University of Exeter coursework header sheet

ECMM102 Group Project (Meng) (A, TRM1+2 2016/7)

014768

1027442

620009941

Coursework: Individual contribution to the group achievement
Submission Deadline: Thu 4th May 2017 12:00
Personal tutor: Dr Allen He

Marker name: G_Tabor
Word count: 15753

By submitting coursework you declare that you understand and consent to the University policies regarding plagiarism and mitigation (these can be seen online at www.exeter.ac.uk/plagiarism, and www.exeter.ac.uk/mitigation respectively), and that you have read your school's rules for submission of written coursework, for example rules on maximum and minimum number of words. Indicative/first marks are provisional only.

First marker's comments

Indicative mark

Second marker's comments

Second mark

Moderator's comments

Agreed mark

I2 Report

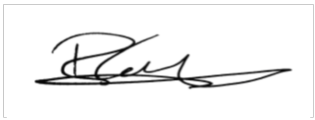
An Investigation into Bayesian Methods for CFD Optimisation

Rhys Gilbert

2017

4th year MEng Group Project

I certify that all material in this thesis that is not my own work has been identified and that no material has been included for which a degree has previously been conferred on me.

Signed 

College of Engineering, Mathematics, and Physical Sciences
University of Exeter

I2 Report

ECMM102

Title: An Investigation into Bayesian Methods for CFD Optimisation
Word count: 15753
Number of pages: 40

Date of submission: Monday, 01 May 2017

Student Name: Rhys Gilbert
Programme: Mechanical Engineering MEng
Student number: 620009941
Candidate number: 014768

Supervisor: Dr Gavin Tabor

Abstract

Optimisation plays an important role within industry. This is due to the requirement that the designs are the most efficient be it in regards to drag, lift or even energy loss from the system as a whole, amongst others.

A range of optimisers are available but the majority require a large number of function calls to the objective function. Therefore, if the objective function is a Computational Fluid Dynamics (CFD) case where it is not uncommon for the solution to take multiple days, even weeks, it is impractical to utilise many of these methods. This is where the Bayesian methodology is introduced which generates a surrogate model for the objective function.

CFD optimisation was investigated through the optimisation of a draft tube which are required to increase the efficiency of the turbine. The aims of the overall project were to optimise the geometry and the aims for this investigation were to investigate decreasing the number of function calls and time for the optimiser.

It has been shown that Bayesian methods are a vast improvement on the genetic algorithms and can be used to find global optima and not local optima unlike other methods. As this has been shown the investigation led onto the development of reducing the number of function calls. This led onto the use of adjoint CFD solvers and surface sensitivities to generate the gradients of the objective function. Unfortunately, as grasping the mathematics took longer than expected, these methods were not integrated into the Bayesian methods and therefore their benefits and drawbacks could not be compared against the other methods as to number of function calls and time for the optimiser to take place.

Keywords: Optimisation, Bayesian optimisation, adjoint optimisation, CFD

Table of contents

1. Introduction and Background

1.1. Role of Optimisation

Optimisation is generally considered to be the minimisation or maximisation of a function for a given bound. This function can be anything from the maximisation of profit for a product for a given material to the minimisation of drag on an aircraft. There are many ways in which optimisation can be achieved, some include genetic algorithms (GA), which mimic survival of the fittest through natural selection. Other types of optimisation are gradient descent methods which are used to find local minima given gradients of the function.

These optimisation problems can be either single objective, meaning one function is being optimised or multi objective, meaning there are many functions being optimised. An example of a multi objective problem is the optimisation of an aircraft to reduce drag but increase lift for a given configuration.

Optimisation is a very useful and powerful tool as it can decrease expenditure on manually searching the search space. Optimisation can be automated which means that the designer can generate a base case and optimise this automatically leaving them to be free to be able to carry out other tasks. Optimisers are also extremely useful for optimising problems with many input dimensions, where the trade-off between inputs may not be clearly visible to the designer through trial and error.

1.2. Computational Fluid Dynamics

Computational Fluid Dynamics (CFD) is the solution of the Navier-Stokes equations [1] on arbitary geometry. The process used is usually an iterative process and as such requires a large amount of computing power and time to solve detailed cases.

CFD is time consuming to optimise as the associated time cost for evaluation of the solution limits the number of evaluations that can be undertaken. This limit on the number of evaluations caps the amount of exploration within the search space that can be achieved and as such the optimisation may not find the global optima of the problem. Therefore there is a clear need for an efficient optimiser that reduces the number of queries to the function being evaluated.

This optimisation problem has been cast as a draft tube optimisation, but could be applied to any case. Draft tubes are located below turbines in hydro-eclectic power generation dams. Their role is to minimise the total amount of kinetic energy loss, which relates to a greater pressure head on the turbine, thus generating greater power from the fluid. Therefore there is a benefit for the draft tube to be an optimum design to increase the power generated.

1.3. Bayesian Optimisation

Bayesian optimisation utilises a Gaussian process regression model for generating an approximation for the function from a collection of known values. From this the approximation function can be queried for the location in search space that will benefit the accuracy of the model most, this then is computed using the actual function being optimised and then the Gaussian process regression model is updated and the process is repeated.

This process will be beneficial for the optimisation of CFD as it reduces the number of function calls, which in this case will be time consuming. Therefore, a larger range of designs can be investigated and it is more likely that the optimiser will find the global optima.

1.4. Aims

The overall aim of this group project is to develop a Bayesian optimiser that can be used efficiently to optimise the 3D geometry of a draft tube to maximise the pressure difference between inlet and outlet. This method should also be quicker than any other method available and subsequently used in preference.

However, the aim of the project undertaken is to develop a methodology which reduces the total number of function evaluations. This method should be able to find the optimum with a reduced number of function calls in a shorter amount of time.

1.5. Objectives

In order to achieve the aim several smaller objectives have been assigned. Initially the main objective is for the mathematical concepts for optimisation using Bayesian methods to be understood. Following this the objective will be to develop a Bayesian optimiser for a simple 1D function, such as x^2. Leading to the next objective which will be to test the optimiser against other optimisers and compare accuracy and efficiency. Another objective following this is to increase the number of dimensions of the problem.

To be able to reduce the number of function calls investigations into extracting the gradient information will be undertaken. These will be tested on 2D CFD cases to show that they are achievable and then the methods developed will be applied to the full 3D draft tube geometry.

1.6. Report Structure

Following this section there will be a section on the current work related to the project, with the literature being reviewed. This section will be split between the work on optimiser methodology along with the current work on optimising draft tubes which will be the final aim of the project.

Successive sections will explain the methodology and mathematics of the optimisers that will be generated for eventually optimising the draft tube.

The results that have been generated from these optimisers will be presented in the following section. This will display the development process undertaken in the creation of the optimisers.

Ultimately there will be a discussion about the results generated for the optimisers written. Here the results will be described in detail and will also explain why certain optimisers were chosen. Also contained within this section will be a discussion on future directions of the project.

A section on project management will follow. Here the managerial details will be explained and discussed that had been utilised throughout the project's duration.

Finally, there will be a section on how this current work relates and interacts with the work undertaken by other members in the group.

2. Literature Review

2.1. Optimisation

There are many types of optimisation that could be utilised to optimise a function. Some possible methods include GAs or Bayesian optimisation (BO). There are other optimisation methods such as adjoint methods, however they require that the function being optimised is known and they also only find local minima, whereas GA and BO methods do not.

Genetic Algorithms

There has been a large body of work conducted in the development of genetic algorithms. The method was first investigated in the late 1950's and throughout the 1960's by various researchers. The first paper on nature inspired computation was published in 1963 [2]. This paper introduces the concept of evolutionary algorithms such as GAs. This paper gives an in-depth account of how to apply GAs to various problems. This paper is abstract enough that the concepts for a simple problem can be transferred onto more challenging problems.

There is an issue raised by this paper in that there are a large choice of parameters for the algorithm. This paper does not describe the best choice for these as they are determined on a per problem basis. The paper does not give general starting points for these parameters based on rules of thumb, and therefore the reader is left plucking parameter values out of thin air. Further to this the paper is abstract and does not show the full potential of the algorithm for industrial application.

One other contribution to the field of GAs is [3]. This book assumes that the reader has no experience in the field and builds up complexity and ideas until a very in depth description of the algorithm is explained. It delves into possible industrial application of the algorithm but only for simple cases and not complex industrially relevant cases.

[3] builds on [2] as it attempts to give starting points and explains why the values of parameters are chosen for specific cases. [3] also makes it clear how GAs can be applied to a range of

problems. [3], however, does not show how to apply GAs to all types of problems and therefore there are many methods that could be utilised to apply GAs to the use in CFD.

[4] develops [3] by applying parallel GAs to optimisation of a heat exchanger. This would be similar to the work that would be conducted in this project. [4] does indeed show one possible method that GAs could be applied to CFD optimisation.

This paper does extend the work of [2] and [3] by introducing an industrially relevant aspect to the previous work. However, [4] shows the downside of the algorithm. This being that there is a large computational cost to undertake the algorithm, around 10 minutes in parallel, even for a simple 2D heat exchanger. This is due to the vast amount of function calls, in this case CFD runs, required by the algorithm. Therefore, for the algorithm to work on a 3D CFD case that requires a couple of days to run would require a vast amount of time that would not be appealing to industry. It is clear from [4] that another method is required for the optimisation of CFD.

All of the papers evaluated treat the function they are optimising as a black box. In other words, they do not require the structure of the objective functions to be known. All they require is the output for a given input. This is beneficial as the algorithms can remain abstract and will function for any objective function that is required to be optimised.

Bayesian Optimisation

Without Gradient Information

BO was first introduced by Jonas Mockus in the 1970s and 1980s. [5] initially develops the idea of utilising Gaussian process methods for optimising a function. It is clear from this that there is a major benefit from this method above that of GAs as BO requires a reduced number of function evaluations, as the method produces a surrogate model and queries this as to the best place to evaluate the solution to benefit the surrogate model.

[5] also shows how the method can be applied to multi-dimensional problems as it is formulated with vectors of inputs in mind. Therefore, this presentation of the method easily extends to arbitrary dimensional problems. This paper also shows numerical applications of the problems by optimising a range of various functions.

One major drawback of the method is that it is unclear how this method could be applied to multi-objective problems, ones with many objective functions. Another drawback is that multi-modal functions, ones with many optima, are not described how the Bayesian method would optimise these.

Further to this [5] does not inform the reader of the best choice for the methods hyperparameters. One other drawback is that the process is described but does not show a visual representation of what is happening in the algorithms. This may be required by some readers to be able to understand the process.

[6] furthers the work of that in [5] as it shows a graphical example of what the algorithm is undertaking. Additionally [6] delves into a range of possible methods that could be utilised to represent the underlying function, called kernels. This shows that there are a range of different representations of the function being optimised. The drawback of this is that the paper does not go on to mention which kernel functions are the most adequate for the problem type.

On top of this [6] explains a range of different methods for computing the location which will best improve the surrogate model. Again, the details of which is the most adequate for each problem type is unclear.

[7] furthers the work of [5] and [6] by giving pseudo code of the algorithms utilised in Bayesian optimisation. This is beneficial as it provides further understanding of what the algorithm is undertaking and also gives a possible method of implementation that the reader could utilise. This also encourages the reader to further their understanding of the process by implementing the method, testing and investigating what impact altering the parameters has on the result.

This paper also delves into possible methods for optimising the acquisition function which is the function that is used to provide the location to evaluate the function that will give the most beneficial results. This is not described in [6]. Again, as with previous papers [7] does not show how a multi-objective optimisation using a Bayesian approach could be undertaken.

[8] develops on [7] by the introduction of an acquisition function that enables fewer function evaluations than other methods. This is beneficial to the optimisation of CFD as it reduces the number of time consuming evaluations. For this reason, the method is called efficient global optimisation (EGO).

This paper is related to the electronics industry; however, it is clear how the method could be applied to any arbitrary industrial application. One other benefit of this paper is that it shows how accurate the model and methodologies are for such few function evaluations. This is shown through the comparison of the true function against the surrogate model, where there is hardly any difference between them. These are also compared against other methods, which show a large difference and inability to predict optima. [8] also does not show how to develop Bayesian methodologies for multi-objective optimisation.

From the papers researched there is a distinct lack of application to the optimisation of CFD using Bayesian methods. This may be due to a range of reasons including how to represent the model in a format that the Gaussian process can achieve. One other possible reason why these methods have not been implemented with CFD is how to manipulate the meshes for the range of design changes.

With Gradient Information

[9] reports that there is some benefit in introducing gradient information of the function into the Gaussian process model. It is clear that this is of major benefit over the standard Bayesian method. This method is also shown to be able to be capable of modelling multi-dimensional problems.

One major issue with [9] is that it does not show the full implementation of the method. This is a major disadvantage in that the reader may wish to investigate the impacts of gradient information but may be put off by the lack of clear detail in the section. The lack of information is clear as the method of incorporating the gradients is only referenced in the further issues section.

[10] develops on [9] by introducing more detail about the method. Further to this [10] clearly shows graphically, albeit in 1D, why the introduction of gradient information would be beneficial to the model. This makes the method more tractable in that readers can see visually what the algorithms are undertaking and therefore makes the description and mathematics easier to follow.

One drawback of [10] is that there is lack of information about how the method can be increased to be able to deal with increased dimensions of the problem. Further to this some components of the report are difficult to follow and would be improved if there was a simple example to follow. It appears that the authors assume a certain level of knowledge, which again may discourage some readers from investigating the method.

[11] differs from [9] and [10] as the paper does outline how the method is implemented and also investigates different acquisition functions that utilise the gradient information to provide a location which will improve the surrogate model the most.

[11] outlines various methods for the acquisition function of the Bayesian method. This is followed by a detailed account of the benefits and drawbacks of each method whilst utilising the gradient information that has been generated.

It is slightly clearer to follow the methodology in [11] on the implementation of the method, however this paper assumes a high level of knowledge of Gaussian processes and Bayesian methods. This again may deter readers from fully investigating the method.

2.2. Draft Tube Optimisation

From an efficiency perspective there is a great need for the optimisation of draft tubes and some investigations have been attempted. [12] optimised a draft tube using an evolutionary algorithm approach. This was achieved through the parameterisation of the geometry and the algorithm altering these parameters. The function that was being optimised was that of the pressure difference between the inlet and outlet.

One drawback stated is the large cost of evaluating the CFD for each and every geometry in the population. [12] makes around 250 separate CFD evaluations. It is unclear from this if the optima that has been found for this geometry is indeed the global minima or if it is a local minimum. This is due to the design changing very little to that of the initial geometry.

[13] has also used a parameterisation of the draft tube to perform the optimisation. The method used is to take cross sectional areas of the geometry and to use these as the basis for altering the geometry. These cross sections are then lofted to create the final shape of the draft tube.

The function for optimisation used is to minimise the total head loss and swirl of the overall system.

In [13] the authors utilise a weighted sum of objective functions. While this is useful for simply optimising a multi-objective problem, as it merges the objective functions into one function, a number of major drawbacks appear with this method. One being that the weighting values for each of the objective functions must be known prior to the optimisation. These weightings are very difficult to obtain and require deep knowledge of the entire system.

One other drawback of this method is that the interaction and relationship between the objective functions cannot be evaluated. Therefore it is unclear why the optimisation took a certain optimisation path. Further to this the method of weighted sums casts all objective functions into a similar form, i.e. all in minimisation or maximisation, which gives the functions a non-real aspect to them.

What is not stated is how many iterations of the geometry the process was run for. This could be a problem for industrial application as this method could take a large amount of time to generate the solution.

One benefit of the parameterisation method is that virtually any geometry can be defined in this way and, as such, the method can be utilised for a range of applications such as internal and external flows.

[14] utilises splines to define the draft tube geometry. This is beneficial as splines are simple and will always be smooth. This is an alternative approach to those of [12] and [13] because the splines are altered rather than the cross-sectional areas of the geometry.

However, one drawback of the optimisation method used is that the spline parameters were altered sequentially. This basically ensured a brute force method for the optimisation of the draft tube. This is clearly not efficient in that some considerable time would be required to generate all the solutions for each design change. It is also unclear if this method truly finds the global minima of the process.

There are other optimisation methods in CFD, notably the adjoint method [15], [16], [17], [18] and [19] which have been applied to aerospace, automotive, internal duct and multiphase flows. However, it appears that they have not been applied to draft tube optimisation. This could be one possible avenue of further research. However, as explained in [15], adjoint methods are only applicable if the design variables that are being investigated, surfaces for example, are continuous.

One other drawback, as explained in [15], is that as the adjoint method is a gradient based method and therefore will only find the minima closest to the initial design. This means that the adjoint probably would not find the global minima of the function. For the method to find the global minima, the initial design must be fairly close to the global minima. Therefore, the Bayesian method is more likely to find the global minima as it does not search in the negative gradient direction.

The major drawback of the method is that it cannot be applied to functions that are treated as black boxes. The adjoint method requires mathematical manipulation of the equations and therefore the function must be known a priori.

3. Theoretical Background

3.1. Python and Software Development Theory

As the model for the optimiser should be able to tackle many different problem functions it must be written in such a way that allows arbitrary functions, that require optimisation, to be given to the optimiser without the need for specialisation of the code.

To achieve such code python allows the parsing of functions to other functions and to classes to be called upon at a later date when the function is needed. This is beneficial as it allows the code to be written in an abstract manner.

Further to this the code should be written utilising polymorphism. Polymorphism states that classes can inherit from others, i.e. an abstract base class should have a given structure, which all classes that inherit from the base class should follow, but re-implement the functions.

All this the code should also be commented and tested. The main benefit for testing the code is that there will be other utilising the code written and therefore they should be able to understand what is happening if they are required to extend the libraries. This also cut down on the errors within the code.

3.2. Genetic Algorithm

The basic idea of a genetic algorithm from [3] is that an initial population is generated. The initial population must be bound within a range of values, the search space for the algorithm. Once this population has been randomly assigned values they are then evaluated by giving the values to the function being optimised, here called the fitness function.

Once the population have been evaluated, parent selection takes place. Parent selection is a method chosen to select the parents that will be used in generation of children. There are many methods for this such as selecting the best two, or creating a tournament where a random selection from the population are selected and the best two individuals are chosen out of this. A full breakdown of possible methods is not given here.

Once the parents have been selected the children are produced with a crossover between the parents. There are many methods to produce the crossover but the method chosen is that a percentage of each parent produces the children. This method allows exploration over the search space.

Once the children have been produced their values are mutated and again there are various methods used in mutation. The basic mutation is that a small random value is added to the

children. Following this the children have their fitness value evaluated, using the fitness function, and are added to the population. Finally the worst individuals from the population are removed.

3.3. Gaussian Process

A Gaussian process is one application of machine learning for regression, meaning finding a relationship between input and output. The main idea is to fit a surrogate model, an approximation for the function through the known points, that has a mean and a standard deviation.

Assuming we have a function of the following type, $y = f(\boldsymbol{x})$ that requires to be optimised. This function can be any function. Assuming that we have a known dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) | i = 1 \dots N\}$ meaning that for a given input $\boldsymbol{x}_i$ the function is evaluated giving y_i.

The surrogate function is completely described using the mean, $m(\boldsymbol{x})$, and covariance, $k(\boldsymbol{x}, \boldsymbol{x}')$ of the original function, shown in equations 1 [9] and 2 [9].

$$m(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})] \tag{1}$$

$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}[(f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\boldsymbol{x}') - m(\boldsymbol{x}'))] \tag{2}$$

Where $\mathbb{E}$ denotes the expected outcome for the given value. The covariance represents how similar the two $\boldsymbol{x}$ and $\boldsymbol{x}'$ are. If they are similar covariance is strong, otherwise the covariance is weak.

To build the surrogate model the covariance between all known N inputs of the dataset must be generated. This produces a $N \times N$ matrix, called the covariance matrix, shown in equation 3 [9].

$$K = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \tag{3}$$

Then to generate the mean at any given point $\boldsymbol{x}_*$ the covariance matrix between this point and the known points are required, this is attained using equations 4 [9] and 5 [9].

$$K(\boldsymbol{x}, \boldsymbol{x}_*) = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_*) \\ \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_*) \end{bmatrix} \tag{4}$$

$$K(\boldsymbol{x}_*, \boldsymbol{x}) = \begin{bmatrix} k(\boldsymbol{x}_*, \boldsymbol{x}_1) & \dots & k(\boldsymbol{x}_*, \boldsymbol{x}_N) \end{bmatrix} \tag{5}$$

Then finding the covariance between $\boldsymbol{x}_*$ and itself, is shown in equation 6 [9].

$$K(\boldsymbol{x}_*, \boldsymbol{x}_*) = k(\boldsymbol{x}_*, \boldsymbol{x}_*) \tag{6}$$

Finally, the mean, $\bar{y}_*$, and variance, $var(\bar{y}_*)$, for a given point, $\boldsymbol{x}_*$, can be calculated using equations 7 [20] and 8 [20] respectively.

$$\bar{y}_* = K(\boldsymbol{x}_*, \boldsymbol{x})K^{-1}\boldsymbol{y} \tag{7}$$

$$var(\bar{y}_*) = K(\boldsymbol{x}_*, \boldsymbol{x}_*) - K(\boldsymbol{x}_*, \boldsymbol{x})K^{-1}K(\boldsymbol{x}, \boldsymbol{x}_*) \tag{8}$$

Covariance Function

There are many choices for the covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$. One simple choice is the squared exponential covariance function shown in equation 9 [21]. These functions are also referred to as kernels.

$$k(\boldsymbol{x}_p, \boldsymbol{x}_q) = {\sigma_f}^2 \exp\left(0.5(\boldsymbol{x}_p - \boldsymbol{x}_q)M(\boldsymbol{x}_p - \boldsymbol{x}_q)^T\right) + {\sigma_n}^2\delta_{pq} \tag{9}$$

Where σ_f is the variance of the noise free signal, M can take many forms, one being that of equation 10, σ_n is the noise variance. Noise is added to avoid singular K matrices and δ_{pq} is the Kronecker delta which is 1 if $p = q$ and 0 otherwise.

$$M = diag(\boldsymbol{\ell})^{-2} \tag{10}$$

Where $\boldsymbol{\ell}$ is the vector of length scales roughly relating to how far a point must be moved in a direction to become uncorrelated with another, and $diag$ is the diagonal matrix operator. The length of $\boldsymbol{\ell}$ depends on the number of dimensions of the input vector $\boldsymbol{x}$. $\boldsymbol{\ell}$ are known as the hyperparameters and are unknown, therefore they must be learned from the known dataset.

Updating Hyperparameters

In order select the correct hyperparameters for the known data $\mathcal{D}$ the marginal likelihood, z, function must be maximised. This equation relies in the value of the hyperparameters through the matrix K, equations 3 and 9, and by maximising equation 11 [9] will mean that the best solution for hyperparameters are chosen.

$$z = -\frac{1}{2}\boldsymbol{y}^T K^{-1}\boldsymbol{y} - \frac{1}{2}\log(det(K)) - \frac{N}{2}\log(2\pi) \tag{11}$$

This can be optimised using a variety of techniques, here it has been optimised utilising the GA written previously.

3.4. Bayesian Optimiser

Bayesian optimisation is a relatively simple concept. The main idea is to query the Gaussian process to find the location of the next best sample point. This is done through optimising the acquisition function and finding its maximum. There are many acquisition functions such as optimistic, information and improvement based methods [7]. The method chosen for this purpose is the expected improvement method of [8]. The main reason for this is that it selects points based on the improvement that they will give to the Gaussian process and therefore

should reduce the total number of function evaluations. One of the main functions of the acquisition function is to not only explore the search space but also exploit known good solutions. Its form is given in equation 12 [8] for minimisation problems.

$$\mathbb{E}[I(\boldsymbol{x})] = (f_{min} - \bar{y}_*)\Phi\left(\frac{f_{min}-\bar{y}_*}{s}\right) + s\phi\left(\frac{f_{min}-\bar{y}_*}{s}\right) \tag{12}$$

Where f_{min} is the lowest known minimum evaluation of $f(\boldsymbol{x})$, $\bar{y}_*$ is evaluated from equation 7, Φ is the standard normal distribution function, ϕ is the standard normal density function and s is the square root of equation 8. Again, as with the marginal likelihood, the expected improvement can be optimised to find the maximum with any optimiser, here the GA has been utilised.

The standard normal distribution function is calculated using equation 13 [22].

$$\Phi(x) = \frac{1}{2}\left[\frac{1}{(1+a_1x+a_2x^2+a_3x^3+a_4x^4)}\right] \tag{13}$$

Where equation 13 has a maximum error of $5\text{x}10^{-4}$ [22] and assumes that the distribution has a mean of 0 and standard deviation of 1. Where a_1 is 0.278393, a_2 is 0.230389, a_3 is 0.000972 and a_4 is 0.078108.

The standard normal density function is calculated using equation 14 [22].

$$\phi(x) = \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}} \tag{14}$$

3.5. Adjoint Solver

As described in the Bayesian optimisation with gradient information section within the literature review it is clear that there is a major benefit of incorporating function gradient information into the model. Therefore the gradient information for the Gaussian process must be derived. As the Gaussian and Bayesian methodology for incorporating derivatives has been covered in [9], [10] and [11] the requirement is to apply this method to CFD. This will be the main focus for this section and not the development and implementation of incorporating gradient information into Gaussian and Bayesian methods.

The Gaussian process model requires the derivatives of the function that is being minimised with respect to the design variables. In the case of the draft tube the function being minimised is the pressure difference between the inlet and outlet. Again, in regards to the CFD, the variables are the control points for the spline that defines the geometry. Therefore, the gradients of the pressure difference with respect to the spline control points need to be derived.

This can be achieved in many ways. One possible method is to use finite differences, that being moving each control point of the spline individually in each coordinate direction and then calculating the pressure drop and ultimately the gradients.

One other method is to use the adjoint method which was initially proposed by Pironneau [23] in 1974 and further developed by Jameson in [24], [25] and by Elliott in [26].

Normally the flow and adjoint solutions are computed at the same time, meaning that the adjoint solution evolves as the primal flow solution develops. What is proposed here is generating the adjoint solution on the fully converged flow solution. The equations solved are the same in both and are derived in the following section, but the implementation is different.

Following what is set out in [24], [25] and [26] the adjoint equations are derived. Assuming that there is an objective function of the form $J(\boldsymbol{Q}^*, \boldsymbol{X}, \boldsymbol{\alpha})$ where $\boldsymbol{Q}^* = (\boldsymbol{v}, p)$ is the vector of converged flow variables, $\boldsymbol{X}$ is the vector of grid points and $\boldsymbol{\alpha}$ is a vector of design variables, here the control points of the spline. This objective function can be pressure loss, drag, lift or any other function. By differentiating the objective function with respect to the design variables and utilising the chain rule leads to equation 15.

$$\frac{dJ}{d\boldsymbol{\alpha}} = \left(\frac{\partial J}{\partial \boldsymbol{Q}^*}\right)\frac{d\boldsymbol{Q}^*}{d\boldsymbol{\alpha}} + \left(\frac{\partial J}{\partial \boldsymbol{X}}\right)\frac{d\boldsymbol{X}}{d\boldsymbol{\alpha}} + \frac{\partial J}{\partial \boldsymbol{\alpha}} \tag{15}$$

Now by taking the derivatives of the Navier-Stokes equations $\boldsymbol{R}(\boldsymbol{Q}^*, \boldsymbol{X}, \boldsymbol{\alpha})$ using the same process gives equation 16.

$$\frac{d\boldsymbol{R}}{d\boldsymbol{\alpha}} = \left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}^*}\right)\frac{d\boldsymbol{Q}^*}{d\boldsymbol{\alpha}} + \left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}}\right)\frac{d\boldsymbol{X}}{d\boldsymbol{\alpha}} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\alpha}} = \boldsymbol{0} \tag{16}$$

By introducing a vector of Lagrange multipliers, $\boldsymbol{\lambda} = (\boldsymbol{u}, q)^T$, to equation 16 and then combining equations 15 and 16 gives equation 17.

$$\frac{dJ}{d\boldsymbol{\alpha}} = \left[\left(\frac{\partial J}{\partial \boldsymbol{Q}^*}\right) + \boldsymbol{\lambda}^T\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}^*}\right)\right]\frac{d\boldsymbol{Q}^*}{d\boldsymbol{\alpha}} + \left[\left(\frac{\partial J}{\partial \boldsymbol{X}}\right) + \boldsymbol{\lambda}^T\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}}\right)\right]\frac{d\boldsymbol{X}}{d\boldsymbol{\alpha}} + \frac{\partial J}{\partial \boldsymbol{\alpha}} + \boldsymbol{\lambda}^T\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\alpha}} \tag{17}$$

Then by setting the Lagrange multipliers in such a way as to allow equation 18

$$\boldsymbol{\lambda}^T\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}^*}\right) = -\left(\frac{\partial J}{\partial \boldsymbol{Q}^*}\right) \tag{18}$$

allows for the solution of the Lagrange multipliers to be calculated. The boundary conditions which are derived from equation 18 follow. Equation 18 requires iteration, similar to that of the primal CFD solution. Equation 18 also shows that the majority of the flow of adjoint variables is in the opposite direction to those of the CFD, primal, flow solution. This means that information travels from the CFD outlet towards the CFD inlet in the adjoint solution.

Inlet and Wall Boundary Conditions

$$\boldsymbol{u}_t = \boldsymbol{0} \tag{19}$$

$$u_n = -\frac{\partial J_\Gamma}{\partial p} \tag{20}$$

$$\boldsymbol{n} \cdot \nabla q = 0 \tag{21}$$

Where J_Γ is the contribution of the objective function from the boundary, $\boldsymbol{n}$ represents the normal to the boundary and the subscripts n and t represent normal and tangential directions to the boundary.

Outlet Boundary Conditions

$$q = \boldsymbol{u} \cdot \boldsymbol{v} + u_n v_n + \upsilon(\boldsymbol{n} \cdot \nabla) u_n + \frac{\partial J_\Gamma}{\partial v_n} \tag{22}$$

$$\boldsymbol{0} = v_n \boldsymbol{u}_t + \upsilon(\boldsymbol{n} \cdot \nabla)\boldsymbol{u}_t + \frac{\partial J_\Gamma}{\partial \boldsymbol{v}_t} \tag{23}$$

Surface Sensitivity

Once the adjoint solution has been calculated the sensitivity of a specific point, β, on the surface can be calculated which will be utilised in the following section to generate the gradient of the flow objective function with respect to the spline control points. This follows similar methodology set out in [27].

One method to write the objective function without altering the value is to augment it with the residual of the Navier-Stokes equation over the domain Ω is shown in equation 24.

$$L = J + \int_\Omega (\boldsymbol{u}, q)\boldsymbol{R} d\Omega \tag{24}$$

This is only possible due to the fact that for a converged flow solution the residual of the Navier-Stokes equations is zero. Therefore, for a converged flow solution $L = J$. Following this the total variation with respect to some point on the boundary β is given by equation 25.

$$\delta_\beta L = \delta_\beta J + \int_\Omega (\boldsymbol{u}, q)\delta_\beta \boldsymbol{R} d\Omega \tag{25}$$

Taking the variation of the residual of the Navier-Stokes equation to be zero for a converged solution and assuming that its variation is split into contributions from the flow geometry, pressure and velocity terms given in equation 26.

$$\delta \boldsymbol{R} = \delta_\beta \boldsymbol{R} + \delta_{\boldsymbol{v}} \boldsymbol{R} + \delta_p \boldsymbol{R} = 0 \tag{26}$$

Then by rearranging equation 26 and substituting this into equation 25 leads to equation 27.

$$\delta_\beta L = \delta_\beta J - \int_\Omega (\boldsymbol{u}, q)\delta_{\boldsymbol{v}} \boldsymbol{R} d\Omega - \int_\Omega (\boldsymbol{u}, q)\delta_p \boldsymbol{R} d\Omega \tag{27}$$

As the objective function, for the draft tube, only relies on the domain inlet and outlet, i.e. $J_\Omega = 0$, equation 27 can be expanded and simplified for terms contained on the boundary leading to equation 28

$$\delta_\beta L = \delta_\beta J - \int_\Gamma d\Gamma \boldsymbol{u} \cdot \boldsymbol{n} \delta p - \int_\Gamma d\Gamma (\boldsymbol{n}(\boldsymbol{u} \cdot \boldsymbol{v}) + \boldsymbol{u}(\boldsymbol{v} \cdot \boldsymbol{n}) + 2\nu \boldsymbol{n} \cdot D(\boldsymbol{u}) - q\boldsymbol{n}) \cdot \delta \boldsymbol{v}$$
$$+ \int_\Gamma d\Gamma 2\nu \boldsymbol{n} \cdot D(\delta \boldsymbol{v}) \cdot \boldsymbol{n} \tag{28}$$

where D($\mathbf{u}$) is the rate of strain tensor, i.e. $D(\boldsymbol{u}) = \frac{1}{2}(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T)$. As this objective function is solely reliant on the values at the inlet and outlet there is no explicit dependence of the flow values at the wall. Therefore, the objective function $\frac{\partial J}{\partial \beta} = 0$ and that $\mathbf{u} = 0$ at the wall equation 28 reduces to equation 29.

$$\delta_\beta L = -\int_{wall} d\Gamma \beta \upsilon (\boldsymbol{n} \cdot \nabla)\boldsymbol{u}_t \cdot (\boldsymbol{n} \cdot \nabla)\boldsymbol{v}_t \tag{29}$$

From equation 29 the surface sensitivity can be derived and is shown in equation 30.

$$\frac{\partial L}{\partial \beta} = -A\upsilon (\boldsymbol{n} \cdot \nabla)\boldsymbol{u}_t \cdot (\boldsymbol{n} \cdot \nabla)\boldsymbol{v}_t \tag{30}$$

This value for sensitivity will be positive for movement with the surface normal and negative for movement in the direction of the negative surface normal. This sensitivity value is, in essence, the magnitude of the movement in the normal direction of the surface point. Moving the surface could optimise the geometry, however these methods tend to find the closest minima to the initial configuration, whereas Bayesian methods find the global optima.

There is no requirement for equation 30 to be calculated in an iterative fashion as the adjoint and CFD flow variables, primal, are known.

Gradient of the Objective Function

As the gradient of the objective function with respect to the spline control points is required for the Bayesian optimiser with gradient information there requires a formulation that converts the gradients at specific points on the surface to gradients with respect to the spline control points.

There are a few possible proposed methods for the generation of this information. These possible methods are proposed in the following sections. As the control points only manipulate the points on the spline they do not impact the surface points on the other walls of the draft tube. Therefore, as the control points have no impact on the other surfaces their surface sensitivity values can be ignored.

Single Point Method

Initially a method which is relatively simple is required. One possible simple method is to project the control point onto the closest point on the spline surface and then take the value of the closest location where the surface sensitivity has been calculated. It is argued that the control point of the spline influences the point closest to it most and therefore the major influence of the gradient of the objective function with respect to the control point would be the approximately the value of the surface sensitivity of the nearest point to the control point.

Multiple Point Method

In order to improve the accuracy of the method of generating the gradients of the objective function with respect to the control point of the spline a more complex method is proposed. Control points within the spline have major influence over certain regions, convex hulls, in their close vicinity and less influence further away. Therefore, following a similar argument to

the previous method, it is argued that the gradient is being influenced largely by the surface sensitivities within the convex hull region of the control point.

To generate the gradient value from multiple surface sensitivities an average is proposed. This could be an equal weighting between all the surface sensitivity values or weighted by the distance to the control point of the spline.

Reverse Spline Method

A possible method for generating more accurate gradient information from the surface sensitivities would be to reverse the method in generating the spline in the first instance. The method of generating the spline follows a prescribed algorithm. This algorithm is method dependent and the method chosen here is Catmull-Clark splines. The main algorithm is to generate further points based on known control points, this is one level of refinement but multiple levels of refinement can be achieved.

The proposed method calculates how much of each control point influences its position and this percentage can then be utilised to take the average of the sensitivity values that the control points have influence over. This, then should incorporate information of the surface sensitivity from all the locations on the surface to all of the initial control points and as such have higher accuracy than the methods proposed previously.

Spline Point Gradient Method

The final method that has been investigated promises to be highly accurate due to its mathematical grounding within the chain rule. The method proposed can be seen in equation 31.

$$\frac{\partial L}{\partial \boldsymbol{x}} = \frac{\partial L}{\partial \boldsymbol{\beta}} \frac{\partial \boldsymbol{\beta}}{\partial \boldsymbol{x}} \tag{31}$$

Where $\boldsymbol{x}$ is the vector of spline control points. As the surface sensitivities $\frac{\partial L}{\partial \boldsymbol{\beta}}$ have already been calculated from equation 30 for all points on the surface what is finally required is the change in the surface points $\boldsymbol{\beta}$ due to a change in the spline control points $\boldsymbol{x}$.

The method to generate this change in surface point due to a change in spline control point is relatively straightforward. It utilises the first order gradient calculation utilising a forward or backward differencing scheme, meaning that the distances moved by the surface points are investigated when the spline control points are moved a little.

4. Presentation of Analytical Results

In order to optimise CFD geometries initially the optimisers must be able to optimise a range of functions. Initially all code was tested on x^2. Following this more complex equations were tested. The result for each optimisation follow. The processes are random averages and variances are taken over 50 samples.

4.1. Genetic Algorithm

As discussed in section 3.2 there are many parameter choices for the GA. To keep the testing fair these parameters were kept constant for each of the functions evaluated.

Table 1: Showing the parameter choice for the GA.

Parameter	Value
Number of Iterations	1000
Population	50
Alpha factor	30%

Performance on $f(\boldsymbol{x}) = x^2$

Having written the GA it requires to be tested on various test cases. Initially the code was written for one dimension. For this case the domain for the minimisation is $x \in$ [-5, 5].

Table 2: Showing the mean and standard deviation for the minimum generated by the GA.

	Mean of $f(x)$	Standard Deviation
Analytical	0.0	-
GA	$1.5\text{x}10^{-6}$	$1.1\text{x}10^{-11}$

Performance on $f(\boldsymbol{x}) = x_1^2 + x_2^2$

The GA was then tested on a function that takes higher dimensions. Again, the domain of interest was $x_1, x_2 \in$ [-5, 5].

Table 3: Showing the mean and standard deviation for the minimum generated by the GA.

	Mean of $f(\boldsymbol{x})$	Standard Deviation
Analytical	0.0	-
GA	$1.3\text{x}10^{-3}$	$3.1\text{x}10^{-6}$

4.2. Gaussian Process

Various methods are used to test whether the Gaussian process is implemented correctly. One showing the sample paths for the kernel matrix, another being the fitting to arbitrary data.

Sample from Prior

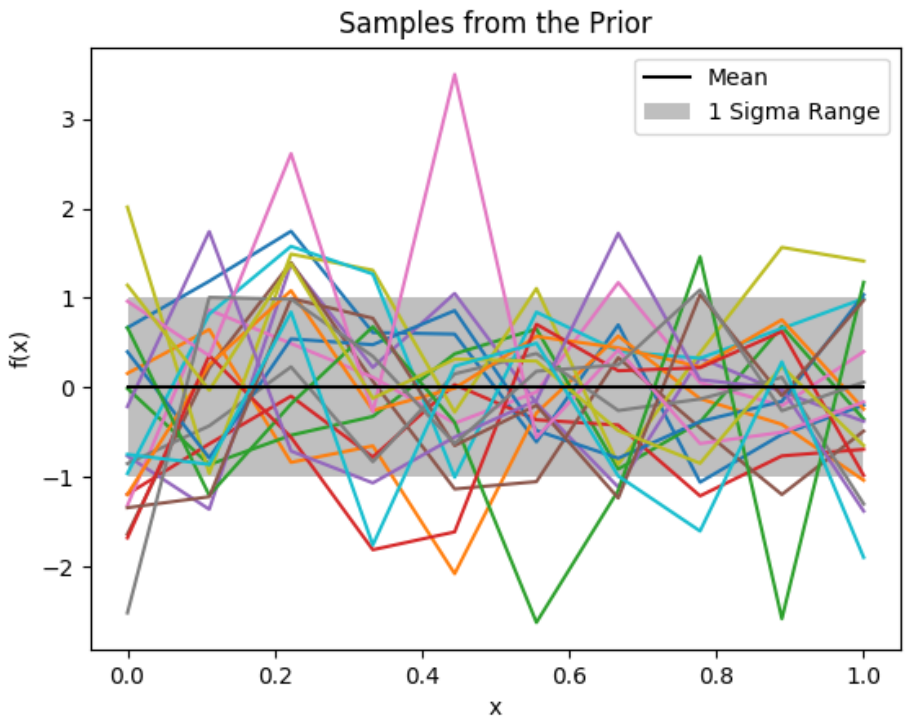


Figure 1: Showing the range of distribution for 20 possible function predictions for the Gaussian process.

Samples from the Posterior

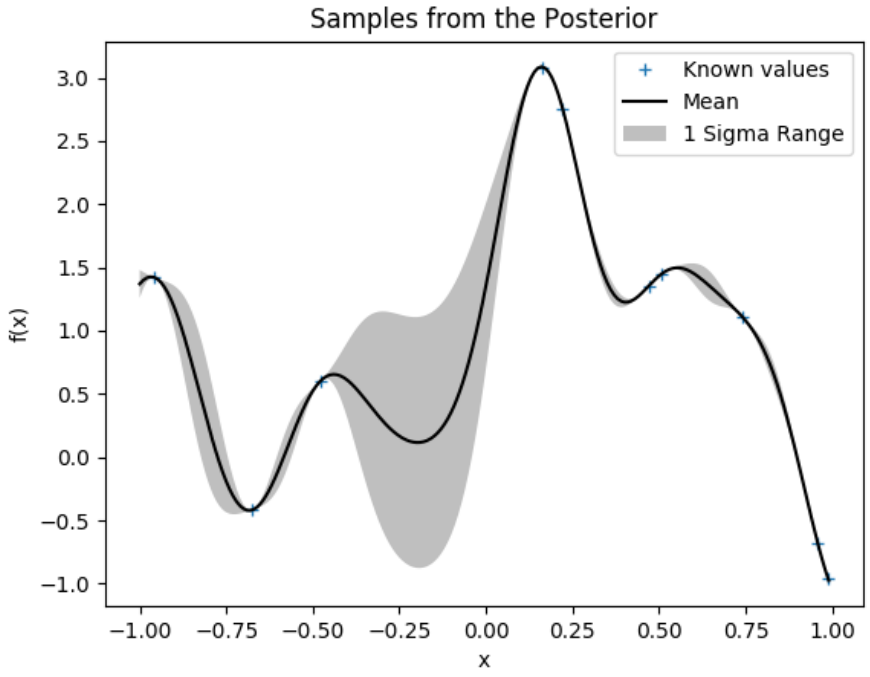


Figure 2: Showing the prediction of the Gaussian process of f(x)

4.3. Bayesian Optimiser

As discussed in section 3.4 there are many parameter choices for the Bayesian method. To keep the testing fair these parameters were kept constant for each of the functions evaluated.

Table 4: Showing the parameter choice for the BO.

Parameter	Value
Number of Additional Points	11

Performance on $f(\boldsymbol{x}) = x^2$

Having written the BO it requires to be tested on various test cases. Initially the code was written for one dimension. For this case the domain for the minimisation is $x \in$ [-5, 5].

Table 5: Showing the mean and standard deviation for the minimum generated by the BO.

	Mean of $f(x)$	Standard Deviation
Analytical	0.0	-
BO	$1.6\text{x}10^{-6}$	$4.4\text{x}10^{-12}$

Performance on $f(\boldsymbol{x}) = x_1^2 + x_2^2$

The BO was then tested on a function that takes higher dimensions. Again, the domain of interest was $x_1, x_2 \in$ [-5, 5].

Table 6: Showing the mean and standard deviation for the minimum generated by the BO.

	Mean of $f(\boldsymbol{x})$	Standard Deviation
Analytical	0.0	-
BO	$5.3\text{x}10^{-3}$	$4.6\text{x}10^{-5}$

4.4. Comparison of Optimisers

So far the optimisers have been compared for a given number of function evaluations to show that they can indeed find the theoretical minima of functions. However, the whole purpose is to find the optimiser that finds the minima in the fewest number of function evaluations. This is why this comparison is required.

In order to determine the best method to continue with a plot of the two methods, GA and BO, their mean and standard deviation were plotted against number of function evaluations. This is shown in figure 3. These plots were created using an average over 50 optimisations for each

number of function evaluations. This also helps determine if there is any requirement in researching the gradients of the objective function with regards to the spline control points.

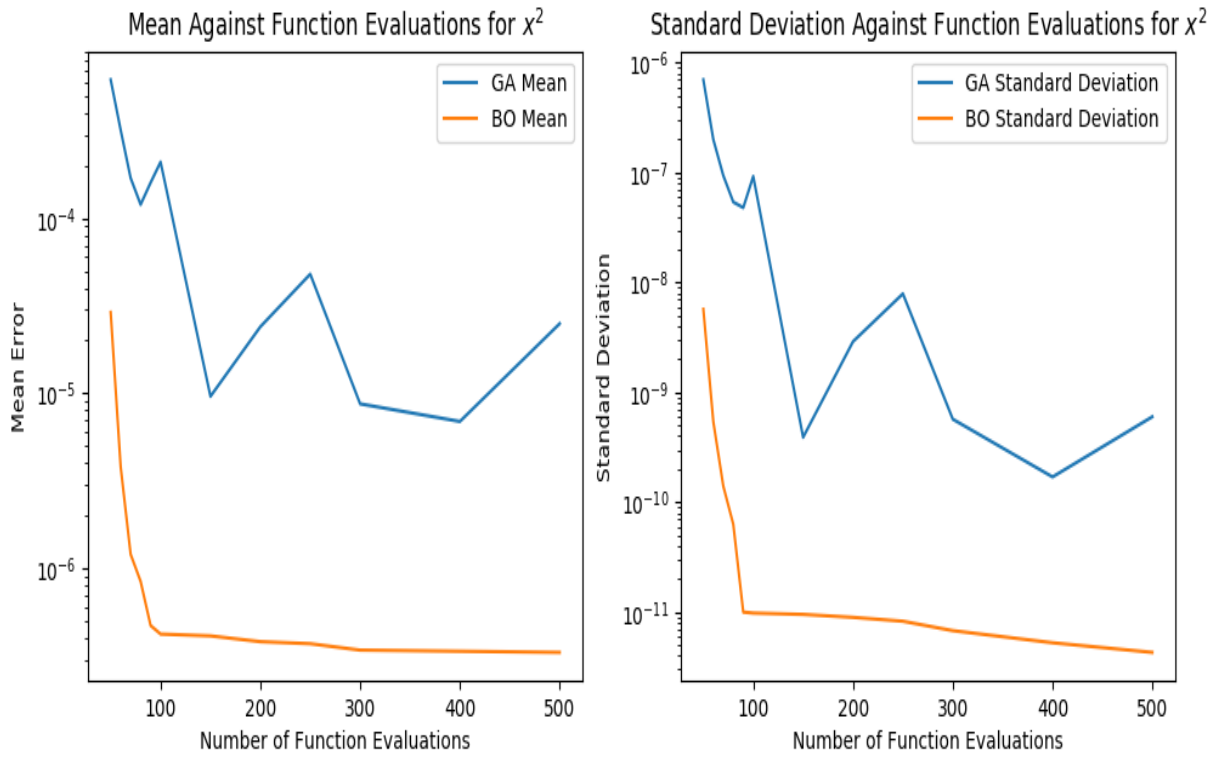


Figure 3: Showing mean and standard deviation against function evaluations for x^2.

4.5. Adjoint Solver

To determine if the code has been implemented correctly, and as there is no physical meaning to the Lagrange multipliers, the code was compared against a pre-written code that computes the adjoint solution as the simulation evolves. This again was tested on a simple pseudo-2D case.

Adjoint Variables

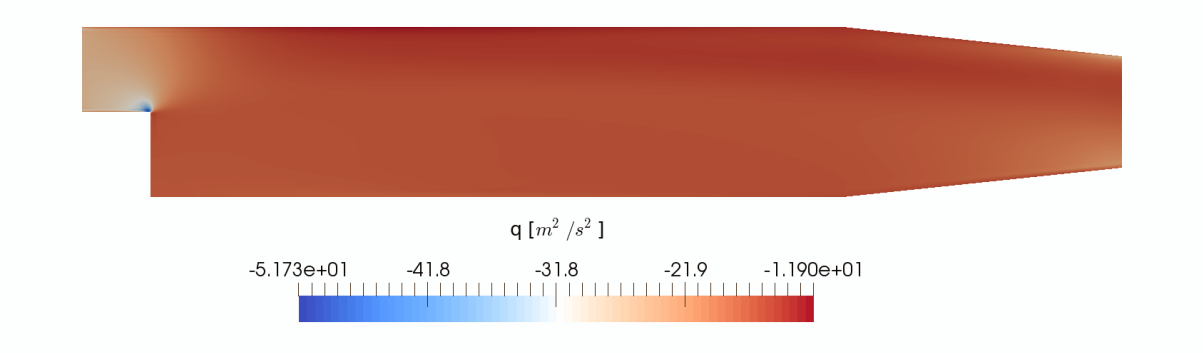


Figure 4: Showing the adjoint variable q for the adjoint solution from a converged primal CFD solution.

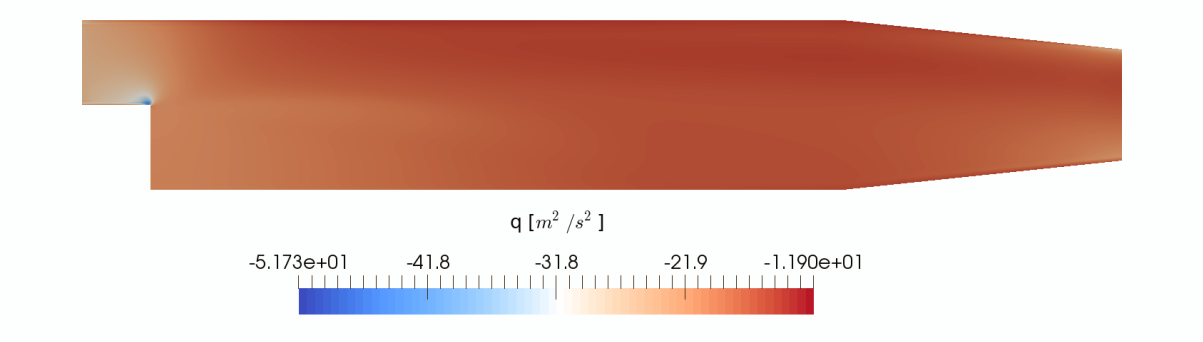


Figure 5: Showing the adjoint variable q for the adjoint solution from the adjoint solver as the CFD evolves.

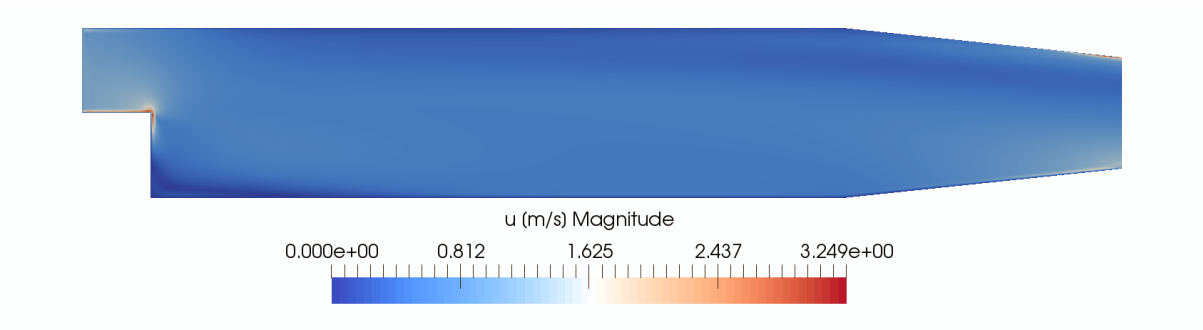


Figure 6: Showing the magnitude of adjoint vector u for the adjoint solution from a converged primal CFD solution.

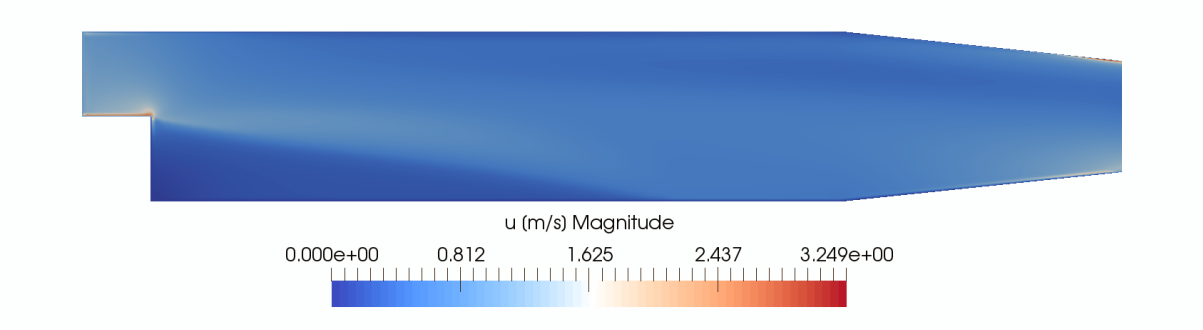


Figure 7: Showing the magnitude of adjoint vector u for the adjoint solution from the adjoint solver as the CFD evolves.

Surface Sensitivity

Following this calculation of the adjoint flow solution the surface sensitivity could be calculated. Figure 8 shows the surface sensitivity for the wall in the region around the backward facing step.

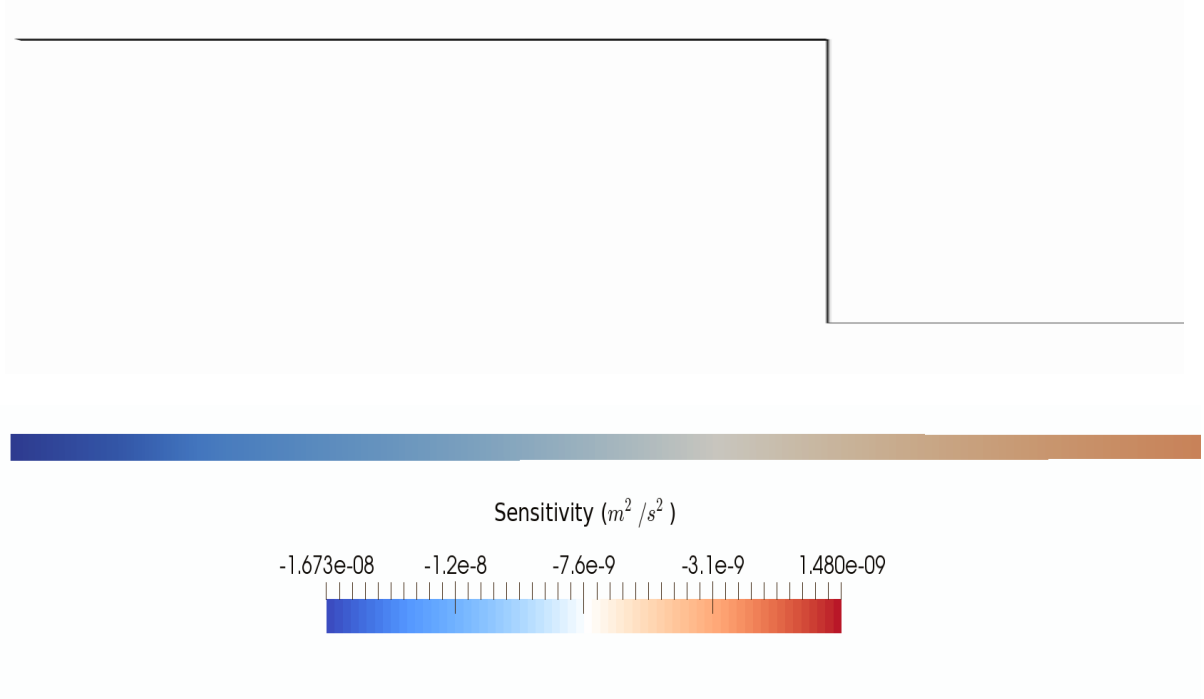


Figure 8: Showing the sensitivity of the objective function with respect to the location on the backward facing step.

Gradient of the Objective Function

The method was tested on simple channel flow over a bump which was controlled by a spline. Its surface sensitivities in the surface normal direction are shown in figure 9 where the vectors are in the surface normal direction.

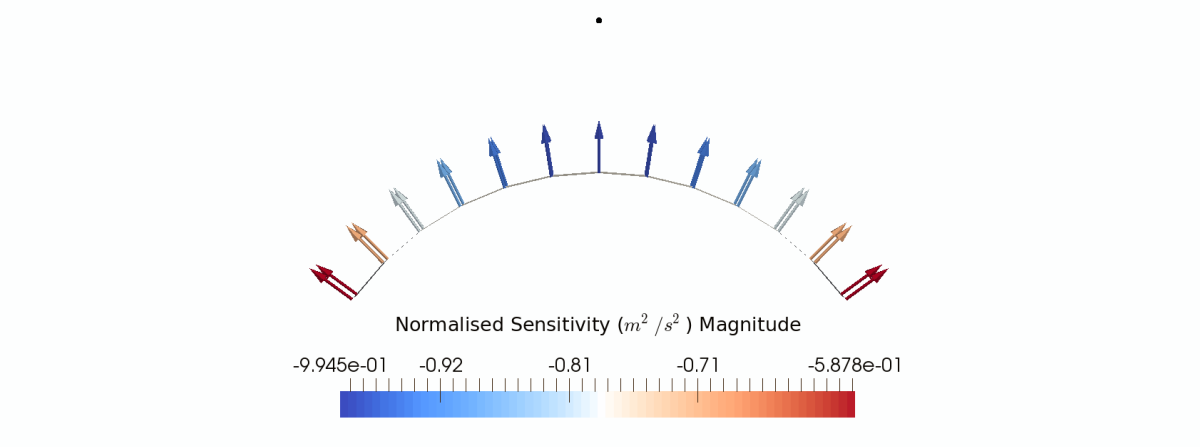


Figure 9: Showing the normalised surface sensitivity values along with the control point for the spline.

Table 7: Showing the calculated gradients at the control point from the surface sensitivities.

Method	$\frac{\partial L}{\partial x_1}\ (m^2 s^{-2})$	$\frac{\partial L}{\partial x_2}\ (m^2 s^{-2})$
Single Point	0	-0.995
Multiple Point (Equal Weighting)	0	-0.784
Multiple Point (Distance Weighting)	0	-0.867
Reverse Spline	0	-0.883
Spline Point Gradient	0	-0.876

5. Discussions and Conclusion

Parameter tuning for the optimisers was undertaken by running the simple cases a few times and adjusting the parameters. From this the best values for the parameters could be chosen and utilised. This was undertaken for each optimiser.

5.1. Optimiser Abstract Base Class

An abstract base class was written for the optimiser. As all optimisers are required to do the same function they should inherit their behaviour from a template of what the class should do. This then enabled both the GA and BO classes to have the same function names. This allowed the general code to remain constant but initialise the optimiser to whichever the user requires, keeping core functionality of the code the same.

5.2. Genetic Algorithm

Code Implementation

One Dimension

Initially the code was developed utilising the methodology written in section 3.2. The implementation initially allowed only one dimension. This was to allow for the code to be tested so that the algorithms produced the desired outcomes for a simple function.

Various aspects of python were leveraged, including the ability to parse function names into classes so that they could be called from within the class. This enabled the writing of the optimiser to be generic in a sense that the optimiser could be given any one-dimensional function and would be able to optimise this.

Further to this the optimiser was initially written to be able to minimise the function. This was altered to allow both minimisation and maximisation problems to be dealt with. This was achieved through the utilisation of $min(f(x)) = max(-f(x))$. Therefore, all that was added

to the code was the ability to multiply the objective function by -1 if a maximisation problem was encountered.

All of the possible parameters that are available to the optimiser were also initialised when the class was created allowing the user to have full control over what the optimiser does. Also this enables the adaption of the optimiser to each individual problem.

Arbitrary Dimensions

In order to deal with more complicated problems, such as real world problems, the code written for the one-dimensional case was altered to allow for arbitrary dimensions. This was achieved through the parsing of a list of variables rather than a single variable. A few other alterations were required to be able to deal with these, however they were not major alterations. The reason why the code was re-written to allow for multiple dimensions was that many real-world optimisation problems are multi-dimensional problems.

Results

From table 2 it can be seen that the optimiser produces very accurate results. This is expected as there are 1,000 generations produced. This relates to there being 2,050 function evaluations being executed, two children are produced every generation and also the initial 50. This optimiser took little time for the simple case however it is clear that if the time taken for the function evaluation is large then this method would not be very efficient. The standard deviation suggests that the algorithm has good grouping and hence can find the minimum consistently when performed multiple times.

Table 3 shows that as the dimensions of the problem increases the accuracy decreases and standard deviation increases. This is mainly due to there being a larger space to search in. This is generally the case for any optimisation problem. That being said, the optimiser is still able, on average, to get fairly close to the optimum value. Again, the optimiser would be time consuming if the function took a long time to compute. The algorithm had some distribution of results, due to the slightly higher standard deviation, which is expected as there is a larger space to search, but the algorithm does fairly well on grouping the solutions around the minimum.

5.3. Gaussian Process

Code Implementation

One Dimension

Initially the code was written to be able to handle one dimension. This was achieved by having a class for the Gaussian process and the class containing reference to the kernel class. This allowed the use of the polymorphic behaviour of classes.

An abstract base class was written for the kernel. This abstract base class states that every derived class must contain implementations and functions outlined in the base class. By leveraging this method the Gaussian process could be written to deal with any arbitrary

implementation of a kernel function. This also allowed to code for kernels to be kept separately from the main Gaussian process code, which has the added benefit of ease of debugging the code.

Initially the Gaussian process was written using the details in section 3.3. As the Gaussian process itself holds the matrices K, $K(\boldsymbol{x}, \boldsymbol{x}_*)$, $K(\boldsymbol{x}_*, \boldsymbol{x})$ and $K(\boldsymbol{x}_*, \boldsymbol{x}_*)$ and all the required maths for computing the mean, variance and optimising the marginal likelihood this did not require to be multi-dimensional, and so the only component that was constrained to one dimension was the kernel class.

Initial implementation of the kernel could only manage one dimension. This was due to the kernels reliance on the GA written previously being only one dimensional. This GA was utilised in the optimisation of the hyperparameters of the kernel along with returning the function optima.

The implementation made heavy use of the numpy library for matrix inversion, matrix multiplication and also calculating the determinant of the matrix.

Multi-Dimensional

Having proved that the Gaussian process was able to predict the one-dimensional case the code was adapted to deal with arbitrary dimensional problems. This, again, mostly relied on implementing the vector multiplication contained within the kernel function. This was altered, along with the updated GA written previously, to now deal with arbitrary dimensions.

Results

From figure 1 it is clear that without any data generated the Gaussian process displays a mean function value of 0. This is expected because this Gaussian process assumes that the function has a mean value of 0 and also the standard deviation is 1. Figure 1 clearly shows this.

Figure 1 also shows a small sub set of all possible function forms. This is the main idea with a Gaussian process in that the mean value is the mean of all possible functions that fit the current data. This is also where the standard deviation is generated from. It can be seen that the majority of the function values are within 1 standard deviation, therefore the process generates a normal distribution of functions around the mean value. Figure 1 shows a range of possible functions of the Gaussian process, these functions are rather sharp due to the kernel choice and many other kernels would produce different, smoother function evaluations.

From figure 2 it can be seen that the Gaussian process produces a very good fit with the data. This is shown by the mean function passing through each and every data point. Figure 2 also shows that the standard deviation at a known point is zero. This is due to the functions that do not intersect this point being ‘removed’ and not considered possible for the function mean. This means that functions that do not pass through the points are invalid.

Further to this it can be seen that in areas where there are no nearby known points there is large standard deviation. This is due to the process being unsure of which of the possible functions

could be the true function in this region. Therefore, to improve the estimation a point in this region could be added.

The data used for the testing of the Gaussian process was randomly generated. This ensured that the process could not learn a function.

5.4. Bayesian Optimiser

Code Implementation

One Dimensional

Having developed the one dimensional Gaussian process and ensuring that it performs as expected, the BO could be written. This involved following the mathematics written in section 3.4. As the Gaussian process initially was only capable of one dimension the BO followed suit.

The optimiser inherits from the abstract base class that was mentioned previously. BO requires its own internal Gaussian process model. The model that was used was the algorithm as described in section 5.3. This was beneficial as the knowledge of the implementation allowed the easy integration within the Bayesian methodology.

The acquisition function, equation 12, was implemented. This required the implementation of both standard normal density and distribution using equations 13 and 14. Equation 12 required maximising to find the location of the search point that would improve the surrogate model the most. This was achieved utilising the GA written previously.

Following this the model parameters, namely the number of additional points to be evaluated, were implemented in such a way that on initialisation the user could specify these.

Multi-Dimensional

Having tested the BO code on one dimension and updating the kernel function in the Gaussian process to deal with multiple dimensions, the BO could be updated to deal with multiple dimensions. This was as simple as parsing lists rather than individual values.

Results

From table 5 it can be seen that the BO methodology has a greater accuracy than that of the GA. This accuracy is present with only 13 function evaluations, 2 initial data values and a further 11. This shows that the Bayesian optimiser performs significantly better, due to the increased accuracy and also the reduced number of function evaluations, 13 over GA's 2050 for a similar magnitude of accuracy. Further to this the minimum has a very low value for the standard deviation which means that the algorithm consistently finds very close to the minimum.

This shows promise for the application of Bayesian methods to the optimisation of CFD problems as function evaluations in CFD can be extremely expensive and time consuming. Therefore it is clear that this accuracy and reduced number of function is beneficial for industry.

Table 6 shows, again, that as the number of dimensions of the problem increases the accuracy decreases. This is expected as there is a larger space to search in and therefore it is more difficult to find the minimum. The algorithm does show an increased standard deviation again for the same reason as mentioned earlier.

One of the main reasons why the Bayesian method produces superior results to those of the GA is that the GA does not create a model for the function, it just minimises and makes decisions based on the fitness values of the population. On the other hand the Bayesian method generates a surrogate model, a function estimate, for the function being evaluated. This has the added benefit that each and every evaluation of the function enhances the estimation of the surrogate model and therefore reduces the total number of possible functions that the surrogate model could be. This leads to, after a relatively few number of evaluations, very accurate representation of the function.

5.5. Comparison of Optimisers

What is clear from figure 3 is that GAs are a completely random process and are therefore not guaranteed to find the minimum with an increase in function evaluations. This is evident by the mean and standard deviation for the GA not reducing at a constant rate with an increase of function evaluations.

Figure 3 also shows that the mean for the GA plateaus after about 150 function evaluations. This is shown by the lack of decrease in the mean and standard deviation. This again, is due to the random nature of the method and once it is close to the optima the crossovers between individuals may not be guaranteed to improve the solution.

The BO method, on the other hand, is drastically lower for the same number of function evaluations. This is mainly due to the method used as it creates a surrogate model for the function and therefore increasing the number of function evaluation increases the accuracy of the surrogate model.

It is also clear that the method is not random like the GA. This is seen by the continued, albeit slow, decrease after 100 function evaluations in mean and standard deviation with increased number of function evaluations.

Also available from figure 3 is that there is very little benefit in increasing the number of function evaluations above a certain point for both the GA and BO methods as this will have little benefit for the mean value obtained. This again is at a lower value for the BO method, around 90, than that of the GA, around 150, however, this is problem dependent.

Figure 3 shows the main benefits of utilising the BO method, which include increased accuracy for lower number of function evaluations and in the case of CFD is beneficial due to the cost of the function evaluations. Also, each function evaluation increases the accuracy of the model as opposed to the GA method being random. It is for these reasons that utilising the Bayesian method will be investigated further.

It is also difficult to investigate the time taken by the optimisers as they are completed within a few seconds for each case, which is due to the relative simplicity of the objective function being optimised. It would be beneficial to investigate the time taken on real CFD cases, however this would be undertaken between the three optimisers, GA, BO and BO with gradient information, developed on the same case.

5.6. Adjoint Solver

Adjoint Variables

The reason why the finite difference method was not used is that it requires $(N \times M) + 1$ function evaluations for the derivatives, where N is the number of variables and M is the dimension of the flow solution. Whereas the adjoint solver only requires one CFD solution and one adjoint solution to calculate the gradients. Therefore, it is for this reason that the adjoint method was chosen.

Code Implementation

To simply implement the solution of equation 18 and implementing equations 19-23 the CFD toolbox OpenFOAM was used. OpenFOAM was used because it already has the main functionality for reading in the mesh and all of the solution methods and various other libraries and utilities were pre-written. These benefits made the implementation of the code relatively easy as separate classes and functions for reading in solutions and meshes did not need to be written and thus saved time.

Further to this the case was tested on a relatively simple geometry with the solution being generated in OpenFOAM. Therefore, this is one other benefit of writing the implementation into the OpenFOAM framework. Further to this the CFD sub-team are all using OpenFOAM and therefore no additional code is required to be written to convert into a format that OpenFOAM can read in.

Many of the functions were pre-written, including various functions for calculating the gradients within cells. These would all be useful as it reduces the chances that the equations would be implemented incorrectly. It is worth comparing the results against another adjoint solver as this gives confidence that the code is implemented correctly or indicate where to investigate if there is an implementation error.

This other adjoint solver is pre-written and calculates equation 18 with the implementation of geometry optimisation. This other adjoint solver introduces a term that penalises the flow in cells that are determined to be detrimental to the flow, this is achieved through reducing the porosity within the cell. This then slightly alters the final primal flow solution and hence the adjoint solution. Therefore, it is useful to compare against but appreciate that there will be differences as equation 18 and the method chosen do not alter geometry.

Results

Figures 4 and 5 show the solution for the same adjoint variable q. Figure 4 shows the implementation whereby the adjoint solution is computed from a converged primal CFD solution whereas figure 5 shows the solution of the adjoint variable q as the CFD solution evolves.

From these two figures it is clear that the solution of the adjoint from the converged CFD solution gives very little difference compared to computing the adjoint as the flow evolves. Therefore, based on this, the implementation appears to be correct and there appears to be no reason as to why this method cannot be used.

Equation 18 also shows that the flow of the adjoint variables is in the opposite direction to those of the CFD solution. This can be seen in the figures by the low value of q by the backward facing step. This is shown by the wake region of the step being in the direction of the inlet showing that the adjoint equations flow in the opposite direction. This also identifies that the CFD solution is sensitive to this backward facing step whereas the flow is not as sensitive in other regions as the values are relatively close to zero.

Figures 6 and 7 show the solution for the magnitude of the adjoint variable $\boldsymbol{u}$. These figures show a large amount of similarity but near to the backward facing step there are some differences. These differences are visible by a smaller magnitude, shown by the darker region, from the step to the floor about half way along the channel. This difference is due to the method that the solvers utilise.

This difference is in terms of the equations solved. Figure 6 shows the pure adjoint solution from the converged CFD solution whereas in figure 7 the equations solved includes a penalty term to cells, as described above, that are detrimental to the flow solution and this value determines where the geometry should be removed. This appears in the region downstream of the step as this is the region causing the most amount of losses within the flow. Besides this the solutions appear to be very similar in the regions that do not have these penalty terms introduced. It therefore appears that the adjoint solver has been implemented correctly.

One issue with validating the adjoint flow solution is that the terms, the Lagrange multipliers, do not represent anything physical, and therefore it is difficult to determine if the solution is a solution to equation 18. One important aspect to be aware of is that the adjoint equation is a transport equation and therefore must still have boundedness, conservativeness and transportiveness, which appears to be the case.

One other method to ensure that the solutions of the adjoint equations are correctly implemented is ensuring that the residual, difference between iterations, drops below a given value. Here the value chosen was $1\text{x}10^{-5}$. Finally, by visual inspection of figures 4-7 it appears that the values are of reasonable orders of magnitude. It is therefore assumed that the equations have been implemented correctly as there are no experimental or physical data to compare the results to.

Further to this the adjoint flow scheme for the backward facing step appears to what would be expected for the transport of variables that go in the opposite direction to the CFD solution. This also helps indicate that the adjoint solver has been implemented correctly. This is supported by the lack of shockwaves and discontinuities in the flow suggesting that the implementation is correct.

Surface Sensitivity

Code Implementation

Again, as with the implementation of the adjoint flow solver the surface sensitivity was implemented in OpenFOAM. The code was initially written to calculate the sensitivity on a hard-coded name of a patch. A patch in OpenFOAM is a group of faces called with similar properties. This was initially undertaken to ensure that the implementation of the code was correct.

Once the implementation had been validated and was deemed to be implemented correctly then the code was altered to take in command line argument for calculating the sensitivity on any given patch within the mesh. Further to this that code was altered to be able to calculate the sensitivities on any number of patches. This was implemented by taking in command line arguments and looping over them. If a patch was specified that was not contained within the mesh, then the code ignores it and moves onto the next patch name.

Results

Again, as with the adjoint variables, there is no experimental data to validate the sensitivity solution. The surface sensitivity does have some physical meaning in that it suggests which direction to move the geometry to improve the solution. However figure 8 shows that the surface should be adjusted in very small, $1x10^{-9}$, values. This again is due to the formulation of the adjoint method. It is a gradient descent method, which does not find the minimum of the objective function in one step but finds the gradient and then alters the geometry and searches in the negative gradient direction by a small amount. This has the effect of taking small steps down the gradient towards the minimum, hence, why the sensitivity is a small value.

Figure 8 also shows that the flow is greatly impacted by the initial step. This is seen by the negative and positive sensitivity values either side of the step evidencing that the objective function is heavily dependent on the step. Consequently the surface sensitivity attempts to smooth out the step. This can be seen in figure 8 as the portion before the step is suggested to move downward and the vertical step is to move to the right, having the effect of smoothing the step out.

It appears that the surface sensitivity has been implemented correctly as the conclusions drawn from the results seem to be logical. The values obtained also fit with the gradient descent method in that the values are small due to the method taking small steps in the direction of the negative gradient, shown by the negative sign in the sensitivity equation.

Gradient of the Objective Function

The total timescale for generating the gradient is around the same time as the primal CFD solution. This is because the adjoint solver is solved for every cell within the mesh iteratively, thus taking a similar amount of time as the primal solution. The adjoint solution is the component that takes the most amount of time as the surface sensitivity and the gradient calculation only implement a function and do not require an iterative solution. This is a significant improvement in time than that of the finite difference method of calculating the gradient. The adjoint method requires two times the CFD solution not the $(N \times M) + 1$ CFD solutions mentioned earlier.

From figure 9 it can be seen that the sensitivity suggests that the surface should be moved in the opposite direction to the normal, hence why there are negative magnitudes. Further to this the surface is symmetrical and as such the gradient in the x_1 direction is zero due to the sensitivities and the surface normals cancelling each other out.

Single Point Method

Implementation

The method was implemented in OpenFOAM as it contains a few pre-written libraries to aid in the implementation. The code was written to be able to find the closest point to any given point in space, this was due to maintaining flexibility within the code and ensuring that the code could be scaled efficiently to multiple dimensions.

Validation was undertaken through plotting the point on the surface and taking a visual inspection to ensure that the selected point on the surface was indeed the closest. Following this it was straightforward to extract the surface sensitivity at that point and convert it to the gradient required.

Results

From visually inspecting figure 9 it can be seen that the gradient at the closest point to the control point is the one generated in table 7. This shows promise that the code has been implemented correctly. However, as there is no experimental data to compare against the gradient information may be completely incorrect.

Multiple Point Method

Implementation

Again, the method was implemented in OpenFOAM due to its pre-written libraries. The code again was written in a flexible manner that allowed the convex hull of the spline to be identified for each control point. This was achieved by calculating the Euclidean distance to each control point and the convex hull was taken as the group of points that were closest to a specific control point.

Once the convex hull had been found calculating the average was straightforward. Both methods with equal weighting and weighting based on distance were implemented and utilised an entry from the command line to select between the method. The distance weighting method

was based on the inverse distance from the surface point to the control point, this enabled the closer points to impact the solution more than those points further away.

Results

It can be seen that these methods have an impact on the value of the gradients. This is due to the gradient gathering data from a larger range of points on the surface. The distance weighting gradient is closest to the single point method due to the points closer to the control point gathering more weight.

The equal weighting method reduces the value of the gradient because the gradients at the extremes which have lower sensitivities in the x_2 direction reduce the average of the gradient. However, the results from both methods are still similar and therefore provide more confidence that the gradient information are around the correct magnitude. One downside is that there is no experimental data to compare against and therefore it is difficult to validate the accuracy of the method.

Reverse Spline Method

Implementation

The implementation of this method was slightly different. A python script was written that calculated how much influence the control points had on the points on the surface. This was calculated as the spline was generated. This influence was calculated by the method of storing a percentage, in this case as 3 control points generate a new control point this leads to 33% influence on the new control point from each parent. Finally, the surface points each had a given percentage of influence from each control point.

Following this the method then utilises the distance weighted averaging algorithm written for the previous method to calculate the gradient at the control point.

Results

The proposed method generates similar gradients to the other two previous methods. This provides further confidence that the correct value for the gradient is being calculated with the lack of experimental data to compare against. It can be seen that the gradient values are similar to the convex hull method with the distance weighting. This is due to the influence of the control point ranging over a larger area than that of the convex hull.

Spline Point Gradient Method

Implementation

The implementation for generating the splines with altered control point values was written in python. Once these had been generated the distance the surface points would move was calculated utilising the distance from a point to a surface algorithm written for the single point method. The distances for each point was calculated and as such the matrix $\frac{\partial \boldsymbol{\beta}}{\partial \boldsymbol{x}}$ could be generated.

Following this the gradient with respect to the control point could be calculated by a simple matrix multiplication with the surface sensitivities.

Results

The results generated by this method are in similar agreement with those generated by the previous methods. As this method has a certain level of mathematical background, along with the fact that its value is similar to the previous methods, it provides a large amount of reassurance that the code along with methods are validated.

5.7. Significance of Work

There are significances from the work conducted. Some include efficiency, new methodologies and procedures, amongst others.

One of the main significances of the work is to increase the efficiency in finding global optima for solutions. It appears that CFD optimisation is in its relative infancy and therefore has not been investigated fully. The process developed being more efficient at finding the global optimum may drive the industry to draw from other areas to attempt to increase efficiency even more.

Further to this another significance of the work is the applicability of the method to a range of areas with minimal modifications. Therefore, industries could adopt the method and also drive it forward as there is a requirement in industry for efficient and cheap optimisers.

One other possible outcome from the work is that there could be the development of industrial optimisation test cases. There are currently validation test cases for the CFD code itself but none for the optimisation aspect. This work could lead to the development of a range of standard optimisation test cases that could be utilised to test the performance and accuracy of the methods developed.

Further to the points raised previously one other possible significance of the method is to show that there are other possible methods of optimisation that can be applied to CFD. This could potentially drive the industry into investigation of other possible methods that could be present in other fields and areas and adapt them for CFD.

5.8. Conclusion

Genetic algorithms and Bayesian optimisers have been shown to be able to calculate the minimum of various functions of a range of dimensions. They have been able to get within $1x10^{-6}$ error of the analytical minimum with a standard deviation of around $1x10^{-10}$. This shows that the implementations are correct and that they are able to find the global minima. It has also been shown that the Bayesian method provides greater accuracy than that of the genetic algorithm for a reduced number of function calls, which is beneficial for CFD where the function call may take anything up to a week or longer to complete.

A Gaussian process model has been written, which is utilised within the Bayesian optimiser, and has been shown to be able to fit a curve, a surrogate model, through a random set of points. This also provides the mean and standard deviation from the surrogate model which is utilised within the acquisition function within the Bayesian optimiser. Further to this the standard deviation of the function is zero at known points which is additional confirmation that the code has been implemented correctly.

For the development of the gradient information an implementation of the adjoint solver has been developed which solves for the adjoint variables from a converged solution. It has also been shown to provide similar results to those generated by an adjoint solver which solves for the adjoint variables simultaneously with the CFD flow variables. It has been difficult to validate the adjoint variables as they have no real meaning and are not generated in experiments.

Further to the adjoint solver the surface sensitivity has also been implemented. As with the adjoint variables, the surface sensitivity has been difficult to validate. However visual inspection of the results has been investigated and has produced results that would be expected, such as reducing sharp angles to reduce energy loss and therefore decrease pressure drop between inlet and outlet.

From the surface sensitivity, the gradient of the objective function has been calculated with respect to the control points of the spline. A range of possible methods have been used, all providing similar results.

5.9. Future Work

Gaussian processes generally do not take gradient information into account and as such there are no pre-written libraries for Gaussian processes that take in gradient information. Therefore, future work on the project will involve writing a Gaussian process model that will be able to take in gradient information to increase the accuracy of the prediction. Following this the method will have to be tested to ensure that the implementation is correct.

Further down the line once the Gaussian process that incorporates gradient information has been tested, all of the various aspects of the work conducted, such as the adjoint, surface sensitivities and gradient calculation methods can be integrated to work with the Gaussian process.

This again will have to be tested against the basic Bayesian method to ensure that the introduction of the gradient information does not impact the location of the global minima in the search space. One method to test the implementation is to compare the minima that both methods produce, as they should be identical. Ultimately the workflow developed could be applied to real world, industrially relevant geometries.

Furthermore, once the optimiser has been written one possible investigation that would be useful for industry would be to attempt to modify the optimiser so that it runs in parallel for the majority of its operation.

6. Project Management

6.1. Risk

Risk is active in every project. It describes anything that could have an impact on the projects outcome. Therefore it is important to consider risk throughout the project's duration. By considering risk the project would hopefully run smoothly.

One possible risk was that the concepts and understanding for the Bayesian optimiser and Gaussian process would take a little while to fully grasp. This could have had an impact on the project in that it would delay generating the code and therefore there may be a chance that the optimum would not be found.

One method to mitigate this was to seek a wide range of papers and descriptions of the method, with the view that a range of papers may describe the method in an understandable way. Further to this another method to mitigate the risk was to be in contact with post-Doctoral researchers who have implemented and worked with these methodologies before.

One other possible risk was that the implementation of the ideas and numerical aspects for the Bayesian and Gaussian processes would not be completed. This again would lead to there being no chance of optimising the geometry.

One method that mitigates this risk is that software development has been undertaken prior to the project and therefore many of the main ideas utilised in implementation should be already understood. Further to this there are others around the university who have implemented concepts before and they could be contacted for advice.

Again, as with the understanding of the Bayesian optimiser the understanding of the adjoint solver, surface sensitivities along with the gradients were a little more difficult. This was because the concepts were entirely new and a little more difficult to follow as they were purely theoretical concepts and had no diagrams explaining the process, unlike the Gaussian process and Bayesian optimisation methods.

To mitigate the risk follows similar procedures as to those of the Gaussian process and Bayesian optimisation methods, however the understanding required a meeting with a Ph.D. student who has derived the equations for the adjoint solver from scratch. It was decided that this, along with reading a large amount of papers on the subject, would be the best course of action.

Implementation of the adjoint solver along with the surface sensitivity and gradient calculation was a risk. This would not necessarily impact generation of the global minima however could impact the testing of it against the standard Bayesian method.

To mitigate this software development in both C++ and utilising OpenFOAM has been undertaken prior to the project however with simpler concepts. Mitigation of this risk also involves watching tutorials about how to implement a new solver and boundary conditions within OpenFOAM to aid development of the adjoint solver, surface sensitivity and gradient calculation.

A risk of the project is that there would be little to no time left for the implementation of the Gaussian process with gradients. This again is not an issue because the overall task for the project was to optimise the draft tube. One possible method to mitigate this is to ensure that all previous risks have been mitigated correctly and methods put in place to detect when aspects are taking too long.

6.2. Carbon Footprint

Although the carbon footprint is not directly applicable to the project, however the designs generated by the method could have their efficiency increased. This increase in efficiency of the product that the Bayesian optimiser method could generate would have a direct impact on the carbon footprint of those products.

One other benefit of the method is that as few CPU cycles are conducted in serial as possible. This means that the hardware is fully utilised. Resulting in a quicker simulation and as such lower carbon emissions from the running of the code.

With the development of an optimiser that requires reduced number expensive function calls mean that there is less time spent finding the optimum solution. This then, has the impact of running hardware for less time and being able to generate more solutions in the same period. Both of which save time and reduce the total carbon footprint.

6.3. Health and Safety

There are very few health and safety concerns with a research project such as this one. One main one being that as the majority of the work was conducted on a computer that the seating position and ergonomics of the work area comply with health and safety and ergonomic guidelines.

One other aspect that must be adhered to is that because a computer screen is being used on a regular basis that the user takes regular breaks from viewing the screen. This would reduce the risk of damaging the user's eyesight.

Another aspect of health and safety whilst utilising a computer is the risk of repetitive strain injury (RSI). This again would be avoided by proper ergonomics of the work area along with taking regular breaks from the computer.

It is difficult to prioritise the health and safety risks in this project as they are all relating to computer use. Therefore, it is difficult to say which activity would have the biggest impact on the user's health. Adhering to health and safety guidelines would mitigate all the possible health and safety risks so reducing the risk of one reduces the risk of them all.

6.4. Sustainability

Again, sustainability of code is not at all similar to the sustainability of the material for a product, for example. Code sustainability relates to a few areas. One being that the code itself is long lasting. Long lasting in the code sense is that it will be able to be used on a range of hardware for many years to come, which hopefully the concepts and code generated throughout the project will be.

One other aspect of sustainability that applies to programming is that the code must easily be added to. This is because new methodologies may be developed and to keep adding them to the code base must be easy, simple and require minimal effort. Keeping the code easy to add to and manage means that if there are bugs they would be more easily de-bugged.

One final aspect of code sustainability is the ease of reuse of the code. This is beneficial as the optimiser could, once written and validated, be applied to a range of problems, not necessarily CFD problems. The use of code reuse also reduces the number of bugs and therefore provides better software.

6.5. Project Management Procedures

There have been many methods for project management undertaken prior to and during the project. One of those being that a critical path analysis was undertaken. As this project was fairly linear there were few tasks that could be conducted in parallel to others. The main ones that could be undertaken in parallel were the understanding of the concepts.

One drawback with critical path analysis is that it does not show project milestones. One method to solve this is the use of a Gantt chart. One was created for the project prior to the undertaking of the project. This allowed timescales to be monitored and if tasks were taking longer than predicted then mitigation techniques could be implemented.

From the Gantt chart is was identified that the time taken to understand the mathematics of the adjoint solver was taking longer than expected. Once this was identified the mitigation for this was set up. Mitigation was undertaken, however it did have a negative impact on the project as it took up time for undertaking other aspects that relied on the adjoint solver. This then led to there being insufficient time to implement the Bayesian optimiser which took advantage of the gradient information.

In order to undertake the overall task at hand it was decided that the group should be split into three sub-groups. One team taking the experimental side of the project, one the CFD and one, finally, the optimisation. This was undertaken as it was argued that by splitting the group up

the teams could focus on their task at hand and this also was hypothesised to be more efficient over the method of everyone taking part in every aspect and as such drive progress.

To aid all members being up to date with each other's work weekly meetings were held. These had the impact of keeping the supervisor and group members up to date with the overall progress of the project and at what stage each group was at. It also enabled cross discussion between the groups.

Each member had their role in the meetings rotated, this gave every member the opportunity to be meeting chair and also hold the role of secretary. This has the benefit of giving the team a range of experience which can be transferred to industry application.

With regards to maintaining notes and ideas a notepad dedicated to the project was used. This allowed brainstorms and general ideas that could be elaborated at a later date to be maintained in an organised fashion. This also provided a journey map of what was undertaken and why. Within this notepad results, concepts and various ideas were noted which made the project flow better as notes were kept in order and together.

In order to maintain the code, git was used. This had the benefit of the code being backed up on external servers. Further to this there are added benefits for git, some being that code is stored in versions and by commit, and as such if the code no longer functions can be reverted to a state in which it was functioning. Additionally the branching of the project enables ideas to be tested without impact on the main functionality of the code.

A significant benefit of git is that it allows the creation of private repositories. This provides full control over intellectual property by the author, managing who is granted access to the code and methodology meaning that monetisation of the final product could be achieved also.

6.6. Ethics

There are few ethics associated with the project. One could be that the optimiser is utilised for military purposes. Some people may be against this but as the authors have control over who has access to the code it can be withheld from such organisations and industries.

7. Contribution to Group Functioning

For the group to function efficiently and as the group was split up into three sub-groups collaboration between individuals in the sub-groups and also between the sub-groups was necessary. Some examples of what interfacing and contributions are given below.

Members of the CFD group provided CFD solutions for the research undertaken in this project to be developed. For example, the adjoint solver requires a fully converged CFD solution. As the adjoint solver was being implemented members of the CFD team were generating the CFD solution for a range of cases. These were then supplied for the adjoint solver to be tested

against. As these were provided it enabled the validation of the adjoint solver, which further on in the project would come together to generate a whole work flow.

Prior to the running of CFD the CFD group required a mesh. As they had not generated many meshes of industry standard prior to the undertaking of the task, whereas I have had experience of industrial meshing, I was able to give them advice as to the best approaches. I was also able to suggest improvements to areas within their meshes that could cause the CFD solution to become unstable or unphysical.

Further to this I have had experience of programming prior to the project and therefore advice was able to be given to those generating the meshes to write automated scripts that would read in the geometry and generate high quality meshes.

As the CFD group required some help with initial and boundary conditions, with my knowledge and experience of CFD I was able to advise on the best initial and boundary conditions for each turbulence model. When simulations diverged I was able to give advice as to possible reasons why. For example, advice was given to change the turbulence model to a low Reynolds k-OmegaSST model from the high Reynolds k-Epsilon model due to its superior performance for separated flow and wall bounded flows.

Additionally I was able to advise the experimental team as to the best places to provide pressure tappings which was based on analysing the results of the CFD with the CFD team. Further to this the experimental team also provided constraints for the geometry that would feed into the optimiser based on the largest dimensions that their 3D printer could produce which then would influence the code that I was writing for the optimisation.

Throughout the project various discussions between each sub-group were conducted which I took part an active in along with being the holding meeting secretary and chairman roles that were cycled throughout the group. Tasks that were undertaken included brainstorming possible avenues of research and discussing the benefits and drawbacks of each method. These involved various Ph.D students, post-Doctoral researchers and supervisors.

As I had initially written a basic Bayesian optimiser before moving on to more complex ideas, others within the optimisation group were finding it difficult to grasp the ideas and how to implement them. I was therefore able to share the code with the group and explain and debate the best way to implement such a method. Further to this, other members in the optimisation group provided feedback on the quality and accuracy of the code that I had written. This led to the discovery of a few minor bugs, the fixing of which made the code more efficient and also made the code more adaptable for a range of problems.

8. References

[1] H. K. Versteeg and W. Malalasekera, An Introduction to Computational Fluid Dynamics The Finite Volume Method, 2nd Edition ed., Essex: Pearson, 2007.

[2] I. Rechenberg, "Cybernetic Solution Path of an Experimental Problem. Ministry of Aviation," Royal Aircraft Establishment, UK, 1965.

[3] M. Mitchell, An Introduction To Genetic Algorithms, 5th Edition ed., Cambridge: The MIT Press, 1999.

[4] R. Hilbert, et al, "Multi-Objective Shape Optimization of a Heat Exchanger Using Parallel Genetic Algorithms," *International Journal of Heat and Mass Transfer,* vol. 49, p. 2567–2577, 2006.

[5] J. Mockus, V. Tiesis and A. Zilinskas, "The Application of Bayesian Methods for Seeking the Extremum," *Towards Global Optimization,* p. 117–129, 1978.

[6] J. Snoek, H. Larochelle and R. Adams, "Practical Bayesian Optimization of Machine Learning Algorithms," in *Advances in Neural Information Processing Systems 25*, 2012.

[7] B. Shahriari, et al, "Taking the Human Out of the Loop: A Review of Bayesian Optimization," *Proceedings of the IEEE,* vol. 104, pp. 148-175, 2016.

[8] D. Jones, M. Schonlau and W. Welch, "Efficient Global Optimization of Expensive Black-Box Functions," *Journal of Global Optimization,* vol. 13, p. 455–492, 1998.

[9] C. E. Rasmussen and C. K. Williams, Gaussian Processes for Machine Learning, Massachusetts: MIT Press, 2006.

[10] E. Solak, et al, "Derivative Observations in Gaussian Process Models of Dynamic Systems".

[11] J. Wu, et al, "Bayesian Optimization with Gradients".

[12] J. McNabb, et al, "CFD Based Draft Tube Hydraulic Design Optimization," in *IOP Conference Series: Earth and Environmental Science*, 2014.

[13] T. Ciocan, R. Susan-Resiga and S. Muntean, "Improving Draft Tube Hydrodynamics Over a Wide Operating Range," in *Proceedings of the Romanian Academy, Series A*, 2014.

[14] J. B. Sosa, et al, "Computational Fluid Dynamics Simulation and Geometric Design of Hydraulic Turbine Draft Tube," *Advances in Mechanical Engineering,* vol. 7, 2015.

[15] M. B. Giles and N. A. Pierce, "An Introduction to the Adjoint Approach to Design," *Flow, Turbulence and Combustion,* vol. 65, p. 393–415, 2000.

[16] A. Stück, Adjoint Navier–Stokes Methods for Hydrodynamic Shape Optimisation, Hamburg: TUHH, 2011.

[17] A. Tzanakis, "Duct Optimization Using CFD Software 'ANSYS Fluent Adjoint Solver'," Chalmers University, Goteborg, 2014.

[18] D. A. Boger, "A Continuous Adjoint Approach to Design Optimization in Multiphase Flow," Pennsylvania State University, 2013.

[19] G. K. Karpouzaz, et al, "Adjoint Optimisation for Vehicle External Aerodynamics," *International Journal of Automotive Engieneering,* vol. 7, pp. 1-7, 2016.

[20] M. Kuss, "Gaussian Process Models for Robust Regression, Classification, and Reinforcement Learning," Technischen Universitat Berlin, Berlin, 2006.

[21] A. G. Wilson and R. P. Adam, "Gaussian Process Kernels for Pattern Discovery and Extrapolation," Cornell University Library, 2013.

[22] M. Abramowitz and I. A. Stegun, "Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables," Dover, 1965.

[23] O. Pironneau, "On Optimum Design in Fluid Mechanics," *Journal of Fluid Mechanics,* no. 64, pp. 97-110, 1974.

[24] A. Jameson, "Aerodynamic Design via Control Theory," *Journal of Scientific Computing,* no. 3, 1988.

[25] A. Jameson, "Optimum Aerodynamic Design Using CFD and Control Theory," in *AIAA95-1729-CP*, 1995.

[26] J. Elliott, "Aerodynamic Optimization Based on the Euler and Navier–Stokes Equations Using Unstructured Grids," 1998.

[27] C. Othmer, "A Continuous Adjoint Formulation for the Computation of Topological and Surface Sensitivities of Ducted Flows," *International Journal for Numerical Methods in Fluids,* vol. 58, pp. 861-877, 2008.