \left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right] \\
& - y_2^{max} \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) \left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right] \\
& + \left\{\left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right] \times \left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right]\right\} \\
& + y_1^g \boldsymbol{\Phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right) y_2^{max} \left[\boldsymbol{\Phi}\left(\frac{y_2^{max} - \hat{y}_2}{\hat{s}_2}\right) - \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right] - y_1^g \boldsymbol{\Phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right) \\
& \times \left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^{max} - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^{max} - \hat{y}_2}{\hat{s}_2}\right) - \hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) + \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right] \\
& - y_2^{max} \left[\boldsymbol{\Phi}\left(\frac{y_2^{max} - \hat{y}_2}{\hat{s}_2}\right) - \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right] \left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right)\right] \\
& + \left\{\left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right)\right]\right. \\
& \left. \times \left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^{max} - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^{max} - \hat{y}_2}{\hat{s}_2}\right) - \hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) + \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right]\right\},
\end{aligned}
\tag{B.8}
$$

and for a goal point next to an end point (y_1^e, y_2^e),

$$
\begin{aligned}
E[I_H(\boldsymbol{x})]_G = {} & y_1^{max} \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) y_2^g \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) \\
& - y_1^{max} \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) \left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right] \\
& - y_2^g \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) \left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right] \\
& + \left\{\left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right] \times \left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^g - \hat{y}_2}{\hat{s}_2}\right)\right]\right\} \\
& - y_1^{max} \left[\boldsymbol{\Phi}\left(\frac{y_1^{max} - \hat{y}_1}{\hat{s}_1}\right) - \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right] \left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right)\right] \\
& + y_2^g \boldsymbol{\Phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right) y_1^{max} \left[\boldsymbol{\Phi}\left(\frac{y_1^{max} - \hat{y}_1}{\hat{s}_1}\right) - \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right] - y_2^g \boldsymbol{\Phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right) \\
& \times \left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^{max} - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^{max} - \hat{y}_1}{\hat{s}_1}\right) - \hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) + \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right] \\
& + \left\{\left[\hat{y}_2 \boldsymbol{\Phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right) - \hat{s}_2 \boldsymbol{\phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right)\right]\right. \\
& \left. \times \left[\hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^{max} - \hat{y}_1}{\hat{s}_1}\right) - \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^{max} - \hat{y}_1}{\hat{s}_1}\right) - \hat{y}_1 \boldsymbol{\Phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right) + \hat{s}_1 \boldsymbol{\phi}\left(\frac{y_1^g - \hat{y}_1}{\hat{s}_1}\right)\right]\right\}.
\end{aligned}
\tag{B.9}
$$

B.0.3 Maximin

Given the goal-based maximin expected improvement $E[I_M(\mathbf{x})]_G$, equation (7.18), the goal-based maximin expected improvement for a goal point next to an end point (y_1^1, y_2^1) can be expressed as,

$$E[I_M(\mathbf{x})]_G = E[I_M(\mathbf{x})]_G + \boldsymbol{\Phi}\left(\frac{\hat{y}_2 - y_2^g}{\hat{s}_2}\right)\left[(y_1^g - \hat{y}_1)\,\boldsymbol{\Phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right) + \hat{s}_1\boldsymbol{\phi}\left(\frac{y_1^1 - \hat{y}_1}{\hat{s}_1}\right)\right] \tag{B.10}$$

and for a goal point next to an end point (y_1^e, y_2^e),

$$E[I_M(\mathbf{x})]_G = E[I_M(\mathbf{x})]_G + \boldsymbol{\Phi}\left(\frac{\hat{y}_1 - y_1^g}{\hat{s}_1}\right)\left[(y_2^g - \hat{y}_2)\,\boldsymbol{\Phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right) + \hat{s}_2\boldsymbol{\phi}\left(\frac{y_2^e - \hat{y}_2}{\hat{s}_2}\right)\right]. \tag{B.11}$$

Appendix C

Aerodynamic Analysis

Using computational tools to analyse aerodynamic performance has become commonplace in engineering design. Fluid flow can usually be described using the Navier-Stokes equations but since fluid flow is highly nonlinear in nature, using full Navier-Stokes solvers leads to very large and unrealistic computational demands. A popular approach to reduce the complexity of the full Navier-Stokes is by using Reynolds averaged Navier-Stokes (RANS). This approach still allows for compressibility and viscosity effects, but predicts components of turbulent flow. Assuming the turbulent components are well modelled, using RANS gives a high fidelity but requires fine and often complex grids, resulting in a large computational expense and long run times.

It is possible to simplify the governing equations by treating the flow as inviscid. This assumption leads to the Euler equations. This removes the influence of boundary layers allowing much coarser grids to be used. Although this reduces computational demands considerably, removing the boundary layer often results in a lower fidelity prediction of the fluid flow. Assuming the effects of viscosity can be predicted separately, it is possible to generate viscous-coupled models that can be used to correct the Euler equations to include the effects of viscosity. These approaches tend to be less computationally demanding than RANS but still capable of providing a high level of fidelity.

Lower fidelity approaches can be formed by introducing further assumptions. If in addition to viscosity, rotation in the flow is also ignored, the flow can be modelled using the full nonlinear potential method. If also incompressible, this results in the linearized potential method. The correct choice of CFD solver depends on the application and must take into account the type of flow, the complexity of geometry, computational resources and time constraints. In many applications, providing high fidelity analysis with reasonable computational resources is still a challenge.

Here we consider the application of CFD to transonic wing design. Flight at transonic speeds incorporates local regions of both subsonic and supersonic flow. Since we are concerned with optimization it is necessary to evaluate hundreds of wing designs. Running a high fidelity solver such as RANS is simply intractable for this kind of study and the accuracy of flow prediction is sacrificed using a cheaper analysis. Here an inviscid full potential (FP) method is used with a viscous drag correction.

C.1 Overview of FP and Viscous Drag Correction

The flow solver used is based on the FP algorithm for three dimensional wings developed and made available by ESDU in 2002. This package includes grid generation, flow solution and post processing for isolated wings and wing-body combinations [ESDU, 2002]. This algorithm has been extended to include a viscous drag correction and used for optimization of transonic wing design by Toal [2009].

For a given wing geometry, FP first generates an appropriate mesh and then, by finite differencing, computes a solution to the full potential equation of inviscid compressible flow in three dimensions. The computed pressure coefficients can then be used to calculate aerodynamic coefficients for inviscid flow.

To incorporate viscous effects on the drag prediction, the method documented in ESDU [2008a,b] is used. This includes the computation of several components of viscous drag using a second flow solver VGK [ESDU, 1996].

Toal [2009] concluded that the FP method is a fast and reliable method for computing inviscid flow over simple wings, showing very good results on the ONERA-M6 test case. However, due to the limited number of validation test cases, results should be considered with care, especially for high speed flows and unusual geometries.

The viscous correction is much more limited, unable to handle complex geometries and only reliable on shock free attached flow. Nonetheless, the FP method coupled with VGK viscous prediction is an affordable approach of evaluating the performance of transonic wing designs and useful for the comparison of optimization techniques.

C.2 ONERA-M6 Test Case

Here we demonstrate the use of FP on the ONERA-M6 wing shown in Figure C.1. This wing is based on the ONERA-D symmetrical airfoil section with no dihedral and no twist. This is a classic test case for CFD validation at transonic speeds and high Reynolds numbers due to its simple geometry and availability of experimental test data.

The pressure distribution computed using FP at a Mach number 0.84 and an angle of attack of 2°is illustrated in Figure C.2. This shows that FP is capable of capturing the

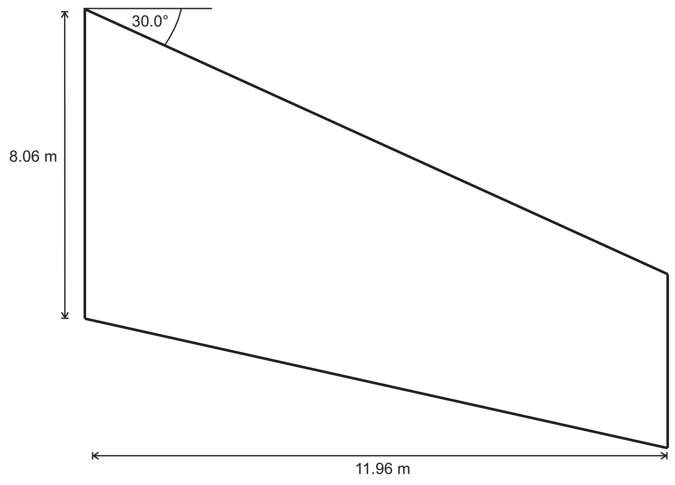


FIGURE C.1: ONERA-M6 wing planform.

main transonic flow features, including the ‘λ’ shock wave on the upper surface of the wing.

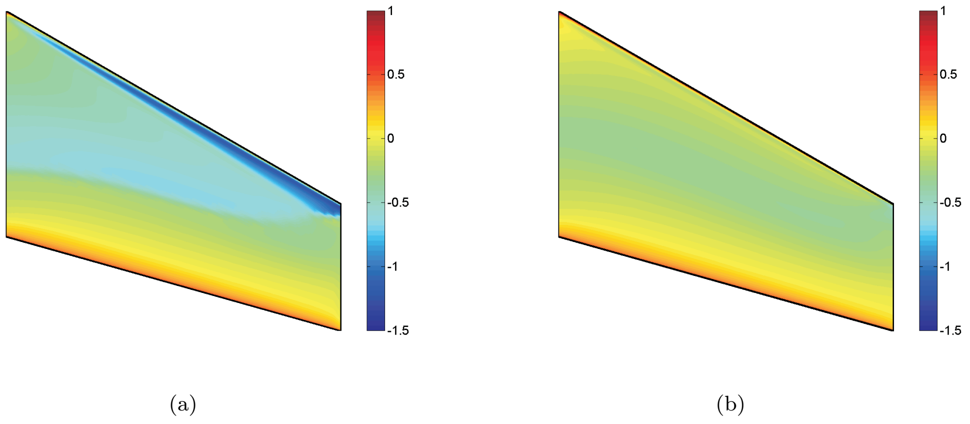
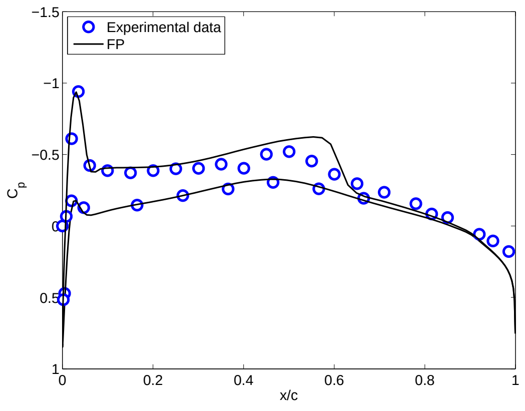
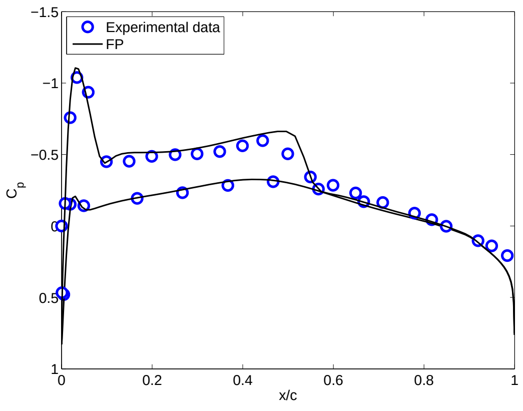


FIGURE C.2: ONERA-M6 pressure distribution computed using FP. (a) Upper surface. (b) Lower surface.

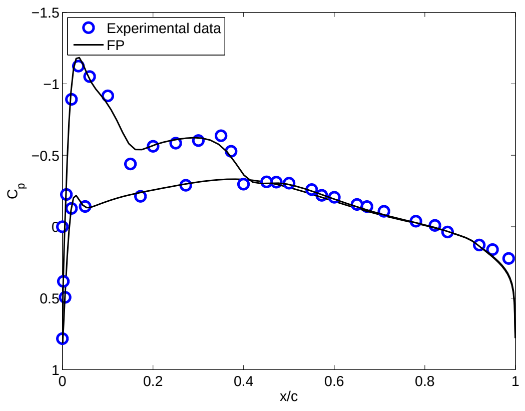
Figure C.3 compares the pressure distribution at different spanwise sections with the experimental test data found in Schmitt and Charpin [1976]. As pointed out in Toal [2009], it is clear FP is capable of capturing major flow features identified in experiments. The pressure distribution along the lower surface is accurately predicted along the length of the span whilst the pressure distribution on the upper surface is less accurate but does capture the location of leading edge suction and the location of shocks along the span.



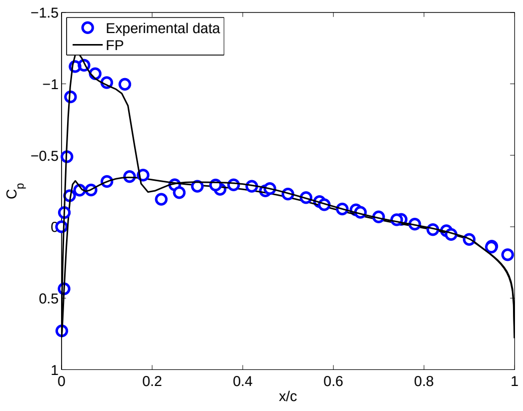
(a)



(b)



(c)



(d)

FIGURE C.3: Comparison of experimental data and computed pressure distribution using FP at spanwise positions (a) 0.2, (b) 0.44, (c) 0.8 and (d) 0.95.

Appendix D

Structural Analysis

The most common tool used in structural analysis is the finite element method (FEM). For simple linear elastic problems with well defined boundary conditions, finite elements are easy to use and can achieve exact solutions. As the problem becomes harder due to geometry or becomes nonlinear due to material properties, contact and friction, using finite elements becomes more complex. Nonetheless, with the correct set up, FEM can achieve very accurate solutions.

In its simplest form, FEM first discretizes the geometry into many elements with shared nodal points. The aim is to then predict the response of the structure at each node after some loads and boundary conditions have been applied. For a linear elastic stress/displacement problem, the deflection of the structure is linearly related to its stiffness and the displacement at each node can be predicted solving the equations of equilibrium using matrix methods. Once the displacements have been found, stress and strains can be easily computed.

In most real design problems, structures are loaded dynamically and materials tend to have nonlinear stiffness. In these problems it is necessary to solve systems of equations iteratively, becoming much more computationally expensive. In all cases, the accuracy of the results depends of the element type, shape and size. As the mesh density increases, the results usually converge towards better solutions but the time to compute the solution increases.

In this study it is necessary to analyse hundreds of wing designs and using a high fidelity analysis based on nonlinear equations under dynamic load cases is simply intractable. To ensure optimization is a feasible task, all structures are assumed to be linear elastic and buckling loads are computed using an eigenvalue analysis.

D.1 Overview of Abaqus

Abaqus/Standard is used to analyse the wingbox structure studied in Chapter 5 and Chapter 8. Based on FEM, Abaqus/Standard provides a library of different elements that can be used to solve a range of linear and nonlinear structural problems under different static and dynamic loading conditions. Abaqus/CAE provides functionality for model generation, submission of jobs and analysis of results. This functionality can be accessed using a graphical interface or alternatively using Python scripts.

Two analysis procedures are used. The first uses a linear static stress analysis to find the displacements and stresses through the structure. This procedure solves the linear equilibrium equations and requires specification of initial conditions, boundary conditions, material options, elements and loads. For the wingbox structure, initial conditions, boundary conditions and material options are known and the loads are determined by the FP flow solver. The effect of the element type and size is investigated on a simple cantilever beam in the next section.

The second procedure uses an eigenvalue buckling prediction to find critical loads. Abaqus looks for the loads which cause the structure stiffness matrix to become zero. This procedure requires specification of a base state, boundary conditions, material options, loads, elements and an eigenvalue extraction method. As in the stress analysis, the base state, boundary conditions material and loads are known. In all cases the eigenvalue extraction uses the Lanczos method to find the first eigenvalue λ_1. This eigenvalue indicates the critical buckling load as a factor of the applied load, therefore a $\lambda_1 \leq 1$ indicates buckling at the applied load. The effect the choice of element type and size has on the buckling prediction is investigated in the next section.

D.2 Element Comparison

The selection of the correct element type has a large impact on the quality of results obtained using FEM. It is important to consider the element type but also properties of the mesh. Throughout this study, unstructured meshing is used. In many cases a structured mesh will provide better results but is much more difficult to generate automatically, especially on complex geometry. Using free meshing in Abaqus reliably generates meshes on all types of geometry being more suitable for automated optimization.

Here three types of conventional homogeneous shell elements are tested. This includes a general purpose four node shell element S4 suitable for both thin and thick shell elements. A reduced integration four node shell element S4R and a quadratic eight node shell element S8R suitable for thick shell elements. All these shell elements assume the thickness of the structure is significantly smaller than the other dimensions. This is considered to be the case for all components in the wingbox structure modelled, a common assumption in conceptual and preliminary wingbox design [Giles, 1986].

D.2.1 Linear Static Stress Analysis

The performance of different elements and mesh sizes using the linear static stress analysis is compared using a simple cantilever design problem, Figure D.1. This considers a beam of length $l = 20$ m, depth $d = 1$ m and thickness $t = 0.01$ m with a tip load $F = 10$ kN, made from Aluminium with Young's modulus $E = 71.7$ GPa.

The tip displacement computed by Abaqus is compared with that given by simple beam theory as $\delta = 0.4463$ m. Given the Airy stress function $\phi = Ay^3 + By^3x + Cyx$, satisfying equilibrium and compatibility equations, the stress in a cantilever beam with tip load is given analytically as

$$\sigma_x = \frac{Fy}{I}(l - x)\,. \tag{D.1}$$

This assumes that the constrained end of the beam is free to distort which is not true in practice. Nonetheless, by virtue of St. Venant's principle this solution can be considered exact away from the constrained end. A comparison of stress is made at mid span and on the top face ($x = 10, y = 0.5$). Following the above formulae the stress at this point is given as $\sigma_x = 60$ MPa.

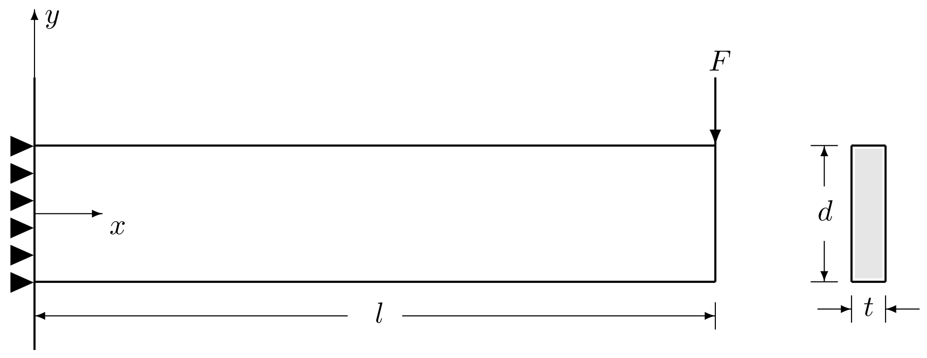


FIGURE D.1: Beam.

Table D.1 demonstrates the error in the displacement and stress prediction when using different element types and mesh seed sizes. Due to additional nodes in the quadratic S8R element, this provides the most accurate prediction, even at larger seed sizes. The less computationally demanding S4 and S4R elements provide a reasonable accuracy when using a finer mesh (smaller seed size). The S4 element also performs well with a seed size of 0.5, resulting in a relatively coarse mesh. This is due to the free mesh tool generating a well structured mesh. However, this is not guaranteed to be the case in practise when the geometry in more complex and the mesh is unlikely to be so well structured.

TABLE D.1: Error in tip displacement and stress prediction.

	S4		S4R		S8R	
Seed size	Displacement error (%)	Stress error (%)	Displacement error (%)	Stress error (%)	Displacement error (%)	Stress error (%)
0.5	0.32	0.38	33.31	32.84	0.03	0.00
0.4	2.97	8.38	21.13	15.28	0.02	0.02
0.3	2.06	4.22	6.76	20.04	0.03	0.07
0.2	0.34	0.48	2.95	14.19	0.06	0.00
0.1	0.07	1.14	0.68	8.47	0.08	0.00

D.2.2 Eigenvalue Buckling Prediction

The behaviour of the eigenvalue buckling prediction is investigated using a simply supported square plate with sides $b = 2$ m, thickness $t = 0.01$ m, Young's modulus $E = 71.7$ GPa and a Poisson ratio $\nu = 0.33$. With a compressive load is applied to opposite ends of the plate, the analytical solution for the critical buckling load is given as

$$N_{CR} = \frac{4\pi^2 E t^3}{12 b^2 \left(1 - \nu^2\right)}. \tag{D.2}$$

Given a uniform load of $N = 2.5$ kNm^{-1} this is gives an expected eigenvalue $\lambda_1 = 6.6178$.

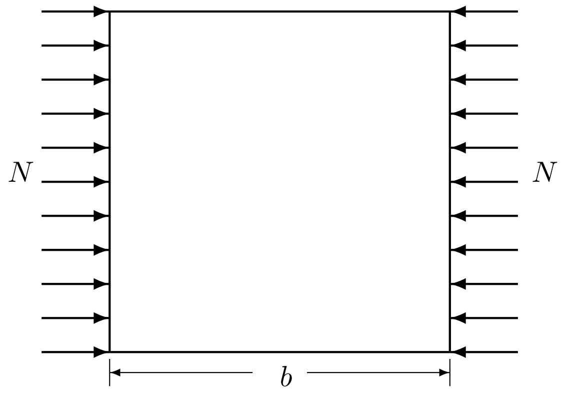


FIGURE D.2: Plate.

Table D.2 shows the influence of the element type and mesh seed size on the eigenvalue buckling prediction. Again the S8R element provides an accurate buckling prediction for all the mesh seed sizes tested. Although less accurate, both the S4 and S4R elements are capable of predicting λ_1 within a reasonable accuracy and becomes more accurate when using a finer mesh.

TABLE D.2: Error in buckling prediction.

	S4	S4R	S8R
Seed size	Eigenvalue error (%)	Eigenvalue error (%)	Eigenvalue error (%)
0.5	2.18	2.68	0.07
0.4	1.39	1.70	0.01
0.3	0.70	0.86	0.00
0.2	0.35	0.42	0.03
0.1	0.08	0.09	0.03

Appendix E

Wing Design Example

In both Chapter 5 and Chapter 8 we optimize the design of a wingbox structure. In Chapter 5 the structural weight is minimized for a fixed planform whilst in Chapter 8 we allow the planform to vary and optimize for both minimum drag and minimum wingbox weight.

Each wing design is based on a 150 seat passenger aircraft flying at a cruise speed of Mach 0.8 at 11,000 m. The wing is based on the ONERA-D airfoil section and is required to produce enough lift for a maximum cruise weight of 60,000 kg. The following sections contain further details on the drag calculation, the wingbox design and the application of aerodynamic loads.

E.1 Drag Calculation

To achieve the required C_L, each planform design is evaluated up to four times using the FP design tool. The first two simulations are used to compute the wings lift curve slope. A third simulation is then run at the angle of incidence that is predicted to achieve the required C_L. If the resulting C_L from the third simulation is within 0.0025 of the required C_L, this is taken to be the correct angle of incidence for that particular wing planform. If required, a fourth simulation is run to find a new C_L closer to what is required. The final simulation is followed by the VGK based drag correction to compute the contribution of drag due to viscous effects. The final drag is taken as a combination of the wave and vortex drag computed using FP and the viscous drag computed using VGK. This process provides the wing drag and pressure distribution for the upper and lower wing surfaces, taking up to 30 minutes to run on a single processor.

The wing planform is based on four design variables consisting of the root chord, span, taper ratio and quarter chord sweep angle.

E.2 Wingbox Design

The design of the wingbox has been simplified in a number of ways, resulting in a model that deviates from an actual wing structure but provides a model that can be easily parameterized and gives representative results in a convenient time frame. The wingbox is simplified assuming the following:

1. Wing control surfaces (flaps, slats, ailerons etc) are not considered and the wing is taken as a single structure.

2. Any minor structural details are ignored. This may include fillets, rivets, fasteners and small elements of reinforcement.

3. The wing skin, spars and ribs are all considered as solid thin shells with constant thickness.

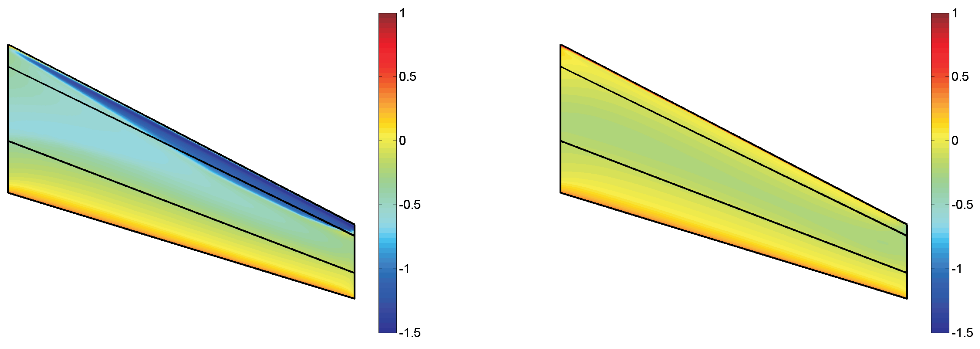
4. Aluminium alloy 7050-T7451 is used throughout the structure and assumed to be homogenous.

A wing model based on these simplifications is expected to be different from an actual wing structure but for the overall aims of this research these simplification are sufficient.

The front and rear spar are positioned at 15% and 65% chord respectively and the wingbox structure is defined by a further four design variables, the spar thickness, rib thickness, skin thickness and the rib pitch (the spacing between each rib).

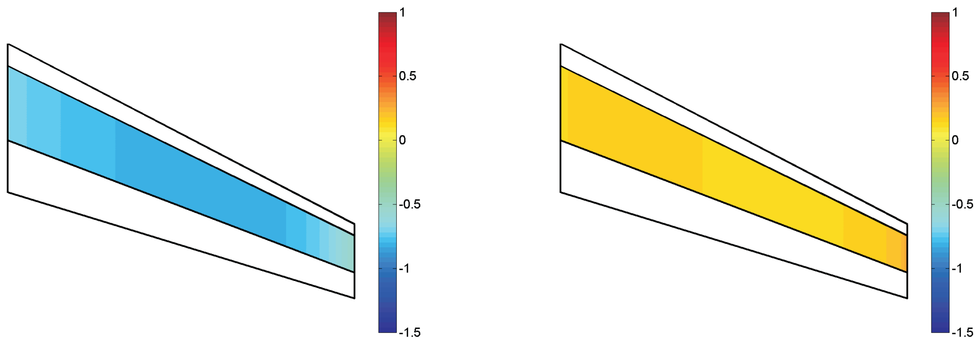
E.3 Wing Loading

For each wing planform, aerodynamic loads are computed and applied to the wingbox structure. FP provides a distribution of pressure loads over the entire planform of the wing. To transfer these loads to the wingbox model, the chordwise pressure distribution is integrated at each spanwise position. This spanwise distribution is then applied to the wingbox assuming a uniform pressure along the chord. Note that by assuming a uniform pressure, we simplify the loading distribution and reduce the influence of any chordwise bending moment. This is a consequence of only modelling the structure of the wingbox section rather than the entire wing and likely to be a limited approximation for detailed analysis. Although the approach taken here is unlikely to capture all the applied loads accurately, we consider this simplification reasonable for the overall aims of this study. See Hürlimann [2010] for a more sophisticated approach, applying aerodynamic loads as nodal forces. Figures E.1 and E.2 demonstrate the transformation of loading between the entire wing section to just that of the wingbox. The loading profile is modelled using a high order polynomial and applied in Abaqus expressed as an equation defined pressure field.



(a) (b)

FIGURE E.1: Pressure distribution over wing planform computed using FP. (a) Upper surface. (b) Lower surface.



(a) (b)

FIGURE E.2: Integrated pressure distribution over wingbox section. (a) Upper surface. (b) Lower surface.

E.4 Wingbox Constraints

In conceptual design, it is a common approach to treat an aircraft wing as a simple cantilever beam. In practise, the stresses at the root of an aircraft wing are not as severe as predicted by a cantilever due to the deformation of the fuselage. To accommodate these effects, an encastre boundary condition is placed on a section away from the main wingbox structure. This avoids an unrealistic concentration of forces at the wing root. The main wingbox section and additional encastre section is pictured in Figure E.3.

To ensure the wingbox design is feasible, design constraints exist on structural stresses, deformation and buckling. A full analysis would consider stress constraints on every element of the model, displacement constraints on every node and a buckling prediction for several wing sections. To keep this problem manageable and suitable for optimization, only the maximum stress at the root of the wingbox and the maximum displacement at the wing tip are constrained. The buckling prediction is performed over the whole

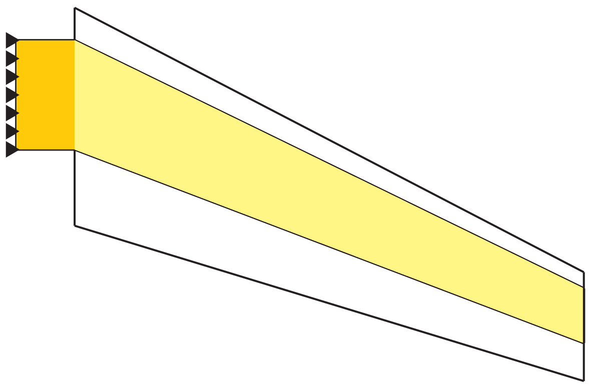


FIGURE E.3: Constrained wingbox section. 🟧–Constrained box section. 🟨–Main wingbox section.

wing as a single section. Although this may not accurately represent the location of maximum stress and maximum deformation on every design iteration, this is considered to be representative and suitable for the aims of this study.

Table E.1 shows all design variables, limits, objective and constraint values. In this study the maximum allowable stress is represented in terms of a safety factor equivalent to the material yield stress $\sigma_y = 469$ MPa divided by the maximum Von-Mises stress at the wingbox root. The optimized structure for this particular planform are those documented in Chapter 5, Table 5.2.

TABLE E.1: Baseline design variables, limits, constraint values, and objective value.

Lower limit	Baseline value	Upper limit	Type	Quantity
6	7.5	9	Variable	Root chord, m
15	17.5	20	Variable	Span, m
0.2	0.5	0.8	Variable	Taper ratio
10	25	40	Variable	Quarter chord sweep angle, deg
0.010	0.034	0.050	Variable	Spar thickness, m
0.010	0.050	0.050	Variable	Rib thickness, m
0.010	0.030	0.030	Variable	Skin thickness, m
0.020	0.034	0.200	Variable	Rib pitch, m
1.50	1.67	—	Constraint	Stress safety factor
—	1.48	1.50	Constraint	Tip displacement, m
1.00	1.71	—	Constraint	Buckling eigenvalue
—	16458	—	Objective	Wingbox weight, kg
—	0.020	—	Objective	Wing drag coefficient

E.5 Wingbox Mesh

To determine the most suitable element type and mesh seed size for the structural analysis of the wingbox design, a mesh dependency study is performed, see Table E.2, E.3 and E.4. Here we compare the stress, displacement and buckling prediction based on the baseline wingbox design with design variables listed in Table E.1. Since we aim to optimize the structural weight of the wingbox design, the runtime of each analysis is an important consideration.

Based on this baseline design, the S8R element is the most accurate but suffers from longer run times when compared to the other elements. The S8R element is also much more unreliable and fails to find solutions when using finer meshes due to distorted elements, E.4. Using S4 elements provides more consistent solutions when compared to the S4R element but has a longer run time when using finer meshes. The difference in this run time is small in this baseline design example but becomes more significant on wings with a larger chord, span and smaller rib pitch. In favour of reducing the overall runtime when evaluating many wings, the less computational expensive S4R elements are used in this study. We select a seed size of 0.25 as a compromise between runtime and accuracy but also since this seed size ensures a minimum of two elements along the smallest dimension of the wingbox structure.

Figure E.4 demonstrates the mesh based on this seed size, stress and displacement prediction for the baseline design variables in Table E.1.

Table E.2: S4 element.

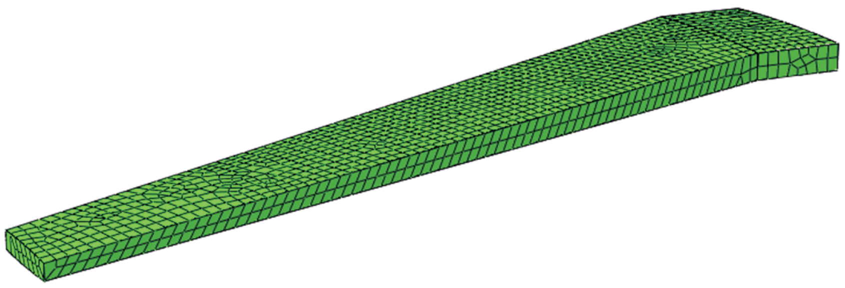
Seed size	Node count	Displacement (m)	Safety Factor	Eigenvalue	Run time (sec)
0.5	1342	1.82	1.48	0.55	30.7
0.4	1926	1.81	1.46	3.59	34.9
0.3	3213	1.73	1.47	1.32	41.7
0.2	6488	1.69	1.47	1.82	59.9
0.1	23840	1.63	1.46	1.61	198.4

Table E.3: S4R element.

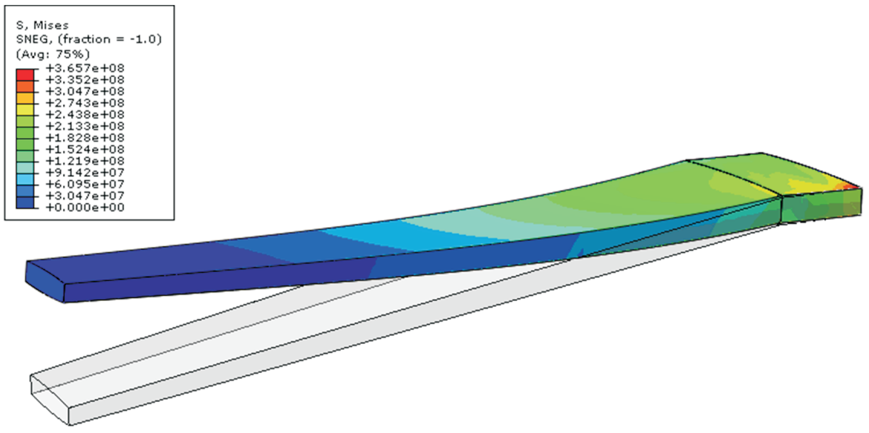
Seed size	Node count	Displacement (m)	Safety Factor	Eigenvalue	Run time (sec)
0.5	1342	1.76	1.56	0.52	28.4
0.4	1926	1.67	1.52	3.31	35.0
0.3	3213	1.72	1.49	1.29	39.3
0.2	6488	1.69	1.48	1.81	51.3
0.1	23840	1.63	1.47	1.60	126.8

Table E.4: S8R element (blanks indicate failed analysis due to distortion).

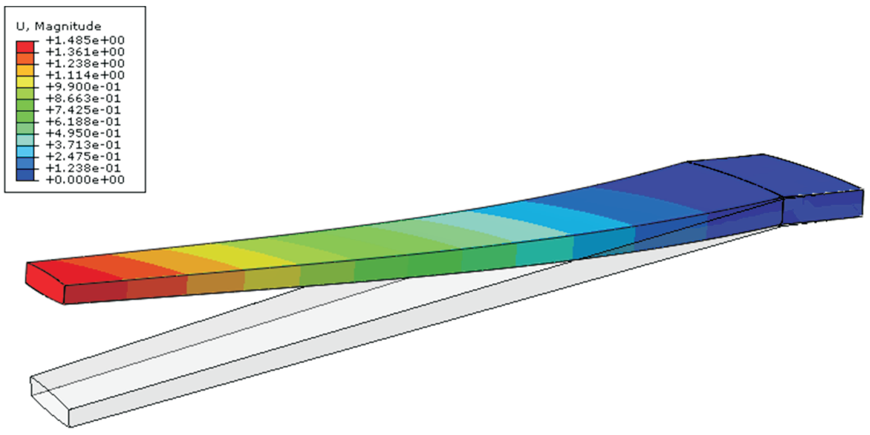
Seed size	Node count	Displacement (m)	Safety Factor	Eigenvalue	Run time (sec)
0.5	3491	1.72	1.46	1.97	38.6
0.4	5151	1.70	1.46	1.84	50.9
0.3	3328	–	–	–	–
0.2	18307	–	–	–	–
0.1	69352	–	–	–	–



(a)



(b)



(c)

FIGURE E.4: Structural analysis on baseline wing design. (a) Mesh with seed size 0.25. (b) Deflection and Von-Mises stress prediction. (c) Deflection and displacement prediction.

Bibliography

N. Alexandrov, J. E. Denis, R. M. Lewis, and V. Torczon. A trust region framework for managing the use of approximation models in optimization. *Structural and Multidisciplinary Optimization*, 15(1):16–23, 1998.

M.R. Anderberg. *Cluster Analysis for Applications.* Academic Press, 1973.

A. Angantyr, J. Andersson, and J. O. Aidanpaa. Constrained optimization based on a multiobjective evolutionary algorithm. In *Proceedings of the Congress on Evolutionary Computation*, volume 3, Piscataway, New Jersey, Canberra, Australia,, 2003.

C. Audet and J. E. Dennis. A pattern search filter method for nonlinear programming without derivatives. *SIAM Journal on Optimization*, 14:980–1010, 2004.

C. Audet, J. E. Dennis, D. W. Moore, and P. D. Frank A. Booker. A surrogate-model-based method for constrained optimization. In *8th Proceedings of the AIAA/-NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 2000.

D. C. Bautista. *A Sequential Design for Approximating the Pareto Front using the Expected Pareto Improvement Function.* PhD thesis, Ohio State University, 2009.

B. J. Bichon, M. S. Eldred, L. P. Swiler, S. Mahadevan, and J. M. McFarland. Efficient global reliability analysis for nonlinear implicit performance functions. *AIAA Journal*, 46(10):2459–2468, 2008.

P. T. Boggs and J. W. Tolle. Sequential quadratic programming algorithm. *Acta Numerica*, 4:1–51, 1995.

A. J. Booker, J.E. Dennis, P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. *Structural Optimization*, 17(1):1–13, 1999.

R. Brekelmans, L. Driessen, H. Hamers, and D. den Hertog. Constrained optimization involving expensive function evaluations: A sequential approach. *European Journal of Operational Research*, 160:121–138, 2005.

A. D. Bull. Convergence rates of efficient global optimization algorithm. *Journal of Machine Learning Research*, 12:2879–2904, 2011.

R. Carrese, A. Sóbester, H. Winarto, and X. Li. Swarm heuristic for identifying preferred solutions in surrogate-based multi-objective engineering design. *AIAA Journal*, 49(7): 1473–1449, 2011.

R. Carrese, H. Winarto, X. Li, A. Sóbester, and S. Ebenezer. A comprehensive preference-based optimization framework with application to high-lift aerodynamic design. *Engineering Optimization*, 44(10):1209–1227, 2012.

R. Chelouah and P. Siarry. Genetic and nelder-mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions. *European Journal of Operational Research*, 148:335–348, 2003.

M. R. Chernick. *Bootstrap Methods - A Guide for Practitioners and Researchers.* John Wiley & Sons, 2008.

C. A. Coello Coello. Treating constraints as objectives for single-objective evolutionary optimization. *Engineering Optimization*, 32:275–308, 2000a.

C. A. Coello Coello. Constraint-handling using an evolutionary multiobjective optimization technique. *Civiil Engineering and Environmental Systems*, 17:319–346, 2000b.

C. A. Coello Coello. Theoretical and numerical constraint handling techniques used in evolutionary algorithms: A survey of state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191:1245–1287, 2002.

C. A. Coello Coello. Twenty years of evolutionary multiobjective optimization: A historical view of the field. *IEEE Computational Intelligence Magazine*, 1(1):28–36, 2006.

C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems.* Genetic and Evolutionary Computation Series. Springer, second edition, 2007.

D. W. Corne, K. Deb, P. J. Fleming, and J. D. Knowles. The good of the many outweighs the good of the one: Evolutionary multi-objective optimization. *coNNectionS*, 1:9–13, 2003.

J. Cousin and M. Metcalfe. The british aerospace ltd. transport aircraft synthesis and optimization program. In *Systems and Operations Conference*, Dayton OH, September 1990. In AHS and ASEE.

K. Deb. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation*, 7(3):205–230, 1999.

K. Deb. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186:311–338, 2000.

K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms.* Wiley-Interscience Series in Systems and Optimization. John Wiley & Sons, first edition, 2001.

K. Deb and S. Gupta. Understanding knee points in bicriteria problems and their implications as preferred solution principles. *Engineering Optimization*, 43(11):1175–1204, 2011.

K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

M. Emmerich, A. H. Deutz, and J. W. Klinkenberg. Hypervolume-based expected improvement: Monotonicity properties and exact computation. In *IEEE Congress on Evolutionary Computation*, pages 2147–2154, New Orleans, Louisiana, 5-8 June 2011 2011.

M. T. M. Emmerich. *Single and Multiobjective Evolutionary Design Optimization Assisted by Gaussian Random Field Metamodels.* PhD thesis, University of Dortmund, 2005.

F. Erbatur, O. Hasançebi, İ. Tütüncü, and H. Kiliç. Optimal design of planar and space structures with genetic algorithms. *Computers and Structures*, 75:209–224, 2000.

ESDU. *VGK Method for Two-Dimensional Airfoil Sections.* ESDU-96028, London, 1996.

ESDU. *Full-Potential (FP) Method for Three-Dimensional Wings and Wing-Body Combinations - Inviscid Flow Part 1: Principles and Results.* ESDU-02013, London, 2002.

ESDU. *Wing Viscous Drag Coefficient in Shock-Free Attached Flow.* ESDU-07002, London, 2008a.

ESDU. *Modelling of Wing Viscous Drag Coefficients in Shock-Free Attached Flow.* ESDU-07003, London, 2008b.

R. Fletcher and S. Leyffer. Nonlinear programming without a penalty function. *Mathematical Programming*, 91:239–269, 2002.

C. M. Fonseca and P. J. Fleming. Genetic algorithm for multiobjective optimization: Formulation, discussion and generalization. In *Genetic Algorithms: Proceedings of the Fifth International Conference*, San Mateo, July 1993.

C. M. Fonseca and P. J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 3:1–16, 1995.

A. I. J. Forrester and A. J. Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45:50–79, 2009.

A. I. J. Forrester, A. J. Keane, and N. W. Bressloff. Design and analysis of "noisy" computer experiments. *AIAA Journal*, 44:2331 – 2339, 2006.

A. I. J. Forrester, A. Sóbester, and A. J. Keane. *Engineering Design via Surrogate Modelling*. John Wiley and Sons, 2008.

A. Ghosh and S. Dehuri. Evolutionary algorithms for multi-criterion optimization: A survey. *International Journal of Computing and Information Sciences*, 2(1):38–57, 2004.

G. L. Giles. Equivalent plate analysis of aircraft wing box structures with general planform geometry. *Journal of Aircraft*, 23(11):859–864, 1986.

D. Ginsbourger, R. Le Riche, and L. Carraro. *Computational Intelligence in Expensive Optimization Problems*, chapter Kriging is Well-Suited to Parallelize Optimization, pages 131–162. Springer, 2010.

A. A. Giunta. *Aircraft Multidisciplinary Design Optimization using Design of Experiments Theory and Response Surface Modeling Methods*. PhD thesis, Virginia Polytechnic and State University, 1997.

P. K. Gudla and R. Ganguli. An automated hybrid genetic-conjugate gradient algorithm for multimodal optimization problems. *Applied Mathematics and Computation*, 167: 1457–1474, 2005.

N. Henkenjohann and J. Kunert. An efficient sequential optimization approach based on the multivariate expected improvement criterion. *Quality of Engineering*, 19:267–280, 2007.

J. H. Holland. Outline for a logical theory of adaptive systems. *Journal of the ACM*, 3: 297–314, 1962.

R. Hooke and T. A. Jeeves. Direct search solution of numerical and statistical problems. Scientific paper, Westinghouse Research Labs, 1960.

F. Hürlimann. *Mass Estimation of Transport Aircraft Wingbox Structures with CAD/CAE-Based Multidisciplinary Process*. PhD thesis, ETH Zurich, 2010.

A. Jameson. Efficient aerodynamic shape optimization. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Albany, New York, 2004. AIAA-04-4369.

M. J. Jensen. *Guiding Single-Objective Optimization using Multi-Objective Methods*, volume 2611, chapter Applications of Evolutionary Computing, pages 199–210. Springer, 2003.

S. Jeong and S. Obayashi. Efficient global optimization (ego) for multiobjective problem and data mining. In *IEEE Congress Evolutionary Computing*, volume 3, pages 2138–2145, New York, 2005.

Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, 9:3–12, 2005.

Y. Jin, M. Olhofer, and B. Sendhoff. A framework for evolutionary optimization with approximate fitness functions. *IEEE Transactions on Evolutionary Computation*, 6 (5):481–494, 2002.

D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21:345–383, 2001.

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

A. J. Keane. Experiences with optimizers in structural design. In *Adaptive Computing in Engineering Design and Control*, Plymouth, September 1994.

A. J. Keane. Genetic algorithm optimization of multi-peak problems: Studies in convergence and robustness. *Artificial Intelligence in Engineering*, 9:75–83, 1995.

A. J. Keane. Wing optimization using design of experiment, response surface, and data fusion methods. *Journal of Aircraft*, 40(4):741–750, 2002.

A. J. Keane. Statistical improvement criteria for use in multiobjective design optimization. *AIAA Journal*, 44(4):879–891, April 2006.

A. J. Keane and A. P. Bright. Passive vibration control via unusual geometries: Experiments on model aerospace structures. *Journal of Sound and Vibration*, 190(4): 713–719, 1996.

A. J. Keane and P. B. Nair. *Computational Approaches for Aerospace Design.* John Wiley and Sons, 2005.

A. J. Keane and J. P. Scanlan. Design search and optimization in aerospace engineering. *Philosophical Transactions of the Royal Society A*, 365:2501–2529, 2007.

V. Khare, X. Yao, and K. Deb. Performance scaling of multi-objective evolutionary algorithms. In *2nd International Conference on Evolutionary Multi-Criterion Optimization*, pages 376–390, Faro, 8-11 April 2003 2003. Springer-Verlag.

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

J. Knowles and H. Nakayama. Meta-modeling in multiobjective optimization. In *Multiobjective Optimization: Interactive and Evolutionary Approaches*, page 245. Springer-Verlag, 2008.

J. D. Knowles, R. A. Watson, and D. W. Corne. Reducing local optima in single-objective problems by multi-objectivization. In *Proceedings of the Evolutionary Multi-Criterion Optimization Conference*, pages 269–283. Springer-Verlag, 2001.

D. G. Krige. A statistical approach to some basic mine valuation problems on the witwatersrand. *Journal of the Chemical, Metallurgical and Mining Engineering Society of South Africa*, 52:119–139, 1951.

H. J. Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86:57–76, 1964.

X. B. Lam, Y. S. Kim, A. D. Hoang, and C. W. Park. Coupled aerostructural design optimization using the kriging model and integrated multiobjective optimization algorithm. *Journal of Optimization Theory and Applications*, 142:533–556, 2009.

J. I. Loeppky, L. M. Moore, and B. J. Williams. Batch sequential designs for computer experiments. *Journal of Statistical Planning and Inference*, 140(6):1452–1464, 2010.

B. Naujoks M. Emmerich, K. Giannakoglou. Single and multi-objective evolutionary optimization assisted by gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10:421–439, 2006.

M. D. Mackay, 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*, 21:239–245, 1979.

R. Mallipeddi and P. N. Suganthan. Ensemble of constraint handling techniques. *IEEE Transactions on Evolutionary Computation*, 2010.

J. D. Martin and T. W. Simpson. Use of adaptive metamodeling for design optimization. In *9^{th} AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Atlanta, Georgia, September 2002.

J. R. R. A. Martins, J. J. Alonso, and J. J. Reuther. High-fidelity aerostructural design optimization of a supersonic business jet. *Journal of Aircraft*, 41:523, 2004.

E. Mezura-Montes and C. A. Coello Coello. *Multiobjective Problem Solving from Nature*, chapter Constrained Optimization via Multiobjective Evolutionary Algorithms, pages 53–75. Springer-Verlag, 2008.

M. D. Morris and T. J. Mitchell. Exploratory designs for computational experiments. *Journal of Statistical Planning and Inference*, 43:381–402, 1995.

J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

F. Neumann and I. Wegener. *Multiobjective Problem Solving from Nature*, chapter Can Single-Objective Optimization Profit from Multiobjective Optimization?, pages 115–130. Springer-Verlag, 2008.

J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operational Research. Spinger, second edition edition, 2006.

H. Nowacki. *Computer Science Computer Aided Design Modelling, Systems Engineering, CAD-Systems*, chapter Modelling of Design Decisions for CAD, pages 177–223. Springer-Verlag, 1980.

Y. S. Ong, P. B. Nair, A. J. Keane, and K. W. Wong. *Knowledge Incorporation in Evolutionary Computation*, chapter Surrogate-Assisted Evolutionary Optimization Frameworks for High-Fidelity Engineering Design Problems, pages 307–331. Studies in Fuzziness and Soft Computing. Springer, 2004.

A. Oyama, K. Shimoyama, and K. Fujii. New constraint-handling method for multi-objective multi-constraint evolutionary optimization and its application to space plane design. In *Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems*, Munich, 2005.

J. M. Parr, C. M. E. Holden, A. I. J. Forrester, and A. J. Keane. Review of efficient surrogate infill sampling criteria with constraint handling. In *2nd International Conference on Engineering Optimization*, Lisbon, September 2010.

J. M. Parr, A. I. J. Forrester, A. J. Keane, and C. M. E. Holden. Handling constraints in surrogate-based optimization. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, Honolulu, Hawaii, April 2012a.

J. M. Parr, A. I. J. Forrester, A. J. Keane, and C. M. E. Holden. Enhancing infill sampling criteria for surrogate-based constrained optimization. *Journal of Computational Methods in Science and Engineering*, 12:25–45, 2012b.

J. M. Parr, A. J. Keane, A. I. J. Forrester, and C. M. E. Holden. Infill sampling criteria for surrogate -based optimization with constraint handling. *Engineering Optimization*, 44(10):1147–1166, 2012c.

V. Picheny, D. Ginsbourger, O. Roustant, R. T. Haftka, and N. Kim. Adaptive designs of experiments for accurate approximation of a target region. *Journal of Mechanical Design*, 132, 2010.

W. Ponweiser, T. Wagner, D. Biermann, and M. Vincze. Multiobjective optimization on a limited budget of evaluations using model-assisted s-metric selection. In G. Rudolph, T. Jensen, S. Lucas, C. Poloni, and N. Beume, editors, *Parallel Problem Solving from*

Nature - PPSN X, volume 5199 of *Lecture Notes in Computer Science*, pages 784–794, Dortmand, Germany, September 2008a. Springer-Verlag.

W. Ponweiser, T. Wagner, and M. Vincze. Clustered multiple generalized expected improvement: A novel infill sampling criterion for surrogate models. In Z. Michalewicz, editor, *IEEE Congress on Evolutionary Computation*, pages 3541–3521, 2008b.

K. Praditwong and X. Yao. How well do multi-objective evolutionary algorithms scale to large problems. In *IEEE Congress on Evolutionary Computation*, pages 3959–3966, Singapore, 25-28 September 2007 2007.

P. Ranjan and D. Bingham. Sequential experiment design for contour estimation from complex computer codes. *Technometrics*, 50:527–541, 2008.

S. S. Rao. *Engineering Optimization: Theory and Practice*. Wiley-Interscience, third edition, 1996.

A. Ratle. Accelerating the convergence of evolutionary algorithms by fitness landscape approximation. In A. E. Eiben, T. Bäck, M. Schoenauer, and H-P. Schwefel, editors, *Parallel Problem Solving from Nature - PPSN V*, volume 1498 of *Lecture Notes in Computer Science*, pages 87–96, Amsterdam, The Netherlands, September 1998. Springer-Verlag.

T. Ray, T. Kang, and S. K. Chye. An evolutionary algorithm for constrained optimization. In *Proceedings of the Genetic and Evolutionary Computing Conference*, Las Vegas, 2000.

T. Ray, A. Isaacs, and W. Smith. Surrogate assisted evolutionary algorithm for multiobjective optimization. In *Multiobjective Optimization: Techniques and Applications in Chemical Engineering*, volume 1, chapter 5, pages 131–151. World Scientific, 2009.

R. G. Regis. Stochastic radial basis function algorithms for large-scale optimization involving expensive black-box objective and constraint functions. *Computers and Operations Research*, 38:837–853, 2011.

R. G. Regis and C. A. Shoemaker. Constrained global optimization of expensive black box functions using radial basis functions. *Journal of Global Optimization*, 31:151–171, 2005.

J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4:409 – 435, 1989.

S. Sakata, F. Ashida, and M. Zako. Structural optimization using kriging approximation. *Computer Methods in Applied Mechanics and Engineering*, 192:923–939, 2003.

L. V. Santana-Quintero, A. A. Monta no, and C. A. Coello Coello. *Computational Intelligence in Expensive Optimization*, chapter A Review of Techniques for Handling Expensive Functions in Evolutionary Multi-Objective Optimization, pages 29–59. Springer, 2010.

T. J. Santner, B. J. Williams, and W. I. Notz. *The Design and Analysis of Computer Experiments*. Springer Series in Statistics. Spinger, 2003.

M. J. Sasena. *Flexibility and Efficiency Enhancements for Constrained Global Design Optimization with Kriging*. PhD thesis, University of Michigan, 2002.

M. J. Sasena, P. Papalambros, and P. Goovaerts. Exploration of metamodeling sample criteria for constrained global optimization. *Engineering Optimization*, 34:263–278, 2002.

J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, 1985.

V. Schmitt and F. Charpin. Pressure distributions on the onera-m6-wing at transonic mach numbers. Technical Report AGARD-AR-138, Office National d'Etudes et de Recherches Aerospatiales, 1976.

M. Schonlau. *Computer Experiments and Global Optimization*. PhD thesis, University of Waterloo, 1997.

O. Schütze, A. Lara, and C. A. Coello Coello. On the influence of the number of objectives on the hardness of a multiobjective optimization problem. *IEEE Transactions on Evolutionary Computation*, 2010.

T. W. Simpson. Kriging models for global approximations in simulation-based multidisciplinary design optimization. *AIAA Journal*, 39:2233–2241, 2001.

T. W. Simpson, D. K. J. Lin, and W. Chen. Sampling strategies for computer experiments: Design and analysis. *International Journal of Reliability and Applications*, 2 (3):209–240, 2001.

A. Sóbester and A. J. Keane. Empirical comparison of gradient-based methods on an engine-inlet optimization problem. In *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Atlanta, Georgia, September 2002.

A. Sóbester, S. J. Leary, and A. J. Keane. A parallel updating scheme for approximating and optimizing high fidelity computer simulations. *Structural Multidisciplinary Optimization*, 27:371–383, 2004.

N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Journal of Evolutionary Computation*, 2(3):221–248, 1995.

P. D. Surry and N. J. Radcliffe. The comoga method: Constrained optimization by multi-objective genetic algorithms. *Control and Cybernetics*, 26:391–412, 1997.

J. D. Svenson. *Computer Experiments: Multiobjective Optimization and Sensitivity Analysis*. PhD thesis, Ohio State University, 2011.

D. J. J. Toal. *Proper Orthogonal Decomposition and Kriging Strategies for Design*. PhD thesis, University of Southampton, 2009.

D. J. J. Toal, N. W. Bressloff, and A. J. Keane. Kriging hyperparameter tuning strategies. *AIAA Journal*, 46:1240–1252, 2008.

V. Torczon. On the convergence of pattern search algorithms. *SIAM Journal on Optimization*, 7(1):1–25, 1997.

E. Vazquez and J. Bect. Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and Inference*, 140(11):3088–3095, 2010.

F. A. C. Viana and R. T. Haftka. Surrogate-based optimization with parallel simulations using the probability of improvement. In *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, Fort Worth, Texas, September 2010a.

F. A. C. Viana and R. T. Haftka. Why not run efficient global optimization algorithm with multiple surrogates? In *51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2010b.

A. Vicini and D. Quagliarella. Airfoil and wing design through hybrid optimization strategies. *AIAA Journal*, 37(5):634–641, 1998.

I. Voutchkov and A. J. Keane. Multiobjective optimization using surrogates. In *7^{th} International Conference Adaptive Computing in Design and Manufacture*, 2006.

I. Voutchkov and A. J. Keane. *Computational Intelligence in Optimization*, chapter Multi-objective Optimization using Surrogates, pages 155–175. Springer-Verlag, 2010.

T. Wagner, M. Emmerich, A. Deutz, and W. Ponweiser. On expected-improvement criteria for model-based multi-objective optimization. In R. Schaefer, C. Cotta, J. Kolodziej, and G. Rudolph, editors, *Parallel Problem Solving from Nature - PPSN XI*, volume 6238 of *Lecture Notes in Computer Science*, pages 718–727, Kraków, Poland, September 2010. Springer-Verlag.

A. Waldock and D. Corne. Multiple objective optimization applied to route planning. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, Dublin, July 2011.

A. G. Watson and R. J. Barnes. Infill sampling criteria to locate extremes. *Mathematical Geology*, 27(7):580–608, 1995.

B. J. Williams, T. J. Santner, W. I. Notz, and J. S. Lehman. *Statistical Modelling and Regression Structures*, chapter Sequential Design of Computer Experiments for Constrained Optimization, pages 449–472. Mathematics and Statistics. Physica-Verlag, Berlin, 2010.

R. Yoneta, D. Sasaki, and K. Nakahashi. Aerodynamic optimization of an over the wing nacelle mount configuration. In *48th AIAA Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, January 2010.

D. Yuret and M. Maza. Dynamic hill climbing: Overcoming the limitations of optimization techniques. In *Proceedings of the 2nd Turkish Symposium on Artificial Intelligence and ANN*, 1993.

Q. Zhang and H. Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

Q. Zhang, W. Liu, E. Tsang, and B. Virginas. Expensive multiobjective optimization by moea/d with gaussian process model. *IEEE Transactions on Evolutionary Computation*, 2009.

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

E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.