UNIVERSITY OF SOUTHAMPTON

Proper Orthogonal Decomposition & Kriging Strategies for Design

by

David J.J. Toal

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

in the
FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS
School of Engineering Sciences

October 2009

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS
SCHOOL OF ENGINEERING SCIENCES

Doctor of Philosophy

Proper Orthogonal Decomposition & Kriging Strategies for Design

by David J.J. Toal

The proliferation of surrogate modelling techniques have facilitated the application of expensive, high fidelity simulations within design optimisation. Taking considerably fewer function evaluations than direct global optimisation techniques, such as genetic algorithms, surrogate models attempt to construct a surrogate of an objective function from an initial sampling of the design space. These surrogates can then be explored and updated in regions of interest.

Kriging is a particularly popular method of constructing a surrogate model due to its ability to accurately represent complicated responses whilst providing an error estimate of the predictor. However, it can be prohibitively expensive to construct a kriging model at high dimensions with a large number of sample points due to the cost associated with the maximum likelihood optimisation.

The following thesis aims to address this by reducing the total likelihood optimisation cost through the application of an adjoint of the likelihood function within a hybridised optimisation algorithm and the development of a novel optimisation strategy employing a reparameterisation of the original design problem through proper orthogonal decomposition.

Contents

List of Figures

List of Tables

Declaration of Authorship

I, David John James Toal, declare that the thesis entitled "*Proper Orthogonal Decomposition and Kriging Strategies for Design*" 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 have contributed myself;
- parts of this work have been published as: Toal et al. [2008a,b, 2009a,b].

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

Acknowledgements

There are a great many people who have provided insight, assistance and encouragement to me throughout the course of my three years of study.

Special thanks should be given to my supervisors, Dr. Neil Bressloff, Prof. Andy Keane and Dr. Carren Holden of Airbus U.K., firstly for giving me the opportunity to work with them and secondly for allowing me to benefit from their vast experience and knowledge in the field of engineering design optimisation.

Thanks should go to Dr. Alexander Forrester and Dr. Andras Sóbester for their encouragement, insight and advice. Thanks should also go to Dr. Ivan Voutchkov who I'm indebted to for his knowledge and patience when it comes to matters of compute cluster operation. Dr. David Philpott of IHS ESDU International deserves a special mention for his examination of the FP source code upon my request, no mean feat considering he "almost lost the will to live" upon first opening it.

Thanks should also go to Dr. Maria Ribera-Vicent, Mrs. Jennifer Forrester and Mr. Anthony Scurr for their moral support and for their lunchtime discussions which helped brighten up many dreary afternoons.

Dr. Thomas Barrett, Mr. Alan Burke-Veliz, Mr. Giles Endicott, Miss. Elizabeth Hart, Dr. Athanasios Makrodimopoulos, Mr. Adam Nurdin, Mr. Steven Powell and Mr. Praveen Thokala all deserve special thanks for their help, support and general tomfoolery, without which the office would be a much duller place. Thanks should go to the remaining members of the Computational Engineering and Design group and especially Ms. Rosalind Mizen for welcoming me into the fold and making me feel right at home.

Special thanks should go to my mother, father and brother, Philip, who's unwavering support of my studies and career I'll never forget and yes mother, this does mean I can finally get a "proper job".

Finally I'd like to thank Miss. Lindsay-Marie Armstrong for her support, encouragement and for generally keeping me sane when things go wrong.

Nomenclature

Greek Characters

$\boldsymbol{\alpha}$	Vector of modal coefficients
$\boldsymbol{\Lambda}$	Vector of eigenvalues
$\boldsymbol{\Omega}$	Matrix of eigenfunctions
$\boldsymbol{\omega}_i$	i^{th} eigenfunction
$\Delta\boldsymbol{x}$	BFGS step length
η	Non-dimensional spanwise position
γ	Step length fraction
λ	Regression constant
$\Lambda_{1/4}$	Wing quarter chord sweep angle
μ	Kriging mean
ϕ	Concentrated log-likelihood
ψ	Any kriging hyperparameter
σ^2	Kriging variance
θ	Hyperparameter governing rate of correlation decrease

Roman Characters

$\boldsymbol{A}$	Matrix of modal coefficients
A	Amplitude of Hicks-Henne function
A_r	Wing aspect ratio
$\boldsymbol{C}$	POD correlation matrix
c	Aerofoil section chord

c_1	Particle swarm cognitive parameter
c_2	Particle swarm social parameter
$\bar{c}$	Wing mean aerodynamic chord
C_D	Wing drag coefficient
C_d	Aerofoil drag coefficient
C_{df0}	Aerofoil zero lift skin friction drag coefficient
C_{DPmin0}	Wing's minimum profile drag coefficient
$(\Delta C_{DPmin})_\varepsilon$	Increment in wing profile drag dude to twist
C_{Dvisc}	Wing viscous drag coefficient
C_L	Wing lift coefficient
C_l	Aerofoil lift coefficient
C_{Lmin}	Wing lift coefficient for minimum viscous drag
$\boldsymbol{D}$	Fluctuations of snapshot ensemble from the mean
d	No. of dimensions
$\boldsymbol{E}$	Matrix of eigenvectors
$E(I)$	Expected improvement
$\boldsymbol{H}$	Hessian matrix
I	Improvement over the current best point
K	Particle swarm constriction factor
k_B	Boltzmann constant
K_{visc}	Lift-dependent viscous drag factor
$\boldsymbol{L}$	Cholesky factorisation
$\boldsymbol{l}$	Variable lower bounds
M	No. of snapshots
n	No. of sample points
P	Boltzmann probability factor
p	Hyperparameter governing the degree of smoothness

$\boldsymbol{R}$	Kriging correlation matrix
r_1	Vector of random numbers $\in [0, 1]$
$\bar{\boldsymbol{R}}$	Adjoint of the kriging correlation matrix
$R_{\bar{c}}$	Reynolds number based on wing mean aerodynamic chord
$\boldsymbol{S}$	Ensemble of snapshot vectors
$s^2(\boldsymbol{x}^*)$	Error of the kriging predictor at $\boldsymbol{x}^*$
$\bar{\boldsymbol{s}}$	Mean of snapshot ensemble
$\boldsymbol{s}_i$	i^{th} snapshot vector
$\boldsymbol{T}$	Temporary vector used in the likelihood adjoint calculation
T	Temperature
t	Sharpness of Hicks-Henne function
$\boldsymbol{u}$	Variable upper bounds
$\boldsymbol{V}$	Particle's velocity vector
v_i	An algorithm's i^{th} intermediate variable
$\dot{v}_i$	Forward differentiation of the i^{th} intermediate variable
$\bar{v}_i$	Reverse differentiation of the i^{th} intermediate variable
w	Inertial weight of particle swarm velocity update
x	Design variable
$\boldsymbol{x}^*$	Unknown point in surrogate model
$\boldsymbol{x}_{Gbest}$	Current global best position found by swarm
$\boldsymbol{x}_{Pbest}$	Individual best position found by a particle
x_p	Position of maximum of Hicks-Henne function
y	Objective function
$y(\boldsymbol{x}^*)$	Kriging prediction at $\boldsymbol{x}^*$
$y_{min}(\boldsymbol{x})$	Current best objective of sampled points

Chapter 1

Introduction

1.1 Why Optimise?

Throughout the course of history man has pushed forward the boundaries of technology with the aim of improving the quality of life for his fellow man and, rather less idealistically, generating revenue. This is evident throughout every scientific discipline but particularly so in the field of engineering.

Advances made by engineers can come in the form of sudden revolutions in design or more subtle improvements to existing technology. Innovation and refinement are equally important in the quest for a better solution to a problem. An innovative idea may only truly offer a substantial improvement over an existing technology after a number of refinements. Likewise subtle changes to existing technology only provide a finite level of improvement; without substantial innovation potentially significant performance gains cannot be achieved.

(a)

(b)

Figure 1.1: The Wright Flyer at Kitty Hawk (a) and a Rolls Royce Trent 900 engine on an Airbus A380 (b)

The history of aircraft propulsion is an excellent example of the balance between refinement and innovation. The propeller, as used by the Wright brothers in their first flight in 1903, Figure 1.1(a), was deemed the only viable method of aircraft propulsion until the end of the Second World War. Its dominance led to a series of improvements in blade section design, structural design, materials and manufacturing techniques. The invention of the turbo-jet in the 1930's proved to be the next leap in propulsion technology but it wasn't until after several years of research and development that the turbo-jet became a practical means of aircraft propulsion. It was longer still before it began to overtake the propeller as a means of propulsion on civil aircraft.

One could consider this general process of refinement and innovation as an optimisation process, with an initial concept improved gradually through the mechanism of traditional research and development or a completely new approach discovered, seemingly, through a stroke of genius. In the early 20$^{\text{th}}$ century such optimisation followed a more practical approach with only rudimentary calculations possible and scale model testing the order of the day. However, thanks to modern computational power optimisation techniques have evolved considerably into an effective and efficient aid to designers.

Optimisation methodologies can come in a variety of different guises from a simple trial and error approach through to the classic Newton-Raphson method and modern global optimisers such as genetic algorithms. All optimisation algorithms however, have the same general goal, to improve upon an initial design, be it through a local Newton like improvement or through the global exploration offered by a genetic algorithm which can lead to radically different designs.

Optimisation therefore plays an extremely important role in engineering and only through the application of optimisation techniques can an engineer achieve a design's full potential.

1.2 Global Optimisation for Design

Over the past 10-15 years global optimisation techniques, once the domain of only specialised research groups, have gradually filtered down to the level of industry to such a degree that they are actively employed in various engineering disciplines. The proliferation of these methods has gone hand in hand with an increase in computational power, allowing simulations which once were thought to be in the realms of fantasy, to become a matter of routine.

Nowhere is this more evident than in the field of aerodynamic design where once cutting edge potential solvers have been superseded by Reynolds-Averaged Navier-Stokes

(RANS) simulations. The capabilities of such simulations have sped up the overall design process, reducing the need for large numbers of expensive and time consuming wind tunnel simulations and allowing the utilisation of optimisation techniques.

A typical aerodynamic optimisation involves the parameterisation of a geometry, the controlling variables of which perturb the wetted surfaces. These variables are then selected in such a manner as to optimise an objective function, for example, to minimise drag. An optimisation can lead to either a global or local optimum and take a few or many function evaluations depending on the strategy employed.

Although faster and more flexible than wind tunnel experiments in terms of geometry generation, simulations of 3D wings, or complete aircraft configurations, are still expensive relative to other optimisation problems, taking perhaps 8 hours per simulation even with the assistance of parallel processing. This prohibits the direct use of global optimisation techniques which can require many function evaluations, such as genetic algorithms, Muyl et al. [2004], Shahrokhi and Jahangirian [2007].

An alternative approach is for the global optimiser to evaluate a cheaper model of the true problem. This "surrogate" is constructed from an initial design of experiments (DOE) based sampling of the problem and attempts to model the response of the objective function to changes in the magnitude of the variables via an analytical function. The evaluation of the predicted objective from this model is considerably cheaper than an evaluation of the true objective through, for example, a CFD simulation. The model is however an approximation to the true response and as such requires updating in regions of interest, but on the whole, considerably fewer true objective function evaluations are required for a complete optimisation.

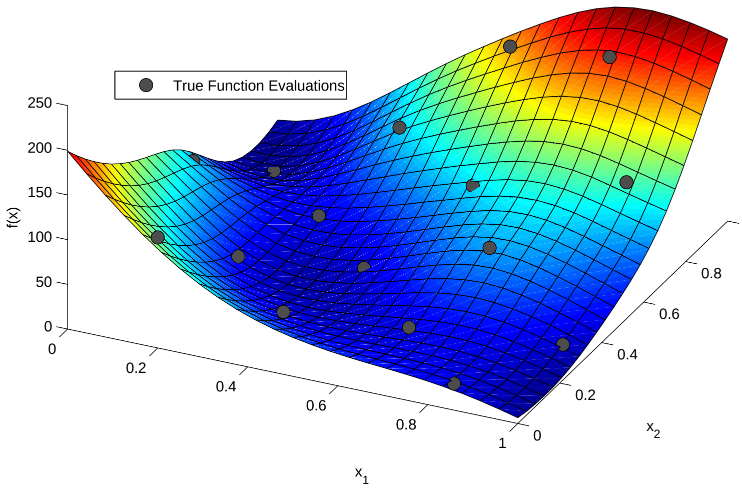


Figure 1.2: A simple example of a surrogate model constructed from a 16 point sampling of the two-dimensional Branin function

There are a variety of different surrogate modelling techniques, from simple polynomials, to Shepard weighting, radial basis functions, support vector regression and kriging, each with their own advantages and disadvantages. However, as the complexity of the model increases so to does the cost incurred through the selection of an appropriate set of modelling parameters. Where the coefficients of a polynomial can be calculated via least squares, the parameters of a kriging model must be selected, or “tuned”, via a complex multi-modal optimisation. This total modelling cost can become a considerable issue when considering high dimensional problems with many sample points.

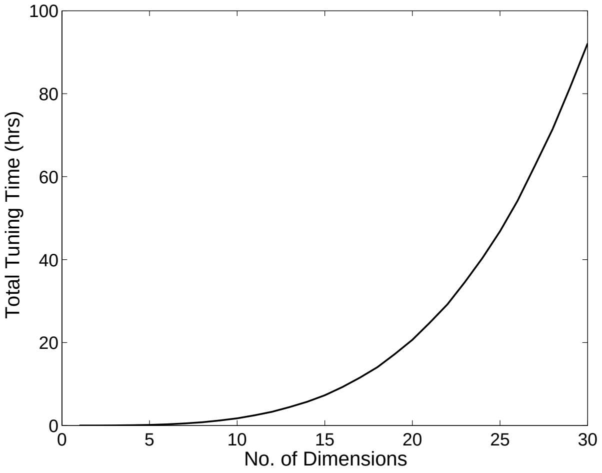


FIGURE 1.3: An illustration of the total time spent tuning the modelling parameters of a krig over the course of an optimisation as problem dimensionality increases

Figure 1.3 helps to illustrate the magnitude of this problem which in a real world engineering environment can lead to significant bottlenecks in the design process. Consider, as an example, a complex 25 variable optimisation problem within an industrial setting where there is a limited amount of time available to obtain a solution. 375 simulations[1], each taking eight hours[2], take 12.5 days to run if ten simulations are run in parallel. If approximately 46 hours[3] are spent tuning the surrogate model over the course of an optimisation an “answer” can therefore be obtained within 14.5 days. In this example the tuning cost is quite significant and reduces the amount of time an engineer has available to evaluate potential designs. A negligible modelling cost would therefore result in the utilisation of a further 60 simulations within the optimisation which could in turn produce a better final design.

The problem is compounded further as the dimensionality of the optimisation problem increases. Not only is the expense of generating the model increased but the number of available evaluations is no longer sufficient to optimise the problem effectively. The issue

[1] Consisting of a 125 point DOE and 25 batches of 10 updates to the model

[2] Approximate time for a RANS simulation of a wing in Fluent with 2 million cells and eight processors

[3] Total tuning time on a Intel Core 2 processor running at 2.4Ghz with 1Gb of RAM assuming the surrogate is tuned after every set of updates

of optimising high dimensional problems given a limited simulation budget is therefore an extremely pertinent one.

1.3 Research Objectives

The objectives of the presented research are two-fold, to reduce the construction cost of kriging based response surface models and to attempt to improve the performance of such models at higher dimensions. Both goals are coupled somewhat, as a reduction in construction effort allows a designer to reallocate time to the evaluation of the true objective function and hence improve performance at higher dimensions.

Over the course of a typical kriging based optimisation, successive updates to the model, in the form of true objective function evaluations, require the reassessment of the surrogate model's controlling hyperparameters. This process involves a complex global optimisation of a multi-modal space and can be very costly. The initial focus of the presented research is therefore to determine how often and how well to tune these "hyperparameters" and what impact this has on an optimisation's ability to produce good designs.

The natural conclusion of this area of research is the consideration of the precise optimisation strategy employed within the hyperparameter tuning process. Is it sufficient to have a local search, or must a global search with a terminal local search, which is traditionally the case, be utilised? Can the hyperparameter tuning effort be reduced via the implementation of more complex hybridised strategies which employ gradient information?

A designer may typically apply a variable screening technique to a high dimensional optimisation problem in an attempt to reduce the number of variables to a more manageable level. This process does however have a number of drawbacks. First a number of expensive function evaluations are required to determine which variables are the most important and even then there is no guarantee that the screening procedure has selected the correct set of variables. An incorrect reduced set of variables and a reduction to the remaining budget of function evaluations for the actual optimisation reduces the quality of the final design. Secondly, the act of variable reduction removes a great deal of flexibility from the design space and may again reduce the quality of a final design. The second area of research therefore concerns how to better reduce the number of variables in a problem without wasting valuable function evaluations whilst attempting to maintain a greater degree of flexibility within the reduced design space.

1.4 Thesis Overview

The following thesis begins with a review of existing optimisation techniques. Local quasi-Newton based searches are considered as well as stochastic optimisers such as genetic algorithms, simulated annealing and dynamic hill climbing. Particular emphasis however is given to particle swarm optimisation and response surface modelling in the form of kriging, as these methods form the backbone of the novel techniques described in subsequent chapters.

The importance of the process of hyperparameter tuning within a kriging based response surface optimisation is then considered. In particular the effect on an optimisation of how often and how well the hyperparameters are tuned. An n dimensional aerofoil inverse design problem is employed within this study and provides an excellent opportunity to demonstrate the effect of increasing dimensionality on the performance of an optimisation given a restricted budget of function evaluations.

Having discerned the impact of tuning within the overall optimisation process, the actual optimisation method employed within this process is then considered. Typically the tuning process involves a stochastic optimisation followed by a terminal search. However, this does not take into consideration the smoothness of the likelihood space and the existence of an analytical gradient of the likelihood, both of which provide ideal circumstances for the application of a gradient based search. To improve the efficiency of the gradient calculation an adjoint of the likelihood is derived via reverse algorithmic differentiation. The efficiency of this method of gradient calculation is then compared to the traditional analytical gradient calculation, finite differencing and a forward algorithmic differentiation formulation.

The multi-modal nature of the likelihood space makes it unsuitable for a purely local optimisation. To this end a sequential quadratic programming (SQP) algorithm utilising the adjoint formulation of the likelihood is hybridised with a particle swarm. The particle swarm therefore carries out a global search, with the SQP rapidly exploiting promising regions. The developed tuning method is once again applied to the aerofoil inverse design problem in order to directly compare it to the more traditional, and more costly, tuning processes.

The improvement of kriging at higher dimensions through the reparameterisation of the design problem is then considered. The traditional kriging process is divided into two kriging optimisations linked via a reparameterisation of the original design problem. The first optimisation is performed on the original problem and includes a number of updates while the second is performed on a reparameterised version of the original problem where the common geometric features of the best designs have been extracted through the application of proper orthogonal decomposition. This optimisation strategy, termed geometric filtration, results in a considerable reduction in the size of the kriging correlation matrix and therefore reduces the cost of the tuning optimisation further.

The controlling parameters of this novel optimisation technique are then investigated. Particular attention is paid to the effort applied in the initial optimisation, the size of the sampling plans within both optimisations, and the number of designs used in the reparameterisation process. The observations made regarding the impact of a reduced correlation matrix size on the cost of the hyperparameter tuning process is also applied to the traditional kriging process. The impact of this restricted sampling size on the optimisation process is then compared to the geometric filtration technique.

Finally, the techniques developed throughout the course of the presented research are combined and implemented in the optimisation of a transonic wing. This final optimisation demonstrates completely the performance gains achieved by the presented research.

Chapter 2

A Review of Existing Optimisation Techniques

2.1 Local Optimisation

2.1.1 An Overview of Local Optimisation

The optimisation of any engineering design problem usually begins with an initial design point. This point could be the result of a previous design process or the culmination of an engineer's experience and knowledge of what should constitute a good design. For example an experienced aerodynamicist wishing to create an optimal aerofoil would begin with an existing aerofoil which displays characteristics close to those desired.

Even the most inexperienced engineer without any knowledge of optimisation is aware that subtle changes to an initial design can improve upon that initial design's performance. The addition of a small bump to the upper surface of an aerofoil in the region of a shockwave, for example, can alleviate the shock and improve the drag characteristics of the aerofoil for a particular flight condition, Huyse et al. [2002].

This process of altering an initial design point in an attempt to continually improve an objective function results in a gradual descent towards an optimum. However, this optimum is usually only local in nature with the objective function smaller than all other points in the vicinity, but not necessarily the smallest objective function value within the complete design space, see Figure 2.1.

Local optimisation techniques are therefore unlikely to locate the global optimum unless the optimisation is started in the region of that optimum, the objective function is unimodal in nature or multiple restarts of the optimisation are employed. Even with these obvious drawbacks, local optimisation techniques are a popular tool in engineering design as they require relatively few objective function calculations, when there are a modest number of variables, compared to global stochastic methods, and can make effective use of gradient or curvature information when available.

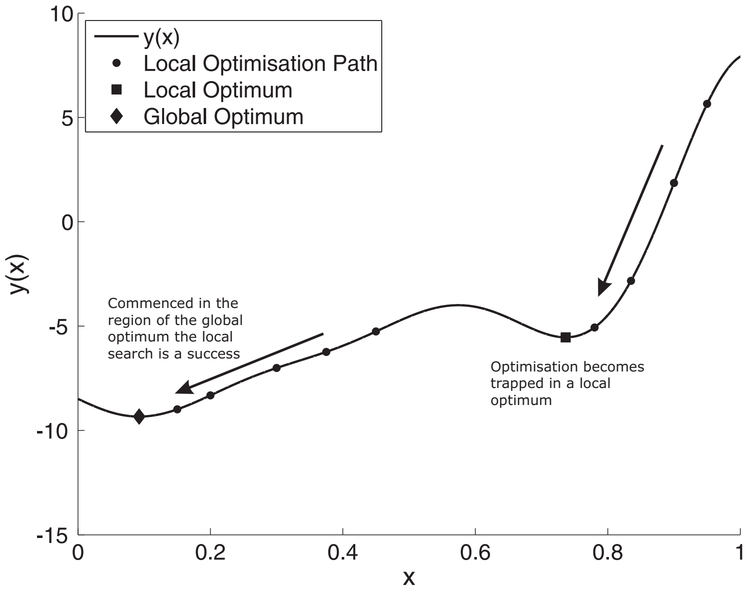


Figure 2.1: An example of the pitfalls of local optimisation given the multi-modal Forrester function, $y(x) = 0.5(6x-2)^2 \sin(12x-4) + 10(x-0.5) - 5$ where $x \in [0,1]$

There are many different local optimisation techniques throughout the literature, from the simple, but robust, Nelder-Mead simplex algorithm, Press et al. [1986], to conjugate gradient algorithms, pure Newton methods, Nocedal and Wright [1999] and quasi-Newton methods, Broyden [1970] each of which has been extensively used in both the optimisation of analytical functions and engineering design problems, Burgreen and Baysal [1994], Elliot and Peraire [1996], Giles and Pierce [2000].

2.1.2 Quasi-Newton Methods

The quasi-Newton method is one of the most popular local optimisation algorithms for both analytical functions and design optimisations. Quasi-Newton methods are based upon the Newton search method, though instead of utilising the Hessian of the objective function at a point, a model of the inverse of the Hessian is gradually constructed as the search progresses. This is particularly advantageous when the precise Hessian of the objective function is unavailable, which is usually the case in a design optimisation. To begin to comprehend how a quasi-Newton optimisation algorithm searches a design space it is necessary to first review the Newton method upon which it is based.

An objective function, $y(\boldsymbol{x})$, can be approximated by a Taylor series expansion of the form,

$$y(\boldsymbol{x}) = c + \sum_i \frac{\partial y}{\partial x_i} x_i + \frac{1}{2} \sum_{ij} \frac{\partial^2 y}{\partial x_i \partial x_j} x_i x_j + \ldots, \tag{2.1}$$

by neglecting terms of $O(x^3)$ and higher, a quadratic approximation to the objective function is obtained, Press et al. [1986],

$$y(\boldsymbol{x}) \approx c + \boldsymbol{b}\boldsymbol{x} + \frac{1}{2}\boldsymbol{x}^T \boldsymbol{H}\boldsymbol{x}. \tag{2.2}$$

Where $\boldsymbol{H}$ is a positive definite square matrix, denoted the Hessian matrix, $\boldsymbol{b}$ is a vector and c is a constant. Differentiating this expression produces an approximation of the gradient of the objective function,

$$\nabla y(\boldsymbol{x}) = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{b}. \tag{2.3}$$

At an extremum of the function this derivative will equal zero. Newton's search method aims to find the zero of the gradient, hence the gradient at a new point, $\boldsymbol{x}_{i+1}$, is a function of the current point, the gradient at the current point and the Hessian matrix,

$$\nabla y(\boldsymbol{x}_{i+1}) \approx \nabla y(\boldsymbol{x}_i) + \boldsymbol{H}(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i). \tag{2.4}$$

Equating the gradient at the new point to zero produces an equation defining the step length, $\Delta\boldsymbol{x}$, to the minimum of the quadratic approximation as,

$$\Delta\boldsymbol{x} = \boldsymbol{H}^{-1}\nabla y(\boldsymbol{x}_i). \tag{2.5}$$

It is this step length which is used to calculate the next point in the optimisation,

$$\boldsymbol{x}_{i+i} = \boldsymbol{x}_i + \gamma\Delta\boldsymbol{x}, \tag{2.6}$$

where γ is a fraction of the step length calculated from a line search in the direction of the step.

As mentioned previously, quasi-Newton methods attempt to slowly build up a model of the inverse of the Hessian matrix as the number of steps in the optimisation increases,

$$\lim_{i\to\infty} \tilde{\boldsymbol{H}}_i = \boldsymbol{H}^{-1}, \tag{2.7}$$

where $\tilde{\boldsymbol{H}}$ is the approximation to the inverse. The quasi-Newton algorithm begins with an initial guess of the inverse of the Hessian at the starting point. This initial guess is typically a function of the identity matrix, Nocedal and Wright [1999]. Once the algorithm has moved to a new point the model of the inverse Hessian, $\tilde{\boldsymbol{H}}$, is updated using a correction term which typically utilises the position and gradient information of both the old and new point. The approximation of the inverse of the Hessian at a new point is therefore of the form,

$$\tilde{\boldsymbol{H}}_{i+1} = \tilde{\boldsymbol{H}}_i + \text{correction}, \tag{2.8}$$

where the formulation for this correction differs between each quasi-Newton method. The popular Broyden-Fletcher-Goldfarb-Shanno (BFGS), Broyden [1970], formula is,

$$\tilde{\boldsymbol{H}}_{i+1} = \tilde{\boldsymbol{H}}_i + \frac{\boldsymbol{g}\boldsymbol{g}^T}{\boldsymbol{g}^T\boldsymbol{s}} - \frac{\tilde{\boldsymbol{H}}_i\boldsymbol{s}\boldsymbol{s}^T\tilde{\boldsymbol{H}}_i}{\boldsymbol{s}^T\tilde{\boldsymbol{H}}_i\boldsymbol{s}}, \tag{2.9}$$

where

$$\boldsymbol{s} = \boldsymbol{x}_{i+i} - \boldsymbol{x}_i, \tag{2.10}$$

and

$$\boldsymbol{g} = \nabla y(\boldsymbol{x}_{i+1}) - \nabla y(\boldsymbol{x}_i). \tag{2.11}$$

Other simpler updating formulae, such as that of Davidon-Fletcher-Powell, exist but BFGS is widely considered to be superior, Press et al. [1986]. With an updated Hessian, the Newton step, Equation 2.5, can be calculated and used to determine the direction of the next line search, the process is then repeated until convergence or a budget of function calls is exhausted.

Quasi-Newton methods are very efficient at locating local minima and may even outperform the Newton method when the starting point is far from an optima and the objective function is not quadratic in nature. However, Quasi-Newton methods, like the Newton method upon which they are based may perform badly when the objective function is noisy in nature, which is typically the case when an optimisation employs a computational experiment to calculate the objective function that is either iterative or employs a discretisation scheme, Keane and Nair [2005].

Although the above formulation utilises gradient information directly, a series of finite differences can be used to calculate the gradients when an analytical gradient is unavailable. For problems with large numbers of variables this can be very expensive, though considerably less expensive than attempting to calculate the Hessian via finite differencing. It is the quasi-Newton method's direct application of gradient information in the optimisation process which makes it particularly popular when the adjoint formulation of a computational experiment is available, Giles and Pierce [2000].

The quasi-Newton approach described above is only applicable to unconstrained optimisation problems. However the technique can be modified to deal with constraint boundaries in a number of ways, for example, through the application of a gradient projection method. The BFGS updating formula can also be used to derive Hessian approximations in other optimisation algorithms. Sequential quadratic programming, for example, can utilise a BFGS based approximation to the Hessian of the Lagrangian thus enabling the local optimisation of problems with non-linear constraints.

2.2 Stochastic Optimisation Methods

2.2.1 Genetic Algorithms

The genetic algorithm (GA) is one of the most popular stochastic optimisation methodologies. Initially developed by Holland [1962], genetic algorithms are inspired by the Darwinian theory of natural selection, whereby desirable or advantageous traits become more common as a population reproduces and undesirable or disadvantageous traits die out. Unlike the local optimisation techniques previously discussed, the GA, as well as the other stochastic methods discussed here, are global optimisers.

Genetic algorithms, like most other stochastic optimisation techniques, employ a population which gradually evolves over the course of a number of generations. It is the mechanism of this evolution which is unique to the genetic algorithm. Genetic algorithms generally employ three processes, "selection", "crossover" and "mutation" in the search for a global optimum. These processes attempt to move the members of a population away from undesirable regions of a design space and towards a global optimum. This is analogous to the theory of natural selection whereby ailing members of a population of animals gradually die out leaving only the strongest members to breed and hence benefit the species.

In terms of the genetic algorithm, the "fitness" of a population is generally derived from the objective function through the application of a suitable scaling. In a minimisation, for example, those members of the population which minimise the objective function most have a high level of fitness associated with them. Members of a population with a higher fitness are more likely to be selected for reproduction and added to the mating pool. Hence the higher the fitness value the more likely the characteristics of that member of the population will be passed onto members of the next generation.

The selection of the members to make up the mating pool is followed by cross-over, also known as mating. Generally, two individuals, randomly chosen from the mating pool, are selected for cross-over. In the case of a binary encoded population the process of cross-over entails the exchanging of bits between two parents which results in two offspring. These offspring then form the population of the next generation.

The third and final procedure is that of mutation which involves the alteration of a randomly selected variable by a small amount for a small proportion of the offspring. The process of mutation implies a random walk through the design space enabling the optimisation to escape regions of local minima and introducing a level of global exploration. The performance of genetic algorithms are governed by four important parameters, the size of the population, the probability of cross-over, the probability of mutation and the number of generations that the algorithm is run for. The selection of appropriate values of these parameters greatly affects the convergence of an optimisation.

Considerable time and effort can be spent manually adjusting these parameters to gain the best performance of a genetic algorithm over a range of problems, Keane [1995], Gudla and Ganguli [2005].

Another method for improving the performance of a genetic algorithm is the adoption of elitism. This entails the transfer of the best member of each population, unaltered, directly to the next generation. The introduction of an elitist strategy within a genetic algorithm ensures a monotonic improvement in the best fitness value of a population from one generation to another, Gudla and Ganguli [2005].

Despite the introduction of elitism and the selection of appropriate values of the parameters mentioned above, a typical GA can suffer from some drawbacks, namely poor exploitation capabilities and premature convergence. Premature convergence can occur due to a lack of population diversity, Goldberg [1989], Chelouah and Siarry [2000], but can be countered by changing the degree and probability of mutation. Poor exploitation is mainly due to the nature of the encoding of each member of the population which can prohibit the more subtle changes to each variable necessary to obtain a precise global optimum. A GA may therefore locate the general region of the global optima but not the precise answer.

The accuracy of the optimum found can be increased through the hybridisation of a GA with a local search strategy. Many such examples of this technique exist throughout the literature with conjugate gradient, quasi-Newton and Nelder-Mead simplex algorithms all employed, Chelouah and Siarry [2003], Gudla and Ganguli [2005], Muyl et al. [2004]. Each of these hybrid strategies works on the same basic principle; the GA is used to locate regions of optimal design and the local search strategy is used to exploit these regions. The only differences between these various hybrid strategies is the local optimisation strategy used in the exploitation and when this exploitation occurs during the course of an optimisation. Some hybrid strategies use a local optimisation strategy to improve upon the best member of the population at each generation, Muyl et al. [2004], while others start from the best point of the final generation, Chelouah and Siarry [2003].

Be it in a more traditional form, or in a hybridised form, the genetic algorithm is a popular optimisation technique and has been used extensively throughout the literature. Genetic algorithms have been applied to numerous aerodynamic optimisation problems, Obayashi and Takanashi [1996], Shahrokhi and Jahangirian [2007], and are particularly popular in the optimisation of problems with large numbers of variables, Chelouah and Siarry [2000], for example, the tuning of surrogate models, Keane [2006].

2.2.2 Simulated Annealing

Simulated annealing is derived from the field of statistical mechanics namely the work of Metropolis et al. [1953] which was adapted by Kirkpatrick et al. [1983] for use in global optimisation. In metallurgy, annealing involves raising the temperature of a substance beyond its recrystalisation point and then slowly lowering the temperature again allowing optimal crystals within the substance to form. If the temperature is lowered too quickly suboptimal crystals form. Simulated annealing is the optimisation equivalent of this process whereby the motion of a number of atoms within a substance at a temperature is simulated.

The simulated annealing process begins with an initial sampling of the design space at a number of points. Each point is the equivalent of an atom in a heated material. The objective function is evaluated for each "atom" and recorded. Each atom then undergoes a small random perturbation and the objective function at its new position is calculated. This random perturbation is accepted if it results in a reduction in the objective function compared to the previous position. The selection of positions resulting in a continual reduction in objective function is analogous to the rapid quenching scenario described previously, and results in only local optima.

To prevent the optimisation getting trapped in local optima a probabilistic approach to the selection of atom positions which result in a worse objective function is introduced. If the random perturbation of an atom results in a worse objective function the probability of that motion being accepted is given by the Boltzmann probability factor,

$$P = \exp\left(\frac{-\Delta y}{k_B T}\right), \tag{2.12}$$

where Δy is the change in the objective function, k_B is the Boltzmann constant and T is the temperature. In a simulated annealing algorithm a random number in the range [0,1] is selected and compared to this probability factor, if this number is less than the probability factor the new position of the atom is retained.

As the optimisation proceeds the temperature is gradually reduced until the system freezes and the atoms no longer move about freely. The starting temperature and rate at which the temperature reduces therefore has a massive impact on the performance of the optimisation.

This impact can be ascertained when one considers the equation for the probability factor. For a given temperature, perturbations in an atom which produce slight increases in the objective function are more likely to be accepted than those which result in large increases in the objective function. As the temperature reduces the likelihood of these perturbations being accepted is reduced forcing the optimisation to move into local minima. Too high a starting temperature results in a global exploration of the design space with little exploitation occurring. The rate of cooling therefore determines the rate at which an optimisation moves from a global exploration to a more local exploitation. Hence, a careful consideration of both of these parameters is necessary for an efficient optimisation.

Like the genetic algorithm, simulated annealing has been used in a variety of optimisation problems involving large numbers of variables, for example the optimisation of Latin hypercubes, Morris and Mitchell [1995], and has found numerous applications in aerodynamic design, Sherif et al. [1996], Wang and Damodaran [2002].

2.2.3 Dynamic Hill Climbing

Dynamic hill climbing, developed by Yuret and Maza [1993], adopted both dynamic coordinate changing and the exploitation of local minima as methods to counter the inherently poor performance of traditional hill climbing techniques on multi-modal functions.

At its core dynamic hill climbing consists of a rather simple, but effective hill climber. An individual is selected at random in the design space, from which a population of points is then constructed through a series of mutations to this individual. In a process similar to that of the mutations within a genetic algorithm, a scalar is added to and subtracted from each of the coordinates of the initial point. The population therefore consists of a total of $2d + 1$ individuals, where d is the number of dimensions in the problem.

Each of the members of the population is evaluated using the objective function and the fittest member is selected. This member then undergoes another set of mutations to produce a new population and so forth. The magnitude of the mutation applied to the best individual of each population is halved every iteration until it is considered small enough to exclude the possibility of any points nearby having a better fitness. When this stopping criterion is reached, the hill climber has reached a local optimum.

Such a simple hill climbing technique suffers from two major flaws. The optimisation can stall because the direction of ascent is not within the permitted directions of travel and the optimisation can become trapped in local optima.

Yuret et al. introduced a dynamic coordinate frame to enable the hill climber to freely choose a more appropriate direction of travel. A Gramm-Schmidt orthogonalisation, using information from a number of previous iterations of the hill-climber, is employed to derive a new set of basis vectors, along which, future mutations can be applied. These orthogonal basis vectors allow the optimisation to proceed in directions which would normally be impossible with a rigid coordinate frame. The optimisation is therefore free to find better optima.

The optimisation algorithm is prevented from being trapped in local optima through the exploitation of the locations of these optima. When the hill climbing algorithm completes, the location of the optimal point found by the algorithm is recorded. The algorithm can then be restarted, but instead of selecting a new starting point at random, a starting point is selected which maximises the diversity from any previous local optima recorded. The hill climbing algorithm is repeated from this starting point until another local minima is found. The complete optimisation process then continues until a specified stopping criterion, such as the number of function evaluations, is reached.

Although not quite as popular as, for example, a genetic algorithm, dynamic hill climbing has been used in the optimisation of kriging hyperparameters, Keane [2006] and has compared favourably to a traditional genetic algorithm in the optimisation of analytical test functions, Yuret and Maza [1993].

2.2.4 Particle Swarms

The particle swarm is one of the most recent stochastic optimisation methods, and also one of the simplest. It was originally developed by Eberhart and Kennedy [1995] after they adopted simulations of simple social behaviour for use in optimisation.

Like the genetic algorithm the particle swarm is inspired by nature, though instead of modelling evolutionary processes, the particle swarm endeavours to model the social behaviour of a population of animals. Initial simulations of social behaviour by Eberhart et al. drew on the previous work of Heppner and Grenander [1990], where a flock of birds flew around searching for a cornfield before landing. In these simulations every member of the population could remember its previous best position, in this case how far it was from the cornfield, and each member could "see" the global best position that another member of the population had found. By permitting each bird to remember a previous best position, birds that overflew a good position were pulled back to that position. Allowing the birds to see the current global best position, in social terms, provides the group with a standard which the other group members attempt to attain.

The extension of this social behaviour model to the field of optimisation resulted in a very simple equation which updates the velocity of each swarm member. Given an initial velocity, $\boldsymbol{V}_i$, and information on the current global best position, $\boldsymbol{x}_{Gbest}$ and the best position that an individual has found, $\boldsymbol{x}_{Pbest}$, the velocity of the particle at the next iteration can be found by,

$$\boldsymbol{V}_{i+1} = w\boldsymbol{V}_i + c_1\boldsymbol{r}_1(\boldsymbol{x}_{Pbest} - \boldsymbol{x}_i) + c_2\boldsymbol{r}_2(\boldsymbol{x}_{Gbest} - \boldsymbol{x}_i). \tag{2.13}$$

Where $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are vectors of random numbers in the range [0,1] and w, c_1, and c_2 are the inertial weight, the cognitive parameter and social parameter respectively, Blasi and Del Core [2007]. With the new velocity of a particle calculated, the position of a particle at the next iteration can be easily found,

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \boldsymbol{V}_{i+1}. \tag{2.14}$$

The updated velocity given by Equation 2.13 has three main components. The first component is the velocity that the particle is currently travelling at; this is termed the inertial component. The second component of the velocity calculation tends to return the particle to a previous best point, this is sometimes referred to as "simple nostalgia", Kennedy and Eberhart [1995]. The final component of the velocity calculation tends to move the particle towards the location of the best point found so far in the optimisation.

The two learning parameters, c_1 and c_2, in Equation 2.13 balance the local exploitation and global exploration of the algorithm. A large c_1 relative to c_2 forces each particle to return to the best positions that they, themselves, have found, largely ignoring the global best point. A large c_2 relative to c_1 results in each particle moving rapidly towards the global best position with little exploitation of any local minima found. A balance is therefore required between these two factors to ensure the algorithm moves to a global optimum, but in doing so exploits any locally optimal regions it encounters. For this reason in Eberhart and Kennedy's original algorithm, both c_1 and c_2, equal 2.

The inertial weight was an addition to the original formulation by Shi and Eberhart [1998] which improved the performance of the algorithm. Typically this inertial weight decreases linearly with each iteration of the algorithm. Eberhart and Shi [2000] found however that modifying Equation 2.13 to use a constriction factor, K and restricting the velocity of each particle to V_{max} negates the need for a gradual reduction in inertial weight, and results in a performance increase. Equation 2.13 therefore becomes,

$$\boldsymbol{V}_{i+1} = K[\boldsymbol{V}_i + c_1\boldsymbol{r}_1(\boldsymbol{x}_{Pbest} - \boldsymbol{x}_i) + c_2\boldsymbol{r}_2(\boldsymbol{x}_{Gbest} - \boldsymbol{x}_i)] \tag{2.15}$$

where

$$K = \frac{2}{|2 - \gamma - \sqrt{\gamma^2 - 4\gamma}|} \qquad \gamma = c_1 + c_2 \quad \text{and} \quad \gamma > 4. \tag{2.16}$$

The constriction factor is in effect a special implementation of the traditional particle swarm equation, only instead of the inertial weight reducing with each iteration it remains constant and is a function of the learning factors. The starting magnitude of the inertial weight and the rate at which it decays no longer need to be specified a priori.

The restriction in the velocity applied to each particle prevents them from leaving the design space and also simulates an incremental learning process. The selection of this parameter has an impact on the performance of the optimisation. Reducing this parameter prevents the particles from overshooting interesting regions but reduces the convergence rate of the algorithm. Setting this parameter too small however prevents the particle escaping from local minima. Typically V_{max} is defined as a fraction of the difference between the upper and lower bounds of the design space, Liu et al. [2006].

A typical particle swarm algorithm begins with the sampling of the design space at a number of points to construct the initial population. Each member of the swarm has its corresponding objective function calculated and is given a random initial velocity within the limits of V_{max}. The global best point is recorded and the previous best point for each particle is initialised to the current position. The velocity is then updated using either Equations 2.13 or 2.15 and from this velocity the particle's new position is calculated. The objective function is then calculated at this new position and the global and previous best points are updated as necessary. The process then continues until a specified stopping criterion is reached.

Particle swarms have grown in popularity since their inception by Eberhart and Kennedy. They have been applied in the conceptual design of aircraft, Blasi and Del Core [2007], and compared favourably to genetic algorithms and simulated annealing in the optimisation of aerofoils, Ray and Tsai [2004], and analytical functions, Angeline [1998], Brandstatter and Baumgartner [2002], Venter and Sobieszcaznski-Sobieski [2003].

Particle swarms do however suffer from the same exploitative deficiencies as genetic algorithms, that is, they are capable of finding the region of an optimal design but not the precise answer. As with genetic algorithms some effort has gone into improving the performance of particle swarms in this area, with the introduction of both hybrid particle swarms which employ a local search, Shu-Kai et al. [2004], or which make specific use of gradient information in the velocity update equation, Noewl and Jannett [2004], Ninomiya and Zhang [2008].

2.2.5 Particle Swarm Neighbourhood Topology

The basic particle swarm of Eberhart and Kennedy bases the velocity update equation on the particle's previous best location and on the location of the global best point. However, using a global best point has been demonstrated to result in the premature convergence of the swarm to a suboptimal solution on some test problems, Kennedy and Mendes [2002], Mohais et al. [2005]. When a global best point is used the particles tend to move towards the best point found in the early iterations of the optimiser. This reduces the extent that each particle explores and can result in the premature convergence of the search.

In an attempt to remove this unwanted behaviour different neighbourhood topologies were introduced to the basic particle swarm. These topologies define the connections between individual particles and therefore control the flow of information around the swarm. Rather than use the best location found by the entire population the velocity update equation would now employ the best location found by a particle's neighbours.

There are typically three criteria for defining the connections between particles, Akat and Gazi [2008]. Particles can be connected to their closest neighbours in either variable or function space or they can be connected randomly to each other. The topology itself can also remain fixed throughout an optimisation or can behave dynamically with connections forming and breaking as the particles explore the space. Dynamic topologies can result in the creation of sub-populations in some problems which may be advantageous, for example, when searching along a constraint boundary.

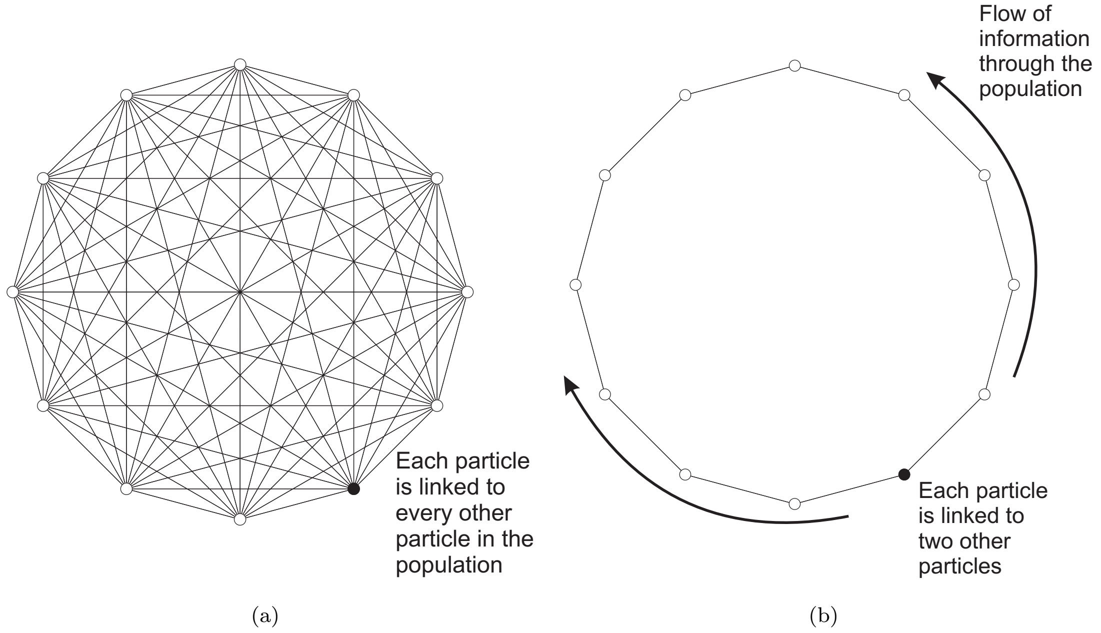

Each particle is linked to every other particle in the population
(a)
Flow of information through the population
Each particle is linked to two other particles
(b)

Figure 2.2: Graphical interpretations of the “star” and “ring” topologies.

The global best or “star” topology is the most simple and therefore the most commonly used. In this topology every particle is connected to every other particle. The previous best position of a neighbouring particle is therefore the best found by any particle so far in the search. This however leads to the premature convergence problems mentioned previously.

The “ring” topology connects each particle to only two others. The velocity update equation therefore uses the previous best position of these other two particles. Only the direct neighbours of the global best particle will therefore be influenced by it with the remaining particles exploring local regions of attraction. As the number of generations increases the influence of the global best point will begin to spread around the swarm with the global best particles neighbours finding better solutions and influencing their neighbours and so forth, Kennedy [1999].

The “ring” topology does not therefore suffer the premature convergence of the “star” topology, but due to the much slower spread of information throughout the swarm, once the true optimum is located it is much slower to be exploited. The “ring” topology could therefore be considered as slower to converge to an optimum but more likely to find the true optimum of a multi-modal function. This has been demonstrated in the literature to occur with a number of analytical test functions, Kennedy [1999], Kennedy and Mendes [2002].

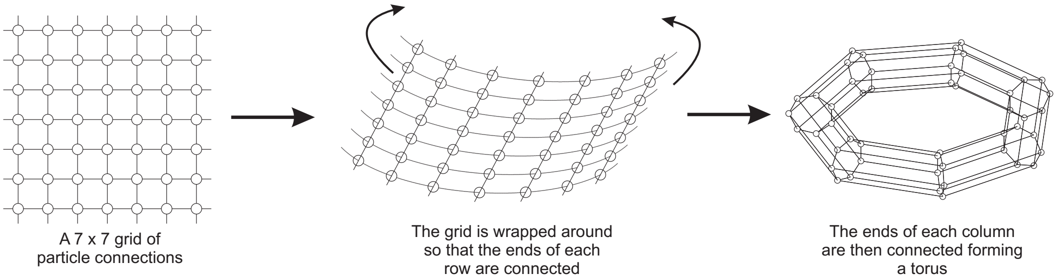


Figure 2.3: A graphical representation of the Von Neumann topology for a population of 49 particles.

An efficient neighbourhood topology must therefore offer a reasonable rate of convergence but without the tendency for premature convergence to a local optimum. Throughout the literature a number of different topologies have been considered each with varying degrees of connectivity between particles. Kennedy and Mendes [2002] and Mohais et al. [2005], for example describe and compare the performance of a number of such topologies with respect to a series of analytical test functions.

Within these comparisons the Von Neumann topology, (so called because of its use in cellular automata by John Von Neumann), is generally considered to be one of the best offering a faster rate of convergence than the "ring" topology but reducing the tendency for premature convergence associated with the "star" topology, Kennedy and Mendes [2002, 2006]. Each particle within the swarm is connected to four others, with the topology resembling a grid wrapped around to form a torus, see Figure 2.3 above. The increased number of neighbourhood connections results in a faster spread of information through the swarm than the "ring" topology which improves the speed of convergence, while the restricted neighbourhood increases the degree of local exploration. The superior performance of the Von Neumann topology relative to the "star" and "ring" topologies was demonstrated further by Huang and Mohan [2005], who compared each topology with respect to the real world problem of microwave image reconstruction.

2.2.6 A Fully Informed Particle Swarm

The fully informed particle swarm, (FIPS), is a further development of the application of neighbourhood topologies to the basic particle swarm. Mendes et al. [2004] considered that important information may be neglected through the overemphasis of a single best neighbour. The FIPS strategy was therefore developed with the intention of having all of a particle's neighbours contribute to the velocity update equation by taking a weighted mean of each of the neighbours previous best positions.

The velocity update equation for FIPS is similar to that of Equation 2.15 except that $\boldsymbol{x}_{Gbest}$ and $\boldsymbol{x}_{Pbest}$ have been replaced by the weighted average of the previous best positions of a particle and all of its neighbours. The velocity update equation therefore becomes,

$$\boldsymbol{V}_{i+1} = K \left[\boldsymbol{V}_i + \frac{\boldsymbol{r}}{N_i} \sum_{j=1}^{N_i} (\boldsymbol{x}_{jbest} - \boldsymbol{x}_i) \right], \tag{2.17}$$

where $\boldsymbol{r}$ is a vector of random numbers in the range $[0, \gamma]$, N_i defines the number of particles within a neighbourhood and $\boldsymbol{x}_{jbest}$ is the previous best position of the j^{th} particle in the neighbourhood. The coefficients K and γ refer to the constriction factor given by Equation 2.16.

Although a constant weighting is used above, the equation can be expanded further to include a weighting scheme based on a metric such as the particle's fitness or the distance to the current particle. Mendes et al. [2004] investigated a number of different weighting schemes but demonstrated that there was little difference in overall performance compared to a uniform weighting.

The investigations into the performance and convergence behaviour of FIPS by Mendes et al. [2004] and Montes De Oca and Stutzle [2008] highlighted the sensitivity of the strategy to neighbourhood topology. Where the traditional swarm performs consistently no matter the topology employed, the performance of FIPS can degrade significantly if an appropriate topology is not selected. Montes De Oca and Stutzle [2008] demonstrated that as neighbourhood connectivity increases the performance of FIPS decreases due to spatial convergence. With a fully connected "star" topology the particle's exploration tends to be biased towards a region close to the centroid of the swarm. Particles can therefore become trapped if this region is close to a local optimum. The FIPS strategy is therefore typically implemented in conjunction with the Von Neumann topology, where the smaller neighbourhood sizes allow the particles to search around the centroid of each neighbourhood rather than of the whole population.

Even though FIPS is a recent adaptation of the traditional swarm and has not been applied to a large number of real world optimisation problems it has been hybridised with a BFGS local improvement step for the purposes of neural network training, Ninomiya and Zhang [2008] and even extended to multi-objective problems by Jia et al. [2007].

2.3 The Surrogate Modelling Approach to Optimisation

2.3.1 The Basic Surrogate Modelling Optimisation Strategy

Although stochastic methods can be relied upon to reach a global optimum they typically require a very large number of evaluations of the objective function to do so. This is not an issue when simple analytical functions or fast low fidelity computational simulations are considered. However, when the objective function is evaluated using an expensive high fidelity simulation, the time required to optimise using a stochastic method can be quite significant. Even with the parallelisation of the objective function calculation it may be infeasible to use a stochastic method of optimisation for some problems.

Surrogate modelling, also known as response surface modelling or metamodelling, aims to facilitate optimisation but with a reduced number of objective function evaluations. In general the stochastic optimisation methods described previously, sample an objective function at a number of points and then define a new set of points based on mutations or translations to the previous set. Surrogate modelling however, aims to use the objective function calculated at a number of points in the design space to construct a model of the entire space. This model then acts as a surrogate for the original computational simulation. It is then this inexpensive model which is evaluated using a stochastic process instead of the original simulation.

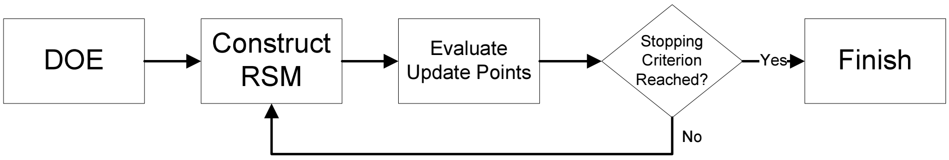
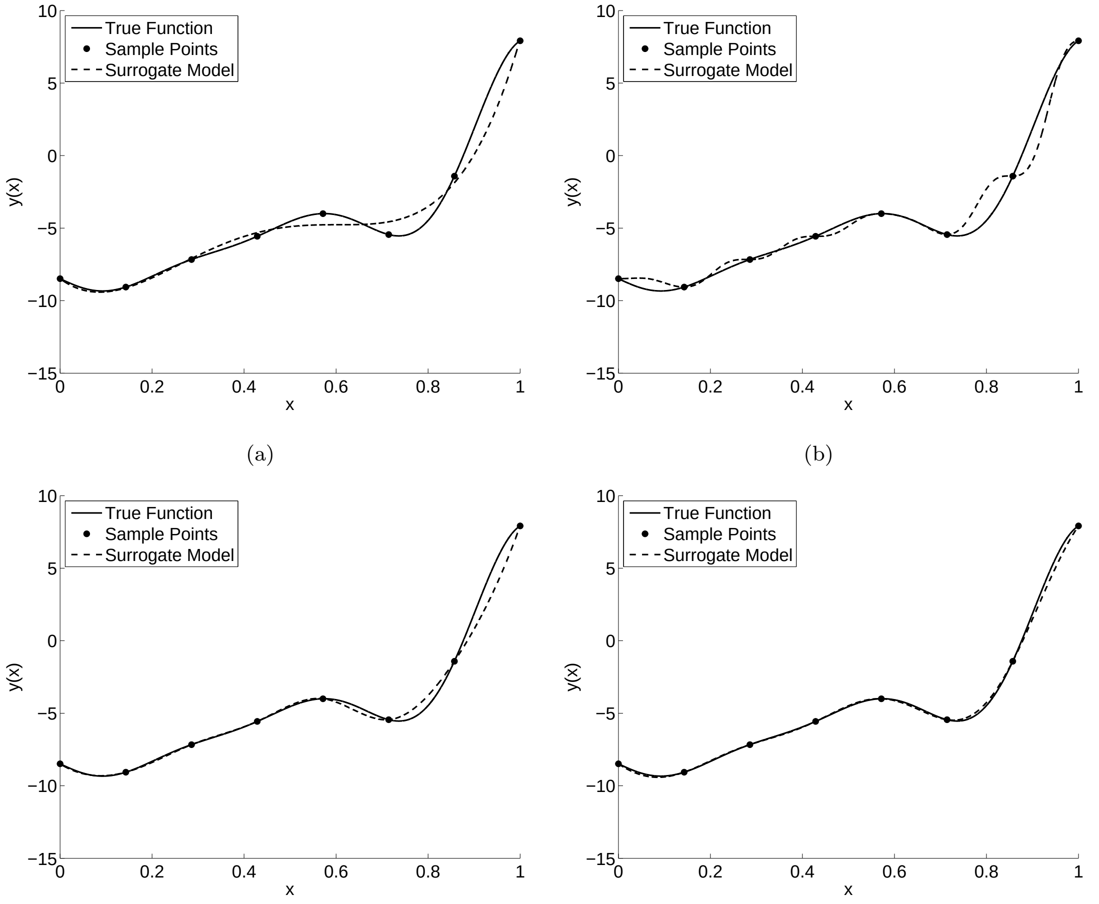


Figure 2.4: Flowchart of the traditional surrogate modelling approach to optimisation.

As with any modelling procedure there will exist inaccuracies, the optimum returned by the stochastic method is therefore unlikely to correspond to the true optimum solution. An updating strategy is then used to increase the accuracy of the model in those regions indicated as optimal. The updated model can then be searched again to provide more points which can be evaluated again using the high fidelity simulation. The process then repeats until a stopping criterion is reached. A number of different updating schemes can be adopted in this process. Updates can, for example, be based on a models prediction of the objective function, the probability of improvement or the expected improvement over the current best sample point, Forrester and Keane [2009].

A typical surrogate modelling optimisation strategy, Figure 2.4, therefore involves an initial sampling of the design space using some sort of sampling plan such as an optimal Latin Hypercube, Morris and Mitchell [1995]. A surrogate model of the response of the objective function to changes in the magnitude of the variables is then constructed from this initial sampling. The surrogate model is then searched for optimal regions using a global optimisation method, such as a genetic algorithm. The searching process can either evaluate the surrogate model's prediction of the objective function, in other words the stochastic method is minimising the objective function, or a statistical quantity related to the surrogate model, such as the maximum likelihood of improvement, Jones [2001]. The update points returned by the global search are then evaluated using the true computational simulation. The true objective function at these points can then be used to update the surrogate model. Continual evaluations of, and updates to, the surrogate model are made until a stopping criterion is reached.



(a)

(b)

(c)

(d)

FIGURE 2.5: Four different surrogate models fit to eight sampling points of the modified Forrester function, (a) 4^{th} order polynomial, (b) Shepard Weighting, (c) cubic spline radial basis function and (d) kriging.

One of the most important factors in any surrogate model based optimisation is the mathematical nature of the model used. There exist a number of different models throughout the literature ranging in complexity from simple polynomials and Shepard weighting, to radial basis functions, support vector regression and krigs, see Figure 2.5.

Each model has a number of associated advantages and disadvantages which affect the accuracy of the model produced and hence the search for a global optimum. For example, modelling a response using a least squares fit quadratic polynomial is a relatively simple process, but a quadratic polynomial will inevitably find it very difficult to model a multimodal shape with any degree of accuracy. A kriging based response surface will more accurately model such multi-modal responses but at an increased modelling cost associated with the selection of hyperparameters.

2.3.2 Kriging

Kriging was first used by geologists to estimate mineral concentrations within a particular region, Krige [1951], and has since been adapted for use in the creation of surrogate models of deterministic computational experiments; a process popularised by Sacks et al. [1989]. Of the numerous types of response surface models, kriging is perhaps one of the most effective due to its ability to model complicated responses through interpolation or regression whilst also providing an error estimate of the predictor. Since its initial application to surrogate modelling, kriging has been applied to a variety of engineering problems including, aerodynamic, Hoyle et al. [2006], Forrester et al. [2006a], structural, Sakata et al. [2003], and multi-objective design problems, D'Angelo and Minisci [2005], Keane [2006].

Given a pair of objective function values, $y(\boldsymbol{x}_i)$ and $y(\boldsymbol{x}_j)$ which are a function of the vectors of design variables $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, of length d, the objective function values will be similar if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are close together within a design space. This can be modelled statistically by assuming that the correlation between two sets of random variables $Y(\boldsymbol{x}_i)$ and $Y(\boldsymbol{x}_j)$ is given by,

$$\mathrm{Corr}\left[Y(\boldsymbol{x}_i), Y(\boldsymbol{x}_j)\right] = \exp\left(-\sum_{l=1}^{d} 10^{\boldsymbol{\theta}_l} \|\boldsymbol{x}_{i_l} - \boldsymbol{x}_{j_l}\|^{\boldsymbol{p}_l}\right) \tag{2.18}$$

Here the hyperparameters $\boldsymbol{\theta}_l$ and $\boldsymbol{p}_l$ determine the rate at which the correlation decreases and the degree of smoothness in the l^{th} coordinate direction, respectively. Consider now a vector $\boldsymbol{y}$ consisting of n objective function values,

$$\boldsymbol{y} = \begin{bmatrix} y(\boldsymbol{x}_1) \\ \vdots \\ y(\boldsymbol{x}_n) \end{bmatrix}, \tag{2.19}$$

where the mean is $\mathbf{1}\hat{\mu}$, and $\mathbf{1}$ is an n by 1 vector of ones. The covariance of $\boldsymbol{y}$ may then be written as

$$\mathrm{Cov}(\boldsymbol{y}) = \sigma^2 \boldsymbol{R}, \tag{2.20}$$

where σ^2 is the variance and the elements of the matrix $\boldsymbol{R}$, the correlation matrix, are given by Equation 2.18. Values of $\boldsymbol{\theta}_l$ and $\boldsymbol{p}_l$ (the hyperparameters), are then chosen to maximize the likelihood on the observed data set $\boldsymbol{y}$. This maximum likelihood function is defined as:

$$\frac{1}{(2n)^{\frac{n}{2}}(\sigma^2)^{\frac{n}{2}}|\boldsymbol{R}|^{\frac{1}{2}}}\exp\left[\frac{-(\boldsymbol{y}-\boldsymbol{1}\hat{\mu})^T\boldsymbol{R}^{-1}(\boldsymbol{y}-\boldsymbol{1}\hat{\mu})}{2\sigma^2}\right], \tag{2.21}$$

or after taking natural logarithms of the function,

$$-\frac{n}{2}\ln(\sigma^2)-\frac{1}{2}\ln(|\boldsymbol{R}|)-\frac{(\boldsymbol{y}-\boldsymbol{1}\hat{\mu})^T\boldsymbol{R}^{-1}(\boldsymbol{y}-\boldsymbol{1}\hat{\mu})}{2\sigma^2}. \tag{2.22}$$

Expressions for the optimal values of the mean

$$\hat{\mu}=\frac{\boldsymbol{1}^T\boldsymbol{R}^{-1}\boldsymbol{y}}{\boldsymbol{1}^T\boldsymbol{R}^{-1}\boldsymbol{1}} \tag{2.23}$$

and variance

$$\hat{\sigma}^2=\frac{1}{n}(\boldsymbol{y}-\boldsymbol{1}\hat{\mu})^T\boldsymbol{R}^{-1}(\boldsymbol{y}-\boldsymbol{1}\hat{\mu}) \tag{2.24}$$

can be found by taking partial derivatives of the log likelihood and equating them to zero. A concentrated likelihood function can then be derived by substituting the expressions for the optimal mean and variance into the log likelihood function,

$$-\frac{n}{2}\ln(\hat{\sigma}^2)-\frac{1}{2}\ln(|\boldsymbol{R}|)-\frac{n}{2}. \tag{2.25}$$

It should be noted that often the constant $-\frac{n}{2}$ is neglected in the calculation of the likelihood. The concentrated likelihood function is dependant only on the correlation matrix and hence on the hyperparameters which are tuned in an attempt to maximise the function.

The selection, or tuning, of these hyperparameters is therefore an optimisation problem in its own right; an optimisation process that can be considerably expensive and complex with the overall cost dependent on both the dimensionality of the optimisation problem and the number of sample points which defines the size of the correlation matrix, $\boldsymbol{R}$.

Park and Baek [2001], for example, note that the Cholesky factorisation, the upper triangular matrix resulting from which is used in the calculation of both the mean and the variance, is the most expensive single operation in the calculation of the concentrated likelihood. The cost of the Cholesky factorisation is of the order $O(n^3)$ and therefore increases dramatically with the number of sampling points used to construct the correlation matrix. Other operations such as the back substitution of the upper triangular matrix and construction of the correlation matrix are of order $O(n^2)$.

Typically the number of points used in a sampling of the objective function is related to the number of dimensions in the problem, for example, Jones et al. [1998] recommend $10d$ sample points in the design of experiments. Observing such rules of thumb can result in relatively expensive calculations of the concentrated likelihood, a process which may have to be repeated many times if a stochastic method is used for hyperparameter optimisation, Keane [2006].

2.3.3 Prediction Using a Kriging Model

Assuming that an appropriate set of hyperparameters have been found it is necessary to "search" the resulting surrogate model for a set of update points. As previously described a global search method is typically employed in such a search. Such optimisation algorithms require a function which is to be minimised in order to define a set of points at which to update the surrogate model. In the case of a surrogate model constructed using a krig, the global optimisation can utilise either the model's prediction of the magnitude of the objective function, the probability of improvement or the expected improvement in the objective function or a combination of such terms.

To make a prediction of the objective function value $y(\boldsymbol{x}^*)$ at a point $\boldsymbol{x}^*$ which is not included in the original sampling, the point $\boldsymbol{x}^*$ is added to the existing data as the $(n+1)^{th}$" observation. An augmented likelihood, Jones [2001], can then be calculated using the hyperparameters defined previously through the optimisation of the concentrated likelihood, Equation 2.25. The surrogate models prediction of $y(\boldsymbol{x}^*)$ is therefore the value which maximises this augmented likelihood.

A vector $\tilde{\boldsymbol{y}}$, is first constructed from the observed set of objective function values and the prediction of the objective function $y(\boldsymbol{x}^*)$,

$$\tilde{\boldsymbol{y}} = [y(\boldsymbol{x}_1), \ldots, y(\boldsymbol{x}_n), y(\boldsymbol{x}^*)]^T . \qquad (2.26)$$

The vector $\boldsymbol{r}$, consists of the correlations of $y(\boldsymbol{x}^*)$ with the points $y(\boldsymbol{x}_i)$ which are calculated using Equation 2.18, using the optimal set of hyperparameters. This vector is then used to construct the augmented correlation matrix,

$$\tilde{\boldsymbol{R}} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{r} \\ \boldsymbol{r}^T & 1 \end{bmatrix} . \qquad (2.27)$$

By substituting this augmented correlation matrix into the previous equation for the log-likelihood, Equation 2.22, the augmented likelihood, which is a function of only $y(\boldsymbol{x}^*)$ is derived,

$$\frac{(\tilde{\boldsymbol{y}} - \mathbf{1}\hat{\mu})\tilde{\boldsymbol{R}}^{-1}(\tilde{\boldsymbol{y}} - \mathbf{1}\hat{\mu})}{2\hat{\sigma}^2}, \qquad (2.28)$$

where $\hat{\mu}$ and $\hat{\sigma}^2$ are known already from Equations 2.23 and 2.24.

The partitioned inverse formula states that given a square matrix $\boldsymbol{A}$ which can be partitioned into two square matrices $\boldsymbol{B}$ and $\boldsymbol{E}$ and two matrices $\boldsymbol{C}$ and $\boldsymbol{D}$,

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{D} & \boldsymbol{E} \end{bmatrix}, \tag{2.29}$$

the inverse of $\boldsymbol{A}$ is given by

$$\boldsymbol{A}^{-1} = \begin{bmatrix} \tilde{\boldsymbol{B}} & \tilde{\boldsymbol{C}} \\ \tilde{\boldsymbol{D}} & \tilde{\boldsymbol{E}} \end{bmatrix}, \tag{2.30}$$

where

$$\begin{aligned} \tilde{\boldsymbol{B}} &= \boldsymbol{B}^{-1} + (\boldsymbol{B}^{-1} \cdot \boldsymbol{C}) \cdot (\boldsymbol{E} - \boldsymbol{D} \cdot \boldsymbol{B}^{-1} \cdot \boldsymbol{C})^{-1} \cdot (\boldsymbol{D} \cdot \boldsymbol{B}^{-1}) \\ \tilde{\boldsymbol{C}} &= -(\boldsymbol{B}^{-1} \cdot \boldsymbol{C}) \cdot (\boldsymbol{E} - \boldsymbol{D} \cdot \boldsymbol{B}^{-1} \cdot \boldsymbol{C})^{-1} \\ \tilde{\boldsymbol{D}} &= -(\boldsymbol{E} - \boldsymbol{D} \cdot \boldsymbol{B}^{-1} \cdot \boldsymbol{C})^{-1} \cdot (\boldsymbol{D} \cdot \boldsymbol{B}^{-1}) \\ \tilde{\boldsymbol{E}} &= (\boldsymbol{E} - \boldsymbol{D} \cdot \boldsymbol{B}^{-1} \cdot \boldsymbol{C})^{-1}. \end{aligned} \tag{2.31}$$

By substituting a partitioned inverse formulation of $\tilde{\boldsymbol{R}}^{-1}$ into Equation 2.28 and taking derivatives with respect to $y(\boldsymbol{x}^*)$ we obtain,

$$\left[\frac{-1}{\hat{\sigma}^2(1 - \boldsymbol{r}^T \boldsymbol{R}^{-1} \boldsymbol{r})}\right] (y(\boldsymbol{x}^*) - \hat{\mu}) + \left[\frac{\boldsymbol{r}^T \boldsymbol{R}^{-1}(\boldsymbol{y} - \boldsymbol{1}\hat{\mu})}{\hat{\sigma}^2(1 - \boldsymbol{r}^T \boldsymbol{R}^{-1} \boldsymbol{r})}\right]. \tag{2.32}$$

When this expression is equated to zero, the basic krig prediction formula is obtained,

$$y(\boldsymbol{x}^*) = \hat{\mu} + \boldsymbol{r}^T \boldsymbol{R}^{-1}(\boldsymbol{y} - \boldsymbol{1}\hat{\mu}). \tag{2.33}$$

A global optimisation algorithm, such as a genetic algorithm, can therefore search the response surface for optimal points using this equation.

2.3.4 Expected Improvement

Unlike using the response surface's prediction of the objective function to locate update points, the expected improvement provides a measure of the improvement in the objective function that may be obtained by sampling at a point.

It has been shown in the literature, Jones [2001], that the estimate of potential error in the kriging predictor is inversely proportional to the curvature, or second derivative, of the augmented likelihood function,

$$s^2(\boldsymbol{x}^*) = \hat{\sigma}^2(1 - \boldsymbol{r}^T \boldsymbol{R}^{-1} \boldsymbol{r}), \tag{2.34}$$

where $s^2(\boldsymbol{x}^*)$ is the error of the predictor at an unknown point $\boldsymbol{x}^*$. If the current best objective function value of the sampled points is denoted as $y_{min}(\boldsymbol{x})$, then the improvement, I, over this value at an unknown point $\boldsymbol{x}^*$ is given by,

$$I = y_{min}(\boldsymbol{x}) - Y(\boldsymbol{x}^*). \tag{2.35}$$

The likelihood of actually achieving this improvement can be calculated using the normal density function, Jones [2001],

$$\frac{1}{\sqrt{2\pi s^2(\boldsymbol{x}^*)}} \exp\left[-\frac{(y_{min}(\boldsymbol{x}) - I - \hat{y}(\boldsymbol{x}^*))^2}{2s^2(\boldsymbol{x}^*)}\right]. \tag{2.36}$$

The expected value of the improvement, $E(I)$, is then calculated by integrating over the normal density function between the bounds of $I = 0$ and $I = \infty$,

$$E(I) = \int_{I=0}^{I=\infty} I \left\{ \frac{1}{\sqrt{2\pi s^2(\boldsymbol{x}^*)}} \exp\left[-\frac{(y_{min}(\boldsymbol{x}) - I - \hat{y}(\boldsymbol{x}^*))^2}{2s^2(\boldsymbol{x}^*)}\right] \right\} \mathrm{d}I. \tag{2.37}$$

The variable, $\hat{y}(\boldsymbol{x}^*)$ denotes the most likely value of the objective function at $\boldsymbol{x}^*$, and is calculated via Equation 2.33. Equation 2.37 can be integrated by parts to give,

$$E(I) = s(\boldsymbol{x}^*)\left[u\Phi(u) + \phi(u)\right], \tag{2.38}$$

where

$$u = \frac{y_{min} - \hat{y}(\boldsymbol{x}^*)}{s(\boldsymbol{x}^*)} \tag{2.39}$$

and Φ and ϕ represent the normal cumulative distribution function and density function respectively, Jones [2001]. Equation 2.38 can then be calculated using the error function as,

$$\begin{aligned} E(I) = (y_{min} - \hat{y}(\boldsymbol{x}^*)) &\left[\frac{1}{2} + \frac{1}{2}\mathrm{erf}\left(\frac{y_{min} - \hat{y}(\boldsymbol{x}^*)}{s(\boldsymbol{x}^*)\sqrt{2}}\right)\right] \\ &+ \frac{s(\boldsymbol{x}^*)}{\sqrt{2\pi}} \exp\left[\frac{-(y_{min} - \hat{y}(\boldsymbol{x}^*))^2}{2s^2(\boldsymbol{x}^*)}\right]. \end{aligned} \tag{2.40}$$

Using this formulation of the expected improvement, in conjunction with a stochastic optimisation method, produces a series of update points which are most likely to result in an improvement upon the current best objective function. This technique has been successfully used throughout the literature in the optimisation of aerodynamic shapes, from aerofoils, Keane [2006] to engine intakes, Hoyle et al. [2006].

2.3.5 Regression of Kriging Models

Given the potentially "noisy" nature of many computational simulations which depend on discretisation and iterative solutions, it is often extremely important to employ regression in the construction of a surrogate model, Forrester et al. [2006b]. Whilst kriging is often set up as an interpolating model, a regressing model can be constructed such that the sampled points no longer have an exact correlation with the resulting model. A regression constant, λ, is added to the diagonal of the correlation matrix $\boldsymbol{R}$ producing $\boldsymbol{R}+\lambda\boldsymbol{I}$. The magnitude of this constant is another hyperparameter varied in the tuning process where the optimal mean becomes

$$\hat{\mu}_r = \frac{\mathbf{1}^T(\boldsymbol{R}+\lambda\boldsymbol{I})^{-1}\boldsymbol{y}}{\mathbf{1}^T(\boldsymbol{R}+\lambda\boldsymbol{I})^{-1}\mathbf{1}}, \tag{2.41}$$

and the variance,

$$\hat{\sigma}_r^2 = \frac{1}{n}(\boldsymbol{y}-\mathbf{1}\hat{\mu}_r)^T(\boldsymbol{R}+\lambda\boldsymbol{I})^{-1}(\boldsymbol{y}-\mathbf{1}\hat{\mu}_r), \tag{2.42}$$

with the concentrated likelihood function given by,

$$-\frac{n}{2}\ln(\hat{\sigma}_r^2) - \frac{1}{2}\ln(|\boldsymbol{R}+\lambda\boldsymbol{I}|). \tag{2.43}$$

Upon completion of the tuning process, the resulting surrogate model can be searched in the normal fashion to locate regions of optimal design using either the model's prediction of the objective function or the expected improvement.

The kriging predictor of Equation 2.33 can be easily modified to include the regression term resulting in,

$$y(\boldsymbol{x}^*) = \hat{\mu}_r + \boldsymbol{r}^T(\boldsymbol{R}+\lambda\boldsymbol{I})^{-1}(\boldsymbol{y}-\mathbf{1}\hat{\mu}_r). \tag{2.44}$$

When regression is used in the construction of a surrogate model, the model no longer interpolates the sampled points. The expected improvement is therefore no longer zero at these sample points. When a series of deterministic computational experiments are used to construct the surrogate model this non-zero expected improvement is incorrect as resampling a previous point will result in an identical objective function value.

The expected improvement must therefore be adjusted to assume a zero value at sampled points. To achieve this, a "reinterpolation" procedure was introduced by Forrester et al. [2006b], resulting in a new expression for the estimate of the variance,

$$\hat{\sigma}_{ri}^2 = \frac{1}{n}(\boldsymbol{y}-\mathbf{1}\hat{\mu})^T(\boldsymbol{R}+\lambda\boldsymbol{I})^{-1}\boldsymbol{R}(\boldsymbol{R}+\lambda\boldsymbol{I})^{-1}(\boldsymbol{y}-\mathbf{1}\hat{\mu}). \tag{2.45}$$

This variance can then be used to calculate the error in the predictor, $s^2(\boldsymbol{x}^*)$, at an unknown point using Equation 2.34. The expected improvement can therefore be calculated as normal and used to provide update points for the optimisation. This technique was successfully employed by Forrester et al. [2006b] in the optimisation of a transonic aerofoil.

2.4 The Curse of Dimensionality

A global optimisation strategy should eventually attain the solution to a problem if afforded enough time. Unfortunately the real world applications of optimisation strategies are within a more restrictive environment. Deadlines and targets must be met due to the integration of the optimisation within a much larger design process. This imposed time limit may have a serious impact on the performance of any resulting design as a time limit generally translates into a restricted number of function evaluations. Without careful consideration of the optimisation strategy employed, and the parameterisation chosen, an optimisation may therefore fall foul of the “curse of dimensionality”.

The so called “curse of dimensionality” is essentially a law of diminishing returns. As the dimensionality of a design space increases so too does the complexity of the optimisation. With increasing dimensionality a larger effort must be spent to effectively explore and then exploit regions within the design space. Given a restricted optimisation budget this becomes increasingly difficult with the increased flexibility introduced via an increase in dimensionality effectively hampering the optimisation process.

Consider Figure 2.6, here one can observe that as the dimensionality of the problem is increased so too does the quality of the final design. In an aerodynamic optimisation, for example, this could be the result of an increased flexibility of the shape parameterisation which expands the design space to include better designs. Given a restricted simulation budget a designer may only be able to effectively optimise a small number of variables. As problem dimensionality increases the optimisation procedure becomes more unreliable producing an increasingly varying range of “optimum” designs. Eventually dimensionality becomes such an issue that an optimisation strategy may, on average, produce poorer designs than at lower dimensions even though the increased flexibility can theoretically produce a better design. Increasing the available budget of function evaluations naturally has the effect of allowing the same optimisation strategy to explore larger design spaces more effectively, pushing this threshold in performance to the right.

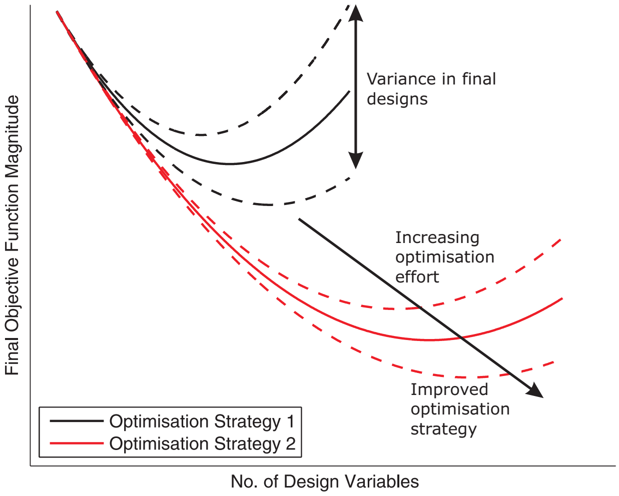

Variance in final designs
Increasing optimisation effort
Improved optimisation strategy
Final Objective Function Magnitude
No. of Design Variables
Optimisation Strategy 1
Optimisation Strategy 2

Figure 2.6: A graphical representation of how an increase in optimisation effort or a better optimisation strategy can stave of the impact of increasing dimensionality.

Altering the optimisation strategy may have a similar effect. The optimisation of a relatively low dimensional problem using a stochastic method, for example, could be completed in a reasonable number of evaluations. However, applying the same evaluation budget to a higher dimensional problem may result in the stochastic method performing poorly. Alter the optimisation method to one more suited to coping with a restricted budget, such as a response surface based technique, and the resulting designs at higher dimensions may improve.

It's therefore of no surprise that dimensionality is at the cornerstone of optimisation algorithm development. All algorithms are developed in an attempt to push this performance envelope and produce better results for lower cost. To this end numerous technologies have been developed over the years, from the local, stochastic and response surface based methods mentioned previously to multi-fidelity optimisation in the form of co-kriging and the plethora of variable screening techniques. Each attempts to produce a better solution to an optimisation problem for a reduced cost.

Given a restriction on the total time for an optimisation a designer must therefore carefully select an appropriate optimisation method and an appropriate parameterisation, a task which can require a great deal of experience and knowledge of the problem at hand and of each optimisation technique's advantages and disadvantages. The following chapters of this thesis attempt to address some of these issues with regard to the use of kriging in aerodynamic design. The overall aim is to minimise the expense of kriging hyperparameter selection and improve the performance of the algorithm at higher dimensions. Remove the bottle neck of hyperparameter tuning from an optimisation and the optimisation can be completed faster or indeed more time can be afforded to actual function evaluations. Improve the performance at higher dimensions and there is no longer such a severe restriction to the selection of the design variables.

Chapter 3

Hyperparameter Tuning Within a Surrogate Modelling Strategy

3.1 Introduction

Although the performance of kriging has been extensively compared to that of other surrogate models, Jin et al. [2001], Jones [2001], Simpson et al. [2001], Won and Ray [2004] and a number of different tuning techniques have been compared, Hollingsworth and Mavris [2003], there is little information in the literature on the effect the degree of hyperparameter tuning has on an optimisation process.

Throughout the literature it is consistently observed that the time spent tuning the hyperparameters of a kriging model can be quite significant and, for high dimensional problems with many data points, comparable to the cost of the high-fidelity simulations used to obtain the objective function values. Hyperparameter tuning can be especially costly if an updating process is used to improve the surrogate model, as in Keane [2006]. Tuning of the hyperparameters effectively becomes a bottleneck in the optimisation process, introducing a significant delay before new updates can be evaluated.

In an attempt to illuminate the issue of hyperparameter tuning, the performance of five different tuning strategies are assessed using the inverse design of the supercritical RAE-2822 aerofoil, Cook et al. [1979], as a test problem. The geometry parameterisation employed in this design problem allows a systematic increase in the number of variables used in the optimisation process whilst maintaining continuity between the geometries as the number of variables is changed.

Each of the five hyperparameter tuning strategies is assessed within a benchmarking framework. This framework remains consistent throughout, ensuring that each optimisation begins with an identical design of experiments, and sufficient optimisations are completed for the purposes of averaging.

3.2 Optimisation Benchmark Framework

The benchmarking procedure employs the Options design exploration system, Keane [2003], and at its core consists of the basic response surface construction and updating process discussed previously, Section 2.3.1. The benchmarking procedure changes the number of variables and repeats each optimisation a specified number of times to best judge the performance of a particular response surface based optimisation strategy (in this case the performance of five different hyperparameter tuning strategies).

Figure 3.1 illustrates the main processes involved in the benchmarking procedure. The initial setup of the procedure requires a few key pieces of information: the design of experiment (DOE) size, the number of update cycles and the maximum number of points in each cycle. Further information specific to the optimisation strategy is also required, such as the method used to construct or tune the response surface as well as the method of searching the response surface.

The complete algorithm consists of three nested loops. The inner loop (denoted loop number one in Figure 3.1) represents the typical activity of a designer employing a surrogate based optimisation. Results from an initial design of experiments are used to generate a response surface which is then searched to provide the specified number of update points. The variables from these update points are then used to carry out more computational experiments, in this case computational fluid dynamics (CFD) simulations. The results from these numerical simulations are used to construct a new response surface which is searched again to obtain a further set of update points if required. This process is continued until a specified number of update cycles has been completed.

After storing the best design obtained, the entire optimisation process, (loop number 1), is repeated using a DOE generated from a different random number seed, (loop number two in Figure 3.1). This prevents the "getting lucky" scenario whereby a point in the initial DOE falls on or near the global optimum. By carrying out a number of optimisations for the same problem, (in terms of the number of variables), a statistical analysis of the results can be obtained which indicates the performance of the optimisation methodology. The number of optimisations carried out depends on the available time for the benchmark. However, increasing the number of optimisations increases the confidence of the statistical analysis. The final outermost loop, (loop number three in Figure 3.1), repeats the entire process but changes the number of variables that the optimisation uses, thus assessing the ability of an optimisation methodology to cope with problems of varying complexity.

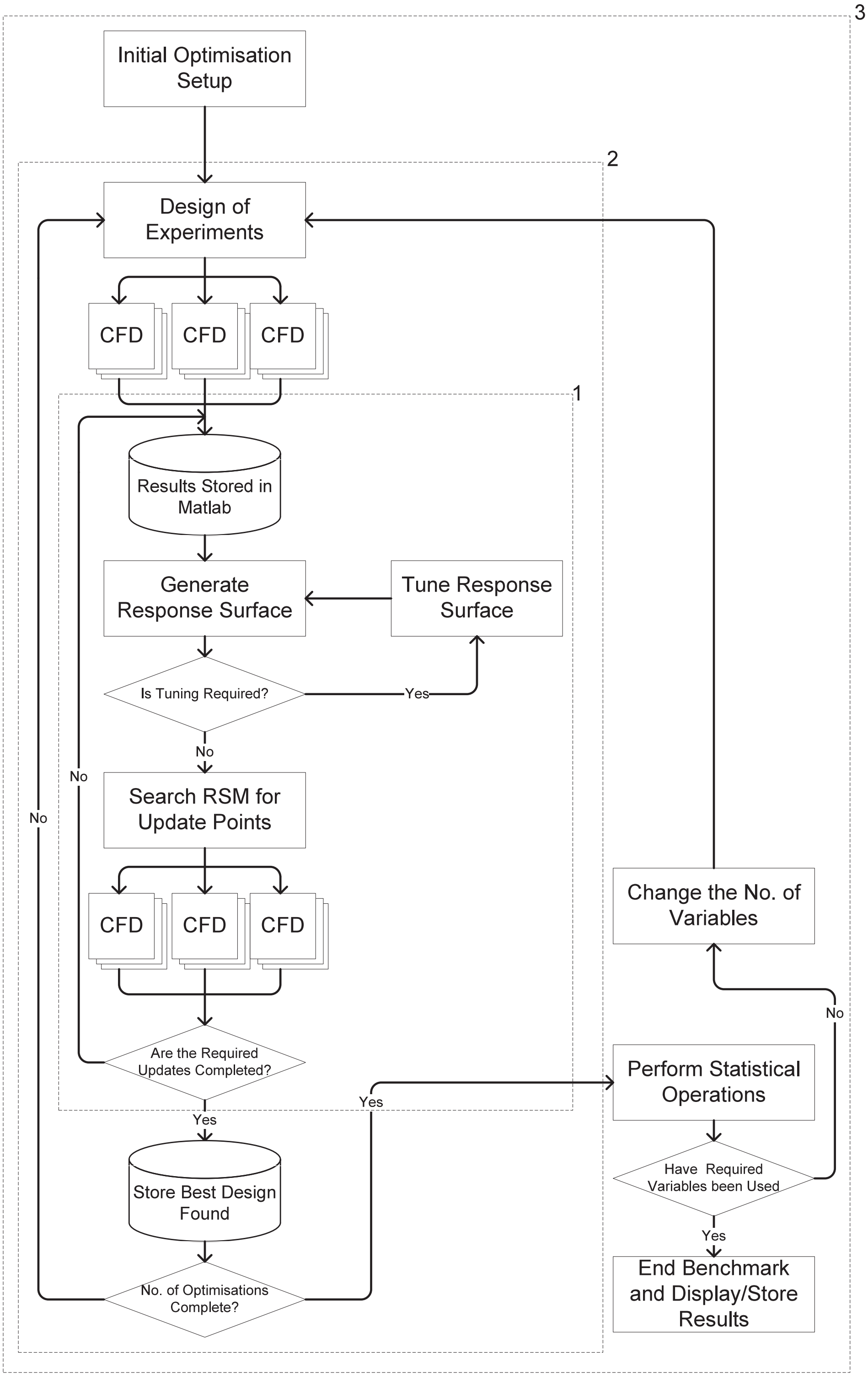

3
Initial Optimisation Setup
2
Design of Experiments
CFD
CFD
CFD
1
Results Stored in Matlab
Generate Response Surface
Tune Response Surface
Is Tuning Required?
Yes
No
Search RSM for Update Points
CFD
CFD
CFD
No
No
Are the Required Updates Completed?
Yes
Yes
Store Best Design Found
No. of Optimisations Complete?
Change the No. of Variables
No
Perform Statistical Operations
Have Required Variables been Used
Yes
End Benchmark and Display/Store Results

FIGURE 3.1: Overview of the benchmark procedure.

The structure of this benchmark procedure allows it to be used to investigate many aspects of an optimisation methodology, or indeed different optimisation methodologies and not just the performance of the tuning strategies studied within this thesis. Although an aerodynamic optimisation problem is considered here, the benchmark procedure could be just as easily applied to any other appropriate optimisation problem.

3.3 Hyperparameter Tuning Strategies

A total of five different tuning strategies, (summarized in Table 3.1), are investigated using a low fidelity inverse design problem. The five strategies each employ a simplex search, followed by a genetic algorithm (GA) and finally a dynamic hill climb (DHC) to minimise the negative of the concentrated likelihood function, Equation 2.25.

The simplex search, using the Amoeba algorithm, Press et al. [1986], commences from a random set of hyperparameters and continues until one of two stopping criterion is reached; a predefined tolerance in the concentrated likelihood or until $50d$ iterations have been completed. Hyperparameters defining the optimum point are then perturbed by up to 10% using a random number and the simplex search repeated.

The result of the simplex search is then included in the initial population of the GA with the best design of the GA then used as the starting point of the DHC. Although a GA is a popular method of global optimisation (and has been shown by Hollingsworth and Mavris [2003] to be an effective and reliable method of tuning hyperparameters) the optimal point predicted can be inaccurate due to the GA's poor performance in final convergence. A hill climbing algorithm (in this case the DHC) is therefore often used to refine the optima predicted by the GA.

	Strategy	Description of Tuning Strategy
1	Heavy Tune	5,000 iteration genetic algorithm & 5,000 iteration dynamic hill climb after the DOE and each update
2	Light Tune	1,000 iteration genetic algorithm & 1,000 iteration dynamic hill climb after the DOE and each update
3	Single Tune	5,000 iteration genetic algorithm & 5,000 iteration dynamic hill climb after the DOE only
4	Alternate Tune	5,000 iteration genetic algorithm & 5,000 iteration dynamic hill climb after the DOE and alternate updates
5	θ Tune	5,000 iteration genetic algorithm & 5,000 iteration dynamic hill climb after the DOE and each update, optimizing for single common value of θ, ($p = 2$ and $\lambda = 10^{-6}$)

Table 3.1: Summary of Tuning Strategies all of which commence from the result of a two stage simplex search.

Each of the five strategies differs in the degree of tuning used after each update. In both the "heavy" and "light" tuning strategies, the hyper-parameters are tuned after the initial DOE and after every single set of updates to the model. The only difference is in the degree of tuning effort in each case. The heavy tune consists of 5,000 evaluations of the concentrated likelihood function using a GA, (100 generations with a population size of 50), followed by a further 5,000 evaluations using a DHC. The light tune uses only 1,000 evaluations of each. The third strategy consists of a single heavy tune after the initial DOE followed by no further tuning after subsequent updates. The fourth strategy consists of a heavy tune after the initial DOE and after each alternate batch of updates. The fifth and final strategy involves the tuning of a single value of θ which is assumed to be the same for every variable with the remaining hyperparameters assumed to be constant, ($p = 2$ and $\lambda = 10^{-6}$).

Two different computational budgets are used in the investigation, consisting of a total of 75 and 150 CFD evaluations. In accordance with the work of Sóbester et al. [2005], each of these budgets has one third of the evaluations reserved for the initial design of experiments with the remainder used in the update process. A maximum of 10 CFD evaluations are permitted for each update cycle. The small budget strategy therefore consists of a DOE of 25 evaluations followed by five update cycles of up to 10 evaluations each. It must be noted that the budget of 10 CFD evaluations per update cycle may not be completely used in each cycle. The number of update points returned by the GA's search of the krigs prediction of the objective function depends on the modality of the response surface generated; a highly modal surface may return the maximum 10 update points while a surface with fewer undulations may return fewer update points. Therefore, the available budgets of 75 and 150 CFD evaluations may not be completely used in each optimisation.

The combination of different computational budgets and tuning strategies results in a total of 10 optimisation strategies. Each strategy is applied to a series of different optimisation problems of varying complexity, achieved by altering the number of variables within the design problem. Each optimisation is carried out a number of times with differing random number seeds used to construct the Latin hypercube of the DOE. As the random number seeding is consistent for all of the tuning strategies, the CFD evaluations for each design of experiment only have to be calculated once. Each tuning strategy uses the same initial set of data to tune the initial surrogate model, and thus reduces some of the computational expense of the investigation whilst providing a more meaningful comparison between the results of the different strategies.

3.4 Aerofoil Inverse Design Problem

The inverse design problem investigated here involves the modification of a baseline aerofoil, the NACA 0012, through the addition of multiple Hicks-Henne bump functions, Hicks and Henne [1978], to the upper and lower aerofoil surfaces in an attempt to recreate the surface pressure distribution of the RAE-2822 aerofoil. A series of Hicks-Henne bump functions, described by

$$y = A\left[\sin\left(\pi x^{\frac{\ln 0.5}{\ln x_p}}\right)\right]^t \qquad x \in [0,1], \tag{3.1}$$

where the parameters A, x_p and t denote the amplitude, position of the maximum and sharpness of the bump respectively, are applied to each surface through addition of the z-coordinates. These Hicks-Henne control parameters form the variables in the optimisation process, and are allowed to vary according to the limits in Table 3.2. An example of a perturbation of the baseline geometry through addition of multiple Hicks-Henne functions is presented in Figure 3.2.

Parameter	Lower Limit	Upper Limit
A (x/c)	-0.02	0.02
x_p (x/c)	0.01	0.95
t	1	6

Table 3.2: Inverse design Hicks-Henne function parameter limits.

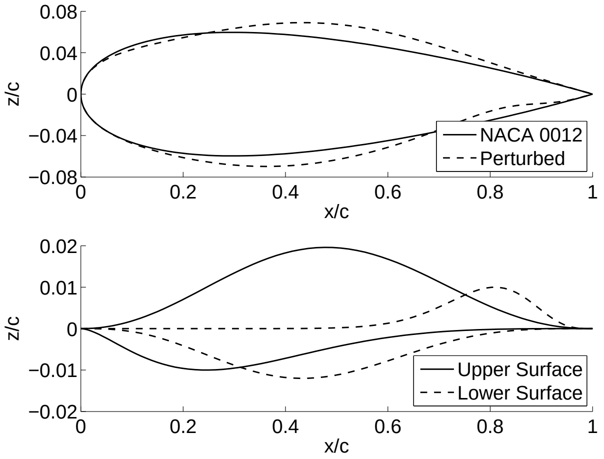


Figure 3.2: An example of the NACA 0012 aerofoil perturbed through the addition of four Hicks-Henne functions.

Through manipulation of these parameters and through addition of multiple bump functions to the aerofoil, the geometry parameterisation can employ any number of variables. Increasing the number of variables increases the range of geometries which the parameterisation can represent while simultaneously increasing the complexity of the optimisation. When increasing the number of variables, additional bump functions are applied to the upper and lower surfaces of the baseline NACA-0012 aerofoil in an alternate manner, with the upper surface bump applied first.

The parameters describing each Hicks-Henne function are designated as variables in a specific order. First the amplitude of the function is defined as a variable, then the location of the maximum, followed by the sharpness. If only a proportion of the parameters are required to be designated as variables the remaining parameters are kept constant with $x_p = 0.47$ and $t = 3.5$, corresponding to halfway between the limits of Table 3.2. A two variable problem, for example, consists of a single bump function applied to the upper surface with the parameters A and x_p permitted to vary and t equal to 3.5.

Identical aerofoil geometries can be produced with a different set of variables resulting in a design space containing multiple minima. As the parameter limits are identical for each of the applied functions, it is possible to swap the parameters and obtain the same geometry. Consider the example aerofoil in Figure 3.2 where two bumps have been applied to the upper surface, the first subtracts a portion of the base aerofoil from the leading edge, while the second thickens the aerofoil at the mid-camber point. The variables defining the first bump could be swapped with those of the second resulting in an identical aerofoil. The presence of multiple minima deliberately increases the difficulty of the optimisation problem in a similar manner to other test functions such as the Rastrigin function or the Keane bump function, Keane and Nair [2005].

The parameter limits of each function permit a design produced using one parameterisation to be recreated using a more complex parameterisation. For example, any geometry resulting from the one variable parameterisation can be reproduced using the two variable parameterisation as the default position of the maximum of the bump function can be replicated. The one variable design space is therefore a line through the two variable design space of constant maximum location, ($x_p = 0.47$). The two variable parameterisation is itself a plane through the three variable parameterisation of constant sharpness, ($t = 3.5$) and so forth. The continuity between the geometry parameterisations provides for a useful comparison between the results of the benchmark optimisations.

The application of the Hicks-Henne functions in the above manner provides an interesting and rather difficult multi-modal problem, one which could be simplified by the careful consideration of the parameterisation so as to reduce the multi-modality which has been artificially introduced here. Nevertheless the multi-modality and multiple global minima at higher dimensions present an excellent test of the overall kriging strategy and the associated tuning techniques investigated in this chapter

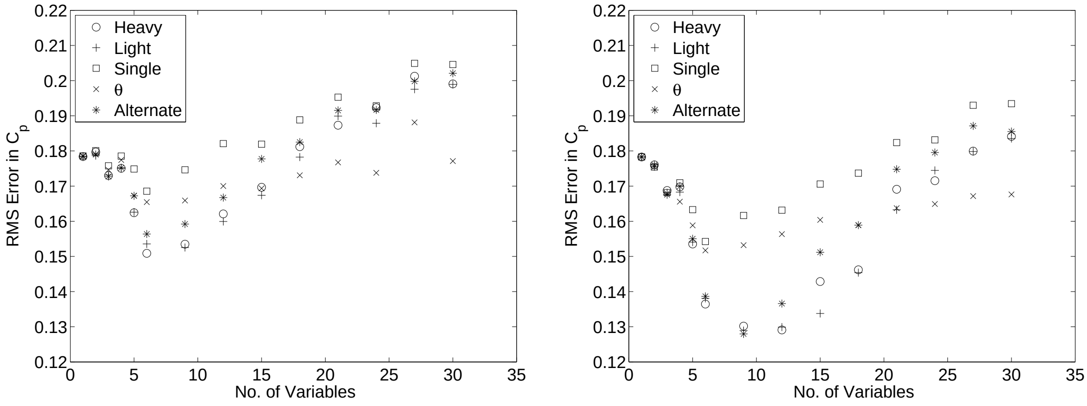
The surface pressure distribution over each aerofoil is calculated using the full potential code VGK, developed by the Engineering Sciences Data Unit, ESDU [1996b], at an angle of attack of 2°, Mach 0.725 and a Reynolds number of 6.5×10^6. This pressure distribution is then compared to that of the RAE-2822, (also computed using VGK), and the root mean squared error between the two distributions is calculated. The error between the pressure distributions then forms the objective function in the optimisation process. The surrogate model therefore models the response of the error in pressure distribution to changes in the magnitude of the Hicks-Henne function parameters for each bump function applied to the baseline aerofoil. Although Figure 3.1 implies the use of parallel CFD, in this case the objective function is evaluated in serial. Due to the speed of VGK, approximately 1 second per evaluation, there is little time penalty due to a serial implementation. However, a parallel implementation would be advised for more computationally expensive simulations.

3.5 Comparison of Optimisation Results

3.5.1 Overview of the Results

The inverse aerofoil design problem has been investigated using both of the computational budgets previously outlined, for each of the five tuning strategies on a total of 14 increasingly complex geometry parameterisations. The parameterisations ranged from a simple one variable problem to a difficult 30 variable problem, where the initial aerofoil geometry was altered using a total of 10 Hicks-Henne functions. Each of the 14 optimisation problems were carried out a total of 50 times, varying the Latin hypercube used to select the points in the DOE each time.

The mean objective function values obtained for each tuning strategy, along with the standard deviation of the objective functions, are presented in the graphs of Appendix A for the 75 evaluation budget and for the 150 evaluation budget but the mean results of each optimisation are presented below in Figure 3.3.



(a)

(b)

FIGURE 3.3: Average objective function values obtained for each of the five tuning strategies for problems of varying complexity using (a) a budget of 75 simulations and (b) a budget of 150 simulations.

The overall accuracy of the surrogate model used to find the final set of update points has been assessed through the calculation of the r^2 correlation. The r^2 correlation compares the true objective function values at 250 design points to those predicted by the surrogate model. The average r^2 correlations for three different geometry parameterisations are presented in Table 3.3 with a value close to one indicating a high correlation between the true objective function and that predicted by the surrogate model.

	75 Simulation Budget			150 Simulation Budget		
Strategy	3 Variables	6 Variables	9 Variables	3 Variables	6 Variables	9 Variables
Heavy	0.731	0.536	0.125	0.854	0.715	0.257
Light	0.760	0.563	0.145	0.852	0.708	0.228
Single	0.568	0.140	0.013	0.826	0.335	0.024
Alternate	0.755	0.556	0.118	0.854	0.716	0.250
θ	0.609	0.214	0.077	0.730	0.327	0.108

Table 3.3: Average r^2 correlation of the final surrogate model for three geometry parameterisations.

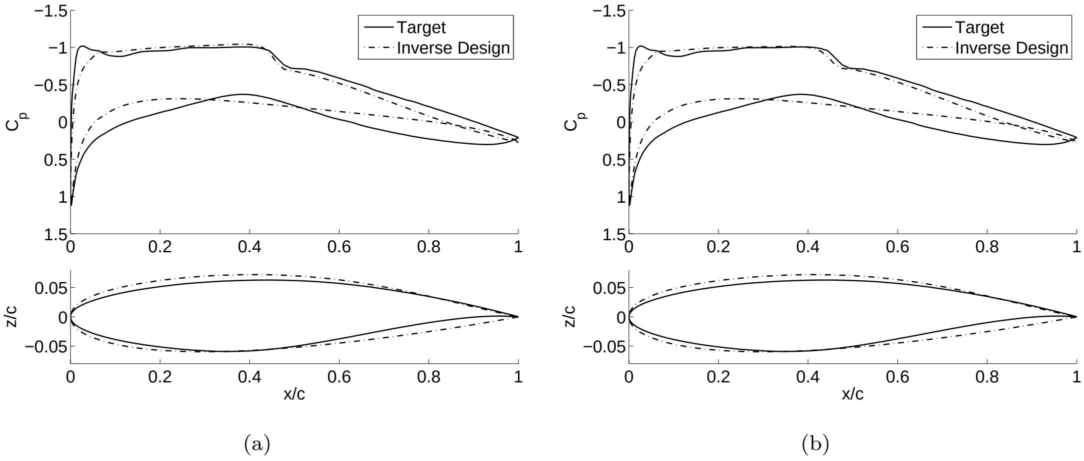
3.5.2 The Implications of Dimensionality

As described in Section 2.4, each of the five tuning strategies displays the adverse affect of increasing the dimensionality of the design problem. Given a fixed simulation budget, as the dimensionality of the problem increases, the optimisation processes are capable of finding increasingly better designs until the dimensionality reaches a certain point. Beyond this threshold, (indicated by the dotted lines in the graphs of Appendix A), the problem becomes too complex for the given optimisation strategy to adequately search. Therefore, as the complexity of the design problem increases further, the optimisation strategy generally produces "good" designs with objective functions worse than designs obtained with less complex parameterisations despite the fact that such parameterisations are guaranteed to be capable of producing results at least as good as the simpler parameterisation.

Consider the heavy tuning strategy with the small simulation budget Figure 3.3(a), as the complexity of the geometry parameterisation increases from one to six variables the strategy is able to improve the mean objective function value. The additional flexibility introduced by the increased number of variables allows the target pressure distribution of the RAE-2822 aerofoil to be more closely attained. When the parameterisation consists of more than six variables the given budget is no longer sufficient to search the design space resulting in increasingly suboptimal designs.

Due to the continuity between the geometry parameterisations, even if the increase in complexity of the parameterisation produces a design offering no improvement over a simpler parameterisation, an optimisation should at the very least produce an identical design. The inability of the optimisation strategy to regularly obtain this design is therefore a direct result of the increase in the dimensionality of the problem.

Increasing the size of the computational budget, as seen in Figure 3.3(b), has the effect of delaying this degradation in performance. The heavy tuning strategy with the 150 simulation budget generally continues to improve the mean objective function up to a 12 variable design problem, after which the performance of this strategy is also degraded by increasing dimensionality.

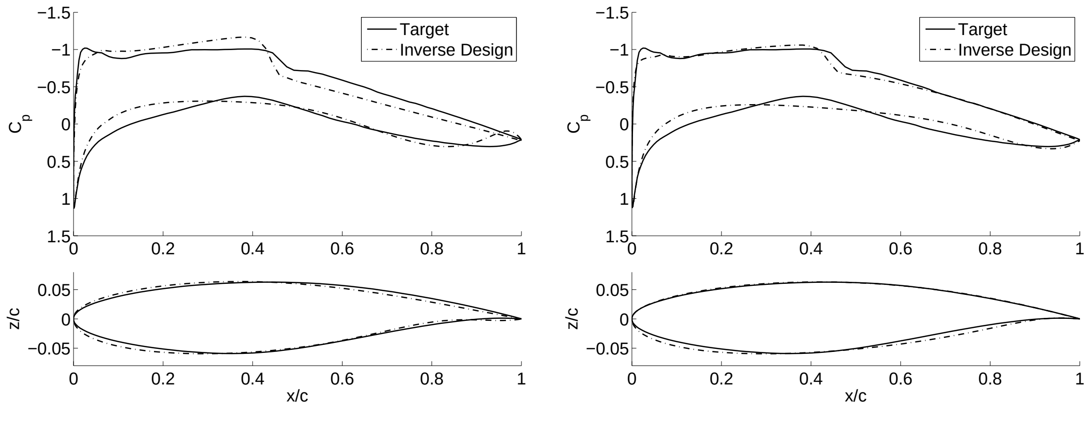


(a)

(b)

FIGURE 3.4: Best aerofoil design obtained with the three variable optimisation using the (a) 75 simulation budget and (b) 150 simulation budget.

The effect of the simulation budget on the search for an optimum design is further emphasised in Figures 3.4 and 3.5, which show the best overall design obtained using both of the simulation budgets for the three and nine variable problems, respectively. Both simulation budgets produce similar designs for the three variable problem, although the design resulting from the larger budget is marginally better giving a root mean square (RMS) error of 0.1608 compared to 0.1622 for the smaller budget. Observe that the upper surface pressure distribution upstream of the shockwave obtained using the larger simulation is very slightly closer to the target pressure.



(a)

(b)

Figure 3.5: Best aerofoil design obtained with the nine variable optimisation using the (a) 75 simulation budget and (b) 150 simulation budget.

When the complexity of the optimisation problem is increased to nine variables, consisting of two Hicks-Henne functions on the upper surface and one on the lower, the effect of the computational budget is much more apparent. There is a significantly greater difference in the best designs, with the larger budget producing a RMS error of 0.076 and the smaller budget producing an error of 0.1138. This difference is demonstrated graphically in Figure 3.5. The best design obtained with the large budget produces a pressure distribution over the upper surface which matches the target pressure much more closely; only the region around the shockwave and the suction peak over the leading edge fail to be accurately reproduced.

The lower surface pressures differ significantly from the target, with the exception of a region close to the trailing edge for the larger budget optimised geometry. This inaccuracy is mainly due to the inadequacies of the lower surface geometry parameterisation; the application of a single bump function to the lower surface of the baseline aerofoil allows the lower surface geometry to be adjusted locally in only one region. To obtain a more accurate reproduction of the RAE-2822 aerofoil requires the modification of the baseline aerofoil close to the leading and trailing edges simultaneously, with little modification in between. Introducing a second bump to the lower surface reduces the objective function further, as one can observe in Figure 3.3(b). However, the increased complexity of the design problem has reduced the ability of the optimisation to search effectively resulting in only a slight improvement over the nine variable problem.

The average r^2 correlations, presented in Table 3.3, reinforce both the negative effect of increasing problem dimensionality given a fixed simulation budget and the positive impact of increasing the size of the simulation budget on the overall accuracy of the surrogate model. As the dimensionality of the optimisation problem increases the available budget becomes increasingly incapable of accurately representing the global response of the objective function (demonstrated by the decrease in the average r^2 correlation).

3.5.3 Heavy & Light Tuning Strategies

The results presented in Figure 3.3 demonstrate only a marginal difference between the heavy and light tuning strategies for those optimisations where the simulation budget could be considered adequate to search the design space i.e. up to the six variable problem for the small budget and the twelve variable problem for the large budget. The mean objective functions and even the standard deviations are approximately equal and the r^2 correlation results, (Table 3.3), indicate little difference in the global accuracy of the surrogate models produced using both tuning strategies. This indicates that the hyperparameters obtained after tuning are similar, and, as a result, the optimisation process searches similar response surfaces. A similar response surface means that the results of each optimisation, and hence the resulting statistics closely correspond. The presented results therefore indicate that the light tuning strategy could be considered sufficient to tune the hyperparameters of the response surfaces constructed for these optimisations.

3.5.4 Tuning Hyperparameters Only Once

The results for very low dimensional problems differ only marginally from the results of the heavy and light tuning strategies when the hyperparameters are tuned only once after the initial design of experiments. However, this difference grows substantially as the complexity of the optimisation problem increases. The mean objective function obtained for the 12 variable problem using the single tuning strategy and large budget is approximately 25% worse than that obtained with the heavy tuning strategy. The performance of the single tuning strategy, as with each of the other tuning strategies, is improved to some degree with the increase of the simulation budget, as shown in Figure 3.3(b).

The ability of the single tuning strategy to produce results comparable to that of the heavy and light tuning strategy for simple problems is due to the ability of the design of experiments to adequately seed the design space. The number of points in the initial DOE is such that the hyperparameters resulting from the initial tuning process accurately capture the correlation between points and the trends of the true response. This is confirmed when one considers the r^2 correlation results for the three variable problem, which are reasonably close to that of the heavy and light tuning strategy. With an initial accurate set of hyperparameters the global exploration of the response surface can therefore find basins of optimal design easily.

When the complexity of the design problem is increased the number of points in the design of experiments cannot produce such an accurate representation of the true response surface (again confirmed by the r^2 correlations of Table 3.3). Subsequent update points are essentially wasted because without further tuning of the hyperparameters the correlation, smoothness and even the degree of regression used in the construction of the response surface cannot be updated and corrected. Consequently, the genetic algorithm finds basins of optima on an inaccurate response surface and the updates are unable to improve upon the current best design. The best designs of such an optimisation are therefore worse than those where the hyperparameters are continually reassessed.

The improvement of the single tune strategy as the simulation budget is increased is therefore a result of the greater number of points making up the initial design of experiments. Table 3.3 indicates that the inclusion of a larger number of points results in a more accurate set of hyperparameters from the initial tune producing a better representation of the true surface.

Given the increased performance of the optimisation strategy with an increase in the size of the design of experiments, it stands to reason that by altering the ratio of the total simulation budget used in the initial DOE the performance of this tuning strategy can be improved. In addition to the results presented above, three of the design problems were chosen and optimised using four additional DOE sizes. Table 3.4 presents the results of this additional investigation including the original results using the DOE simulation budget of 50. The presented results have once again been averaged over 50 optimisations.

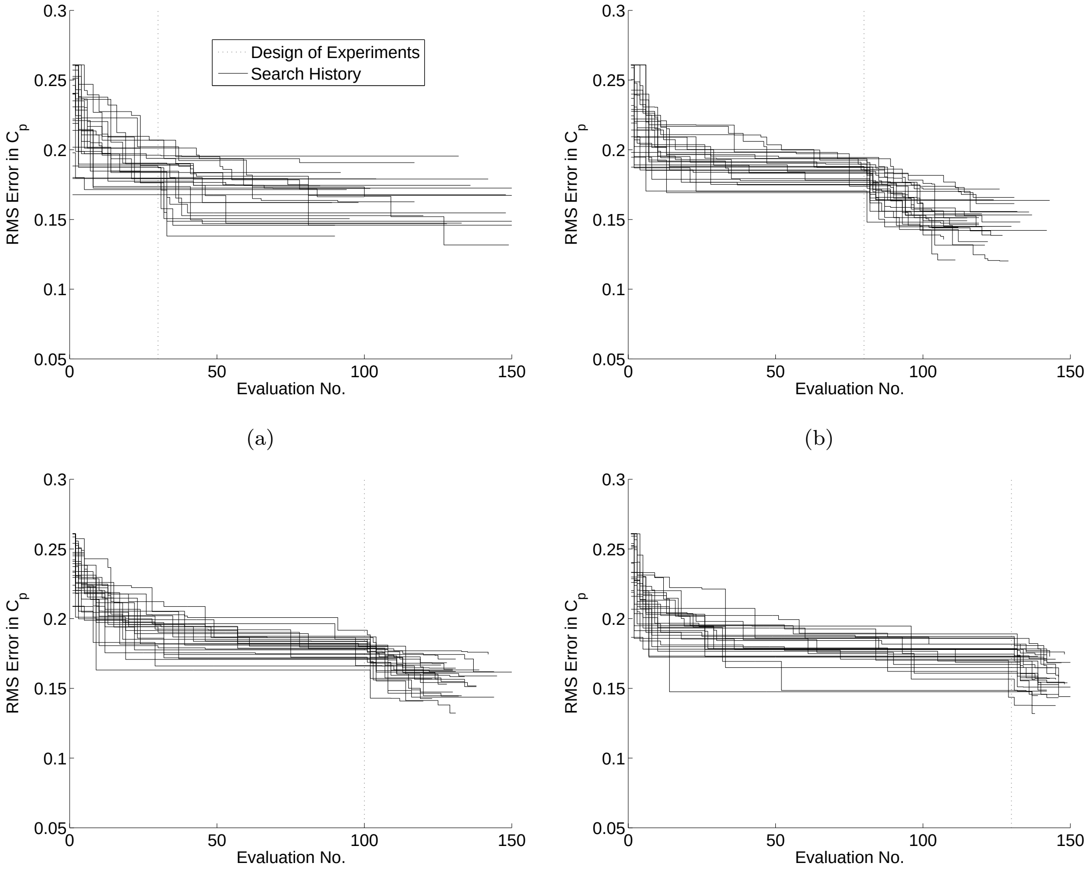
	$\frac{\text{DOE Size}}{\text{Total Eval. Budget}}$	$\frac{30}{150}$	$\frac{50}{150}$	$\frac{80}{150}$	$\frac{100}{150}$	$\frac{130}{150}$
6 Variables	Mean Obj.	0.1656	0.1542	**0.1501**	0.1530	0.1559
	Std. Obj.	0.0153	0.0119	0.0137	**0.0106**	0.0124
9 Variables	Mean Obj.	0.1689	0.1616	0.1550	**0.1509**	0.1517
	Std. Obj.	0.0171	0.0183	0.0179	0.0162	**0.0153**
12 Variables	Mean Obj.	0.1773	0.1632	0.1639	**0.1597**	0.1624
	Std. Obj.	0.0192	0.0200	0.0189	0.0188	**0.0135**

Table 3.4: Comparison of the effect of different DOE budgets on the mean and standard deviation of the objective function.

The results presented in Table 3.4 demonstrate that a reduction in the size of the DOE results in a degradation in performance while increasing the number of DOE points increases the performance of the tuning strategy for each of the problems up to a point. The larger design of experiments produces a more accurate response surface allowing the update points to be chosen more effectively resulting in a better design.

There is however a tradeoff between exploration and exploitation; as the size of the design of experiments is increased a smaller proportion of the budget is available to exploit any potential regions of optimal design. For example, the six variable optimisation sees a deterioration in performance when more than 80 points are used in the DOE. Even though the hyperparameters resulting from the tuning process produce a response surface model which more accurately predicts the true response, there is an insufficient budget of simulations available to exploit it.

The effect of DOE size is shown graphically by the optimisation histories presented in Figure 3.6. The smallest, 30 point, design of experiment results in a set of hyperparameters and corresponding response surface which generate update points that improve very little upon the best design obtained in the DOE in the majority of cases. This can be quantified if one considers the mean improvement of 0.0206, where the mean improvement is the difference between the objective function of the best design found in the DOE and the best design found after all of the updates, averaged over 50 optimisations. Increasing the size of the design of experiments produces a more effective response surface and the updates can improve more upon the best design, (Figure 3.6(b) with a mean improvement of 0.0291). As the size of the design of experiments is increased further still, the remaining simulation budget available for updates is insufficient to fully exploit the response surface and improve designs (Figures 3.6(c) & 3.6(d)). The mean improvement upon the best design of the DOE mirrors this, decreasing from 0.0234 to 0.0153, as the ability of the optimisation to exploit the response surface diminishes.



(a)

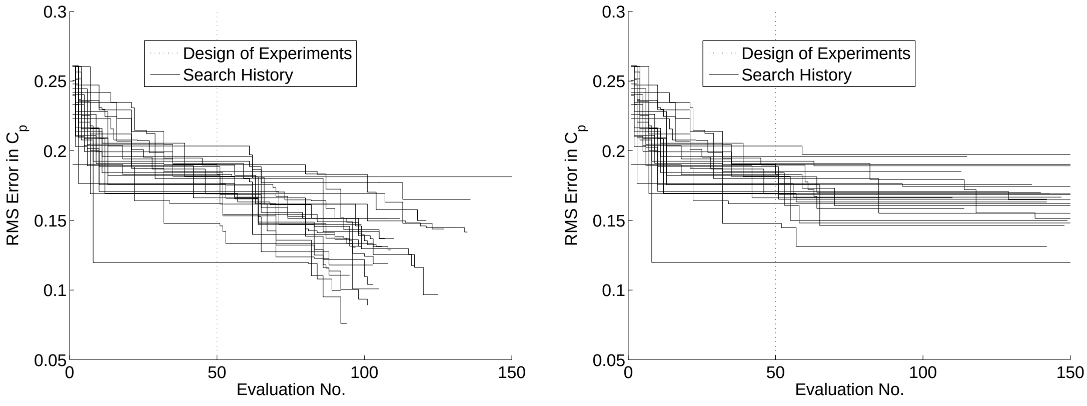
(b)

(c)

(d)

Figure 3.6: Optimisation histories for the 6 variable problem using the single tune strategy and (a) 30, (b) 80, (c) 100 and (d) 130 points in the initial DOE.

In addition to the effect of the DOE size relative to the total available budget, the complexity of the problem has an impact on the optimum number of points in the design of experiments. The designs obtained using the twelve variable geometry parameterisation continue to improve with 100 points in the DOE, (compared to the 80 point DOE in the six and 100 point DOE in the nine variable problems).



(a) (b)

FIGURE 3.7: Optimisation search histories of the 9 variable design problem using the 150 simulation budget and (a) the heavy tuning strategy and (b) single tune strategy.

The optimisation histories of Figure 3.7 illustrate another interesting advantage of the heavy tuning strategy; the reduction in the total number of simulations. As per the previous description of the update strategy, depending on the modality of the response surface the maximum budget of 10 update points may not be used in every update cycle. Hence, the continual tuning of the hyperparameters resulting in smoother response surfaces, enables the available updates to be used more effectively, reducing the simulation cost of the optimisation process. These unused updates could be employed in further exploitation of the metamodel of the heavy tuning strategy possibly resulting in better designs.

The gain in performance associated with increasing the size of the design of experiments, (shown in Table 3.4), is insignificant when one considers that the average objective function obtained using the heavy tuning strategy is 0.136, 0.130 and 0.129 for the six, nine and twelve variable problems respectively. One can surmise that an optimisation utilising the single tune strategy requires a larger simulation budget to perform as well as continuous hyperparameter tuning with a smaller budget . Considering the small cost of the aerofoil simulations used in the current optimisations, increasing the budget in such a manner would have little effect compared to the cost of the hyperparameter tuning. However, when an optimisation involves high fidelity 3D computational simulations requiring many hours of run time, it may be infeasible to significantly increase the computational budget. Nonetheless, if the cost of tuning the hyperparameters is comparable to that of the individual simulations, a balance may have to be struck between the total number of simulations and the level of tuning.

3.5.5 Alternate Tuning of Hyperparameters

Tuning the hyperparameters after alternate updates offers a compromise between the single tune strategy and the heavy tuning strategy. The total amount of tuning carried out is significantly reduced which offers a reduction in the total tuning time per optimisation. As the hyperparameters are reassessed throughout the updating process, although not as often as with the heavy tuning strategy, the response surface is better able to adjust to the potential lack of accuracy in the hyperparameters obtained after tuning the response surface based on the initial design of experiments. This reassessment of hyperparameters enables the optimisation strategy to more effectively exploit regions of optimal design, and, as such, produces better designs compared to the single tune strategy.

The increase in performance over the single tune strategy is clearly demonstrated by the results shown in Figure 3.3. Other than the light tuning strategy, the alternate tuning consistently produces results close to that of the heavy tuning strategy for those optimisations where the simulation budget can be deemed sufficient. Degradation in performance of the strategy with respect to the heavy tuning strategy occurs when the dimensionality of the design problem becomes an issue. Using the small simulation budget the alternate tuning strategy produces designs with objective functions close to those obtained using the heavy tuning strategy for the first four design problems. After this point, although there is a continual reduction in the mean objective function of the designs for the five and six variable problems, the reduction is not as significant as that obtained with the heavy tuning strategy. The results of the optimisations using the larger simulation budget remain close to those of the heavy tuning strategy up to the nine variable problem. The performance of the alternate strategy degrades after this point whereas the heavy tuning strategy produces a slight improvement in the objective function.

The alternate reassessment of the hyperparameters results in a surrogate model with an overall accuracy similar to that of the heavy and light tuning strategies. This is confirmed when one considers the average r^2 correlation of the surrogate models which are consistently close to that of the heavy and light tuning strategies. Alternate tuning of hyperparameters could therefore be considered to perform as well as the heavy and light tuning strategies when the dimensionality of the problem with regard to the simulation budget is not an overriding issue.

The alternate tuning strategy could be improved further when one considers the previous comparison of the heavy and light tuning strategies. The results of this comparison indicate that the 5,000 concentrated likelihood function evaluations of the genetic algorithm and dynamic hill climb used in the alternate tuning could be reduced with minimal loss of performance. Reducing the number of function evaluations in the alternate tuning strategy would produce an optimisation strategy competitive to that of the single tune strategy or θ tuning strategy in terms of total tuning time, but with the potential to produce better designs.

3.5.6 Tuning a Single Common Hyperparameter

Tuning a single, common hyperparameter produces, as would be expected, results very similar to those obtained using the heavy tune when a single variable parameterisation is optimised. As the number of variables used in the geometry parameterisation is increased, the results of each optimisation consistently fall short of that of the heavy, light and alternate tuning strategies, when the simulation budget is sufficient. That is, up to the six and nine variable design problems for the small and large simulation budgets, respectively. The results, however, are never as poor as those obtained through the single tune strategy.

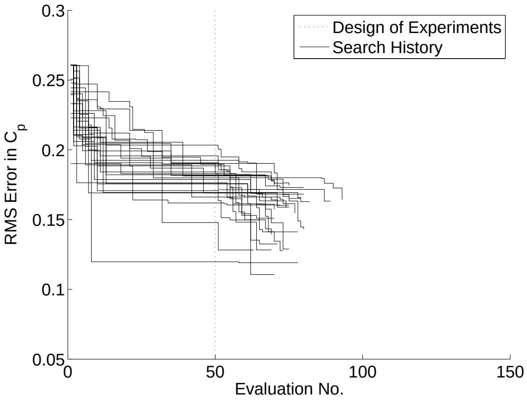


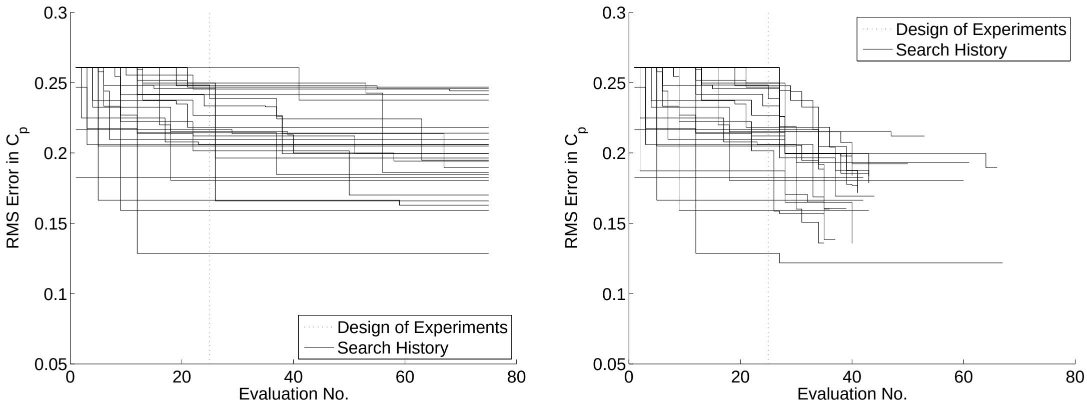
FIGURE 3.8: Optimisation search history of the nine variable design problem using the single θ tuning strategy using the 150 simulation budget.

When one compares the optimisation history of the nine variable design problem, presented in Figure 3.8, to that of the single tune strategy, (Figure 3.7(b)), one can see a continual improvement of the majority of the optimisations after the initial design of experiments. Where the single tune strategy offered little improvement over the best design obtained in the design of experiments, the single θ tuning strategy's continual reassessment of the hyperparameters after each update allows subsequent updates to continue to reduce the RMS error in the pressure distributions. This improvement in objective function occurs even though the global accuracy of the surrogate model may be considerably less than that obtained using the single tune strategy (Table 3.3).

Tuning a single hyperparameter has a smoothing effect on the response surface, removing the significance of each individual design variable to the objective function. This smoothing is apparent when one compares the optimisation history of Figure 3.8 to that of the heavy tuning strategy presented in Figure 3.7(a). The heavy tuning strategy uses a much larger proportion of the available simulation budget, (i.e. closer to the full allocation of 10 evaluations per update cycle), than the single θ tuning strategy. The process of searching the design space for potential regions of interest is, therefore, producing far more update points indicating that the response surface produced with the heavy tuning strategy has more regions of minima to exploit than the surface obtained using the single θ strategy. Given the same initial design of experiments, this reduction in the number of returned update points is likely to result from the smoothing effect of using a single set of hyperparameters.

Increasing the dimensionality of the design problem beyond the apparent limit of the simulation budget, (indicated by the dotted line in graphs of Appendix A), does not produce the same drop in performance that is observed with the other tuning strategies. Considering the most difficult optimisation, that of the 30 variable design problem using the 75 simulation budget, there is a large difference in the objective functions obtained using the heavy tuning strategy and the single θ tuning strategy. Tuning each of the 61 possible hyperparameters individually is an extremely challenging optimisation problem; one for which even the heavy tuning strategy may be insufficient.

The optimisation histories for the 30 variable design problem, presented in Figure 3.9, reinforce the failings of the heavy tuning strategy in this regard. The update points chosen using the metamodel resulting from a heavy tune improve very little upon the best design obtained using the design of experiments for the majority of optimisations. The single θ strategy, on the other hand, can improve on each of the designs. This is confirmed when one compares the average improvement in the objective function of the best design of the DOE to the final best design. The heavy tuning strategy has an average improvement of 0.015, while the θ tuning strategy has a much greater average improvement of 0.037. These results demonstrate the importance of an appropriate set of hyperparameters in an optimisation strategy, i.e. that a small set of well tuned hyperparameters may outperform a larger more complex set which is inaccurately tuned. Of course, for the problem considered here, all of the variables have broadly similar importance and so, using a single value of θ is plausible. In cases where the design variables differ significantly in nature and importance this might well not be the case.



(a)

(b)

Figure 3.9: Optimisation search histories of the 30 variable design problem using the 75 simulation budget and the heavy tuning strategy (a) and single θ tuning strategy using (b).

Finally it should be noted that, in a similar manner to the alternate tuning strategy, the tuning effort used in the single θ strategy could be reduced by decreasing the number of iterations of the genetic algorithm and dynamic hill climb, with little loss in performance.

3.6 Conclusions

In this chapter the inverse design of an aerofoil has been used to compare the performance of five kriging hyperparameter tuning strategies employed in constructing and updating a surrogate model of an aerodynamic response to geometric variation. Two of the strategies involved different degrees of tuning after the design of experiments and every update cycle, designated as the heavy and light tuning strategies. Tuning after only the design of experiments was considered, as was tuning after alternate updates and tuning a single pair of hyperparameters after every update. The inverse design problem utilised multiple Hicks-Henne functions to produce geometry parameterisations of varying complexity whilst maintaining a level of continuity between the different parameterisations.

All of the tuning strategies demonstrated the negative impact of increasing dimensionality on an optimisation given a fixed simulation budget, in particular the increasing inability of an optimisation to produce “good” designs.

The results of the heavy and light tuning strategies displayed little difference for those optimisations with an adequate simulation budget for the problem i.e. for problems with fewer than six variables for the small simulation budget and with fewer than twelve variables for the large simulation budget. The use of the light tuning strategy could therefore be recommended over the heavy tuning strategy for such problems.

Tuning the hyperparameters of the metamodel only once after the design of experiments has been demonstrated to perform extremely poorly compared to the other tuning strategies considered. Only with some form of continual reassessment of the hyperparameters throughout the updating process is the available computational budget used effectively. To produce results on a par with that of the other strategies it was demonstrated that a larger computational budget, in terms of both the initial design of experiments and updates, is required.

Tuning the hyperparameters after alternate updates produces results comparable to those of the heavy and light tuning strategies when problem dimensionality, in terms of the available simulation budget, is not an issue. This particular tuning strategy offers a significant saving over the computational effort required in tuning the hyperparameters after every update while having little impact on the optimisations ability to find “good” designs.

Tuning a single, common hyperparameter, (i.e. the same θ for every variable), has been demonstrated to perform poorly compared to the other strategies when the available simulation budget is sufficient for the problem. However, when problems of high dimensionality with limited simulation budget are considered, the tuning of a reduced set of hyperparameters produces a surprising improvement on the results obtained using the more expensive heavy tuning strategy. This indicates that given a high dimensional problem where extensive tuning of the complete set of hyperparameters is prohibitively expensive, extensively tuning a reduced set of hyperparameters may outperform an inaccurately tuned but complete set of hyperparameters. This final conclusion will depend, however, on the degree of variation in the problem variables.

Chapter 4

An Adjoint for Likelihood Optimisation

4.1 Introduction

The previous chapter of this thesis demonstrated the importance of the hyperparameter tuning process on the outcome of a kriging based optimisation. In these investigations the actual optimisation strategy employed in the tuning process remained consistent, with a two stage simplex search followed by a genetic algorithm and a terminal hill climb. However, this optimisation made no use of the gradient of the likelihood function which can be easily derived, and effectively utilised due to the function's smoothness.

Such gradient information could be used to accelerate convergence of the tuning process when implemented within a terminal search or when hybridised within a global optimisation. A gradient enhanced tuning strategy could result in a reduction in the total number of expensive, $O(n^3)$, likelihood evaluations made throughout an optimisation. This would therefore further reduce the bottleneck in any kriging based optimisation caused by hyperparameter tuning.

The following chapter describes the formulation of an adjoint of the likelihood via reverse algorithmic differentiation with the view to implementing this adjoint within a hybrid global optimisation. The efficiency of this formulation is compared to that of a traditional analytical gradient calculation, a forward mode algorithmic differentiation and finite differencing.

4.2 The Importance of Efficient Hyperparameter Tuning

It was observed in Section 2.3.2 that the calculation of the concentrated likelihood requires the factorisation of the correlation matrix, $\boldsymbol{R}$. If using the Cholesky factorisation this can be of order $O(n^3)$ and, therefore, extremely expensive if the correlation matrix is large. One of the reasons kriging is not typically adopted for design problems with more than 20 variables is the cost of the global optimisation necessary to maximise the likelihood. When the number of variables in the problem is large, a large number of sample points are needed to produce an adequate response surface. Jones et al. [1998], for example, advocate the use of $10d$ initial sample points.

At high dimensions the number of initial design points can therefore be large causing each evaluation of the likelihood to be relatively expensive. Moreover, as the number of dimensions increases so too does the number of hyperparameters requiring optimisation and therefore the length of the optimisation. As a typical kriging based optimisation progresses this cost will only increase further as update points are added and the correlation matrix increases in size. Coupling the increasing expense of a single likelihood evaluation with the application of a global optimiser, such as a genetic algorithm, which requires a large number of such evaluations, the total hyperparameter optimisation cost can quickly spiral out of control and may even approach that of the high fidelity simulations used in the underlying design problem.

It should be noted that the increasing cost of evaluating the likelihood as updates are added to the model can be mitigated to some degree through the application of a fixed number of sample points when optimising the hyperparameters. A selection of the best points, for example, could be used for hyperparameter optimisation thereby optimising the hyperparameters correctly in the regions of interest. The prediction can therefore employ all of the known points as only a single, common inversion of the correlation matrix is required. This process is considered briefly in Chapter 7.

Figure 4.1 helps to demonstrate the cost which can be incurred in optimising the likelihood by demonstrating the increase in time taken to make a single evaluation as the number of sample points increases for an arbitrary 50 variable problem, the Keane Bump Function taken from Keane and Nair [2005].

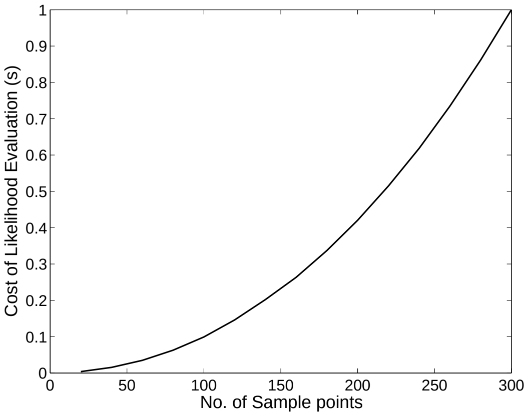


FIGURE 4.1: Demonstration of the real time cost of a single evaluation of the concentrated likelihood as the number of sample points increases for an arbitrary 50 variable design problem

Assume for example that 300 sample points are included in the initial sampling of the problem, based on this plot a single evaluation of the likelihood will take approximately 1 second on a desktop computer. An optimization of the likelihood which carries out a total of 10,000 evaluations, (such as the "heavy" tuning strategy used previously), will therefore take approximately 2.8 hours. This represents a significant bottle-neck in the optimisation process as a series of these likelihood maximisations are required throughout the course of a typical optimisation.

4.3 Traditional Analytical Derivative Calculation

The derivation of the analytical gradients of the likelihood with respect to the hyperparameters θ_l or p_l begins by first considering the derivative of the concentrated likelihood, (Equation 2.25),

$$\frac{\partial \phi}{\partial \psi} = -\frac{n}{2\hat{\sigma}^2}\frac{\partial \hat{\sigma}^2}{\partial \psi} - \frac{1}{2|\boldsymbol{R}|}\frac{\partial |\boldsymbol{R}|}{\partial \psi}, \tag{4.1}$$

where ψ represents any of the hyperparameters or indeed the regression constant λ. The derivative of the determinant of a matrix can be expressed in terms of the derivative of the matrix, Kubota [1994],

$$\frac{\partial |\boldsymbol{R}|}{\partial \psi} = |\boldsymbol{R}|\mathrm{Tr}\left[\boldsymbol{R}^{-1}\frac{\partial \boldsymbol{R}}{\partial \psi}\right]. \tag{4.2}$$

The derivative of the variance with respect to any hyperparameter can be expressed as,

$$\frac{\partial \hat{\sigma}^2}{\partial \psi} = \frac{1}{n}\left[(\boldsymbol{y}-\mathbf{1}\hat{\mu})^T\frac{\partial \boldsymbol{R}^{-1}}{\partial \psi}(\boldsymbol{y}-\mathbf{1}\hat{\mu}) - \right.$$
$$\left. - \left(\mathbf{1}\frac{\partial \hat{\mu}}{\partial \psi}\right)^T \boldsymbol{R}^{-1}(\boldsymbol{y}-\mathbf{1}\hat{\mu}) - (\boldsymbol{y}-\mathbf{1}\hat{\mu})^T\boldsymbol{R}^{-1}\left(\mathbf{1}\frac{\partial \hat{\mu}}{\partial \psi}\right)\right]. \tag{4.3}$$

However the terms involving $\frac{\partial \hat{\mu}}{\partial \psi}$ are typically 15-16 orders of magnitude smaller than the $\frac{\partial \boldsymbol{R}^{-1}}{\partial \psi}$ term, and can therefore be ignored. Hence Equation 4.3 can be simplified considerably, Park and Baek [2001],

$$\frac{\partial \hat{\sigma}^2}{\partial \psi} = \frac{1}{n} \left(\boldsymbol{y} - \mathbf{1}\hat{\mu}\right)^T \frac{\partial \boldsymbol{R}^{-1}}{\partial \psi} \left(\boldsymbol{y} - \mathbf{1}\hat{\mu}\right). \tag{4.4}$$

The derivative of the inverse of the correlation matrix can be expressed in terms of the derivative of the correlation matrix, Petersen and Pederson [2007],

$$\frac{\partial \boldsymbol{R}^{-1}}{\partial \psi} = -\boldsymbol{R}^{-1} \frac{\partial \boldsymbol{R}}{\partial \psi} \boldsymbol{R}^{-1}. \tag{4.5}$$

Substituting Equation 4.5 into Equation 4.4 and then into Equation 4.1 along with Equation 4.2 produces the following expression for the derivative of the concentrated likelihood function with respect to any hyperparameter, Park and Baek [2001],

$$\frac{\partial \phi}{\partial \psi} = \frac{1}{2\hat{\sigma}^2} \left[\left(\boldsymbol{y} - \mathbf{1}\hat{\mu}\right)^T \boldsymbol{R}^{-1} \frac{\partial \boldsymbol{R}}{\partial \psi} \boldsymbol{R}^{-1} \left(\boldsymbol{y} - \mathbf{1}\hat{\mu}\right) \right] - \frac{1}{2} \mathrm{Tr} \left[\boldsymbol{R}^{-1} \frac{\partial \boldsymbol{R}}{\partial \psi} \right]. \tag{4.6}$$

The derivative of the correlation matrix, $\boldsymbol{R}$ with respect to any hyperparameter is therefore the only remaining unknown. This can be expressed in terms of the derivative of every value within the matrix. Given that the i, j^{th} value of the correlation matrix is given by Equation 2.18, the partial derivative with respect to the l^{th}, θ hyperparameter is,

$$\frac{\partial \boldsymbol{R}_{i,j}}{\partial \theta_l} = -10^{\theta_l} \ln 10 \, ||\boldsymbol{x}_{i_l} - \boldsymbol{x}_{j_l}||^{p_l} \boldsymbol{R}_{i,j}, \tag{4.7}$$

and the partial derivative with respect to the l^{th}, p hyperparameter is,

$$\frac{\partial \boldsymbol{R}_{i,j}}{\partial p_l} = -10^{\theta_l} \ln ||\boldsymbol{x}_{i_l} - \boldsymbol{x}_{j_l}|| \, ||\boldsymbol{x}_{i_l} - \boldsymbol{x}_{j_l}||^{p_l} \boldsymbol{R}_{i,j}. \tag{4.8}$$

The derivatives of the concentrated likelihood function can therefore be calculated using Equation 4.6 once the matrices of derivatives of the correlation matrix with respect to each hyperparameter have been defined.

In summary, the partial derivatives of the likelihood can be calculated by first calculating the correlation matrix along with the $2d$ matrices of first derivatives of the correlation matrix, $\frac{\partial \boldsymbol{R}}{\partial \psi}$. The mean, variance and inverse of the correlation matrix can be calculated as normal and then combined to calculate the likelihood as per Equation 2.25 and the $2d$ partial derivatives as per Equation 4.6. The calculation of all of the partial derivatives using this method therefore requires the storage of the $2d$ matrices of first derivatives as well as a number of additional matrix multiplications.

The inclusion of a regression constant λ in the correlation matrix results in a kriging surface which no longer interpolates through the sample points. Adding the regression constant 10^{λ} to the diagonal of the correlation matrix results in the sample points no longer being correlated with themselves. Like the other hyperparameters, θ and p, the regression constant is optimised via maximising the likelihood. Therefore it is important to also consider the calculation of the derivative of the likelihood with respect to this constant.

As with both θ and p the derivative first requires the calculation of the derivative of the correlation matrix with respect to the hyperparameter of interest. As only the diagonal of the correlation matrix is dependent on the regression constant, the partial derivative of the correlation matrix with respect to the regression constant is itself diagonal in nature,

$$\frac{\partial \boldsymbol{R}_{ii}}{\partial \lambda} = 10^{\lambda} \ln 10. \tag{4.9}$$

Using this derivative in conjunction with Equation 4.6 will therefore result in the partial derivative of the concentrated likelihood with respect to the regression constant.

4.4 An Introduction to Algorithmic Differentiation

Algorithmic differentiation approaches the calculation of derivatives in a slightly different manner to that of traditional analytical differentiation. Here the computer algorithm used in the calculation of a function is differentiated line by line through the application of the chain rule. There are a number of programs which perform this operation automatically given a program's source code, this is termed automatic differentiation. However, in some circumstances this can lead to an inefficient program as the automatic differentiation process may fail to take account of the structure of the original problem. Mader et al. [2008], for example, found that the efficiency of their automatically differentiated computational fluid dynamics solver could be drastically improved after careful consideration of the structure of the underlying problem. Automatically differentiating the entire residual routine resulted in a series of unnecessary computations as the differentiation tool took no account of the sparsity of the flux Jacobian.

There are two modes of algorithmic differentiation, forward and reverse. Forward mode is akin to traditional differentiation with the differentiated program run once for every input. This produces a partial derivative of every output with respect to a single input each time the program is run. Reverse mode however runs the differentiated program once for every output and therefore obtains all of the partial derivatives of a single output with respect to all inputs for a single run of the differentiated program. The choice of method therefore depends on the nature of the problem. Simplistically, if there are more outputs than inputs it is more efficient to use the forward mode, but if there are more inputs than outputs then it is more efficient to use the reverse mode.

The assumption of a pass through the forward differentiated code for every input and a reverse pass for every output is a rather simplistic one. Some automatic differentiation tools can facilitate a vector forward mode which can evaluate multiple partial derivatives in a single pass. Likewise some automatic differentiation tools can facilitate a vector reverse mode whereby the derivatives of multiple outputs can be calculated in a single pass. Both methods save on computation time but incur a memory overhead.

To demonstrate the process of algorithmic differentiation we consider the simple analytical function y which is dependent on the variables x_1 and x_2,

$$y = \sin(x_1 x_2) + \left(\frac{x_1}{x_2}\right)^2 . \tag{4.10}$$

Although the partial derivatives of this function can be easily calculated,

$$\frac{\partial y}{\partial x_1} = x_2 \cos(x_1 x_2) + \frac{2x_1}{x_2^2} \tag{4.11}$$

and

$$\frac{\partial y}{\partial x_2} = x_1 \cos(x_1 x_2) - \frac{2x_1^2}{x_2^3}, \tag{4.12}$$

the simplicity of this function allows the basics of algorithmic differentiation to be demonstrated.

The process commences with the definition of the algorithm to calculate the function y. Each line of this algorithm, shown in Table 4.1 using the notation of Griewank [2000], carries out a single operation, i.e. an addition, multiplication, division or trigonometric operation, and terminates in the calculation of y. In this case we begin with the initialisation of the input variables x_1 and x_2 to v_0 and v_1 respectively, where v_i refers to the i^{th} intermediate variable calculated as the algorithm progresses. The third line multiplies v_0 and v_1 to give v_2 which is equivalent to $x_1 x_2$ in Equation 4.10, the forth line calculates $\frac{x_1}{x_2}$, the fifth, $\sin(x_1 x_2)$ and so forth until the function is calculated.

The forward mode of algorithmic differentiation, differentiates each line of this original algorithm in order, resulting in a tangent of the i^{th} intermediate variable v_i, denoted here by $\dot{v}_i$. For example, the third line of the original algorithm calculated $v_2 = v_0 v_1$ the tangent of this line is therefore the derivative of v_2 with respect to v_0 plus the derivative with respect to v_1, which results in the corresponding line in the forward mode algorithm, $\dot{v}_2 = \dot{v}_0 v_1 + v_0 \dot{v}_1$. Repeating the process through the entire algorithm results in an expression for $\dot{y}$ which is equivalent to the derivative of the original function with respect to an input, providing appropriate seedings of $\dot{x}_1$ and $\dot{x}_2$ are defined.

These seedings equate to the derivative of each input variable with respect to the required derivative of the overall algorithm. If, for example, the overall derivative of y with respect to x_1 is required then $\dot{x}_1 = \frac{\partial x_1}{\partial x_1} = 1$ and $\dot{x}_2 = \frac{\partial x_2}{\partial x_1} = 0$. One can observe from Table 4.1 that given these seedings, $\dot{y}$ will indeed correspond to Equation 4.11. Likewise, if $\dot{x}_1 = 0$ and $\dot{x}_2 = 1$ then $\dot{y}$ will correspond to Equation 4.12. The forward differentiation of the original algorithm must therefore be run twice, once for each of the input variables.

Original Algorithm	Forward Differentiation	Reverse Differentiation
$v_0 = x_1$	$\dot{v}_0 = \dot{x}_1$	$\bar{y} = 1$
$v_1 = x_2$	$\dot{v}_1 = \dot{x}_2$	$\bar{v}_6 = \bar{y}$
$v_2 = v_0 v_1$	$\dot{v}_2 = \dot{v}_0 v_1 + v_0 \dot{v}_1$	$\bar{v}_5 = \bar{v}_6$
$v_3 = \frac{v_0}{v_1}$	$\dot{v}_3 = \frac{\dot{v}_0}{v_1} - \dot{v}_1 \frac{v_3}{v_1}$	$\bar{v}_4 = \bar{v}_6$
$v_4 = \sin(v_2)$	$\dot{v}_4 = \dot{v}_2 \cos(v_2)$	$\bar{v}_3 = 2\bar{v}_5 v_3$
$v_5 = v_3^2$	$\dot{v}_5 = 2v_3\dot{v}_3$	$\bar{v}_2 = \bar{v}_4 \cos(v_2)$
$v_6 = v_4 + v_5$	$\dot{v}_6 = \dot{v}_4 + \dot{v}_5$	$\bar{v}_1 = \bar{v}_2 v_0 - \bar{v}_3 \frac{v_3}{v_1}$
$y = v_6$	$\dot{y} = \dot{v}_6$	$\bar{v}_0 = \bar{v}_2 v_1 + \frac{\bar{v}_3}{v_1}$

Table 4.1: A simple example of forward & reverse algorithmic differentiation.

Rather than selecting an input and calculating the sensitivity of every intermediate variable with respect to that input, the reverse, or adjoint mode, proceeds backwards through the original algorithm selecting an output and calculating the sensitivity of that output with respect to every intermediate variable. Where the forward mode is essentially $\dot{v}_i = \frac{\partial v_i}{\partial x}$ the reverse mode becomes $\bar{v}_i = \frac{\partial y}{\partial v_i}$ with $\bar{v}_i$ denoting the adjoint of the i^{th} intermediate variable, Griewank [2000]. The adjoint of the i^{th} intermediate variable is equivalent to the sum of the partial derivatives of those intermediate variables which are dependent on v_i, multiplied by the corresponding adjoint of the dependent intermediate variables. The intermediate variable v_6, for example, affects only y hence the adjoint of v_6 is,

$$\bar{v}_6 = \bar{y}\frac{\partial y}{\partial v_6} = \bar{y}. \tag{4.13}$$

Likewise, v_5 affects only v_6 hence $\bar{v}_5 = \bar{v}_6$. Things are complicated somewhat when an intermediate variable in the original algorithm affects a number of the following intermediate variables. Consider, for example, the adjoint of v_1; as v_1 affects both v_2 and v_3 the adjoint of v_1 is the sum of the partial derivatives of v_2 and v_3 with respect to v_1 multiplied by their respective adjoints,

$$\bar{v}_1 = \bar{v}_2\frac{\partial v_2}{\partial v_1} + \bar{v}_3\frac{\partial v_3}{\partial v_1} \tag{4.14}$$

which is equivalent to,

$$\bar{v}_1 = \bar{v}_2 v_0 - \bar{v}_3\frac{v_3}{v_1}. \tag{4.15}$$

When this reverse differentiation process is completed and the algorithm run, the resulting values of the adjoints, $\bar{v}_0$ and $\bar{v}_1$, are equivalent to the partial derivatives, $\frac{\partial y}{\partial x_1}$ and $\frac{\partial y}{\partial x_2}$. Unlike the forward mode the reverse mode, presented in Table 4.1, requires only a single run to calculate all of the partial derivatives of y given the initial seeding of $\bar{y}$, which, as there is only one output, equals one. If the original algorithm contained a number of outputs then the reverse mode would be run once for each output with the seeding adjusted in a manner similar to that for the forward mode.

Even though the above example is very simple, it demonstrates the performance improvements offered when partial derivatives of a single output, as is the case in likelihood maximisation, are required. Exactly the same techniques can be applied to any computer algorithm though we now apply them to the calculation of the partial derivatives of the likelihood.

4.5 Reverse Algorithmic Differentiation of the Likelihood

The calculation of the concentrated likelihood, Equation 2.25, consists of a single output which is dependent on d pairs of hyperparameters and a single regression constant, λ, if included. Reverse algorithmic differentiation is therefore the most efficient method to apply to this particular problem and it is the application of this technique which we now consider.

The algorithm to calculate the partial derivatives of the concentrated likelihood via reverse algorithmic differentiation begins with the calculation of the likelihood as normal. This is then followed by a reverse differentiation of the original algorithm which, using information stored during the calculation of the likelihood, calculates all of the partial derivatives.

The calculation of the likelihood begins with the construction of the correlation matrix $\boldsymbol{R}$. This symmetrical matrix is then decomposed using the Cholesky factorisation into a lower triangular matrix $\boldsymbol{L}$ where,

$$\boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{R}. \tag{4.16}$$

This matrix can then be used to calculate the mean, $\hat{\mu}$ and variance, $\hat{\sigma}^2$ using Equations 2.23 and 2.24 respectively in conjunction with forward and backward substitution. The variance, for example, is calculated through the forward substitution,

$$\boldsymbol{T}_1 = \boldsymbol{L}^{-1}\left(\boldsymbol{y} - \mathbf{1}\hat{\mu}\right), \tag{4.17}$$

which is followed by the back substitution,

$$\boldsymbol{T}_2 = (\boldsymbol{L}^T)^{-1}\boldsymbol{T}_1, \tag{4.18}$$

and finally the vector multiplication,

$$\hat{\sigma}^2 = \frac{1}{n}\left(\boldsymbol{y} - \mathbf{1}\hat{\mu}\right)^T \boldsymbol{T}_2. \tag{4.19}$$

The vectors $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$ represent two temporary vectors which are necessary for the subsequent calculation of the derivatives. The Cholesky factorisation is also used to calculate the natural log of the determinant via,

$$\frac{1}{2}\ln(|\boldsymbol{R}|) = \sum_i \ln \boldsymbol{L}_{ii}. \tag{4.20}$$

With the variance and the log of the determinant known, the concentrated likelihood can be easily calculated using Equation 2.25.

The reverse mode works backwards beginning from an initial seeding of the adjoint of the likelihood, $\bar{\phi} = 1$. From this starting point the seeding for the adjoint $\bar{\boldsymbol{T}}_2$ can be calculated to be,

$$\bar{\boldsymbol{T}}_2 = -\frac{(\boldsymbol{y} - \mathbf{1}\hat{\mu})}{2\hat{\sigma}^2}. \tag{4.21}$$

This seeding can then be used to calculate $\bar{\boldsymbol{T}}_1$ and $\bar{\boldsymbol{L}}_1$ using the reversely differentiated back substitution algorithm, where $\bar{\boldsymbol{L}}_1$ is the adjoint of the upper triangular matrix $\boldsymbol{L}^T$ used in the back substitution, Equation 4.18. The adjoint, $\bar{\boldsymbol{L}}_2$, of the lower triangular matrix $\boldsymbol{L}$ used in the forward substitution of Equation 4.17 is then calculated using $\bar{\boldsymbol{T}}_1$ and the reversely differentiated forward substitution algorithm. Smith [1995] demonstrated that the adjoint of the log of the determinant of a matrix is equal to the reciprocal of the diagonal of the lower triangular matrix $\boldsymbol{L}$ resulting from the Cholesky factorisation, but in this case as Equation 2.25 involves the negative of log of the determinant,

$$\bar{\boldsymbol{L}}_{3_{ii}} = -\frac{1}{\boldsymbol{L}_{ii}} \tag{4.22}$$

Hence, the total adjoint seeding for use in Smith's reversely differentiated Cholesky factorisation is,

$$\bar{\boldsymbol{L}} = \bar{\boldsymbol{L}}_1^T + \bar{\boldsymbol{L}}_2 + \bar{\boldsymbol{L}}_3. \tag{4.23}$$

When this total adjoint seeding is then used in conjunction with the original matrix $\boldsymbol{L}$, the lower triangular matrix $\bar{\boldsymbol{R}}$ is calculated which can then be used along with information stored during the original calculation of the correlation matrix to calculate all of the partial derivatives. The derivative of the likelihood with respect to the l^{th}, θ hyperparameter is therefore,

$$\frac{\partial \phi}{\partial \theta_l} = \ln 10 \sum_{ij} -10^{\theta_l} ||\boldsymbol{x}_{i_l} - \boldsymbol{x}_{j_l}||^{p_l} \boldsymbol{R}_{ij} \bar{\boldsymbol{R}}_{ij} \tag{4.24}$$

and the derivative with respect to the l^{th}, p hyperparameter is

$$\frac{\partial \phi}{\partial p_l} = \sum_{ij} -10^{\theta_l} ||\boldsymbol{x}_{i_l} - \boldsymbol{x}_{j_l}||^{p_l} \ln ||\boldsymbol{x}_{i_l} - \boldsymbol{x}_{j_l}|| \boldsymbol{R}_{ij} \bar{\boldsymbol{R}}_{ij}. \tag{4.25}$$

The derivative of the likelihood with respect to a regression constant λ can be easily calculated from $\bar{\boldsymbol{R}}$,

$$\frac{\partial \phi}{\partial \lambda} = 10^{\lambda} \ln 10 \sum_i \bar{\boldsymbol{R}}_{ii}, \tag{4.26}$$

assuming that 10^{λ} has been added to the diagonal of the correlation matrix.

The algorithms for the Cholesky factorisation, forward and backward substitution and their respective reversely differentiated algorithms are presented in Appendix B for the interested reader.

4.6 Computational Efficiency of the Derivative Calculations

Having described in detail the procedure for calculating the partial derivatives of the likelihood via both the analytical and the reverse algorithmic differentiation methods, one must now consider each method's computational efficiency. The analytical method requires the calculation of Equation 4.6 for each hyperparameter while the reverse method requires only a single reverse pass of the forward substitution, the backward substitution and the Cholesky factorisation to obtain the matrix $\bar{\boldsymbol{R}}$ which can be used to calculate the partial derivatives via equations Equation 4.24, 4.25 and 4.26. The question is therefore, by how much does this reduction in the number of calculations improve the performance of the derivative calculation?

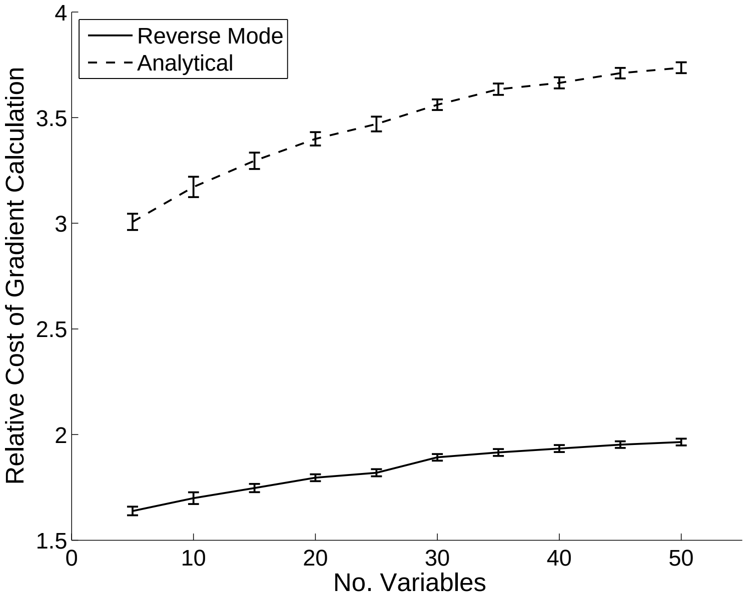


Figure 4.2: A comparison of the relative costs of calculating all of the partial derivatives of the likelihood via reverse algorithmic differentiation and the analytical formulation

We now consider the relative cost of calculating the likelihood and all derivatives to the cost of calculating only the likelihood. Figure 4.2 shows the change in this relative cost as the number of dimensions of the underlying problem, to which a kriging surface is fit, increases. It must be noted that here the number of sample points remains a constant as the number of dimensions increases, 50 points in this case, and that the relative cost includes the calculation of all of the derivatives with respect to θ and p for every dimension as well as the cost of calculating the likelihood itself. The cost obtained for 20 variables therefore equates to the calculation of 40 partial derivatives and the likelihood. By retaining a constant n, Figure 4.2 shows the effect of purely an increase in problem dimensionality. All of the presented methods were coded and analysed in Matlab.

A total of 100 different Latin hypercube sampling plans of the Keane Bump Function, (Keane and Nair [2005]), were calculated and stored. Each evaluation of the likelihood and corresponding partial derivatives in Figure 4.2 were therefore made from a common data set. The likelihood and derivatives of a single set of hyperparameters were evaluated for each of these sample plans. The time for each of these calculations was then recorded and compared to the time taken to calculate only the likelihood, the resulting relative times were then averaged.

It should be noted that a level of commonality has been preserved between all of the relevant Matlab code. Identical forward and backward substitution algorithms are employed, for example, in the evaluation of $\hat{\sigma}^2$ and $\hat{\mu}$ in the initial likelihood evaluation of both the analytical and reverse mode algorithms. The only differences are in the calculation and storage of any intermediate variables required in the analytical or reverse mode calculations of the derivatives. This means that the timings of Figure 4.2 compare like with like as much as possible and therefore offer a more meaningful representation of the differences in the cost of the two methods.

The differences in cost presented in Figure 4.2 can be explained by analysing the way in which the derivatives are calculated. Consider first the analytical method. In this case the final calculation of a derivative, Equation 4.6, requires one additional matrix-matrix multiplication in the calculation of the derivative of the determinant, and three additional matrix-vector multiplications and a vector-vector multiplication in the calculation of the derivative of the variance. Calculating the partial derivative of the likelihood with respect to all of the hyperparameters when fitting a krig to a d dimensional problem therefore requires $2d$ additional matrix-matrix multiplications, $6d$ additional matrix-vector multiplications and $2d$ additional vector-vector multiplications. Including the regression constant increases the expense of calculating all of the derivatives slightly but the cost of calculating this derivative is smaller than for the other hyperparameters due to the sparse nature of the $\frac{\partial \boldsymbol{R}}{\partial \lambda}$ matrix which simplifies the calculations in Equation 4.6.

The reverse mode however, requires a single reverse pass of the back substitution which is followed by a reverse pass of the forward substitution and then a reverse pass of the Cholesky factorisation. Each of these calculations must be performed only once and is therefore independent of the number of dimensions in the underlying problem. Only the final step in the derivative calculation, Equations 4.24 and 4.25, are dependent on the number of dimensions in the underlying problem, with d calculations of each required.

These final calculations are simplified somewhat by the fact that $\bar{\boldsymbol{R}}$ is lower triangular and that the elemental multiplication of the correlation matrix $\boldsymbol{R}$ with $\bar{\boldsymbol{R}}$ is common to all calculations and can therefore be carried out only once. These final two calculations do however require d lower triangular matrices of $-10^{\theta_l}||\boldsymbol{x}_{i_l} - \boldsymbol{x}_{j_l}||^{p_l}$ and d matrices of $||\boldsymbol{x}_{i_l} - \boldsymbol{x}_{j_l}||$ to be stored during the initial likelihood calculation. The calculation of a derivative with respect to p is slightly more expensive than calculating a derivative with respect to θ as the natural log of $||\boldsymbol{x}_{i_l} - \boldsymbol{x}_{j_l}||$ is required.

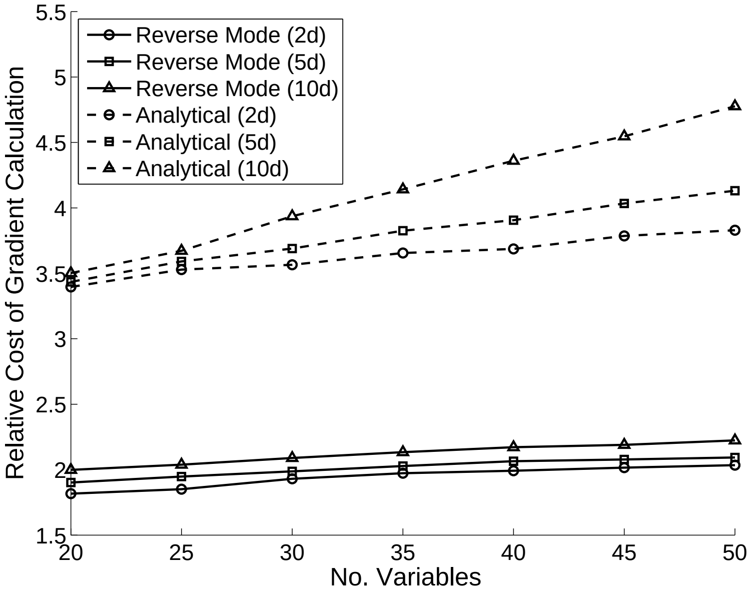


Figure 4.3: A comparison of effect of sampling density on the relative costs of calculating all of the partial derivatives of the likelihood via reverse algorithmic differentiation and the analytical formulation

Figure 4.3 demonstrates how the relative cost of calculating the likelihood and all of the partial derivatives changes as the sampling density of the underlying problem alters. Once again the Keane Bump function is sampled but this time the number of sample points is adjusted according to the number of dimensions in the underlying problem. Sampling densities of $2d$, $5d$ and $10d$ are all employed. An underlying problem with 25 dimensions will therefore have either 50, 125 or 250 sample points.

The results of Figure 4.3 demonstrate the adjoint method's relative resistance to an increase in sampling density. The relative cost of the analytical method for example increases by 24.7% when the sample density of the 50 variable problem increases from $2d$ to $10d$ whereas the relative cost of the adjoint only increases by 9.3%. These results can be explained by once again analysing the way in which the derivatives are calculated. The number of sample points directly influences the size of the kriging correlation matrix and hence the size of any matrices or vectors used in the subsequent calculations. More importantly this directly affects the cost of the additional matrix-matrix, matrix-vector and vector-vector multiplications required to calculate the likelihood derivatives. As the number of sample points increases so too does the cost of these additional calculations. The problem is compounded at higher dimensions where not only are there more of these calculations, but their cost increases as more sample points are required to produce an accurate response surface. This can be observed in Figure 4.3 where an increase in the number of sample points has a growing impact on the relative cost as dimensionality increases.

The calculations carried out by the adjoint method are also affected by the number of sample points and hence size of the correlation matrix. The cost of the reverse forward and backward substitutions and the reverse Cholesky are all dependent on the number of sample points, however unlike the analytical method, these are only carried out once no matter the number of dimensions. Likewise the cost of elemental multiplication of the correlation matrix $\boldsymbol{R}$ with the lower triangular matrix, $\bar{\boldsymbol{R}}$, is dependent on the number of sample points but this is again only carried out once. The calculation of equations 4.24 and 4.25 are dependent on both the number of sample points and the number of dimensions. However, due to the lower triangular nature of the previous elemental multiplication the impact of the number of sample points is reduced somewhat. Combining all of these features produces a method of calculating the partial derivatives of the likelihood which is both more efficient than the analytical method and less prone to large increases in relative cost as sampling density increases.

The analytical derivative of the likelihood, Equation 4.6, assumes a simplified formulation of the derivative of the variance, where the terms due to $\frac{\partial \hat{\mu}}{\partial \psi}$ are neglected due to massive difference in their magnitude relative to the $\frac{\partial \boldsymbol{R}^{-1}}{\partial \psi}$ term. The adjoint formulation presented above also employs this assumption, hence the adjoint of the mean, $\hat{\mu}$, and its subsequent effect on the initial seeding of the adjoint of the reversely differentiated Cholesky factorisation, $\bar{\boldsymbol{L}}$, is not calculated. An automatic differentiation of the likelihood however, may not take into account the relative insignificance of this term and calculate the adjoint of the mean. These additional calculations may result in a less efficient algorithm than the one presented above.

No. of Variables	Reverse Differentiation	Finite Differencing
2	4.41×10^{-13}	2.17×10^{-3}
5	3.80×10^{-13}	4.95×10^{-5}
10	6.71×10^{-13}	2.00×10^{-5}
15	2.16×10^{-13}	3.42×10^{-5}
25	1.14×10^{-13}	3.94×10^{-6}

Table 4.2: A comparison of the RMS error in the gradients calculated via reverse differentiation and finite differencing to that of the traditional analytical method

Table 4.2 provides an indication of the numerical accuracy of the gradients obtained via the adjoint method relative to those obtained via the analytical gradient of Equation 4.6. Using the aerofoil inverse design problem of Section 3.4 a series of DOEs of increasing complexity were created for the purposes of likelihood calculation. The partial derivatives of the likelihood with respect to each of the hyperparameters were then calculated for 50 sets of kriging hyperparameters for each method and compared to those resulting from the analytical method. Table 4.2 demonstrates such a negligible difference in the magnitude of the results that one could consider the gradients calculated to be almost identical.

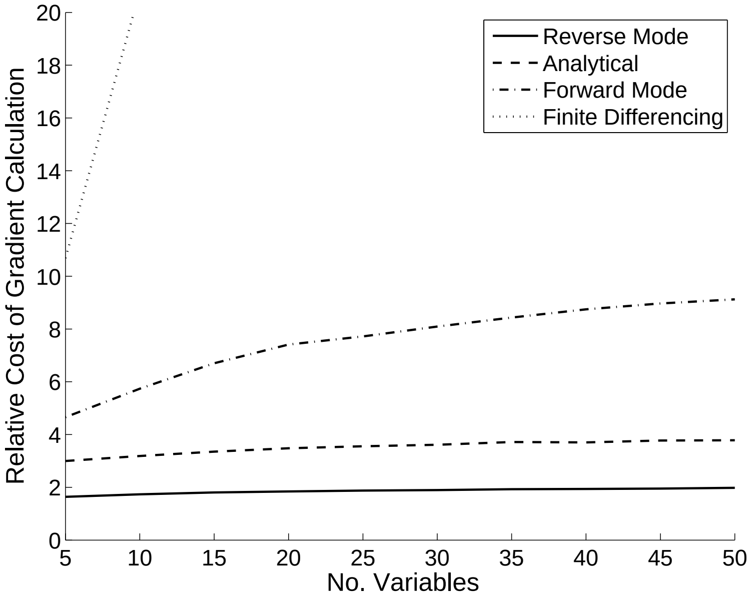


Figure 4.4: Algorithmic differentiation and the analytical methods compared to finite differencing and forward algorithmic differentiation

The relative cost of calculating the partial derivatives via the additional methods of finite differencing and a forward algorithmic differentiation are presented in Figure 4.4. The forward mode results were obtained via a manual forward differentiation of the likelihood calculation. The forward differentiation of the Cholesky factorisation presented in Smith [1995] was employed along with a manually derived forward differentiation of the forward and backward substitutions employed in the calculation of the variance. As with the reverse and analytical formulations, the forward mode takes advantage of the reduction in cost associated with the redundancy of the calculation of $\dot{\mu}$. Although the pseudo code of the forward mode is not considered within this paper, this figure serves to highlight the importance of selecting the appropriate method of algorithmic differentiation for a particular problem.

As one can observe, compared to both the analytical and reverse methods, the forward method is more sensitive to an increase in problem dimensionality. Considering that the forward mode requires a pass through the differentiated algorithm for every hyperparameter it is therefore unsurprising that the cost of the derivative calculation becomes an issue. In this case, finite differencing is the most expensive method of calculating the derivatives, requiring two additional full likelihood calculations for every additional variable in the predictive surrogate. Unlike the other methods considered, finite differencing will not produce an exact derivative but rather an approximation to it which is dependent on the step length used, as shown in Table 4.2.

The above comparison therefore indicates that the reverse algorithmic differentiation of the likelihood is the most efficient method of calculating exact partial derivatives. This performance improvement allows for a faster gradient descent search of the likelihood but will also reduce the effort per generation employed in the calculation of gradients if a local search were to be hybridised with a global search.

4.7 Conclusions

An efficient calculation of the derivatives of the likelihood via an adjoint derived using reverse algorithmic differentiation has been presented. This formulation has been demonstrated to be more efficient than calculating gradients via the traditional analytical method, calculating the likelihood and all of its derivatives for approximately twice the cost of a single likelihood evaluation even on a 50 variable problem. The cost of this process has also been demonstrated to be less sensitive to an increase in the dimensionality of the problem.

The calculation of the adjoint of the likelihood can be approached from another direction. The collection of linear algebra results presented by Giles [2008] can be used to facilitate the calculation of $\bar{\boldsymbol{R}}$ without the need for the reverse differentiation of the Cholesky factorisation or the forward or back substitution algorithms. This method results in an expression for $\bar{\boldsymbol{R}}$ involving one additional vector-matrix multiplication, one additional vector-vector multiplication and one additional matrix-matrix multiplication. However this is at the cost of the calculation and storage of $\boldsymbol{R}^{-1}$. The precise performance gains offered over the presented method employing reverse algorithmic differentiation remains an unanswered question but the linear algebra method can make much more effective use of efficient optimised matrix multiplication subroutines such as those included in BLAS.

Although the reverse algorithmic differentiation process has been applied to a traditional kriging Gaussian kernel, it could be easily extended to other kernels of an alternate formulation or indeed to any other process where a likelihood maximisation is required. A similar process could even be applied to the hyperparameter optimisation of gradient or Hessian enhanced surrogate models or indeed co-kriging models.

Chapter 5

Hyperparameter Optimisation Utilising the Likelihood Adjoint

5.1 Introduction

The previous chapter of this thesis described the formulation of an adjoint of the kriging likelihood function derived via the application of reverse algorithmic differentiation. Demonstrated to be considerably more efficient than the traditional analytical gradient calculation, this adjoint allows for a more efficient local optimisation of the kriging hyperparameters.

Hollingsworth and Mavris [2003] demonstrated that the selection of an optimum set of hyperparameters through maximum likelihood optimisation is a highly multi-modal problem, requiring the use of a global optimisation algorithm to provide reliable results. A purely local optimisation process would, although reasonably fast, prove detrimental when utilised within a kriging optimisation due to the use of inappropriate or suboptimal hyperparameters in the construction of the model.

As discussed in Chapter 2, global optimisation techniques are capable of locating the region of a global optimum but find it difficult to converge to a precise solution. The opposite is true with local optimisers, which converge quickly to a final solution but cannot escape local minima. This chapter aims to implement the adjoint of the likelihood within a hybrid optimisation such that gradient information is utilised by a local search embedded within an overall global optimisation. The adjoint can therefore be effectively used to accelerate convergence to an optimum set of hyperparameters without becoming trapped at a suboptimal solution.

5.2 Current Gradient Enhanced Likelihood Optimisation Methodologies

The literature includes a number of examples of the application of gradient information in likelihood optimisation. Park and Baek [2001] derived an analytical gradient of the likelihood for use in a local quasi-Newton optimisation. Zhang and Leithead [2005] took this a step further and derived an analytical Hessian of the likelihood and employed this in conjunction with a trust region search. One of the most recent techniques for the reduction of tuning cost is that of Leithead and Zhang [2007] which reduces the cost of approximate likelihoods and derivatives to an $O(n^2)$ operation through an approximation to the inverse of the covariance matrix via the BFGS updating formula, Equation 2.9.

Each of these methods is a local optimisation of the likelihood and as such will only locate the global optimum if initialised in the region of that optimum or if an appropriate restart procedure is adopted. It should be noted however, that there are cases when such a local optimiser can be effective in finding the optimal hyperparameters. Zhang and Leithead [2005] note that given a sufficiently large dataset upon which to build the surrogate model the modality of the likelihood space is greatly reduced. Such densely populated design spaces are however very rare in engineering design optimisations, especially when each objective function evaluation involves a costly high fidelity simulation, but may be commonplace in the field of Gaussian process regression.

Whereas the methods of Park et al. and Zhang et al. exploit an exact analytical gradient or Hessian, the approximation method of Leithead et al. still requires an initial exact inverse of the correlation matrix and may require additional exact inversions as the optimisation progresses due to corruption of the approximation. When used in conjunction with an engineering optimisation problem, where there are few sample points and the likelihood is multi-modal in nature, the computational effort spent in carrying out the initial starting inversion and subsequent restart inversions, may be better spent in performing a global exploration of the likelihood.

5.3 Metrics for the Comparison of Likelihood Optimisers

While the development of an efficient hybrid hyperparameter tuning strategy is the ultimate goal of this chapter it is necessary to first define metrics by which the performance of such a tuning strategy can be evaluated. Two metrics are used in the subsequent comparisons, the mean improvement in likelihood and the overall effect of the tuning strategy on a complete optimisation. Both of these metrics employ the d dimensional aerofoil inverse design problem described previously in Section 3.4, as well as the benchmarking procedure outlined in Section 3.2.

The mean improvement in likelihood is based on the difference in likelihood attained by a strategy to that attained by the Options Matlab genetic algorithm. For a given dimensionality the inverse design problem is sampled using a total of 50 different Latin hypercubes. Each sampling of the objective function is then used to construct a kriging model via an optimisation of the likelihood. The likelihood achieved by the new strategy is subtracted by that achieved by the baseline genetic algorithm, (given the same sampling plan), and the mean taken,

$$\frac{1}{50}\sum_{i=1}^{50}\phi_i - \phi_{GA_i}. \tag{5.1}$$

As a likelihood optimisation typically involves the minimisation of the negative of the concentrated likelihood function, a negative value from the above equation therefore indicates a better performing tuning strategy.

While the above metric provides an indication of the improvement offered by an optimisation strategy in terms of the final likelihood obtained, it provides little indication of the effect that implementing such a strategy has on an overall kriging based optimisation. The inverse design optimisation will therefore be adopted again with the optimisation completed using a predefined budget of function evaluations.

Unlike the similar comparisons of Section 3.5, where a fixed budget was applied to a series of problems of increasing dimensionality, each optimisation will employ a total of $15 \times d$ evaluations. An inverse design problem of 15 variables will therefore have a total budget of 225 evaluations. This removes the problems associated with a fixed simulation budget and an increasing dimensionality observed in Section 3.5 and provides a clearer indication of the effect of the tuning strategy. Of the total evaluation budget, one third will be used in the initial design of experiments. This initial sampling of the problem with $5 \times d$ points therefore forms the basis of the calculation of the mean improvement in likelihood.

5.4 Comparison of Optimisation Methodologies

The intention to develop a hybrid optimisation strategy for the purposes of hyperparameter tuning introduces something of a dilemma. Which global optimisation strategy should such a hybrid algorithm be based upon? Chapter 2 described a number of potential global optimisation strategies; genetic algorithms, simulated annealing, dynamic hill climbing and particle swarms. Using the previously defined metrics the performance of these algorithms can be compared to that of the genetic algorithm and an appropriate algorithm can be selected for hybridisation. To this end a total of six different hyperparameter optimisers are evaluated, details of which are presented in Table 5.1.

	Description
Heavy	A 5000 evaluation genetic algorithm followed by a 5000 evaluation dynamic hill climb
GA-DHC	A 2500 evaluation genetic algorithm followed by a 2500 evaluation dynamic hill climb
SA	A 5000 evaluation simulated annealing
DHC	A 5000 evaluation dynamic hill climb
PSO	A 5000 evaluation particle swarm optimisation
SQP	A single sequential quadratic programming optimisation commencing from a random starting point and running to convergence

Table 5.1: Description of the six hyperparameter optimisers compared.

Three basic global optimisers, the particle swarm, the dynamic hill climber and simulated annealing are selected for comparison. One of these three, or the baseline genetic algorithm will form the basis of the hybrid strategy. Also included for comparison is the "Heavy" tuning strategy, (neglecting the two stage simplex search), employed so successfully in Chapter 3 and in the literature by Hoyle et al. [2006] and Keane [2006].

The genetic algorithm, swarm and simulated annealing algorithms all employ a population of 50 members for 100 generations. As the baseline genetic algorithm is restricted to a total of 5,000 likelihood evaluations so too are the particle swarm, dynamic hill climber and simulated annealing. The Heavy strategy however uses a total of 10,000 function evaluations, split evenly between the genetic algorithm and hill climber, and as such enjoys somewhat of an advantage over the other algorithms being tested. Hence, a second version of this strategy, denoted as "GA-DHC" in Table 5.1, employing a total of 5,000 likelihood evaluations is also evaluated.

The final tuning strategy considered involves a simple local search using a sequential quadratic programming (SQP) algorithm commenced from a random starting point. This strategy is included as a contrast to the global optimisers with the intention of demonstrating the importance of the global optimisation of hyperparameters.

No. of Variables	Heavy	GA-DHC	SA	DHC	PSO	SQP
2	-0.785	-0.509	-0.056	-0.929	-0.037	1.452
5	-1.734	-1.442	-0.347	-1.804	-0.789	5.319
10	-3.685	-2.549	-0.860	-2.963	-0.656	5.448
15	-3.169	-2.515	-0.090	-3.716	-1.558	4.925
25	-5.452	-2.885	2.731	-2.639	-0.465	9.941

Table 5.2: Comparison of the mean improvement in likelihood function for six different hyperparameter optimisers over a range of problem sizes.

The mean improvement in likelihood is presented in Table 5.2 for each of the six algorithms. The traditional Heavy strategy performs consistently better than the other strategies, especially on higher dimensional problems. This is perhaps rather unsurprising given that it uses double the number of likelihood evaluations of any other strategy.

A reduction in the total number of likelihood evaluations to 5,000 sees the GA-DHC strategy performing well at lower dimensions compared to the much more expensive Heavy strategy. However, the relative performance of the GA-DHC strategy reduces as problem dimensionality increases. The 25 variable problem, for example, sees a large improvement in the magnitude of the final likelihood when the optimisation is afforded a larger budget.

The dynamic hill climber of Yuret and Maza [1993] performs surprisingly well, outperforming even the Heavy strategy on a number of occasions but the performance reduces significantly on the 25 variable problem. This is rather unsurprising when one considers that the hill climber will perturb a point twice in each dimension during an iteration. As the dimensionality of the problem increases so to does the number of perturbations in each iteration therefore reducing the explorative capabilities of the algorithm. While proving competitive at lower dimensions the dynamic hill climber will struggle at higher dimensions and will therefore not be considered as the basis of the hybrid strategy.

Simulated annealing and the particle swarm both seem capable of locating better likelihood values than the baseline genetic algorithm. Of the two algorithms the particle swarm outperforms simulated annealing in three out of the five tests and comprehensively outperforms it on the 25 variable problem. This is extremely encouraging as the simple swarm proposed by Eberhart and Shi [2000] was employed in these tests. As described in Sections 2.2.5 and 2.2.6 and comprehensively by Bratton and Kennedy [2007], this implementation of the particle swarm is extremely simplistic and takes no advantage of the more recent developments in neighbourhood topologies or fully informed particles which may improve performance. The particle swarm therefore offers the best performance of the basic global optimisers and the most scope for improvement and will therefore form the basis of the subsequent hybridisation.

The simple SQP hyperparameter optimisation performs consistently badly on all cases with the performance reducing rapidly as problem dimensionality increases. This provides a clear demonstration of the advantages of a global strategy when attempting to optimise the likelihood.

Although Table 5.2 compares the actual minimum likelihood found by each algorithm it is the relationship of the corresponding hyperparameters to the quality of an overall optimisation process which is of most interest. In other words, does a better likelihood equate to a better final design? Figure 5.1 displays the average RMS error in pressure obtained by the final aerofoil of the inverse design problem. As in Section 3.4 each optimisation is carried out a total of 50 times and the average taken. These results are presented in tabular form in Appendix C.

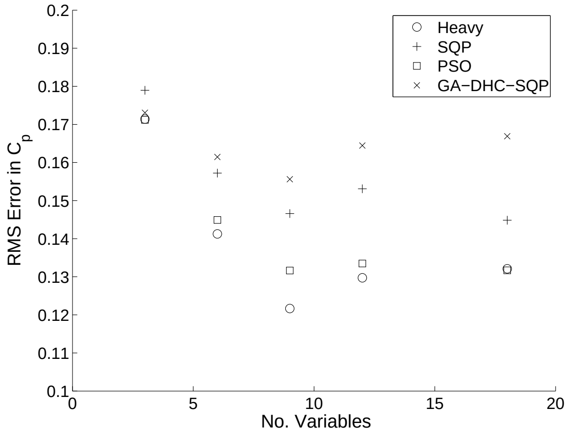


Figure 5.1: Graphical comparison of the effect of each of the hyperparameter optimisers on the overall kriging based optimisation.

Figure 5.1 demonstrates the improvement offered by a more extensive tuning of the hyperparameters. The Heavy strategy outperforms both the particle swarm and the SQP optimisation in the majority of cases. The cheaper particle swarm performs relatively well, but overall the optimisations clearly benefit from the better hyperparameters found by the Heavy strategy.

The results for the SQP strategy indicate that a poorly optimised set of hyperparameters translates to a drop in overall optimisation performance. Commencing a local search from a random point in the likelihood space can lead to the optimisation being trapped in plateaus or converging to unrealistic hyperparameters. Often the "optimum" hyperparameters returned by the SQP search contain a vector of P values equal to one, a vector of θ values equal to 3 or a regression constant of 3. Such values lead to completely unrealistic response surfaces and the poor optimisation performance observed in Figure 5.1.

The convergence of the SQP strategy to unrealistic regions of the hyperparameter space can be countered by providing the search with a good initial solution. This may be possible if the importance of each of the variables is roughly known a priori, if, for example, a similar problem has been encountered before. However, this is rarely the case. In an attempt to counter this a fourth tuning strategy was investigated via the inverse design problem. This strategy, presented as "GA-DHC-SQP" in Figure 5.1, involves an initial optimisation of the hyperparameters via the Heavy strategy with subsequent optimisations using an SQP search. The initial global optimisation aims to provide a good initial set of hyperparameters which can be locally improved upon by the SQP search in subsequent likelihood optimisations. Essentially this is a variation of the "Single Tune" strategy presented in Section 3.3.

The results of Figure 5.1 indicate that this strategy performs consistently worse than any other, with the exception of the simple three variable optimisation. As the dimensionality of the problem increases the designs obtained by the GA-DHC-SQP strategy become increasingly poor compared to those of the other three, with even the simple SQP strategy performing better. These results indicate that not only is the global optimisation of the likelihood necessary for an effective optimisation but so too is a repeated global optimisation as update points are added to the kriging model.

The initial optimisation of the likelihood is based on a sample plan that may not completely represent the true response surface, hence the resulting hyperparameters, although suitable for the initial sampling may change drastically as updates are evaluated. Restricting the hyperparameter optimisation to a local region around the initial set therefore prevents the drastic alterations to the hyperparameters which may be required as the optimisation progresses. As with the single tuning strategy of Section 3.5.4 the performance of this strategy may improve if afforded a larger initial sampling.

The presented results conclusively demonstrate that a repeated global optimisation of kriging hyperparameters is necessary if a kriging based optimisation is to perform to its utmost. However, the timings of Appendix C reinforce the expense of such optimisations. For example, over 14 hours are spent optimising the hyperparameters of a 18 variable design problem using the Heavy strategy. This would cause a significant bottle neck in any optimisation even if high fidelity CFD simulations were used instead of the VGK simulations employed here. Reducing this through the application of a hybrid optimisation strategy employing an adjoint of the likelihood function is therefore an important goal.

5.5 A Review of Existing Hybridised Particle Swarms

The utilisation of a local search within a particle swarm aims to improve the overall convergence of the swarm to a global optimum. Without this addition to the algorithm the particles tend to oscillate around the region of an optimum but make relatively little headway towards finding an exact solution. Local search algorithms typically involve some form of gradient descent and proceed quickly from an initial starting point to a precise minimum. However, they cannot escape from a local minimum unless a restart procedure is employed. Combining the global exploration abilities of the particle swarm with the exploitation abilities of a local search therefore, makes perfect sense.

There exists within the literature a number of different methods of implementing a local search within a particle swarm, each with their own advantages and disadvantages. Victoire and Jayakumar [2004], for example, employed a local search of the problem using sequential quadratic programming, (SQP), commencing from the $\boldsymbol{x}_{Gbest}$ point of a generation. This local search was carried out until convergence, with the $\boldsymbol{x}_{Gbest}$ point then replaced by the result of the SQP optimisation. The velocity of each particle was then updated and another SQP optimisation commenced if the $\boldsymbol{x}_{Gbest}$ found by the swarm was better than that found by the previous SQP optimisation. Guo et al. [2006] employed a similar method to that of Victoire et al. but using a gradient descent method and a constriction factor in the velocity update equation.

Such strategies however, may hamper the global exploration ability of the swarm and result in a premature convergence of the overall optimisation as particles move towards the $\boldsymbol{x}_{Gbest}$ point which may not change for a number of generations. This can be countered by the introduction of a diversity metric and particle repulsion, Riget and Vesterstrø [2002], or through the generation of random particles to increase diversity.

Liu et al. [2005] describe a hybrid particle swarm which utilises a chaotic local search to improve the convergence characteristics of the basic swarm. As with the swarms of Victoire and Guo, the local search was commenced from the best particle in a generation but measures were taken to preserve the diversity of the population. After the evaluation of each particle's objective function a subset of the best particles were retained for the next generation with the remaining population members randomly generated. This is similar in some respects to an elitist genetic algorithm, where the best point is carried through unaltered to subsequent generations.

Whereas this random generation of particles maintains population diversity the method proposed by Liu et al. has no control over where these points are generated and as such new points may be generated in regions of the design space previously visited and subsequently discarded. The chaotic local search also makes no use of sensitivity information which is cheaply available in this case due to the adjoint.

Izui et al. [2007] took the approach of splitting the total population into two subsets, one utilising a pure particle swarm approach while the other has each point iterated until convergence using sequential linear programming (SLP). Both groups are then mixed upon convergence of the SLP optimisation and a new global best member of the swarm subset determined. Whilst effectively using sensitivity information through SLP one could argue that given a fixed evaluation budget for each generation the convergence of a large number of such local searches may reduce the number of available evaluations for any global exploration.

Rather than use a local search strategy, Noewl and Jannett [2004] attempted to adjust the particle swarm update equation to make use of any available gradient information. This resulted in a velocity calculation which adopted a gradient descent term instead of the nostalgia term. The components of the updated velocity were therefore due to the inertia of each particle, the location of the global best point and a step in the steepest descent direction.

While this approach appears attractive by simultaneously exploiting a particle's local gradient and the global best point, its effectiveness is dependant on the cost of the gradient. Consider, for example, the optimisation of the likelihood using this method: a traditional particle swarm with a population of 50 requires 50 likelihood evaluations per generation. But if the cost per optimisation is to be maintained when using the method of Noewl et al. then the population size or number of generations must be reduced accordingly. This reduces either the exploration ability of the algorithm or the convergence. Assuming, for example, that the adjoint is twice as expensive as a simple likelihood calculation immediately reduces the swarm population, or the number of generations by two in order for the total cost to remain consistent. The algorithm therefore requires extremely cheap gradient information to be competitive. One could also argue that just taking a step in the direction of steepest descent is not as efficient as a quasi-Newton based optimisation when it comes to an effective terminal search.

Ninomiya and Zhang [2008] took a similar approach in the development of their hybrid particle swarm. However, rather than using raw gradient information in the velocity update, a step was taken in the direction of a BFGS local search. This step then influenced the velocity of a particles six neighbours through a modified FIPS equation. Ninomiya and Zhang realised that calculating the gradient and subsequent BFGS step for each particle at every iteration would be prohibitively expensive and therefore restricted the local step to only the global best point. Employing a FIPS based swarm with a reduced neighbourhood maintains population diversity to a better extent than the simpler "star" strategy. The hybrid swarm of Ninomiya and Zhang does not therefore suffer the same premature convergence problems as those of Victoire and Jayakumar [2004] or Guo et al. [2006].

The hybrid swarm of Ninomiya and Zhang was based on a random neighbourhood topology, the influence of the local step taken by the global best point on its neighbours, which may in fact be in completely different areas of the design space, could perhaps be considered questionable. The strategy does not include a terminal local search to completely converge the global best point to an optimum.

Another interesting hybrid particle swarm was developed by Wang et al. [2006] which incorporated the features of a number of the other swarms. Like Victoire and Guo a gradient descent was utilised but a reinitialisation of the swarm population was also used to maintain diversity. This method therefore utilised the gradient information very effectively in a single local optimisation but could carry out an effective global exploration without premature convergence problems.

5.6 The Proposed Hybrid Particle Swarm

Reviewing the advantages and disadvantages of a number of hybrid swarms from the literature provides a solid basis upon which to develop an algorithm for the purposes of hyperparameter optimisation. From the above review a number of important features of a final strategy are apparent. A local search must be incorporated into the swarm which uses gradient information in an efficient and effective manner. Precautions must also be taken to prevent premature convergence by maintaining diversity in the population when the results of any local optimisation are fed back into the swarm.

Step	
1	Initialise the population members using a random Latin Hypercube
2	Initialise the particle velocities
3	Evaluate the objective function (concentrated likelihood)
4	Select a population member for refinement via a local search
5	Initialise $\boldsymbol{x}_{Pbest}$ and $\boldsymbol{x}_{Gbest}$
6	Update each particle's velocity and position
7	Reinitialise a proportion of the population to unexplored regions if required
8	Evaluate the objective function
9	Select a population member for refinement via a local search
10	Update $\boldsymbol{x}_{Pbest}$ and $\boldsymbol{x}_{Gbest}$
11	Return to step 6 unless the required number of generations has been reached
12	Terminal local search commencing from $\boldsymbol{x}_{Gbest}$

Table 5.3: Pseudo code of the proposed hybrid particle swarm optimisation algorithm.

To this end a hybrid particle swarm has been developed which uses features from the literature and introduces new features similar to those used in the creation of sampling plans for computational experiments. The proposed algorithm, the pseudo code of which is presented in Table 5.3, begins with the initialisation of every particle's position and velocity. Unlike the majority of particle swarms the initial position of each particle is defined by sampling the space using a random latin hypercube. This distributes the particles more evenly throughout the space when compared to a more traditional random initialisation of points. Each particle is then given a random velocity within the limits of $\boldsymbol{V}_{max}$.

The objective function of each particle is then evaluated as normal but based on this objective function a member of the population is selected for improvement via a local optimisation. Unlike other hybrid swarms the $\boldsymbol{x}_{Gbest}$ is not automatically selected for local improvement, instead the previously calculated objective function is used in a rank selection scheme, similar to that used by a genetic algorithm for the purposes of generating a mating pool.

The use of such a scheme results in a chance that some measure of local improvement is applied to any member of the swarm and not just the global best point. This adds an additional measure of local exploration to the population without the expense of a large number of local searches in each generation. This local optimisation is also not carried out to convergence. Instead a small proportion of the total budget of function evaluations available for each generation is reserved for local improvement. Depending on the magnitude of this proportion the local improvement may be anything from a single to multiple steps towards a local optimum. A similar scheme was employed by Fan et al. [2004] where a Nelder-Mead simplex update was used to replace a single swarm member.

The combination of the rank based selection and partial convergence of any local optimisation also helps to preserve a level of diversity within the population as the optimisation progresses. As previously mentioned complete convergence of the global best point after the initial generation is counter-productive, as subsequent swarm generations tend to move towards that point rather than exploring the space effectively. Here, the view is taken that the effort of a completely converged local optimisation should be spread throughout each swarm generation to boost the local exploitation of the whole population and not just the best point.

Once local improvement steps have been taken, the $\boldsymbol{x}_{Pbest}$ for each particle and overall $\boldsymbol{x}_{Gbest}$ can be calculated and used to update the velocity of each swarm member. Here, the constriction factor method of Eberhart and Shi [2000] is utilised with the resulting velocity restricted to $\boldsymbol{V}_{max}$. The position of each particle can then be updated as normal using Equation 2.14.

Provision is made to prevent particles moving beyond the permitted variable bounds. If the updated position of the particle falls outside the bounds of a variable then the position of that particle is adjusted so the particle lies on the boundary. The corresponding velocity of the particle is then reversed so the inertial component of the velocity update in the next iteration tends to move the particle back from the boundary. A similar approach was used by Huang and Mohan [2005].

A reinitialisation of a proportion of the swarm is carried out before the calculation of the objective function of each particle. This reinitialisation attempts to increase the diversity of the swarm population but unlike the reinitialisation procedure of Wang et al. [2006] the position of any new particle is not random but selected in order to explore regions of the space not previously visited by the swarm or embedded local search. This is achieved through a maximisation of the minimum distance of a proposed restart point to those points previously evaluated, a similar process to that used by Morris and Mitchell [1995] to calculate space filling Latin hypercubes.

The alternative option to a particle reinitialisation procedure is a particle repulsion scheme such as that used by Riget and Vesterstrøm [2002]. In such a strategy when the diversity of the population reaches a predefined minimum the velocity update equation is altered and particles move away from each other in subsequent iterations until diversity has been improved. However, such a repulsion technique offers no guarantees that previously unexplored regions will be searched by the swarm, rather the swarm may expand and contract again within a region of the space which may have been explored during the initial contraction. The repulsion process itself can take a number of generations and may therefore waste objective function evaluations. The strategy proposed here forces the exploration of unexplored regions of the space immediately.

After reinitialisation the objective function of each particle is evaluated and a particle is selected for local improvement as before. The global and personal best locations can then be updated to calculate a particles velocity in the next generation. This process is repeated until the total number of generations is reached after which the global best point is used as the starting point for a terminal local search. The final local optimisation is carried out with the aim of exploiting fully the best point found throughout all of the previous generations of the swarm.

5.7 Optimisation of the Swarm Parameters

Having defined the basic hybrid hyperparameter optimisation strategy, a number of issues still remain concerning the appropriate settings for the hybrid swarm in order to achieve optimal performance. The total number of generations, the size of the population, the point at which the local search should start to improve members of the population, the increase in the degree of local improvement with subsequent generations, the number of evaluations reserved for a terminal local search and the number of points reinitialised, all require investigation.

One way of addressing all of these issues is to cast them in the form of an optimisation problem. To this end the basic hybrid particle swarm strategy described previously has been parameterised using ten variables which adjust the overall structure of the optimiser. These variables can then be adjusted over the course of an optimisation with the objective of improving the hyperparameter tuning performance of the algorithm. A similar procedure was carried out by Keane [1995] when the optimisation of the control parameters of a genetic algorithm were considered in order to improve performance on multi-modal problems.

Each of the ten swarm control parameters considered, (shown in Table 5.4) varies greatly in their effect on the particle swarm, some control the complete nature of the optimiser while others control more subtle features.

The first five control parameters described in Table 5.4 control the complete nature of the optimisation. Given a predefined budget of likelihood evaluations, the first control parameter governs the fraction of this budget reserved for the terminal search. The second control parameter defines the size of the swarm population and therefore the total number of generations. The third control parameter governs the generation at

No.	Parameter description	Lower limit	Upper limit
1	Fraction of total evaluation budget for terminal local optimisation	0	1
2	No. of evaluations per generation	10	100
3	Generation swarm becomes hybridised	1^{st}	Final
4	Initial no. of local search evaluations (fraction of total evaluations)	0	0.5
5	Final swarm size (fraction of evaluations reserved for swarm)	0.1	1
6	Magnitude of $\boldsymbol{V}_{max}$	0	1
7	Initial probability of particle reinitialisation	0	1
8	Final probability of particle reinitialisation	0	1
9	Initial fraction of population reinitialised	0	1
10	Final fraction of population reinitialised	0	1

Table 5.4: Hybrid particle swarm control parameters to be optimised.

which the swarm is hybridised, in essence, the generation a local search is first employed to improve a member of the population. The fourth and fifth parameters define how the number of evaluations per generation used in this local improvement changes as the optimisation progresses.

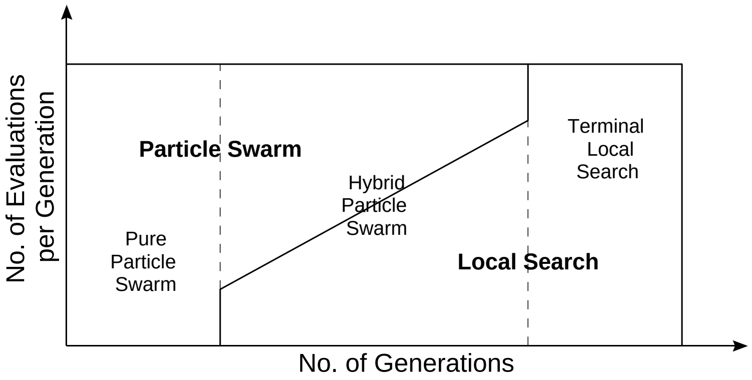


Figure 5.2: A graphical representation of the allocation of resources between the particle swarm and local search within the proposed hybrid optimisation strategy

The effect of these five parameters can be visualised using Figure 5.2. One can observe that adjusting the generation at which local improvement is utilised within the swarm converts the optimiser from the "pure" swarm of Eberhart and Shi [2000] to a hybridised swarm. Likewise, adjusting the proportion of the total evaluation budget used in the terminal local search changes the optimiser from a purely global search to a purely local search from the best point of an initial design of experiments.

The fraction of a population's evaluation budget utilised in a local improvement alters linearly as the optimisation progresses according to parameters four and five. The hybrid optimisation can therefore move from one employing a small local improvement at each generation to one employing an increasing degree of local improvement as the optimisation progresses. This therefore moves the optimisation from a global exploration to a more localised exploitation.

The remaining five parameters control the particle swarm itself. The magnitude of $\boldsymbol{V}_{max}$ controls the incremental nature of the swarm and a particle's ability to escape from local optima. A large $\boldsymbol{V}_{max}$ causes all of the particles to move quickly around the space but perhaps overshooting regions of interest. A small $\boldsymbol{V}_{max}$ helps each particle to exploit regions of interest but may prevent them escaping local minima.

Parameters seven and eight control the probability that a particle reinitialisation will occur as the optimisation progresses. A random number is generated in each generation and compared to this probability, if this number is less than the probability then a reinitialisation occurs. A large initial probability relative to final probability will therefore result in an intense exploration of the space, moving towards a more focused exploitation. A small initial probability on the other hand will result in the swarm progressing towards a more intense exploration in the final generations.

The remaining swarm control parameters control the fraction of the swarm population reinitialised as the optimisation progresses. An optimisation can therefore have no points reinitialised or indeed the whole population. Such an optimisation strategy is an interesting prospect, as when coupled with a terminal local search, this essentially amounts to a space filling sampling of the space and a local optimisation commenced from the best point found.

5.8 Results of the Swarm Parameter Optimisation

Each of the previously described swarm parameters were optimised using the classic response surface method. A kriging model was constructed from an initial 100 point DOE of the 10 swarm control parameters. Twenty batches of up to 20 updates were then evaluated and added to the model. Throughout the course of the optimisation the Heavy tuning strategy was employed in tuning the hyperparameters with the updates based on a search of the kriging models prediction of the performance of the swarm using a genetic algorithm. Each particle swarm was assessed using the mean improvement in likelihood over the baseline genetic algorithm, described in Section 5.3.

A series of optimisations were carried out for different likelihood evaluation budgets and for underlying optimisation problems of different complexities. A total of five different levels of complexity of the inverse design problem were considered from two to 25 variable problems. Naturally the number of hyperparameters that each potential swarm evaluates is double this. For each problem a number of different likelihood evaluation budgets were considered from 100 up to 2000 evaluations. These optimisations produce a Pareto front, Figure 5.3, demonstrating the performance of the hybrid strategy as more effort is applied to the optimisations. Analysing the results of each optimisation therefore provides an interesting indication as to how the nature of the optimisation should change as one moves from an extremely quick to a slower more methodical optimisation.

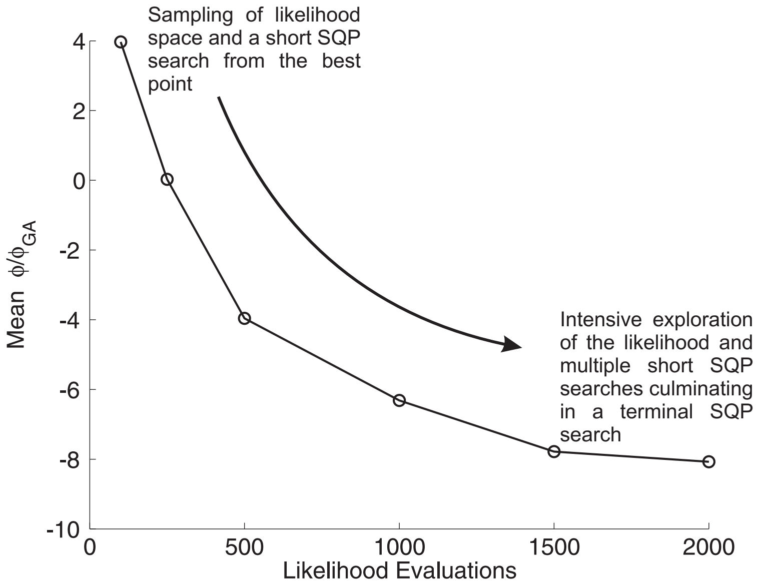


Figure 5.3: A pareto front showing the change in performance and the nature of the optimisation as the magnitude of the likelihood evaluation budget increases for the 25 variable optimisation

We consider first the results obtained by the swarm control parameter optimisation when the likelihood evaluation budget is small. Table 5.5 presents the swarm control parameters for the best performing swarms on the five optimisation problems considered. On all problems the optimisation favours only a single generation of the swarm followed by a single SQP search, all of the remaining swarm parameters are irrelevant in such a case and are therefore not presented in Table 5.5.

No. of Variables.	Population Size	No. of Generations	Objective Function
2	90	1	0.048
5	98	1	0.126
10	89	1	0.402
15	98	1	0.491
25	89	1	3.968

Table 5.5: Parameters for the best performing hybrid swarm given a range of problem dimensionalities and a fixed budget of 100 evaluations.

The results of Table 5.5 indicate that the optimisation favours an initial sampling of the likelihood space using quite a large proportion of the available evaluation budget. This leaves a relatively small number of evaluations for a terminal SQP search commencing from the best point found in this initial sampling. The performance results presented in Table 5.5 do not compare favourably to those of the baseline genetic algorithm but fare much better than the SQP strategy, Table 5.2. This indicates that, in terms of likelihood optimisation, the starting point of the local search is much more important than the effort applied to that local search; a result which corresponds well with the observations regarding the performance of the GA-DHC-SQP optimisation, Figure 5.1.

Table 5.6 defines the best parameters when a total of 2000 likelihood evaluations are employed in the hyperparameter optimisation, the 10 best of each case are presented in Appendix D. Figure 5.3 demonstrates that after 1500 evaluations there is only a slight improvement in the magnitude of the likelihood compared to when 2000 evaluations are employed, we therefore assume that there is no advantage to employing more than 2000 evaluations. The results of the these optimisations will therefore form the basis of a more general hybrid strategy.

No. of Vars.	Pop. Size	No. Gens	V_{max}	Local Search Gen.	Min. Local Evals.	Min. Swarm Size	Prob of Reinitialisation		No. of Points Reinitialised		Obj. Func.
							(Initial)	(Final)	(Initial)	(Final)	
2	32	50	0.810	39	3	17	0.05	0.50	0.80	0.52	-0.090
5	61	29	0.092	29	6	7	0.58	0.59	0.10	0.36	-1.115
10	63	25	0.045	15	1	50	0.44	0.63	0.84	0.005	-3.149
15	20	87	0.073	56	6	3	0.52	0.19	0.86	0.56	-4.038
25	15	87	0.064	22	6	3	0.82	0.27	0.43	0.11	-8.075

Table 5.6: Parameters for the best performing hybrid swarm given a range of problem dimensionalities and a fixed budget of 2000 evaluations.

Increasing the budget of available likelihood evaluations results in a significantly different optimisation involving considerably more global exploration of the likelihood space. Table 5.6 demonstrates a much more effective use of the particle swarm with a considerable number of generations employed throughout the optimisation.

Observe that the SQP search is not employed within the first generation but rather at some point in the middle of the optimisation. This suggests that an initial search of the likelihood space via a particle swarm before hybridisation is much more effective than employing a hybridised search from the start. The local search is therefore wasted if applied too soon before the swarm has found a promising region. This corresponds with the results of the SQP search of Section 5.4 where the search frequently stalled on plateaus in the likelihood space. As a particle is selected for local improvement in the hybrid strategy via a rank based scheme, a poor particle may be selected and the same stalling may be observed at the beginning of the optimisation. Commencing local searches part way through the optimisation therefore reduces the chances of the optimisation stalling and wasting valuable likelihood evaluations.

The results of the parameter optimisations indicate that an initial local search of approximately six likelihood evaluations is favourable. This makes perfect sense when one considers the cost of calculating the likelihood and its gradients is approximately twice that of a single likelihood evaluation. A minimum budget of six evaluations for the SQP search should therefore result in at least one step towards a local optimum.

The presented results also indicate a gradual increase in the level of effort expended in the local improvement of particles as the optimisation progresses. Each optimisation also concludes with a significant terminal search from the best point found so far by the optimisation.

Reinitialisation of points takes place throughout the course of all of the best performing optimisations. The results of Table 5.6 indicate that reinitialisation tends to occur towards the start of the optimisation with more points and the probability of reinitialisation generally higher. This corresponds with the commencement of the local searches midway through the optimisation. Once again the optimiser is attempting to rapidly explore the likelihood space before applying a local search. One must also note that 100% of the population is never reinitialised, this leaves some of the particles to gravitate towards the best current design point in the manner of a traditional particle swarm.

The performance results of Table 5.6 and Appendix D indicate that the hybrid strategy can out-perform the baseline genetic algorithm substantially and has the capability of significantly outperforming the 5,000 iteration GA, 5,000 iteration DHC and Heavy strategy when the underlying problem has more than 15 variables.

5.9 Definition of the Proposed Hybrid Strategy

The results of Table 5.6, while indicating the general form that the final hybrid strategy should take, are quite different depending on the dimensionality of the underlying problem. The optimum hybrid swarm for the two variable problem is quite different to that for the 25 variable problem. A set of controlling swarm parameters which is optimal over all cases is therefore difficult to achieve. Instead emphasis is given to the more difficult cases, namely the 15 and 25 variable problems.

Based on the top ten performing strategies for both of these cases, shown in Appendix D, the hybrid strategy defined in Table 5.7 was selected.

			Local	Min.	Min.	Prob of		No. of Points	
Pop.	No.		Search	Local	Swarm	Reinitialisation		Reinitialised	
Size	Gens	V_{max}	Gen.	Evals.	Size	(Initial)	(Final)	(Initial)	(Final)
20	80	0.075	30	6	3	0.7	0.2	0.75	0.1

Table 5.7: Controlling parameters of the proposed hybrid particle swarm.

The presented swarm control parameters result in an initial particle swarm which is skewed towards an initial extensive exploration of the likelihood space with a high probability of reinitialisation and high fraction of the population reinitialised to regions not previously explored. The local optimisation commences once this exploration has reduced, after 30 generations, and begins with an initial effort equivalent to six likelihood function evaluations. As per the results presented in Table 5.6, a substantial fraction of the overall number of evaluations, 20%, is reserved for a terminal local search.

Essentially this results in a strategy similar to the dynamic hill climber described in Section 2.2.3. A local search is used to exploit regions of interest with exploration simultaneously occuring of previously unexplored regions via the reinitialisation procedure and the particle swarm update process.

5.10 Hybrid Strategy Performance

The performance of the proposed hybrid optimisation strategy with respect to the tuning of the hyperparameters of the inverse design problem is presented in Table 5.8. The results demonstrate an obvious reduction in performance relative to the strategies of Table 5.6 which is unavoidable given the more general nature of the final set of control parameters.

No. of Variables	Mean	Std.
2	-0.0404	0.087
5	-0.787	1.933
10	-2.675	3.366
15	-3.667	4.031
25	-6.911	4.497

Table 5.8: Mean improvement in likelihood function over a range of problem sizes for the proposed hybrid optimisation strategy.

The proposed strategy does still however, comprehensively outperform the baseline genetic algorithm on all problems, the particle swarm and the GA-DHC strategy on the 10, 15 and 25 variable problems and simulated annealing on the 5, 10, 15 and 25 variable problems. The strategy even outperforms the expensive Heavy strategy on the 15 and 25 variable problems. This improved performance comes at a 60% reduction in cost over the PSO, GA-DHC and SA strategies and at an 80% reduction over the Heavy strategy.

We now consider the performance of this strategy with regard to the inverse design problem. The results are presented graphically in Figure 5.4 and in Appendix C. Naturally due to the global nature of the optimisation, the hybrid strategy outperforms the SQP strategy of Section 5.4 on every problem. The strategy outperforms PSO tuning on three out of five cases and exceeds the performance of the Heavy strategy on the three and nine variable problems while approaching its performance on the 6, 12 and 18 variable problems.

A modified version of the GA-DHC optimisation, which uses a total of 2000 likelihood evaluations is included in Figure 5.4 to provide a direct comparison, in terms of overall cost, to the proposed hybrid strategy. Once again the hybrid strategy performs better on three out of five cases. Most importantly the hybrid approach outperforms the reduced cost GA-DHC strategy on the 18 variable problem, the most complex considered in this comparison.

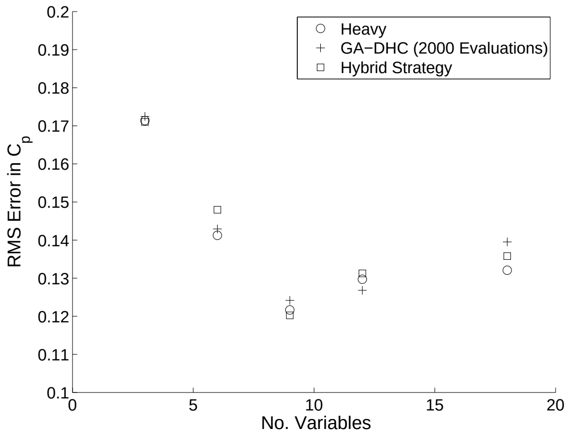


Figure 5.4: Graphical comparison of the effect of the hybrid hyperparameter optimisation relative to the Heavy and reduced cost GA-DHC optimisation strategies.

We now consider the application of the proposed hybrid tuning strategy to an optimisation completely unrelated to the inverse design problem considered so far throughout this chapter. The RAE-2822 aerofoil is optimised for minimum drag at a fixed lift coefficient of 0.3 at Mach 0.65 and a Reynolds number of 6×10^6. The aerofoil is parameterised via two NURBS curves, one representing the upper and one the lower surface, as shown in Figure 5.5. Fourteen control points are permitted to vary in both the x and y directions while the control points on the leading edge are fixed in the x direction resulting in a total of 30 variables.

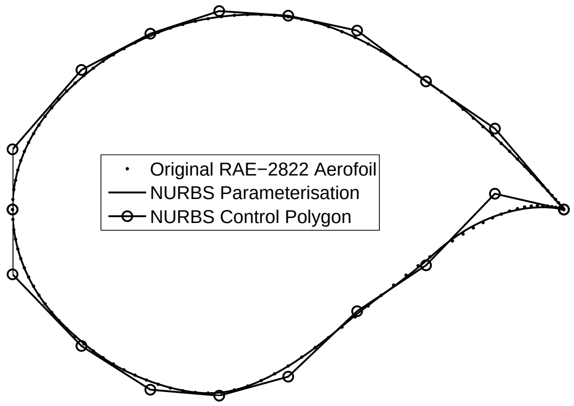
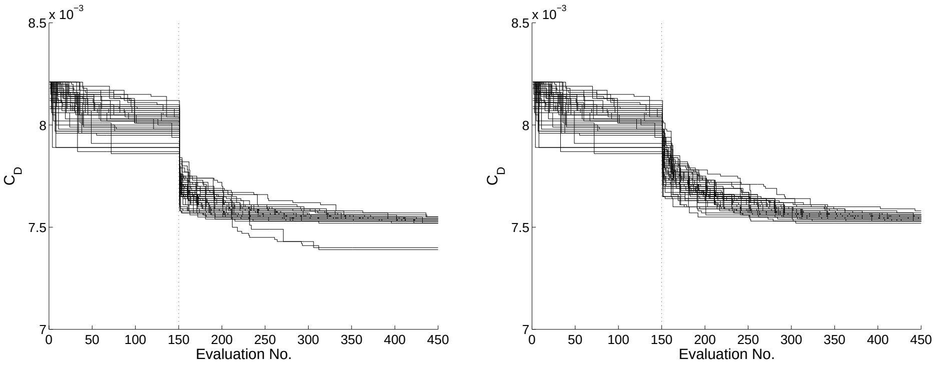


Figure 5.5: Thirty variable NURBS parameterisation of the RAE-2822 aerofoil.

The optimisation is permitted a total of 450 objective function evaluations, of which 150 form the initial DOE. A typical function evaluation requires between three and four VGK simulations at different angles of attack to achieve the required lift, usually to within ±0.005. A total of 30 batches of 10 updates are added to the model as the optimisation progresses with the hyperparameters tuned after every update using either the hybrid particle swarm or the reduced cost GA-DHC strategy. An identical DOE is used in each case thereby providing a meaningful comparison between the final results. A total of 50 optimisations are carried out for each case with the DOE varying each time, the optimisation histories for each are presented in Figure 5.6 with the average drag of the final best designs presented in Table 5.9.



(a) (b)

Figure 5.6: Optimisation search histories resulting from a minimisation of drag for fixed lift using the Hybrid (a) and the reduced cost GA-DHC (b) tuning strategies.

Tuning Strategy	Average C_D	Standard Deviation C_D	% Improvement Over RAE-2822
Hybrid	7.53×10^{-3}	2.88×10^{-5}	8.29%
GA-DHC	7.54×10^{-3}	2.04×10^{-5}	8.13%

Table 5.9: Comparison of the quality of the designs resulting from the optimisation of the RAE-2822 using two different tuning strategies.

The results of Table 5.9 indicate that there is little between the two strategies in terms of the quality of the final aerofoil designs. Both the GA-DHC and the hybrid strategy achieve designs with an average drag coefficient of just over 7.5×10^{-3}. The results indicate that the hybrid strategy does perform slightly better but this could be due to the effect of the two outlying designs observed in Figure 5.6(a) on the overall averages. Likewise the standard deviation associated with this strategy is elevated due to these designs. Both outlying designs exhibit a rapid and non-physical pressure oscillation close to the leading edge on the upper surface. This suggests that the simulation may be inaccurate and that the resulting drag coefficient cannot be relied upon. These optimisations have therefore exploited a weakness in the computational simulation and in doing so have achieved a false optimum. This represents a significant danger when any complex computer code is blindly relied upon during an optimisation.

Removing these outlying designs from both sets of results leads to the hybrid strategy achieving designs with an average C_D of 7.535×10^{-4} and a standard deviation of 7.71×10^{-6}. The reduced cost GA-DHC obtains a slightly higher average drag of 7.545×10^{-4} and standard deviation of 1.22×10^{-5}. The hybrid strategy could therefore be considered to result in slightly better and more consistent designs than the reduced cost GA-DHC strategy.

This can be explained somewhat upon comparison of the optimisation histories of Figure 5.6 and the average convergence histories of Figure 5.7. Here one can observe a much more rapid reduction in the magnitude of the objective function proceeding the DOE when the hyperparameters are tuned via the hybrid strategy. The optimisation therefore converges much more quickly to the approximate region of an optimal design. Subsequent updates to the model are therefore spent fine tuning the optimal solution resulting in the reduction in the variance between final designs. For this particular optimisation problem the hybrid strategy achieves the average drag coefficient of the GA-DHC strategy after approximately 350 objective function evaluations. Although this optimisation utilised VGK simulations in the evaluation of the objective function, a reduction of approximately 100 simulations is quite significant.

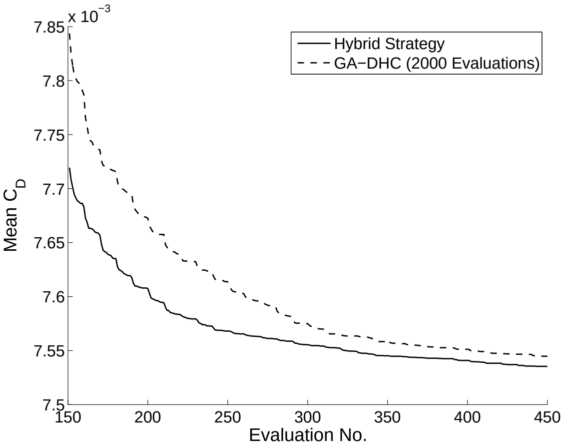


Figure 5.7: Average convergence of the kriging based optimisations based on the hybrid and reduced cost GA-DHC tuning strategies.

5.11 Conclusions

An efficient hybrid optimisation algorithm has been developed which effectively utilises the adjoint formulation of the concentrated likelihood function presented in Chapter 4. The final algorithm combines a basic particle swarm with an SQP search and a particle reinitialisation procedure based on techniques for developing space filling sampling plans.

The final structure of the hybrid swarm was defined through a series of optimisations of the swarm's control parameters with the aim of maximising the algorithms performance with respect to likelihood optimisation. The swarm parameters were optimised for problems of differing complexity and for different budgets of likelihood evaluations. The results of these optimisations demonstrate, not only how the performance of the algorithm changes as more effort is applied, but also how the very nature of the algorithm alters in order to make effective use of the available budget.

These optimisations lead to an algorithm which intensely explores the likelihood space at the beginning of the optimisation and then commences short local exploitations of promising regions after a number of generations. This local exploitation increases with each generation and concludes with a terminal search. In essence the algorithm is similar in its operation to dynamic hill climbing but makes effective use of the available gradient information. The whole optimisation employs the equivalent of 2000 likelihood evaluations, a 80% reduction over the Heavy strategy employed in Chapter 3 but achieves comparable results.

The hybrid strategy outperforms the baseline GA on all problems, SA on four out of five problems and the GA-DHC and PSO on three out of five problems with respect to optimising the likelihood. The strategy also outperforms the Heavy strategy on the two most complex problems. With respect to its performance within a complete optimisation, the hybrid strategy outperforms the PSO on three out of five cases and the Heavy strategy on two out of five and approaches the performance of the Heavy strategy on the remaining problems.

A final 30 variable optimisation, unrelated to the inverse design problem used in the development of the strategy, demonstrated both an improvement in the quality and consistency of the final designs obtained over a GA-DHC strategy of equivalent cost. The hybrid strategy also demonstrated a considerable acceleration in the convergence of the optimisation, achieving the final design of the reduced cost GA-DHC strategy with 100 function evaluations to spare.

The investigations of other tuning strategies comprehensively demonstrated the necessity of a global optimisation of the kriging hyperparameters throughout the course of a kriging based optimisation. Continual local optimisation via SQP or local improvement to an initial set of globally optimised hyperparameters proved detrimental to the kriging optimisations performance. The results of the hybrid swarm parameter optimisation go some way to confirm these findings by demonstrating that for a small budget of likelihood evaluations it is more important, in terms of the likelihood achieved, to select an appropriate starting point for a partial local search rather than to completely converge a terminal search from a poorer initial point.

In conclusion the developed hybrid hyperparameter tuning strategy offers performance close to or exceeding that of the traditional Heavy strategy but for an 80% reduction in tuning cost.

Chapter 6

Geometric Filtration Via Proper Orthogonal Decomposition

6.1 Introduction

The optimisation of complex geometries is prevalent throughout the field of aerodynamic design, from the optimisation of two dimensional aerofoils, to the optimisation of complete wings and aircraft. Designers are, however, restricted somewhat in the parameterisation of such shapes. A typical aerofoil parameterisation, for example, may have anything from a few to more than 30 variables, Hicks and Henne [1978], Sobieczky [1998], Lépine et al. [2001], Song and Keane [2004], Painchaud-Ouellet et al. [2006], Kulfan [2008]. Parameterise an aircraft wing using a series of such aerofoils and the number of variables can quickly be in the order of hundreds. Such large numbers of variables naturally restrict the ability of an optimisation algorithm to achieve a globally optimal design given a limited budget of simulations.

To enable the efficient optimisation of such geometries it is typical for a designer to employ some method of screening to identify those variables which influence the objective function most. With these variables identified the optimisation can proceed using the reduced variable set. Painchaud-Ouellet et al. [2006], for example, reduced a 34 variable NURBS parameterisation of an aerofoil to a total of 11 variables, while Song and Keane [2007], reduced a 33 variable parameterisation of an engine nacelle to seven important variables. Such large reductions in the complexity of an optimisation are especially beneficial when the objective function is evaluated using an expensive high fidelity computational simulation. The expense of such simulations, even with the growth of parallel computations, prohibits the exhaustive search of design spaces with large numbers of variables. In such problems the reduction of the design space offered by variable screening is invaluable, allowing the designer to use a specified simulation budget more effectively.

Variable screening does however suffer from a number of significant problems. The cost of screening a complex optimisation problem which requires evaluations using high fidelity simulations may rival or exceed that of the optimisation of the reduced variable set. Given that the screening procedure may not in fact return a definite set of variables which contribute most significantly to the change in objective function, the screening budget might be better used in the optimisation of the original complete variable set. Variable screening also results in a reduction in the flexibility of the geometry parameterisation, reducing the ability of the optimisation to achieve certain high performance designs which may only be found with the original, complete, variable set.

Optimisation of complex geometries with large numbers of variables and a limited simulation budget therefore equates to a simple trade-off. The designer can attempt to optimise the original parameterisation and potentially achieve a high performance design, though with a limited budget this optimal design will be difficult to locate. Alternatively designers can spend a proportion of their simulation budget to screen out unimportant or insignificant variables and optimise the reduced design space using the remainder of the budget. The optimal design within the reduced space may be easier to locate, but the actual design may be significantly worse than that which could be obtained using the complete variable set. In the worst case scenario the variable screening procedure may indicate no significant difference in the importance of each variable. A sub-optimal reduced set of variables may therefore be chosen, significantly hampering the optimiser's ability to achieve a good design.

In light of the above it is desirable to have an optimisation strategy which combines the advantages of design space reduction associated with variable screening, whilst retaining the majority of the flexibility of the original optimisation. To this end the current chapter describes a strategy which combines surrogate modeling techniques and proper orthogonal decomposition (POD) in an attempt to optimise aerodynamic problems consisting of large numbers of variables but given only a limited simulation budget. Within the framework of this optimisation strategy, proper orthogonal decomposition is used to reparameterise the problem in an attempt to filter out badly performing geometries.

The proposed strategy, termed geometric filtration, operates, not on the variables of the original optimisation problem but on the geometry of the resulting designs. This, therefore, is the key difference between this strategy and traditional variable screening approaches such as that of Morris [1991], Welch et al. [1992] and Trocine and Malone [2000]. Whereas these approaches are restricted to the original variables, by operating at a geometric level the proposed strategy can combine similar features resulting from different variables and therefore reduce the overall variable count. By considering only the best performing designs the method captures the features which produce those designs thereby removing, to some degree, the features which produce poor designs. This focuses the optimisation and once again reduces the number of variables.

6.2 Natural Geometric Filtration

Aerodynamicists already employ a form of geometric filtration, perhaps without even realising it, for some aerodynamic optimisation problems. Consider, for instance, the design of an aerofoil for optimal performance at transonic speeds. Here a designer utilises their knowledge of existing aerofoil shapes and bases the optimisation on an existing supercritical aerofoil, such as the RAE-2822. An optimisation for a particular transonic flight condition therefore results in a series of perturbations to this baseline geometry, be it through the manipulation of NURBS control points or through the addition of analytical functions. The choice of this baseline aerofoil has immediately filtered out badly performing designs and in doing so has reduced the design space before even commencing the parameterisation.

Robinson and Keane [2001] used this concept to construct an aerofoil parameterisation based on a series of orthogonal bases derived from an ensemble of supercritical aerofoils. This parameterisation technique was based upon the principle that aerofoils which perform well at a particular flight condition, the transonic regime in this case, have a number of common geometric features which can be extracted through the orthogonalisation process and used for optimisation. This is perfectly feasible in the case of 2D aerofoil design, where large databases of the performance of different aerofoils exist, Abbott and Von Doenhoff [1960] being a popular example. Problems arise when an optimisation problem is encountered for which there is no such literature. There is, for example, no similar database of the performance of different wing body fairings, the best designs of which could be used to construct a similar set of orthogonal bases. While the generation of such basis functions is undoubtedly a useful approach, this method cannot be applied to every aerodynamic design problem.

6.3 Overview of the Proposed Methodology

The proposed optimisation strategy, shown in Figure 6.1, begins with an initial surrogate based optimisation, in this case employing a kriging model. This initial optimisation utilises the designer's initial geometry parameterisation and a proportion of the total simulation budget. This is followed by a reparameterisation procedure using proper orthogonal decomposition which attempts to both reduce the number of design variables and filter out badly performing designs. A secondary surrogate model optimisation is then performed utilising this new geometry parameterisation and the remainder of the total simulation budget.

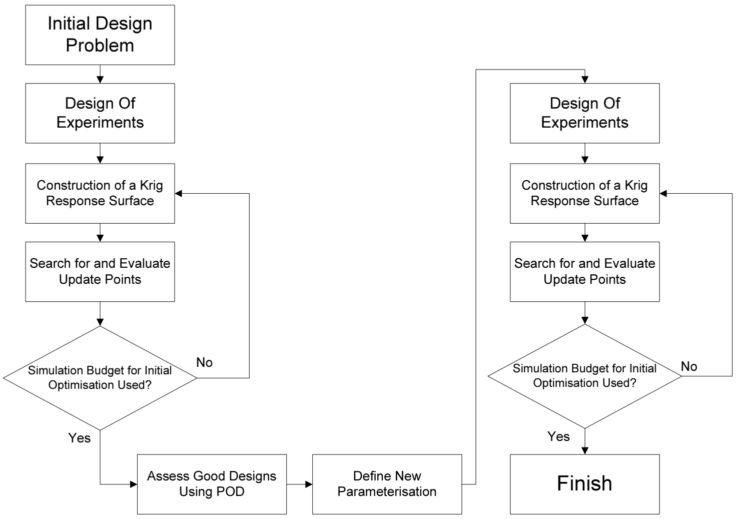

Initial Design Problem
Design Of Experiments
Construction of a Krig Response Surface
Search for and Evaluate Update Points
Simulation Budget for Initial Optimisation Used?
No
Yes
Assess Good Designs Using POD
Define New Parameterisation
Design Of Experiments
Construction of a Krig Response Surface
Search for and Evaluate Update Points
Simulation Budget for Initial Optimisation Used?
No
Yes
Finish

Figure 6.1: An overview of the geometric filtration optimisation methodology

In the context of the proposed optimisation strategy, the initial surrogate model optimisation followed by the POD, performs a job similar to that of traditional variable screening, except that the screening is performed at the geometric instead of the variable level. This prevents the reduction in variables being restricted to the those of the original parameterisation. It is intended that the initial optimisation provides a number of good designs which are then decomposed using POD into a series of orthogonal bases which capture the common features of these good designs. A similar process has been used by Kamali et al. [2007] to reduce a design space but using points from a Latin hypercube sampling of the design space and not a subset of points from an initial optimisation.

As shown throughout the literature, LeGresley and Alonso [2003], Bui-Thanh and Willcox [2004], POD cannot accurately represent data outside of the initial snapshot ensemble. Here, this particular feature of POD, combined with an initial search offers an advantage. By considering only the current best geometries in the construction of the orthogonal bases, the bad geometries cannot be recreated through a combination of POD bases and have therefore been filtered out. The second surrogate model optimisation therefore has the benefit of a reduced number of variables with a minimal reduction in geometric flexibility which focuses on designs that perform well.

6.4 Introduction to Proper Orthogonal Decomposition

Proper orthogonal decomposition (POD), otherwise known as Principal Component Analysis, or Karhunen-Loève expansion, has been extensively used throughout engineering with regard to computational fluid dynamics (CFD). It has been used in the derivation of reduced order models for the purposes of control, Hung and Hien [2001], Lewin and Haj-Hariri [2005], Siegal et al. [2006], Gross and Fasel [2007], optimisation, LeGresley and Alonso [2000], Li et al. [2007], My-Ha et al. [2007] and has also been used as a flow analysis tool, Kim et al. [2005]. POD decomposes a series, or ensemble, of snapshots of data into a set of optimal orthogonal basis functions of decreasing importance. The basis functions are optimal in the sense that no other set of basis functions will capture as much information in as few dimensions, Lucia and Beran [2004]. Applying POD, therefore, to an ensemble of aerofoils results in a series of orthogonal bases similar to those of Robinson and Keane [2001].

The decomposition process, as per Sirovich's method of snapshots, Sirovich [1987], begins with the definition of an ensemble of snapshot vectors, $\boldsymbol{S}$. In the case of the geometric optimisations considered here, the snapshot ensemble is constructed from a series of M vectors consisting of the x and y coordinates of a number of designs selected from the initial optimisation i.e. the strategy is working at the geometry level and not the design variable level,

$$\boldsymbol{S} = [\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_M]. \tag{6.1}$$

The matrix of snapshots is then decomposed into a mean, $\bar{\boldsymbol{s}}$ and a matrix of the fluctuations of each snapshot from this mean,

$$\boldsymbol{S} = \bar{\boldsymbol{s}} + \boldsymbol{D} \tag{6.2}$$

$$\text{where} \qquad \bar{\boldsymbol{s}} = \frac{1}{M}\sum_{i=1}^{M} \boldsymbol{s}_i. \tag{6.3}$$

The orthogonal basis functions are calculated by considering the solution to the eigenvalue problem,

$$\boldsymbol{C}\boldsymbol{E} = \boldsymbol{E}\boldsymbol{\Lambda}, \tag{6.4}$$

where $\boldsymbol{\Lambda}$ is a vector of eigenvalues and the square symmetric correlation matrix $\boldsymbol{C}$ is given by,

$$\boldsymbol{C} = \boldsymbol{D}^T\boldsymbol{D}. \tag{6.5}$$

The matrix of eigenvectors, $\boldsymbol{E}$, can then be used to calculate the matrix of eigenfunctions,

$$\boldsymbol{\Omega} = \boldsymbol{D}\boldsymbol{E}, \tag{6.6}$$

where $\boldsymbol{\Omega}$ is a matrix of M eigenfunctions. These eigenfunctions, along with the corresponding vector of modal coefficients, $\boldsymbol{\alpha}$, allow the fluctuations and hence the original snapshots to be reconstructed,

$$\boldsymbol{s}_i = \bar{\boldsymbol{s}} + \boldsymbol{\Omega}\boldsymbol{\alpha}_i \tag{6.7}$$

The advantage of POD is that not all of the M bases (eigenfunctions) are necessary to recreate the original snapshot ensemble to a required degree of accuracy. The cumulative percentage variation, Jolliffe [2002], can be used to define a reduced number of basis functions with which the original ensemble can be approximately recreated. The importance of each POD basis function is related to the relative magnitude of the corresponding eigenvalue, a large eigenvalue therefore indicates an important basis function. The cumulative percentage variation,

$$\frac{\sum_{i=1}^{N}\lambda_i}{\sum_{i=1}^{M}\lambda_i} \times 100, \tag{6.8}$$

is therefore a measure of the combined importance of the first N bases. Using this simple calculation a reduced number of bases can be selected in order to meet a minimum required percentage variation. Using this reduced number of bases, the original snapshot vectors can be approximated by,

$$\boldsymbol{s}_i \approx \bar{\boldsymbol{s}} + \sum_{n=1}^{N} \boldsymbol{\alpha}_{i_n}\boldsymbol{\omega}_n, \tag{6.9}$$

where $\boldsymbol{\omega}_n$ and $\boldsymbol{\alpha}_{i_n}$ are the N most important basis vectors and corresponding modal coefficients for the i^{th} snapshot of the original ensemble. The modal coefficients corresponding to each of the original snapshot vectors can be calculated using the orthonormality property of the basis vectors.

The geometric filtration optimisation methodology therefore moves from a surrogate modelling optimisation based on the magnitude of the original design variables to one which considers the magnitude of the POD modal coefficients. The bounds of the secondary optimisation are defined by the minimum and maximum modal coefficients of the original snapshot ensemble. The optimisation problem therefore moves from the formulation,

$$\begin{gathered} \text{Minimise} \qquad y(\boldsymbol{x}) \\ \text{subject to} \\ \boldsymbol{l}_i \leq \boldsymbol{x}_i \leq \boldsymbol{u}_i, \end{gathered} \tag{6.10}$$

in d dimensions to one in N dimensions,

$$\begin{gathered} \text{Minimise} \qquad y(\boldsymbol{\alpha}) \\ \text{subject to} \\ \boldsymbol{\alpha}_{\min_i} \leq \boldsymbol{\alpha}_i \leq \boldsymbol{\alpha}_{\max_i}, \end{gathered} \tag{6.11}$$

where $\boldsymbol{\alpha}_{\min_i}$ and $\boldsymbol{\alpha}_{\max_i}$ denote the minimum and maximum modal coefficient of the original snapshot ensemble corresponding to the i^{th} basis vector. Provided that N is less than d, this reduces the complexity of the resulting optimisation problem.

6.5 Optimisation of a Transonic Aerofoil

6.5.1 Description of the Problem

To demonstrate the effectiveness of the proposed strategy the optimisation of a 2D aerofoil is used as a test case. The RAE-2822 aerofoil is parameterized using two NURBS curves, one each for the upper and lower surface with the positions and weights of the controls points defining both NURBS curves optimised via a local BFGS optimisation as per Lépine et al. [2001]. The resulting parameterisation of the RAE-2822, shown in Figure 6.2, consists of a total of 20 variables. The control points at the leading and trailing edges are fixed, the two control points on the line, $x = 0$, are only permitted to move vertically to maintain curvature continuity at the leading edge, while the remaining control points can move in both axes. Bounds are placed on the movement of each of the control points to help prevent unrealistic designs. Although the weight of each control point remains fixed for the purposes of the following optimisations they could be permitted to vary, increasing the dimensionality of the problem to 31 variables.

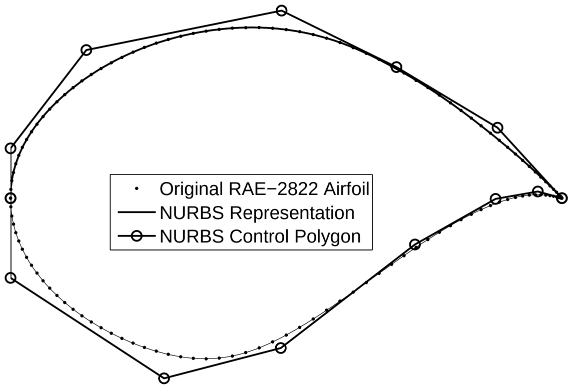
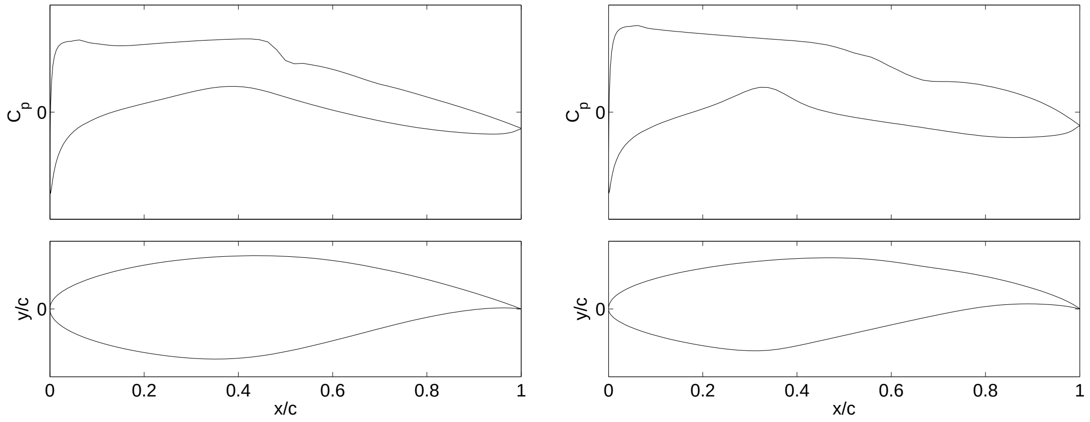


FIGURE 6.2: Twenty variable NURBS parameterisation of the RAE-2822 aerofoil

The aerofoil is optimised to minimise the drag to lift ratio at Mach 0.725, Reynolds number of 6×10^6 and a fixed angle of attack of 2°, using the full potential solver VGK, ESDU [1996b]. At these flow conditions the RAE-2822 has a drag to lift ratio of 0.0148 equating to a lift to drag ratio of 67.8, and exhibits a noticeable upper surface shockwave just before the mid-chord point, Figure 6.3(a).

The speed of the VGK solver, approximately one second per simulation, allows extensive averaging to be carried out thus giving a more accurate picture of the performance of both the traditional kriging and geometric filtration approaches to design optimisation.



(a)

(b)

Figure 6.3: Pressure distribution and geometry of the original RAE-2822 aerofoil with $\frac{C_D}{C_L} = 1.48 \times 10^{-2}$ or $\frac{C_L}{C_D} = 67.8$ (a), and for an example aerofoil resulting from an optimisation utilising a genetic algorithm followed by dynamic hill climber with $\frac{C_D}{C_L} = 1.08 \times 10^{-2}$ or $\frac{C_L}{C_D} = 93.0$ (b)

6.5.2 Direct Optimisation Using a Genetic Algorithm

The speed of the objective function evaluations also allows a number of direct searches of the design space using a genetic algorithm, (GA). Stochastic methods, such as the genetic algorithm, provide a reliable way of locating the region of the global optimum within a design space, however, as they typically require a large number of function evaluations, it is not generally feasible to use such methods when expensive high fidelity simulations are required. Applying a genetic algorithm to the above aerofoil optimisation problem provides a useful indication as to the true optimum which both the geometric filtration strategy and the traditional kriging strategy are attempting to attain.

The aerofoil design problem was optimised using a GA followed by a dynamic hill climber (DHC), both implemented using the OptionsMatlab design exploration system, Keane [2003]. A budget of 10,000 function evaluations was used in the optimisation, with the budget split evenly between the GA and DHC. The 5,000 available function evaluations for the genetic algorithm equated to 100 generations of 50 points each. Although genetic algorithms are typically very good at locating the general region of the global optimum, they can be very slow to converge to a precise answer. The DHC is therefore used to converge the optimisation towards a more accurate solution.

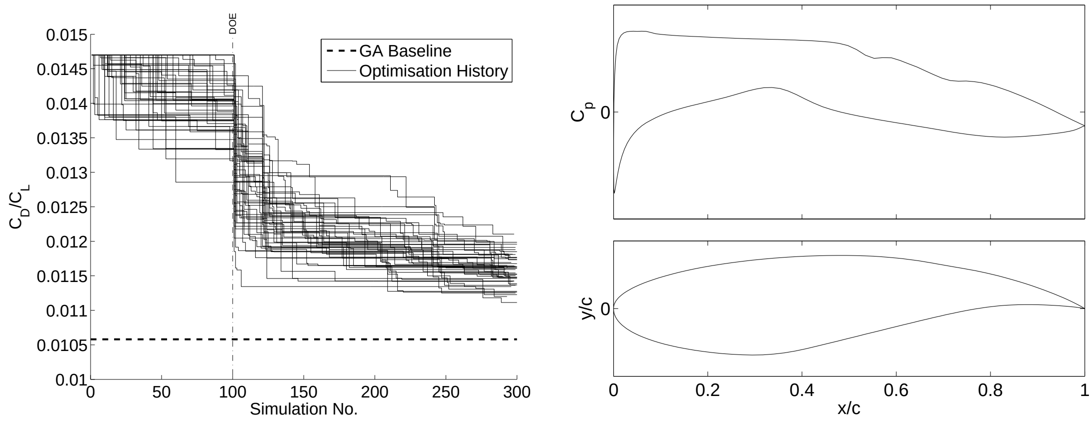
This extensive optimisation was carried out a total of 10 times, producing an average $\frac{C_D}{C_L}$ of 1.06×10^{-2} with a standard deviation of 4.59×10^{-4} which equates to a $\frac{C_L}{C_D}$ of 94.7, an improvement of some 28.4% over the original RAE-2822. Figure 6.3(b) provides an indication of the designs this extensive optimisation produced. Upon comparison to the original aerofoil, Figure 6.3(a), one can clearly see a complete removal of the upper surface shockwave, a reduction in the upper surface pressure and an increase in the lower surface pressure, which results in an overall decrease in the drag to lift ratio. The optimisation process has resulted in a reduction to the leading edge curvature and an overall reduction in thickness. The trailing edge camber of the aerofoil has also been increased and the thickest region of the aerofoil has moved forward compared to that of the original RAE-2822 aerofoil.

6.5.3 Standard Kriging Optimisation

The geometric filtration methodology considered here comprises of two kriging based optimisations linked by a reparameterisation of the design problem. It is, therefore, necessary to determine the performance of a standard kriging optimisation with respect to the current aerofoil design problem in order to provide a meaningful measure of the performance gains offered by the geometric filtration strategy.

Consider now a basic kriging optimisation consisting of a total budget of 300 objective function evaluations, a value much more indicative of what would be available in a typical design optimisation than the 10,000 evaluations used in the previous direct search. Of this total simulation budget, one third are used in the initial design of experiments (DOE), in accordance with the work of Sóbester et al. [2005], with the remaining budget reserved for updates to the surrogate model which are evaluated in batches of ten. All updates to the kriging models used in each of the following investigations are based on the model's prediction of the objective function. A genetic algorithm is used to minimise the objective function predicted by the model with the cluster centroids of the final population selected as the update points.

The kriging model has its hyperparameters tuned after every other set of updates. This strategy was demonstrated to offer a significant reduction in tuning cost. Each of the hyperparameters, (θ_l and p_l), are permitted to vary and regression is included in the kriging model. The 41 hyperparameters controlling the kriging model are optimised using a genetic algorithm followed by a dynamic hill climber, the "Alternate" tuning strategy of Section 3.3.



(a)

(b)

FIGURE 6.4: Optimisation histories for each of the 50 traditional kriging optimisations (a), pressure distribution and geometry for an example aerofoil resulting from a traditional kriging optimisation process with $\frac{C_D}{C_L} = 1.16 \times 10^{-2}$ or $\frac{C_L}{C_D} = 86.2$ (b)

This traditional kriging based optimisation strategy is applied to the aerofoil design problem a total of 50 times, with the random number seed used to generate the DOE changing each time. The optimisation histories for each of these 50 optimisations are presented in Figure 6.4(a), with a typical aerofoil resulting from one such optimisation presented in Figure 6.4(b).

The 50 optimisations result in an average $\frac{C_D}{C_L}$ of 1.16×10^{-2} with a standard deviation of 2.17×10^{-4} which equates to a $\frac{C_L}{C_D}$ of 86.4 and an improvement in the objective function of 21.7%. The improvement over the initial aerofoil design is therefore not as significant as that obtained using the direct GA. However, one must note that the direct search used over 33 times the number of function evaluations. The kriging strategy should therefore be commended for attaining 76.3% of the improvement obtained by the extensive optimisation with 3% of the simulation budget. Given enough time and updates one would expect the traditional kriging strategy to approach the results obtained by the GA, denoted by the dashed line in Figure 6.4(a).

Similar to the aerofoils resulting from the direct search, (Figure 6.3(b)), the kriging based optimisation results in a reduction in the upper surface pressure and an increase in lower surface pressure over the whole aerofoil. The shockwave appears to have been removed but there are a number of noticeable oscillations in the upper surface pressure distribution compared to that resulting from the direct genetic algorithm. This results in an overall decrease in drag to lift ratio, but one which is not quite as significant as that obtained using the genetic algorithm.

When one considers the average time taken for the traditional kriging optimisation, approximately 22.0 hours, one can once again observe the issue of hyperparameter tuning. Given the low cost of each simulation in this particular optimisation the cost of the complete simulation budget is therefore a small fraction of the total optimisation time (approximately 0.38%). The majority of the time is due to the hyperparameter tuning process, and more specifically the $O(n^3)$ decomposition of the correlation matrix necessary in each evaluation of the likelihood. The size of this correlation matrix and hence the cost of every evaluation of the likelihood increases as more update points are added to the surrogate model.

The number of such likelihood evaluations is directly linked to the dimensionality of the underlying problem. A large number of variables equates to a large number of kriging hyperparameters, of which there are typically $2d + 1$ if all θ and p values are considered and regression is included. Such large tuning optimisations may therefore require more likelihood evaluations to optimise the likelihood effectively. The cost of constructing the kriging correlation matrix is also directly related to the number of variables in the problem. A reduction in the number of variables can therefore improve hyperparameter tuning costs on a number of fronts.

6.5.4 Geometric Filtration

The optimisation of the RAE-2822 aerofoil, utilising the geometric filtration strategy described in Section 6.3, is considered next. In this initial investigation a total budget of 300 simulations, (identical to that used in the traditional kriging strategy), is employed, with half of this budget used in the initial optimisation and the remainder used in the secondary optimisation.

In the initial kriging based optimisation the available budget is split approximately evenly between the initial DOE and the updates to the kriging model. A DOE consisting of one third of the available budget for this initial optimisation was deemed inadequate due to the relatively high dimensionality of this optimisation. The secondary optimisation however uses one third of the available simulations in its DOE. A smaller ratio of DOE to updates is more reasonable for this optimisation as there is a smaller number of variables after the reparameterisation, hence fewer points are required to construct an adequate response surface.

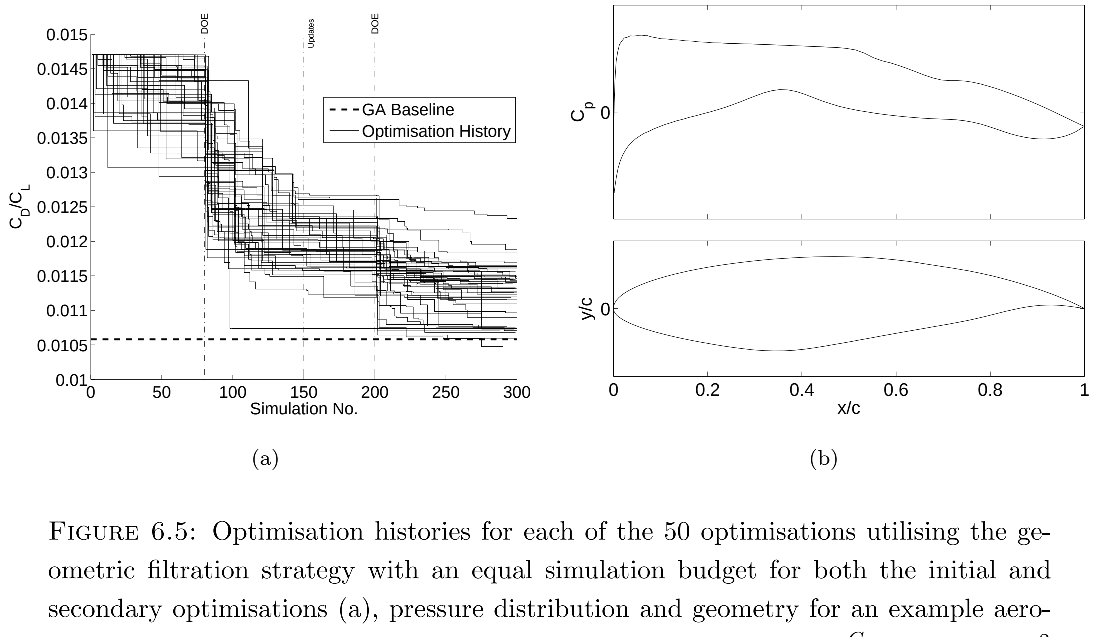
In this particular implementation of the geometric filtration strategy a total of 30 aerofoil geometries are selected from those generated during the initial kriging based optimisation. The aerofoils are selected using a KMEANS clustering algorithm, Anderberg [1975], in an attempt to maintain a measure of diversity between the selected aerofoils. Upon decomposing the snapshot ensemble into the POD bases, the first N bases which captured a cumulative total percentage variation of greater than 99.99% are selected as the POD modes for the secondary optimisation. Such a large cumulative percentage variation ensures that the original aerofoils contained within the snapshot ensemble can be recreated to a high degree of accuracy. This high accuracy means that the original

30 aerofoils exist within the design space resulting from the reparameterisation. As the objective function values are already known for these aerofoils, they allow, what is essentially, 30 additional "free" design points to be added to the design of experiments of the secondary optimisation.

In summary the optimisation commences with a DOE of 80 points, followed by 70 updates to the kriging model in batches of 10. From this initial, 20 variable (d) optimisation, 30 geometries are selected and reparameterised using POD resulting in typically 14 bases (N). The secondary optimisation then optimises the coefficients of these bases, beginning with a DOE of 50 points, (plus the 30 aerofoils from the snapshot ensemble), and followed by 100 updates to the model, again in batches of ten. As with the traditional kriging optimisation discussed previously, the hyperparameters are tuned after alternate updates.

Strategy	No. of Evaluations	Mean $\frac{C_D}{C_L}$	Std $\frac{C_D}{C_L}$	Mean $\frac{C_L}{C_D}$	Std $\frac{C_L}{C_D}$	Time (hrs)
Genetic Algorithm	10,000	1.06×10^{-2}	4.59×10^{-4}	94.7	4.60	2.78
Traditional Krig	300	1.16×10^{-2}	2.17×10^{-4}	86.4	1.62	22.0
Geometric Filtration	300	1.13×10^{-2}	3.76×10^{-4}	89.0	2.98	4.15

TABLE 6.1: Comparison of the three optimisation strategies with respect to the optimisation of the RAE-2822 for minimum drag to lift ratio.



(a)

(b)

FIGURE 6.5: Optimisation histories for each of the 50 optimisations utilising the geometric filtration strategy with an equal simulation budget for both the initial and secondary optimisations (a), pressure distribution and geometry for an example aerofoil resulting from the geometric filtration optimisation process with $\frac{C_D}{C_L} = 1.12 \times 10^{-2}$ or $\frac{C_L}{C_D} = 89.59$ (b)

Once again a total of 50 different optimisations are carried out, with the random number seed used in the Latin Hypercube changing each time. The histories for each of these optimisations are presented in Figure 6.5(a). The 50 optimisations result in an average drag to lift ratio of 1.13×10^{-2} with a standard deviation of 3.76×10^{-4} equating to an average lift to drag ratio of 89.0. This equates to a 23.6% improvement in the objective function over that of the baseline aerofoil. Applying the geometric filtration optimisation strategy to this design problem has therefore resulted in an average of 84.1% of the improvement obtained with the genetic algorithm. From Figure 6.5(a) it is also observed that a number of the final designs actually achieve a drag to lift ratio similar to that of the direct GA. The average results of these optimisations along with those of the traditional krig and the GA are presented together in Table 6.1.

Figure 6.5(b) shows a typical aerofoil resulting from an optimisation using this implementation of the geometric filtration strategy. Upon comparison to the aerofoil in Figure 6.4(b), resulting from the traditional kriging optimisation, one can observe the removal of the shockwave but with a reduction to the oscillations in upper surface pressure. The reduction in the upper surface pressure and increase in the lower surface pressure has been maintained. The optimisation has also resulted in an aerofoil with a more significant reduction to the leading edge curvature, something which is observed with the direct GA, but which is not as pronounced in Figure 6.4(b).

Upon comparison of the optimisation histories of Figures 6.4(a) and 6.5(a) a disadvantage of the current implementation of the geometric filtration strategy can be observed. The application of geometric filtration appears to have resulted in an increase in the variance of the final designs. The standard deviation of the drag to lift ratio for the traditional kriging strategy was 2.17×10^{-4} while geometric filtration resulted in an increased standard deviation of 3.76×10^{-4}. Therefore, although geometric filtration results in a better design on average, there is slightly less consistency in the quality of the final designs.

Consideration of the total optimisation time leads to the observation of another advantage of geometric filtration over the traditional kriging based optimisation. Instead of an average optimisation time of 22.0 hours, the utilisation of geometric filtration has resulted in an average optimisation time of 4.15 hours. As the krig tuning process is again responsible for the majority of the optimisation time, geometric filtration has resulted in a substantial reduction in tuning cost. This reduction in cost can be explained when one considers the actual tuning process in each optimisation, particularly the size of the correlation matrix, $\boldsymbol{R}$, used to calculate the likelihood, Equation 2.25.

In traditional kriging, the correlation matrix begins 100×100 in size and steadily grows as more update points are evaluated until, when the hyperparameters are to be tuned for the final time, the matrix is 280×280 in size. The above implementation of the geometric filtration strategy however begins with a correlation matrix of 80×80 which grows to 140×140 by the final tune of the initial optimisation. Following the reparameterisation process, 110 of these design points are filtered out and 50 new points added from the design of experiments of the secondary optimisation. The correlation matrix therefore grows steadily from 80×80 to 160×160 when the hyperparameters of the secondary krig are tuned for the final time. As the correlation matrix remains consistently smaller, the $O(n^3)$ cost of the factorisation in each likelihood evaluation is reduced hence reducing the overall cost of the hyperparameter tuning considerably.

The reduction in the total number of variables from 20 to approximately 14 in the secondary optimisation problem, also has an impact on the overall tuning cost. The reduction in dimensionality reduces the cost of constructing the correlation matrix in each likelihood evaluation. This reduces further the overall tuning cost but to a lesser extent than the reduction in correlation matrix size. Although the number of likelihood evaluations made in the optimisation of the hyperparameters throughout both of the above cases are consistent; a 5,000 evaluation GA is followed by a 5,000 evaluation DHC. The reduction in the number of variables also reduces the number of hyperparameters requiring optimisation. In theory one could therefore reduce the effort afforded to the hyperparameter tuning in the secondary optimisation of the geometric filtration process compared to the initial optimisation.

One should also note that the tuning cost associated with the traditional kriging process can be reduced through the utilisation of a restricted number of points in the correlation matrix. As described in Section 4.2, this restricts the size of the correlation matrix and prevents the tuning cost growing as update points are evaluated and added to the dataset. However, in this instance the number of hyperparameters remains constant throughout. In the case of geometric filtration the hyperparameter optimisation reduces in complexity due to the reduced number of variables. This both reduces slightly the cost of correlation matrix construction and the overall complexity of the hyperparameter optimisation problem.

6.6 Conclusions

An optimisation strategy involving two kriging based response surface optimisations and a POD based reparameterisation has been introduced and applied to the optimisation of an aerofoil for minimum drag to lift ratio. This strategy, termed geometric filtration, was demonstrated to outperform a traditional kriging based optimisation, producing better designs for a considerable reduction in overall optimisation cost.

The geometric filtration strategy applies an initial kriging response surface model optimisation to the original problem. From the results of this optimisation a number of good design points are selected to form a snapshot ensemble for the purposes of proper orthogonal decomposition (POD). The POD basis functions then act as a reparameterisation of the original problem, filtering out badly performing designs and reducing the number of variables. A secondary kriging response surface based optimisation is then carried out in which the modal coefficients of the POD bases are optimised.

The optimisation of a transonic aerofoil for minimum drag to lift ratio was used as a test case to compare the geometric filtration strategy to a traditional kriging based optimisation and an extensive direct optimisation using a genetic algorithm. The traditional kriging strategy achieved 76.3% of the improvement obtained by the genetic algorithm but with only 300 objective function evaluations. However, applying geometric filtration to the same problem, again using 300 objective function evaluations, produced designs achieving 84.1% of the improvement obtained with the genetic algorithm, a substantial improvement over the traditional kriging strategy.

Due to a reduction in the total number of variables and the size of the correlation matrix used in the calculation of the concentrated likelihood post reparameterisation, the application of the geometric filtration strategy realises a drastic reduction in the cost of surrogate model construction reducing from a total of 22.0 hours when using the traditional kriging strategy to a much more respectable 4.2 hours.

Chapter 7

Further Analysis of Geometric Filtration

7.1 Introduction

The previous chapter introduced the basic concept of geometric filtration and demonstrated the capabilities of the technique with respect to the optimisation of a two dimensional transonic aerofoil. Although the strategy improved upon the traditional kriging approach in terms of both an increase in mean lift to drag ratio and a reduction in overall tuning time, there was no attempt to refine the parameters controlling the strategy.

The following chapter therefore attempts to illuminate the impact of these controlling parameters on the performance of the geometric filtration strategy. First the influence of the proportion of the total simulation budget used prior to the reparameterisation is considered. This is followed by an investigation into the influence that the size of the snapshot ensemble has on performance. The influence that the number of POD bases selected has on both the recreation of the ensemble geometry and aerodynamic coefficients within the reparameterised design space is also considered along with the effect on the optimiser's performance.

The reduction in hyperparameter optimisation time associated with the geometric filtration strategy was mainly due to the reduction in the size of the kriging correlation matrix after the POD based reparameterisation. In an attempt to reduce the tuning time, this chapter also considers the effect of a restricted correlation matrix size on the performance of the traditional kriging strategy.

With the aim of maximising the performance of geometric filtration, this chapter concludes by applying the results of each of these investigations into a single optimisation, the final results of which, aim to demonstrate the full performance gains offered by the technique.

7.2 The Influence of the Initial & Secondary Optimisation Budgets

Of the many parameters influencing the geometric filtration strategy, the ratio of the total optimisation budget used in the initial and secondary optimisations could be considered one of the most significant. The designs evaluated during the course of the initial optimisation are used in the reparameterisation process, hence one would expect that the more effort expended in this initial optimisation the better the snapshot aerofoils and the more important the features captured by the POD bases.

As always, optimisation is somewhat of a battle between exploration and exploitation. In this case, increasing the effort applied to exploration in the initial optimisation reduces the budget available to the secondary optimisation to exploit the reduced set of variables resulting from the reparameterisation. To investigate this fully two further optimisations are considered using the geometric filtration strategy employing different simulation budgets in the initial optimisation.

The first of these uses one third of the total simulation budget in the initial optimisation, while the second uses two thirds. The other parameters remain identical to that of the initial geometric filtration optimisation. The DOE of both the initial optimisations use approximately half of the total budget for that optimisation, while the secondary optimisation uses one third. The alternate tuning strategy is again employed with 30 aerofoils selected to form the snapshot ensemble using a clustering algorithm. Once again the first N POD bases are selected to reproduce 99.99% of the cumulative percentage variation. The results for each of these optimisations along with those for the initial optimisation, which splits the total simulation budget evenly between the initial and secondary optimisations, are presented in Table 7.1. It should be noted that the optimisation using two thirds of the simulation budget in the initial optimisation commences with a DOE identical to that of the traditional krig defined in Section 6.5.3. This case therefore provides a direct performance comparison to the traditional kriging approach.

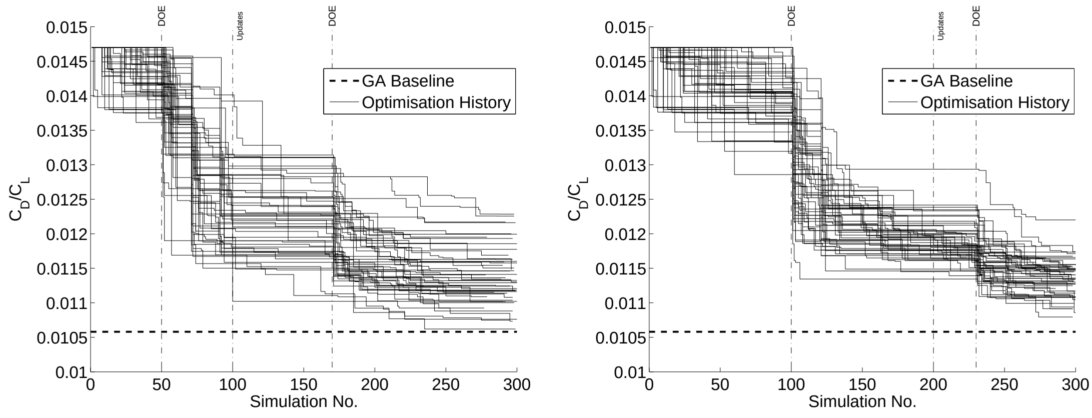
	Initial : Secondary Budget	Mean $\frac{C_D}{C_L}$	Std $\frac{C_D}{C_L}$	Mean $\frac{C_L}{C_D}$	Std $\frac{C_L}{C_D}$	Time (hr)
A	1:2	1.14×10^{-2}	4.14×10^{-4}	88.0	3.15	5.50
B	1:1	1.13×10^{-2}	3.76×10^{-4}	89.0	2.98	4.15
C	2:1	1.14×10^{-2}	2.69×10^{-4}	88.0	2.08	5.11
D	2:1*	1.13×10^{-2}	2.65×10^{-4}	88.6	2.06	4.95

* Secondary optimisation containing no additional DOE points, only updates to a surrogate model defined using the snapshot ensemble points.

Table 7.1: Comparison of geometric filtration optimisations using different ratios of simulation budgets in the initial and secondary optimisations.

The results presented in Table 7.1 indicate that although geometric filtration consistently outperforms the traditional krig, in terms of the average final objective function, the size of the initial optimisation budget has a noticeable impact on the consistency in the final designs and a slight effect on the quality of these designs. Reducing the initial budget to one third, (run A), slightly increases the average objective function of the final design but also introduces a larger variation in the final objective function. Increasing the ratio to one half, (run B), results in a reduction in the variation and slightly better designs on average. A further increase to two thirds, (run C), results in a slight increase in the average final objective function but a further reduction in the variation of these designs.

The results of this optimisation in particular serves to reinforce the performance improvement offered by geometric filtration over the traditional kriging strategy. Both run C and the traditional kriging optimisation begin from an identical DOE, but implementing the reparameterisation of the problem results in better designs on average.



(a)

(b)

Figure 7.1: Optimisation histories for each of the 50 optimisations utilising the geometric filtration strategy with one third (a) and two thirds (b) of the total simulation budget used in the initial optimisation

The optimisation histories of Figures 7.1(a), 7.1(b) and 6.5(a) help to explain these results more clearly. From these optimisation histories one can observe that as more effort is applied to the initial optimisation the quality of the designs obtained by the end of the initial optimisation improves. This results in a snapshot ensemble containing better designs hence the resulting filtration is able to retain better geometric features and produce better designs in the secondary optimisation. One can also observe that the variation in the final designs is related to the variation in the designs obtained from the initial optimisation. A large spread in the quality of the initial optimisation produces a corresponding spread in the final design (Figure 7.1(a)). Once again this is related to the quality of the designs from the initial optimisation, providing the snapshot ensemble with a series of poor designs where the reparameterisation will not contain information to significantly improve the design.

However, increasing the effort applied to the initial optimisation comes at the cost of a reduction in the ability to effectively optimise the reparameterised problem. Although better designs are found from which to construct the reparameterisation, fewer function evaluations remain to exploit the reparameterisation fully. This results in a slight worsening of the objective function of the final design observed when two thirds of the total simulation budget is used in the initial optimisation. This can be countered to some degree by reducing the size of the DOE evaluated at the beginning of the secondary optimisation. As the POD bases have been chosen to represent 99.99% of the cumulative percentage variation, these points effectively exist within the secondary optimisation design space. Using only these points as the DOE of the secondary optimisation frees up more simulations to be used as updates to the kriging model. As seen by the final result of Table 7.1, this produces designs with an average objective function close to that of the initial geometric filtration optimisation, but with a much smaller variation in the objective function of the final designs.

7.3 The Influence of Snapshot Ensemble Size

The initial geometric filtration optimisation used a total of 30 snapshot aerofoils in the ensemble. To judge the effect of ensemble size the secondary optimisation is repeated a further four times, varying ensemble size. For each of these additional optimisations the same initial optimisation using the original aerofoil parameterisation is used, thereby ensuring only the impact of ensemble size is observed. Snapshot ensembles of 10, 20, 30, 40 and 50 are all considered, the results of which are presented in Table 7.2.

Size of Snapshot Ensemble	Mean $\frac{C_D}{C_L}$	Std $\frac{C_D}{C_L}$	Mean $\frac{C_L}{C_D}$	Std $\frac{C_L}{C_D}$	Time (hr)	Mean No. of Bases
10	1.16×10^{-2}	3.36×10^{-4}	86.4	2.25	2.85	8.5
20	1.14×10^{-2}	3.46×10^{-4}	87.8	2.65	3.64	12.7
30	1.13×10^{-2}	3.76×10^{-4}	89.0	2.98	4.15	14.1
40	1.13×10^{-2}	3.48×10^{-4}	88.8	2.71	4.52	14.8
50	1.11×10^{-2}	3.48×10^{-4}	90.1	2.82	5.87	15.4

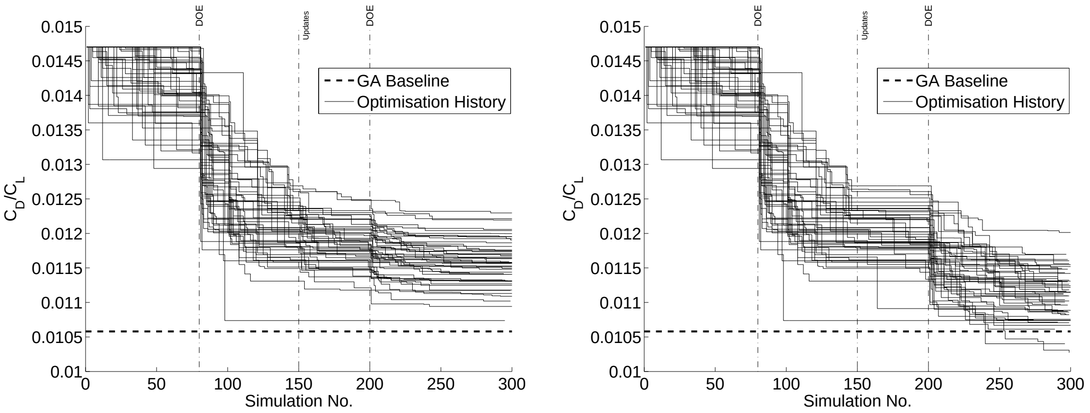
Table 7.2: Comparison of results using the geometric filtration optimisation methodology with an increasing snapshot ensemble size for a consistent 1:1 ratio of simulations in the initial and secondary optimisations.

Increasing the size of the snapshot ensemble, especially if a clustering algorithm is used to select the design points, increases the diversity of designs within the ensemble hence increasing the number of bases required to represent 99.99% of the cumulative percentage variation. The number of bases required increases with the number of snapshot aerofoils, however, after about 30 snapshots the increase in the number of bases is not as pronounced. This indicates that even though further snapshots are adding some information they are similar to those already included in the ensemble so fewer additional bases are required in order to fully represent them.

Increasing ensemble size results in an improvement in the average objective function of the final design. As suggested by the increase in the number of bases, adding more designs to the ensemble increases the variation of geometries which can be represented hence increasing the size of the design space. A larger more versatile design space results in the capacity to create better geometries and thus a greater reduction in objective function.

This can be visualised by considering the optimisation histories of Figures 7.2(a) and 7.2(b). In Figure 7.2(a) one can observe the progress of the secondary optimisation based on the POD bases defined using an ensemble of 10 snapshot aerofoils. Using the reparameterisation resulting from these snapshots in the secondary optimisation results in less improvement in objective function compared to that of Figure 7.2(b). Here the reparameterisation is based on 50 snapshots resulting in a more flexible reparameterisation and therefore produces a greater improvement in objective function on average.

One should note that as the number of bases increases so too does the number of variables in the secondary optimisation, thereby increasing the difficulty of this optimisation given a limited simulation budget. However, as discussed previously, if such a large cumulative percentage variation is used to select the POD bases the ensemble design points exist within the design space of the secondary optimisation. These points can therefore be added to the initial DOE of the secondary optimisation. A larger ensemble size therefore means a larger DOE in the secondary optimisation and the negative impact of the increase in dimensionality is somewhat reduced. The larger number of points in the secondary optimisation however does result in an increase in krig tuning time due to the larger correlation matrix, as shown by the results of Table 7.2. An increase in the number of variables in the secondary optimisation also results in an increase in the number of hyperparameters and increases the complexity of the tuning process. The cost of constructing the correlation matrix in each likelihood evaluation also increases.



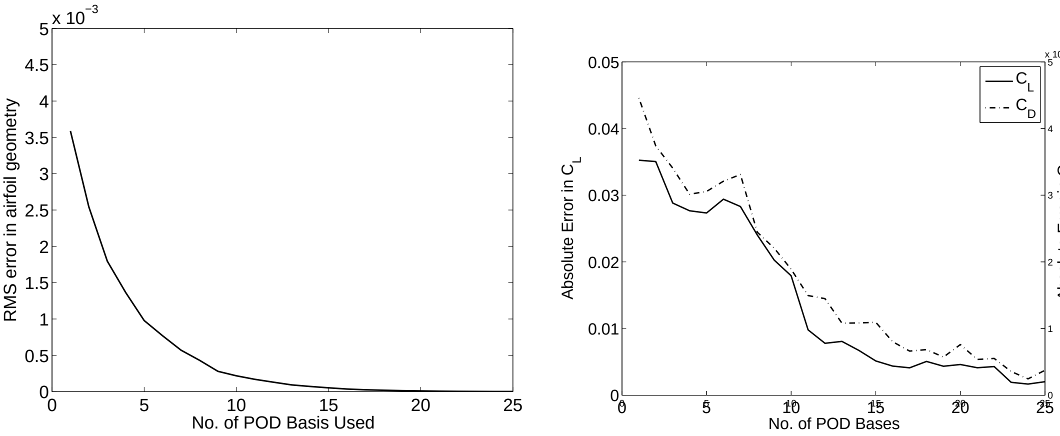
(a)

(b)

Figure 7.2: Optimisation histories for the geometric filtration strategy with the reparameterisation based on 10 (a) and 50 (b) aerofoil snapshots

7.4 The Influence of POD Bases Selection

In the above analysis it has been assumed that a cumulative percentage variation of 99.99% is enough to reproduce the snapshot aerofoils to such a degree that they exist within the design space of the secondary optimisation. We now consider the relaxation of this limit on the cumulative percentage variation and the subsequent ability of the remaining POD basis functions to recreate the geometry of the original snapshot aerofoils and, more importantly, the effect of any error in the geometric reconstruction on the correct prediction of lift and drag.



(a)

(b)

Figure 7.3: Demonstration of the accurate recreation of the original snapshot aerofoils with increasing number of POD bases (a) and the effect this has on the error in lift and drag coefficients (b)

We begin by analysing the recreation of 50 snapshot aerofoils from a typical ensemble used in the previous analysis. This large ensemble was selected so as to provide a more meaningful average of the effect of reducing the number of POD bases. Figure 7.3(a) shows the change in the average error in the recreation of the geometry of the original aerofoils as the number of bases used in the recreation increases. Obviously with only a few bases the original aerofoils cannot be accurately recreated but increasing the number of bases produces a rapid reduction in the error. For this particular ensemble the first 16 POD bases result in a cumulative percentage variation of greater than 99.99%. This equates to a small average root mean square (RMS) reconstruction error of 3.64×10^{-5}.

Let us now consider the effect of this error in geometry recreation on the aerodynamic properties used in the current aerofoil optimisation, specifically lift and drag. Figure 7.3(b) shows the average absolute error in both C_L and C_D as the number of bases used to recreate the original ensemble of aerofoils is increased. As expected, increasing the number of bases, and hence the accuracy of the geometric recreation, results in a reduction in the error of both lift and drag. This reduction is however not as rapid as that of the reduction in geometric error.

It is generally recognised that even small changes to a geometry can significantly alter these aerodynamic coefficients. This is especially true in the transonic flow regime considered here, where slight changes to geometry can affect shock strength and position and hence lift and drag. A relatively small error in the recreation of the geometry is therefore required to accurately reproduce the aerodynamic response.

The results presented in Figure 7.3(b) show that when 16 POD bases are used, the error in C_L and C_D are, respectively, 4.40×10^{-3} and 8.02×10^{-5} on average. An average C_D error of just under one drag count could be considered acceptable for the purposes of including the snapshot objective function values within the DOE of the secondary optimisation. However, including these objective function values in a secondary design which has utilised a smaller number of POD bases would be unwise unless provision is made for the possible error in these results. Without such a provision the response surface would not accurately represent the true response of the objective function to changes in the POD basis coefficients, therefore introducing discontinuities in the response surface which may hamper the secondary optimisation.

To further investigate the impact of POD bases a total of eleven further optimisations are carried out, results of which are presented in Table 7.3. Each of these optimisations uses an identical initial optimisation and an identical snapshot ensemble of 30 aerofoils. The only difference here is that, after the POD bases are calculated, only the specified number of bases are used in the aerofoil parameterisation of the secondary optimisation. With the above observations in mind the objective function values of the snapshot aerofoils are not included in any of the DOEs of the secondary optimisations. In all cases a budget of 150 simulations is used in the secondary optimisation with one third of these reserved for the DOE.

No. of POD Bases	Mean $\frac{C_D}{C_L}$	Std $\frac{C_D}{C_L}$	Mean $\frac{C_L}{C_D}$	Std $\frac{C_L}{C_D}$	Time (hr)
1	1.20×10^{-2}	3.79×10^{-4}	83.5	2.70	2.16
2	1.20×10^{-2}	3.77×10^{-4}	83.7	2.69	2.25
4	1.18×10^{-2}	4.42×10^{-4}	84.7	3.26	2.41
6	1.16×10^{-2}	4.06×10^{-4}	86.4	3.05	2.55
8	1.15×10^{-2}	3.74×10^{-4}	87.4	2.88	2.68
9	1.14×10^{-2}	3.80×10^{-4}	87.9	2.95	2.78
10	1.13×10^{-2}	3.85×10^{-4}	88.7	3.02	2.89
11	1.13×10^{-2}	3.66×10^{-4}	88.5	2.89	2.98
12	1.13×10^{-2}	3.38×10^{-4}	88.6	2.64	3.06
13	1.13×10^{-2}	3.62×10^{-4}	88.5	2.86	3.13
14	1.13×10^{-2}	3.49×10^{-4}	88.5	2.72	3.25

Table 7.3: Performance results for the geometric filtration strategy when the secondary optimisation employs a reduced number of POD bases in the parameterisation.

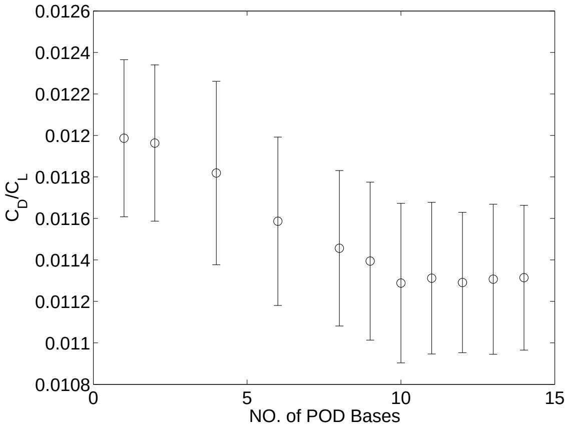


FIGURE 7.4: Demonstration of the change in average objective function of the best design found as the number of POD bases used in the reparameterisation is increased, error bars denote ± one standard deviation

The results of Table 7.3, which are also presented graphically in Figure 7.4, demonstrate the importance of including a significant number of the POD bases in the reparameterisation. As expected, the more bases included the greater the flexibility of the secondary parameterisation and the better the designs which can be produced, up to a point. Figure 7.4 demonstrates this very clearly, with a downward trend in the average objective function of the final design as more POD bases are used.

However, when more than ten bases are used in the reparameterisation there is no additional improvement in the objective function, in fact the designs are, on average, slightly worse. As the number of bases increases so to does the number of variables in the secondary optimisation. Given a constant simulation budget there will come a point when the number of simulations is insufficient to result in a continual improvement in objective function. Continually increasing the number of variables may therefore result in a reduction in the quality of the final designs obtained, a phenomenon which was observed previously in Chapter 3.

Including the objective functions from the snapshot aerofoils in the DOE of the secondary optimisation increases the quality of the initial kriging response surface and improves the average objective function of the final design. This accounts for the improvement in average objective function obtained with the initial geometric filtration results, Section 6.5.4, over those presented in Table 7.3. Even though the average number of bases is larger, including the additional points helps the secondary optimisation achieve a better design.

The reduction in tuning time over the previous strategies is a direct result of not including the objective functions of the snapshot aerofoils within the secondary DOE. Without these extra points the size of the correlation matrix is reduced and therefore so is the total tuning cost. Once again the number of hyperparameters is reduced in the secondary optimisation when fewer bases are utilised. Although not implemented here, the effort applied to the solution of the hyperparameter tuning problem could be reduced accordingly.

In conclusion, if a reduced number of POD bases is used in the secondary optimisation, careful consideration is required of the available simulation budget, the degree of geometric flexibility in the reparameterisation and the accuracy to which the ensemble of aerofoils are recreated.

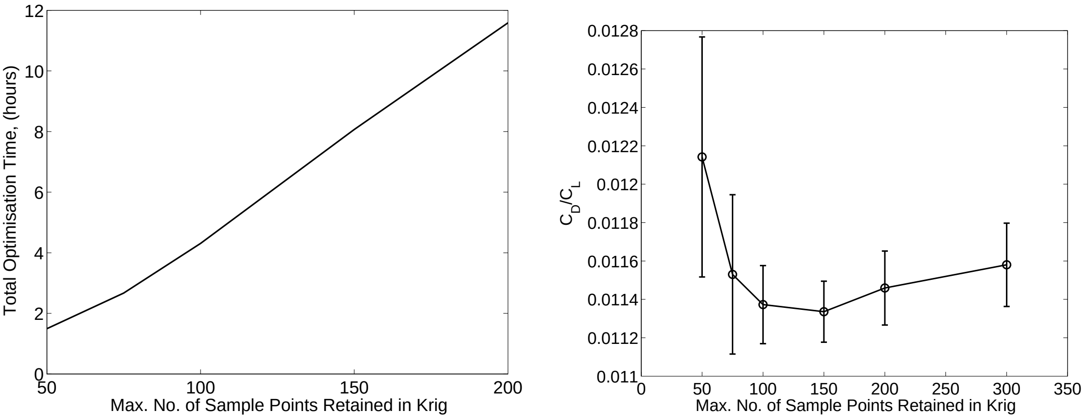
7.5 Performance of a Restricted Kriging Dataset

Throughout the previous analysis of the geometric filtration strategy the impact of the reduction in the number of sample points on the total optimisation time has been evident. The reduction in the number of sample points reduces the cost of every evaluation of the concentrated likelihood reducing the total optimisation cost from 22.0 hours, using the traditional kriging strategy, to an average 4.2 hours for the initial geometric filtration optimisation. The total optimisation time is reduced even further, to 2.9 hours, when the objective functions for the snapshot ensemble are not included in the DOE of the secondary optimisation and ten POD bases are used in the reparameterisation.

Reconsidering the traditional kriging strategy it is obvious that maintaining a constant, restricted number of sample points reduces the overall tuning time. However, what effect does such a restriction have on this strategy's overall performance, and how does the performance compare to that of geometric filtration at an equivalent total optimisation cost? To this end a total of five additional optimisations are performed using the traditional kriging optimisation strategy but restricting the number of sample points within the design space to the best 50, 75, 100, 150 and 200. For example, restricting the design space to 150 points results in the optimisation progressing normally after the DOE until the limit of 150 points is reached. Once this limit is reached the objective function values of future updates are compared to those of the points currently in the design space and from these points the best 150 are retained. The results of this investigation are presented in Table 7.4 and graphically in Figures 7.5(a) and 7.5(b).

Kriging Sample Size	Mean $\frac{C_D}{C_L}$	Std $\frac{C_D}{C_L}$	Mean $\frac{C_L}{C_D}$	Std $\frac{C_L}{C_D}$	Time (hr)
50	1.21×10^{-2}	6.25×10^{-4}	82.6	4.22	1.49
75	1.15×10^{-2}	4.15×10^{-4}	86.8	3.01	2.67
100	1.14×10^{-2}	2.03×10^{-4}	88.0	1.56	4.31
150	1.13×10^{-2}	1.59×10^{-4}	88.2	1.24	8.07
200	1.15×10^{-2}	1.93×10^{-4}	87.3	1.45	11.59
300	1.16×10^{-2}	2.17×10^{-4}	86.4	1.62	21.97

Table 7.4: Performance results for the standard kriging optimisation approach with varying sample size.



(a)

(b)

FIGURE 7.5: Total wall time for a kriging optimisation as the number of sample points used is reduced (a), average and standard deviation of the objective function of the final design as the number of sample points used is reduced (b)

Figure 7.5(a) demonstrates the large reduction in the total optimisation time achievable by restricting the maximum number of points within the response surface model. Limiting the model to include only the 200 best points reduces the total time by approximately 47% while limiting the model to 100 points reduces the total time by 80%. From these results one can observe that restricting the number of points to a maximum of 100, results in a total optimisation time much more competitive with that of the initial geometric filtration results. However, geometric filtration still slightly outperforms the restricted krig in terms of the average final objective function. The restricted krig results in an average final objective function of 1.14×10^{-2}, while geometric filtration results in an average final objective function of 1.13×10^{-2}.

It should be noted that restricting the number of sample points has resulted in an improvement in the overall performance of the traditional kriging strategy, with the average final objective function decreasing from 1.16×10^{-2} with an unrestricted krig, to 1.13×10^{-2} when the number of sample points is restricted to 150. This is however still not as great an improvement in objective function as obtained with the initial geometric filtration strategy.

In Section 7.4 a further reduction in the geometric filtration tuning time has been demonstrated through a reduction in the number of POD bases used in the reparameterisation. This reduction in bases prevents the original snapshots being reconstructed accurately hence they are not included in the DOE of the secondary optimisation. This reduces the total optimisation time to approximately 2.87 hours, 14% of the total time of the traditional krig without any restriction on the number of sample points. In order for the traditional krig to achieve a total time approaching this, the number of sample points would have to be restricted to around 75. As the results of Table 7.4 show, this produces an average final objective function greater than geometric filtration achieves. The consistency between the final designs is also better when using the geometric filtration strategy, producing a standard deviation in the objective function of 3.85×10^{-4} compared to the kriging strategy's standard deviation of 4.15×10^{-4} when restricted to 75 sample points.

It must be noted that the above investigations employ a subset of the best points in the construction of the correlation matrix during both the likelihood optimisation and the krig prediction. This focuses the optimisation to regions around the best points with the predictor reducing to the mean in those regions where points have been excluded due to their poor quality. This is similar in some respects to geometric filtration where the optimisation is focused via the reparameterisation procedure. Considering only the best points in both the likelihood optimisation and prediction poses problems if expected improvement, Section 2.3.4, were adopted to generate update points.

Although not considered here, an alternative is to consider a subset of the best points during only the hyperparameter tuning process, with the complete dataset being employed in the construction of the kriging predictor. This produces a correct set of hyperparameters in the region of the best designs and with broadly the correct features over the whole design space. The resulting predictor could therefore be considered correct in regions of interest but with more of the global features of the design space captured than the strategy employed above. However, open questions remain as to which strategy performs best and precisely how points should be selected to form the reduced dataset, especially in the case of expected improvement. Instead consider the above results as a demonstration of the potential of a restricted dataset within the framework of a traditional kriging optimisation.

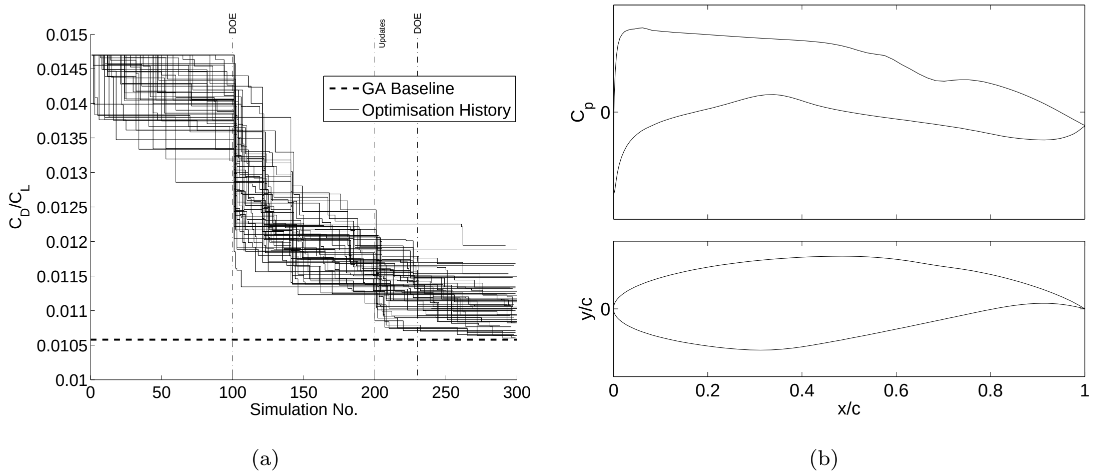
7.6 A Restricted Dataset Within the Geometric Filtration Framework

As the geometric filtration optimisation strategy is based upon a series of surrogate model optimisations it stands to reason that any technique which can be applied to a normal surrogate model can also be applied to geometric filtration. The initial or secondary optimisations could make use of gradient or Hessian enhanced surrogate models, different updating formulations and could even incorporate variable fidelity simulations through co-kriging.

We have observed that by restricting the number of sample points within a design space through the retention of only the best points, the total optimisation time of a traditional krig can be significantly reduced. However, not only was the time reduced but the quality of the final design was observed to improve to some degree. We now consider the application of this restriction in the number of sample points to both the initial and secondary optimisations of the geometric filtration strategy, along with a number of other enhancements to the strategy based on the results presented previously.

Increasing the initial optimisation simulation budget was found to increase the consistency in the designs obtained but at the cost of a slight reduction in the average objective function of the final design. This reduction was countered by removing the additional DOE points from the secondary optimisation and using only the points from the snapshot ensemble as the basis for the kriging response surface. Increasing the size of the snapshot ensemble was observed to increase the variation in the resulting geometries and hence increase the flexibility of the parameterisation used in the secondary optimisation. However, this came at the cost of a slight increase in the number of variables in the secondary optimisation and a slight increase in overall optimisation time.

We now consider an implementation of the geometric filtration strategy which takes all of the observations made throughout the current chapter into account. Once again a total of 300 simulations are used in the optimisation. To attempt to increase the consistency between the final designs, two thirds of the total simulation budget is used in the initial optimisation with half of these reserved for the DOE. The snapshot ensemble comprises of a total of 50 aerofoils in an attempt to increase the flexibility of the parameterisation used in the secondary optimisation. To improve the performance of the secondary optimisation the additional DOE is once again neglected and the remaining 100 simulations are used as updates to a kriging model initially defined using the objective functions of the snapshot points. Once again N bases are selected to represent at least 99.99% of the cumulative percentage variation. A restricted dataset is employed in both the initial and secondary optimisation strategies. In a manner similar to that of Section 7.5 the best 100 points are kept throughout the course of the initial optimisation, while the best 50 points are kept throughout the course of the secondary optimisation.



(a)

(b)

FIGURE 7.6: Optimisation histories for each of the 50 geometric filtration optimisations using a restricted number of sample points (a), pressure distribution and geometry for an example aerofoil with $\frac{C_D}{C_L} = 1.11 \times 10^{-2}$ or $\frac{C_L}{C_D} = 90.3$ (b)

The above implementation of the geometric filtration strategy obtained an average $\frac{C_D}{C_L}$ of 1.11×10^{-2} which equates to a $\frac{C_L}{C_D}$ of 90.3. By adjusting the setup of the strategy in this way, 88.6% of the improvement in objective function obtained by the direct search has now been obtained with approximately 3% of the simulation budget. Utilising a larger proportion of the total simulation budget in the initial optimisation results in an increase in the consistency between the final designs with a standard deviation of the final objective function reducing to 3.20×10^{-4} from 3.76×10^{-4} for the initial implementation of the strategy. This is still not quite as good as that obtained during the original investigation into the impact of initial optimisation size and is probably due to the negative impact of the increase in the number of variables in the secondary optimisation.

The optimisation histories of Figure 7.6(a) help to demonstrate graphically the clear improvement over both the traditional krig and the various previous implementations of the geometric filtration strategy presented in Section 6.5.4. Figure 7.6(b) presents an example aerofoil resulting from a typical optimisation. As with the aerofoil resulting from the initial geometric filtration strategy the upper surface shockwave has been removed but there is a further reduction in upper surface pressure and an increase in lower surface pressure. The combination of these features produce a better performing aerofoil with a smaller drag to lift ratio than was obtained by the traditional kriging strategy, Section 6.5.3.

Not only has this particular implementation of the geometric filtration strategy resulted in consistently good designs but a restriction in the number of sample points in both surrogate models has resulted in a substantial reduction in overall optimisation time with each optimisation taking an average of 2.51 hours. The total optimisation time has in fact been reduced to such a degree by the implementation of geometric filtration that it is now faster than the extensive optimisation using the GA and DHC, which took approximately 2.78 hours to complete.

Applying the lessons learned from the analysis of the performance of the geometric filtration strategy has therefore resulted in an optimisation strategy which not only outperforms a traditional kriging strategy but does so for a considerable reduction in overall optimisation time even when applied in conjunction with a solver as fast as VGK.

7.7 Conclusions

The major parameters controlling the performance of the geometric filtration strategy, introduced in Chapter 6, have been investigated. The impact of the size of the initial optimisation budget, the size of the POD snapshot ensemble and the number of POD basis functions selected, were all considered.

The results of these investigations indicate that increasing the size of the initial optimisation budget effects consistency between final designs. Increasing the number of snapshot aerofoils produces better final designs by increasing the flexibility of the parameterisation of the secondary optimisation. Selecting a reduced number of POD bases reduces the cost and complexity of the secondary optimisation but at the expense of a reduction in flexibility of the parameterisation. It was demonstrated that the number of POD bases used in the secondary optimisation should be capable of capturing a large percentage of the cumulative variation if the objective function values from the snapshot aerofoils are to be included within the DOE of the secondary optimisation. Without a sufficient number of bases the inclusion of these points may introduce discontinuities into the response surface hampering the optimisation.

A restriction to the number of sample points used in constructing the response surface was introduced and applied to the traditional kriging optimisation process to make it more competitive in terms of the total optimisation time. This was found to not only reduce the total cost of each optimisation but in some cases improve the average objective function of the final design. Even with this improvement the traditional strategy still produced designs which were on average worse than those obtained using geometric filtration.

The flexibility of geometric filtration, with regard to the implementation of performance enhancing features used in traditional surrogate models was demonstrated with the application of a restriction in the number of sample points. The previous observations made regarding the effect of an increased initial optimisation budget and an increase in the size of the snapshot ensemble were also applied. The resulting optimisations achieved a further improvement in the final average objective function, achieving 88.6% of the improvement obtained by a direct genetic algorithm for 3% of the total simulation budget.

In conclusion, Chapter 6 introduced the basic concept of geometric filtration and achieved both a 3% improvement in the quality of the final design and a 81% reduction in tuning cost over the traditional kriging strategy. Through investigating the performance parameters of the strategy and implementing the subsequent conclusions the strategy achieved a 88.6% reduction in tuning time and a 4.5% improvement in objective function.

Chapter 8

Transonic Wing Optimisation

8.1 Introduction

The previous five chapters have considered a range of techniques designed to both improve the performance and reduce the tuning cost of kriging.

Chapter 3 demonstrated the importance of a continual reassessment of the hyperparameters of a kriging model throughout the course of an optimisation. The results of this chapter also showed that the overall cost of tuning could be reduced by simply tuning the hyperparamaters after every other set of updates to the model with little loss in performance.

Chapters 4 and 5 first defined an adjoint of the concentrated likelihood function and then utilised this within a hybridised particle swarm optimisation. This resulted in an efficient tuning strategy similar in form to dynamic hill climbing, with simultaneous exploration of unexplored regions of the likelihood space and exploitation via SQP using the cheap gradient information. Chapter 5 further highlighted the need for a global exploration of kriging hyperparameters as updates are added to the kriging model.

Chapters 6 and 7 introduced the geometric filtration optimisation strategy. This strategy was improved, through considerable analysis of the strategy's control parameters, to a point where it outperformed the traditional kriging approach in terms of both the quality of the final design and the total tuning time.

With the exception of geometric filtration, where the alternate tuning strategy was employed throughout, each of the optimisation techniques discussed above has been employed, and hence investigated, in isolation. It is therefore the aim of this chapter to pull these various threads together into one encompassing process. The optimisation of a transonic wing for minimum drag at a fixed lift is considered as a test case to compare the performance of this strategy to traditional kriging with hyperparameters tuned after every update via the baseline "Heavy" tuning strategy introduced in Chapter 3.

8.2 Transonic Wing Design Problem

The baseline geometry for the transonic wing design case, presented in Figure 8.1, is based upon a similar planform and thickness distribution to the DLR-F4 wind tunnel model, Redeker and Müller [1985]. The wing has a semi-span of 0.52m with a crank 0.16m from the centreline. Leading edge sweep is identical to that of the DLR-F4 at 27.1° while washout varies linearly from root to tip culminating in the tip twisted downwards by 5° relative to the wing setting angle. The wing is defined by the 12% thick RAE-2822 transonic aerofoil at the tip and crank while a modified 14% thick RAE-2822 aerofoil is used at the root with the sections varying linearly between root and crank. The resulting thickness distribution is similar to that of a typical civil airliner, with a thicker root to accommodate the internal wing structure and fuel tanks.

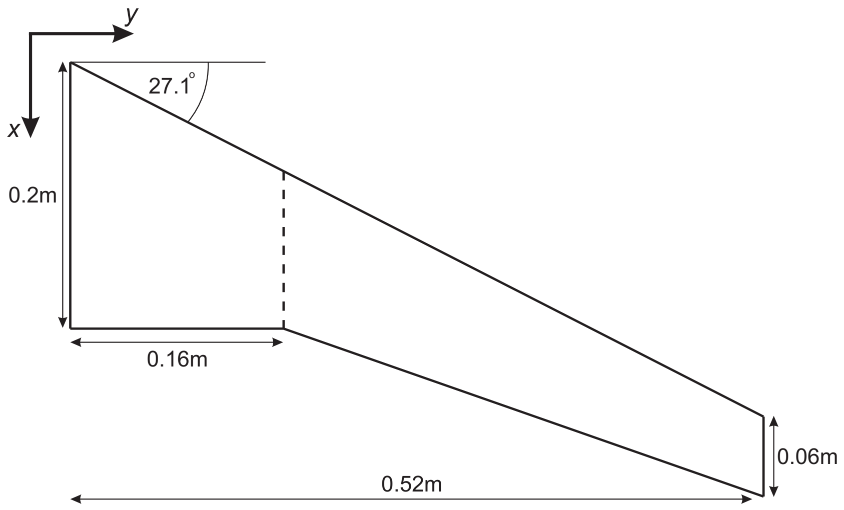


Figure 8.1: Baseline wing planform.

Aspect Ratio	9.4	Wing Area (m^2)	0.115
Semi-span (m)	0.52	Wing Root Chord (m)	0.2
$\Lambda_{1/4}$	25°	Taper Ratio	0.3
Washout	5°	Dihedral	0°
Aerofoil Section	RAE-2822	Mach No.	0.8
Reynolds No.	6×10^6	Target C_L	0.45
Baseline C_D	0.0204		

Table 8.1: Definition of the baseline transonic wing geometry and flow conditions.

The wing is parametrised in a similar manner to the inverse design problem presented in Section 3.4. Two Hicks-Henne bump functions are applied to both the upper and lower surfaces of the defining aerofoil sections. The same bump functions are applied to the tip, crank and root with the amplitude, position and sharpness of each bump permitted to vary. Essentially, this results in a wing constructed from a linear spanwise lofting of the RAE-2822 aerofoil which has been modified by four bump functions. A further two variables define the twist of the wing at the crank and tip resulting in a total of

14 variables. Reuther et al. [1999] use a similar parameterisation based on Hicks-Henne functions in the optimisation of a business jet wing.

The wing is optimised for a minimum drag at a fixed lift coefficient of 0.45 at Mach 0.8 and a Reynolds number of 6×10^6 with respect to the root chord, this being the typical cruise conditions of a modern wide-body airliner. The analysis of the wing is carried out using a combination of the full potential solver FP and a viscous drag correction employing VGK. Although this wing analysis procedure, presented in Appendix E, is not quite as accurate at transonic conditions as it is at low speed it does provide a relatively cheap estimation of total drag which shall suffice for the purposes of this investigation.

The wing is analysed up to four times for every objective function evaluation in order to calculate the setting angle producing the required lift. Two simulations, at different angles of attack, are used to calculate the wings lift curve slope. Based on this lift curve slope, and the assumption of linear change in lift with angle of attack, a third simulation is carried out at the angle of attack predicted to give the required lift. If necessary a fourth simulation, based on a linear interpolation of the lift coefficients and corresponding angles of attack, is carried out if the error in the lift coefficient resulting from the third simulation is more than ± 0.0025.

With the required setting angle found, the viscous drag prediction is carried out and added to the wave and vortex drag predictions from FP resulting in a prediction of the total drag. This entire procedure takes approximately 45 minutes on a single processor. The accuracy of the drag prediction at the desired flow conditions is therefore sacrificed for a cheaper overall simulation. A RANS solver, for example, would require multiple processor cores and take upwards of eight hours to evaluate a wing at a single angle of attack. In the 24-32 hours taken to evaluate a single objective function using such a solver, an entire optimisation can be completed using FP. This difference in evaluation cost is essential to enable averaging of the following optimisation strategies. In essence accuracy is substituted for repeatability since the aim is to examine the design process rather than the design itself.

8.3 Traditional Kriging

In this test the traditional kriging optimisation employs a total budget of 210 objective function evaluations. Of this total budget 70 are employed in the initial design of experiments with the remaining 140 evaluations providing 14 batches of 10 updates to the surrogate model. The kriging hyperparameters are tuned after every batch of updates to the model via the “Heavy” strategy which involves an initial 5,000 likelihood evaluations using a GA followed by a further 5,000 using a dynamic hill climber. Once again, for the purposes of averaging, a total of 50 complete optimisations are carried out, each commencing from a different initial design of experiments.

The 50 optimisations obtain an average final drag coefficient of 0.0170 equating to a 16.8% improvement over the drag of the baseline wing. The search histories are presented in Figure 8.2 for each of the 50 optimisations. The outlying design of Figure 8.2 is the result of an apparent under prediction of the viscous drag relative to that obtained by the other optimised designs and has therefore been removed from the calculations of mean and standard deviation. Over the course of an optimisation an average of 4.8 hours was spent during each run tuning the hyperparameters.

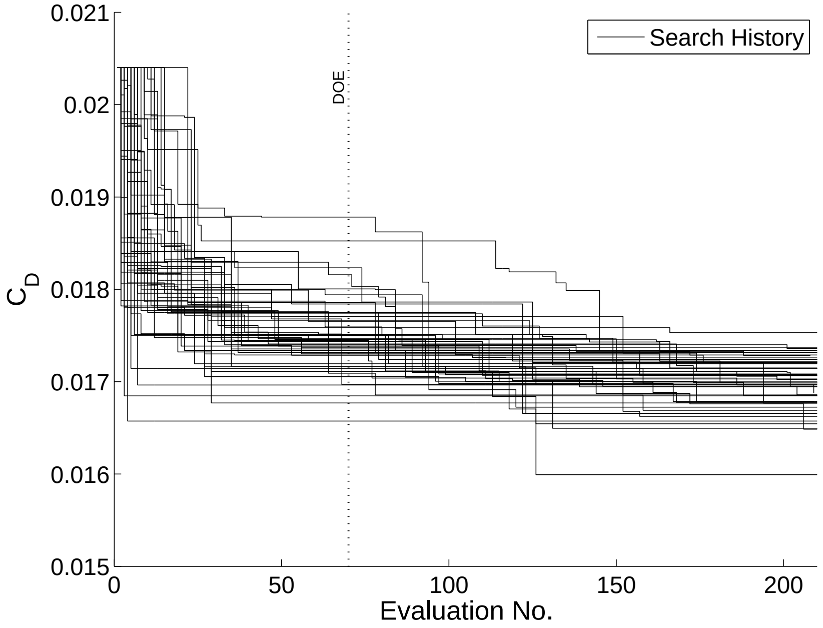
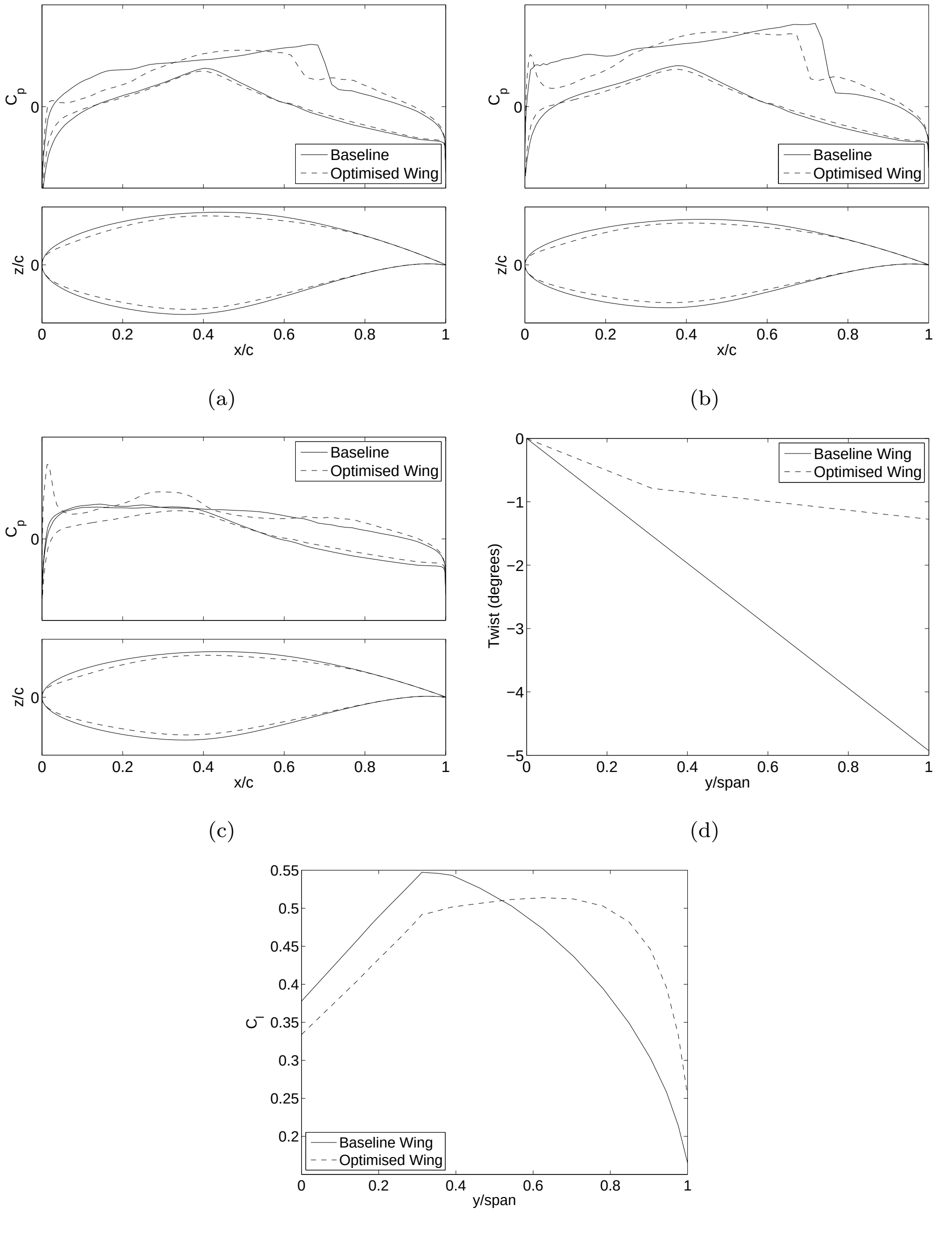


Figure 8.2: Search histories of the traditional kriging based optimisation employing the Heavy tuning strategy after every set of updates.

Figure 8.3 demonstrates the changes in both the sectional geometry and spanwise twist distribution of the best design resulting from an example optimisation. As can be observed the Hicks-Henne functions have reduced the thickness of the wing sections from the leading edge to around the three quarter chord point. Aft of this point the aerofoil sections return to the baseline shape. Such shape changes are typical of design optimisations carried out at a single operating point since robustness to variations in angle of attack, for example, are not required.

The root and crank pressure distributions of Figures 8.3(a) and 8.3(b) demonstrate a reduction in shock strength along the wing and the movement of the shockwave forward by approximately 5% of the chord. The reduction in shock strength results in a reduction in the thickening of the boundary layer aft of the shockwave which leads to a reduction in skin friction drag. The viscous drag coefficient therefore drops from 0.0107 for the baseline wing to 0.0100 for the wing presented in Figure 8.3.

The final design results in a reduction to the degree of washout at both the crank and tip. This coupled with the modifications to the wing sections results in a movement of the spanwise generation of lift away from the wing crank and towards the tip. This is reflected in the difference between the pressure distributions at the wing root, Figure 8.3(c), where an increase in upper surface pressure relative to the baseline wing can be observed in conjunction with a relatively consistent lower surface pressure.



(a)

(b)

(c)

(d)

(e)

FIGURE 8.3: Comparison of the root (a), crank (b) and tip (c) aerofoils, sectional pressure, twist distribution (d) and spanwise loading (e) of a typical design resulting from the traditional kriging based optimisation ($C_D = 0.0170$) to the baseline wing.

Essentially the optimisation is attempting to balance a reduction in viscous drag coefficient and shock strength with the generation of the required lift coefficient. The modifications to the presented wing reduce the viscous and wave drag inboard of the wing crank but also decrease the lift generated by these sections. To balance this the optimisation reduces the degree of washout which tends to increase the loading on wingtip. A similar phenomenon can be observed in the transonic wing optimisations of Keane and Petruzzelli [2000].

While not considered in this optimisation, the increase in tip loading coupled with the reduction in thickness could lead to structural problems. Loading the wing tip more increases the bending moment at the root requiring a stronger and therefore heavier wing. This could be countered through an estimation of the weight of the wing structure and then adjusting the required lift accordingly. Heavier wings therefore require more lift and are penalised in the optimisation by an increase in vortex drag, (see, for example, Chapter 12 of Keane and Nair [2005]). A constraint on the wing thickness could also be employed though this would require the construction of an additional surrogate model and as a result would increase total tuning costs.

8.4 Geometric Filtration with Adjoint Enhanced Tuning

Employing both the geometric filtration strategy defined in Chapter 6 and the hybrid hyperparameter tuning strategy defined in Chapter 5, the final optimisation strategy is a little more complicated than that of the traditional kriging strategy employed above.

The optimisation again employs a total budget of 210 objective function evaluations, commencing with an initial design of experiments of 70 points. The same 50 design of experiments employed by the traditional kriging optimisation are re-used. This not only provides a more meaningful comparison between the strategies but reduces the total number of actual function evaluations required over the course of the study.

Once again a total of 140 objective function evaluations are used to update the surrogate model. Of these, 70 are employed in the initial optimisation with the remaining 70 used in the secondary optimisation. Updates are again evaluated in batches of 10 but the hyperparameters are now only tuned after every other batch of updates. Instead of the "Heavy" tuning strategy employed above, the hybrid hyperparameter optimisation strategy of Chapter 5 is employed. This utilises the adjoint of the likelihood and the equivalent of approximately 2,000 likelihood evaluations.

The combination of the alternate tuning strategy of Chapter 3 and the geometric filtration strategy results in a total of eight tunes of the hyperparameters, four in the initial optimisation and a further four in the secondary optimisation. The traditional kriging strategy above, however, employs a total of 14 hyperparameter tunes over the course of the optimisation.

Upon the completion of the initial optimisation, 35 designs are selected via KMEANS clustering, (Anderberg [1975]), for use in the reparameterisation procedure. The snapshot vectors include the perturbations to the upper and lower surface from the Hicks-Henne functions along with the tip and crank twist angles. As per Chapter 6 a high cumulative percentage of variation is chosen so that the original designs exist within the new design space. The 35 designs therefore form the design of experiments of the secondary optimisation with the remaining 70 objective evaluations used purely to update the model.

	Traditional Optimisation	Geometric Filtration & Hybrid Tuning	Percentage Improvement
Average C_D	0.0170	0.0169	0.8%
Standard Deviation C_D	2.6×10^{-4}	2.9×10^{-4}	-12.9%
Total Tuning Time	4.82	1.28	73.4%

Table 8.2: A comparison of the results of the traditional kriging optimisation and the geometric filtration optimisation with hybrid alternate tuning.

The 50 optimisations obtain an average drag coefficient of 0.0169, which equates to a 17.4% improvement in drag over the baseline wing design. As shown in Table 8.2, this equates to a 0.8% improvement over the drag obtained by the traditional kriging optimisation but at the cost of a 12.9% increase in the consistency between the final designs. The standard deviation increased from 2.6×10^{-4} when using the traditional strategy to 2.9×10^{-4} when using geometric filtration. These results are illustrated by the optimisation histories of Figure 8.4. As with Figure 8.2 the outlying design of Figure 8.4 is the result of an under prediction of viscous drag and has been removed from any statistical calculations.

The application of the geometric filtration strategy results in a significant reduction in overall tuning time, with the total time spent tuning hyperparameters dropping by approximately 73% from 4.8 hours to 1.3 per run. Not only does the application of the hybrid tuning strategy in conjunction with tuning after alternate updates, reduce the total number of likelihood evaluations but the reparameterisation procedure reduces the size of the correlation matrix from 130×130, at the final tune of the initial optimisation, to 35×35 at the initial tune of the secondary optimisation. In contrast, the correlation matrix constructed by the traditional strategy increases continually from 70×70 to 200×200 by the final hyperparameter tune. As observed in Chapter 6 such a reduction in correlation matrix size, and to a lesser extent the reduction in the number of variables in the secondary optimisation, results in a significant reduction in tuning expense.

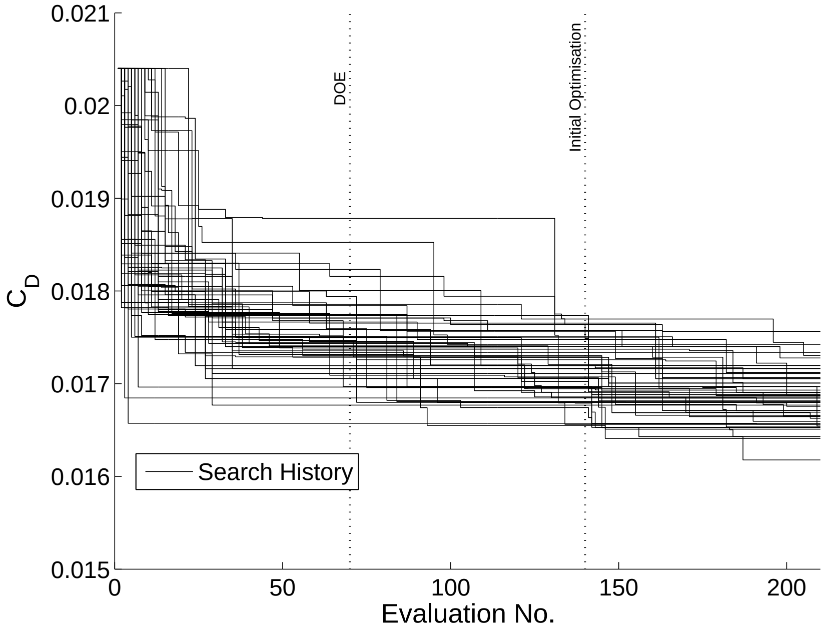


FIGURE 8.4: Search histories of the combination geometric filtration and hybrid alternate tune optimisation.

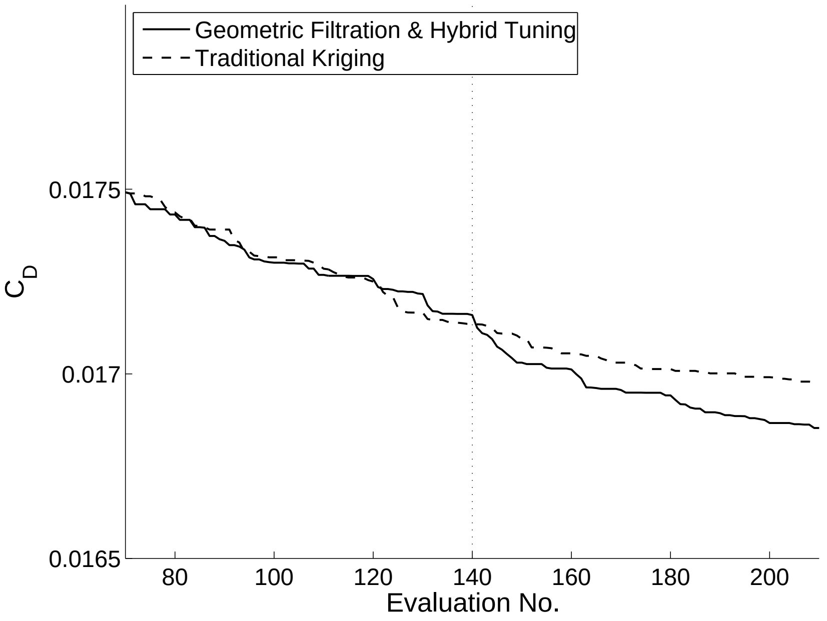


FIGURE 8.5: Comparison of the convergence of the transonic wing design problem after the DOE when employing traditional kriging and geometric filtration with alternate hybridised tuning.

Figure 8.5 compares the average optimisation histories of both strategies. Over the course of the initial optimisation the geometric filtration strategy keeps pace with the traditional strategy very well and only begins to lose out over the final two sets of updates. This is impressive considering that, in terms of the equivalent number of likelihood evaluations, the geometric filtration strategy employs only 11% of the evaluations carried out by the traditional approach.

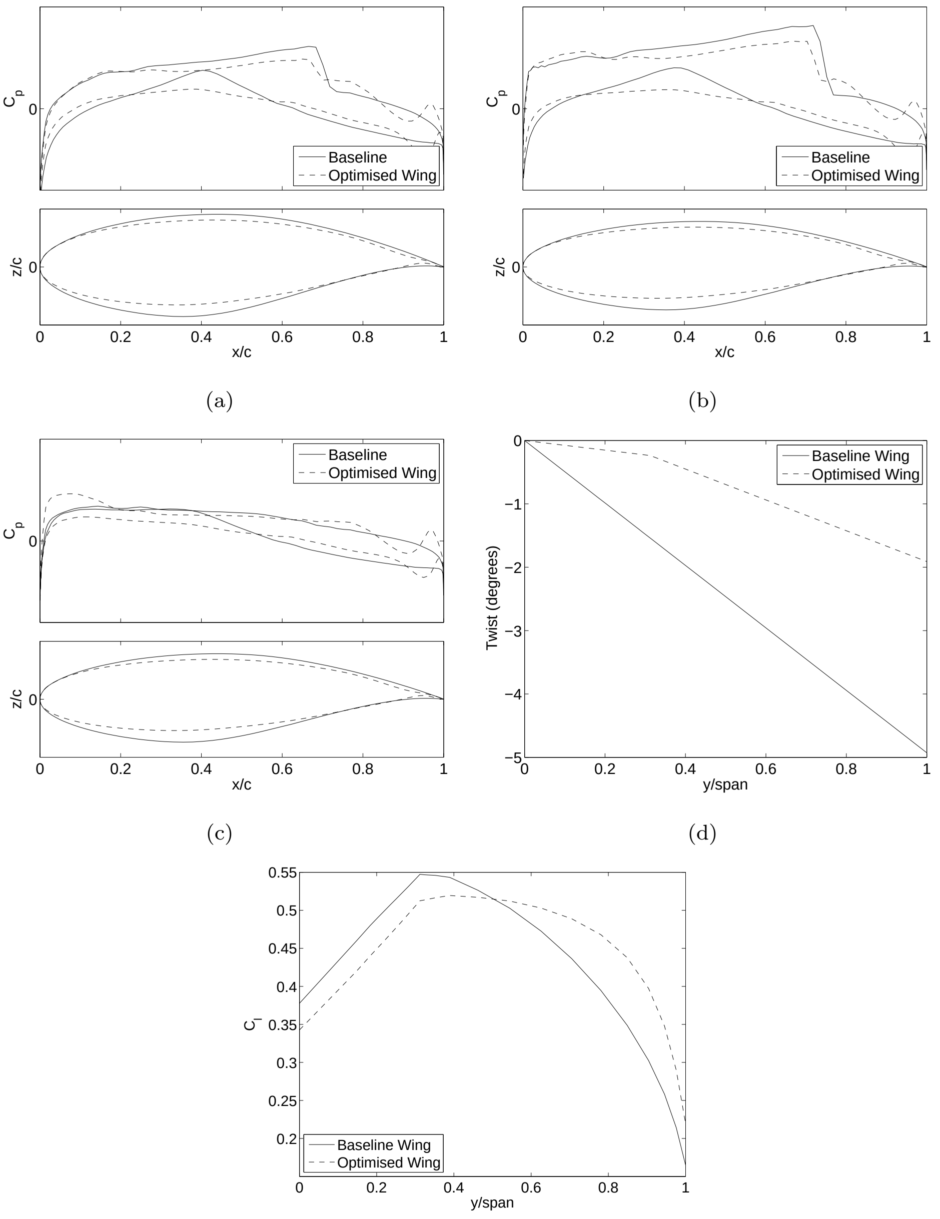
The reparameterisation procedure, employing 35 of the designs resulting from the initial optimisation, results in an average of 12.7 POD bases being required to reconstruct 99.995% of the cumulative percentage variation. This equates to a 9% reduction in the number of variables in the secondary optimisation. Although the number of variables does reduce, it is not to the same extent observed with the aerofoil optimisation of Chapter 6. However, unlike the aerofoil optimisation, a slightly larger cumulative percentage variation is required to recreate the original wings to the desired degree and to therefore have them effectively exist within the design space of the secondary optimisation. Nevertheless when the reparameterisation is employed in the secondary optimisation, the geometric filtration strategy immediately begins to outperform the traditional strategy, Figure 8.5.

Figure 8.6 demonstrates the changes in both the sectional geometry and spanwise twist distribution of a wing resulting from an example geometric filtration optimisation. The thickness of the wing has been reduced around 10-100% chord on the upper surface and around 0-70% chord on the lower surface. The lower surface of the trailing edge also has a slight indentation around the 95% chord point. Like Figure 8.3, the degree of twist has been reduced but to a lesser extent.

Like the design presented in Figure 8.3 the shockwave on the upper wing surface has been moved forward, but not quite as far. Whereas the shockwave at the wing's crank, Figure 8.3(b), was at approximately 70% of the chord, the shockwave is now at approximately 75% chord, much closer to the shock's original position on the baseline wing. The strength of the shockwave on the wing's upper surface has also been reduced slightly over the design of Figure 8.3.

The reduction in shock strength has once again reduced the thickening of the boundary layer aft of the shock. However, as the shockwave of Figure 8.6 is closer to the trailing edge of the wing than that of Figure 8.3 the region of the wing affected by the increase in skin friction drag due to the shockwave is smaller. This and the removal of the adverse pressure gradient found along the wing leading edge towards the wingtip, Figure 8.3(c) results in a reduction in the viscous drag coefficient. The viscous drag coefficient of the wing presented in Figure 8.6 is 0.0093 compared to 0.0100 of the wing presented in Figure 8.3.

As with the design resulting from the traditional optimisation, the wing loading has been increased towards the wingtip as the optimisation attempts to reduce the wave and viscous drag but maintain the required lift coefficient.



(a)

(b)

(c)

(d)

(e)

Figure 8.6: Comparison of the root (a), crank (b) and tip (c) aerofoils, sectional pressure, twist distribution (d) and spanwise loading (e) of a typical design resulting from the geometric filtration optimisation ($C_D = 0.0168$) to the baseline wing.

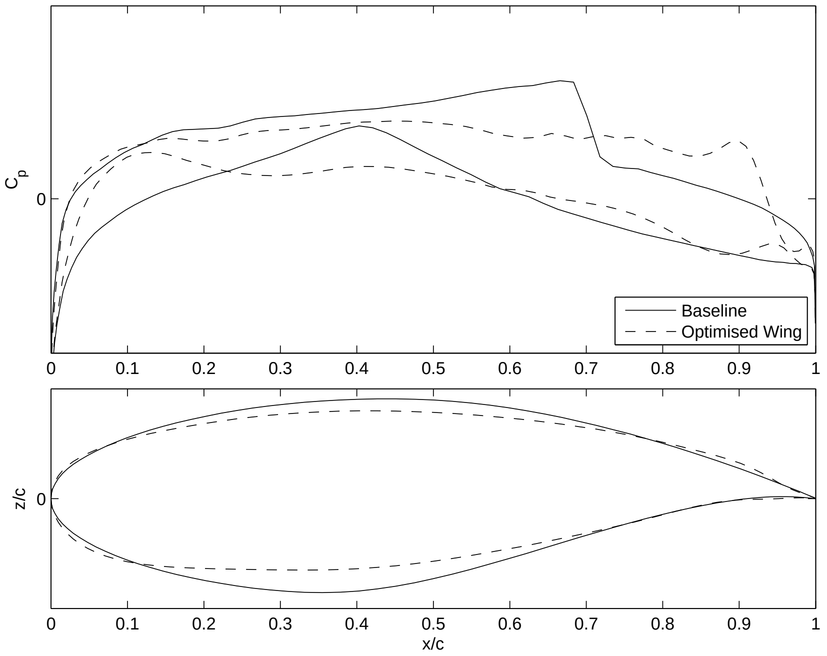


FIGURE 8.7: Root wing section of a final design, ($C_D = 0.0169$), where the secondary parameterisation is capable of producing geometries impossible with the initial parameterisation.

Figure 8.7 presents the root aerofoils section and pressure distribution of an additional design resulting from the geometric filtration optimisation. While this design demonstrates the reduction in drag through the movement of the shockwave to the trailing edge of the wing it also demonstrates an important feature of the reparameterisation process.

In this case the parameterisation of the secondary optimisation is capable of producing designs not previously possible in the initial parameterisation. Consider the upper surface of Figure 8.7. Here, the aerofoil thickness has been increased around both the leading and trailing edges while simultaneously being reduced in between. The addition of two Hicks-Henne bump functions to this surface, as is the case in the initial parameterisation, would be incapable of reproducing such a geometry. Rather, the reparameterisation procedure has identified that local changes to the leading and trailing edges are required along with a more general change in the 10-70% chord region. The same is true for the lower surface with the leading edge radius increased, the thickness reduced between the 10-70% chord region and a small bump added to the trailing edge. Geometric filtration can therefore not only reduce the number of variables but, for some parameterisations, simultaneously expand the secondary parameterisation to regions of the design space unobtainable with the initial parameterisation.

This can be explained further by considering the simple Hicks-Henne based parameterisation presented in Figure 8.8. Here a one variable parameterisation has been defined consisting of a single Hicks-Henne function where the position of the peak is permitted to vary and the amplitude and sharpness parameter remain a constant. Clearly the original parameterisation can produce a peak at only one point. Assuming two promising designs were returned, one with a peak at $x = 0.1$ and another with a peak at $x = 0.9$ and the proper orthogonal decomposition of a snapshot ensemble containing these two designs is carried out, only one basis function is required to completely recreate both of the original geometries. The resulting mean, shown in Figure 8.8 along with the POD basis function, is not a flat line, instead it contains two peaks the heights of which are augmented by the addition of the basis function times a modal coefficient. Obviously when the modal coefficient is equal zero the resulting design will contain two bumps. The process of geometric filtration has therefore resulted in a new parameterisation which has expanded the design space to include designs not previously possible with the original parameterisation but which can still exactly recreate the two selected designs from the original parameterisation.

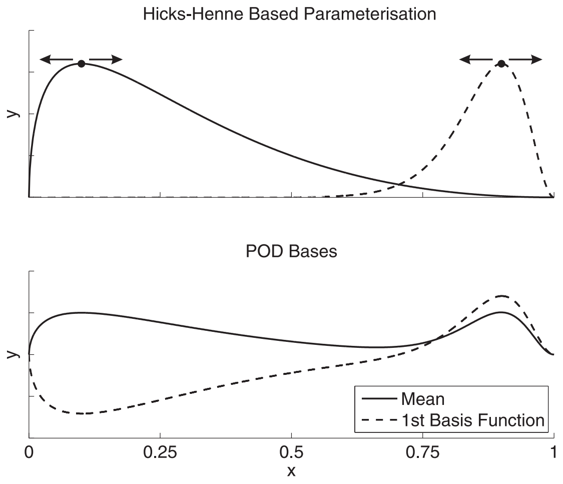


FIGURE 8.8: A simple example of a design space expansion using Geometric Filtration.

8.5 Potential Time Savings

Although a 14 variable problem is considered here, the potential reduction in tuning cost will improve further as the number of dimensions and hence the number of updates and function evaluations increases. Figure 1.3 of Chapter 1 demonstrated the total cost of the hyperparameter tuning process given a $15d$ total evaluation budget, a $5d$ DOE, updates added in batches of 10 evaluations and the hyperparameter optimisation requiring 10,000 evaluations of a Matlab coded likelihood function, after every update.

The timings presented in Figure 1.3 were calculated based on a full factorial sampling of timings for a single evaluation of the likelihood function given a particular dimensionality and sampling density. The resulting timings were then interpolated and used to predict the time for 10,000 likelihood evaluations at each stage of the entire optimisation process as updates are added to the kriging model. A total tuning time could then be calculated through a summation of these individual times. Utilising this data it is possible to ascertain an approximation to the total time each of the presented tuning strategies spends tuning hyperparameters throughout the course of an optimisation. The variation in total tuning time over a range of problem dimensionalities is presented graphically in Figure 8.9 for each tuning strategy.

Applying the alternate strategy reduces the total cost by approximately 52%, whilst the application of the hybrid tuning strategy reduces the total cost by approximately 80%. Applying the geometric filtration strategy, assuming a 25% reduction in the number of variables upon reparameterisation and a snapshot ensemble of approximately 2.5d, reduces the total tuning cost by approximately 66.5%. The application of these strategies in isolation to a 30 variable design problem, for example, would reduce the total tuning time from 93.1 hours, to approximately 45.0 hours when tuning after alternate updates, 32.2 hours when applying geometric filtration and 18.6 hours when utilising the hybrid tuning strategy.

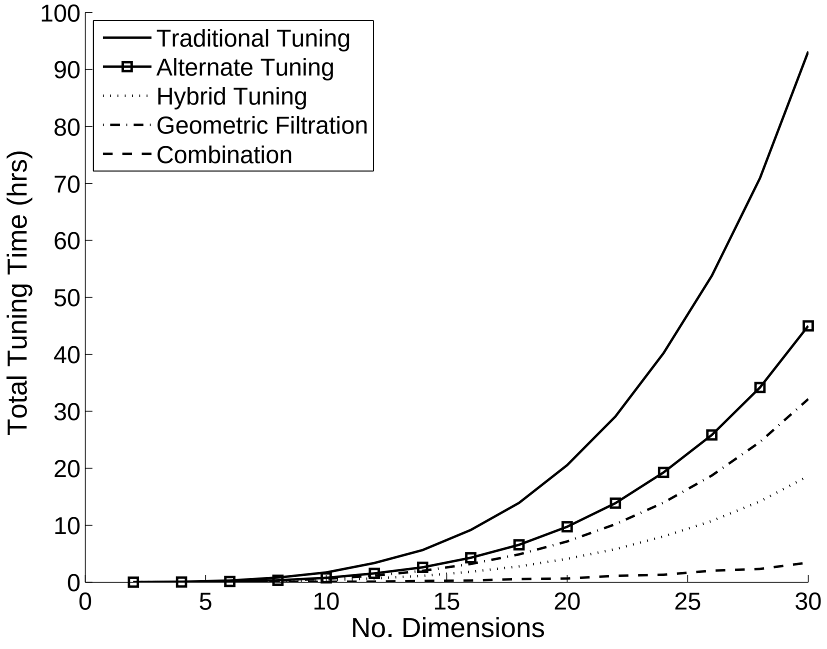


Figure 8.9: A Comparison of the potential tuning cost savings when employing the alternate tuning strategy, hybridised tuning, geometric filtration and a combination of the three.

The application of all three strategies however, has the potential to reduce the total tuning cost by approximately 96%. This is a significant improvement and would reduce the total cost incurred in tuning the hyperparameters of a 30 variable problem to approximately 3.5 hours. Such a large reduction of over 89 hours is enough to carry out a series of additional function calls of even a high fidelity computational simulation if desired.

The 96% reduction in tuning time could be reduced even further by the application of a restricted dataset in the construction of the correlation matrix used in the tuning process. As demonstrated in Section 7.5 reducing the size of the correlation matrix in such a manner can have a considerable impact on the total tuning cost incurred over the course of an optimisation. Questions remain however, as to how much the dataset should be restricted, which points should be retained and how this would affect the optimisation if, say, expected improvement were used to generate update points.

8.6 Conclusions

In this chapter hybrid hyperparameter tuning, alternate tuning and geometric filtration have been demonstrated in tandem to be effective in the optimisation of a transonic wing. This combination of optimisation strategies not only considerably reduces the total tuning time but does so while offering a slight improvement in the quality of the final designs.

A traditional kriging based optimisation, employing the “Heavy” tuning strategy of 5,000 likelihood evaluations of both a genetic algorithm and dynamic hill climb achieved an average drag coefficient of 0.0170 which equates to a 16.8% reduction over the baseline wing design. Combining the three optimisation strategies presented within this thesis resulted in an average drag coefficient of 0.0169, an improvement of 0.8% over that obtained by the traditional optimisation.

Tuning the hyperparameters over the course of the traditional optimisation took, on average, a total of 4.8 hours per run, while a total of only 1.3 hours were spent tuning over the course of the proposed strategy. This equates to a considerable, 73.4% reduction in overall tuning time.

When applied to the Hicks-Henne based wing parameterisation, the geometric filtration strategy demonstrated an interesting result. Not only did the POD based reparameterisation procedure reduce the total number of variables in the secondary optimisation by approximately 9%, but the secondary parameterisation was capable of producing geometries impossible to reproduce with the initial parameterisation. The upper surface of one final design was demonstrated to require the addition of more than two bump functions to recreate the geometry. The geometric filtration procedure was therefore able to recognise that bumps were required in more than two locations with the resulting POD bases then reflecting this. Geometric filtration can therefore reduce the total number of variables while, for some parameterisations, simultaneously expanding the secondary parameterisation to regions of the design space previously unobtainable.

Based on the assumption of a like for like comparison it is predicted that the combination of geometric filtration and hybrid and alternate tuning would result in a 96% reduction in total tuning time. The total tuning time of a 30 variable design optimisation would therefore drop from 93.1 hours to much more acceptable 3.5 hours.

Chapter 9

Conclusions & Recommendations for Further Work

9.1 Conclusions

The proliferation of surrogate modelling in the field of engineering design optimisation has resulted in the effective application of expensive, high fidelity computer simulations within the optimisation process. Requiring fewer function evaluations than direct global optimisation techniques, such as genetic algorithms or simulated annealing, surrogate modelling techniques, such as kriging, construct a surrogate of the response of an objective function to changes in the variables from an initial sampling of the design space. The resulting model is then updated in regions of interest found by searching the model with a global optimiser. While the number of true objective function evaluations is reduced, the construction of such surrogates involves the global optimisation of a multi-modal likelihood function which can be time consuming, particularly at high dimensions.

The research contained within this thesis is aimed at reducing kriging hyperparameter tuning costs and improving the performance of kriging optimisations at high dimensions. To this end a number of important contributions to the field of kriging based optimisation have been made:

- The global reassessment of kriging hyperparameters, as updates are added, has been demonstrated to be a necessity.
- The tuning of hyperparameters after every other set of updates can be effective and reduce the total tuning cost.
- An efficient adjoint of the likelihood function has been derived.
- A hybridised particle swarm algorithm has been developed which makes effective use of the adjoint of the likelihood.

- A novel kriging based optimisation strategy, termed geometric filtration, has been developed which attempts to capture the common features of good designs and develop a reparameterisation based upon them.

The optimisation of the likelihood function required in the construction of a kriging model is often referred to as hyperparameter tuning. In Chapter 3 a number of different tuning strategies were investigated with regard to the inverse design of a transonic aerofoil. The degree and frequency of the hyperparameter tuning process and the number of hyperparameters optimised, were all considered. It was demonstrated, and reinforced in Chapter 5, that a continual global reassessment of a kriging model's hyperparameters is a necessity for an effective optimisation. A single global optimisation of the hyperparameters after only the design of experiments, for example, was found to severely hamper the optimisation process, as was a single global optimisation followed by only local improvement. However, it was shown that a continual, but reduced, global reassessment of the hyperparameters, (tuning after alternate updates), could reduce the overall tuning cost by approximately 50% but have little impact on the quality of the final design.

To reduce the cost of the likelihood optimisation an adjoint of the likelihood was derived via manual reverse algorithmic differentiation. This adjoint was demonstrated to be considerably more efficient than a more traditional analytical differentiation of the likelihood and was found to be capable of calculating the likelihood and all its gradients for approximately twice the cost of a single likelihood evaluation. The adjoint was also demonstrated to be less cost sensitive to an increase in the number of sampling points, especially at high dimensions.

An efficient hybrid optimisation algorithm based on the combination of a particle swarm, SQP and a particle reinitialisation procedure was developed. The controlling parameters of this algorithm were derived through a series of optimisations aimed at maximising the algorithms performance with respect to hyperparameter tuning. The final strategy employs the equivalent of approximately 2,000 likelihood evaluations, an 80% reduction over the tuning strategy of Chapter 3 whilst obtaining comparable results. A 30 variable aerofoil design problem was also used to compare the hybrid strategy to a GA-DHC tuning strategy of comparable cost. The hybrid strategy demonstrated a considerable acceleration in convergence, achieving the final design of the GA-DHC strategy with 100 fewer function evaluations.

A novel optimisation strategy involving two kriging based response surface optimisations and a proper orthogonal decomposition (POD) based reparameterisation, termed geometric filtration, has been introduced and investigated in considerable detail. The strategy employs an initial kriging optimisation of the original design problem with a subset of the resulting designs forming a snapshot ensemble for the purposes of POD. The resulting POD basis functions then act as a reparameterisation of the original problem, filtering out badly performing designs, reducing the number of variables and in some instances allowing regions of the design space not available in the original parameterisation to be explored. The basic geometric filtration strategy achieved a 3% improvement over the designs obtained using a traditional kriging strategy while also reducing the total tuning time from 22 to 4.2 hours. Upon careful investigation of the effort expended in the initial optimisation, the number of snapshots and the number of bases functions selected, the strategy achieved a 4.5% improvement over the traditional kriging optimisation for a total tuning time of 2.5 hours.

The optimisation and tuning strategies investigated, were demonstrated to be just as effective when combined. The optimisation of a 14 variable wing design problem was utilised to compare the traditional kriging approach to a combination of geometric filtration, hybrid hyperparameter optimisation, and alternate tuning. This combination of strategies achieved an average 0.8% improvement in the drag obtained by the traditional optimisation but for a significant reduction in overall tuning cost. The 4.8 hours spent tuning hyperparameters when using the traditional optimisation was reduced to a total of 1.3 hours, which equates to a 73.4% reduction in tuning cost.

In conclusion, the work presented in this thesis has gone some way to achieving a considerable reduction in kriging hyperparameter tuning cost whilst resulting in an improvement in final design performance. Be it through the likelihood adjoint's inexpensive gradients which are relatively cost insensitive to both problem dimensionality and the number of sampling points, the simplicity of the alternate tuning strategy, or the complexity of geometric filtration or the hybrid particle swarm. The strategies developed can work in isolation or in combination to reduce the cost of the hyperparameter tuning process and result in more efficient optimisations where the effort can be expended in evaluating designs rather than in constructing surrogate models.

9.2 Recommendations for Further Work

9.2.1 Likelihood Adjoint and Tuning Strategies

Although a number of improvements with respect to the cost and performance of kriging have been demonstrated throughout the course of this thesis, there remain a number of areas ripe for further investigation.

Chapters 4 and 5 introduced and demonstrated the efficiency of an adjoint of the concentrated likelihood function employed in the kriging hyperparameter tuning process. The application of this cheap gradient information proved extremely effective in reducing the total number of likelihood evaluations. However, within the literature there are a number of extensions and modifications to the basic kriging strategy which also require a likelihood optimisation in their construction.

Gradient and Hessian information, if available, can be exploited in gradient and Hessian enhanced kriging models where the resulting surrogate not only represents the given objective function values but matches the first and second derivatives of the objective function. Such models can accurately represent the true design space with relatively few sampling points, (Leary et al. [2004]).

Objective functions obtained from simulations of varying fidelity can be exploited through co-kriging. This allows a large quantity of cheap data to be coupled with a small quantity of expensive data, for example, data from a simple panel code can be coupled with data from a RANS simulation. The resulting model constructed from a large number of cheap evaluations and corrected by a few expensive evaluations is more accurate than a model constructed from the expensive calculations alone, Forrester et al. [2007].

As with kriging the construction of such models requires the selection of an appropriate set of hyperparameters. In the case of gradient and Hessian enhanced kriging the hyperparameter optimisation process can be simplified to some extent with the restriction, $p = 2$. However, at high dimensions the size of the correlation matrix grows considerably. The correlation matrix of a gradient enhanced kriging model, for example, must not only contain correlations between data points but also between data points and gradients and between gradients and themselves. The size of the correlation matrix of a gradient enhanced krig is therefore $(d+1)n \times (d+1)n$ while a Hessian enhanced krig, which includes correlations between second derivatives, is $(2d+1)n \times (2d+1)n$. Such large matrices incur a high factorisation cost which naturally makes the hyperparameter tuning process an expensive one.

An adjoint of the likelihood of a gradient or Hessian enhanced kriging model may therefore be extremely effective in reducing model construction cost. Ascertaining an efficient hybridised optimiser to utilise such an adjoint would also be an interesting goal. With such vast quantities of data it may prove to be the case that a simple gradient based local search is all that is required.

Co-kriging suffers from slightly different issues with respect to hyperparameter tuning, but nevertheless may still benefit from an adjoint of the likelihood. Co-kriging approximates the expensive data as a model of the cheap data, multiplied by a scaling factor ρ, plus a second model which represents the difference between the cheap scaled data and the expensive data. There are therefore twice as many hyperparameters and a scaling constant requiring optimisation. The number of hyperparameters increases further as more levels of fidelity are included in the overall model. More hyperparameters coupled with the cheapness of the low fidelity simulations equates to both more likelihood optimisations and more expensive likelihood evaluations as more data points are included.

Chapter 5 explored the performance of a number of different global optimisation methodologies with respect to hyperparameter optimisation. Missing from the list of strategies, however, was the technique of surrogate modelling itself. Although the construction of a kriging model of the likelihood space is not advocated here, as it would of course lead to the perpetual tuning of an ever increasing number of hyperparameters, it would be possible to construct a gradient enhanced RBF model relatively cheaply. The question however, is at what point does the construction and searching of such a model become cheaper than the original likelihood evaluations? Only at this point is a surrogate based optimisation of the likelihood plausible. This was not the case for the examples presented in Chapter 5 but such a strategy may become efficient when the correlation matrix becomes extremely large such as in the construction of a high dimensional gradient or Hessian enhanced kriging model.

9.2.2 Geometric Filtration

The geometric filtration strategy, introduced in Chapter 6, has been demonstrated to be effective at further reducing the cost of the hyperparameter tuning process whilst producing equivalent or better final designs than a traditional kriging optimisation. However, there remain a number of aspects of this strategy which could be investigated in the future. The mixing of parameterisations and a repeated reparameterisation process could be considered.

In Chapters 6 and 7 geometric filtration was applied to the optimisation of an aerofoil parameterised via two NURBS curves. In Chapter 8 the strategy was applied to the optimisation of a wing, the sections of which were modified through a series of Hicks-Henne bump functions. However, there exist within the literature a range of different parameterisation techniques.

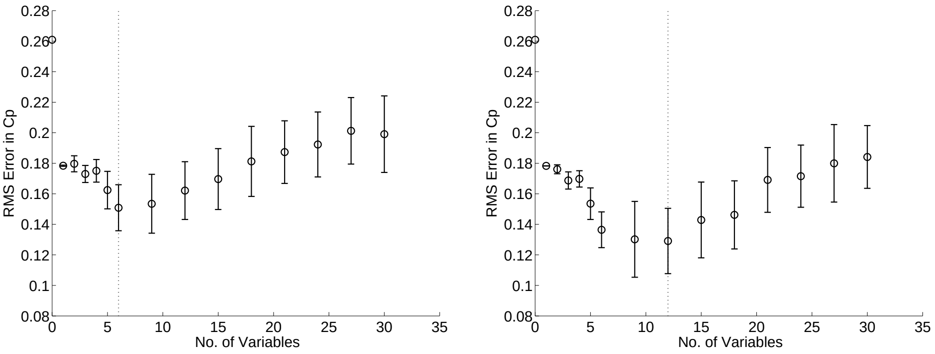
Kulfan [2008], for example, introduced shape and class functions which relate directly to recognisable features of an aerofoil geometry, such as the leading edge radius or boat-tail angle. By representing the shape function as an n^{th} order Bernstein polynomial any aerofoil can be recreated and optimised by altering the coefficients. Sobieczky [1998] introduced the PARSEC method of aerofoil generation via 11 basic and recognisable parameters such as leading edge radius, thickness and boat tail angle. Although similar to the method of Kulfan [2008] in this respect, it can only be applied to the parameterisation of aerofoils and wings. Bloor and Wilson [1995] introduced the PDE method, which generates surfaces via a series of partial differential equations.

Given a particular optimisation problem, the question remains as to which parameterisation is the most effective. A study could be performed on each and the most suitable parameterisation selected, but this may waste valuable information from the simulations of the other parameterisations. Geometric filtration could therefore be employed to combine a series of parameterisations into a new secondary parameterisation which captures the best features of all of them. However, careful consideration would be required as to the number of snapshots selected from each parameterisation.

Very high dimensional problems may benefit from a repeated reparameterisation procedure. At high dimensions, given a suitably large evaluation budget it may be possible to split the optimisation process into a series of sub-optimisations each separated by a POD based reparameterisation. The optimisation would therefore see a gradual reduction in the number of variables as the optimisation progresses which, it is hoped, would gradually converge on the best design features. A repeated reparameterisation would also drastically reduce the cost of the hyperparameter tuning process as the correlation matrix is repeatedly reduced in size. However, careful consideration must be given to the sufficient exploration of the design space before each reparameterisation. Failure to do so might result in suboptimal performance with snapshots constructed from poorer designs.

Appendix A

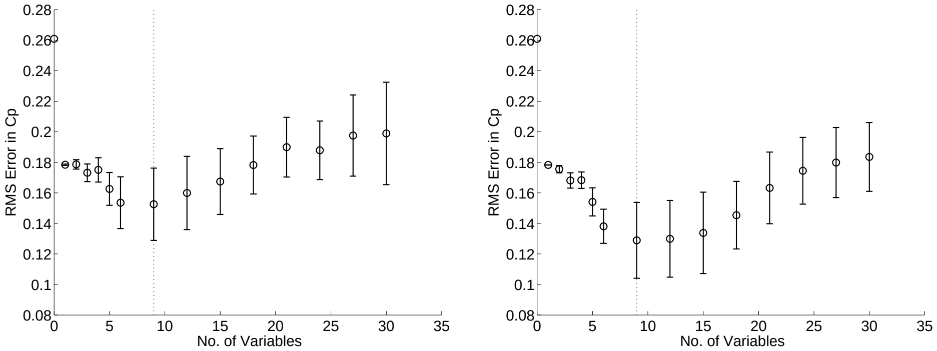
Hyperparameter Tuning Strategy Graphs



(a)

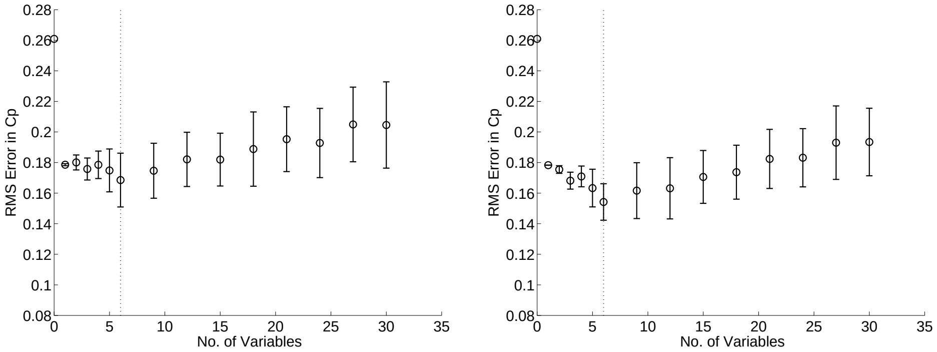
(b)

Figure A.1: Averaged results of the inverse design optimisation using the Heavy tuning strategy and a budget of 75 simulations (a) and 150 simulations (b); best result indicated by a dotted line, error bars indicate ± one standard deviation.



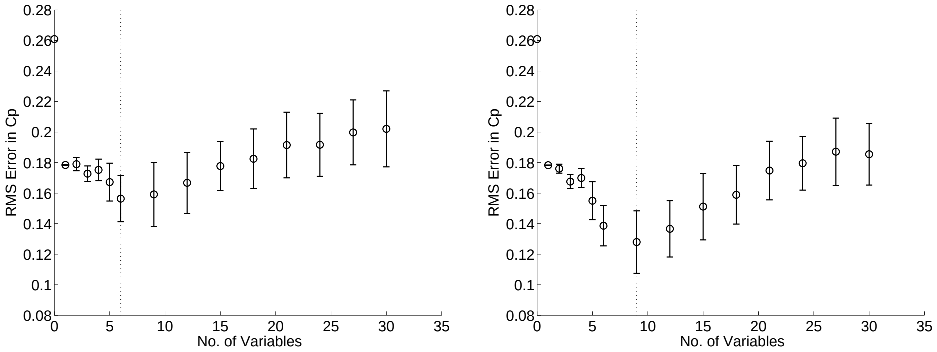
(a) (b)

Figure A.2: Averaged results of the inverse design optimisation using the Light tuning strategy and a budget of 75 simulations (a) and 150 simulations (b); best result indicated by a dotted line, error bars indicate ± one standard deviation.



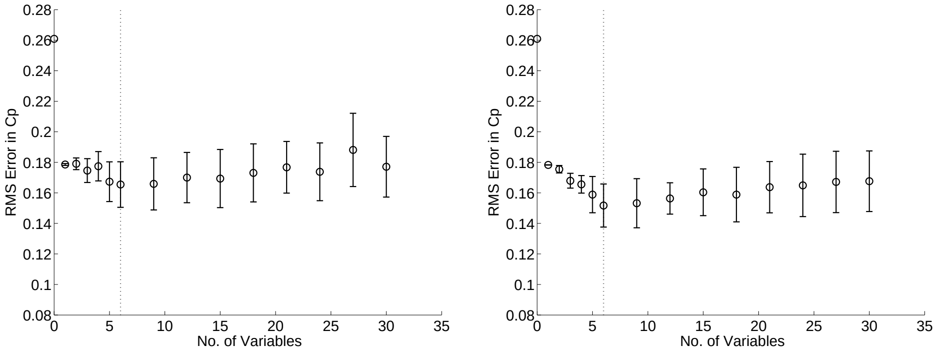
(a) (b)

Figure A.3: Averaged results of the inverse design optimisation using the Single tune strategy and a budget of 75 simulations (a) and 150 simulations (b); best result indicated by a dotted line, error bars indicate ± one standard deviation.



(a) (b)

Figure A.4: Averaged results of the inverse design optimisation using the Alternate tuning strategy and a budget of 75 simulations (a) and 150 simulations (b); best result indicated by a dotted line, error bars indicate ± one standard deviation.



(a)

(b)

Figure A.5: Averaged results of the inverse design optimisation using the ? tuning strategy and a budget of 75 simulations (a) and 150 simulations (b); best result indicated by a dotted line, error bars indicate ± one standard deviation.

Appendix B

Reverse Mode Likelihood Sub-Functions

```
function [$\boldsymbol{L}$] = Cholesky[$\boldsymbol{R}$]

% initialize L as the lower triangle of R
for $k$ = 1:$n$
    $\boldsymbol{L}(kk) = \sqrt{\boldsymbol{L}(kk)}$
    for $j$ = $k$+1:$n$
        $\boldsymbol{L}(jk) = \frac{\boldsymbol{L}(jk)}{\boldsymbol{L}(kk)}$
    end
    for $j$ = $k$+1:$n$
        for $i$ = $j$:$n$
            $\boldsymbol{L}(ij) = \boldsymbol{L}(ij) - \boldsymbol{L}(ik)\boldsymbol{L}(jk)$
        end
    end
end
```

Cholesky factorization, Smith [1995]

```
function [$\bar{\boldsymbol{F}}$] = Reverse_Cholesky[$\boldsymbol{L}$,$\bar{\boldsymbol{L}}$]

$\bar{\boldsymbol{F}} = \bar{\boldsymbol{L}}$
for $k$ = $n$:1
    for $j$ = $k$+1:n
        for $i$ = $j$:$n$
            $\bar{\boldsymbol{F}}(ik) = \bar{\boldsymbol{F}}(ik) - \bar{\boldsymbol{F}}(ij)\boldsymbol{L}(jk)$
            $\bar{\boldsymbol{F}}(jk) = \bar{\boldsymbol{F}}(jk) - \bar{\boldsymbol{F}}(ij)\boldsymbol{L}(ik)$
        end
    end
    for $j$ = $k$+1:$n$
        $\bar{\boldsymbol{F}}(jk) = \frac{\bar{\boldsymbol{F}}(jk)}{\boldsymbol{L}(kk)}$
        $\bar{\boldsymbol{F}}(kk) = \bar{\boldsymbol{F}}(kk) - \boldsymbol{L}(jk)\bar{\boldsymbol{F}}(jk)$
    end
    $\bar{\boldsymbol{F}}(kk) = \frac{\bar{\boldsymbol{F}}(kk)}{2\boldsymbol{L}(kk)}$
end
```

Reverse Cholesky factorization, Smith [1995]

```
function [$\boldsymbol{T}_1$] = fwardsub[$\boldsymbol{L}$,$\boldsymbol{a}$]
% Where $\boldsymbol{a}$ = $(\boldsymbol{y} - \boldsymbol{1}\hat{\mu})$

$\boldsymbol{T}_1(1) = \frac{\boldsymbol{a}(1)}{\boldsymbol{L}(11)}$
for $i$ = 2:$n$
    $S = 0$
    for $j$ = 1:$i-1$
        $S = S + \boldsymbol{L}(ij)\boldsymbol{T}_1(j)$
    end
    $\boldsymbol{T}_1(i) = \frac{\boldsymbol{a}(i)-S}{\boldsymbol{L}(ii)}$
end
```

Forward Substitution, Press et al. [1986]

```
function [$\bar{\boldsymbol{L}}_2$] = Reverse_fwardsub[$\bar{\boldsymbol{T}}_1$,$\boldsymbol{L}$,$\boldsymbol{T}_1$]

for $i$ = $n$:2
    $\bar{\boldsymbol{a}}(i) = \frac{\bar{\boldsymbol{T}}_1(i)}{\boldsymbol{L}(ii)}$
    $\bar{\boldsymbol{L}}_2(ii) = -\bar{\boldsymbol{a}}(i)\boldsymbol{T}_1(i)$
    for $j$ = 1:$i-1$
        $\bar{\boldsymbol{L}}_2(ij) = -\bar{\boldsymbol{a}}(i)\boldsymbol{T}_1(j)$
        $\bar{\boldsymbol{T}}_1(j) = \bar{\boldsymbol{T}}_1(j) - \bar{\boldsymbol{a}}(i)\boldsymbol{L}(ij)$
    end
end
$\bar{\boldsymbol{a}}(1) = \frac{\bar{\boldsymbol{T}}_1(1)}{\boldsymbol{L}(11)}$
$\bar{\boldsymbol{L}}_2(11) = -\bar{\boldsymbol{a}}(1)\boldsymbol{T}_1(1)$
```

Reverse Differentiation of Forward Substitution

```
function [$\boldsymbol{T}_2$] = bwardsub[$\boldsymbol{L}^T$,$\boldsymbol{T}_1$]

$\boldsymbol{T}_2(n)$ = $\frac{\boldsymbol{T}_1(n)}{\boldsymbol{L}^T(nn)}$
for $i$ = $n-1$:1
    $S = 0$
    for $j$ = $i+1$:$n$
        $S = S + \boldsymbol{L}^T(ij)\boldsymbol{T}_2(j)$
    end
    $\boldsymbol{T}_2(i) = \frac{\boldsymbol{T}_1(i)-S}{\boldsymbol{L}^T(ii)}$
end
```

Back Substitution, Press et al. [1986]

```
function [$\bar{\boldsymbol{L}}_1$,$\bar{\boldsymbol{T}}_1$] = Reverse_bwardsub[$\bar{\boldsymbol{T}}_2$,$\boldsymbol{L}^T$,$\boldsymbol{T}_2$]

for $i$ = 1:$n-1$
    $\bar{\boldsymbol{T}}_1(i) = \frac{\bar{\boldsymbol{T}}_2(i)}{\boldsymbol{L}^T(ii)}$
    $\bar{\boldsymbol{L}}_1(ii) = -\bar{\boldsymbol{T}}_1(i)\boldsymbol{T}_2(i)$
    for $j$ = $i+1$:$n$
        $\bar{\boldsymbol{L}}_1(ij) = -\bar{\boldsymbol{T}}_1(i)\boldsymbol{T}_2(j)$
        $\bar{\boldsymbol{T}}_2(j) = \bar{\boldsymbol{T}}_2(j) - \bar{\boldsymbol{T}}_1(i)\boldsymbol{L}^T(ij)$
    end
end
$\bar{\boldsymbol{T}}_1(n) = \frac{\bar{\boldsymbol{T}}_2(n)}{\boldsymbol{L}^T(nn)}$
$\bar{\boldsymbol{L}}_1(nn) = -\bar{\boldsymbol{T}}_1(n)\boldsymbol{T}_2(n)$
```

Reverse Differentiation of Back Substitution

Appendix C

Comparison of Hyperparameter Optimisers

	Heavy Tuning Strategy			Basic Particle Swarm		
No. of Variables	Mean Obj.	Standard Deviation	Tuning Time (hrs)	Mean Obj.	Standard Deviation	Tuning Time (hrs)
3	0.171	0.006	0.028	0.171	0.006	0.014
6	0.141	0.015	0.298	0.145	0.016	0.149
9	0.122	0.024	1.246	0.132	0.026	0.623
12	0.130	0.029	3.366	0.133	0.022	1.683
18	0.132	0.031	14.054	0.132	0.026	7.027

Table C.1: Details of the quality of the final designs resulting from the standard "Heavy" tuning strategy and the basic particle swarm.

	SQP Tuning Strategy			
No. of Variables	Mean Obj.	Standard Deviation	Mean No. of Likelihood Evals.	Tuning Time (hrs)
3	0.179	0.009	57.2	1.5×10^{-4}
6	0.157	0.011	101.9	0.003
9	0.147	0.022	245.9	0.031
12	0.153	0.019	314.9	0.11
18	0.145	0.021	487.2	0.69

Table C.2: Details of the quality of the final designs resulting from the SQP tuning strategy.

Heavy & SQP Strategy				
No. of Variables	Mean Obj.	Standard Deviation	Mean No. of Likelihood Evals.	Tuning Time (hrs)
3	0.173	0.008	84.8	0.003
6	0.161	0.014	37.5	0.014
9	0.156	0.021	67.0	0.041
12	0.165	0.014	85.0	0.091
18	0.167	0.013	74.1	0.276

Table C.3: Details of the quality of the final designs resulting from the a kriging optimisation with hyperparameters tuned initially via the "Heavy" strategy and subsequently by a SQP search commenced from the previous set of hyperparameters.

	GA-DHC (2000 Evaluations)			Hybrid Swarm Tuning Strategy		
No. of Variables	Mean Obj.	Standard Deviation	Tuning Time (hrs)	Mean Obj.	Standard Deviation	Tuning Time (hrs)
3	0.1725	0.0050	0.005	0.1710	0.0060	0.005
6	0.1429	0.0144	0.060	0.1480	0.0171	0.060
9	0.1242	0.0293	0.249	0.1202	0.0271	0.249
12	0.1268	0.0253	0.673	0.1313	0.0281	0.673
18	0.1395	0.0306	2.811	0.1358	0.0276	2.811

Table C.4: Details of the quality of the final designs resulting from the developed hybrid particle swarm tuning strategy.

Appendix D

Hybrid Particle Swarm Optimisation Results

Pop. Size	No. Gens	V_{max}	Local Search Gen.	Min. Local Evals.	Min. Swarm Size	Prob of Reinitialisation (Initial)	Prob of Reinitialisation (Final)	No. of Points Reinitialised (Initial)	No. of Points Reinitialised (Final)	Obj. Func.
32	50	0.810	39	3	17	0.05	0.50	0.80	0.52	-0.090
50	34	0.532	22	3	7	0.59	0.39	0.02	0.94	-0.086
77	21	0.761	17	8	8	0.77	0.36	0.80	0.78	-0.083
97	18	0.879	8	12	25	0.15	0.25	0.97	0.91	-0.083
97	18	0.879	8	36	20	0.15	0.25	0.97	0.04	-0.082
54	31	0.408	16	3	17	0.09	0.39	0.09	0.19	-0.082
60	28	0.574	21	12	46	0.59	0.14	0.41	0.29	-0.081
19	87	0.994	39	4	2	0.71	0.51	0.59	0.01	-0.081
60	28	0.573	24	12	24	0.78	0.15	0.40	0.17	-0.081
97	18	0.880	4	17	29	0.51	0.10	0.96	0.02	-0.081

Table D.1: Parameters for the 10 best performing hybrid swarms given an underlying two variable optimisation problem and a fixed budget of 2000 evaluations.

Pop. Size	No. Gens	V_{max}	Local Search Gen.	Min. Local Evals.	Min. Swarm Size	Prob of Reinitialisation (Initial)	Prob of Reinitialisation (Final)	No. of Points Reinitialised (Initial)	No. of Points Reinitialised (Final)	Obj. Func.
61	29	0.092	29	6	7	0.58	0.59	0.10	0.36	-1.115
81	20	0.061	6	14	46	0.61	0.53	0.06	0.28	-1.067
66	26	0.085	15	9	16	0.59	0.53	0.03	0.35	-1.032
62	25	0.078	19	2	51	0.63	0.43	0.14	0.64	-1.031
81	23	0.056	16	36	43	0.54	0.32	0.07	0.39	-1.031
11	154	0.128	150	3	7	0.51	0.53	0.78	0.36	-1.022
56	32	0.036	20	5	23	0.61	0.48	0.17	0.36	-1.020
61	17	0.114	10	15	28	0.67	0.03	0.18	0.07	-1.015
62	26	0.078	21	17	44	0.63	0.79	0.14	0.59	-1.009
70	26	0.078	15	2	46	0.60	0.49	0.16	0.37	-1.006

Table D.2: Parameters for the 10 best performing hybrid swarms given an underlying five variable optimisation problem and a fixed budget of 2000 evaluations.

			Local	Min.	Min.	Prob of		No. of Points		
Pop.	No.		Search	Local	Swarm	Reinitialisation		Reinitialised		Obj.
Size	Gens	V_{max}	Gen.	Evals.	Size	(Initial)	(Final)	(Initial)	(Final)	Func.
63	25	0.045	15	1	50	0.44	0.63	0.84	0.005	-3.149
15	99	0.083	77	6	9	0.52	0.67	0.93	0.04	-3.125
81	20	0.061	2	22	55	0.55	0.62	0.49	0.14	-3.024
16	99	0.064	81	1	10	0.55	0.17	0.96	0.04	-3.007
34	46	0.075	38	0	26	0.50	0.71	0.86	0.05	-2.994
74	21	0.065	16	6	50	0.55	0.20	0.54	0.04	-2.994
96	16	0.064	6	23	73	0.540	0.81	0.50	0.01	-2.989
10	166	0.077	94	0	10	0.540	0.81	0.71	0.002	-2.976
22	71	0.061	6	6	16	0.63	0.35	0.71	0.16	-2.966
20	80	0.074	26	8	3	0.56	0.13	0.99	0.01	-2.953

Table D.3: Parameters for the 10 best performing hybrid swarms given an underlying 10 variable optimisation problem and a fixed budget of 2000 evaluations.

			Local	Min.	Min.	Prob of		No. of Points		
Pop.	No.		Search	Local	Swarm	Reinitialisation		Reinitialised		Obj.
Size	Gens	V_{max}	Gen.	Evals.	Size	(Initial)	(Final)	(Initial)	(Final)	Func.
20	87	0.073	56	6	3	0.52	0.19	0.86	0.56	-4.038
24	73	0.094	7	7	6	0.52	0.19	0.58	0.54	-3.811
74	22	0.091	17	10	62	0.61	0.26	0.86	0.30	-3.467
24	71	0.100	11	11	13	0.52	0.21	0.97	0.53	-3.452
78	21	0.055	5	11	58	0.60	0.26	0.28	0.28	-3.444
56	27	0.084	28	5	50	0.97	0.05	0.98	0.38	-3.392
17	98	0.134	46	1	16	0.65	0.37	0.86	0.13	-3.388
99	17	0.084	14	46	53	0.52	0.23	0.97	0.53	-3.361
15	114	0.100	109	3	12	0.12	0.22	0.96	0.61	-3.357
37	42	0.095	9	3	34	0.64	0.31	0.84	0.93	-3.340

Table D.4: Parameters for the 10 best performing hybrid swarms given an underlying 15 variable optimisation problem and a fixed budget of 2000 evaluations.

			Local	Min.	Min.	Prob of		No. of Points		
Pop.	No.		Search	Local	Swarm	Reinitialisation		Reinitialised		Obj.
Size	Gens	V_{max}	Gen.	Evals.	Size	(Initial)	(Final)	(Initial)	(Final)	Func.
15	87	0.064	22	6	3	0.82	0.27	0.43	0.11	-8.075
15	85	0.059	6	6	2	0.73	0.24	0.74	0.08	-7.705
15	84	0.062	14	6	3	0.82	0.27	0.37	0.08	-7.527
15	85	0.061	15	5	4	0.82	0.26	0.64	0.11	-7.504
17	75	0.057	20	7	2	0.79	0.27	0.93	0.10	-7.284
15	88	0.060	11	5	2	0.76	0.27	0.21	0.12	-7.274
17	75	0.060	14	7	3	0.76	0.26	0.18	0.08	-7.270
16	79	0.058	8	6	4	0.84	0.27	0.06	0.06	-7.228
16	79	0.062	7	7	2	0.59	0.27	0.50	0.07	-7.225
16	81	0.063	9	7	2	0.86	0.28	0.30	0.09	-7.209

Table D.5: Parameters for the 10 best performing hybrid swarms given an underlying 25 variable optimisation problem and a fixed budget of 2000 evaluations.

Appendix E

Description & Validation of the FP Wing Analysis Process

E.1 Overview of the FP Algorithm

The inviscid full-potential (FP) method for three dimensional wing and wing-body combinations was developed and released by ESDU in 2002. The package encompasses grid generation, flow solution and post-processing, ESDU [2002].

FP generates computational meshes for both isolated wing and wing-body cases via a conformal mapping scheme, similar to that employed within VGK, ESDU [1996b]. All wing configurations are assumed to be symmetric about the aircraft centerline while any fuselage is assumed to be axially symmetric about the centerline. This provides a simple geometric definition of the fuselage but restricts the analysis of real world configurations where fuselages are not usually axis-symmetric, especially towards the tail. FP can however deal with reasonably complex wing geometries including wings with curved leading and trailing edges, edge discontinuities (cranks) and forward or rearward sweep. As the geometry is defined at a series of spanwise sections, spanwise variations in twist, camber and thickness can be easily represented.

With an appropriate conformal mesh generated, FP proceeds to compute the exact solution to the inviscid compressible three dimensional potential flow equations via the application of a finite differencing scheme. Convergence speed is improved by the adoption of a multi-grid scheme with three levels of grid fineness. The finest grid, corresponding to the original mesh of up to 115200 cells is employed at the final stage of the computation with medium and coarse grids, of 14400 and 7200 cells respectively, employed in the proceeding stages. Full convergence can be achieved in typically 800 iterations with 200 iterations using the coarse grid and 100 using the medium grid. Run convergence criteria can be set so the analysis completes when lift or drag has converged to a predefined degree.

Spanwise lift and drag coefficients are calculated through the integration of the computed pressure coefficients at each wing section, with the overall wing lift and drag coefficients calculated by integrating along the wing span. The embedded calculation of drag does not however, distinguish between vortex and wave drag, ESDU [2002].

Part four of the FP manual, ESDU [2006a], describes a post processor provided with the FP package which evaluates the trailing vortex drag and wave drag components of a wing's inviscid drag coefficient. Trailing vortex drag is calculated using a method based on linearised theory and therefore ignores the effects of rolling-up and downward deflection of the trailing vortex sheet, "Model A" of Ashill and Fulker [1987]. This method is discussed in more detail by Lock [1985]. The wave drag component is calculated via both the "first-order" and the "improved method" of Lock which is implemented in ESDU-87003, ESDU [1987]. These methods of calculating wave drag are also implemented within VGK, ESDU [1996b].

E.2 Viscous Drag Correction

E.2.1 Overview

As described above, the FP wing analysis package will only provide the vortex and wave drag components of a wing's total drag. The viscous drag component therefore remains an unknown. However, ESDU recently published a method for the prediction of the viscous drag coefficient for a wing in shock-free and attached flow. Such a prediction can therefore be added to the inviscid drag components from FP to obtain a prediction of the total drag for a wing.

The method described briefly in ESDU [2008a], with a full derivation presented in ESDU [2008b], is used to calculate a wing's minimum profile drag coefficient, C_{DPmin0}, increment in profile drag due to twist, $(\Delta C_{DPmin})_\varepsilon$, lift-dependent viscous drag factor, K_{visc} and lift coefficient for minimum viscous drag, C_{Lmin}. Each of these coefficients is then used to calculate the viscous drag coefficient of a wing via,

$$C_{Dvisc} = C_{DPmin0} + (\Delta C_{DPmin})_\varepsilon + K_{visc}(C_L - C_{Lmin})^2, \tag{E.1}$$

where C_L is the lift coefficient of the wing.

E.2.2 Minimum Profile Drag Coefficient of an Untwisted Wing

The local profile drag is defined as the force acting in the freestream direction at a specific station of the wing and therefore arises from two physical components, the local pressure and skin friction drag coefficients. ESDU [2008b] demonstrates that at a spanwise position along a wing the minimum local profile drag can be shown to be,

$$C_{dpmin} = (C_{dmin} - C_{df0}) \cos^2 \Lambda_{1/4} + C_{df0}. \tag{E.2}$$

With C_{dmin}, the minimum profile drag coefficient and C_{df0}, the zero lift skin friction coefficient of the streamwise aerofoil section in 2D flow and $\Lambda_{1/4}$ the local quarter chord sweep angle of the wing. Both C_{dmin} and C_{df0} can be obtained using ESDU [2006b] and ESDU [2006c] respectively. Alternatively the program described in ESDU [2006d] or indeed VGK, (which was in fact used in the derivation of these empirical methods), could be used.

A prediction of the minimum profile drag of an untwisted wing can then be calculated by integrating the local minimum profile drag across the wing,

$$C_{DPmin0} = \int_0^1 C_{dpmin} \left(\frac{c}{\bar{c}}\right) \mathrm{d}\eta. \tag{E.3}$$

It was demonstrated in ESDU [2008b] however, that given a relatively simple straight wing with a constant or linear spanwise variation in thickness and a constant camber and chordwise boundary layer transition point, the above integral can be approximated by the local minimum profile drag coefficient of the mid-span section,

$$C_{DPmin0} = [C_{dpmin}]_{\eta=0.5}. \tag{E.4}$$

Alternative forms of Equation E.2 are presented in Appendix A of ESDU [2008b] which use spanwise aerofoil sections normal to the quarter chord line of the wing. The equation for minimum local profile drag will therefore become either,

$$C_{dpmin} = [(C_{dmin})_n - (C_{df0})_n] \cos^3 \Lambda_{1/4} + (C_{df0})_n \cos^{0.2} \Lambda_{1/4}, \tag{E.5}$$

or,

$$C_{dpmin} = [(C_{dmin})_n - (C_{df0})_n] \cos^3 \Lambda_{1/4} + (C_{df0})_s, \tag{E.6}$$

with the subscripts n and s defining the normal or streamwise section respectively.

E.2.3 Increment in Minimum Profile Drag Due to Twist

ESDU [2008b] demonstrated that twisting a wing increases the minimum profile drag coefficient above that of a similar untwisted wing. This increment in profile drag due to wing twist, $(\Delta C_{DPmin})_\varepsilon$, was found to be dependent on the local angle of twist relative to wing setting angle at two spanwise locations, the maximum local twist angle, taper ratio, aspect ratio, section thickness and Reynolds number. Given a wing with a relatively simple twist, the increment in profile drag coefficient due to twist is,

$$(\Delta C_{DPmin})_\varepsilon = 1.2 \times 10^{-5} k_G k_{R_{\bar{c}}} \eta_{|\varepsilon|_m} |\varepsilon|_m \left[\frac{1 - \eta_{|\varepsilon|_m}(1-\lambda)}{(1+\lambda)^2} \right] (|\varepsilon|_{0.2} + |\varepsilon|_{0.8}), \qquad \text{(E.7)}$$

where $|\varepsilon|_m$ is the magnitude of the maximum relative twist in degrees and $\eta_{|\varepsilon|_m}$ is the spanwise position of this twist. $|\varepsilon|_{0.2}$ and $|\varepsilon|_{0.8}$ refer to the magnitudes of the relative twist at 20% and 80% span respectively.

The factor $k_{R_{\bar{c}}}$ accounts for the effect of Reynolds number based on mean aerodynamic chord and is calculated by,

$$k_{R_{\bar{c}}} = 6.62 + 0.0743(\log R_{\bar{c}})^2 - 1.36 \log R_{\bar{c}}. \qquad \text{(E.8)}$$

The factor k_G accounts for the combined effects of aspect ratio and the thickness of the aerofoil section at $\eta = 0.5$. This relationship is derived in ESDU [2008b] and can be calculated using Figure 1 of ESDU [2008a].

E.2.4 Lift-Dependent Viscous Drag Factor

ESDU [2008b] also derives an equation relating the wing lift dependent viscous drag factor to the viscous lift curve slope of the wing, a_{1w}, and the ratio of viscous to inviscid lift-curve slopes, $\frac{a_0}{a_{0T}}$, of an aerofoil section at a spanwise reference station,

$$K_{visc} = \frac{1.15}{a_{1w}} \left[1 - \left(\frac{a_0}{a_{0T}} \right)_{\eta_{ref}} \right]. \qquad \text{(E.9)}$$

The location of the spanwise reference station can be calculated from the wings taper ratio, aspect ratio and quarter chord sweep from the figures presented in ESDU [2008a]. Alternatively ESDU [2008b] states that η_{ref} corresponds to the spanwise station at which the local lift coefficient equals the total lift coefficient for the whole wing. As FP provides a spanwise lift distribution this can be used to calculate η_{ref}, this technique was actually employed in the derivation of the figures presented in ESDU [2008a].

The lift-curve slope of the wing in inviscid flow was demonstrated in ESDU [2008b] to be reasonably similar to that of the wing in viscous flow, hence the inviscid lift curve slope can be employed in Equation E.9. This can be calculated using the figures or software of ESDU [1996a].

The ratio of viscous to inviscid lift-curve slope for the aerofoil section at η_{ref} can be calculated directly using the software package of ESDU [2006d] or through a combination of this package and VGK.

E.2.5 Lift Coefficient for Minimum Viscous Drag

ESDU [2008b] demonstrates that for either a twisted or untwisted wing of varying spanwise camber the lift coefficient for minimum viscous drag is,

$$C_{Lmin} = \frac{-1.21 a_{1w} (\alpha_{0vw})_c}{1 - \left(\frac{a_{0T}}{a_0}\right)_{\eta_{ref}} \frac{a_{1w}}{\pi A_r}}. \tag{E.10}$$

The lift-curve slope of the wing, a_{1w}, and ratio of viscous to inviscid lift curve slope of the η_{ref} aerofoil section are known from the calculation of the lift dependent drag factor. While the aspect ratio, A_r, is also known.

The component of zero-lift angle of attack for the wing in viscous flow due to camber, $(\alpha_{0vw})_c$, however, is yet to be determined. This parameter requires the determination of the spanwise variation of local zero-lift angle of attack for the streamwise aerofoil sections in viscous flow, α_{0v}. As shown in Appendix A of ESDU [2008a], these angles can be combined with the known spanwise lift distribution, overall lift coefficient of the wing and the known wing geometry to predict a wing's zero-lift angle of attack,

$$(\alpha_{0vw})_c = \int_0^1 \left(\frac{C_{l_\eta}}{C_L} \frac{c_\eta}{\bar{c}}\right) \alpha_{0v} \, \mathrm{d}\eta. \tag{E.11}$$

The zero-lift angle of each spanwise aerofoil section can be found using the software package of ESDU [2006d] or once again via VGK simulations of the aerofoil. For the interested reader, ESDU [2008a] contains a number of worked examples demonstrating the calculation of the viscous drag coefficient for a series of wings.

E.3 FP for Matlab

While the FP wing analysis package is undoubtably a useful tool, there exists no batch mode which can be used to analyse a series of wings of varying geometry. Likewise the above procedure for calculating a prediction of the viscous drag coefficient is useful and validated in ESDU [2008a] at low speed, but it requires manual calculation of the coefficients. To this end a series of Matlab functions were developed which construct the necessary input files for FP, run the FP post-processor and parse the output data.

Another function was developed which calculates a prediction of the viscous drag coefficient. This function incorporates a number of modifications to the simplified method presented above and in ESDU [2008a]. The spanwise integral, Equation E.3, is used to calculate C_{DPmin0} rather than the simplification of Equation E.4. This integral can be based on the local minimum profile drag of the streamwise aerofoil sections or of sections normal to the wing quarter chord line, Equations E.2 and E.5 respectively.

Instead of calculating the minimum profile drag via the software package of ESDU [2006d], the Matlab function utilises a series of VGK simulations. The software package of ESDU [2006d] is the combination of a series of previous investigations into subsonic aerofoil performance prediction, notably ESDU [1997, 1998, 2006b,c], utilising VGK simulations of a series of existing aerofoils. Using actual VGK simulations in the wing drag prediction function however, provides an improved capability for dealing with the more unusual aerofoil geometries encountered over the course of an aerodynamic optimisation. The software package of ESDU [2006d] is also restricted somewhat beyond Mach 0.5-0.6. VGK simulations relax this restriction, providing data up to approximately Mach 0.725. The VGK simulations also provide the viscous lift curve slope of the aerofoil and the zero-lift angle of incidence, which are necessary for the calculation of K_{visc} and C_{Lmin}.

The factor K_G required in the calculation of the increment in minimum profile drag due to twist is obtained via a curve fits to the data presented in Figure 1 of ESDU [2008a]. Likewise the inviscid lift curve slope of the wing, a_{1w} required in the calculation of both K_{visc} and C_{Lmin}, is calculated via a series of curve fits to the data presented in Figure 1 of ESDU [1996a]. An ESDU software package is available which contains these figures and calculates the lift curve slope of a wing, however this was found to be somewhat unreliable. Using the raw data within Matlab was found to give equivalent results but neglected some of the additional forward sweep data embedded within the software package.

Other required information, such as the wing geometry, the spanwise loading and the aerodynamic coefficients are provided by the preceding FP simulations.

E.4 FP Validation Cases

Three validation cases are now presented for FP in combination with the viscous drag prediction. Both subsonic and transonic cases are considered. An overview of the wing geometries and flow conditions for all three cases are presented in Table E.1. To aid in the visualisation of each test case the corresponding planforms are presented in Figure E.2.

The first validation case is taken from Brebner and Wyatt [1961] and consists of a highly swept wing with a taper ratio of 1 and a simple RAE 101 aerofoil section. This is a low speed case, with a freestream Mach number of 0.2, for which an experimental drag polar is available. The second case is the classic transonic, low aspect ratio ONERA-M6 wing, taken from Schmitt and Charpin [1979]. This wing is a standard test case for

	Brebner Wing	ONERA-M6	Lockheed Wing "A"
Aspect Ratio	5	3.8	8
Wing Area (m^2)	1.29	0.753	0.053
$\Lambda_{1/4}$	45°	23.3°	25°
Taper Ratio	1	0.562	0.4
Washout	0°	0°	4.8°
Dihedral	0°	0°	0°
Aerofoil Section	RAE 101 (12%)	ONERA-D	-
Mach No.	0.2	0.84	0.8
Reynolds No.	2.1×10^6	14.8×10^6	8.22×10^6

Table E.1: Description of the low speed and transonic wing validation cases.

the validation of pressure profiles at transonic conditions. The third and final case is the Lockheed wing "A" taken from Burdges and Hinson [1984]. This test case is more representative of the wing of a modern transonic transport jet. Both sectional pressure distributions and drag polars are available for this wing from the literature.

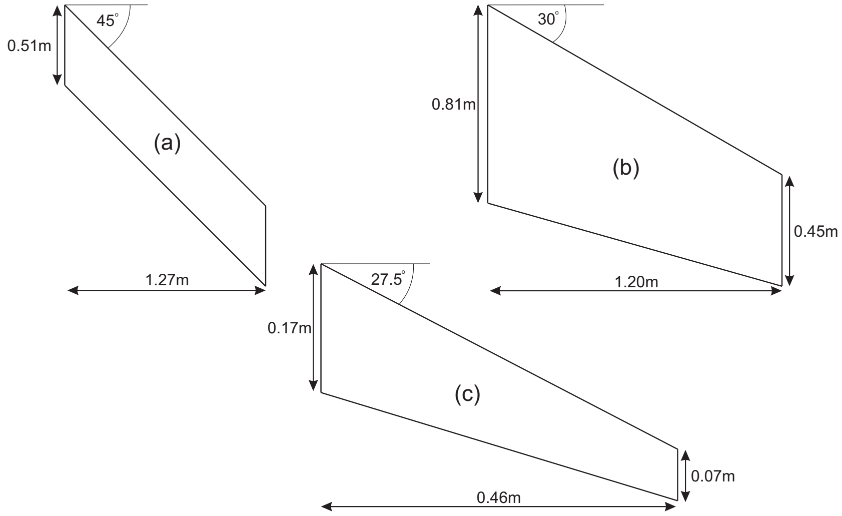


Figure E.1: Planform and dimensions of the (a) Brebner wing, (b) ONERA-M6 and (c) Lockheed Wing "A".

E.5 Brebner Wing Validation

The Brebner low speed wing case, shown in Figure E.1, was modelled in FP through the definition of the root and tip aerofoil sections. The wing was simulated at a series of angles of attack and the lift and vortex drag recorded. The viscous drag correction described previously was then used to calculate the coefficients of the viscous drag polar. These coefficients were then used to predict the viscous drag at each lift coefficient.

A total of 10 streamwise aerofoil sections were used in the calculation of the viscous drag polar parameters via the integral method described previously. Each of the VGK simulations assumed that transition of the boundary layer occurred at 1% chord, with the root aerofoil section having a Reynolds number of 2.1×10^6.

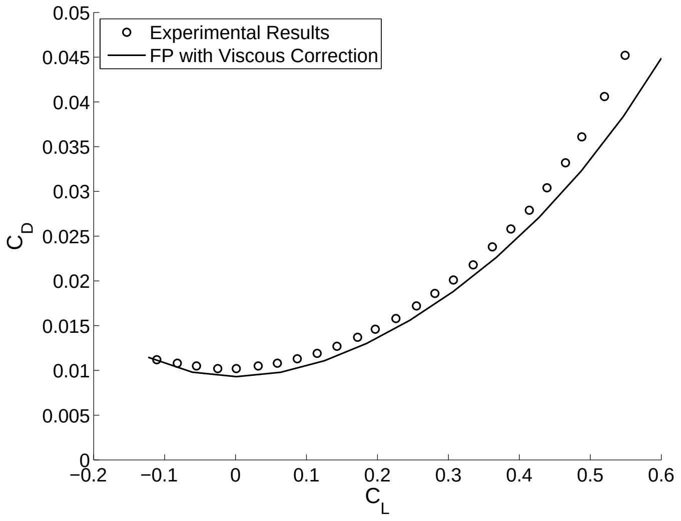


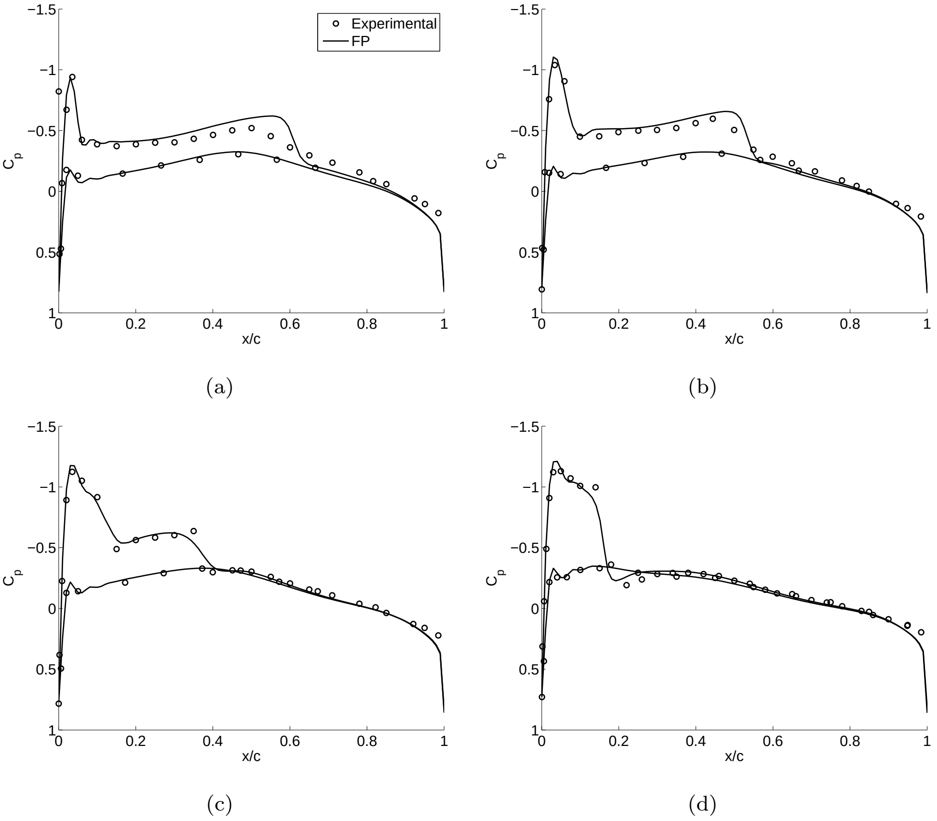
FIGURE E.2: Comparison of experimental drag polar and those predicted by FP with the viscous correction for the Brebner wing.

Figure E.2 compares the experimental drag polar of Brebner and Wyatt [1961] with that predicted by a combination of FP and the vicous drag correction. One can observe that there is a close correlation between the two drag polars. The lift-dependent viscous drag factor, K_{visc}, appears to correspond closely with the experimental results, given that the curvature of the two drag polars closely matches. The lift coefficient for minimum drag, C_{Lmin}, is also close to that of the experimental with both being approximately equal to zero. There is however a slight error in the minimum wing profile drag coefficient, C_{DPmin}. The viscous correction predicts a value of C_{DPmin} equal to 0.093 while the experimental results appear to suggest a minimum profile drag of 0.102. An error of 9 drag counts is reasonably close to that obtained in the validation of the viscous drag correction in ESDU [2008a].

Unsurprising the correlation in drag coefficients drops of at higher lift coefficients when the effects of separation come into play. These effects are not taken into account by either FP or the viscous drag prediction.

E.6 ONERA-M6 Validation

The ONERA-M6 wing was analysed by FP at a Mach number of 0.84 and at an angle of attack of 2°. The resulting pressure distributions at a series of spanwise sections are compared to the experimental results of Schmitt and Charpin [1979] in Figure E.3.



(a)

(b)

(c)

(d)

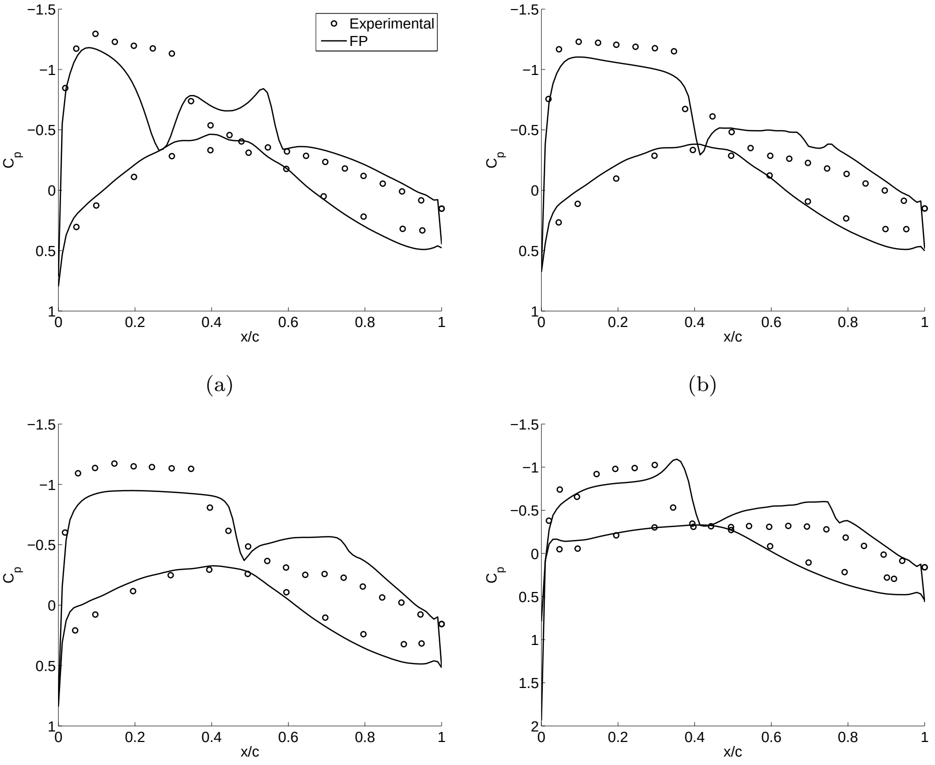
FIGURE E.3: Comparison of experimental pressure distributions with those predicted by FP at four spanwise sections of the ONERA-M6 wing, (a) $\eta = 0.2$, (b) $\eta = 0.44$, (c) $\eta = 0.8$ and (d) $\eta = 0.95$.

One can observe from the comparisons of pressure distributions that FP is clearly capable of capturing major flow features to a reasonable degree of accuracy. Along the span of the wing FP accurately predicts the leading edge suction peak. The location and strength of any shocks present also appear to be accurately captured towards the wing tip. Some errors are present however, closer to the wing root where FP appears to predict a sharper pressure change close to the mid-chord point than is displayed by the experimental results. The predicted lower surface pressures match those of the experimental results almost perfectly, which is unsurprising given that there are no major pressure discontinuities present.

E.7 Lockheed Wing “A” Validation

The third and final validation case is that of the Lockheed Wing “A” taken from Burdges and Hinson [1984]. Unlike either the Brebner wing or the ONERA-M6, this wing has not been previously used in the validation of either the viscous drag correction or FP. Both pressure profiles and drag polars are presented in the literature for a series of freestream Mach numbers.

The geometry of Wing “A” is also significantly different from that of the Breber and ONERA-M6, with a high aspect ratio, swept wing defined by two different unsymmetrical root and tip aerofoil sections with a straight-line loft between them. With these features and 4.8°of washout, Lockheed Wing “A” is a more complex geometry and therefore much more indicative of a modern civil aircraft wing.



(a)

(b)

(c)

(d)

FIGURE E.4: Comparison of experimental pressure distributions with those predicted by FP at four spanwise sections of Lockheed Wing “A”, (a) $\eta = 0.3$, (b) $\eta = 0.5$, (c) $\eta = 0.7$ and (d) $\eta = 0.95$.

Figure E.4 compares the pressure distributions predicted by FP to the experimental distributions at four spanwise sections. The wing was analysed at Mach 0.8 with a lift coefficient of 0.536. Unlike the ONERA-M6 there is a noticeable difference in the pressure distributions, particularly towards the wing root. At $\eta = 0.3$ the experimental results exhibit a single well behaved shockwave on the upper surface, while the FP results appear to exhibit a series of large pressure discontinuities across the surface.

At the remaining stations FP appears to predict the approximate location of the shockwave but tends to over predict the shock strength. Unlike the results for the ONERA-M6, FP no longer accurately predicts the leading edge suction. Instead the pressures on the upper surface forward of the shockwave tend to be greater than the experimental with the regions aft of a shock being predicted as lower than the experimental. However FP’s prediction of the lower surface pressure is generally good across the span of the wing.

The calculation of lift via the integration of pressure across the wing therefore results in an identical lift to the experimental. This lift coefficient however is at a significantly lower angle of attack to that of the experiment. FP requires the wing be angled at 0.92° to the freestream while the experimental results are from a wing with an angle of attack 2.94°. Such a large discrepancy is puzzling given that FP predicted the flow over the ONERA-M6 so well at an identical angle of attack to the experiment and at a much higher freestream velocity than this current case.

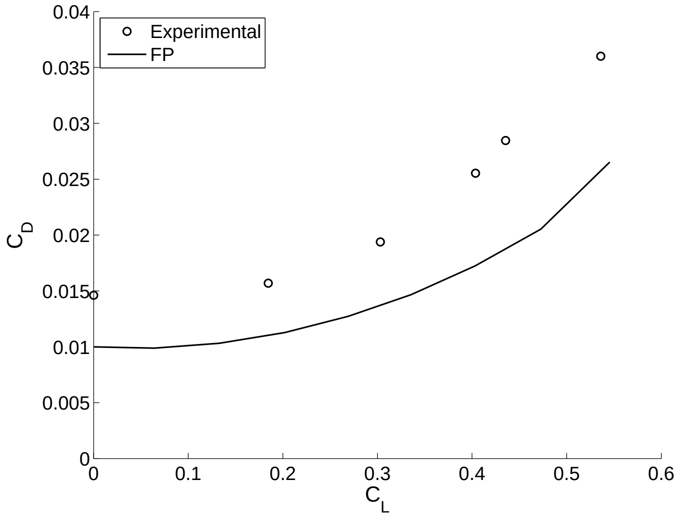


FIGURE E.5: Comparison of experimental drag polar and those predicted by FP with the viscous correction for Lockheed Wing “A”.

A series of simulations of the Lockheed wing “A” were carried out at nine different angles of attack and the corresponding wave and vortex drag coefficients obtained. These results were combined with the minimum profile drag coefficient, increment in profile drag due to twist, lift dependent drag factor and lift coefficient for minimum viscous drag to construct the drag polar presented in Figure E.5. Upon comparison to the experimental results at the same freestream velocity one can observe a severe under prediction of the total drag coefficient by approximately 32% at $C_L = 0$. This is rather unsurprising considering the errors in pressure distribution and the unsuitability of the method employed in the viscous drag prediction to flows in the transonic regime. The combination of FP and viscous correction does however, appear to correctly predict the overall shape of the wing’s drag polar.

E.8 Limitations of FP & Viscous Drag Correction

The validation results presented above indicate that FP is a fast and reasonably reliable method of calculating the inviscid performance of a simple wing at both low and high speed. Although the geometry definition employed by FP is flexible enough to model complex wing geometries with, for example, forward sweep, drooped or upswept wing tips and multiple planform discontinuities the results, especially at high speed, should be considered with care as there are few similar experimental cases for validation.

The viscous prediction is limited much more than FP both geometrically and in terms of the applicability of the method upon which it is based. The method is based purely upon a combination of wing planform, twist and wing sectional data. The accurate prediction of viscous drag of a wing with unusual planforms or wing tips is therefore unlikely. The datasets employed within the prediction of viscous drag also have some restrictions. The determination of a_{1w}, for example, is restricted to wings with a minimum mid-chord sweep of 0°. Both the ratio of viscous to inviscid lift curve slope and the zero-lift angle employed in the calculation of the lift-dependent viscous drag factor and lift coefficient for minimum drag are assumed to be from wing sections in the streamwise direction and not normal to the quarter chord, which is the case if Equation E.5 is employed at higher Mach numbers where VGK may fail at the streamwise velocity.

The method itself was developed purely for wings in shock-free attached flow. Its application to transonic cases, such as those above and in Chapter 8 could therefore be considered questionable. However, one can observe an advantage of this method when one compares the computational expense to that of a RANS simulation involving upwards of 2 million cells and taking several hours on multiple processor cores. Whether employed in conjunction with more expensive simulations or on its own at the conceptual/preliminary design stage, FP in conjunction with the viscous prediction provides a cheap means of assessing a potential wing design, which for simple wings at low speeds has been demonstrated to be quite accurate.

Bibliography

I.H. Abbott and A.E. Von Doenhoff. *Theory of Wing Sections.* Dover Publications Inc., 1960.

S.B. Akat and V. Gazi. Particle swarm optimization with dynamic neighborhood topology: Three neighborhood strategies and preliminary results. In *2008 IEEE Swarm Intelligence Symposium*, St. Louis, MO, USA, 2008.

M.R. Anderberg. *Cluster Analysis for Applications.* Academic Press, 1975.

P. Angeline. Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. In *Proceedings of the 7th International Conference on Evolutionary Programming VII*, pages 601–610, 1998.

P.R. Ashill and J.L. Fulker. *Calculation of the Viscous and Vortex Drag Components of Wing/Body Configurations.* RAE Technical Report TR 87028, 1987.

L. Blasi and G. Del Core. Particle swarm approach in finding optimum aircraft. *Journal of Aircraft*, 44(2):679–682, 2007.

M.I.G. Bloor and M.J. Wilson. Efficient parameterization of generic aircraft geometry. *Journal of Aircraft*, 32(6):1269–1275, 1995.

B. Brandstatter and U. Baumgartner. Particle swarm optimization - mass-spring system analogon. *IEEE Transactions on Magnetics*, 38(2):997–1000, 2002.

D. Bratton and J. Kennedy. Defining a standard for particle swarm optimization, SIS 2007. In *Proceedings of the 2007 IEEE Swarm Intelligence Symposium*, pages 120–127, 2007.

G.C. Brebner and L.A. Wyatt. Boundary layer measurements at low speed on two wings of 45°and 55°sweep. Technical Report ARC-CP-554, Aeronautical Research Council, 1961.

C.G. Broyden. The convergence of a class of double-rank minimization algorithms. *Journal of the Institute of Mathematics and its Applications*, 6(1):76–90, 1970.

T. Bui-Thanh and K. Willcox. Aerodynamic data reconstruction and inverse design using proper orthogonal decomposition. *AIAA Journal*, 42(8):1505–1516, 2004.

K.D. Burdges and B.L. Hinson. Transonic wing and farfield test data on a high aspect ratio transport wing for three dimensional computational method evaluation. Technical Report AGARD-AR-138-ADDENDUM, Lockheed-Georgia Company, 1984.

G.W. Burgreen and O. Baysal. Aerodynamic shape optimization using preconditioned conjugate gradient methods. *AIAA Journal*, 32(11):2145–2152, 1994.

R. Chelouah and P. Siarry. A continuous genetic algorithm designed for the global optimization of multimodal functions. *Journal of Heuristics*, 6(2):191–213, 2000.

R. Chelouah and P. Siarry. Genetic and nelder-meed algorithms hybridized for a more accurate global optimization of continuous multiminima functions. *European Journal of Operational Research*, 148(2):335–348, 2003.

P.H. Cook, M.A. McDonald, and M.C.P. Firmin. *Aerofoil RAE-2822 - Pressure Distributions, and Boundary Layer and Wake Measurements.* AGARD-AR-138, 1979.

S. D'Angelo and E.A. Minisci. Mulit-objective evolutionary optimization of subsonic airfoils by kriging approximation and evolution control. In *2005 IEEE Congress on Evolutionary Computation*, volume 2, pages 1262–1267, 2005.

R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In 6^{th} *International Symposium on Micro Machine and Human Science*, pages 39–43, 1995.

R. Eberhart and Y. Shi. Comparing inertia weights and constriction factors in particle swarm optimization. In *IEEE Conference on Evolutionary Computation*, pages 84–88, 2000.

J. Elliot and J. Peraire. Practical 3D aerodynamic design and optimization using unstructured meshes. In *NASA, and ISSMO, Symposium on Multidisciplinary Analysis and Optimization, 6th, Bellevue, WA, Sept. 4-6*, 1996.

ESDU. *A Method of Determining the Wave Drag and its Spanwise Distribution on a Finite Wing in Transonic Flow.* ESDU-87003, London, 1987.

ESDU. *Lift-Curve Slope and Aerodynamic Centre Position of Wings in Inviscid Subsonic Flow.* ESDU-70011, London, 1996a.

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

ESDU. *Slope of Aerofoil Lift Curve for Subsonic Two-Dimensional Flow.* ESDU-97020, London, 1997.

ESDU. *Aerofoil Incidence for Zero Lift in Subsonic Two-Dimensional Flow.* ESDU-98011, London, 1998.

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. *Full-Potential Method for Three-Dimensional Wings and Wing-Body Combinations - Inviscid Flow Part 4: Evaluation of Trailing-Vortex Drag and Wvae Drag Components.* ESDU-06016, London, 2006a.

ESDU. *Aerofoil Profile Drag for Mach Numbers Below the Drag-Rise Condition.* ESDU-00027, London, 2006b.

ESDU. *Aerofoil Skin Friction Drag for Mach Numbers Below the Drag-Rise Condition.* ESDU-06001, London, 2006c.

ESDU. *Computer Program for Estimation of Aerofoil Characteristics at Subcritical Speeds.* ESDU-06020, London, 2006d.

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

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

S.S. Fan, Y. Liang, and E. Zahara. Hybrid simplex search and part swarm optimization for global optimization of multimodal functions. *Engineering Optimization*, 36(4): 401–418, 2004.

A.I.J. Forrester and A.J. Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1-3):50–79, 2009.

A.I.J. Forrester, N.W. Bressloff, and A.J. Keane. Optimization using surrogate models and partially converged computational fluid dynamics simulations. *Proceedings of the Royal Society A*, 462(2071):2177–2204, 2006a.

A.I.J. Forrester, A.J. Keane, and N.W. Bressloff. Design and analysis of "noisy" computer experiments. *AIAA Journal*, 44(10):2331–2339, 2006b.

A.I.J. Forrester, A. Sóbester, and A.J. Keane. Multi-fidelity optimization via surrogate modelling. *Proceedings of the Royal Society A*, 463(2088):3251–3269, 2007.

M.B. Giles. Collected matrix derivative results for forward and reverse mode algorithmic differentiation. *Lecture Notes in Computational Science and Engineering*, 64:35–44, 2008.

M.B. Giles and N.A. Pierce. An introduction to the adjoint approach to design. *Flow Turbulence and Combustion*, 65(3-4):393–415, 2000.

D.E. Goldberg. *Genetic Algorithms in Search, Optimization & Machine Learning.* Addison-Wesley, 1989.

A. Griewank. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation.* Society for Industrial and Applied Mathematics, 2000.

A. Gross and H.F. Fasel. Control-oriented proper orthogonal decomposition models for unsteady flows. *AIAA Journal*, 45(814-827):4, 2007.

P.K. Gudla and R. Ganguli. An automated hybrid genetic-conjugate gradient algorithm for multimodal optimization problems. *Applied Mathematics and Computers & Geosciences*, 167(2):1457–1474, 2005.

Q. Guo, H. Yu, and A. Xu. A hybrid PSO-GD based intelligent method for machine diagnosis. *Digital Signal Processing*, 16(4):402–418, 2006.

F. Heppner and U. Grenander. *A Stochastic Nonlinear Model for Coordinated Bird Flocks.* AAAS Publications, 1990.

R.M. Hicks and P.A. Henne. Wing design by numerical optimisation. *Journal of Aircraft*, 15(7):407–412, 1978.

J.H. Holland. Outline for a logical theory of adaptive systems. *Journal of the ACM*, 9 (3):297–314, 1962.

P.M. Hollingsworth and D.N. Mavris. Gaussian process meta-modelling: Comparison of gaussian process trainingmethods. In *AIAA* 3^{rd} *Annual Aviation Technology, Integration and Operations (ATIO)*, 2003.

N. Hoyle, N.W. Bressloff, and A.J. Keane. Design optimization of a two-dimensional subsonic engine air intake. *AIAA Journal*, 44(11):2672–2681, 2006.

T. Huang and A.S. Mohan. Significance of neighborhood topologies for the reconstruction of microwave images using particle swarm optimization. In *2005 Asia-Pacific Microwave Conference*, Suzhou, China, 2005.

V.L. Hung and T.L. Hien. Modeling and control of physical processes using proper orthogonal decomposition. *Mathematical and Computer Modelling*, 33(3):223–236, 2001.

L. Huyse, S.L. Padula, R.M. Lewis, and Li. W. Probabilistic approach to free-form airfoil shape optimization under uncertainty. *AIAA Journal*, 40(9):1764–1772, 2002.

K. Izui, S. Nishiwaki, and M. Yoshimura. Swarm algorithms for single- and mulitobjective optimization problems incorporating sensitivity analysis. *Engineering Optimization*, 39(8):981–998, 2007.

Z. Jia, H. Chen, and J. Tang. A new multi-objective fully-informed particle swarm algorithm for flexible job-shop scheduling problems. In *Proceedings of the IEEE International Conference on Micro Electro Mechanical Systems*, pages 191–194, Kobe, Japan, 2007.

R. Jin, W. Chen, and T.W. Simpson. Comparitive studies of metamodelling techniques under multiple modelling criteria. *Structural and Multidisciplinary Optimization*, 23 (1):1–13, 2001.

I.T. Jolliffe. *Principle Component Analysis*. Springer, 2002.

D.R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21(4):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(4):455–492, 1998.

M. Kamali, K. Ponnambalam, and E.D. Soulis. Integration of surrogate optimization and pca for calibration of hydrologic models, a watclass case study. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 2733–2737, 2007.

A.J. Keane. Genetic algorithm optimization of multi-peak problems: Studies in convergence and robustness. *Artificial Intelligence in Engineering*, 9(2):75–83, 1995.

A.J. Keane. *The Options Design Exploration System: Reference Manual and User Guide (Version B3.1)*. http://www.soton.ac.uk/ ajk/options.ps, 2003.

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

A.J. Keane and P.B. Nair. *Computational Approaches for Aerospace Design: The Pursuit of Excellance*. John Wiley & Sons Ltd., 2005.

A.J. Keane and N. Petruzzelli. Aircraft wing design using GA-based multi-level strategies. In 8^{th} *AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimisation*, 2000.

J. Kennedy. Small worlds and mega-minds: Effects of neighborhood topology on particle swarm performance. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99*, pages 1931–1938, 1999.

J. Kennedy and R. Eberhart. Particle swarm optimization. In *IEEE Internation Conference on Neural Networks*, 1995.

J. Kennedy and R. Mendes. Population structure and particle swarm performance. In *Proceedings of the 2002 Congress on Evolutionary Computation*, pages 1671–1676, 2002.

J. Kennedy and R. Mendes. Neighborhood topologies in fully informed and best-of-neighborhood particle swarms. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, 36(4):515–519, 2006.

Y. Kim, D. Rockwell, and A. Liakopoulos. Vortex buffeting of aircraft tail: Interpretation via proper orthogonal decomposition. *AIAA Journal*, 43(3):550–559, 2005.

S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

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

K. Kubota. Matrix inversion algorithms by means of automatic differentiation. *Applied Mathematics Letters*, 7(4):19–22, 1994.

B.M. Kulfan. Universal parametric geometry representation method. *Journal of Aircraft*, 45(1):142–158, 2008.

S.J. Leary, A. Bhaskar, and A.J. Keane. Global approximation and optimisation using adjoint computational fluid dynamics codes. *AIAA Journal*, 42(3):631–641, 2004.

P.A. LeGresley and J.J. Alonso. Airfoil design optimization using reduced order models based on proper orthogonal decomposition. In *Fluids 2000 Conference and Exhibit*, 2000.

P.A. LeGresley and J.J. Alonso. Dynamic domain decomposition and error correction for reduced order models. In *41st AIAA Aerospace Sciences Meeting & Exhibit*, 2003.

W.E. Leithead and Y. Zhang. $O(N^2)$-Operation approximation of covariance matrix inverse in gaussian process regression based on quasi-newton bfgs method. *Communications in Statistics - Simulation and Computation*, 36(2):367–380, 2007.

J. Lépine, F. Guibault, and J. Trépanier. Optimized nonuniform rational B-spline geometrical representation for aerodynamic design of wings. *AIAA Journal*, 39(11): 2011–2041, 2001.

G.C. Lewin and H. Haj-Hariri. Reduced-order modelling of a heaving airfoil. *AIAA Journal*, 43(2):270–283, 2005.

G. Li, M. Li, S. Azarm, J. Rambo, and Y. Joshi. Optimizing thermal design of data centre cabinets with a new multi-objective genetic algorithm. *Distributed and Parallel Databases*, 21(2-3):167–192, 2007.

B. Liu, L. Wang, Y. Jin, F. Tang, and D. Huang. Improved particle swarm optimization combined with chaos. *Chaos, Solitions and Fractals*, 25(5):1261–1271, 2005.

W. Liu, K. Wang, B. Sun, and K. Shao. A hybrid particle swarm optimization algorithm for predicting the chaotic time series. In *Proceedings of the 2006 IEEE International Conference on Mechatronics and Automation*, pages 2452–2458, 2006.

R.C. Lock. *Prediction of the Drag of Wings at Subsonic Speeds by Viscous/Inviscid Interaction Techniques.* Paper 10, AGARD Report 723, 1985.

D.J. Lucia and S. Beran. Reduced-order model development using proper orthogonal decomposition and volterra theory. *AIAA Journal*, 42(6):1181–1190, 2004.

C.A. Mader, J.R.R. Martins, J.J. Alonso, and E. van der Weide. Adjoint: An approach for the rapid development of discrete adjoint solvers. *AIAA Journal*, 46(4):863–873, 2008.

R. Mendes, J. Kennedy, and J. Neves. The fully informed particle swarm: Simpler, maybe better. *IEEE Transactions on Evolutionary Computation*, 8(3):204–210, 2004.

N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6): 1087–1091, 1953.

A.S. Mohais, R. Mendes, C. Ward, and C. Posthoff. Neighborhood re-structuring in particle swarm optimization. In *AI 2005: Advances in Artificial Intelligence, 18th Australian Joint Conference on Artificial Intelligence*, pages 776–785, Sydney, NSW, Australia, 2005.

M. A. Montes De Oca and T. Stutzle. Convergence behavior of the fully informed particle swarm optimization algorithm. In *GECCO'08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, pages 71–78, 2008.

M.D. Morris. Factorial sampling plans for preliminary computational experiments. *Technometrics*, 33(2):161–174, 1991.

M.D. Morris and T.J. Mitchell. Exploratory designs for computational experiments. *Journal of Statistical Planning and Inference*, 43(3):381–402, 1995.

F. Muyl, D. Laurent, and V. Herbert. Hybrid method for aerodynamic shape optimization in automotive industry. *Computers & Fluids*, 33(5-6):849–858, 2004.

D. My-Ha, K.M. Lim, B.C. Khoo, and K. Willcox. Real-time optimisation using proper orthogonal decomposition: Free surface prediction due to underwater bubble dynamics. *Computers & Fluids*, 36(3):499–512, 2007.

H. Ninomiya and Q. Zhang. Particle with ability of local search swarm optimization: PALSO for training of feedforward neural networks. In *2008 IEEE International Joint Conference on Neural Networks*, pages 3009–14, 2008.

J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer, 1999.

M.M. Noewl and T.C. Jannett. Simulation of a new hybrid particle swarm optimization algorithm. In *Proceedings of the* 36^{th} *Southeastern Symposium on System Theory*, pages 150–153, 2004.

S. Obayashi and S. Takanashi. Genetic optimization of target pressure distributions for inverse design methods. *AIAA Journal*, 34(5):881–886, 1996.

S. Painchaud-Ouellet, C. Tribes, J. Trépanier, and D. Pelletier. Airfoil shaped optimization using a nonuniform rational B-spline parameterization under thickness constraint. *AIAA Journal*, 44(10):2170–2178, 2006.

J. Park and J. Baek. Efficient computation of maximum likelihood estimators in a spatial linear model with power exponential covariogram. *Computers & Geosciences*, 27(1): 1–7, 2001.

K.B. Petersen and M.S. Pederson. *The Matrix Cookbook.* 2007.

W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipies: The Art of Scientific Computing.* Cambridge University Press, 1986.

T. Ray and H.M. Tsai. Swarm algorithm for single and mulit-objective airfoil design optimization. *AIAA Journal*, 42(2):366–373, 2004.

G. Redeker and R. Müller. *A Comparison of Experimental Results for the Transonic Flow around the DFVLR-F4 Wing Body Configuration.* GARTEUR TP-018, 1985.

J.J. Reuther, A. Jameson, J.J. Alonso, Rimlinger M.J., and D. Saunders. Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers, part 2. *Journal of Aircraft*, 36(1):61–74, 1999.

J. Riget and M. J. Vesterstrø. A diversity-guided particle swarm optimizer - the ARPSO. Technical report, EVALife Project Group, 2002.

G.M. Robinson and A.J. Keane. Concise orthogonal representation of supercritical aerofoils. *Journal of Aircraft*, 38(3):580–583, 2001.

J. Sacks, W.J. Welch, T.J. Mitchell, and H.P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4(4):409–435, 1989.

S. Sakata, F. Ashida, and M. Zako. Structural optimization using kriging approximation. *Computer Methods in Applied Mechanics and Engineering*, 192(7-9):923–939, 2003.

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 Researches Aerospatiales, 1979.

A. Shahrokhi and A. Jahangirian. Airfoil shape parameterization for optimum navier-stokes design with genetic algorithm. *Aerospace Science and Technology*, 11(6):443–450, 2007.

A. Sherif, O. Madara, and P. Richard. Stochastic approach to optimal aerodynamic design. *Journal of Aircraft*, 33(5):956–961, 1996.

Y. Shi and R. Eberhart. Parameter selection in particle swarm optimization. In *Evolutionary Programming VII*, 1998.

S.F. Shu-Kai, L. Yun-Chia, and E. Zahara. Hybrid simplex search and particle swarm optimization for the global optimization of multimodal functions. *Engineering Optimization*, 36(4):401–418, 2004.

S.G. Siegal, K. Cohen, J. Seidel, and T. McLaughlin. Proper orthogonal decomposition snapshot selection for stat estimation of feedback controlled flows. In *44th AIAA Aerospace Sciences Meeting and Exhibit*, 2006.

T.W. Simpson, J.D. Peplinski, and P.N. Kock. Metamodels for computer-based engineering design: Survey and recommendations. *Engineering with Computers*, 17(2): 129–150, 2001.

L. Sirovich. Turbulence and dynamics of coherent structures part 1: Coherent structures. *Quarterly of Applied Mathematics*, 45(3):561–571, 1987.

S.P. Smith. Differentiation of the cholesky algorithm. *Journal of Computational and Graphical Statistics*, 4(2):134–147, 1995.

A. Sóbester, S.J. Leary, and A.J. Keane. On the design of optimization strategies based on global response surface approximation models. *Journal of Global Optimization*, 33 (1):31–59, 2005.

H. Sobieczky. Parametric airfoils and wings. *Notes on Numerical Fluid Mechanics*, 68: 71–88, 1998.

W. Song and A.J. Keane. A study of shape parameterisation methods for airfoil optimization. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, pages 2031–2038, 2004.

W. Song and A.J. Keane. Surrogate-based aerodynamic shape optimization of a civil aircraft engine nacelle. *AIAA Journal*, 45(10):2562–2574, 2007.

D.J.J. Toal, N.W. Bressloff, and A.J. Keane. Kriging hyperparameter tuning strategies. *AIAA Journal*, 46(5):1240–1252, 2008a.

D.J.J. Toal, N.W. Bressloff, and A.J. Keane. Geometric filtration using POD for aerodynamic design optimization. In 26^{th} *AIAA Applied Aerodynamics Conference*, 2008b.

D.J.J. Toal, N.W. Bressloff, and A.J. Keane. Geometric filtration using POD for aerodynamic design optimisation. *AIAA Journal*, Under Review, 2009a.

D.J.J. Toal, A.I.J. Forrester, N.W. Bressloff, A.J. Keane, and C. Holden. An adjoint for likelihood maximization. *Proceedings of the Royal Society A*, Accepted for Publication, 2009b.

L. Trocine and L.C. Malone. Finding important independent variables through screening designs: A comparison of methods. In *Proceedings of the 2000 Winter Simulation Conference*, pages 749–754, 2000.

G. Venter and J. Sobieszcaznski-Sobieski. Particle swarm optimization. *AIAA Journal*, 41(8):1583–1589, 2003.

T.A.A. Victoire and A.E. Jayakumar. Hybrid PSO-SQP for economic dispatch with valve-point effect. *Electric Power Systems Research*, 71(1):51–59, 2004.

X. Wang and M. Damodaran. Inverse transonic airfoil design using parallel simulated annealing and computational fluid dynamics. *AIAA Journal*, 40(4):791–794, 2002.

Y. Wang, J. Zhang, and Y. Zhang. An effective and efficient two stage algorithm for global optimization. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), v 3930 LNAI, Advances in Machine Learning and Cybernetics - 4^{th} International Conference, ICMLC 2005*, pages 487–496, 2006.

W.J. Welch, R.J. Buck, and J. Sacks. Screening, predicting, and computer experiments. *Technometrics*, 34(1):15–25, 1992.

K.S. Won and T. Ray. Performance of kriging and cokriging based surrogate models within the unified framework for surrogate assisted optimization. *Proceedings of the 2004 Congress on Evolutionary Computation*, 2(2):1577–1585, 2004.

D. Yuret and M. Maza. Dynamic hill climbing: Overcoming the limitations of optimization techniques. In *Proceedings of the 2^{nd} Turkish Symposium on AI and ANN*, 1993.

Y. Zhang and W.E. Leithead. Exploiting hessian matrix and trust-region algorithm in hyperparameters estimation of gaussian process. *Applied Mathematics and Computation*, 171(2):1264–1281, 2005.