An Adaptive Parameterisation Method for Shape Optimisation Using Adjoint Sensitivities

by

Rejish Jesudasan

Supervisor: Dr. Jens-Dominik Müeller

A thesis submitted in partial fulfillment of the requirements of the Degree of Doctor of Philosophy

School of Engineering and Material Science

Queen Mary University of London

Acknowledgements

First of all, I would like to thank my supervisor, Dr. Jens-Dominik Müeller, for giving me the opportunity to conduct this research work under his guidance. This thesis would not have been possible without his help. He has provided me with endless helpful advice to overcome all difficulties throughout the years.

I would also like to thank present colleagues at the CFD optimisation group. The supports I received from Susan Barker, Mateusz Gugala, Oluwadamilare Rahman Imam-Lawal, Xingchen Zhang and Orest Mykhaskiv are highly appreciated you have all enriched my university experience in various ways

I would like to thank my better half for her love, support and constant encouragement in difficult times over the past few years.

I especially acknowledge my mother's great contributions to my life who dedicated all her life for the betterment of her two sons.

This research would not have been possible without the financial assistance from QMUL-IODA project. This work is a part of the Industrial Optimal Design using Adjoint CFD (IODA) project and received funding from the European Union's Seventh Framework Programme (H2020) for research, technological development and demonstration under grant agreement no. 642959.

Statement of Originality

I, Rejish Jesudasan, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

The copyright of this thesis rests with the author and no quotation from it or information derived from it may be published without the prior written consent of the author.

Signature: Rejish Jesudasan
Date: December 2, 2019

Abstract

Adjoint methods are the most efficient approach to compute the design sensitivities as the entire gradient vector of a single objective function is obtained in a single adjoint system solve. This in turn opens up a wide range of possibilities to parameterise the shape. Most shape parameterisation methods require manual set-up which typically results in a restricted design space. In this work, two parameterisation methods that can be derived automatically from existing information are extended to include adaptive design space in shape optimisation.

The node-based method derives parameterisation directly from the computational mesh employed for simulation and normal displacements of the surface grid nodes are taken as design variables. This method offers the richest design space for shape optimisation. However, this method requires an additional surface regularization method to annihilate high-frequency shape modes. Hence the best achievable design depends on the amount of smoothing applied on the design surface. An improved adaptive explicit surface regularization method is proposed in this thesis to capture superior shape modes in the design process.

The NSPCC approach takes CAD descriptions as input and perturbs the control points of the NURBS boundary representation to modify the shape. The adaptive NSPCC method is proposed where the optimisation begins with a coarser design space and adapts to finer parameterisation during the design process. Driven by adjoint sensitivity information the control points on the design surfaces are adaptively enriched using knot insertion algorithm without modifying the shape. Both parameterisation methods are coupled in the adjoint-based shape optimisation process to reduce the total pressure loss of a turbine blade internal cooling channel. Based on analyses regarding the quality of the optima and the rate of convergence of the design process the adaptive NSPCC method outperforms both adaptive node-based and the static NSPCC approach.

Contents

List of Figures

List of Tables

Nomenclature

ACRONYM

BFGS	Broyden–Fletcher–Goldfarb–Shanno
CAD	Computer Aided Design
DMFFD	Direct Manipulated FFD
FFD	Free-Form Deformation
NSPCC	NURBS-based Parameterisation with Complex Constraints
NURBS	Non-Uniform Rational B-Splines
PARSEC	Parametric Section
RBF	Radial Basis Function

Greek

α	Design variables
β	Smoothing coefficient
γ	Step length

Roman

$\bar{s}, \bar{t}$	knots in a knot vector S and T respectively
$\mathbf{d}$	descent direction
$\delta\mathbf{P}$	Perturbations to control points

δX_s	Displacements of surface mesh coordinates
$\delta \widetilde{X}_s$	Smoothed surface displacements
$\widetilde{\mathcal{G}}$	Smoothed CFD sensitivity
$\mathcal{B}$	B-spline basis functions
$\mathcal{G}$	CFD sensitivity: "raw"gradient of the objective function with respect to the surface mesh nodes
$\mathcal{M}$	Baseline CFD mesh
$\mathcal{M}'$	Updated CFD mesh
$\mathbf{K}$	Filter kernel
$\mathbf{P}$	Control points
$\mathbf{P}_{\mathrm{G}}$	Geometric parameters in PARSEC parameterisation method
$\mathbf{V}$	Base vector with noise
$\widetilde{\mathbf{V}}$	Smoothed vector
d	deformation function
J	Objective function
N	Total number of design variables
n	Number of smoothing iteration
p, q	degree of a NURBS surface in s and t parameter direction respectively
q	total number of pilot points
R	Non-linear flow equations
r	filter radius
S, T	knot vector of the NURBS surface along s and t parameter direction respectively

s, t	parameters of a NURBS surface
S_n	Unit surface normal
U	Vector of state variables
w	weights of the NURBS surface
X_s	Surface mesh
X_s'	Updated surface mesh
X_v	Volume mesh
D_h	Hydraulic diameter of the VKI U-Bend
$\mathcal{U}$	Smoothing operator

Chapter 1

Introduction

1.1 Background

Simulation-based shape optimisation receives great attention in aerospace [56, 74], automotive [142, 129], marine [126] and civil industries. In the past, design methodologies were centered around expensive experiments. For example, building a prototype and evaluate the performance using wind tunnels and then modify the design based on the knowledge gained and restart the design. This process should be repeated until the final satisfactory design is obtained for a given design condition. Building and maintaining such experimental facilities are always time consuming and expensive.

Advances in both Computer Aided Design (CAD) systems and numerical simulation methods have reduced the need for experimental facilities. A CAD model can be created and its performance can be computed using numerical-based simulation methods such as Computational Fluid Dynamics (CFD), Computational Structural Mechanics (CSM) etc. This is also an iterative process and it normally takes a few hours to few days which is less time-consuming and cost effective. However designer should redesign or update a CAD model repeatedly based on the knowledge gained from previous simulations. Major disadvantage of this approach is that, the final design highly depends on the designer's experience and to reduce design complexities designers often include only a few design variables in the design process. Hence important shape modes may not be explored in the design process. A typical design work flow with the designer in the loop is shown in Fig. 1.1

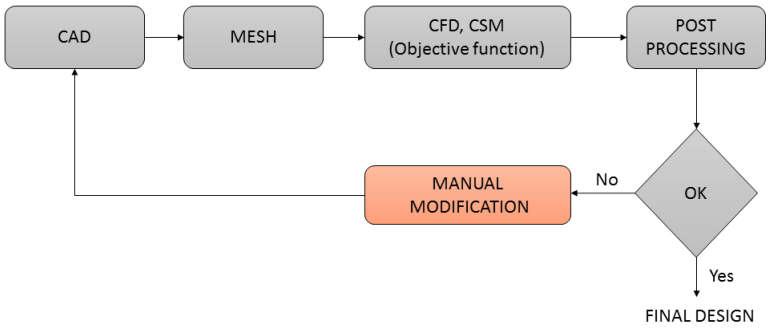


Figure 1.1: Simulation-based manual design process

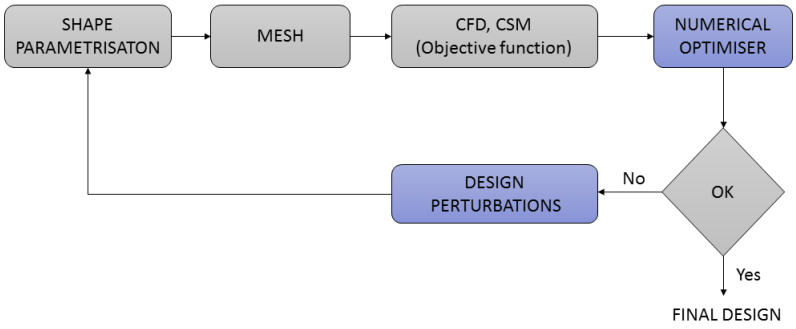


Figure 1.2: Simulation-based design process with a numerical optimiser

1.1.1 Numerical optimisation

Engineering design process often required automatic design loop with a large of number of design variables. Typical design loop with numerical optimisation algorithm is shown in Fig. 1.2. As a first step, designer should parameterise the shape using a set of design variables which determines the shape modes during the design process. After performance evaluation, the numerical optimiser determines the updated set of design variables that can generate an updated shape with better performance. Then the simulation process continues with the updated shape. This is an iterative process and the design loop terminates when optimisation converges. Using a numerical optimiser, one can systematically explore all possible shape modes determined by the shape parameterisation method employed in the design process with minimum user intervention.

Early studies used gradient-free approaches to solve shape optimisation problems. Gradient-free methods require only an objective function value and they are well suited to handle multi-objective, multi-modal, non-differentiable objective

functions [34]. These methods can be incorporated with any commercially available solvers without having access to its source code. However, the major disadvantage is that they are slow to converge and may require a large number of performance evaluations [33]. If costly evaluation models are used in the design loop then the designer is reluctant to use gradient-free optimisation techniques for shape optimisation. Alternatively, one can use surrogate evaluation models in the design loop to reduce optimisation turn around time [46]. However computational cost for performing design of experiments also increases heavily if a large number of design variables are used [40]. Hence gradient-free methods are suitable only to handle coarser design spaces with only a few design variables. Some examples of gradient-free methods are Non-dominated Sorting Genetic Algorithm (NSGA-II), particle swarm optimisation and simulated annealing. A comprehensive review of recently introduced deterministic based optimisation methods can be found in [138].

On the other hand, gradient-based methods are very efficient if the objective function is differentiable and can handle a large number of design variables with less evaluations and hence they are preferred. However, gradient-based methods require gradient of the objective function with respect to each design variable to drive the design process and the computed gradients must be accurate. This is because, inaccurate gradient information may affect the design path taken by an optimiser and may require a large number of design iterations to converge. In addition, they often converge to local optimum and cost of computing the gradients becomes huge if a large number of design variables are used. Therefore it is very important to compute accurate gradients in an efficient manner [99].

1.1.2 Adjoint methods

Generally, gradients can be computed by any one of the following methods namely finite-difference based approach, complex-step derivative approximation [79], tangent linearisation and adjoint-based methods [78]. Finite-difference method (FD) can be implemented easily with any black-box commercial solvers without having access with its source code. However, accuracy of the method is limited to the chosen step width. It is clearly evident that, a range of step width must be used to achieve minimum error and further, this minimum error does not occur at a fixed step size for

all the design variables. Unlike FD, complex-step derivative approximation method does not involve a difference operation hence not subjected to subtractive cancellation error. However, as similar to FD choice of step width is not obvious, hence not suitable. Tangent linearisation computes exact gradient, however the computational cost for gradient computation is scaled with the number of design variables. This is also applicable to finite difference and complex step derivative method. Hence these methods are not suitable for gradient-based shape optimisation process.

Of particular interest is the adjoint method which can compute the exact gradients at a cost that is essentially independent of the number of design variables. Hence adjoint methods are essential when a large number of design variables are used in the design process. Over the last decades, adjoint methods have been successfully applied to optimise various turbomachinery [134, 72, 143], aerodynamic [74, 75] and automotive components [142, 129]. There exists two different approaches to derive the adjoint gradient. They are a) continuous approach and b) discrete approach. In the continuous adjoint approach, adjoint of the governing partial differential equations are derived analytically and then discretised. This approach involves lengthy hand derivation which may be error prone. On the other hand, in the discrete adjoint method, the governing equations are discretised and then the discrete adjoint equations are formulated. In contrast to the continuous adjoint approach, Automatic Differentiation (AD) tools can be used for developing discrete adjoint solver which reduces the solver development and maintenance cost.

Automatic Differentiation (AD) software tools allow to compute the exact derivatives for complex algorithms by differentiating the statements of a computer program that executes the algorithm. Hence exact gradients can be computed with minimal effort. In addition to that, gradients computed using discrete adjoints can be verified using tangent linearisation approach which is not available in the continuous adjoint approach. Hence gradients via discrete adjoints received more attention and in the present work discrete adjoint solver named STAMPS [93] developed by the CFD optimisation group at QMUL is used to compute the gradient of the objective function with respect to the surface mesh nodes.

1.1.3 Shape parameterisation

Choice of shape parameterisation is crucial which determines the set of shape modes that can be captured during the design process [112]. Since the best achievable design belongs to this set, shape parameterisation method influence the final solution and the rate of convergence of the optimisation process. Adjoint methods do not penalize the size of the design space hence one can consider the displacement of every surface grid node as a design variable [64, 57, 27], called as the node-based method. This method offers richest design space for shape optimisation. However this design space is very rich and contains unwanted oscillatory modes which needs to be filtered by the use of surface regularisation method. Additional surface regularisation is necessary, and implicit [62] as well as explicit [64] Sobolev smoothing methods have been proposed. However, both of them require a smoothing coefficient and/or number of smoothing iterations to control the amount of smoothing applied on the shape. In standard node-based parameterisation, the value of this coefficient or the number of smoothing iterations for explicit is chosen by the user a-priori and remain fixed throughout the optimisation. This choice is a case dependent and strongly influence the design space for shape optimisation. For example, larger value corresponds to over smoothing which suppress the generation of superior designs [70] and smaller value leads to the generation of highly oscillatory shape modes.

This method is coming under the category of CAD-free parameterisation methods, while the baseline mesh is derived from a CAD model, this model is not included in the optimisation loop and hence the resulting optimal shape is not available in a CAD format and a 'return to CAD' step needs to be added to make this shape available for further design, analysis and/or manufacturing. This step typically requires an automatic post-processing tools to approximate the optimised shape using a collection of NURBS patches.

CAD-based parameterisation approaches, have been typically employed in both gradient-based and gradient-free optimisation methods. In these methods, shape parameterisation is defined in a CAD model which is included in the design loop. The optimisation hence produces a consistent CAD model of the optimal shape. Therefore the optimised CAD model can be used for further multi-disciplinary analysis and manufacturing.

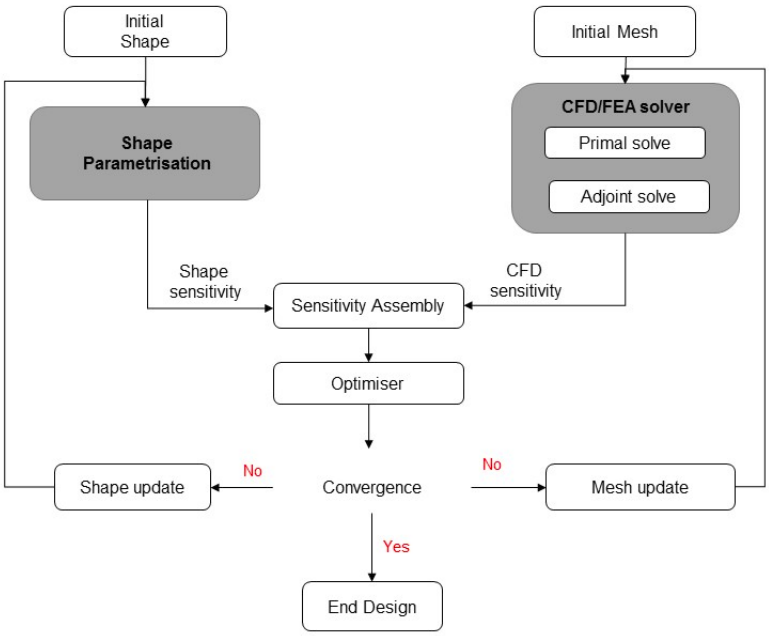


Figure 1.3: Gradient-based shape optimisation loop

However in a gradient-based shape optimisation loop, sensitivity of the shape with respect to each design variables (shape derivative) is essential to compute the gradient of the objective function with respect to the design variables. A typical gradient-based shape optimisation loop is shown in Fig 1.3. CAD-free approaches can be differentiated hence can be easily coupled with the gradient-based optimiser. However when coupled CAD in the loop one must differentiate the CAD-model. Commercial CAD systems do not provide shape derivatives hence often relied on finite difference method. As mentioned earlier, the choice of step width is crucial and the presence of truncation and subtractive cancellation errors restrict the use of finite difference in the design loop [108, 4].

If the source code of the CAD kernel is available, e.g. as in the case of open-source CAD engine OpenCASCADE, shape sensitivities can be obtained by applying Automatic Differentiation (AD) Software tools to the complete CAD kernel. This is a object-oriented library written using C++ programming language with more than 10,000 classes hence relied on operator-overloading based AD tools such as ADOL-C which is straightforward to implement. However, for a complex program with a large computational graph often leads to excessive run-time and memory footprints[19].

Alternatively, researchers from the CFD optimisation group at Queen Mary Uni-

versity of London developed 'NURBS-based Parameterisation with Complex Constraints (NSPCC)' method that derives parameterisation directly from the boundary representation (BRep) of a CAD-model. The BRep, in the typical standardised STEP format, represents the shape using a number of NURBS patches. NSPCC approach uses control points of the NURBS patches to deform shape in the design process and has been tested in wide range of applications which includes automotive [142], aerospace [150] and turbomachinery [65]. Since the shape parameterisation is derived from the BRep of a CAD model, construction history of a CAD model is not requirred hence only subset of functionalities can be added. The implementation is done in Fortran language hence the efficient derivative code is obtained using source transformation AD tool TAPENADE.

1.2 Motivation

Most shape parameterisation methods require manual setup [112, 145]. Setting up auxiliary grids for lattice-based methods, such as e.g. auxiliary grids with Hicks-Henne bumps on airfoils [56] or stacked spline curves for turbomachinery blades [134], involve substantial effort and are difficult to extend to complex geometries. To reduce complexities, the designer often ends up with choosing small number of design variables and the design space that capture all the possible shape modes remain fixed throughout the optimisation. This traditional static parameterisation approach restricts the generation of superior designs outside the fixed envelope and the final solution highly depends on how the designer parameterise the baseline shape. To obtain superior designs, the designer should terminate the design process and reparameterise the shape manually in a periodic manner [139, 90, 116]. Even though design acceleration can be achieved, still incur the limitation of a manually defined nested level of shape parameterisation. To avoid designer in the loop, the additional design variables need to be added automatically only if necessary.

Hradil et al. [59] proposed an adaptive parameterisation method based on FFD approach and Masters et al [83] presented an adaptive subdivision surfaces for shape optimisation. The results also show design acceleration however optimised geometry is not available in CAD format for further analysis and manufacturing. Agarwal et

al. [5] presented a CAD-based adaptive parameterisation method by adding multiple CAD features to a parametric CAD model. Results showed that the inserted CAD features are not good enough to capture superior designs hence it affects the rate of convergence and leads to a sub-optimal solution. This strongly justifies the need for adding more design variables that can capture important shape modes for design improvement.

NURBS have become the de-facto industry standard for data exchange between CAD systems and offers local shape modification property. NSPCC method uses control points of the NURBS patches to deform shape in the design process. Hence NURBS manipulations algorithms such as knot insertion algorithm and degree elevation can be used to enrich control points on the design surface. In this work, the NSPCC method has been extended to handle adaptive design space in the shape optimisation. The use of NURBS-based adaptive parameterisation method is yet to be studied and developed in depth. To author's knowledge no work has been done to refine control points of the NURBS patches in shape optimisation.

1.3 Thesis Contributions

In the optimisation loop, the best achievable design is strongly influenced by the number of degrees of freedom and their distribution in the shape parameterisation. A design space with a small number of design variables places the burden on the user to express all relevant shape modes to achieve an optimal result hence often end up with a suboptimal solution [124, 113, 137, 65, 133]. On the other hand, a design space with a larger number of design variables may capture superior designs however places burden on the optimisation algorithm to accommodate problems with higher degrees of freedom such as multiple local minima and the rate of convergence of the optimisation [133, 74, 77, 80, 38]. Therefore the design loop should be able include adaptive design space where the low-frequency shape modes are handled in early stages of the design to accelerate the design process and once the design variables offered negligible design improvement then high sensitivity regions needs to be identified to refine parameterisation where it is necessary without modifying the shape. Moreover, a consistent CAD model should be preserved in the design loop

for further multi-disciplinary analysis and/or manufacturing. Furthermore accurate shape derivatives need to be computed.

Two parameterisation methods are considered here, node-based and NSPCC. The objective of this thesis is to extend the functionalities to handle adaptive design space in shape optimisation. The following list presents the major contributions to achieve this goal:

1. Adaptive NSPCC parameterisation method using knot insertion algorithm is proposed to refine the control net distributions without modifying the geometry.

2. The adaptive refinement is driven by node-based sensitivity information, therefore the control points are added only in the region where larger design improvement can be achieved when the optimiser has reached sufficient convergence. As a consequence the adaptive NSPCC replaces user in the design loop as design variables are automatically created based on the adaptive refinement.

3. Investigated the flexibility and efficiency of the Adaptive NSPCC method with the adaptive node-based method.

4. The entire design chain is reverse differentiated using source transformation Algorithmic Differentiation tool which is essential for handling large number of design variable in the design process.

5. The use of the proposed parameterisation methods in an industrial design chain is evaluated using one-shot optimisation strategy

1.4 Organization of Thesis

The remaining of the thesis is organized as follows:

Chapter 2 presents a brief survey on the most important geometry parameterisation methods and its important requirements for shape optimisation.

In Chapter 3, details about STAMPS a finite-volume primal and discrete adjoint solver, sensitivity assembly for the entire design chain are presented.

Chapter 4 describes the proposed Adaptive NSPCC parameterisation method and algorithm.

In Chapter 5, NSPCC method for handling assembly constraints is presented. In addition, shape optimisation tools developed in this work are compared with the currently available open-source SU2 shape optimisation tools.

Comparison of shape optimisation results obtained using the proposed parameterisation methods are presented in Chapter 6.

Chapter 7 concludes with the summary of the thesis contributions and recommendations for future work.

Chapter 2

Literature Review: Shape Parameterisation

2.1 Introduction

In this chapter, the most useful shape parameterisation approaches are presented. Shape parameterisation has been one of the main areas of research in shape optimisation and over the years a wide range of parameterisation methods have been developed [112, 60, 122] to list the most important ones: Node-based parameterisation, Free-Form Deformation (FFD), Radial Basis Function morphing (RBF), Parametric Section (PARSEC), Hicks and Hene bump functions and CAD-based parameterisation. Samareh [112] provides a brief survey on geometry parameterisation methods, Sobester et al. [123] provides a detailed review on parameterisation methods suitable for aircraft design and optimisation.

One can classify the parameterisation methods in different ways. Among the available methods, in this work, the shape parameterisation methods are broadly classified into two categories. The first one is the CAD-free parameterisation that doesn't include the CAD description in the design loop. Even though the baseline mesh is derived from a CAD model, in CAD-free methods, a CAD- model is not included in the design loop hence not updated. Therefore additional automatic tool for 'return to CAD' step is essential for the parameterisation methods that doesn't include CAD in the loop [43, 32, 68]. Methods coming under this category is termed as CAD-free parameterisation. Some researchers referred this category as

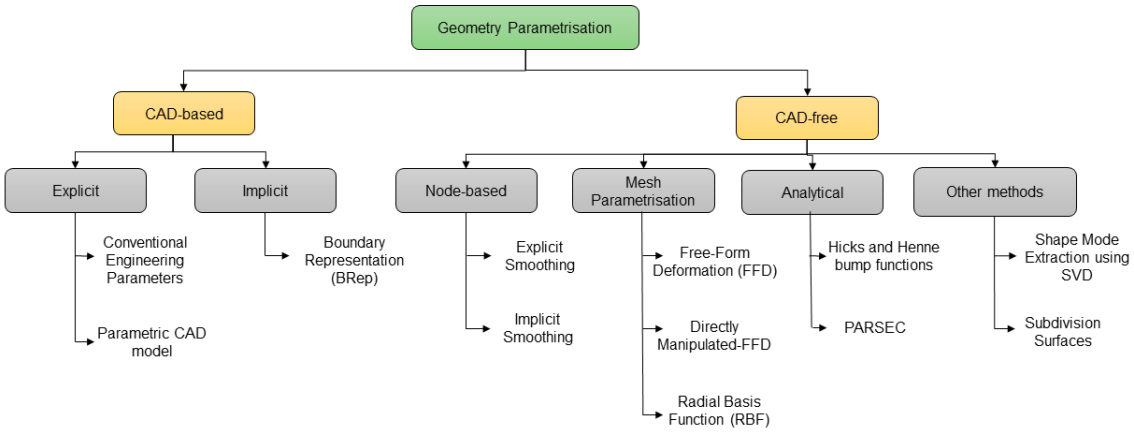


Figure 2.1: Classification of geometry parameterisation methods

'Mesh-based' parameterisation methods [143]. After meshing there is no connection between a discrete geometry and a CAD model. Hence design variables are not related to a CAD model

The second one is the CAD-based methods which preserves the CAD description in the design loop hence one can export the optimised shape in CAD form. CAD-based methods are essential for multi-disciplinary shape optimisation and analysis. When employed CAD in the loop, one must differentiate a CAD kernel to obtain exact shape sensitivities. CAD-free methods can be differentiated however constraint imposition is a tedious process for complex geometry. Based on this idea, in this work parameterisation methods are classified as CAD-free and CAD-based methods. In this chapter, some of the most important methods coming under each category will be briefly discussed. Figure 2.1 shows the taxonomy of the shape parameterisation methods discussed in this survey.

2.2 CAD-free Parameterisation

In this method, shape parameterisation is done by using a computational mesh used without the use of any external CAD-geometry description. Even though the baseline mesh is derived from a CAD model, this model is not included in the design loop and hence not updated. Given a design perturbations δX_s between the initial and the updated shape, the computational mesh $\mathcal{M}$ can then be updated as,

$$X_s' = X_s + \delta X_s \tag{2.1}$$

$$\delta X_s = \gamma \mathbf{d} \tag{2.2}$$

where X_s and X_s' are vector of surface mesh coordinates corresponds to the initial mesh $\mathcal{M}$ and the updated mesh $\mathcal{M}'$ respectively, δX_s represents the displacements of the surface mesh coordinates. $\mathbf{d}$ and $\gamma > 0$ are the descent direction and the step length respectively which are computed by the optimiser. Based on the type of deformation method employed in the design loop, CAD-free parameterisation can be further divided into three types. First one is the node-based parameterisation method which employs surface grid nodes as degrees of freedom in the design loop. The second one is the mesh parameterisation method in which the computational mesh is parameterised using a set of control points or lattice which are used to deform the design surface in the design loop. Third type is the analytical approach.

2.2.1 Node-based parameterisation

The node-based parameterisation is the most noticeable approach in which the deformation field is defined exactly on the same discrete space as of the geometry. In other words, this method employs each node on the design surface as degrees of freedom [104, 61, 87, 27, 127, 68, 57, 21, 20]. For example, if a wing surface is discretised by N grid points, then this method has N design variables defining the design space. This represent the highest possible design space offered by the computational mesh. Therefore in this work, the term rich design space is used to indicate a design space that contains a large number of design variables. A typical representation of design nodes used for shape optimisation ofthe VKI U-Bend geometry is shown in Figure 2.2. The main advantages of node-based parameterisation approach are:

- It provides rich design space for shape optimisation hence very useful for design space exploration.
- It can be set up automatically hence does not require time-consuming shape parameterisation preprocess.
- The same computational mesh is used to discretise both flow and shape thus any complex shape can be used for shape optimisation.
- Eliminates the use of CAD model in the design loop hence additional CAD

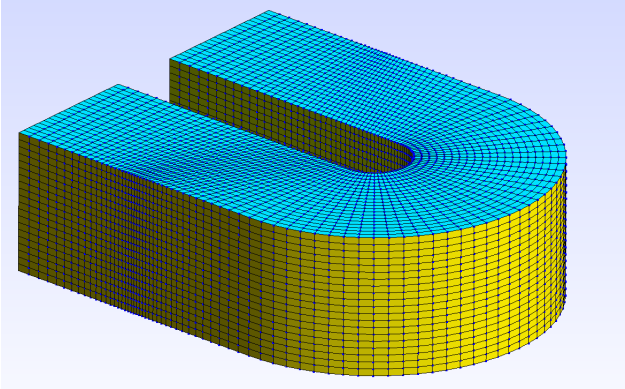


Figure 2.2: 3D segment of U-Bend geometry with the design nodes

processing tools and data interfaces involved in the optimisation are not required.

Need for Surface Regularisation

Unfortunately, node-based method includes odd-even oscillatory shape modes in the design space. In most aerodynamic cases it is very important to have smooth geometry to avoid bumps along the boundary to maintain attached flow, avoid boundary layer separation and turbulent flow which are all leads to increased drag and a loss of performance[17]. In addition to that, the presence of small scale oscillations may lead to impractical and non-manufacturable shapes. In general, a shape can be represented as superposition of several basis functions each characterized by their wavelength. For example, a wing shape can be represented as superposition of sin and cosine waves of different amplitudes and frequencies. In here, different frequencies can be referred as shape modes. For example, if the distance between a surface crest is large then it has long wavelength hence it can be viewed as a low frequency shape mode (slow changing surface). On the other hand, if the distance is small then it has short wavelength hence it can be viewed as a high frequency shape mode (fast changing surface). Therefore additional surface regularisation methods need to be incorporated to remove small scale oscillations while preserving desirable and important shape modes in the design process [127, 57].

Surface regularisation is a process by which surface node points are smoothed or averaged with their neighbors. This usually has the effect of damping the high-frequency shape modes in the design process. In simplest form, smoothing can be

written as,

$$\widetilde{V}^{n+1} = V^n + \beta\, \mathcal{U}(V) \tag{2.3}$$

where $V = [v_1^T, v_2^T ...., v_N^T]$,is the input vector with small scale oscillations on the design surface, N is the total number of surface grid nodes and $\widetilde{V}^{n+1}$ is the smoothed vector. $\mathcal{U}(V) = [\delta v_1^T, \delta v_2^T ...., \delta v_N^T]$ is the vector of modifications to remove small scale oscillations. $\mathcal{U}$ is the smoothing operator, β is the smoothing coefficient and and n is the number of smoothing iterations both controls the smoothing intensity. In the design loop, surface regularisation can be applied either to smooth the computed gradients [63, 68] or smooth the shape directly [64, 58]. In the former type, gradients are smoothed before fed into the optimiser then V and $\widetilde{V}$ in Eqn. 2.3 becomes $\mathcal{G}$ which is the raw gradient and $\widetilde{\mathcal{G}}$ which is the smoothed gradient respectively. In the latter method, design perturbations are smoothed before updating the surface mesh nodes. In this case V and $\widetilde{V}$ in Eqn. 2.3 becomes δX_s which is the input perturbations and $\delta \widetilde{X}_s$ is the smoothed perturbations respectively.

Sobolev gradient method has been widely used in many practical applications [61, 62, 54, 63, 114]. Sobolev gradient method is a Laplacian or diffusion type smoothing method in which the smoothed gradient field $\widetilde{\mathcal{G}}$ is obtained from initial gradient $\mathcal{G}$ by solving diffusion equation implicitly:

$$\widetilde{\mathcal{G}} - \beta \nabla^2 \widetilde{\mathcal{G}} = \mathcal{G}, \tag{2.4}$$

where β is a tuning factor which controls the intensity of the smoothing.

Image processing based convolution filters are also used as a design tool to filter out high-frequency modes in the gradient fields. It has been widely used in topology optimisation [120, 121], structural shape optimisation [32, 14] and CFD-based shape optimisation [127, 58]. The basic idea is that small scale oscillations in the gradient fields which are smaller than the kernel length scale or filter radius will be damped out. The kernel for gradient smoothing defines the shape of the function that is used to take the average of the neighbouring gradients. In the discrete form, filtered gradient can be written as,

$$\widetilde{\mathcal{G}}_i = \frac{\sum_{l=1}^{d} \mathbf{K}_l \cdot \mathcal{G}_l}{\sum_{i=1}^{d} \mathbf{K}_l} \tag{2.5}$$

where $\mathbf{K}$ is the filter kernel, d is the number of nodal points in the filter domain

including the center point i. Sigmund et al. [120, 121] defined the kernel function based on a geometric distance between nodal points. Daoud et al. [32] extended this distance function to include both topological and geometrical distance to avoid inconsistent in evaluating the distance between node points on the curved design surface. Stück and Rung et al. [127] used uniform Gaussian kernel for CFD-based shape optimisation,

$$\mathbf{K}(r) = \frac{1}{4\pi\beta} \exp\left(-\frac{r^2}{4\beta}\right) \tag{2.6}$$

where r is the local filter radius and β is the factor controlling smoothing intensity as involved in Eqn. 2.4 which is given as half the variance $\sigma^2 = \gamma t$ of the Gaussian filter kernel. Authors applied this method to the shape optimisation of $3D$ double bend duct with three different kernel width $\sigma = 0.1, 0.15, 0.2$. They noticed that wider filter kernel effectively filtered out small scale oscillations in gradient fields than narrow kernels hence converge towards better optimal design while the narrow filter kernel got stuck with the steepest descent optimiser.

In these methods, the smoothed gradients are used by the optimiser hence $\mathcal{G} = \frac{dJ}{d\alpha}$ is replaced with $\widetilde{\mathcal{G}}$ to compute the search direction $\mathbf{d}$ as required in Eqn. 2.1. Kim et al. [67] noted that the implicit Sobolev gradient method produces oscillations on the airfoil surface even for a large value of β and Firl et al. [39] pointed out that gradient direction $\mathcal{G}$ is not preserved when smoothing the sensitivities hence might affect the estimation of curvature of the objective function to employ the Quasi-Newton method as an optimiser. In addition to that, the choice of smoothing intensity factor, filter kernel and filter radius, influence the design space hence the resultant optimal design. More importantly, these are all case dependent and user needs to select prior to the optimisation process based on experience from similar applications.

Since the designer is interested in obtaining a smooth geometry in a shape optimisation, the Laplacian-based regularisation method can be applied directly on the shape perturbations before updating the surface mesh nodes [64, 27, 104]. According to this approach, the gradient term $\mathcal{G} = \left(\frac{dJ}{d\alpha}\right)$ is need to be processed by the optimiser rather than $\widetilde{\mathcal{G}}$. Therefore to compute the total gradient one should differentiate the surface regularisation method to compute the shape sensitivity and map the CFD sensitivity $\mathcal{G}$ onto the design variables α.

2.2.2 Mesh Parameterisation

In the node-based parameterisation approach α has a same discretisation as the surface mesh X_s. So there exist N design variables α, hence offers the richest design space for shape optimisation. Alternatively, one can also reduce the size of the design space or use different discretisations for X_s and α. This can be done by projecting the surface mesh nodes to a handful of control points or volumetric lattice which controls the deformation of computational mesh embedding the volume. These types are commonly called as mesh parameterisation methods. Mesh parameterisation is the problem of computing a deformation function that smoothly maps the displacement of control points embedding the geometry to the displacement of surface grid points.

Given a deformation function (d), update to the computational mesh as given in the Eqn. 2.1 can be rewritten as,

$$X_s' = X_s + d(X_s) \tag{2.7}$$

Most commonly used mesh parameterisation methods for shape optimisation are Free-Form Deformations (FFD), Directly Manipulated Free-Form Deformations (DMFFD) and Radial Basis Function (RBF) morphing. Each method constructs the deformation function differently.

Free Form Deformation

The Free-Form Deformation (FFD) is a well-established mesh parameterisation method and has been widely used to create smooth deformations in a simulation-based shape optimisation process [111, 110, 12]. The FFD method is first introduced by Sederberg et al. [115] using trivariate Bezier volume and it has been extended to include B-splines [147] and NURBS volume [59] to utilize local shape modification properties which are found to be more useful to capture superior designs in shape optimisation [112, 133]. In this method, the baseline mesh ($\mathcal{M}$) to be deformed is embedded into a parametric space built by a set of structured control points ($\mathbf{P}_{ijk}$) where i, j, k represents the index along each Cartesian direction. In FFD, one needs to compute the parametric coordinates (u_1, u_2, u_3) for all grid points embedded inside the control volume. This requires an iterative root-finding Newton method. Once the mapping from physical space to parametric space has been

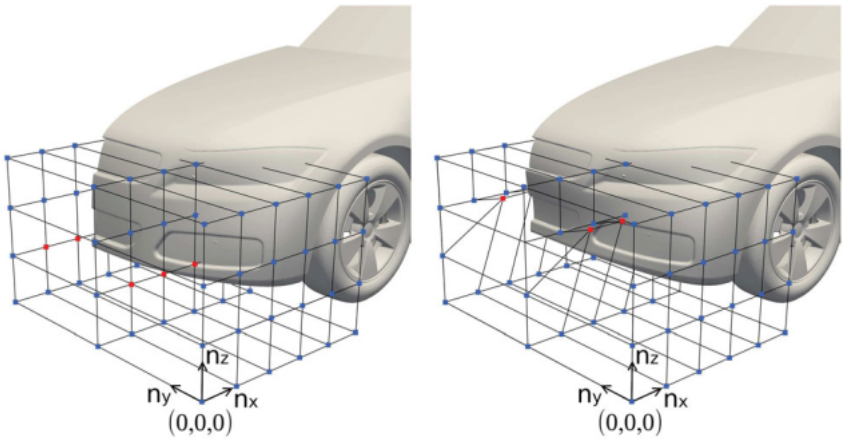


Figure 2.3: FFD based geometry deformation with its control hull [110]. Control points highlighted in red are only deformed

established, one can update the mesh inside the control volume by perturbing the control points. By using FFD one can deform both surface and volume mesh simultaneously. Figure 2.3 shows the deformation of a car front bumper using FFD approach with $6 \times 3 \times 4$ control hull. When compared with node-based parameterisation, additional surface regularisation is not required since geometry deformation is parameterised with smooth deformation function using a handful of control points. However, FFD requires the construction of auxillary control hull around the geometry which consumes a lot of pre-processing user set up time and its construction extremely cumbersome for complex geometries. Moreover, it requires a problem specific knowledge to achieve an effective set of trivariate control points around the computational mesh. If not careful, the more important sensitive region may be covered with a few number of control points. Nemec et al. [10] presented an approach to construct a second overlaid local control hull around a specific region to have a more localized geometry deformation which further makes the construction process more tedious.

Direct Manipulated FFD

In standrad FFD approach, the control lattice may be located far away from the shape which offers unintuitive design parameters for optimisation and makes it difficult to obtain constrained deformations [119, 145]. In direct manipulated FFD

(DMFFD) [86], user specifies the displacements ($\delta\widetilde{X}_h$) to some of the points on the shape $\widetilde{X}_h \in \mathcal{M}$ and computes the displacements to the control points $\delta\mathbf{P}$ that satisfies the user-specified displacements. Therefore instead of moving the control points directly as like in FFD, in this method the user directly moves points on the geometry which are normally referred as pilot or handle points $\widetilde{X}_h$. By doing in this way the constrained deformations can be achieved independent of the influence of control points. Since DMFFD offers more intuitive design variables than FFD, DMFFD converged to better design with improved convergence rates in shape optimisation [16, 86, 118]. Of course, DMFFD depends on the construction of the control lattice around the geometry hence facing issues similar to FFD.

Radial Basis Function (RBF) Morphing

RBF mesh morphing method is a global interpolation method used to deform both surface and volume mesh simultaneously in the design loop [91, 102]. It has properties similar to FFD method. For example, RBF method parameterise the shape displacements instead of parameterising the shape directly and deforms the geometry by means of control points surrounding the geometry. However, structured distribution of control grids are not required, the mesh points and the control grids are considered as a point clouds [89]. Morris et al. [89] presented RBF based parameterisation method for shape optimisation of airfoils and later, authors have extended the RBF method to $3D$ geometries including modern transport wing [90] and hovering rotors [9].

2.2.3 Analytical methods

Hicks and Henne [56] employed a set of smooth analytic functions to perturb the shape in the optimisation process. For example, airfoil geometry can be parameterised using a linear combination of basis functions and the baseline shape. Coefficients of the basis functions are taken as the design variables which determine the contribution of each basis function to the perturbed airfoil shape. In [56], Hicks and Henne proposed a set of sine functions for shape optimisation. This method is found effective for wing optimisation however difficult to extend it for a complex shapes [66, 128].

Sobieczky et al.[124] proposed a parameterisation method called PARSEC (Parametric SECtion) for shape optimisation of airfoils. This method parameterise an airfoil using eleven engineering parameters such as leading edge radius, trailing edge angle, upper and lower crest position, upper and lower crest curvature, airfoil max thickness etc. These geometric parameters ($\mathbf{P}_{G,i}$,$i = 1, 2, ...11$) are taken as design variables in the shape optimisation process. PARSEC parameterisation depends on the geometric properties of an airfoil which is easy to setup. However it offers only few parameters for shape optimisation which restricts the generation of superior designs during the design process. In addition to that, Wu et al. [141] pointed out that PARSEC method is difficult to generalize for a complex geometry other than wings.

Kulfan [69] proposed a shape parametrisation method called Class Shape Function Transformation (CST) method. In this method, any general 3D shapes can be represented by a distribution of class functions, shape functions and body size parameters. By varying the class parameters one can obtain different cross sectional shapes. This method parameterise most commonly used aircraft geometries such as wing shapes, fuselage cross sections and nacelles with a few number of design variables. Feng [151] parameterised wide range of aircraft components using CST parameterisation method including belly-fairing, fuselage, wing, nacelle etc. Even though only few design variables are used to parameterise the geometry a significant amount of computational effort is required to set-up the geometry for shape optimisation. Furthermore, the number and the selection of design variables are purely based on user experience hence important design variable may left out during the parameterisation stage which may affect generation of superior designs during a shape optimisation process.

2.2.4 Other Methods

Proper orthogonal decomposition (POD) is a order reduction technique highly used in statistical analysis to obtain a low dimensional approximation to a high dimensional vector data by computing dominant modes using SVD. Toal et al. [130] proposed an airfoil parameterisation method using SVD to derive an ordered set of orthogonal shape modes from a pre-selected airfoil database. Basic idea is that,

an existing airfoil database for particular flight condition contains a large number of best designs and by using an orthogonalisation process important geometric features could be extracted easily and used to generate best designs in the optimisation. Ghoman et al. [45] used this concept for shape optimisation of supercritical airfoils and employed a set of airfoils from specific family to extract shape modes for transonic flight condition. Poole et al. [105] applied this approach using a large airfoil database such as a library of NACA 4-series airfoils and the UIUC airfoil database for airfoil shape optimisation. Allen et al. [8] extended this approach to include shape optimisation of wing geometry using a sectional approach. Mode shapes extracted using this method are 2D and the design space is highly depends on the library of airfoils chosen which relies on user experience to select the suitable database. Straightforward extension to shape optimisation of complex 3D geometries is not easy because this method requires the large database of such 3D geometries which is not easy to create.

Subdivision surfaces are widely used in computer graphics and animation [35]. They describe a smooth surface using a control mesh. By successive use of subdivision scheme, various levels of control mesh can be obtained. Local refinement of control mesh around the region of interest is also possible [83]. More recently subdivision surfaces are received reasonable attention from the shape optimisation community. Masters et al. [84] presented a parameterisation method based on subdivision curves for the shape optimisation of $2D$ airfoils and considers coordinates of the control mesh as the design variables. Authors used a coarse level of control mesh in early stages of the optimisation and refined to a fine level as the design progres. In [82], authors from the same research group extended this approach to aerodynamic shape optimisation of the ONERA-M6 wing.

Even though CAD-free parameterisation methods eliminate the use of CAD-software or kernel in the design loop, the optimal shape exists as a computational mesh. Hence no datum shape is available for multi-disciplinary shape optimisation and manufacturing process. Therefore additional post-processing is required to approximate the optimal mesh back to CAD format. This approximation process could incur a significant error hence may loose optimal performance gained through an optimisation process.

2.3 CAD-based Parameterisation

On the other hand, in CAD-based parameterisation CAD model of a geometry is preserved in the design loop. Hence suitable for multi-disciplinary shape optimisation and manufacturing process. Based on how the parameterisation is defined using a CAD model, this method can be further divided into two types. They are 1) Explicit parameterisation and 2) Implicit parameterisation.

2.3.1 Explicit Parameterisation

Explicit parameterisation depends on a parameterised CAD model. Hence all included CAD parameters can be used as design variables. It is common for designers to use conventional engineering parameters for the design process. For example, a parameterisation of three-dimensional turbomachinery blade can be done by using engineering parameters such as inlet and exit blade angle, stager angle along the blade height, axial chord length, leading and trailing edge radius etc [134, 113]. This method can be easily integrated in the industrial design chain and geometric constraints can be built into the parameterisation. However, this method restricts the design space and performance improvement is highly depends on the chosen design parameters. Innovative or superior designs may not be obtained using this restricted design space.

Verstraete et al. [134] and Salvatore et al. [92] used Bezier and B-spline curves to parameterise 2D turbomachinery blade profiles respectively. Authors obtained the 3D shape of a blade by using cross-sectional design approach ($i.e$) by stacking parameterised 2D profiles from hub to tip. The number of 2D slices used in stacking is a designers choice. For example, Salvatore et al. [92] parameterised compressor stator blade using 7 slices and Tom et el. [134] used 3 slices for parameterising axial turbine blade. Similarly, Banovic et al. [19] also employed cross-sectional approach and used large number of 2D slices to parameterise a 3D turbine blade cooling channel. Optimum geometry obtained using this type of parameterisation purely depends on how the designer parameterised the baseline geometry.

On the other hand, one can also use parameters associated with the features of a CAD model as design variables. Feature-based parametric CAD model has been

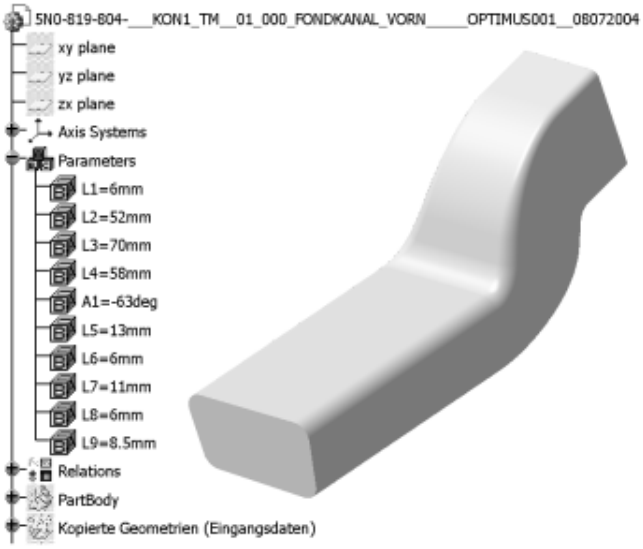


Figure 2.4: Parametric CAD model of S-bend geometry with design parameters [13]

widely adopted in commercial CAD systems such as CATIA, PRO-E, SolidWorks and open source CAD engine such as open CASCADE. In modern CAD systems, a geometry can be created by combining a number of modeling features such as extrude, revolve and sweep etc. Whole history of construction is recorded in the form of tree-like structure. For example, Fig. 2.4 shows the CAD model of a S-bend air duct with its feature tree and ten design parameters including nine length $L1 - L9$ and one angle parameter $A1$.

In shape optimisation, the geometry can be updated by changing the parameters of the feature and the CAD system automatically regenerate the parts affected by the update. The selected parameters should modify the geometry independent of each other and every update should result in a valid CAD model so that the design loop continues to run without any termination. Number of researches have developed API interfaces to CAD system to exposes suitable CAD parameters for shape optimisation. Robinson et al. [13, 109, 4] employed Visual Basic API for CATIA V5 to access feature tree parameters. Haimes et al. [51, 22] developed Computational Analysis PRogramming Interface (CAPRI) to expose and assign bounds to each CAD parameters to avoid regeneration error.

2.3.2 Implicit Parameterisation

Alternatively one can derive a parameterisation implicitly from the Boundary Representation of a CAD model. The BRep, in the typical standardised STEP or IGES format, represents the shape using a number of NURBS patches and considers location and weights of the NURBS control points to deform the geometry in the design process. The local shape modification property of NURBS offers wide range of shape modes in the design space and hence richest design space this representation can express. Martin et al. [76] and Wendisch et al. [148] presented parameterisation method based on BRep in which NURBS control points are directly used as design variables. However, the design surface consists of single NURBS patch which is only suitable for simple geometries. A complex geometry need to be represented using a collection of NURBS patches hence multiple NURBS patches need to be deformed in the design process. However, care must be taken to ensure continuity between NURBS patches when deforming multiple NURBS patches together in the design process. The CFD optimisation group at QMUL developed "NURBS based Parameterisation method with Complex Constraints (NSPCC)" for shape optimisation application.

The important contribution of NSPCC to CAD-based parameterisation based on the BRep is the formulation of geometric constraints, e.g. $G_0 - G_2$ continuity at NURBS patch interfaces or box, radius and thickness constraints. In addition, shape update preserves the topology of the CAD model hence one can employ the NSPCC method in a shape optimisation loop without any additional treatment. Parameterisation using NSPCC has been explained in 2D by Yu et al. [146] and in 3D with handling multiple NURBS patches by Xu et al. [142, 143] and later it was extended to deal with adaptive and intersecting NURBS patches in [65, 95]. More often, NURBS patches extracted from the BRep can contain geometric errors such as gaps, self-intersecting and/or trimmed NURBS patches and geometrical errors such as overlapping faces and unshared adjacent edges etc [101, 26]. These errors are introduced by the CAD engine itself during the data exchange process due to the absence of constraints on the boundary of the CAD model. In most cases, these errors are too small to be identified using visualization tools hence major problems are encountered when employed for mesh generation and shape optimisation process.

Therefore preprocessing step such as re-parameterisation of the existing CAD model is required to obtain a clean BRep model without geometric errors.

Auriemma et al. [15] proposed an approach for re-parameterisation based on a cross-sectional design approach that fits NURBS surface to the CAD geometry. Using NURBS one can represent a CAD geometry using a wide range of control net distributions starting from a coarser level that fits a CAD geometry within a suitable minimum level of tolerance to a finer parameterisation. However, it is a daunting task to determine the suitable parameterisation a-prior to the shape optimisation process. Therefore it is important to investigate the influence of the parameterisation on the shape optimisation process. In this present work, particular attention is given to investigate the influence of dimensionality of the design space, quality of the optima and their sensitivity to the choice of control net distribution on the aerodynamic shape optimisation.

2.4 Benchmark Test Cases

In this work, two optimisation cases are considered:

- Case 1: Lift constrained drag minimization of the ONERA M6-wing at trasonic flow conditions with thickness constraints. The freestream conditions considered in this work for validation and subsequent optimisation are similar to the standard benchmark conditions set out in the NPARC Alliance CFD Verification and Validation programe [3].

- Case 2: Reducing the mass-averaged total pressure loss of the VKI U-Bend geometry. This is a typical 180° bend duct used to circulate cooling air inside the turbine blade. This is a benchmark test case prepared for adjoint-based shape optimisation by the About Flow project organized by the Queen Mary University of London. Boundary conditions considered in this work for validation and subsequent optimisation are similar to the standard benchmark conditions specified in the About Flow project [135].

Other optimisation test cases are also available in the About Flow project. They are:

1. VKI LS89: This is a axial turbine nozzle guide vane, the objective of this case is to reduce the mass averaged entropy production in the cascade with constrained exit flow angle of 74°.

2. TU Berlin Turbolab Stator: Reduce the total pressure loss of the TU Berlin compressor stator.

3. TUM Drivaer Vehicle: Reducing drag of the vehicle by varying shape of the side view mirror.

The Aerodynamic Design Optimisation Discussion Group (ADODG) [2] also provided a range of benchmark cases for optimisation. They are:

1. Viscous drag minimisation of an RAE 2822 airfoil.

2. Subsonic, inviscid drag minimisation of a rectangular wing.

3. Drag minimisation of CRM wing.

2.5 Summary

In this chapter, most important shape parameterisation methods coming under CAD-free and CAD-based parameterisation methods are presented. When compared with the CAD-free methods, CAD-based methods preserves CAD model in the design loop which is more useful when employing multi-diciplinary shape optimisation and also for manufacturing. Hence CAD-based methods are normally preferred in industry for shape optimisation. However most of the shape parameterisation methods require manual setup [112, 145] which results in a restricted design space for shape optimisation. Based on information presented in this chapter, in this thesis more focus will be given on CAD-based parametisation methods. To be more specific, this work will mainly investigate NURBS-based paramterisation method for shape optimisation.

Chapter 3

Gradient-based shape optimisation

3.1 Introduction

Advancements in numerical simulation methods in combination with powerful computers have enabled the designers to create various design tools based on numerical optimisation algorithms. As a result, simulation-based methods such as Computational Fluid Dynamics (CFD) and Computational Structural Mechanics (CSM) are now no longer used only for the performance evaluation of a given complex configuration. With the development of adjoint approach, it is now possible to compute how a given objective function depends on the shape at a cost independent of the number of design variables and with the help of numerical optimiser it is now possible to improve the shape automatically in the design loop.

In the previous chapter, current state of the art to parameterise a given shape is presented. In this chapter, the details about how one can compute the accurate gradients efficiently in a design loop are presented. In this work an aerodynamic shape optimisation is performed in which the scalar objective function (J) is not only depends on the design variables but also on the physical state of the system U, input volume mesh X_v and surface mesh X_s which intern depends on design variables α. Mathematically it can be written as,

$$\begin{aligned} &\underset{\alpha}{\text{minimize}} && J(\alpha, U(X_v), X_v(X_s), X_s(\alpha)) \\ &\text{subject to} && R(\alpha, U) = 0 \\ &\text{and other geometric constraints.} \end{aligned} \tag{3.1}$$

where α represents the vector of design variables and U is the vector of state variables and R represents the steady state Reynolds Averaged Navier-Stokes Equations.

In a gradient-based shape optimisation, sensitivity of the objective function with respect to design variables are required. Automatic Differentiation (AD) software tools allow to compute derivatives for complex algorithms by differentiating the statements of a computer program that executes the algorithm. This has a number of advantages: a) the derivatives are exact and b) the AD process can be included in the build process of the code and any modifications of the primal code (the model to be differentiated) is automatically carried through to its derivative code. The details about the AD are presented in Section 3.2.

In general, the objective function as given in Eqn. 3.1 is computed through evaluation of separate blocks of code in a high-level programming language such as Fortran, C or C++. Differentiation of the objective function block by block can be represented as chaining up the derivatives for each of the blocks and it can be written as,

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial \alpha} + \underbrace{\frac{\partial J}{\partial U}}_{1}\underbrace{\frac{\partial U}{\partial X_v}}_{2}\underbrace{\frac{\partial X_v}{\partial X_s}}_{3}\underbrace{\frac{\partial X_s}{\partial \alpha}}_{4}. \tag{3.2}$$

In general, combined form of the first and the second term is called as the CFD sensitivity which is obtained by differentiating the flow solver (Sec. 3.3). The third term is called as the mesh sensitivity which is obtained by differentiating mesh deformation or mesh generation algorithm (Sec. 3.3.2). The fourth term is called as the shape sensitivity obtained by differentiating the shape parameterisation method (Sec. 3.3.3).

3.2 Automatic Differentiation

As mentioned in the previous section, in this present work only single discipline is considered. However, Automatic Differentiation tools can be applied to multi-disciplinary cases as well. Hence in this section J is assumed as vector valued objective function. This assumption is valid only for this section. If $J : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is an vector valued objective function and to solve an sensitivity-based optimisation problem we need to compute the sensitivity of J with respect to a vector of design

variables α which is called as the Jacobian matrix $\mathbf{J}$ and it is written as,

$$\nabla J = \begin{bmatrix} \frac{\partial J_1}{\partial \alpha_1} & \frac{\partial J_1}{\partial \alpha_2} & \cdots & \frac{\partial J_1}{\partial \alpha_n} \\ \vdots & \ddots & \cdots & \vdots \\ \frac{\partial J_m}{\partial \alpha_1} & \frac{\partial J_m}{\partial \alpha_2} & \cdots & \frac{\partial J_m}{\partial \alpha_n} \end{bmatrix} \tag{3.3}$$

Each column in the Jacobian matrix represents the sensitivity of m number of vector valued objective functions J with respect to one design variable α_j and each row represents the sensitivity of i^{th} objective function with respect to n number of design variables α. In large scale industrial applications, the objective function J is typically not given in analytic form but rather available as a computer code written in a high-level programming language Fortran, C or C++.

Automatic or Algorithmic Differentiation (AD) have become more and more widely used in sensitivity-based shape optimisation problems [31, 28, 27]. AD produces codes that compute exact derivatives of arbitrarily complex programs in which each line can be expressed as a sequence of elementary instructions computing an elementary differentiable function. There exist two fundamental modes of algorithmic differentiation: tangent and adjoint (reverse) mode. Detailed discussion about the two modes of AD techniques can be found in [49] and recent advances in AD can also be found in [42].

An AD tool in forward mode generates the Tangent Linear Model (TLM) ($\dot{J}$) of the given program that computes (J) and for one forward sweep with a given seed vector $\dot{\alpha} \in \mathbb{R}^n$ computes the product of the Jacobian with $\dot{\alpha}$,

$$\dot{J} = \nabla J \cdot \dot{\alpha}. \tag{3.4}$$

If a given seed vector $\dot{\alpha}$ is initialized by setting 1 for the j^{th} variable and 0 for all other variable then the forward sweep computes j^{th} column of the Jacobian matrix. Thus n forward sweeps are requirred to compute the full Jacobian ∇J. In the Reverse mode, AD tool generates the adjoint code and for one reverse sweep with a given seed vector $\bar{J} \in \mathbb{R}^m$ computes the product of the transpose Jacobian with seed vector as,

$$\bar{\alpha} = \big(\nabla J\big)^T \cdot \bar{J}. \tag{3.5}$$

If a given seed vector $\bar{J}$ is initialized by setting 1 for the i^{th} element and 0 for all other elements then the reverse sweep computes i^{th} row of the Jacobian matrix $\dot{J}_i$. Thus m reverse sweeps are requirred to compute the full Jacobian ∇J. If T is a computational time of a program which computes J, then obtaining full Jacobian ∇J using tangent-linear program with a seed vector $\dot{\alpha} \in \mathbb{R}^n$ requires a runtime of $\mathcal{O}(nT)$. Alternatively, computing full Jacobian ∇J, by calling the adjoint program with a seed vector $\bar{J} \in \mathbb{R}^m$ requires a runtime of $\mathcal{O}(mT)$. Therefore the adjoint program reduce the computational time if the number of inputs (n) are larger than the number of functions (m), $m << n$.

Computational costs associated with each method can also be analyzed using computational graph as shown in Figs 3.3 and 3.2. In this case only 4 design variables with single objective function (J) are considered and the principle is similar for n number of design variables. The graph shown in Fig. 3.1 traces the flow of computation of the primal program and in this case from 4 design variables from the top to 1 objective function at the bottom. Each node corresponds to intermediate values of a function or a line of code in the program. Connecting edges between nodes represents that each node value is the input to the other node. By using these connecting edges the partial derivatives of the node with respect to each input can be traced. In the forward mode we have to seed one input at a time and it moves towards the end. As a result it tracks how each α affects every node. Therefore we need to execute the tangent linear code 4 times which is shown in Fig. 3.2. However, reverse-mode AD records the computational graph of a program and starts at the output J and moves towards α $(i.e)$ it propagates the seed backwards and computes derivative of J with respect to every node. As a result derivative of J with respect to each input $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ computed in a single reverse sweep which is shown in Fig. 3.3.

3.2.1 AD Implementation

Algorithmic differentiation can be implemented in two different ways. They are: a) source-code transformation and b) operator overloading.

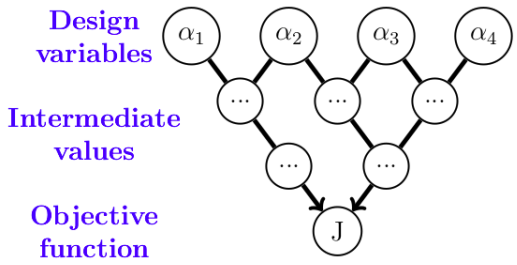


Figure 3.1: Computational graph for primal calculation (left)

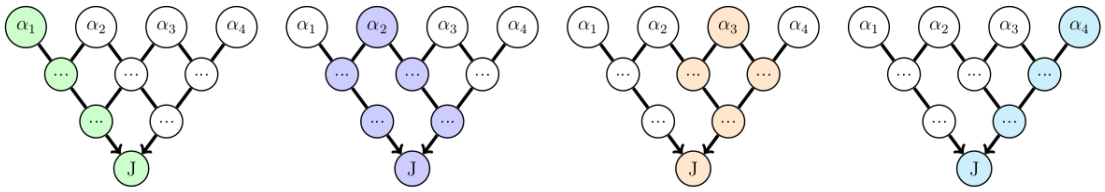


Figure 3.2: Computational graph for forward mode derivative calculation

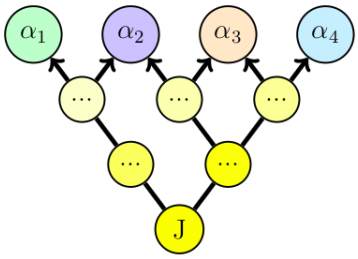


Figure 3.3: Computational graph for reverse mode derivative calculation

3.2.2 Operator Overloading

In operator overloading, declarations of all the floating-point variables are replaced with a new data type that stores both derivative and primal information, and overloading the operators such that they compute the primal as well as propagate its derivatives. In reverse-mode, operators compute the primal function and record the computational graph using the 'tape' which stores each elemental code operation and interpreted in reverse to accumulate derivatives. This method is easy to implement and can be applied to straightforwardly to large codes. However for primal program with large computational graph 'tape' mechanisms leads to excessive memory requirements and also impede compiler optimisations. AD tools based on operator overloading method are ADOL-C [48], DCO/C++ [97].

3.2.3 Source-Code Transformation

Source-transformation (s-t) AD parses the source code of the primal and produces the required new source code for the derivative computation. The 'tape' hence does not exist as a list operations, but as a compact set of instructions similar to the primal code. Consider e.g. point inversion, the computation of the parametriccoordinates on the surface of the projection of a surface mesh point onto the geometry, which is obtained by solving a linear system of equations using Newton's method for each point. The reverse mode of operator overloading AD needs to record the operations for all Newton steps for all points.

Using source transformation AD on the other hand would re-use the differentiated source for all nodes and hence avoid tape storage. This makes the source transformation AD tool a more attractive option and this is what is used in the current work. Furthermore, AD process can be included in the build process of the code and any modifications of the primal code (the model to be differentiated) is automatically carried through to its derivative code. AD tools based on source transformation are Tapenade [53], TAF [47] and OpenAD/F [131]. In this present work, AD tool Tapenade is used to obtain both CFD and shape sensitivities as given in Eqn. 3.2.

3.3 Discrete Adjoint Formulation

This section presents the classical discrete adjoint formulation used for aerodynamic shape optimisation process. As presented in Sec. 3.1, the function of intersect shown in Eqn. 3.1 depends not only on the design variables (α) but also on the physical state of the system (U). In short form Eqn. 3.1 can be written as,

$$J = J(\alpha, U(\alpha)), \tag{3.6}$$

where α represents the vector of design variables for $n = 1, ..., N$ and U is the vector of state variables. For a given vector α, the solution of the governing equations of the system yields a state vector U, thus establishing the dependence of the state vectors on the design variables. Generally, these steady state governing equations are non-linear and this system of equations are solved using an iterative method by driving the residuals R to zero which arises from the discretisation of the conservation equation. Hence, the governing equations are denoted as,

$$R(\alpha, U(\alpha)) = 0, \tag{3.7}$$

where R represents steady-state Reynolds Averaged Navier-Stokes equations. In a gradient-based optimisation, sensitivity of the objective function with respect to design variables are computed by applying chain rule to Eqn. 3.6. It can be written as,

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial \alpha} + \frac{\partial J}{\partial U}\frac{dU}{d\alpha}. \tag{3.8}$$

In Eqn. 3.8, the term $\frac{dU}{d\alpha}$ represents the change of the state variable U with respect to α which is called perturbation field. Therefore to evaluate the total sensitivity of the cost function, one needs to evaluate the perturbation flow field for each design variables. Linearising the non-linear discrete governing equations yields,

$$\frac{\partial R}{\partial \alpha} + \frac{\partial R}{\partial U}\frac{dU}{d\alpha} = 0. \tag{3.9}$$

This could be solved iteratively to compute the perturbation field $\frac{dU}{d\alpha}$ with respect to each design variable,

$$\frac{\partial R}{\partial U}\frac{dU}{d\alpha} = -\frac{\partial R}{\partial \alpha}, \tag{3.10}$$

which can be written in short form as

$$Au = f, \tag{3.11}$$

where

$$A = \frac{\partial R}{\partial U}, \quad u = \frac{\partial U}{\partial \alpha} \quad \text{and} \quad f = -\frac{\partial R}{\partial \alpha}. \tag{3.12}$$

In Eqn. 3.11, A represents Jacobian and f is the source term which is the negative partial derivative of the residual with respect to the design variables and u is the perturbation field which represents the change in the flow field with respect to α. Eliminating u from Eq. (3.8), the total sensitivity equation becomes,

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial \alpha} - \frac{\partial J}{\partial U}\left(\frac{\partial R}{\partial U}\right)^{-1}\frac{\partial R}{\partial \alpha}, \tag{3.13}$$

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial \alpha} + g^T A^{-1} f, \tag{3.14}$$

where

$$g^T = \frac{\partial J}{\partial U}.$$

We first need to solve the Eq.(3.11) for $\frac{dU}{d\alpha}$ and then substitute the result in expression (3.14) for the computation of total sensitivity for each design variable α. However, solving the system of equation Eqn. 3.11 for the perturbation field is usually as costly as solving the governing equations. When we multiply this cost of evaluations by the total number of design variables, then the total cost for calculating the sensitivity vector may become prohibitive

However, this is the starting point of discrete version of the adjoint-based approach. By transposing and rearranging Eq. 3.14, we can observe that for computing the adjoint sensitivity $(\frac{dJ}{d\alpha})^T$, the auxiliary vector ψ can be obtained by solving the linear system of adjoint equations see Eqn. 3.17 generated by the second and third terms in the right hand side of the total sensitivity Eqn. 3.13.

$$\left(\frac{dJ}{d\alpha}\right)^T = \left(\frac{\partial J}{\partial \alpha} + g^T A^{-1} f\right)^T, \tag{3.15}$$

$$\left(\frac{dJ}{d\alpha}\right)^T = \left(\frac{\partial J}{\partial \alpha}\right)^T + f^T (A^{-1})^T g. \tag{3.16}$$

The term $A^{-T}g$ is solved for the adjoint variable ψ,

$$A^T \psi = g, \tag{3.17}$$

where

$$\psi = \left(\frac{\partial J}{\partial R}\right)^T. \tag{3.18}$$

In expanded form, the adjoint Eqn. 3.17 can be written as,

$$\left(\frac{\partial R}{\partial U}\right)^T \left(\frac{dJ}{dR}\right)^T = \left(\frac{\partial J}{\partial U}\right)^T. \tag{3.19}$$

Using the adjoint solution which obtained by solving Eqn. 3.17, the adjoint sensitivity equation as shown in Eqn. 3.16 can be rewritten as,

$$\left(\frac{dJ}{d\alpha}\right)^T = \left(\frac{\partial J}{\partial \alpha}\right)^T + f^T \psi, \tag{3.20}$$

$$\left(\frac{dJ}{d\alpha}\right)^T = \left(\frac{\partial J}{\partial \alpha}\right)^T - \left(\frac{\partial R}{\partial \alpha}\right)^T \psi. \tag{3.21}$$

Source term f is a function of design variables. The dependency structure of the residuals R of the governing equations describing the flow to the design variables can be written as,

$$R = R(X_v(X_s(\alpha))). \tag{3.22}$$

Hence the source term now can be written as,

$$f = -\frac{\partial R}{\partial \alpha} = \frac{\partial R}{\partial X_v}\frac{\partial X_v}{\partial X_s}\frac{\partial X_s}{\partial \alpha}, \tag{3.23}$$

which modifies the adjoint sensitivity Eqn. 3.20 as,

$$\left(\frac{dJ}{d\alpha}\right)^T = \left(\frac{\partial J}{\partial \alpha}\right)^T + \left(\frac{\partial X_s}{\partial \alpha}\right)^T \left(\frac{\partial X_v}{\partial X_s}\right)^T \left(\frac{\partial R}{\partial X_v}\right)^T \psi. \tag{3.24}$$

The adjoint method traces the sensitivity backwards from the cost function through all intermediate blocks to the design variables. Hence a change in α now affects only the left most term which is computed last. Unless there is a change in objective function J or a change in the flow field U, the adjoint CFD sensitivity term $\left(\frac{\partial J}{\partial X_v}\right)^T$ is evaluated only once. Therefore after solving the governing equations as given in Eqn. 3.7, an adjoint system of equations needs to be solved once for each objective which is shown in Eqn. 3.17 and it is valid for all the design variables. However, if there is more than a single objective, all additional objectives require an additional adjoint solve. This reduces the computational time and makes the adjoint approach more efficient for handling a large number of design variables. Hence this approach is used in this present work.

3.3.1 STAMPS: Flow and Discrete Adjoint Solver

The flow and discrete adjoint solver employed in this work is called STAMPS (Source Transformed Adjoint Multi-Physics Solver) [93] which is an in-house code developed by the CFD optimisation group at the Queen Mary Unversity of London (QMUL). It has been used in the references [27, 28, 19, 92, 65, 150, 149, 50]. Initially the CFD code is named as mgOpt and as the functionalites of the solver is increasing continuously by the developers it is now named as STAMPS. The primal flow solver of STAMPS uses a standard node-centred, edge-based compressible finite volume discretisation using MUSCL type reconstruction of primitive variables with second order accuracy and stable implicit JT-KIRK scheme [144]. The viscous source terms are obtained using an edge-corrected Green-Gauss formula. Turbulence modelling is performed with Spalart-Allmaras RANS model with AUSM scheme for the convective fluxes. The extended details about the numerical method, primal and adjoint implementations in STAMPS can be found in [27, 50].

Typical structure of the compressible flow solver with explicit Forward-Euler timestepping is shown in Listing. 3.1 in which U are the flow variables, X_v represents the volume mesh, Nrm is the edge and boundary normals, $\mathbf{R}$ represents the residuals of the flow equations and J is the scalar objective function. For a given volume mesh X_v, the algorithm solves the governing flow equations and computes the objective function of interest. The downward and upward pointing arrow indicates inputs and

outputs to the subroutine respectively.

To compute the sensitivity of the objective function with respect to volume mesh coordinates ($i.e$) the volume sensitivity $\overline{X}_v = \frac{\partial J}{\partial X_v}$, the primal algorithm can be passed to AD tool for brute force differentiation and the resultant sensitivity code is shown in Listing. 3.2 in which overline represents the reverse differentiated code with its corresponding input and output variables. The major disadvantage of this approach, it includes repeated computation of sensitivity of the residual with respect to Nrm ($i.e$) $\frac{\partial \mathbf{R}}{\partial X_v}$ at every iteration which increases the computational cost. From Eqn. 3.24, one can see that this term is normally computed only once after the adjoint solution is converged. Hence repeated computation is not required. Furthermore the adjoint variables $\overline{U}$ always initialized with zero which is also not suitable if one wants to perform a hot start with previous solution.

As an alternative, Faidon et al. [28] suggested to differentiate only the required subroutines from the solver and use the same primal time stepping algorithm for adjoint which guaranteed convergence if the primal is converged. Here the `cost_function` subroutine is reverse differentiated with respect to U and Nrm, and the `residual` subroutine is differentiated with respect to U and Nrm. The `update` subroutine is self adjoint hence the same subroutine as used in primal computation is used in adjoint calculation as well. The resultant primal time stepping adjoint algorithm is shown in Listing. 3.3. By using this algorithm, adjoint solution $\overline{U}$ can be initialized with partially or fully converged adjoint solution from previous design step. Each operations in the algorithm shown in Listing. 3.3 computes the sensitivities in the following order:

1. $\overline{\texttt{cost\_function}}$ computes the source term, $g = \left(\frac{\partial J}{\partial U}\right)^T$.

2. do-loop: computes the adjoint solution $\psi = \left(\frac{\partial J}{\partial R}\right)^T = A^{-T}g$

3. $\overline{\texttt{residual\_nrm}}$ computes: $\left(\frac{\partial J}{\partial Nrm}\right)^T = \left(\frac{\partial \mathbf{R}}{\partial Nrm}\right)^T \left(\frac{\partial J}{\partial \mathbf{R}}\right)^T$

4. $\overline{\texttt{metrics}}$ computes: $\overline{X}_v = \left(\frac{\partial J}{\partial X_v}\right)^T = \left(\frac{\partial Nrm}{\partial X_v}\right)^T \left(\frac{\partial J}{\partial Nrm}\right)^T$.

```
PROGRAM primal
CALL initialise_flow (↑U)
CALL metrics (↓X_v, ↑Nrm)
DO It = 1, nIter
    CALL residual (↓U, ↓Nrm, ↑R)
    CALL update (↓R, ↑U)
 END DO
CALL cost_fun (↓U, ↓Nrm, ↑J)
END PROGRAM
```

Listing 3.1: Typical structure of the primal solver

```
PROGRAM adjoint
J̄ = 1
CALL cost_fun(↑g, ↑Nrm̄, ↓J̄)
DO nIter = mIt, 1, -1
   CALL update (↓R̄,↑Ū)
   CALL residual_u (↑R̄, ↓Ū)
END DO
CALL metrics (↑X̄, ↓Nrm̄)
END PROGRAM
```

Listing 3.2: Adjoint code via 'Brute force' differentiation

```
PROGRAM adjoint
J̄ = 1
CALL cost_fun(↑ g,  ↑ Nrm,  ↓ J̄)
DO It = 1, nIter
   CALL residual_u (↑ R̄,  ↓ Ū)
   R̄ = R̄ - g
   CALL update (↓ R̄, ↑ Ū)
END DO
CALL residual_nrm (↓ Ū,  ↑ Nrm)
CALL metrics (↑ X̄,  ↓ Nrm)
END PROGRAM
```

Listing 3.3: Adjoint code with primal time stepping for compressible solver

3.3.2 Computation of Design Surface Sensitivity

Mesh Morphing Algorithm in STAMPS:

At each design step, optimiser updates the shape for design improvement. To continue the design cycle, the computational mesh involved in the simulation needs to be synchronized with the updated shape. This can be done by the use of structured meshing algorithm with frozen mesh topology. If the design perturbations are small one can use available mesh morphing algorithms to smoothly propagate the surface mesh displacements to volume mesh coordinates. Wide range of mesh morphing algorithms are available spring-analogy, Radian Basis Functions(RBF), Free-Form Deformation (FFD), linear elasticity and inverse-distance weighting (IDW). STAMPS offers inverse-distance weighted interpolation method for mesh deformation. The algorithm of IDW is well documented in literature and it is only briefly discussed here. Unlike other mesh deformation methods IDW is an explicit method can be implemented in a matrix-free form, hence it is computationally less expensive and it is written as,

$$\delta X_v = \frac{\sum_{i=1}^{N_s} \mathbf{W}_{s,i} \delta X_{s,i}}{\sum_{i=1}^{N_s} \mathbf{W}_{s,i}} = \sum_{i=1}^{N_s} k_{s,i} \delta X_{s,i} = \mathbf{K} \delta X_s, \tag{3.25}$$

where N_s is the number of surface mesh nodes and $\mathbf{W}_{s,i}$ describes the weighting function based on inverse-distance ($\| X_v - X_s^i \|$). Our experiments show that IDW preserves high quality boundary layer mesh during geometry deformation.

Once the primal and adjoint solution are computed, sensitivity of the objective function with respect to volume mesh nodes which typically called as volume sensitivity are assembled as shown in Listing. 3.3. Then the adjoint sensitivity of the objective function with respect to the surface mesh nodes $\overline{X_s}$ which also called as the CFD sensitivities are computed by projecting the volume sensitivity onto the surface mesh nodes.

Therefore, the inverse distance weighted mesh deformation algorithm needs to be differentiated. IDW is a linear operator and in the present work we exactly differentiate the IDW mesh deformation algorithm to project volume sensitivity to surface sensitivity. By using Eqn. 3.25, sensitivity of mesh deformation algorithm is written as,

$$\left(\frac{\partial X_v}{\partial X_s}\right)^T = \mathbf{K}^T. \tag{3.26}$$

Then the final projection can be written as,

$$\overline{X}_s = \left(\frac{\partial J}{\partial X_s}\right)^T = \left(\frac{\partial X_v}{\partial X_s}\right)^T \overline{X}_v. \tag{3.27}$$

3.3.3 Projection of Design Surface Sensitivity to Shape

To complete the chain rule as given in the Eqn. 3.24 we need to compute the sensitivity of the objective function with respect to design variables α. The product term $\left(\frac{\partial J}{\partial \alpha}\right)^T = \left(\frac{\partial X_s}{\partial \alpha}\right)^T \left(\frac{\partial J}{\partial X_s}\right)^T$ can be computed using AD tool without explicitly computing and storing the transposed shape sensitivity matrix $\left(\frac{\partial X_s}{\partial \alpha}\right)^T$. The final projection can be written as,

$$\overline{\alpha} = \left(\frac{\partial J}{\partial \alpha}\right)^T = \left(\frac{\partial X_s}{\partial \alpha}\right)^T \overline{X_s}. \tag{3.28}$$

From Eqn. 3.28 one can see that, seeding $\bar{X}_s$ to the reverse differentiated parameterisation subroutine $\left(\frac{\partial X_s}{\partial \alpha}\right)^T$ computes $\overline{\alpha}$ efficiently independent to the size of the number of design variables.

3.4 Shape optimisation Framework using 'one-shot' Methodology

Adjoint methodology presented in the previous section computes the gradient of the objective function with respect to design variables which describes the linearized behavior of the objective function in the vicinity of the current design. By using the gradient-based optimiser the direction of the design improvement can be computed. In this work, gradient-based optimiser provided by the *SciPy* library such as steepest descent and Broyden–Fletcher–Goldfarb–Shanno (BFGS) methods are employed to compute the direction of the design improvement. In general, perturbations ($\delta\alpha$) to design variables can be written as,

$$\delta\alpha = -\gamma \mathbf{H} \frac{dJ}{d\alpha} \tag{3.29}$$

where γ represents the step size which is given by Armijo and Wolf based line search conditions and $\mathbf{H}$ is identity matrix for SD or inverse Hessian approximation matrix for the BFGS method.

In the present work, an optimisation workflow based on the one-shot methodology is employed to simultaneously converge flow (Eqn. 3.7), adjoint (Eqn. 3.17) and the design (Eqn. 3.24). One-shot method places more importance on the direction of the gradients rather than their magnitude and last CFD and adjoint solutions are used as the initial guess for the subsequent design iterations. However both primal and adjoint solutions are required to converged fully for the first design iteration. Then, stopping criteria proposed by Christakopoulos et al. [27] as shown in the Eqn. 3.30 is employed for the remaining design iterations which control the convergence of primal and adjoint solutions dynamically via norm of the gradients.

$$g_{RMS} = \frac{\sum \mid \frac{dJ}{d\alpha} \mid}{C} \tag{3.30}$$

where g_{RMS} is the stopping criterion and C is a user-defined constant. The schematic overview of the optimisation process which couples NSPCC and STAMPS is presented in Fig. 3.4.

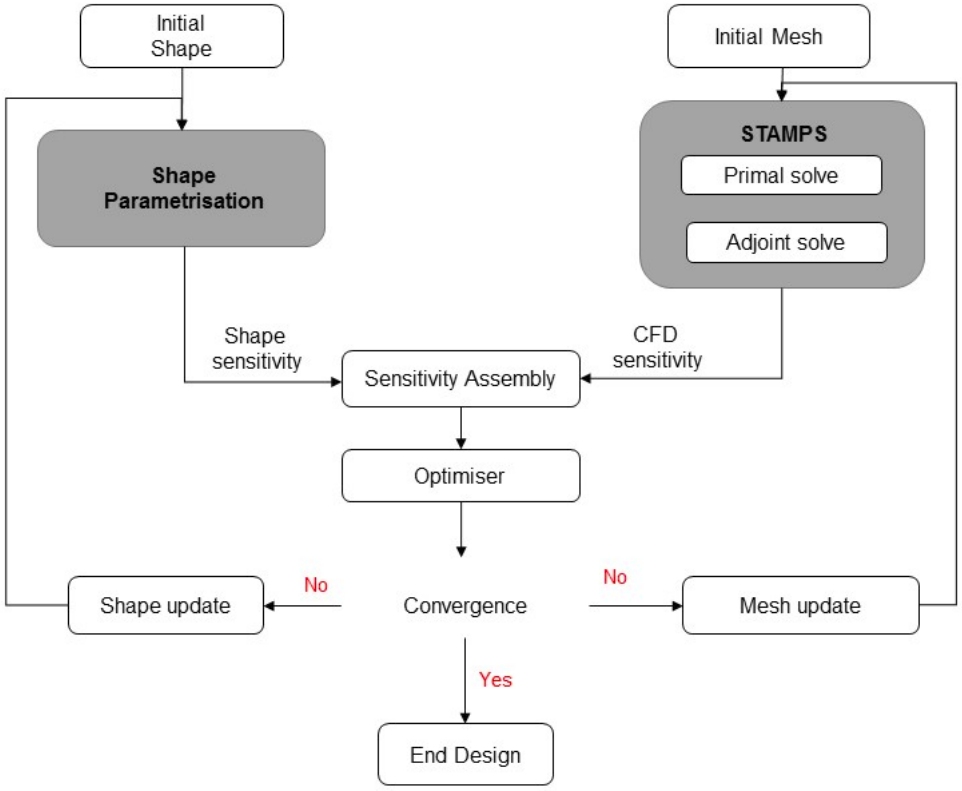


Figure 3.4: Shape optimisation work flow with STAMPS

3.5 Summary

This chapter presents the adjoint methodology to compute the gradient of the objective function with respect to design variables and its role in aerodynamic shape optimisation. After a brief description of the Algorithmic Differentiation (AD) tools, the derivation of the adjoint gradients and its computational efficiency are shown. Brief details about the in-house flow and discrete adjoint solver STAMPS are presented and examined the role of source transformation AD tool in the generation of adjoint code. Finally, details about the one-shot methodology for shape optimisation are presented.

Chapter 4

Adaptive Parameterisation

4.1 Introduction

In the last chapter, details about the primal solver and the discrete adjoint methodology to compute the sensitivity of the objective function with respect to the surface mesh coordinates are presented. Once the 'raw' sensitivities are computed, they can be coupled with the shape parameterisation method to perform shape optimisation. This is the subject of the present chapter.

Computer Aided Design (CAD) systems are extensively used to create 3D model of a complex geometry. This involves a series of geometric operations such as sweeping, filleting, booleans etc. CAD models are not only used for visualization purpose but also extremely useful for computational mesh generation, manufacturing, multidisciplinary analysis and optimisation. More importantly interactions among different disciplines require a common CAD geometry, for example, design of flexible structures such as wings. However there are some challenges must be addressed before using CAD models in a gradient-based shape optimisation process. They are:

1. Construction of a parametricCAD model is a time consuming process and depends on the user input hence important parameters may not be included during the parameterisation. For a complex geometry it is difficult to add additional parameters automatically in the region of interest during the design process.

2. To complete the chain rule as given in the Eqn. 3.24 we need to evaluate the shape derivative term which gives the sensitivity of the surface mesh node X_s with respect to the design variable α, $\frac{\partial X_s}{\partial \alpha}$. This must be computed over a long list of successive geometric operations and may contain non-differentiable operations such as booleans. Hence analytic derivatives are difficult to obtain. Moreover commercial CAD systems do not provide shape derivatives and often closed source hence impossible to apply algorithmic differentiation tools.

To address these issues, CFD optimisation group at Queen Mary University of London developed 'NURBS-based Parameterisation with Complex Constraints (NSPCC)' method for shape optimisation. NURBS are the de-facto industry standard for geometry representation and data exchange between CAD systems. NSPCC approach derives parameterisation directly from the Boundary Representation (BRep) of a CAD model without any user input and uses control points and/or weights to deform the geometry. NURBS offer local shape modification property and can represent both analytic and free-form shapes with relatively few number of control points.

In the NSPCC approach, surface node sensitivities are projected onto the control points of the design surface. In general, computational mesh resolution is decided based on RANS requirements. For example, fine mesh resolution is required for wall bounded flows to capture flow separation accurately. However, control points distribution on the geometry depends on the local curvature of the shape and when compared with B-splines surface, NURBS can represent the geometry with a small number of control points. Hence this projection filters out high frequency gradient modes and generates smooth geometry in the design process. Previously, the NSPCC approach has been tested in shape optimisation with fixed control point net on the design surface. The optimal distribution of control net is difficult to obtain a-prior to the shape optimisation process. If the control net is too coarse then the important gradient modes may not be projected effectively which restricts the generation of superior designs outside this fixed envelope. On the other hand finer distribution of control points may leads to inefficient navigation in the design space and may converge to local minima. Therefore the choice of the control net distribution on the design surface determines the set of shapes during the shape optimisation hence

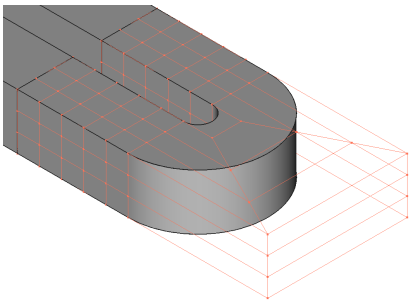
influence the best achievable design and rate of convergence of the design process.

In this present work, NSPCC approach is extended to include adaptive design space for shape optimisation by refining the control points locally on the design surface using knot insertion algorithm. However we still need to find when to terminate the current design space level and which knot value to insert. This chapter is organized as follows. Brief introduction about the boundary representation and a knot insertion algorithm is presented in Section 4.3. Section 4.5 presents the formulation of geometric constraints, e.g. $G_0 - G_1$ continuity at NURBS patch interfaces using test point approach. In Section 4.9, details about the adaptive algorithm are presented.

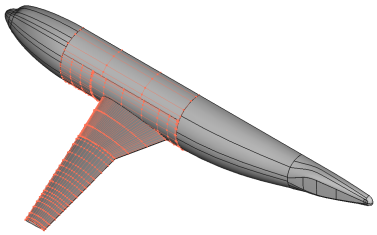
In order to test the proposed adaptive NSPCC method, a simple adaptive node-based parameterisation method is developed which derives parameterisation directly from the computational mesh employed in the simulation. In the node-based parameterisation, displacements of the surface grid nodes are the design variables which offers the richest design space the CFD discretisation can consider. This design space is even too rich for the CFD as the parameterisation method can express high-frequency modes which are not adequately resolved by the CFD and hence remain poorly damped. Additional regularisation is necessary, and implicit [63] as well as explicit [64] smoothing methods have been proposed, both of them requiring to tune a smoothing coefficient. In this work a simple explicit smoothing method is proposed in which optimisation starts with the large amount of smoothing and as the design progresses smoothing is reduced to include high-frequency shape modes in the design space. In Section 4.10, details about the proposed simple explicit surface regularisation method are presented.

4.2 Boundary Representation (BRep) of a CAD model

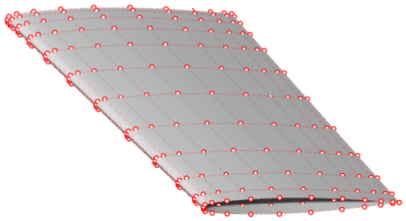
The BRep, in the typical standardised STEP format, represents the geometry as a collection of NURBS patches. Typical boundary representation of various CAD models with its control points distribution on the design patches are shown in Fig. 4.1. The BRep consists of both topological and geometrical entities of a CAD



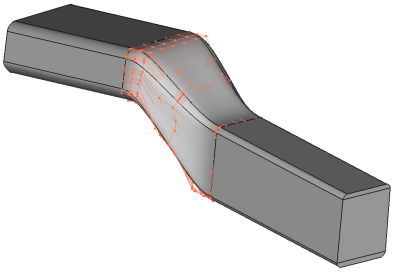
(a) U-Bend cooling channel



(b) Wing-body configuration



(c) Onera M6 wing



(d) S-bend

Figure 4.1: BRep of various CAD models with its control points distributions

model. The brief overview of the BRep structure of a CAD model is shown in Fig. 4.2 in which topological entities are highlighted in blue, and geometric entities are highlighted in green. In the BRep, a CAD model is represented as a closed shell formed by several faces. A face is represented by a surface and each face is bounded by a closed-loop entity called wire. A wire consists of a set of adjacent edges and an edge is represented by a curve and bounded by two vertices and a vertex is represented by a point.

In the BRep, surfaces and curves are represented by NURBS which is a current industrial standard in CAD systems. Before presenting the NSPCC approach, brief overview of NURBS curves and surfaces are discussed in this section. The reader is referred to Piegl et al [103] for more detailed discussion on NURBS curves and surfaces.

Curve representation

The NURBS curve exported by a CAD system is a univariate parametriccurve defined by piecewise rational functions with degree p and parameter s. Mathematically it is written as,

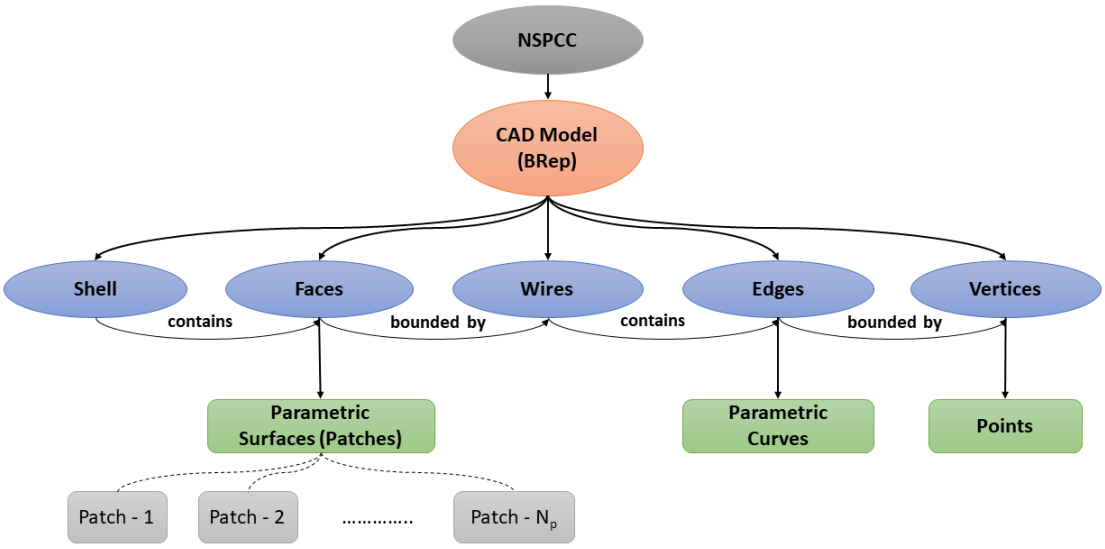


Figure 4.2: Overview of the BRep structure of a CAD model

$$\mathbf{C}(s) = \frac{\sum_{i=0}^{n-1} \mathcal{B}_{i,p}(s) w_i \mathbf{P}_i}{\sum_{i=0}^{n-1} \mathcal{B}_{i,p}(s) w_i} \tag{4.1}$$

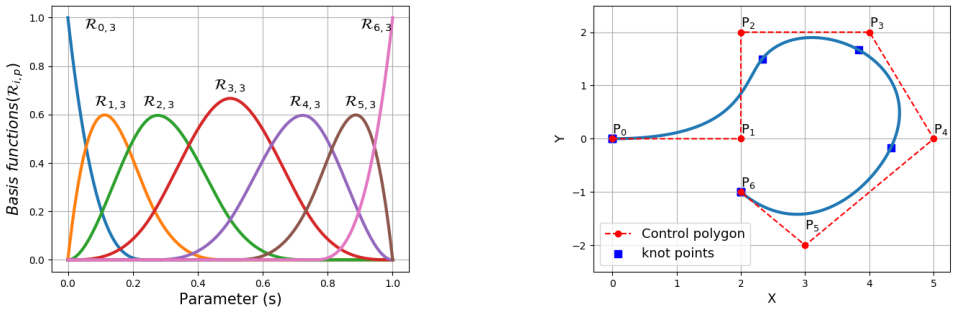
where $\mathbf{P}_i$ are the n number of control points defining a control polygon, $\mathcal{B}_{i,p}(s)$ are the B-spline basis functions and w_i are scalars called as weights. If all the weights are set to one then Eqn. 4.1 represents a B-spline curve. The pth-degree basis function $\mathcal{B}_{i,p}$ is uniquely defined on a knot vector $S = \{\bar{s}_0, ...\bar{s}_{r-1}\}$ which consists of r elements of real numbers in a non-decreasing order. The general form of non-periodic knot vector is given in the Eqn. 4.2,

$$S = \{\underbrace{a, \ldots, a}_{p+1}, \bar{s}_{p+1}, \ldots\ldots, \bar{s}_{r-p-1}, \underbrace{b \ldots, b}_{p+1}\} \tag{4.2}$$

where $\bar{s}$ are called knots and S is the knot vector. Equation 4.2 corresponds to open knot vector in which first and last knot values have $p+1$ multiplicity. The degree of the basis function (p), the number of knots in a knot vector (r) and the number of control points of the NURBS curve (n) are related and it is written as,

$$r = n + p + 1. \tag{4.3}$$

The pth-degree B-spline basis functions are computed by the Cox-de Boor recursion formula as given below,



(a) Basis functions

(b) B-spline curve

Figure 4.3: An example cubic B-spline curve with rational basis functions

$$\mathcal{B}_{i,0}(s) = \begin{cases} 1 & \text{if } s_i \leq s < s_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathcal{B}_{i,p}(s) = \frac{s - s_i}{s_{i+p} - s_i}\mathcal{B}_{i,p-1}(s) + \frac{s_{i+p+1} - s}{s_{i+p+1} - u_{i+1}}\mathcal{B}_{i+1,p-1}(s) \tag{4.4}$$

Figure 4.3 shows an example of cubic basis functions defined by the knot vector $S = \{0, 0, 0, 0, 0.25, 0.5, 0.75, 1, 1, 1, 1\}$ and the corresponding B-spline curve. Figure 4.3 also shows the knot points on the curve which are the points on the NURBS curve corresponds to each knot points $\bar{s}_i$ in a knot vector. These knot points divide the NURBS curve into curve segments defined on a knot span.

Surface representation

In the BRep, NURBS patches are used to represent the surface of a geometry. NURBS patches exported by a CAD system are bivariate parametricsurface defined by piecewise rational functions with degree p and q in the parameter direction s and t respectively. Mathematically it is written as [103],

$$\mathbf{S}(s,t) = \frac{\sum_{i=0}^{n-1}\sum_{j=0}^{m-1}\mathcal{B}_{i,p}(s)\mathcal{B}_{j,q}(t)\mathbf{P}_{i,j}w_{i,j}}{\sum_{i=0}^{n-1}\sum_{j=0}^{m-1}\mathcal{B}_{i,p}(s)\mathcal{B}_{j,q}(t)w_{i,j}}, \tag{4.5}$$

where $\mathbf{P}_{i,j}$ are the control points which form a bidirectional control polygon with n and m number of control points along each oparameter direction s and t respectively. $\mathcal{B}_{i,p}(s), \mathcal{B}_{j,q}(t)$ are the B-spline basis functions defined on the knot vectors $S = \{\bar{s}_0, \ldots, \bar{s}_r\}$ and $T = \{\bar{t}_0, \ldots, \bar{t}_k\}$ with r and k are the number of knots along each parameter direction respectively. The relation given in Eqn. 4.3 is valid for both

parameter direction of the NURBS surface. For the current work, we use a non-periodic or clamped type of knot vectors as given in the Eqn. 4.6 and Eqn. 4.7 in which the first and last knots have multiplicity of $p+1$ and $q+1$ in S and T knot vectors respectively.

$$S = \{\underbrace{a, \ldots, a}_{p+1}, \bar{s}_{p+1}, \ldots\ldots, \bar{s}_{r-p-1}, \underbrace{b \ldots, b}_{p+1}\} \tag{4.6}$$

$$T = \{\underbrace{\bar{a}, \ldots, \bar{a}}_{q+1}, \bar{t}_{q+1}, \ldots\ldots, \bar{t}_{s-q-1}, \underbrace{\bar{b} \ldots, \bar{b}}_{q+1}\} \tag{4.7}$$

4.3 Manipulating NURBS surfaces

NURBS offers different algorithms for geometry handling which includes knot insertion, degree elevation, knot removal and degree reduction. Among these knot insertion [52, 55, 117, 116] and degree elevation [36, 37] algorithms are used in shape optimisation to refine control points on the design surface.

4.3.1 Knot insertion

As the name suggests, knot insertion inserts a knot into the existing knot vector and computes the refined control net without modifying the geometry. For example, in a $2D$ case, knot insertion inserts a new knot $\tilde{s}$ into the existing knot vector S and computes the updated control points $\mathbf{Q}_i$ without changing the curve. Once the knot span $\{\bar{s}_x, \bar{s}_{x+1}\}$ that contains this new knot $\tilde{s}$ is determined, the $p-1$ control points from the initial control polygon must be replaced with the p number of new control points ($\mathbf{Q}_{x-p+1}$ to $\mathbf{Q}_x$) as a function of initial control points $\mathbf{P}_i$ [103].

$$\mathbf{Q}_i = \gamma_i \mathbf{P}_i + (1-\gamma_i)\mathbf{P}_{i-1} \tag{4.8}$$

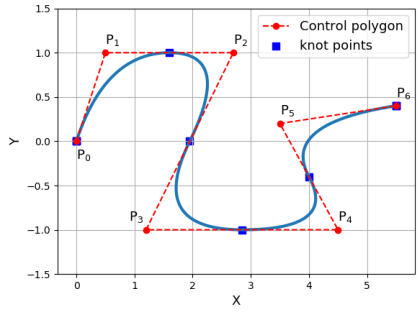
$$\gamma_i = \begin{cases} 1 & \text{if } i \leq x-p \\ \frac{\tilde{s}-\bar{s}_i}{\bar{s}_{i+p}-\bar{s}_i} & \text{if } x-p+1 \leq i \leq x \\ 0 & \text{if } i \geq x+1 \end{cases}$$

where p is the degree of the curve. An simple demonstration of knot insertion in a quadratic B-spline curve is shown in Figs 4.5a and 4.6a. Figure 4.4a shows a quadratic curve with initial control polygon defined by the knot vector $S =$

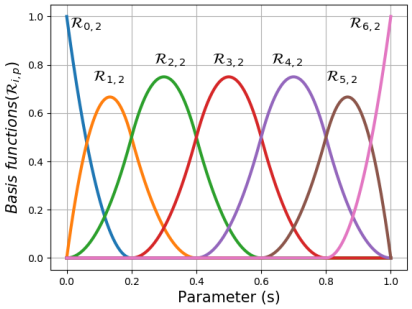
$\{0, 0, 0, 0.2, 0.4, 0.6, 0.8, 11, 1\}$ and Fig 4.4b shows the basis functions of the curve. The initial curve is divided into 5 segments which are all shown by the knot points in the Fig 4.4a. A single non-repeated knot $\tilde{s} = 0.3$ is inserted into the knot vector, as a result additional control points are added and the number of segments increased to 6. Figure 4.5a shows the comparison of the control polygon before and after insertion with the knot points corresponds to the refined control polygon. It is important to note that knot insertion does not alter the shape of the curve and inserting non-repeating knots preserved smoothness of the basis functions. Figure 4.5b shows the refined basis functions after inserting a non-repeated knot.

However inserting a repeated knot reduces the smoothness of the basis function. For example, a repeated knot $\tilde{s} = 0.2$ is inserted into the initial knot vector. Figure 4.6a shows the comparison of the control polygon before and after insertion with the knots points corresponds to the refined control polygon. While the new control points are created, the number of curve segments remain unchanged and reduced smoothness of the basis function which is shown in Fig. 4.6b. In general, pth-degree NURBS curve/surface is C^{p-l} continuous at knot point, where l is a number of knot multiplicity. In this example, the refined curve is C^0 continuous at $\bar{s}_4 = 0.2$. Hence the curve forced to pass through a control point $\mathbf{Q}_3$. During the shape deformation, this property of knot multiplicity reduces the smoothness of the geometry and may introduce sharp features hence additional constraints should be imposed on the control points to maintain smooth geometry deformation in the design process. Hence in this work, care has been taken to avoid inserting repeated knots in the design process.

As similar to the NURBS curve, one can insert a new knot in s and/or t parameter direction of a NURBS surface. From Eqn. 4.5, $\mathbf{P}_{i,j}$ are the bidirectional control net of the initial NURBS patch with dimension $n \times m$, i represents the row index $0 \leq i \leq n-1$ and j represents the column index $0 \leq i \leq m-1$. When a new knot $\tilde{s}$ is inserted to the existing knot vector S, from Eqn. 4.3 the number of control points along the s direction increases by one. Note that degree of the surface/curve remain fixed during the knot insertion. Then the refined control net $\mathbf{Q}^s_{i,j}$ with the dimension $(n+1) \times m$ is obtained by doing a knot insertion $\tilde{s}$ on each of the m columns of control points. Similarly, refinement along t direction

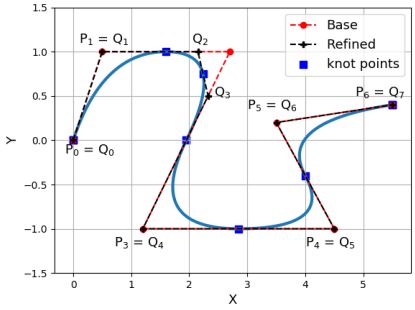


(a) Curve with base control polygon

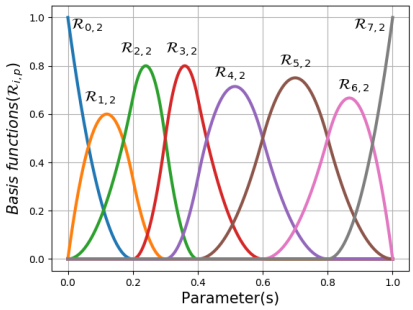


(b) Original basis

Figure 4.4: A quadratic B-spline curve with basis functions

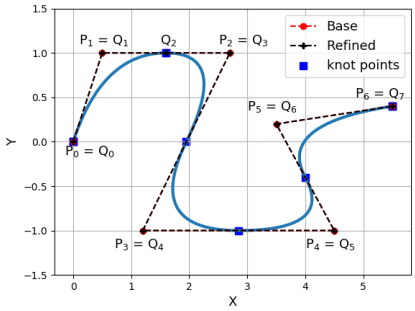


(a) Base vs refined control polygon

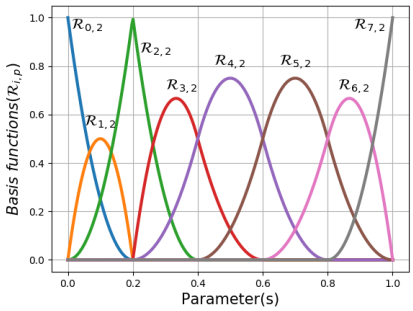


(b) Refined basis

Figure 4.5: Effect of inserting non-repeated knot



(a) Base vs refined control polygon



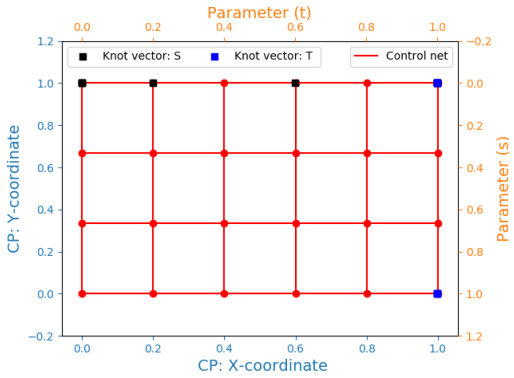
(b) Refined basis

Figure 4.6: Effect of inserting repeated knot

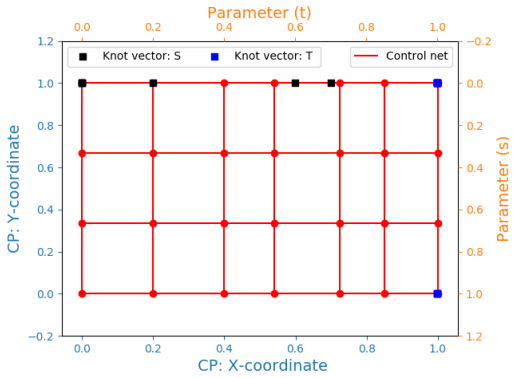
$(\mathbf{Q}^t_{i,j})$ is performed by doing a knot insertion $\tilde{t}$ on each of the n rows of control points. A simple demonstration of knot insertion in a bi-cubic NURBS patch is shown in Fig 4.7. As mentioned earlier, knot insertion alters only the control net of the NURBS patch not its shape hence the geometry remain unchanged both geometrically and parametrically. To show the effect of knot insertion only the control net of the NURBS patch is shown in the Fig. 4.7. Fig. 4.7a shows the control net of the initial bi-cubic NURBS patch with the knot vector $S = \{0, 0, 0, 0, 1, 1, 1, 1\}$ and $T = \{0, 0, 0, 0, 0.2, 0.6, 1, 1, 1, 1\}$ in the parameter direction s and t respectively. A single non-repeated knot $\bar{t} = 0.7$ is inserted into the knot vector T, as a result additional column of control points are computed (See Eqn. 4.8) as a function of initial control points $\mathbf{P}_{i,j}$. Figure 4.7b shows the refined control net $(\mathbf{Q}^t_{i,j})$ with the updated knot vector $T = \{0, 0, 0, 0, 0.2, 0.6, 0.7, 1, 1, 1, 1\}$. Figure 4.7c shows the refined control net $(\mathbf{Q}^s_{i,j})$ after inserting a knot $\tilde{s} = 0.2$ in the s direction with the updated knot vector $S = \{0, 0, 0, 0, 0.2, 1, 1, 1, 1\}$. Figure 4.7d shows the refined control net after inserting the above mentioned both knots. Note that knot insertion not only increases the control points it also modifies the location of neighboring control points.

4.3.2 Degree elevation

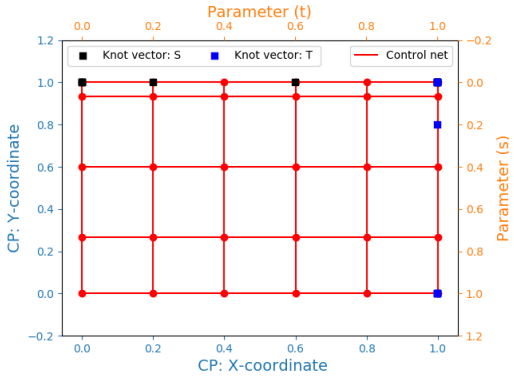
Degree elevation is the process of increasing the degree of the basis function in s and/or t direction without changing the geometry. Due to the fundamental equality as given in Eqn 4.3, degree elevation by 1 modifies corresponding knot vector and control points. A simple demonstration of degree elevation is shown in Fig. 4.8 where the initial cubic B-spline curve is defined by 7 control points and after degree elevation it is defined by 11 control points. Unlike knot insertion, degree elevation is a global process which modifies all control points except the corner points. Hence degree elevation is not suitable for local refinement. Dervieux et al.[36] used degree elevation algorithm to construct three different levels of FFD frames and employed multilevel parameterisation approach for shape optimisation.



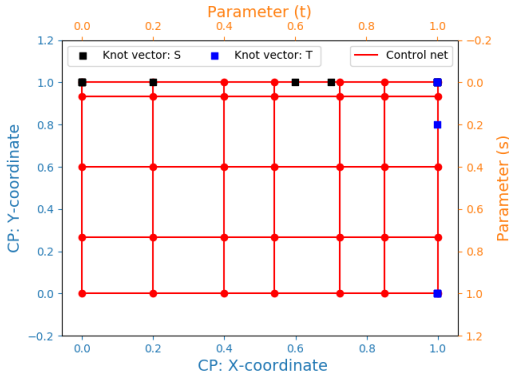
(a) Before knot insertion



(b) After knot insertion: $\tilde{t} = 0.7$

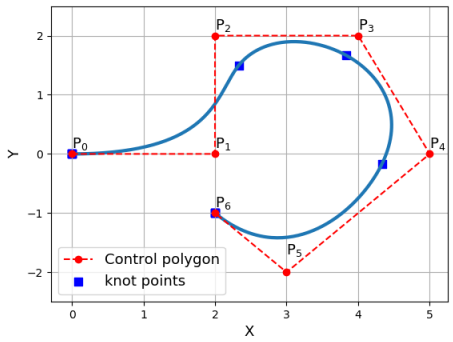


(c) After knot insertion: $\tilde{s} = 0.2$

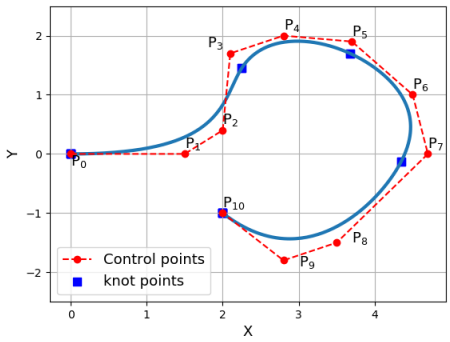


(d) Inserting both knots

Figure 4.7: Effect of knot insertion in a NURBS patch. (a) Original control net (b) Inserting a knot $\tilde{t} = 0.7$ one time in the t direction, (c) Inserting a knot $\tilde{s} = 0.2$ one time in the s direction, (b) Inserting both knots $\tilde{t} = 0.7$, $\tilde{s} = 0.2$ one time in the parameter t and s respectively



(a) Cubic B-spline curve



(b) After degree elevation, $p = 4$

Figure 4.8: Effect of degree elevation in a B-spline curve

4.3.3 Knot removal

Knot removal is the reverse operation of knot insertion in which an existing knot value is removed from the knot vector by fixing the degree hence the associated control points are removed. However knot removal is not always successful, it modifies the geometry hence not suitable for shape optimisation. Knot removal is mainly used for data reduction in computer graphics to obtain a compact representation of a geometry [73].

4.4 Surface Mesh Mapping

In general, a CAD model and its corresponding computational mesh are generated using two different tools. Therefore it is important to obtain a computational link between the surface mesh points and the CAD geometry. During mesh generation surface mesh points belonging to the design surfaces can be distinguished and exported with separate surface tags in the mesh file. By using this tag, **MESH EXTRACTOR** module reads all the surface mesh points. The link between computational mesh and the underlying CAD geometry is established by finding the (s,t) coordinates of each mesh point in the parametricspace of the NURBS patch it belongs to. In the literature, the process of mapping from geometric space to parametricspace ($\mathbb{R}_{x,y,z} \rightarrow \mathbb{R}_{s,t}$) is termed as the point inversion problem [103] which computes the parametriccoordinates (s,t) by minimising the distance between the surface mesh point X_s to the NURBS surface $\mathbf{S}(s,t)$. Mathematically this can be written as [103],

$$min \left\{ \|\mathbf{S}(s,t) - X_s\|^2 \right\} \tag{4.9}$$

The computed parametriccoordinates are remain fixed throughout the optimisation and stored for surface mesh update.

4.5 Deformation with Geometric Constraints

The NSPCC approach [142, 143] employs control points of a conformal patch topology to deform the geometry in the design process. Hence, a finite control point displacement $\mathbf{P}_{i,j}$ on or near a patch interface affects the surface continuities such

as G_0 (no gaps), G_1 (tangency) and G_2 (curvature). For example, Fig. 4.9 shows the local shape modification property of NURBS ($\mathbf{S}(s,t)$) by perturbing control points at two different locations one at the interface and one at the distance far away from the interface. A control point displacement at the interface between two adjacent

(a) Original NURBS (b) Perturbed NURBS

Figure 4.9: Shape deformation of a NURBS patch with its control net

patches creates a surface discontinuity which is clearly visible in Figure 4.9b.

Given the BRep of a CAD model as explained in Section 4.2, by using the NSPCC approach it is possible to deform multiple patches in shape optimisation without violating surface continuity requirements. In Computer Aided Geometric Design (CAGD), continuity is the mathematical way to indicate how smooth the NURBS patches merge at the interface. For example, to ensure G_0 or positional continuity two NURBS surfaces must have a common edge. To ensure G_1 or tangency continuity, G_0 must be satisfied first and two NURBS surfaces must share the same tangent plane for any point along the common edge. G_2 or curvature continuity ensures that two NURBS surfaces must have a common edge, the tangent vectors lying along the same direction and having the same rate of change of tangent direction (curvature) at the interface. Therefore, based on the smoothness requirements, geometric constraints are need to be maintained among the adjacent patches in a CAD model in the design process.

In the NSPCC approach constraints are evaluated at a number of linearly distributed test points on the common edge of the adjacent patches. However, this does not require that adjacent patches should maintain the same number and control points distribution. The only requirement is that the test points need to be distributed only in pairs with one on each adjacent NURBS patch. Figure 4.10 shows deployed test points along a common edge of the two NURBS patches. The calculation of required number of test points along each edge is discussed in the Section 4.6.

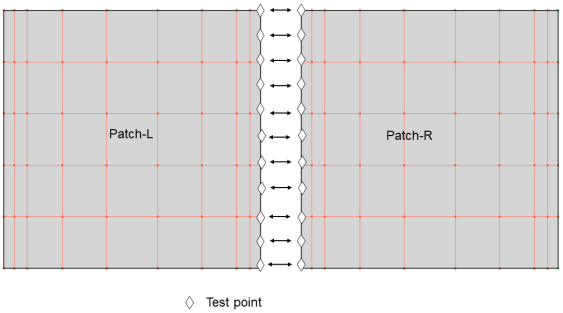


Figure 4.10: Test points along a common edge and corresponding control net of adjacent NURBS patches

To ensure G_0 and G_1 continuity along an edge, the following constraint equations which is linear for G_0 as given in the Eqn. (4.10) and non-linear for G_1 as given in Eqn. (4.11) are need to be maintained and evaluated discretely at each pair of test points:

$$G_0 = X_{p,L} - X_{p,R} = 0 \tag{4.10}$$

$$G_1 = \vec{n}_L \times \vec{n}_R = 0, \tag{4.11}$$

where $X_{p,L}$ and $X_{p,R}$ are the positional coordinates, $\vec{n_L}$ and $\vec{n_R}$ are the unit normal tangent plane. These values are evaluated at the test points deployed along a common edge and suffix L and R corresponds to Patch-L and Patch-R respectively. The unit normal vector $\vec{n}$ at a point is defined as,

$$\vec{n} = \frac{\frac{\partial X_p}{\partial s} \times \frac{\partial X_p}{\partial t}}{\left\| \frac{\partial X_p}{\partial s} \times \frac{\partial X_p}{\partial t} \right\|}. \tag{4.12}$$

where s and t are the parametriccoordinates of the NURBS. In each design update the change in constraints as given in the Eqn. (4.13) should remain zero,

$$G_g^n - G_g^{n+1} = 0, \tag{4.13}$$

where G_g^n and G_g^{n+1} represents the constraint values at design iteration n and $n+1$ respectively with $g = 0$ and 1 for G_0 and G_1 constraints respectively. Therefore, by linearising the Eqn. (4.13) we obtain,

$$G_g^{n+1} \simeq G_g^n + \sum_{i=1}^{N} \frac{\partial G_g}{\partial \mathbf{P_i}} \delta \mathbf{P}_i, \tag{4.14}$$

and assembling the Jacobian for each of the constraint equation as given in the Eqn. 4.14 we obtain the following linear system of equations

$$\mathbf{C}\delta\mathbf{P} = 0. \tag{4.15}$$

The matrix $\mathbf{C}$ is called constraint matrix and $\delta\mathbf{P}_i$ denotes the displacement of x, y, z coordinates of N control points and in vector form both terms are written as,

$$\mathbf{C} = \begin{bmatrix} \frac{\partial G_{0,1}}{\partial \mathbf{P}_1} & \frac{\partial G_{0,1}}{\partial \mathbf{P}_2} & \cdots & \frac{\partial G_{0,1}}{\partial \mathbf{P}_N} \\ \vdots & \ddots & \vdots & \\ \frac{\partial G_{0,M_0}}{\partial \mathbf{P}_1} & \frac{\partial G_{0,M_0}}{\partial \mathbf{P}_2} & \cdots & \frac{\partial G_{0,M_0}}{\partial \mathbf{P}_N} \\ \cdots & \cdots & \cdots & \\ \frac{\partial G_{1,1}}{\partial \mathbf{P}_1} & \frac{\partial G_{1,1}}{\partial \mathbf{P}_2} & \cdots & \frac{\partial G_{1,1}}{\partial \mathbf{P}_N} \\ \vdots & \ddots & \vdots & \\ \frac{\partial G_{1,M_1}}{\partial \mathbf{P}_1} & \frac{\partial G_{1,M_1}}{\partial \mathbf{P}_2} & \cdots & \frac{\partial G_{1,M_1}}{\partial \mathbf{P}_N} \end{bmatrix} = \begin{bmatrix} \mathbf{G}_0^j \\ \cdots \\ \mathbf{G}_1^j \end{bmatrix}, \tag{4.16}$$

$$\delta\mathbf{P} = \begin{bmatrix} \delta\mathbf{P}_1 \\ \delta\mathbf{P}_2 \\ \vdots \\ \delta\mathbf{P}_N \end{bmatrix}$$

where M_0 and M_1 correspond to the total number of G_0 and G_1 constraint equations respectively and j is the edge index. By assembling the matrix $\mathbf{C}$ as given in Eqn. 4.16 we can impose different continuity constraints to different edges. The matrix has a total of M_c rows where M_c corresponds to total number of constraint equations with N columns which is the total number of control points. Using a projected gradient approach, the control point perturbations (design space) has to lie in the null space of $\mathbf{C}$ and the design modes are the N basis vector of the null space and determined using Singular Value Decomposition (SVD).

$$\mathbf{C} = \mathbf{U}\Sigma\mathbf{V}^T \tag{4.17}$$

where $\mathbf{U}$ is the $M_c \times M_c$ unitary matrix, Σ is an $M_c \times N$ diagonal matrix with positive real numbers on the diagonal σ_i called singular values of $\mathbf{C}$ and $\mathbf{V}^T$ represents a

$N \times N$ unitary matrix. The number of non-zero singular values in Σ determines the theoretical rank r of the constraint matrix, $\mathbf{C}$ and last $(N - r)$ columns of the matrix $\mathbf{V}$ span the exact null space of $\mathbf{C}$. With the presence of non-linear constraints singular values show a gradual decrease rather than sudden drop to zero. Therefore in NSPCC a cut-off threshold frequency value σ_C is used to determine the rank of the matrix $\mathbf{C}$ which is denoted as numerical rank r' and the corresponding nullspace associated with the numerical rank r' is termed as numerical nullspace and it is denoted as Ker($\mathbf{C}$). Each column in Ker($\mathbf{C}$) gives a deformation mode that satisfies the constraints and orthogonal to each other. Therefore the resultant control point perturbations are computed as the linear combination of the columns of numerical nullspace which offers richest design space for the shape optimisation and it is written as,

$$\delta\mathbf{P} = \sum_{k=1}^{N-r'} v_{k+r'}\delta\alpha_k = \mathrm{Ker}(\mathbf{C})\delta\alpha. \tag{4.18}$$

where $\delta\alpha_k$ with $k = 1, 2, ...N - r'$ are the perturbations to design variables and v_{k+r} are the columns of the numerical nullspace. Hence the size of the design space is $N - r'$ which is determined by the total number of control points and the threshold frequency σ_C.

4.6 Computation of number of test points along a common edge

The required number of test points along each edge needs to be determined. This could be computed by running a series of SVD computations with different number of test points until the number of non-zero singular values in the linearised constraint matrix $\mathbf{C}$ doesn't change any further. This process is computationally expensive. Alternatively Zhang et al. [149] proposed an approach to compute the same number of test points for edges having linear (G_0) and non-linear (G_1) constraints. The edges with non-linear constraints might require a large number of test points hence we extended the approach to define different number of test points for edges having linear and non-linear constraints.

A common edge defined by NURBS can be viewed as a union of polynomial

segments with each segment of degree $p_{e,L}$ and $p_{e,R}$ corresponds to Patch-L and Patch-R (See Fig. 4.10) respectively. For Patch-L we require $p_{e,L}+1$ test-points to fit each polynomial segment exactly. Hence total number of test points for a common edge (Patch-L) should satisfy the following condition,

$$M_{e,L} \geq (p_{e,L}+1)N_{i,L}. \tag{4.19}$$

$N_{i,L}$ is the total number of polynomial segments for a common edge corresponds to patch-L and it can be written as,

$$N_{i,L} = N_{e,L} - p_{e,L} - K_{e,L} \tag{4.20}$$

where $N_{e,L}$ and $K_{e,L}$ are the total number of control points and number of zero knot spans (arising because of knot multiplicities) defining the common edge. Similar expression can also be obtained for common edge corresponds to Patch-R. Hence the total number of test points for a common edge can be obtained as,

$$M_e \geq f_T \; max(M_{e,L}, M_{e,R}) \tag{4.21}$$

where f_T is an inflation factor and can be chosen between $1.0 \leq f_T \leq 1.5$ for edges having G_0 continuity and $2.0 \leq f_T \leq 3.5$ for edges having G_1 continuity.

4.7 Constraint Recovery

Constraint handling using projected gradient approach restricts the geometry perturbations to be tangent to the linearised constraint functions. For non-linear constraints like G_1 each tangent step slightly violates the constraints. Hence additional normal steps in the range space of $\mathbf{C}$ is required to recover violated non-linear constraints. The recovery of the control points perturbation $\delta P_\perp$ is given as,

$$\mathbf{C}_{dev}\delta\mathbf{P}_\perp + \delta G_{dev} = 0, \tag{4.22}$$

where δG_{dev} is the constraint deviation from the target value and $\mathbf{C}_{dev}$ is the constraint Jacobian matrix only related to non-linear constraints that are violated. Each recovery step should strictly satisfy the linear constraints hence $\delta P_\perp$ becomes,

$$\delta\mathbf{P}_\perp = \mathrm{K}er(\mathbf{C}_{sat})\delta\alpha_\perp \tag{4.23}$$

where $\mathbf{C}_{sat}$ is the Jacobian matrix containing only linear (satisfied) constraints and $\alpha_\perp$ is the recovery (normal) perturbations to design variables. By substituting Eqn. 4.23 into Eqn. 4.22 we obtain,

$$\mathbf{C}_{dev}\mathrm{K}er(\mathbf{C}_{sat})\delta\alpha_\perp + \delta G_{dev} = 0. \tag{4.24}$$

We need to first solve the Eqn. 4.24 for $\delta\alpha_\perp$ and it is given as,

$$\delta\alpha_\perp = -[\mathbf{C}_{dev}\mathrm{K}er(\mathbf{C})_{sat}]^{+}\delta G_{dev} \tag{4.25}$$

where $[\mathbf{C}_{dev}\mathrm{K}er(\mathbf{C})_{sat}]^{+}$ is the pseudo inverse obtained by using SVD. Hence Eqn. 4.23 can be rewritten as,

$$\delta\mathbf{P}_\perp = -\mathrm{K}er(\mathbf{C}_{sat})[\mathbf{C}_{dev}\mathrm{K}er(\mathbf{C})_{sat}]^{+}\delta G_{dev} \tag{4.26}$$

Equation 4.26 guarantees that they lie in the null space of $\mathbf{C}_{sat}$ hence linear constraints are satisfied in each recovery step. This recovery step is an iterative process which will converge in a few Newton steps if the deviation value is taken back to below the chosen threshold value. The effect of CAD geometry deformation with and without constraint recovery is shown in Fig 4.11.

4.8 Computation of CAD sensitivity

To complete the chain rule as given Eqn. 3.28, the sensitivity of surface mesh points with respect to design variables (*i.e*) CAD sensitivity needs to be computed. By using Eqns. 4.5 and 4.18 the CAD sensitivity term can be written as,

$$\frac{\partial X_s}{\partial \alpha} = \frac{\partial X_s}{\partial \mathbf{P}}\frac{\partial \mathbf{P}}{\partial \alpha}. \tag{4.27}$$

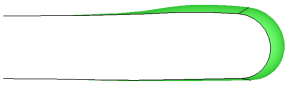
By transposing Eqn. 4.27, the adjoint CAD sensitivity term can be written as,

$$\left(\frac{\partial X_s}{\partial \alpha}\right)^T = \left(\frac{\partial \mathbf{P}}{\partial \alpha}\right)^T \left(\frac{\partial X_s}{\partial \mathbf{P}}\right)^T. \tag{4.28}$$

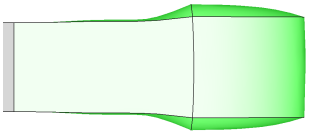
With a CAD model in the design loop, Eqn. 3.28 can be modified to compute the total sensitivity of the objective function with respect to design variables,

$$\overline{\alpha} = \left(\frac{\partial J}{\partial \alpha}\right)^T = \left(\frac{\partial \mathbf{P}}{\partial \alpha}\right)^T \left(\frac{\partial X_s}{\partial \mathbf{P}}\right)^T \bar{X}_s. \tag{4.29}$$

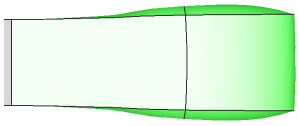
(a) Inner U-Bend without constraint recovery



(b) Inner U-Bend with constraint recovery



(c) Without constraint recovery



(d) With constraint recovery

Figure 4.11: Effect of G_1 constraint recovery: Deformed U-Bend geometry

With geometric constraints, the numerical nullspace as given in Eqn. 3.28 needs to be included in the chain rule. Therefore Eqn. 4.29 can be written as,

$$\overline{\alpha} = \left(\frac{\partial J}{\partial \alpha}\right)^T = \left(\mathrm{Ker}(\mathbf{C})\right)^T \left(\frac{\partial X_s}{\partial \mathbf{P}}\right)^T \overline{X_s}. \tag{4.30}$$

where $\overline{X}_s$ represents the adjoint sensitivity of the objective function with respect to surface mesh coordinates $\left(\frac{\partial J}{\partial X_s}\right)^T$. As discussed in Section 3.3.2, this term is computed by the STAMPS solver and the term $\left(\frac{\partial X_s}{\partial \mathbf{P}}\right)^T$ is computed by differentiation the NSPCC CAD kernel using source-transformation AD tool in reverse mode. The complete static NSPCC framework is summarized in the following Algorithm 1.

4.9 Adaptive Refinement

In NURBS-based geometry parameterisation, number of control points and their distribution on the design surface strongly influence the best achievable design in the shape optimisation. The adaptive NSPCC method developed in this work progressively adds more control points on the design surface using knot insertion algorithm. However it is important to determine when to terminate the current optimisation and which knot value to insert. As discussed in Sec.4.2, a NURBS surface is a function of two parameters, hence one can insert a knot in either the s or t direction or in both parameter direction. Each knot insertion adds a row or column of control points and also modifies the neighboring control points. Therefore it is important to determine the suitable value of knot to insert so that the refined control net has

Algorithm 1 Static NSPCC Framework for Shape Optimisation

Input: CAD file in STEP format and computational mesh. σ_C and δG_T

1: Read BRep of a CAD model as a collection of NURBS patches **(S)**.

2: Read surface mesh points on the design surface (X_s)

3: **Surface mesh mapping:** ($\mathbb{R}_{\mathrm{x,y,z}} \rightarrow \mathbb{R}_{\mathrm{s,t}}$)

a: For each X_s that belongs to NURBS patch **S**, compute (s, t) using point inversion.

4: **Test points computation.**

a: For each common edge compute M_e as indicated in Eqn. 4.21.

b: Deploy test points linearly along each common edge.

5: **Computation of constraint matrix C.**

a: Compute Jacobian for each constraint equations as given in the Eqn. 4.14.

b: Assemble constraint matrix **C** as indicated in the Eqn. 4.16.

c: Assemble $\mathbf{C}_{sat}$ and $\mathbf{C}_{dev}$ using linear and non-linear constraints respectively.

6: **CAD geometry update.**

a: Compute Ker(**C**) and Ker($\mathbf{C}_{sat}$) using SVD as indicated in Eqn. 4.17.

b: Read $\delta\alpha$ from optimiser.

c: Perturb control points using tangent step as given in the Eqn. 4.18.

d: Compute constraint deviation δG_{dev}.

7: **while** δG_{dev} larger than δG_T **do**

e: Perturb control points using normal step as given in the Eqn. 4.26.

f: Update δG_{dev}, $\mathbf{C}_{dev}$

8: **Computational Mesh Update.**

a: Compute $X_{s,new}$ by using Eqn. 4.5 with parameters (s, t) obtained from Step 3 and perturbed control points $\mathbf{P}_{new}$ from Step 7.

b: Compute δX_v using IDW as indicated in the Eqn. 3.25 and update $X_{v,new}$.

Output: Updated CAD model in STEP format and updated computational mesh.

more potential for design improvement. The adaptive refinement consists of two steps which includes Refinement trigger and Refine.

4.9.1 Refinement trigger

Refinement trigger monitors the design progress and determines when to terminate the current design space level. This step has to be performed automatically without designer in the loop. Simply one can run the optimisation upto a pre-determined number of design iteration. Initial we have performed 10 design iterations for each level and observed sufficient acceleration in the design process [65]. However, this approach is case dependent and user require a prior knowledge of the convergence behavior for a design problem.

In this work, the refinement trigger proposed by Anderson et al [11] is used which triggers the refinement when the rate of convergence of the objective function with respect to the design iteration falls below some fraction of the maximum attained in the current design level.

$$\frac{\triangle J_i}{max(\triangle J_d)} < \epsilon_d \tag{4.31}$$

where $\triangle J_i = J_{i-1} - J_i$, d is the design space level and $\epsilon_d = 0.1$ is a cutoff parameter. Anderson et al. [11] pointed out that the refinement trigger shown in Eqn. 4.31 is not suitable for all cases. For example, in a highly non-linear region of the design space, the optimiser may take some time to obtain a faster design improvement hence Eqn. 4.31 may triggered too early and the design space may not be exploited properly.

4.9.2 Refine

This step computes the suitable knot value $\tilde{s}$ and $\tilde{t}$ to insert in the both s and t parameter direction respectively. Zingg et al. [52] used knot insertion algorithm to refine B-spline volumes in shape optimisation and proposed to insert a new knot arbitrarily in each knot span of a knot vector. Martins et al. [55] also used knot insertion to refine FFD frame and insert a new knot in the middle of each knot span. This may be suitable for simple geometries however for a complex geometry with large number of patches inserting a knot in each knot span without any additional

measure may leads to the addition of unnecessary control points on the design surface. Therefore, a suitable adaptation criterion is needed to precisely enrich the control point distribution locally on the design surface.

Mesh mapper module computes parametriccoordinates (s, t) to each mesh point and assign a unique patch ID that it belongs to. Once the refinement trigger terminates the current design level, the magnitude of the adjoint node sensitivities ($\mathcal{G} = \frac{\partial J}{\partial X_s}$) can be used to identify the region of interest to refine control points. To raise the regularity of the computed gradients, an additional explicit weighted Laplacian smoothing method is used. More details about this can be found in Sec. 4.11. Thus the smoothed gradient ($\bar{\mathcal{G}}$) is obtained as,

$$\bar{\mathcal{G}}^{y+1} = \mathcal{G}^{y} + \beta \mathcal{U}(\mathcal{G}) \tag{4.32}$$

y is the number of smoothing iteration, β is the smoothing parameter and $\mathcal{U}$ is the umbrella operator. Then the surface mesh points in each patch with large magnitude of the smoothed adjoint sensitivities are identified and assembled in a descending order. Then for each patch, three different samples of mesh points (z_1, z_2, z_3) are selected in total of n_m number of surface mesh points in a patch ($z_1 = \frac{n_m}{2}, z_2 = \frac{n_m}{4}, x_3 = \frac{n_m}{8}$). Then the knot value to be inserted in a patch is calculated as the average value of parametriccoordinates s and t of the selected mesh points as given in Eqn. 4.33 and 4.34. As a result, three different sets of new knot values for each patch are obtained. They are, $(s_1^\kappa, t_1^\kappa), (s_2^\kappa, t_2^\kappa), (s_3^\kappa, t_3^\kappa)$.

$$\tilde{s}_1^\kappa = \sum_{i=0}^{z_1} \frac{s_i^\kappa}{z_1}, \quad \tilde{s}_2^\kappa = \sum_{i=0}^{z_2} \frac{s_i^\kappa}{z_2}, \quad \tilde{s}_3^\kappa = \sum_{i=0}^{z_3} \frac{s_i^\kappa}{z_3} \tag{4.33}$$

$$\tilde{t}_1^\kappa = \sum_{i=0}^{z_1} \frac{t_i^\kappa}{z_1}, \quad \tilde{t}_2^\kappa = \sum_{i=0}^{z_2} \frac{t_i^\kappa}{z_2}, \quad \tilde{t}_3^\kappa = \sum_{i=0}^{z_3} \frac{t_i^\kappa}{z_3} \tag{4.34}$$

κ is the patch ID, z_1, z_2, z_3 are the number of selected mesh points used in the average and n_m represents the total number of mesh points in a patch. Knot insertion with these set of knot values creates three refined control nets without modifying the geometry. Note that there exists an infinite number of possibilities, however in this work only three refined control nets are obtained at the end of the each design level. To determine which updated control net to choose for the next design level, the

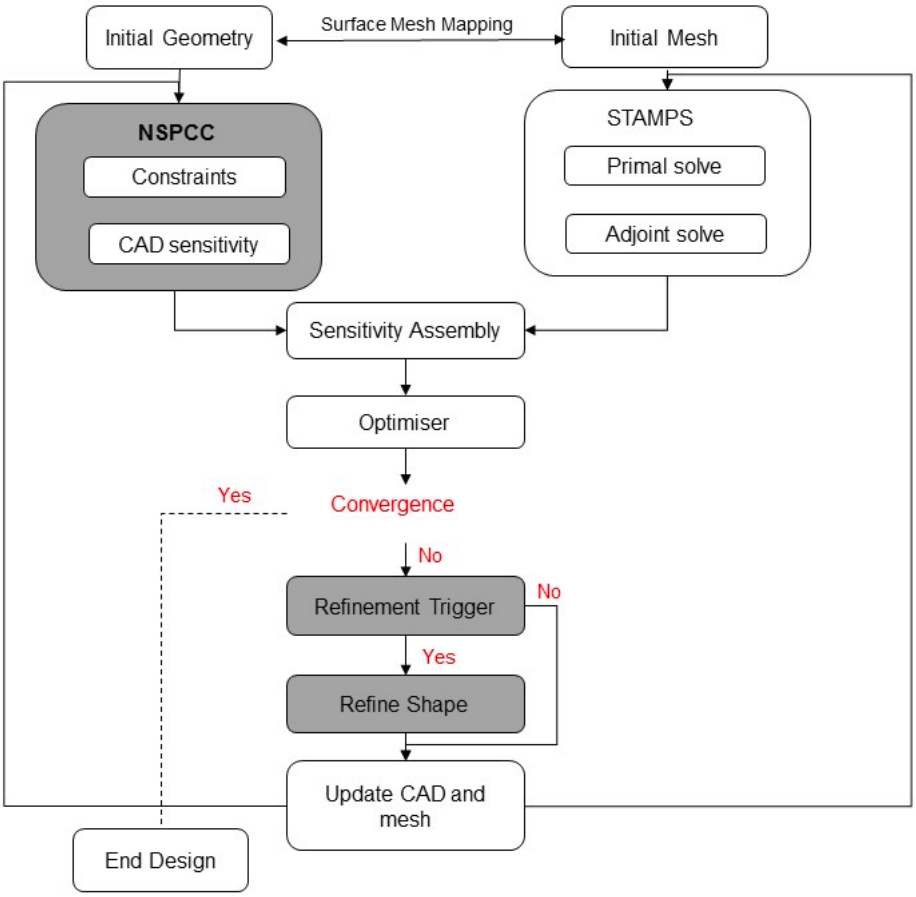
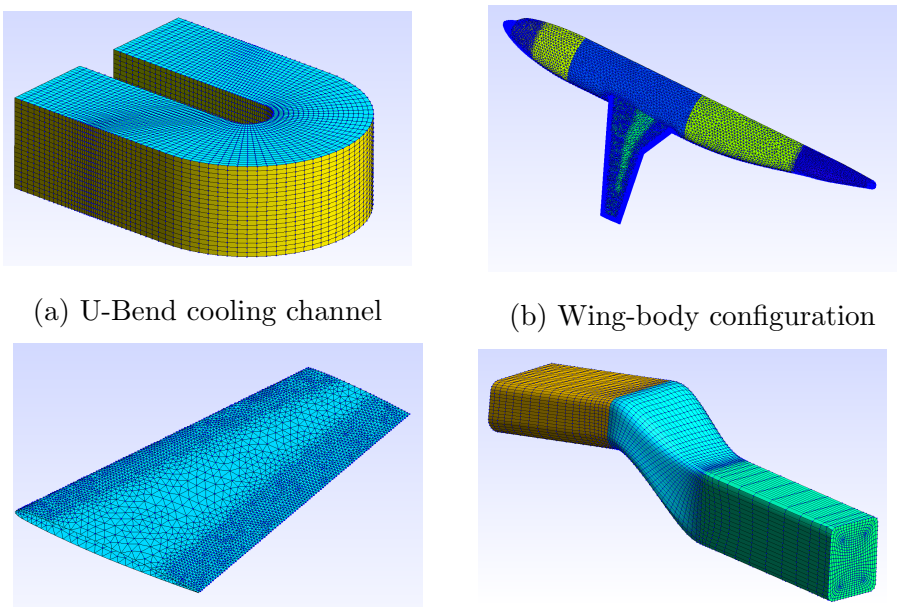


Figure 4.12: Shape optimisation work flow with the adaptive NSPCC parameterisation method

adaptation metric proposed by Martins et al. [55] is used.

$$AM = \frac{1}{2}\sum_{i=1}^{N}\left(\frac{\partial J}{\partial \mathbf{P}_i} - \sum_{j=1}^{N_g}\frac{\partial G_j}{\partial \mathbf{P}_i}\right)^2 \tag{4.35}$$

where N represents the total number of control points $\mathbf{P}_i$, N_g is the total number of geometric constrains including G_0 and G_1. $\frac{\partial G_j}{\partial \mathbf{P}_i}$ represents the Jacobian of the each constraint equation j with respect to each control point P_i. This metric is evaluated for each control net and choose the one which has larger adaptation metric value. Note that gradient smoothing is performed only to determine the high sensitivity region of interest and for optimisation the original node sensitivity ($\mathcal{G}$) is projected onto the NURBS control points. Figure 4.12 shows the work flow of the shape optimisation with the adaptive NSPCC parameterisation method.



(a) U-Bend cooling channel

(b) Wing-body configuration

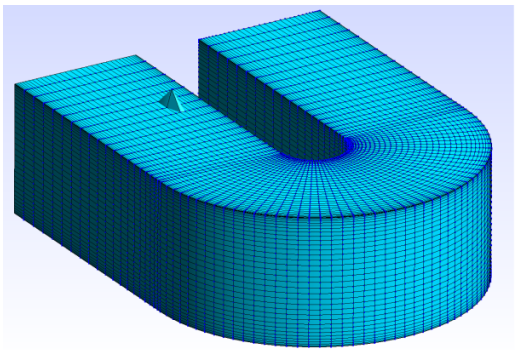
(c) Onera M6 wing

(d) S-bend

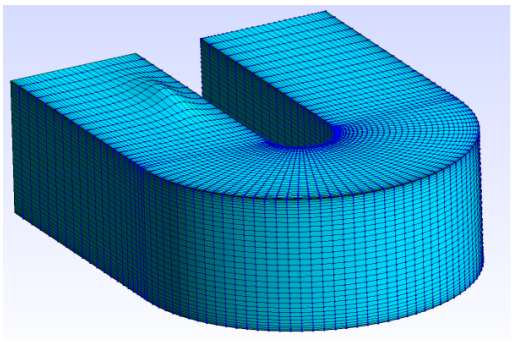
Figure 4.13: Surface mesh of various geometries used for node-based parameterisation

4.10 Adaptive Node-based Parameterisation

The node-based method derives parameterisation directly from the computational mesh employed for simulation and considers normal displacements of the surface grid nodes as design variables [104, 61, 87, 27, 127, 68, 57]. Figure 4.13 shows some of the examples of the surface mesh used for the node-based parameterisation method. Design surfaces of a complex geometry contains thousands of surface nodes, hence node-based method offers richest design space for shape optimisation. However this method includes odd-even oscillatory shape modes in the design space which can lead to undesirable or non-manufacturable shapes in the design process. Therefore additional surface regularisation method is required to filter out high frequency shape modes. Figure 4.14 shows the effect of perturbing a single node independently with and without the surface regularisation method. In the node-based method, surface mesh points are not moved independently, through smoothing, mesh points surrounding a perturbed node are also moved to generate a smooth shape. Figure 4.15 compares an updated computational mesh in the design loop with and without the surface regularisation method.

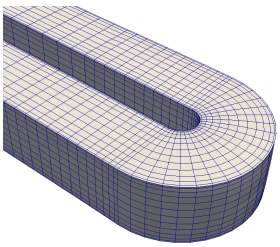


(a) Without surface regularisation

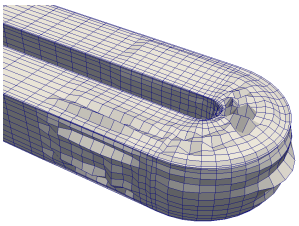


(b) With surface regularisation

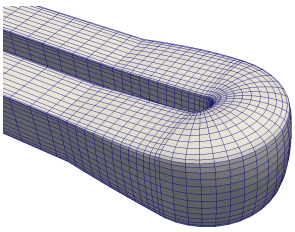
Figure 4.14: Effect of perturbing a single node with and without surface regularisation



(a) Initial mesh



(b) Without smoothing



(c) With smoothing

Figure 4.15: Effect of surface regularisation in shape deformation

4.11 Smoothing the shape displacements

As discussed in Ch. 2, one can use the surface regularisation method either to smooth the gradient as proposed by Jameson et al. [132, 63] or smooth the shape displacement as suggested by Jaworkski et al. [64]. In this work, smoothing is performed directly on the shape displacement to obtain a smooth shape. The resultant smoothed shape displacement using the explicit Laplacian method is given as,

$$\delta\tilde{X}_s^{n+1} = \delta X_s^n + \beta\mathcal{U}(\delta X_s) \tag{4.36}$$

where δX_s and $\delta\tilde{X}_s$ are the shape displacements before and after smoothing, n represents the number of smoothing iteration, β is the smoothing coefficient and $\mathcal{U}$ is the scale dependent umbrella operator which is written as,

$$\mathcal{U}(\delta X_s) = \frac{1}{\sum_i w_i}\sum_i w_i(\delta Q_i - \delta X_s) \tag{4.37}$$

$$w_i = \|\delta X_s - \delta Q_i\|^{-1} \tag{4.38}$$

where w_i are the weights associated with the Laplacian operator. In this work, surface mesh nodes are allowed to move in the normal direction. Therefore after smoothing, the shape displacements are projected along the surface normal direction using Eqn. 4.39,

$$\delta X_\perp = (\delta X_s \cdot \hat{S}_n)\hat{S}_n \tag{4.39}$$

where $\delta X_\perp$ are the surface displacements along the normal direction and S_n is the unit surface normal at each surface mesh node.

4.11.1 Adaptive Surface Regularisation

Jaworski et al. [64] compared both explicit and implicit Sobolev smoothing approaches in the shape optimisation loop and pointed out that explicit smoothing is most effective for damping high frequency modes. However care must be taken to choose the appropriate value for smoothing coefficient (β_e) and number of smoothing iterations (n). As n grows, explicit smoothing completely removes high frequency

shape modes in the design space and exhibits aggressive damping on middle modes as well while this may be desired in cases where only low modes are relevant. However this will not work for inverse design where all shape modes may be needed to capture the desired shape [128, 132].

Therefore in this work, a simple adaptive explicit smoothing approach is proposed in which the design process is started with a large value of n in which low-frequency shape deformations are handled and as the design progress the number of smoothing iterations are reduced to include high-frequency shape deformations in the design space.

4.12 Design Scaling

At the end of the each optimisation cycle, the computational mesh needs to be updated based on the design perturbations given by the optimiser. If the design perturbations are larger, surface regularisation methods discussed in Section 4.10 may not filter the high frequency shape modes with a sufficient number of smoothing iterations. As a result, mesh deformation algorithm may create a low quality computational mesh or in the worst case the updated volume mesh may not matched with the updated shape. In order to resolve this issue, Christakopoulos et al. [27] proposed a scaling method for each design variable to restrict the perturbations based on edge lengths. The shape displacements are scaled using Eqn. 4.40,

$$\delta X_{S,i} = \sigma^i \cdot \delta X_{\perp,i} \tag{4.40}$$

$\delta X_{S,i}$ and $\delta X_{\perp,i}$ are the shape displacements before and after scaling $i = (1, 2, ..N_s)$, N_s is the number of surface mesh nodes. Scaling factor (σ_i) can be obtained using Eqn. 4.41. It is important to note that design scaling may affect the design convergence however it restricts the perturbations for each surface mesh node.

$$\sigma_i = \xi \frac{\sum_{j=1}^{N_e} L_j}{N_e} \tag{4.41}$$

where $\xi \in [0, 1]$ is a user defined global scaling factor, L_j is the length of the edge j connected to the node i and N_e is the total number of edges connected to the node i.

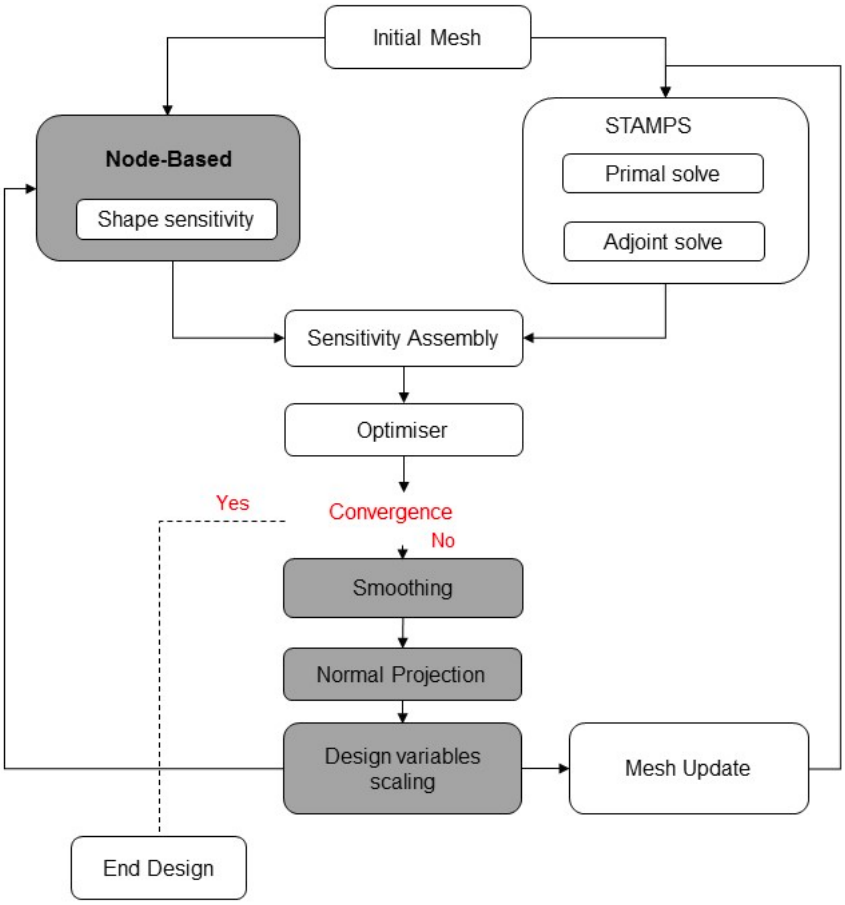


Figure 4.16: Shape optimisation work flow with node-based parameterisation

4.13 Primal Algorithm with Node-based Parameterisation

The primal work flow with Node-based parameterisation is shown in Fig. 4.16. The primal algorithm is written in the Fortran language and source transformation AD tool TAPENADE is used to differentiate the block by block of the primal algorithm in the reverse mode. By combining each individual blocks the adjoint sensitivity as shown in Eqn. 3.28 can be modified as,

$$\bar{\alpha} = \left(\frac{\partial J}{\partial X_s}\right)^T = \left(\frac{\partial \tilde{X}_s}{\partial X_s}\right)^T \left(\frac{\partial X_\perp}{\partial \tilde{X}_s}\right)^T \left(\frac{\partial X_\perp}{\partial X_S}\right)^T \left(\frac{\partial X_S}{\partial X_v}\right)^T \left(\frac{\partial J}{\partial X_v}\right)^T. \tag{4.42}$$

4.14 Summary

In this chapter, the adaptive shape parameterisation method developed in this research work are presented. Firstly, a brief introduction about the boundary representation and the different methods available for manipulating NURBS patches such as knot insertion, degree elevation are presented. Then the details about the test point approach for handling geometric constraints are discussed. The proposed adaptive refinement algorithm which includes the sensitivity driven adaptation metric are presented. Finally, a simple adaptive explicit surface regularisation method is presented for design space exploration.

Chapter 5

Constrained Wing Optimisation

5.1 Introduction

In chapter 4, I have presented the NURBS-based Parameterisation with Complex Constraints (static-NSPCC) method to deform multiple NURBS patches in the design process with user-defined geometric constraints (G_0 and G_1). In this chapter, the NSPCC method is extended to handle manufacturing constraints in aerodynamic shape optimisation. When performing aerodynamic shape optimisation for an industrial application, a designer needs to impose structural constraints to satisfy manufacturing requirements of the design. For example build space or packaging constraints are essential if large number of components are needs to assembled together with the optimised shape in the later product development cycle. To maintain structural integrity the optimised shape should be assembled within the available space defined by the other components in the assembly.

Lift constrained aerodynamic shape optimisation of a wing does not provide sufficient mechanism to prevent the reduction of internal volume which may require to accommodate wing box or fuel tank and/or other structural parts such as ribs, spars, stiffeners etc. In aerodynamic shape optimisation of turbomachinery blades, it is essential to maintain minimum leading and trailing edge radius and thickness constraints to maintain structural strength and to accommodate cooling channels to prevent it from overheating. Therefore the shape parameterisation method needs to handle such manufacturing constraints efficiently without cumbersome preprocessing setup in an gradient-based shape optimisation process. For certain cases,

this kind of constraints can be handled by imposing maximum and/or minimum bounds on the design variables [136]. On the other hand, if the engineering parameters such as section thickness, leading and trailing edge radius are taken as design variables, constraints could be implemented as an inequality constraints. To impose manufacturing constraints within the CAD-based NSPCC approach, the same test point approach is adopted. Effectiveness of the developed shape optimisation tools in this work is compared with the currently available open-source shape optimisation tools. SU2 is an open-source shape optimisation tools written in C++ used to compute flow and adjoint fields for shape optimisation. In addition to that, Free-Form Deformation (FFD) method is also available with some of the geometric capabilities such as internal volume and section thickness computations to handle geometric constraints. In this chapter, efficiency of the developed tools are tested with the open-source SU2 tools. For this purpose, constrained shape optimisation of the ONERA-M6 wing is considered for this study. This test case is available in SU2 website as a tutorial case. Therefore in this chapter, two parametrisation methods static NSPCC and FFD are applied to minimise the drag of the ONERA M6-wing with lift and geometric constraints under transonic flow conditions.

It should be mentioned that the aim of this study is to compare the effectiveness of the developed tools in the QMUL-CFD optimisation group with the currently available state of the art open source SU2 shape optimisation tools. This study allows us to access the capabilities of the tools developed in this work with the open source SU2 tools using widely tested optimization test case M6-wing. Efforts are underway to achieve consistent comparisons between different parameterisation approaches using similar CFD and optimization tools. This will be considered in my future work.

This chapter is organised as follows, shape parametrisation of the ONERA M6-wing using NSPCC and FFD methods are presented in Sec. 5.2. Section 5.3 explains the test-point approach for handling wing-box and trailing edge thickness constraints. In Sec. 5.4, brief details about SU2 tools employed in this work are presented. In Sec. 5.5, details about the computational setup, grid convergence study, verification and validation study performed using M6-wing are presented. Results of the constrained optimisation are presented in Sec. 5.6.

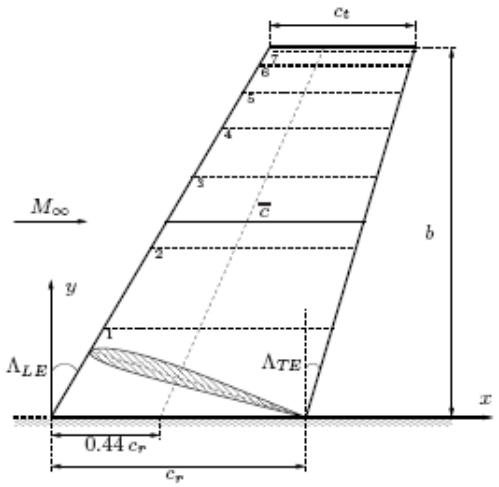
5.2 Shape Parameterisation

In the NSPCC approach, control points of a NURBS patches are used to deform a shape in the design process. In general, the Boundary Representation (BRep) of a baseline shape exported by a CAD system often depends on the local curvature of a shape. For example, region with a large curvature may contain large density of control points and a flat region may contain sparse control points. However, during surface conversion from STL (surface tessellation) to the standard boundary representation, a large number of intersecting and/or trimmed NURBS patches can be generated. In addition to that density of the control points may be too richer than the surface mesh required for a CFD simulation. As discussed in Chapter 6, finer control points affects the rate of design convergence and does not restrict the shape modes captured in the design space hence can provide best achievable design. However this is not always the case, a finer control net can also provide multiple local minima, with a gradient-based optimiser the design process can converge to any local minima. In addition to that additional surface regularisation may also be required if the control net is too richer than the surface mesh required for a CFD simulation.

Therefore as a first step, re-parameterisation is required to achieve best approximation to the baseline shape. Developing algorithm to obtain exact fit is a challenging task and is not considered in this work. Since the shape will be altered by the design the exact fit is not desired. However care must to taken to capture the design intent, for example local streamwise curvature of the wing which determines the pressure profile along the wing. Salvatore et al. [15] proposed a re-parameterisation algorithm using differentiated OCCT kernel. In this work, reparameterisation tool offered by the commercial CAD system SOLIDWORKS is employed to obtain minimum density of control points on the design surface.

5.2.1 ONERA M6 Test Case

The ONERA M6 wing is s semi span wing designed using symmetric airfoil ONERA D section with a swept angle of 26.7°. Geometric details of the ONERA M6 geometry is shown in Fig. 5.1. CAD model of the M6 wing taken from the Cornell university



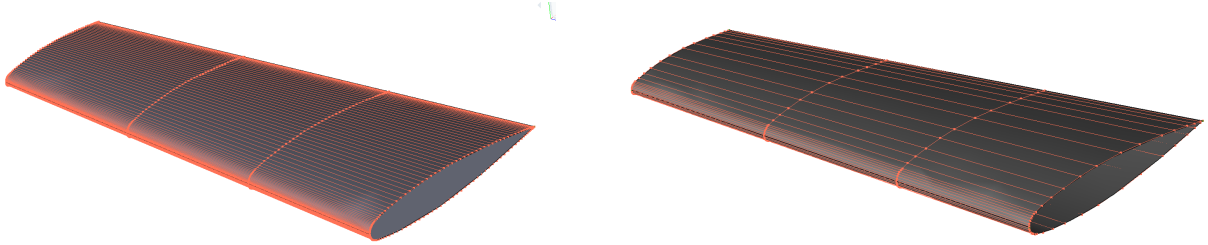
Geometry	
AR	3.8
b	1.196 m
Λ_{LE}	30°
Λ_{TE}	15.8°
λ	0.562
$\overline{c}$	0.646 m
S	1.506 m^2

Figure 5.1: ONERA M6 wing geometry

ANSYS tutorial website [1] is shown in Fig. 5.2a. This BRep consists of two bi-cubic NURBS patches with one on top and another one on bottom surface of the wing. Each patch contains 145×4 control net in which 145 control points are used along the chordwise direction and 4 control points are used along the spanwise direction. Therefore in total this BRep consists of 1160 control points. Density of the control points near the leading and trailing edge is quite large hence may generate highly oscillatory shape modes in the design process which may also affect the design convergence. Therefore reparameterisation step is performed using commercial CAD system SOLIDWORKS to obtain minimum density of control points on the design surface. During this step, G_1 continuity is maintained between the top and bottom surface of the leading edge. The reparameterised M6 wing is shown in Fig. 5.2b which consists of two bi-cubic NURBS patch with 168 control points and each patch contains 21×4 control net.

5.3 Wing-Box Constraints using NSPCC

Lift constrained aerodynamic shape optimisation of a wing does not provide sufficient mechanism to prevent the reduction of internal volume which may require to accommodate wing box or fuel tank and/or other structural parts such as ribs, spars, stiffeners etc. To enforce geometric continuity between patches such as G1 (tangency) or G2 (curvature), constraint equations are numerically evaluated at



(a) Baseline M6 wing

(b) Reparameterised M6 wing

Figure 5.2: BRep of the baseline wing with 145×5 control net and reparameterised M6 wing with 21×4 control on each patch

testpoints distributed along the patch interfaces. To impose wing-box constraints within the CAD-based NSPCC framework the same test point approach used for maintaining continuity constraints are extended in this work and the efficiency of the method is tested by minimising the inviscid drag of the ONERA M6 wing. Figure 5.3 shows the test points for imposing the wing-box and trailing edge thickness constraints. A total of 12 set of test points are deployed on the M6-wing evenly along the spanwise direction. Among these 8 sets of test points are used for wing-box constraints and 4 pairs of test points for TE thickness constraints. The number of test points and their distribution along the spanwise directions are determined to be approximately match the control points distribution of the NURBS patches.

The wing-box and TE thickness constraint functions (T^n) at n^{th} design step is written as,

$$T^n(i) = 1 - min\left(\frac{t_i^n}{t_i^0}, 1\right), \tag{5.1}$$

where the thickness function (t_i^n) at n^{th} step is defined as,

$$t_i^n = \parallel X^n(u_i) - X^n(l_i) \parallel \quad \text{for } i = 1 \text{ to } n_t, \tag{5.2}$$

where $X(u)$ and $X(l)$ are the surface points on top and bottom NURBS patch respectively, n_t is the total number of pairs of test points, t_i^0 is the thickness of the baseline design. The thickness constraints T^n are imposed as equality constraints:

$$T^n(i) = 0 \text{ for } i = 1, n_t. \tag{5.3}$$

This implies that wing-box thickness and TE thickness cannot be reduced below the baseline values throughout the optimisation process. In addition to that G_1

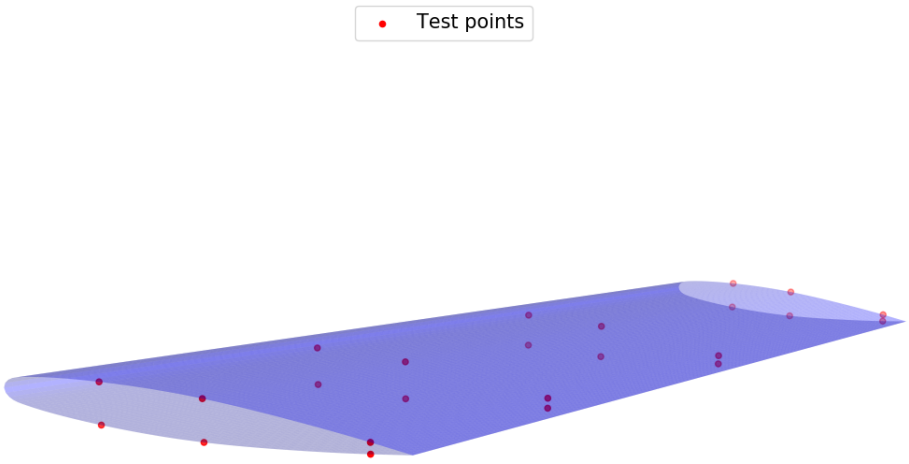


Figure 5.3: Test points for wing-box and trailing edge thickness constraints

continuity is imposed at the leading edge. For ease of notation Jacobian matrix $\mathbf{C}$ as given in Eqn. 4.16 is now rearranged and Jacobian of both geometric and thickness constraints are combined together and assembled as,

$$\mathbf{C}= \begin{bmatrix} \mathbf{G}_0^j \\ \cdots \\ \mathbf{G}_1^j \\ \cdots \\ \mathbf{T}^i \end{bmatrix}, \tag{5.4}$$

where j is the edge index and i is the pair of test points for thickness constraints. The design space is the kernel of this Jacobian which is evaluated using a Singular Value Decomposition as discussed in Ch. 4. The design variables then effectively become the vectors associated with non-singular values. Again there is an adjustable parameter in the form of the threshold value for singular values.

5.4 Free-Form Deformation using SU2 tools

Free-Form Deformation method is the third parametrisation method to be considered in this work. This method require the definition of auxiliary hexahedral volume grids that need to be snapped to the geometry to preserve features. This can be

achieved by the use of trivariate Bezier volume [115] and later it has been extended to include B-splines [147] and NURBS volume [59] to utilize local shape modification properties which are found to be more useful to capture superior designs in shape optimisation [112, 133]. More details about FFD approach can be found in [111].

To demonstrate the effectiveness of the NSPCC approach, current state of the art open-source FFD tools provided by SU2 are also employed to perform lift constrained drag minimisation of the ONERA M6 wing under inviscid transonic flow conditions. These tools has been widely used to create smooth deformation in aerodynamic shape optimisation of wing. Optimisation setup used in this work is similar to the tutorial case presented in the SU2 website. Computational mesh presented in the ONERA-M6 wing SU2-tutorial case is relatively coarse and to ensure valid comparison, the finer mesh (M3) presented in Sec.5.5 is used with the SU2-tools to perform shape optimisation using FFD approach.

Figure 5.4 shows the initial wing geometry with the FFD box constructed using SU2 tool. The distribution of control points on the FFD-box are uniform along both chord-wise and span-wise direction and flat wing tip with sharp trailing edge is used in this work. Control points of the FFD box are allowed to deform only in the z (normal) direction. Note that the root twist is not allowed to vary and the optimization is carried out at a fixed angle of attack of 3.06°. The goal of this optimisation is to minimize the coefficient of drag while imposing lift and wing section thickness constraints. In SU2, geometric quantities such as internal volume of a wing and section thicknesses can be computed and used as constraints during the shape optimisation process. In this work, section thicknesses are computed at five different locations along the span-wise direction such as $y/b = 0.0, 0.2, 0.4, 0.6, 0.8$ and constrained to be greater than or equal to the baseline values.

Software's used

Following SU2-tools are used in this test case to perform FFD-based shape optimisation:

- SU2_CFD is a RANS-based compressible flow CFD solver. It uses a Finite Volume Method with an edge-based data structure. In this work, this module is used to perform EULER flow simulations.

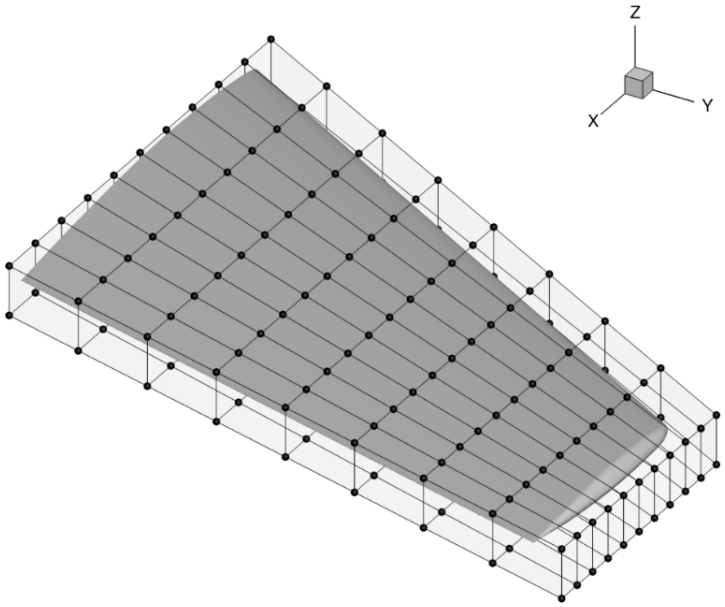


Figure 5.4: FFD-box of the initial wing taken from SU2 tutorial case

- SU2_CFD_AD is an adjoint solver. Both continuous and discrete adjoint methods are implemented in this module. In this work, discrete adjoint approach is employed to compute adjoint surface sensitivities.

- SU2_GEO is a geometry module used to evaluate internal volume of a wing and thickness of a wing section and also used to compute its gradients. In this module, constraint Jacobian and shape sensitivities are computed using finite difference method.

- SU2_DOT_AD is a gradient-projection code used to compute partial derivative of the objective function of interest such as drag, lift with respect to the shape design variables. This module project the adjoint surface sensitivities into the FFD control lattice to compute the gradient of the objective function with respect to the design variables through a dot product operation between CFD $\frac{\partial J}{\partial X_s}$ and and shape sensitivities $\frac{\partial X_s}{\partial \alpha}$.

- SU2_DEF is a mesh deformation module used to deform the volume grid. After perturbing the surface grid using FFD approach, SU2_DEF module deform the volume grid using linear elasticity method.

- Apart from these tools, shape_optimization.py a python framework is also available to integrate all the executable SU2 binaries to perform automated

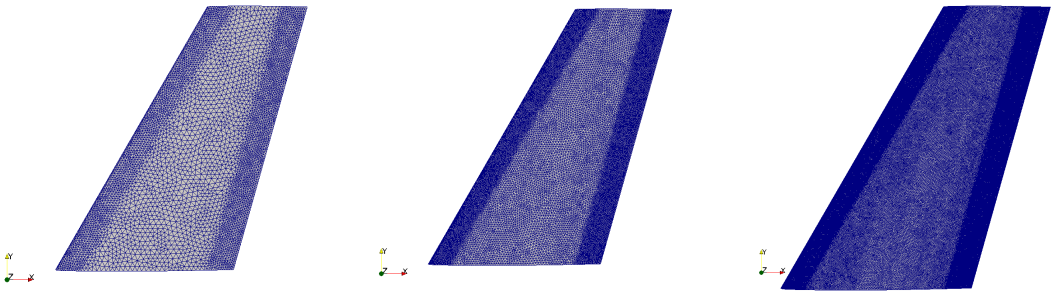
shape optimisation process. In this work, SLSQP optimiser with default settings available in shape_optimization.py script is used to drive the design process.

5.5 Grid Convergence Study and Validation

In this section, the computational setup used in the drag minimisation of the M6 wing under inviscid flow conditions, grid convergence study and CFD model validation are presented. The freestream conditions considered in this work for validation and subsequent optimisation are similar to the standard benchmark conditions set out in the NPARC Alliance CFD Verification and Validation programe [3], and are listed below:

- Freestream Mach number, $M_a = 0.84$.
- Freestream Pressure, $P_\infty = 101325\ Pa$
- Angle of attack = 3.06°
- Freestream Temperature, $T_\infty = 288.15K$

To perform grid convergence study a sequence of 3 computational grid were generated ranging from 56676 to 540309 unstructured cells. The grids are generated using ANSYS-mesher. The farfield is configured using C-O topology and extends to 100 Mean Aerodynamic Chord lengths in all direction from the wing surface. Surface meshes of all the three grid levels are shown in Fig. 5.5. Total number of nodes in all grid levels used in this work is given in Table 5.1. Simulations were performed on all the meshes with second order accuracy using Roe's flux functions and Venkatakrishnan's limiter. Slip wall on the wing surface, farfield and symmetry on the outer domain were used as boundary conditions. The comparison of the coefficient of pressure computed using an in-house flow and adjoint solver STAMPS with the experimental data is given in Figure 5.6. Values of the coefficient of pressure are taken at different locations along the spanwise direction of the M6 wing. The location are highlighted in Figure 5.1. The finer grid (M3) fairly captured both location and strength of the lambda shocks along the spanwise direction. A small



(a) Coarse (M1) (b) Medium (M2) (c) Fine (M3)

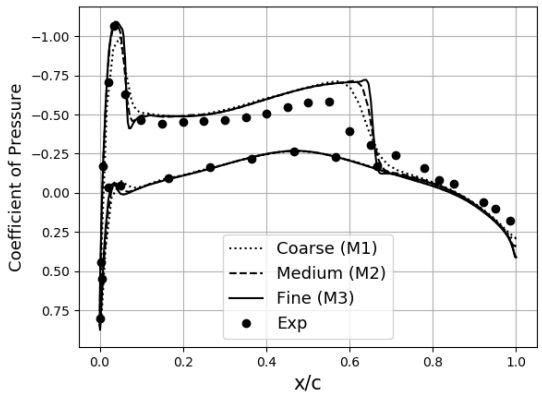
Figure 5.5: Surface meshes of all the three grid levels.

Grid	**Total number of nodes**	**Total number of surface nodes**
Coarse (M1)	56676	7920
Medium (M2)	171501	27493
Fine (M3)	540309	99480

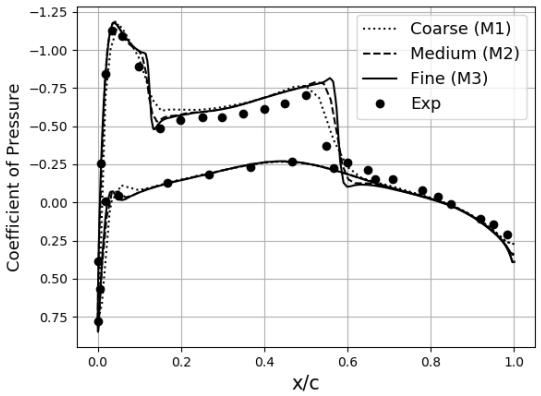
Table 5.1: Grid sizes

discrepancy between the computed and experimental results is observed near the root section of the wing. This behavior is also noted with the other simulation results obtained on highly refined grids. Martins et al. [75] pointed out that this may be due to the presence of the splitter plate in the physical setup and other wall effects in the wind tunnel test section which are not computationally modelled.

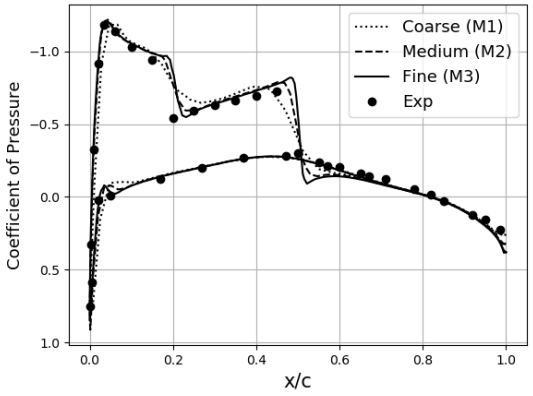
Additionally the open source SU2 solver is also used to verify the capabilities of the STAMPS solver. In SU2 solver, Jameson-Schmidt-Turkel scheme (JST) central scheme is used with Venkatakrishnan's limiter. Comparison of coefficient of pressure between STAMPS and SU2 solver computed using fine grid (M3) is shown in Fig. 5.7. Figure 5.8 shows the comparison of the coefficient pressure taken at different locations along the spanwise direction of the M6-wing. Results shows that the overall features of the lambda shock is fairly matched between the SU2 and STAMPS solver.



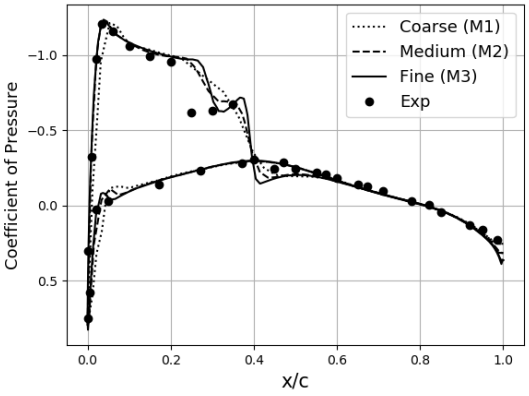
(a) $\frac{y}{b} = 0.20$ (*location* 1)



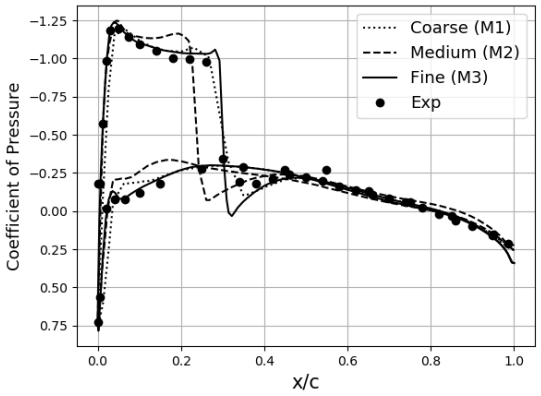
(b) $\frac{y}{b} = 0.44$ (*location* 2)



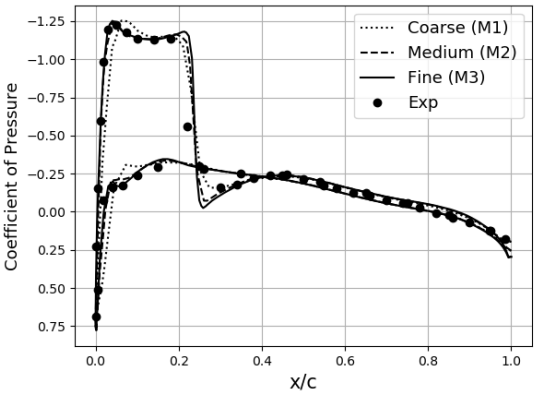
(c) $\frac{y}{b} = 0.65$ (*location* 3)



(d) $\frac{y}{b} = 0.80$ (*location* 4)

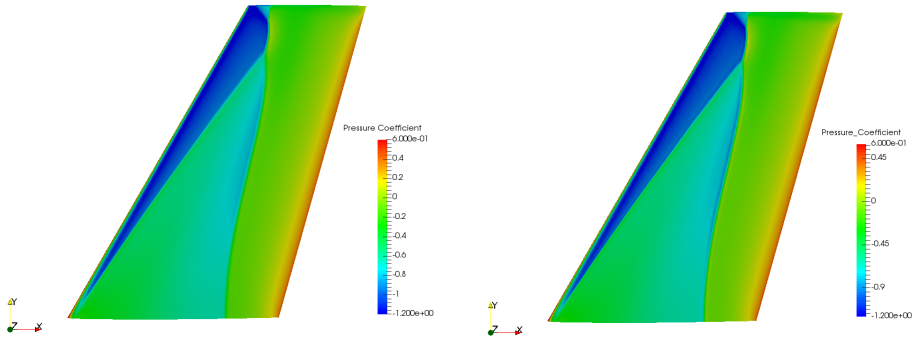


(e) $\frac{y}{b} = 0.90$ (*location* 5)



(f) $\frac{y}{b} = 0.95$ (*location* 6)

Figure 5.6: Coefficient of pressure for each grid level compared with experimental data



(a) STAMPS (b) SU2

Figure 5.7: Comparison of coefficient of pressure between STAMPS and SU2 computed using fine grid (M3)

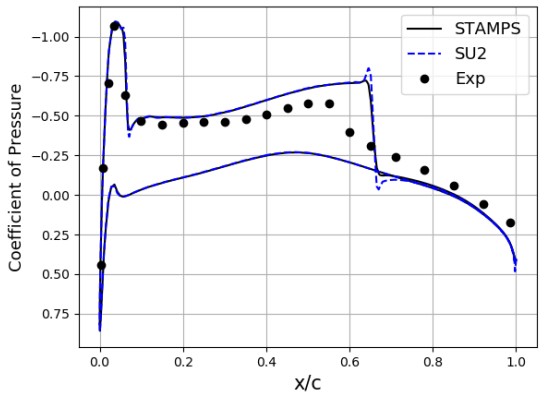
5.6 Shape Optimisation

Shape optimisation tools developed in this work and the state of the art open-source SU2 tools are used to minimise inviscid drag of the ONERA M6 wing under transonic flow conditions with constant lift and thickness constraints. Brief overview of the problem formulation and performance of the both methods are discussed in this section.

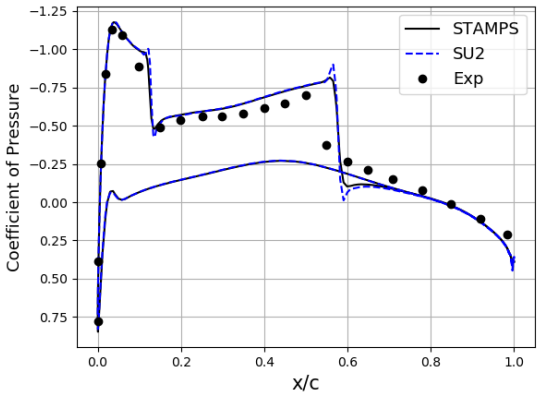
5.6.1 Problem Formulation

Optimisation-1: FFD and SU2

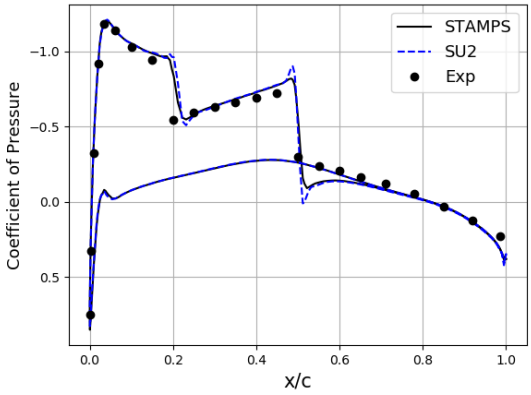
The optimisation problem is the inviscid drag minimisation of the ONERA M6 wing with sharp trailing edge under transonic flow conditions. Boundary conditions are similar to the benchmark conditions given in the NPARC website [3]. Details are presented in Sec 5.5. In this optimisation, FFD parametrisation method is used to parameterise the baseline ONERA M6. FFD control grid is constructed using Bezier form of blending function. Both Bezier and B-Spline forms are available in SU2. In total, FFD control box consists of 198 control points in which 11 control points are placed along chord-wise direction, 9 control points are placed along span-wise direction and 2 of them are placed perpendicular to xy plane. Degree of the Bezier blending functions along each parameter direction u, v, w is $10, 8, 1$ respectively. During the optimisation, each control point is allowed to deform along the z (normal)



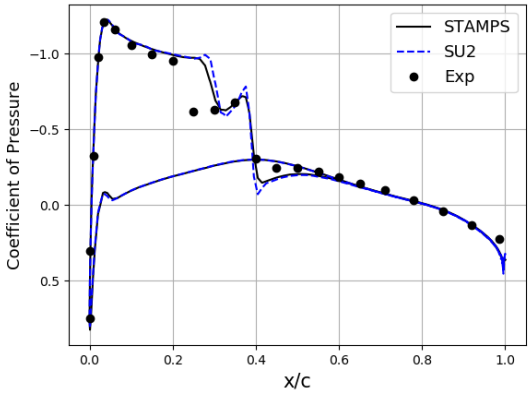
(a) $\frac{y}{b} = 0.20$



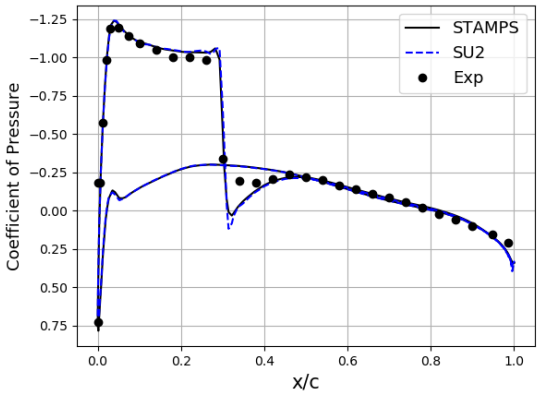
(b) $\frac{y}{b} = 0.44$



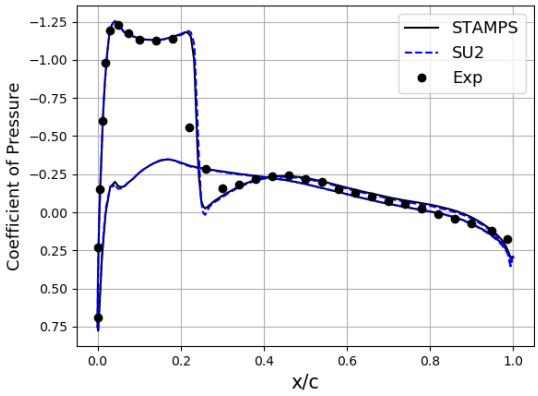
(c) $\frac{y}{b} = 0.65$



(d) $\frac{y}{b} = 0.80$



(e) $\frac{y}{b} = 0.90$



(f) $\frac{y}{b} = 0.95$

Figure 5.8: Comparison of coefficient of pressure between STAMPS and SU2 computed using fine grid (M3). SU2: $C_L = 0.2925, C_D = 0.012$, STAMPS:$C_L = 0.296, C_D = 0.013$

coordinate direction. Control grid formulation is similar to the SU2 ONERA M6 tutorial case

Both leading and trailing edge of the wing are allowed to move, while the angle of attack is fixed. Lift coefficient is constrained to be greater than or equal to its baseline value $C_L^* = 0.2925$ which is computed using SU2_CFD solver with fine-grid (M3). A total of five wing section thickness constraints are used, one at the root section and remaining four of them are distributed along the $20\%, 40\%, 60\%$ and 80% of wing span. For this case, Sequential Least Squares Programming (SLSQP) is used as the optimiser. SLSQP is a Sequential Quadratic Programming (SQP) method, designed for handling large number of equality and inequality nonlinear constraints. It uses Han–Powell quasi–Newton method with a BFGS update. More details about SLSQP algorithm can be found in [140]. To interface with SU2 tools and scipy SLSQP optimiser, a python optimisation script provided in the SU2 tutorial case is used. The optimisation problem considered is described below:

$$\begin{aligned} \underset{\alpha}{\text{minimize}} \quad & J(\alpha) = C_D(\alpha) \\ \text{subject to} \quad & C_L \geq C_L^* \\ & t_i \geq t_0, \quad i = 1, ..., 5 \end{aligned} \tag{5.5}$$

Static NSPCC and STAMPS

As similar to the above case, the objective is to reduce the coefficient of drag while maintaining the lift coefficient. In this case, both top and bottom surface of the ONERA M6 wing are parameterised using bi-cubic NURBS patch with 21 control points are placed along the chord-wise direction and 4 control points are placed along the span-wise direction. In total, BRep of the ONERA-M6 wing consists of 168 control points. Each control point is allowed to deform only in the z coordinate direction. In this case, constraints are handled differently, for example, wing-box and trailing edge thickness constraints are handled by using test points approach presented in Ch.5.3. A total of 12 thickness constraints are used, 4 of them are distributed along 30% of chord line and 4 of them distributed along 50% of chord line to ensure assembly of internal components such as wing-box. Trailing edge (TE) of the wing is fixed and 4 test points are distributed along 95% of chord line

to prevent cross over and reduction in the thickness of the TE. Lift coefficient is constrained to be greater than or equal to its baseline value $C_L^* = 0.296$ which is computed using STAMPS primal solver with fine-grid (M3) and handled by using penalty function method. Angle of attack is fixed during the optimisation. The optimisation problem considered for this case is described below:

$$\begin{aligned} \underset{\alpha}{\text{minimize}} \quad & J(\alpha) = C_D + \lambda(C_L - C_L^*)^2, \\ \text{subject to} \quad & t_i \geq t_0, \quad i = 1, ..., 12 \end{aligned} \tag{5.6}$$

where λ is a penalty coefficient. Table 5.2 shows a brief comparison of the main characteristics of both optimisation approaches employed in the test case.

5.7 FFD vs Static NSPCC

Optimisations are conducted using both the parameterisation methods using fine grid (M3) containing total $540k$ nodes, as the grid-independence study showed that this grid offered sufficient resolution (Sec. 5.5). In case-1, SLSQP optimiser is used and in case-2, BFGS optimiser is used to drive the design process. Figure 5.9 shows a comparison of the optimisation convergence history between the case-1 and case-2. In Table 5.3, comparison of drag reduction and number of design variables (N_{DV}) employed in each optimisation case are presented. In case-1 (FFD+SU2), primal and adjoint flow solutions are fully converged in each design step hence large perturbations can be achieved during the design process. However, in case-2 one-shot methodology is used to drive the design process [27]. Hence shape deformations are controlled by smaller step sizes which are monitored by the Armijo and Wolf based line search conditions. As a result, when compared with the NSPCC approach, the FFD approach achieved a very rapid decrease in the objective function value. However optimal shape obtained using the NSPCC approach achieves 22.3% reduction in the drag coefficient whereas FFD method achieves 20.2% reduction in the drag coefficient. This performance improvement may be caused by the presence of large number of design variables and orthogonal shape modes in the design space. The results of the drag minimisation optimisation are presented in Table 5.3. Large number of researchers [24, 105, 8, 25, 107, 45] pointed out that orthogonal shape modes regularize the design space and essential to achieve more robust shape optimisation.

Steps	FFD	Static NSPCC
CAD Geometry	SOLIDWORKS(.step)	
Control Grid Dimension	11 * 9 * 2 (Bezier FFD lattice)	21 * 4 (NURBS control net)
Design Variables	z-coordinates of the FFD lattice	Linear combination of the nullspace of the constraint matrix
Grid Generation	ANSYS Mesher-Fine grid (M3)	
Flow Solver	SU2_CFD	Primal mode STAMPS
Discrete Adjoint Solver	SU2_CFD_AD	Adjoint mode STAMPS
Gradient Computation (CFD sensitivities)	SU2_DOT	Sensitivity mode STAMPS
Shape Sensitivities	SU2_GEO (Finite difference)	Adjoint mode NSPCC
Constraint Jacobian	SU2_GEO (Finite difference)	Tangent mode NSPCC
Surface Deformation	FFD	Deformation mode NSPCC
Volume Deformation	SU2_DEF (Linear elasticity method)	Deformation mode STAMPS (Inverse Distance Method)
Optimiser	Scipy tools (SLSQP)	Scipy tools (BFGS)
Objective Function	Minimise: $J = C_D$	Minimise: $J = C_D + \lambda(C_L - C_L^*)^2$
Aerodynamic Constraint	C_L^*	C_L^*
Geometric Constraints	Section thickness	G_1 continuity, Wing-box and trailing edge thickness

Table 5.2: Description of the optimisation process using both FFD-SU2 and NSPCC-STAMPS.

Design	$\triangle C_D\%$	N_{DV}
FFD + SU2	-20.27%	198
NSPCC + STAMPS	-22.3%	486

Table 5.3: Optimisation results obtained using FFD and NSPCC

In the NSPCC approach, z-coordinates of the control points of NURBS patches are used to deform the shape and the design space is the kernel of the constraint Jacobian which is evaluated using Singular Value Decomposition. The design variables then effectively become the linear combination of the columns of the nullspace. The deformation modes in the nullspace are orthogonal to each other hence each shape mode in the nullspace corresponds to a unique set of control point perturbations which offers rich design space coverage for shape optimisation.

From Eqn. 4.18 the size of the design space is given by $N - r'$ in which N is the total number of control points allowed to deform in the design process and r' is the numerical rank of the constraint matrix. For this test case, number of control points is fixed throughout the optimisation which is $N = 160$. Value of r' depends on the chosen threshold frequency value σ_C, for this test case the chosen value is 10^{-9} which results 486 number of design variables at the end of the optimisation. On the other hand, in the FFD approach, z-coordinates of the FFD control points are used as design variables and the shape modes are not orthogonal to each other, hence it offer poor design space converge and may leads to a suboptimal solution in the design process [6]. Furthermore, the distribution of control points is uniform on the surface of the FFD-box hence large number of design variables are required for the optimiser to explore the design space. For example, clustering of control points are required in high curvature region to capture superior shape modes. In addition, the FFD-box is constructed using Bernstein polynomial, which provides global shape modification property. Therefore to achieve better performance, B-Spline or NURBS based FFD-box needs to be constructed around the M6-wing, this will be considered as future work.

Figure 5.10 shows the comparison of Mach number distributions between baseline and optimised M6-wing. In both optimisation case, strength of the lambda shock wave relative to the baseline geometry has been reduced significantly. However

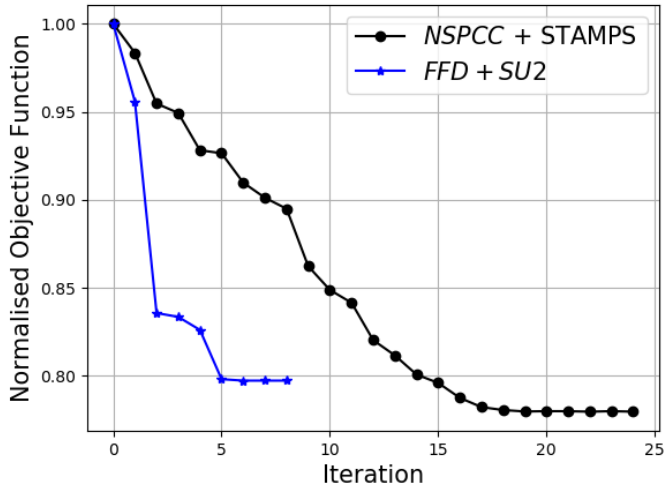
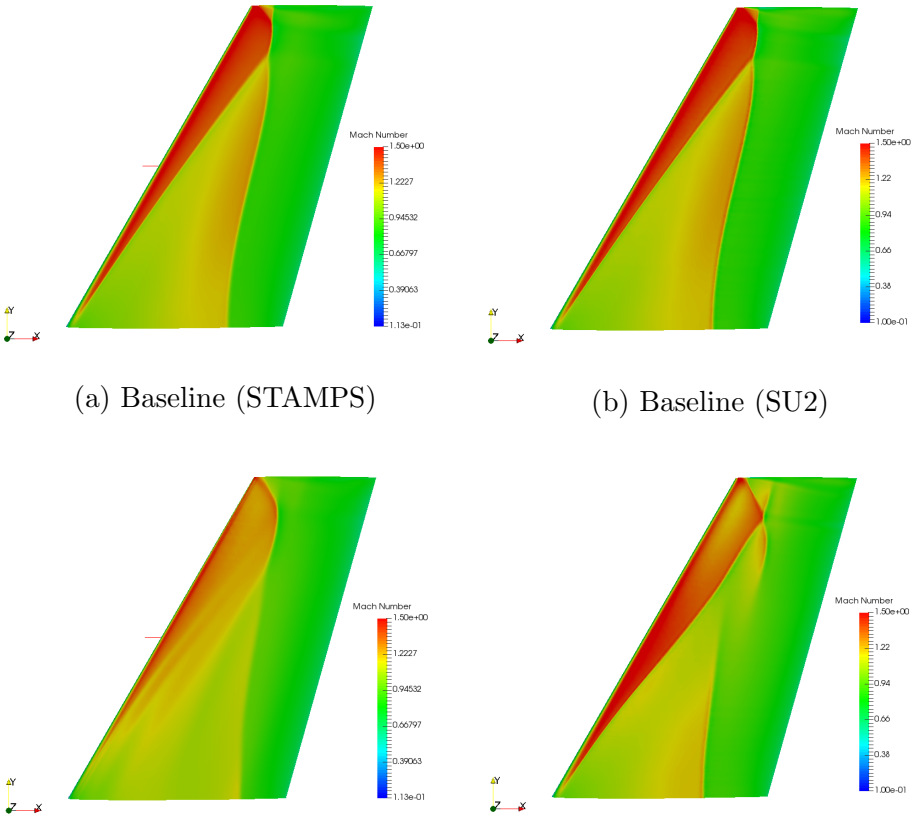


Figure 5.9: Convergence history of the objective function



(a) Baseline (STAMPS)

(b) Baseline (SU2)

(c) Optimised (NSPCC)

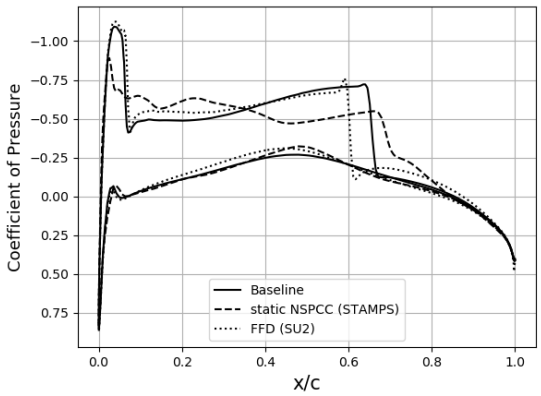
(d) Optimised (FFD)

Figure 5.10: Comparison of Mach number distributions between NSPCC and FFD for the ONERA M6 drag reduction case

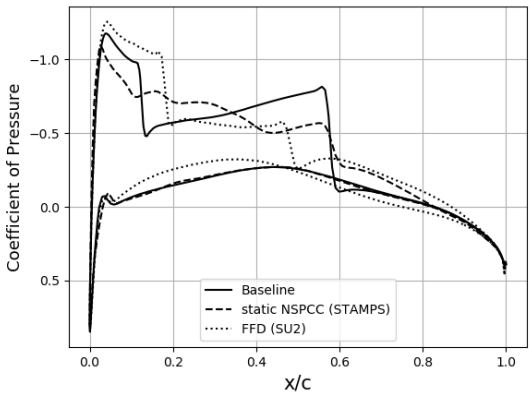
the optimised shape obtained using FFD show a small rear shock around the tip section of the wing which is not present in the geometry obtained using the NSPCC approach. In addition, strength of the first shock near the root section is only slightly reduced in the FFD case. This may be the source of the difference in the drag reduction between opt-NSPCC and opt-FFD. Masters et al.[82] also observed a similar kind of pattern when performed shape optimisation of M6-wing using a coarser level subdivision parametrisation method and also noted that some remnant of front and rear shocks are reduced as they increases the number of design variables in the optimisation. This clearly indicates that sufficient number of design variables are required to achieve shock free solution. Figure 5.11 shows the distribution of coefficient of pressure over the baseline and optimised geometries taken at different sections along the span-wise direction of the M6-wing. We can clearly see that, strength of the shock wave near the root section of the opt-NSPCC has been reduced significantly than the opt-FFD geometry.

Cross-sectional shape changes along the span-wise direction are shown in Fig. 5.12. When compared with the NSPCC, optimised geometry obtained using FFD method captured large surface deformation near the root section of the M6-wing. This is may be due to the fact that FFD-box is constructed using Bernstein polynomial which has global shape modification property. Hence local or thin surface changes that are essential near shock region may not be projected properly onto the FFD-box. On the other hand, with the presence of local shape modification property of NURBS the NSPCC method has a potential to capture the local surface deformation computed by the adjoint sensitivity information, hence achieved large drag reduction of 22.3%. In addition, due to the presence of clustering of control points near the leading edge the NSPCC method was able to capture most of the small surface changes and reduce the strength of the leading edge shock wave significantly.

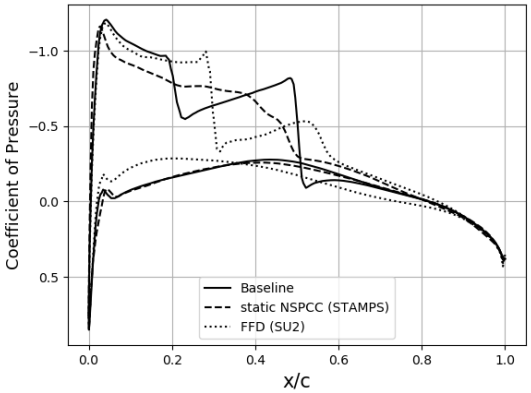
Results show that, the NSPCC approach has the potential to obtain a significantly improved wing shape with a coarse distribution of control points along the span-wise direction and with assembly constraints. In this case, wing-box and trailing edge thickness constraints are implicitly defined using the test point approach. Using the NSPCC approach, assembly constraints can be imposed in an optimi-



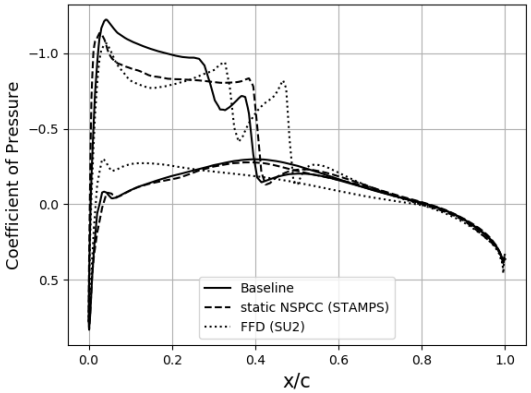
(a) $\frac{y}{b} = 0.20$



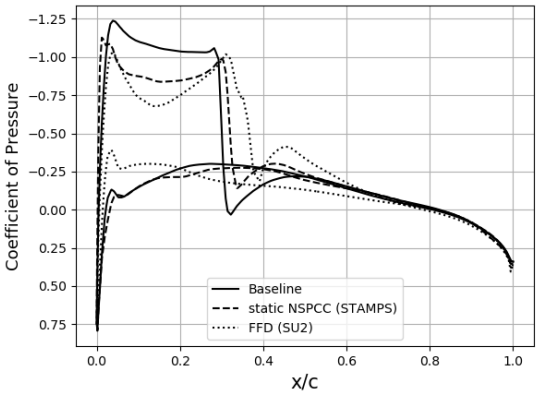
(b) $\frac{y}{b} = 0.44$



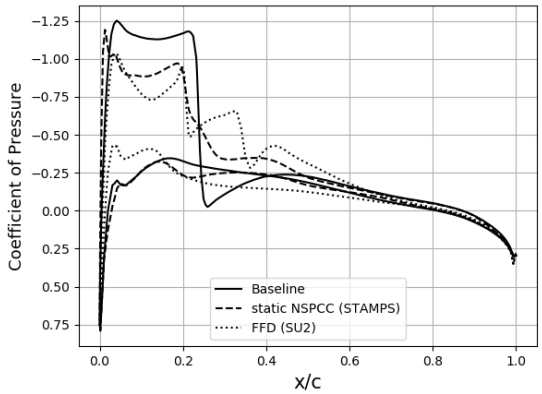
(c) $\frac{y}{b} = 0.65$



(d) $\frac{y}{b} = 0.80$

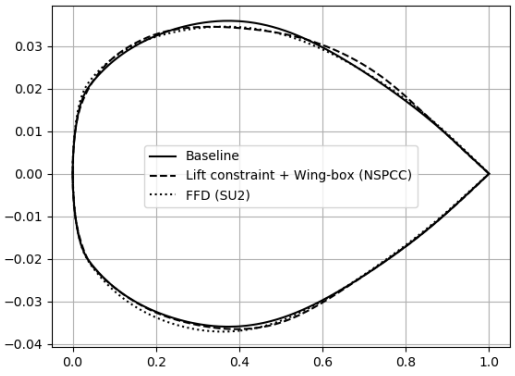


(e) $\frac{y}{b} = 0.90$

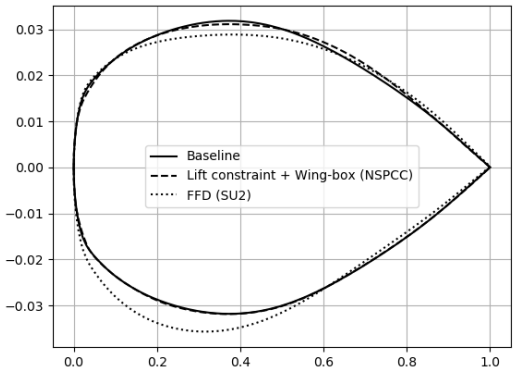


(f) $\frac{y}{b} = 0.95$

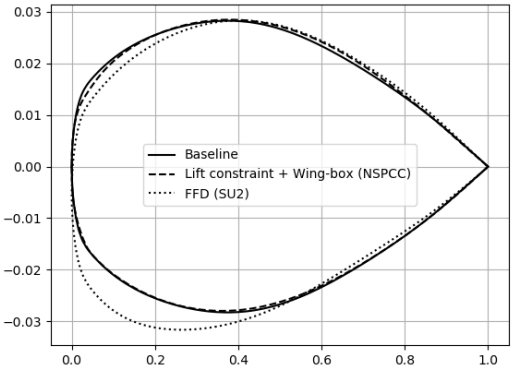
Figure 5.11: Comparison of C_p plot for optimised designs obtained using static NSPCC and FFD



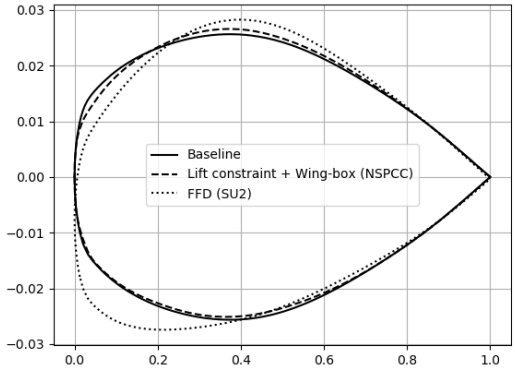
(a) $\frac{y}{b} = 0.20$



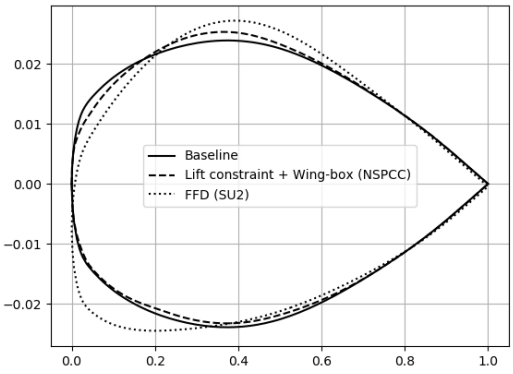
(b) $\frac{y}{b} = 0.44$



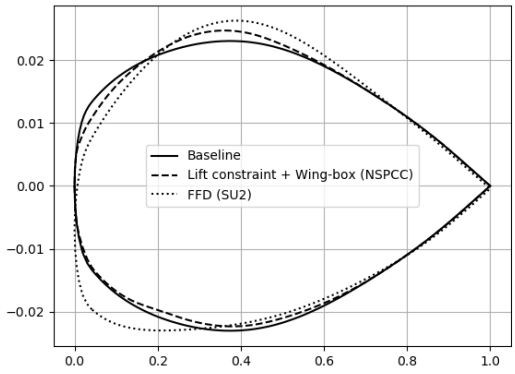
(c) $\frac{y}{b} = 0.65$



(d) $\frac{y}{b} = 0.80$



(e) $\frac{y}{b} = 0.90$



(f) $\frac{y}{b} = 0.95$

Figure 5.12: Comparison of optimal shapes obtained using NSPCC and FFD

sation case without additional pre-processing step. This reduces the user set-up time substantially for the given problem. On the other hand, FFD method requires the construction of axillary hexahedral control lattice around a wing shape. For a complex geometry, this step may be tedious one.

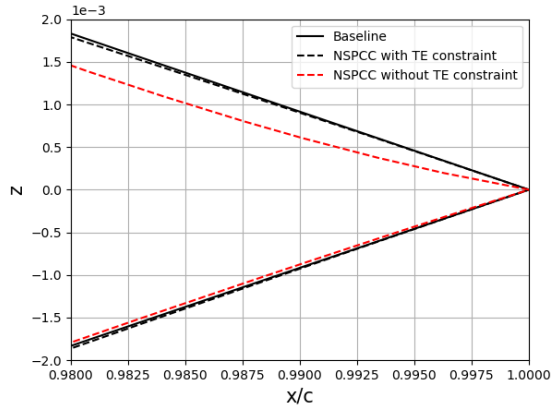
As experimental results are available, shape optimisation of the ONERA M6-wing has been studied by numerous researchers using a wide range of parametrisation methods. Hence the results are compared with other previous works available in the literature and this is shown in Table. 5.4. It can be seen that, parameterisation method based on subdivision surfaces outperforms the NSPCC approach. Masters et al. [81] employed SNOPT optimiser for handling lift and volume constraints. Based on the presented details, a significant amount of deformations are observed near the trailing edge region which may reduce the drag further. Therefore, optimal shapes had better performance improvement than other methods. To highlight the flexibility of the NSPCC approach, comparison of TE shapes of the optimised wing obtained with and without TE thickness constraints are shown in Fig. 5.13. Using the NSPCC approach with assembly constraints, TE thickness is maintained in the span-wise direction. As presented in Table 5.4, drag is reduced by 22.3% while satisfying the assembly constraints. Further improvement can be achieved by adaptively increasing the control points on the design surface.

5.8 Summary

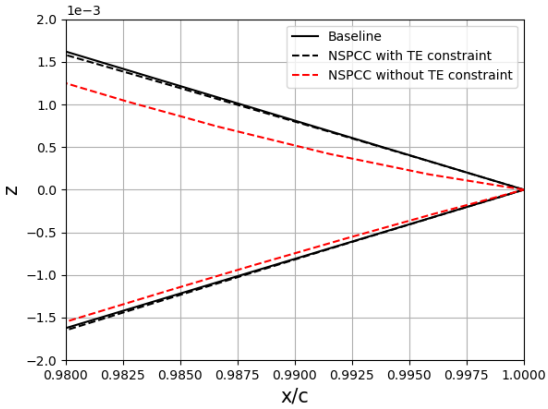
In this chapter, the static NSPCC method is extended to handle assembly constraints such as wing-box and trailing edge thickness constraints in aerodynamic shape optimisation. The proposed methodology was tested by minimising the drag of an ONERA M6-wing at inviscid transonic flow conditions subject to the constraints that lift and thickness should not reduce from the baseline values. The entire implementation is done in Fortran hence source transformation AD tool is used to obtain constraint Jacobian and shape sensitivities for optimisation. In this work, in-house flow and discrete adjoint solver STAMPS is used to obtain flow fields and gradients. A python-based optimisation framework is developed to integrate the CAD kernel and STAMPS solver developed in this work with the scipy-BFGS

Origin	**Grid Size**	**Parametrisation method**	**N_{DV}**	**Objective Function**	**Additional Constraints**	**Drag Reduction**
Masters et al.[82]	300K	Multi-level	15→936	Drag (Euler)	Volume	-33.5%
		Subdivision	60→936			-32.72%
		Subdivision Surface	60			-31.3%
			15			-29.68%
			236			-29.34%
			936			-25.87%
Present Work	**540K**	**Static NSPCC**	**486**	**Drag (Euler)**	**Wing-box and TE thicknesss**	**-22.3%**
Present Work	**540K**	**FFD**	**198**	**Drag (Euler)**	**Thickness**	**-20.2%**
Martins et al.[75]	8M	FFD	150	Drag (Turbulent)	Thickness	-19.1%
Martins et al.[75]	1M	FFD	150	Drag (Turbulent)	Thickness	-18.8%
Zingg et al.[100]	2.2M	FFD	226	Drag (Turbulent)	Volume	-17.1%
Martins et al.[75]	1M	FFD	150	Drag (Euler)	Thickness	-16.6%
Anderson et al.[98]	359K	Engineering Parameters	21	Drag (Turbulent)	NA	-15.5.%
Qin. et al.[88]	312K	Bezier Surface	15	Drag (Turbulent)	Volume	-14.9%
Dheeraj et al.[4]	154K	Parametric CAD model	27	Drag (Euler)	NA	-14%
Orovio. et al.[23]	43K	FFD	12	Drag (Turbulent)	NA	-10%

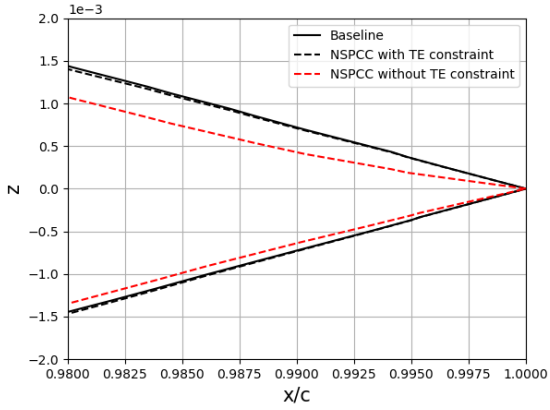
Table 5.4: Comparison of drag reduction with previous work



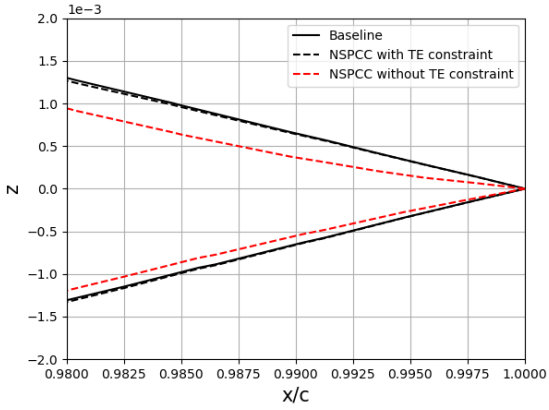
(a) $\frac{y}{b} = 0.20$



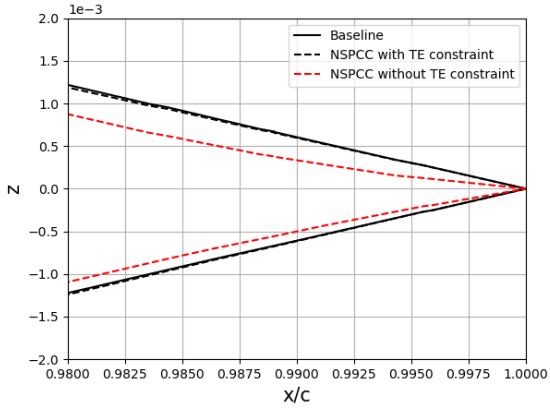
(b) $\frac{y}{b} = 0.44$



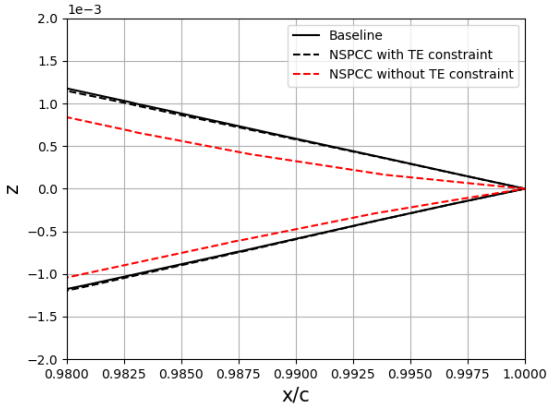
(c) $\frac{y}{b} = 0.65$



(d) $\frac{y}{b} = 0.80$



(e) $\frac{y}{b} = 0.90$



(f) $\frac{y}{b} = 0.95$

Figure 5.13: Comparison of TE shapes of the optimised wing obtained with and with TE thickness constraints using NSPCC approach.

optimiser.

Performance of the developed shape optimisation tools are compared with the currently available state of the art open source SU2 shape optimisation tools. Free-From Deformation (FFD) method is available in SU2 tools and the tutorial case setup provided in the SU2 website is used to perform shape optimisaiton of the ONERA M6 wing. Detailed grid convergence study, verification with SU2 and validation study are performed in this chapter. The results show that the overall features of the lambda shock computed using STAMPS are fairly matched between the SU2 and the available experimental results.

Two parametrisation methods namely static NSPCC and FFD methods have been compared with respect to their effectiveness for minimising drag of the ONERA M6-wing. NSPCC derives the design space directly from a boundary representation (BRep) of the geometry. Constraints are handled using test-point approach and the design space is derived from the kernel of the constraint Jacobian which is evaluated using SVD. The resultant shape modes are orthogonal to each other hence regularize the design space and offers robust shape optimisation process. When compared with the FFD method, NSPCC approach offers better local shape control and able to capture small scale adjoint information in high curvature region which reduces the strength of the leading edge shock hence leads to more drag reduction. Furthermore, using test point approach user can easily impose assembly constraints in the design process and its gradients can be evaluated using AD tools. In addition to that, entire design chain is reverse differentiated hence gradient computation is independent to the number of design variables.

Chapter 6

Shape Optimisation of VKI U-Bend

6.1 Introduction

In chapter 3, details about the in-house flow and discrete adjoint CFD solver named STAMPS are presented and then gradient of the objective function with respect to design variables are assembled using adjoint methodology. Automatic differentiation tool based on source-transformation approach is employed to differentiate the necessary blocks of code in the design chain. In Chapter 4, details about the proposed adaptive NURBS-based parameterisation method with complex constraints (adaptive-NSPCC) are presented in which the control points of the NURBS patches are used to deform geometry in the design process. In Section 4.10, the adaptive-node based parameterisation method is proposed in which the normal displacements of surface grids are taken as design variables. In this chapter, both shape parameterisation methods are coupled with the STAMPS solver and applied to the aerodynamic shape optimisation of internal turbine cooling channel U-Bend [135] with the objective to minimise the total pressure loss.

6.2 Turbine Blade Cooling Channel

To achieve high thermal efficiency modern gas turbines are required to operate at extremely high temperature. To withstand high temperature, turbine blades are

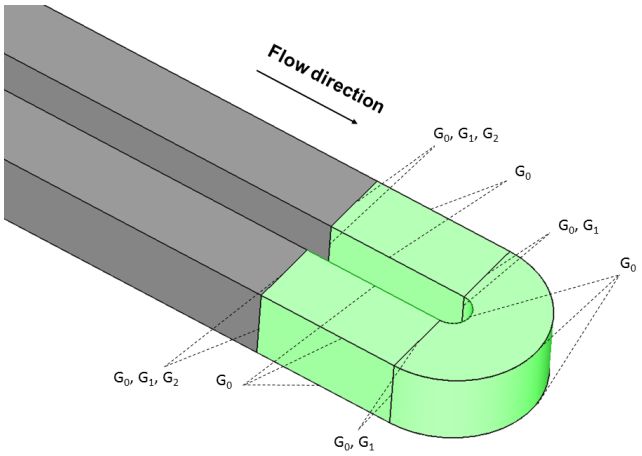


Figure 6.1: U-Bend geometry with design surfaces are highlighted in green

equipped with intricate cooling channel which circulates cooling air through multiple passages inside the blade and discharged through small holes located on the blade wall. This cooling air creates a thin insulating layer along the outer surface of the turbine blade.

This U-Bend passage turns the cooling fluid 180 degrees and of crucial importance since they represent the region of high-pressure loss. The goal of the optimisation process is to deform the U-Bend region of an internal serpentine cooling channel such as to minimise the mass-averaged total pressure loss defined as J,

$$\underset{\alpha}{\text{minimize}} \quad J = \frac{\int_{inlet} P_{tot}\vec{u}\cdot\vec{n}dS - \int_{outlet} P_{tot}\vec{u}\cdot\vec{n}dS}{\int_{inlet}\vec{u}\cdot\vec{n}dS} \tag{6.1}$$

where α represents the design variables, P_{tot} is the total pressure, u is the velocity vector, n is the surface normal direction and S is the cross section area. Figure 6.1 shows the U-Bend geometry with design surfaces are highlighted in green. The dimensions of the baseline U-Bend geometry is shown in Fig. 6.2. It consists of a two squared cross section ducts and a hydraulic diameter of $D_h = 0.075m$. Both ducts are connected by the half-circular U-Bend.

6.3 Shape Parameterisation

To determine the influence of distribution of control points on the design surface, three levels of control net distributions are used to parameterise 180 degree bend

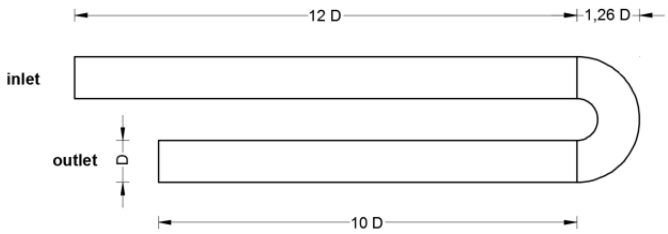


Figure 6.2: Dimensions of the U-Bend geometry.

region of the U-Bend geometry. Control points on the design surfaces are refined globally in both parameter direction which represents coarse (L1), medium(L2) and fine (L3) design spaces for shape optimisation. The U-Bend region composed of 12 patches which include 8 rectangular and 4 half-circular patches. In particular, the total number of control points in each level are refined by the order of two.

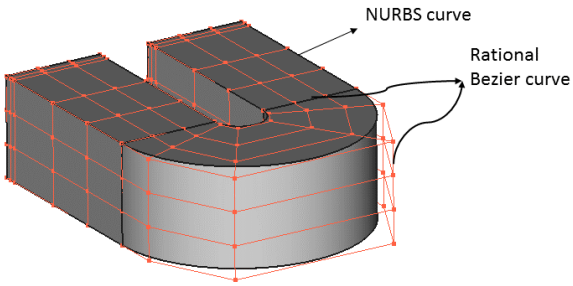
For coarser level (L1) each patch is parameterised using cubic NURBS curve along the streamwise direction with 6 control points and cubic rational Bezier curve along radial direction with 4 control points (perpendicular to streamwise direction). Hence L1 parameterisation exhibits global shape control along the radial direction. In total L1 has 288 control points with each patch defined using 6×4 control net which is shown in Fig. 6.3a. In L2, each patch is parameterised using bi-cubic NURBS patch with 8×6 control net in which streamwise and radial directions are defined using 8 and 6 control points respectively, resulting in a total of 576 control points. In L3, each patch is defined using bi-cubic NURBS patch with 12×8 control net in which 12 and 8 control points along streamwise and radial direction respectively resulting in a total of 1152 control points. Control points distributions defining the U-Bend region corresponding to each level are shown in Fig. 6.3. Figure 6.4 shows the comparison of the sensitivity of the design surface for a single control point projected over the computational mesh for each level presented above. This confirms that the last two parameterisations (L2 and L3) have the property of local control and moreover the influence region of a control point is very small for L3.

In order to test the performance of the proposed adaptive-NSPCC method the U-Bend shape with the coarse net L1 is used as the initial geometry. Once the refinement trigger terminates the current design space level, control points are refined

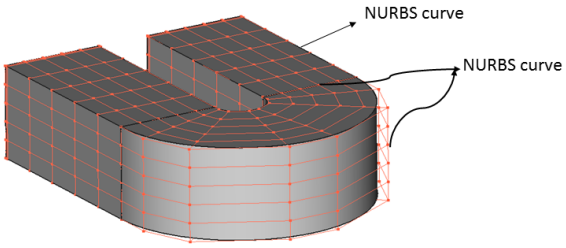
only in the regions where smoothed node-sensitivities are larger by using knot insertion algorithm. In an adaptive framework it is important to determine the suitable value of knot to be inserted in a knot vector. Zingg et al. [52] proposed to insert a new knot arbitrarily in each knot span of a knot vector to refine control points of B-spline volumes. Martins et al [55] insert a new knot in the middle of each knot span to refine FFD frame. This may be suitable for simple geometries however for a complex geometry with a large number of patches inserting a knot in each knot span without any additional measure may leads to the addition of unnecessary control points on the design surface. Therefore, additional sensitivity-based adaptation criteria which is discussed in Section 4.9 is employed to choose a suitable value of knot to be inserted in each parameter direction. Adaptive NSPCC replaces user in the design loop as the control points are refined automatically using the sensitivity driven adaptation criteria which reduces the unnecessary addition of control points on the design surfaces.

At common edges between the deformable and fixed NURBS patches which are near to inlet and exit channel of the U-Bend region, the first three layers of control points of the moveable NURBS patches are fixed so that the entry and exit throats both have G_2 continuity and meet with zero curvature. The geometric continuities on the common edges between the movable design patches are imposed by using the test point approach presented in Section 4.5. G_0 continuity is imposed on all common edges to avoid surface gaps during CAD deformation. On the other hand, G_1 continuity is imposed only on the edges that connect rectangular and circular patches. Figure 6.1 also shows some of the continuity constraints imposed on the common edges.

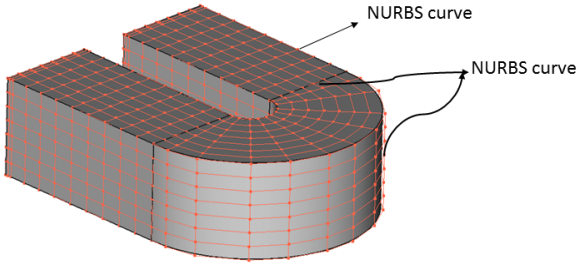
On the other hand, node-based method derives parameterisation directly from the computational mesh employed in simulation. Grid convergence strudy is performed to select to suitable mesh for optimisation. The details of the grid convergence study are presented in Section. 6.4 The selected computational mesh consists of $260k$ nodes and the design surface consists of 9500 surface nodes. The surface mesh used in the node-based parameterisation is shown in Fig .6.5.



(a) Coarse: Level-1

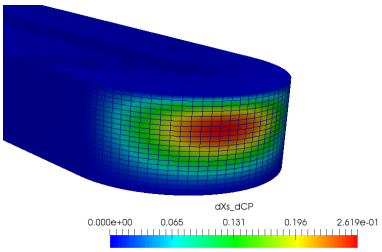


(b) Medium: Level-2

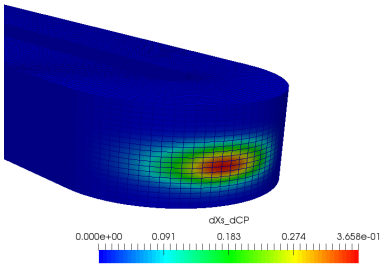


(c) Fine: Level-3

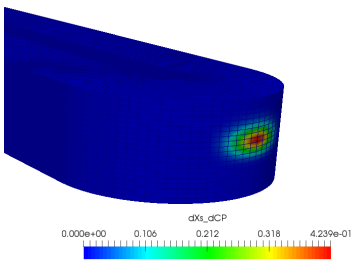
Figure 6.3: Three different levels of NURBS-based parameterisation



(a) Coarse: Level-1



(b) Medium: Level-2



(c) Fine: Level-3

Figure 6.4: Sensitivity of a control point on the outer U-Bend patch $\frac{\partial X_s}{\partial \mathbf{P}}$

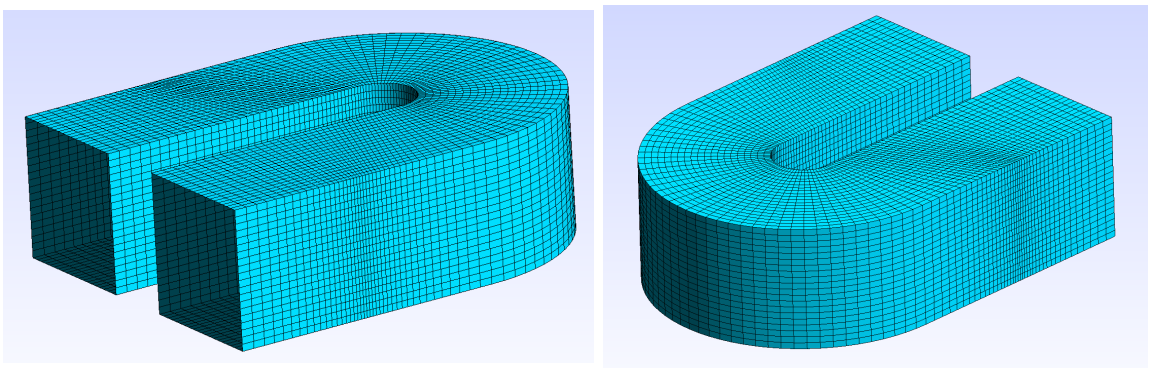


Figure 6.5: Surface mesh of the baseline U-Bend geometry

6.3.1 Shape sensitivity validation

It is necessary to validate the correctness of the differentiated code. Our approach is to validate the forward and reverse differentiation of the shape parameterisation kernel before integrating them into a design chain. L1-CAD model and fine mesh are employed for performing shape sensitivity validation. Details of the fine mesh is presented in the grid convergence study (Section 6.4)

AD Code vs Complex Step Derivative

All the required derivatives of geometric operators, including surface sensitivities with respect to parameters s and t for point inversion (Eqn. 4.5), entries of the constraint matrix $\mathbf{C}$ (Eqn. 4.16 and Eqn. 5.4) and shape sensitivities (Eqn. 4.30 and Eqn. 4.42) are computed using derivative code produced by the source transformation AD tool Tapenade [53].

Figure 6.6 shows the relative error in shape sensitivity $\frac{\partial X_s}{\partial \alpha}$ for a particular surface grid point with respect to a design variable α for a range of step size $h \in [10^{-2}, 10^{-21}]$ computed using a forward difference, a central difference and the complex step method, using the AD value as reference. As expected, the forward difference shows linear convergence, while central difference and complex step derivative show quadratic convergence.

However accuracy of the method depends on the chosen step width. If chosen step width is too small, then subtractive cancellation error will dominate hence relative error diverges for finite difference method when step width reaches its threshold

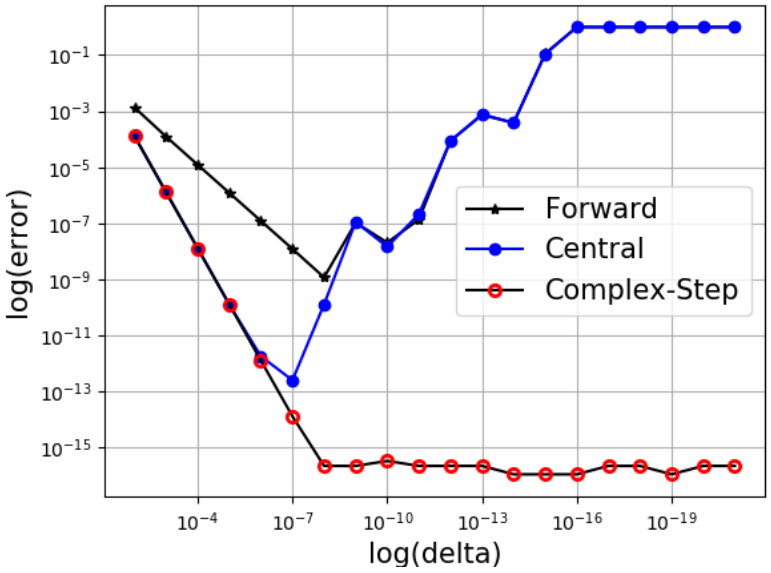


Figure 6.6: Convergence of relative error for a surface point using complex step, forward and central difference.

limit h^*. In this case, $h^* = 10^{-8}$ for forward difference and $h^* = 10^{-6}$ for central difference. On the other hand, the complex-step approximation does not involve a difference operation hence extremely small step width $h^* < 10^{-7}$ can be chosen without subtractive cancellation error.

To verify the AD derivatives we compare to the values obtained with the complex step method [125]. The method is very easy to use in Fortran which allows very simple conversion from all double precision variables to double precision complex variables. However additional care has been taken as discussed in [30, 79] when handling intrinsic functions such as `abs` and conditional branches `IF..THEN..ELSE`. Figure 6.7 shows the convergence of truncation error for seven surface points using complex step derivative method. The comparison of reverse mode surface sensitivity magnitude with respect to a control point on three different NURBS patches against complex step derivative with step width $h^* = 10^{-8}$ is shown in Fig. 6.8. The overall magnitude plots verify that AD sensitivities matches the complex step derivative closely.

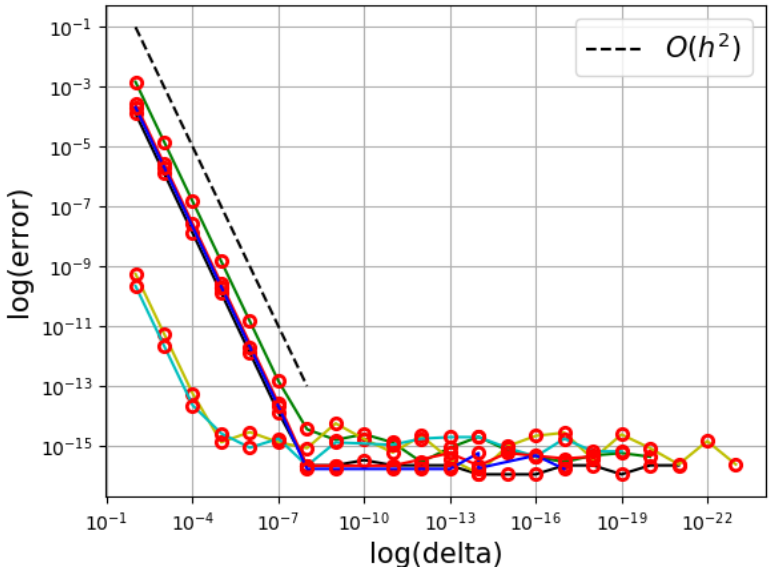


Figure 6.7: Relative error for seven surface points using complex step derivatives. Error: $\epsilon = \frac{|FD-AD|}{|AD|}$.

Taylor Test

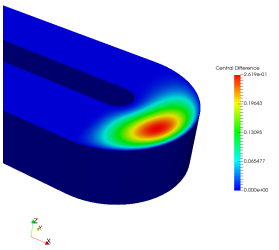
Additionally AD sensitivities are also verified using a Taylor test with a range of step sizes $h \in [10^{-2}, 10^{-20}]$. The Taylor test is useful to fix the problems of the finite difference round-off error being amplified by a small step size.

$$f(x+h) - f(x) - h\frac{\partial f}{\partial x} = \mathcal{O}(h^2). \tag{6.2}$$

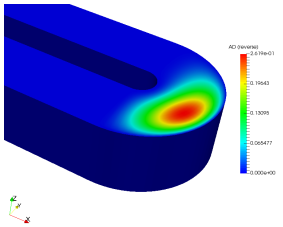
Figure 6.9 shows a overview of a taylor test for constraint Jacobian on a number of test points. As similar to Mladin et al [18], we observe better convergence than the theoretical convergence rate of h^2 and this continues upto error converges to machine precision, in this case $h^* = 10^{-7}$. Comparison of total sensitivities computed with both forward and reverse mode shape are presented in Fig. 6.10. For visual clarity, results are shown here only for a number of sensitivity index and confirm that results coincide to a very high extent.

Shape Matching

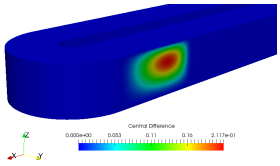
Furthermore, we have developed a shape matching optimisation problem to verify the adjoint version of NSPCC CAD kernel in the design loop. Main objective is to use the differentiated CAD sensitivities in both forward and reverse mode to find a



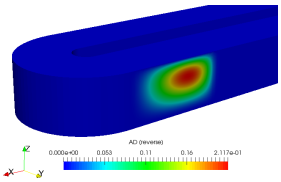
(a) Complex step: patch ID-2



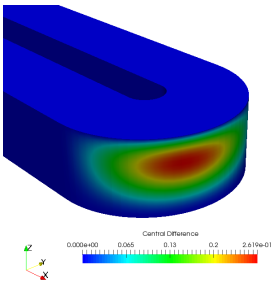
(b) AD (reverse): patch ID-2



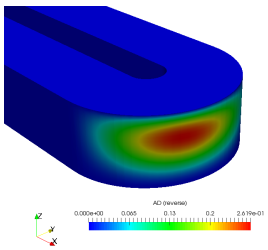
(c) Complex step: patch ID-9



(d) AD (reverse): patch ID-9



(e) Complex: patch ID-10



(f) AD (reverse): patch ID-10

Figure 6.8: Verification of reverse differentiated surface sensitivities with respect to control point using complex step derivative method with step width 10^{-8}.

set of control points to match a given perturbed CAD geometry. This geometry is created by giving arbitrary perturbations to control points using NSPCC algorithm hence both CAD ($\mathbf{P}^{\circ}$) and computational grids are perturbed (X_v°). Figure 6.11 shows the comparison between the initial ($\mathbf{P}$) and perturbed geometry ($\mathbf{P}^{\circ}$) to be matched. We choose to minimize the sum of quadratic distances between perturbed surface mesh points (X_s°) and NURBS surface approximation (X_s'). Therefore entire NSPCC algorithm as presented in Algorithm 1 is utilized without CFD in the loop. This leads to the following objective function,

$$J = \gamma \sum_{i=1}^{N_S} \left(X'_{i,s} - X^{\circ}_{i,s} \right) \tag{6.3}$$

where N_S is the number of surface mesh points and γ is a scaling factor. Figure 6.12 shows the convergence history of the shape matching problem using both modes

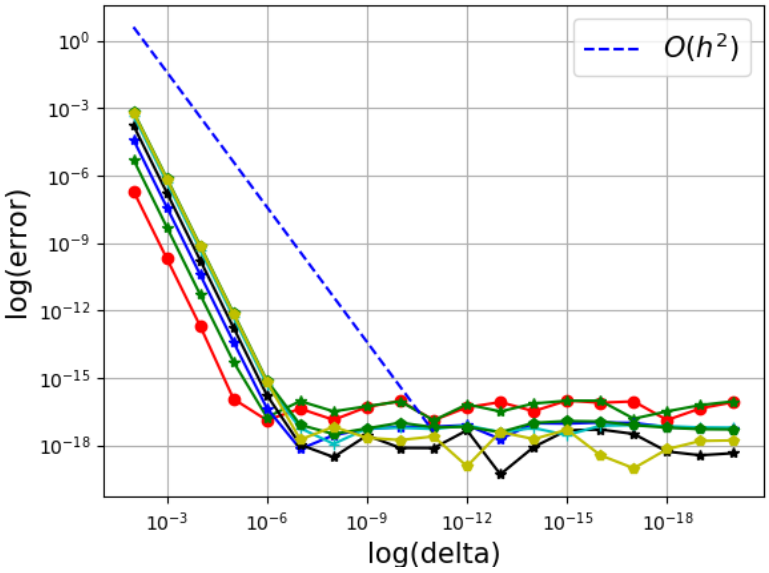
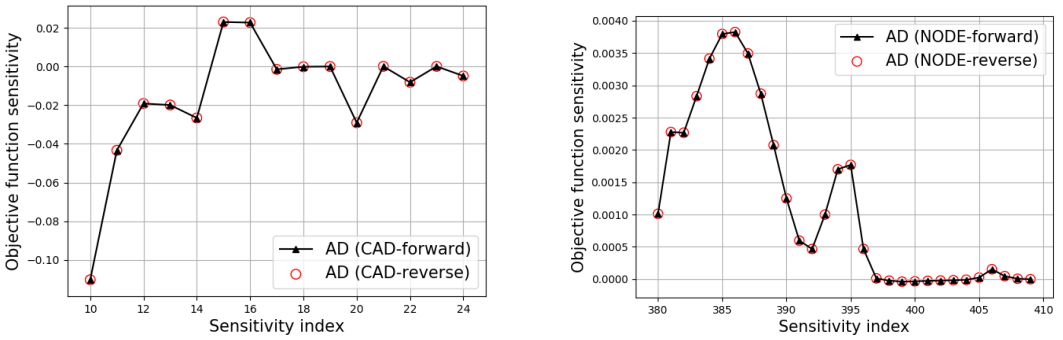


Figure 6.9: Overview of Taylor test for seven test points

of AD with BFGS as optimiser. Since tangent and adjoint CAD sensitivities are converge to machine precision the objective function value is equivalent to machine precision at every design iteration. Hence both cases are converged to the same shape.

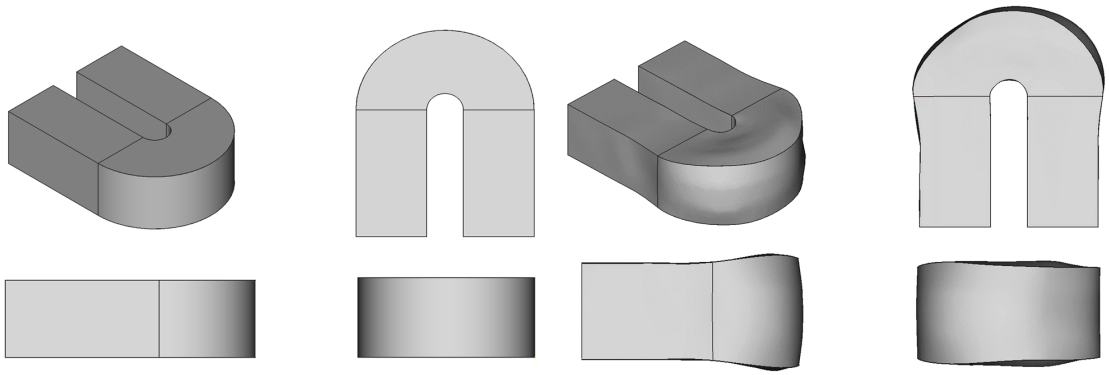
6.3.2 Computational time

To measure the performance of reverse differentiation of NSPCC approach we have parameterized the U-Bend geometry with an additional three control net levels $L4, L5$ and $L6$ with the total number of control points $2304, 4752$ and 9360 respectively. These additional levels were used only for testing the performance of reverse differentiation and not for optimisation purpose. Figure. 6.13 shows the amount of computation time taken for CAD sensitivity in a single design step using forward, adjoint and central difference method with respect to all design variables for control net levels of six different sizes. In this test, fine mesh is used for computing CFD sensitivities which is selected based on the grid convergence study (Section 6.4). For clarity, time taken to compute flow fields (CFD primal), adjoint variables (CFD adjoint), STEP file parser, null space computation and CFD sensitivity assembly were omitted and presented only the computational time taken for the CAD sensitivity evaluation in a single complete design chain.



(a) (b)

Figure 6.10: (a) Verification of objective function sensitivities computed using reverse mode CAD sensitivities with forward mode CAD sensitivities, (b) Verification of objective function sensitivities for node-based method.



(a) Original NURBS (b) Perturbed NURBS

Figure 6.11: U-Bend geometry used in the shape matching optimisation problem.

In this case, the forward mode outperforms the central difference method, this is due to the fact that sensitivity computation with respect to each control point requires surface evaluation for both positive and negative perturbations. This plot clearly shows the performance benefits of using the reverse differentiation of NSPCC approach (adjoint mode) compared to both forward mode and central difference, especially when the number of control points is large. Therefore in the design chain, we employed reverse differentiated NSPCC approach for shape optimisation.

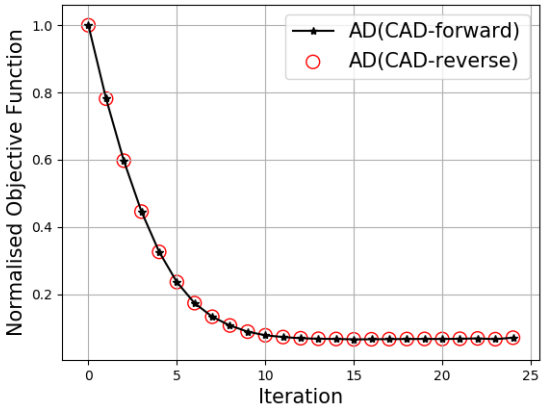


Figure 6.12: Convergence of objective function for shape matching problem using both tangent mode and adjoint mode NSPCC

6.4 Grid convergence study

The Reynolds number based on the hydraulic diameter ($D_h = 0.075m$) of the U-Bend is $Re = 43830$. A Mach number of 0.1 allows for valid assumption of incompressible flow. In order to accelerate the convergence of compressible solver for low Mach number condition, pressure scaling method suggested by Robert et al. [41] is employed in this work. For a given reference velocity $U_0 = 8.4m/s$, density ρ, and Mach number M, the required static pressure p can be calculated as follows,

$$p = \frac{\rho U_0^2}{\gamma M^2}. \tag{6.4}$$

This type of pressure scaling ensures good convergence for low Mach number flows and is especially suitable for internal flows where pressure drop across the domain is of primary interest. The adjoint solver in STAMPS is derived from the flow solver using the automatic differentiation(AD) tool Tapenade. The time-stepping of the adjoint equations is based on a fixed-point method using the same assembly steps as the primal. Details of the numerical method, primal and adjoint implementations can be found in [27, 50].

Based on the suggestions given by McHale et al. [85] four grid levels have been generated to perform a detailed grid convergence study: a coarse (C), a medium (M), a fine (F) and an extra fine (XF) mesh with each level containing $50k$, $125k$, $260k$ and $500k$ nodes respectively. Computational grids are created using Ansys Mesher with the coarse grid (C) having Y^+ value of 3 and remaining all grids

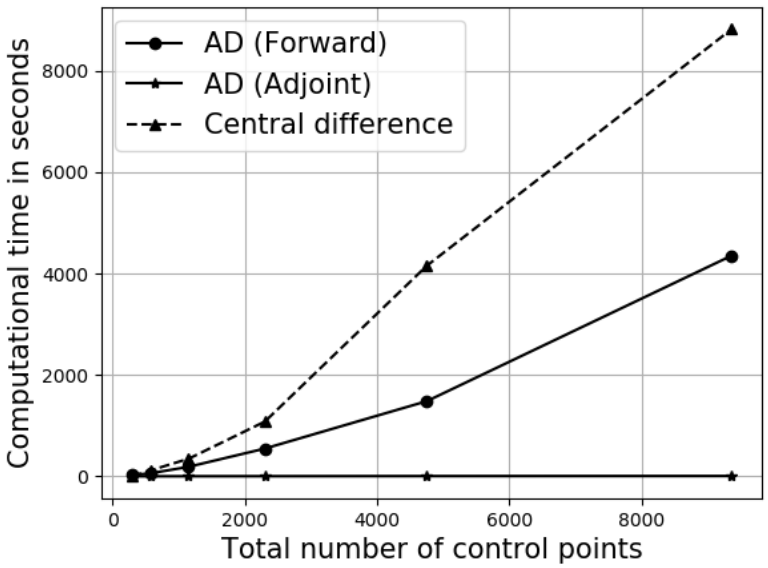
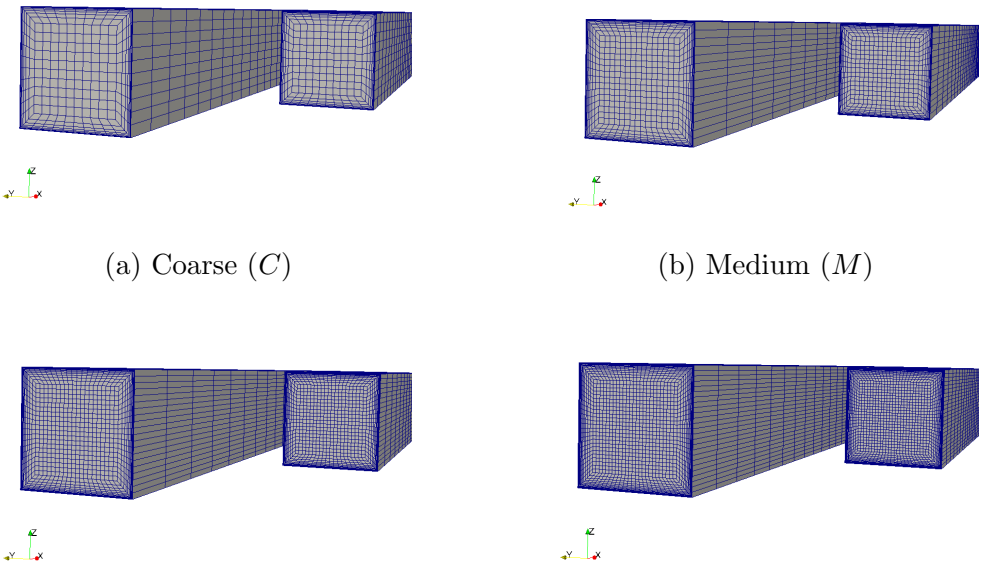


Figure 6.13: Computational time taken for CAD sensitivity evaluation, Forward mode vs Reverse mode CAD sensitivities

having Y^+ value of 1. The U-Bend geometry has square cross-section hence it has sharp corners which generates singular cells while using body-fitted mesh with square topology. To avoid this we created the butterfly topology at the inlet which has a square block in the center and the remaining four blocks filling the peripheral around the square block. Then by using the method of sweeping in Ansys Meshing we sweep the butterfly topology from inlet to the outlet through the interior domain of the U-Bend geometry. The structure of all the grid levels at the inlet and outlet of the U-Bend geometry is shown in Fig. 6.14.

In this work, the objective function value and velocity profiles at three different locations are compared with all the grid levels to make sure the solution is independent of the mesh resolution. These velocity profiles including both streamwise as well as radial directions are taken one at the 90° turn region (location A) and others (location B and C) at the exit of the channel are shown in Fig. 6.15a. Figure 6.15b shows the variation of the normalized objective function value with all the grid levels. Figure 6.16 shows a comparison between the normalized streamwise velocity profiles at the location A, B and C (along the vertical lines). The positive and negative z/D_h indicates the top and bottom surfaces respectively from the center of the U-Bend. Figure 6.17 shows a comparison between the normalized



(a) Coarse (C)

(b) Medium (M)

(c) Fine (F)

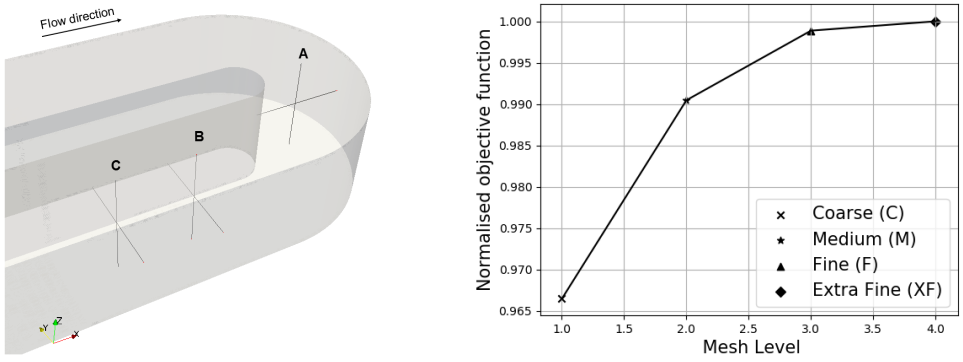
(d) Extra Fine (XF)

Figure 6.14: The inlet and outlet of the structured hexahedral meshes of all levels

radial velocity profiles at the location A, B and C (along the horizontal lines). From the comparison, it is found that both streamwise and radial velocity profiles from the fine (F) and extra-fine meshes are closely matched and hence solution became independent of the mesh resolution. Figure 6.15b shows that the fine mesh results in a difference of less than 0.1% for the objective function value with the extra fine (XF) mesh.This difference is found acceptable for the present study [85] and hence the fine mesh (F) is used for the optimisation purpose. The steady-state nonlinear primal flow solver fully converges and the discrete adjoint solver uses the same time-marching scheme which also fully converges the adjoint solution. The convergence history of both flow and adjoint solution using fine mesh is shown in Fig. 6.18.

6.4.1 CFD solver validation

The numerical results obtained using STAMPS are compared with the computational and experimental ones performed by Coletti et al. [29, 7] for the same Reynolds number $Re = 43830$. In their experimental work, the inlet leg with $23.3D_h$ long is used to guarantee a fully developed flow at the location of the circular bend. Based on the suggestion given in test case description [135] the inlet leg of $10D_h$ with respect to the center of the U-Bend region is used in the present numerical

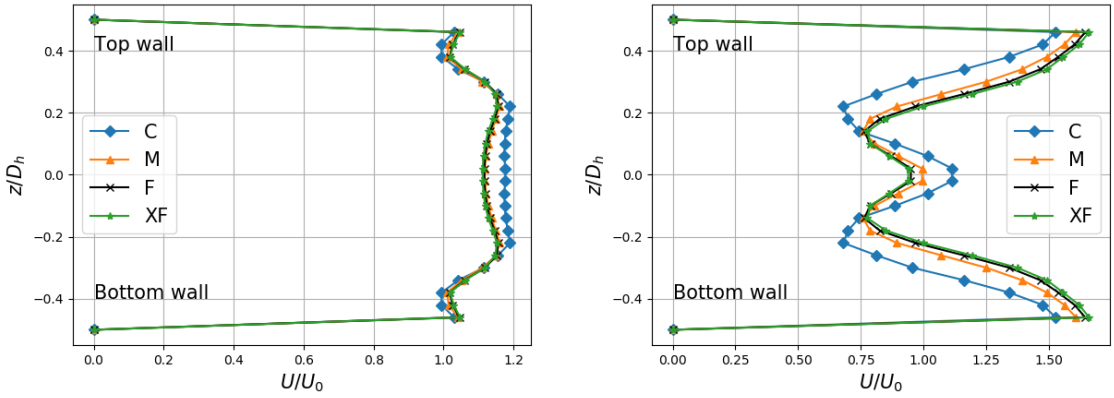


(a) Locations used for velocity profiles (b) Grid convergence of objective function

Figure 6.15: Grid convergence study.

study to reduce the computational cost. For the validation of our present model, the simulation is performed using fine mesh (F) containing total $260k$ nodes, as the grid-independence study showed that this grid offered sufficient resolution, see Fig. 6.15b. Comparison of the computed normalized axial velocity profile taken at the inlet section before the bend region shows very good agreement with experimental results as shown in Fig. 6.19. The comparison of the simulated velocity field and experimental results by Coletti et al.[29], the Large Eddy Simulation results by Alessi et al. [7] and the STAMPS results is shown in Fig. 6.20 for the symmetric mid plane (Z/D_h). The STAMPS solution captures the large flow separation region right after the turn. However height and length of the separation region are underestimated. Similar behavior is also noted in the RANS-based results of Alessi et al. [7].

Figure 6.21 shows a good match between STAMPS and LES for the counter-rotating Dean vortices at the 90° turn region, only small discrepancies near the inner wall separation region can be observed. The comparison of normalized velocity magnitude at the outlet leg $2D_h$ in Fig. 6.22 shows that STAMPS captured the overall flow pattern well. In particular, interactions between the Dean vortices and the recirculation region are well captured and velocity distributions are also consistent with LES results. However, the inner wall flow separation bubble is slightly underestimated in STAMPS, similar to other RANS-based U-Bend studies [29, 94, 96]. The analysis of the STAMPS solution computed on the fine (F) mesh is in good agreement with the published experimental and numerical results.

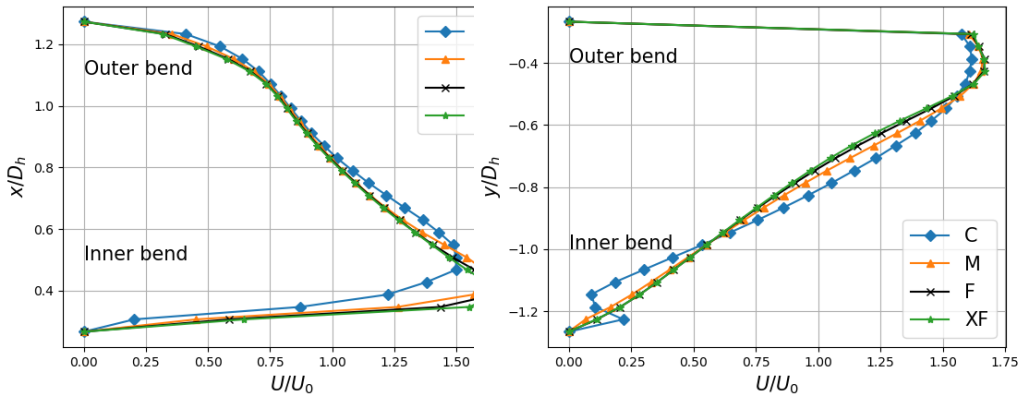


(a) at A

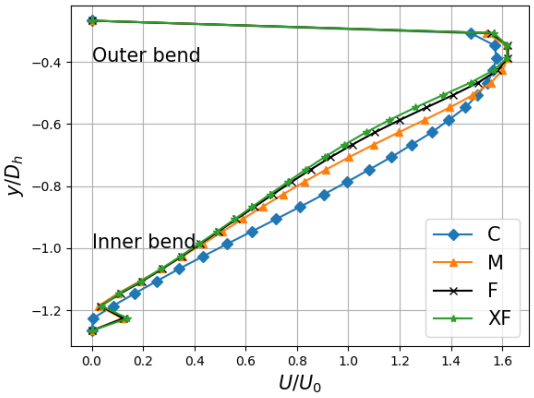
(b) at B

(c) at C

Figure 6.16: Streamwise velocity profiles along the vertical lines



(a) at A (b) at B



(c) at C

Figure 6.17: Radial velocity profiles along the horizontal lines at A and along y axis at B, C

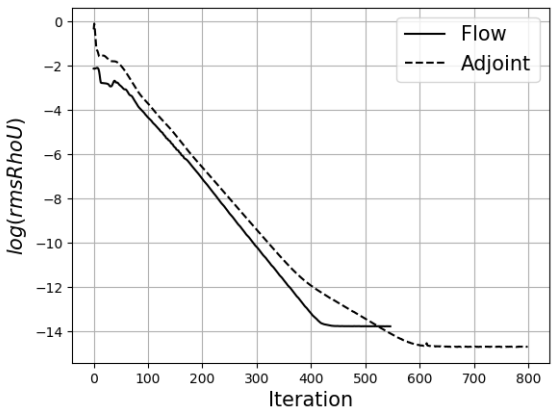


Figure 6.18: Convergence history of flow and adjoint solver

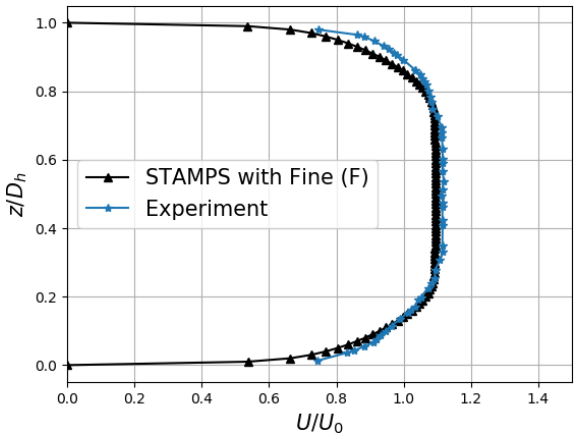


Figure 6.19: Comparison of normalized streamwise velocity profile taken the at inlet leg between experimental and STAMPS

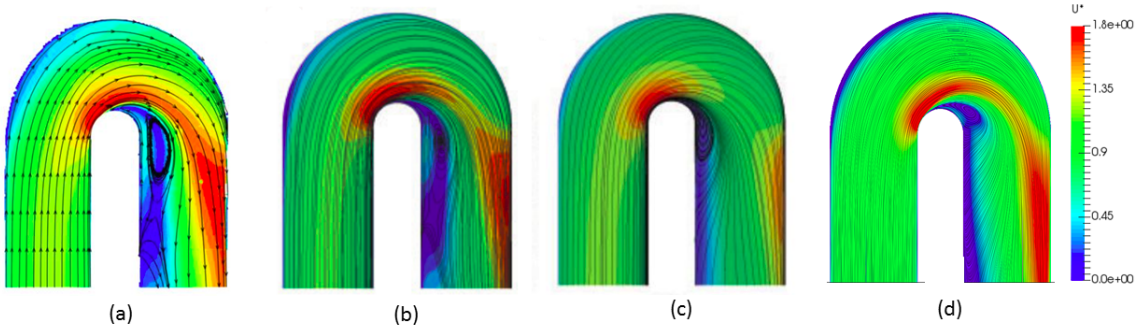


Figure 6.20: Comparison of normalized velocity field (U^*) along streamwise direction between experimental and simulation taken at mid plane. (a) Experimental [29] (b) LES simulation [7] (c) RANS simulation [7] (d) RANS-(STAMPS)

Hence this configuration has been used for the optimisation studies.

6.5 Static vs Adaptive NSPCC

6.5.1 Optimisation Work Flow

In this work, two different types of parameterisation methods are proposed. They are CAD-based adaptive NSPCC method and adaptive node-based method. In this chapter, both parameterisation methods are used to minimise the mass-averaged total pressure loss of the VKI U-Bend channel (Eqn. 6.1). This section describes the optimisation framework for aforementioned shape parameterisation methods. This Algorithm 2 is generic and can be used to setup any industrial test case for shape

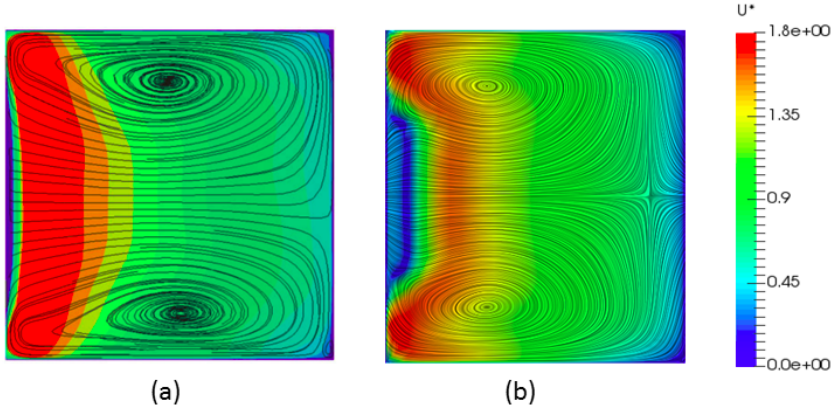


Figure 6.21: Comparison of normalized velocity field at 90° turn region. (a) LES [7] (b) (b) STAMPS

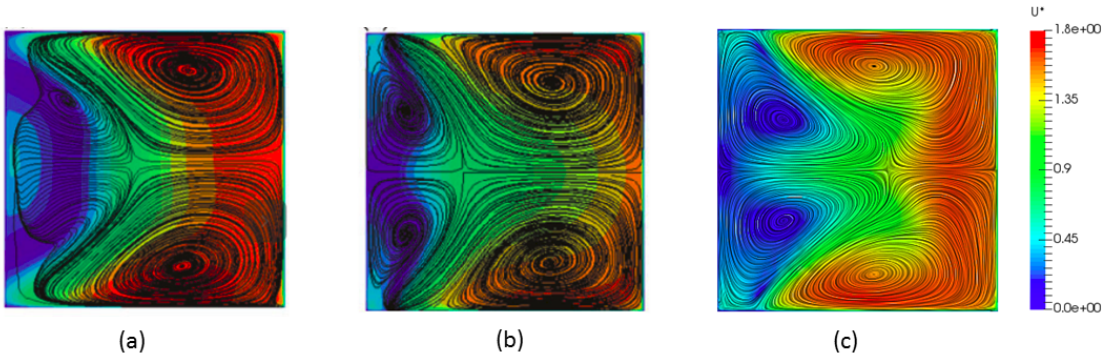


Figure 6.22: Comparison of normalized velocity field at outlet leg $2D_h$. (a) LES [7] (b) RANS [7] (c) STAMPS

optimisation. In this work, a Python framework has been developed to integrate all the required CFD and geometry modules with the Python-SciPy library to drive the design process. To set up a new case, a user needs to provide a CAD file (.Step) and the corresponding CFD mesh of the baseline geometry.

6.5.2 Flow Field of the Baseline Geometry

The pressure losses in a serpentine cooling channel are caused by the effect of both wall friction and momentum exchanges due to the change in the direction of the flow. Figure 6.23 shows the flow field of the baseline geometry obtained using Paraview. The cross sectional slices are taken at different locations along the channel. When a coolant flow passes through the U-Bend, the presence of pressure gradient normal to the streamline provides the centripetal force that required to turn the flow around the bend. This results in very low static pressure close to the inner bend region, hence when the flow exists the bend region it experiences a strong adverse pressure gradient along the streamwise direction. The fluid particles that are close proximity to the inner wall have low-velocity and don't have the potential to overcome the adverse pressure gradient. Hence the flow separates from the inner wall boundary and loses some of its energy in generating local eddies.

In addition to that, losses also take place due to the presence of strong secondary flows in the radial plane of the U-Bend geometry because of the pressure gradient normal to the streamline. The fully developed flow profile disrupts the balance between the pressure gradient and centripetal force. As a result of inertial effects, fluid at the center of the bend moves towards the outer wall at the mid-plane and comes back towards the inner wall near the top and bottom walls. This creates symmetric strong counter-rotating vortices at 90° turn region called Dean type vortices which persists in the long downstream exit channel of the U-Bend. This strong spiral motion in conjunction with the flow separation reduces the effective cross sectional area and accelerates the flow towards the outer wall of the exit channel which increases the velocity gradients at the wall. Hence results in greater frictional losses in a U-Bend cooling channel than the straight pipe under similar conditions. This can be clearly seen in Fig. 6.23. A similar pattern was found in both experimental [29, 44] and numerical simulations [71, 136]. Therefore an effective design should be

Algorithm 2 Shape Optimisation Framework

Input: CAD file in STEP format and the corresponding computational mesh.

1: Choose the parametrisation method and select the design surfaces.

a: **NSPCC**: Read BRep of a CAD model **(S)** and surface mesh points (X_s) and perform surface mesh mapping.

b: **Node-Based**: Read surface mesh points on the design surface (X_s)

2: **Primal and adjoint computation**:

a: Run **STAMPS** solver in primal mode to compute flow fields (Eqn. 3.7) and objective function value (Eqn. 3.6).

b: Run **STAMPS** solver in adjoint mode to compute adjoint fields (Eqn. 3.17) and CFD sensitivity (Eqn. 3.27).

3: **Computation of total gradient** ($\frac{\partial J}{\partial \alpha}$)**:**

a: **NSPCC**: Run NSPCC CAD kernel in adjoint mode to compute shape sensitivity term and total gradient as given in Eqn. 4.30.

b: **Node-Based**: Run node-based kernel in adjoint mode to compute shape sensitivity term and total gradient as given in Eqn. 4.42.

4: **Optimiser**:

a: Compute perturbation to design variables ($\delta\alpha$) using SD or BFGS optimiser.

5: **Refinement:** If refinement trigger satisfied as given in Eqn. 4.31

a: **NSPCC**: Refine NURBS control net as presented in Sec. 4.9.2.

b: **Node-Based**: Reduce number of smoothing iteration (Sec. 4.11).

6: **Shape update**:

a: **NSPCC**: Run NSPCC CAD kernel in deformation mode to compute perturbations to control points ($\delta\mathbf{P}$) as presented in Sec. 4.26.

b: **Node-Based**: Run node-based kernel in deformation mode to compute smooth perturbations to surface mesh points (δX_s) (Sec.4.10).

7: **Computational Mesh Update::** (δX_v) (Sec. 3.3.2)

a: Run **STAMPS** solver in mesh deformation mode to deform volume mesh.

8: **Repeat** steps 2 through 7, until no further improvement in cost function is obtained.

Output:

a: **NSPCC**: Updated CAD model ($\mathbf{S}_{new}$) and computational mesh (X_v).

b: **Node-Based**: Updated computational mesh (X_v)..

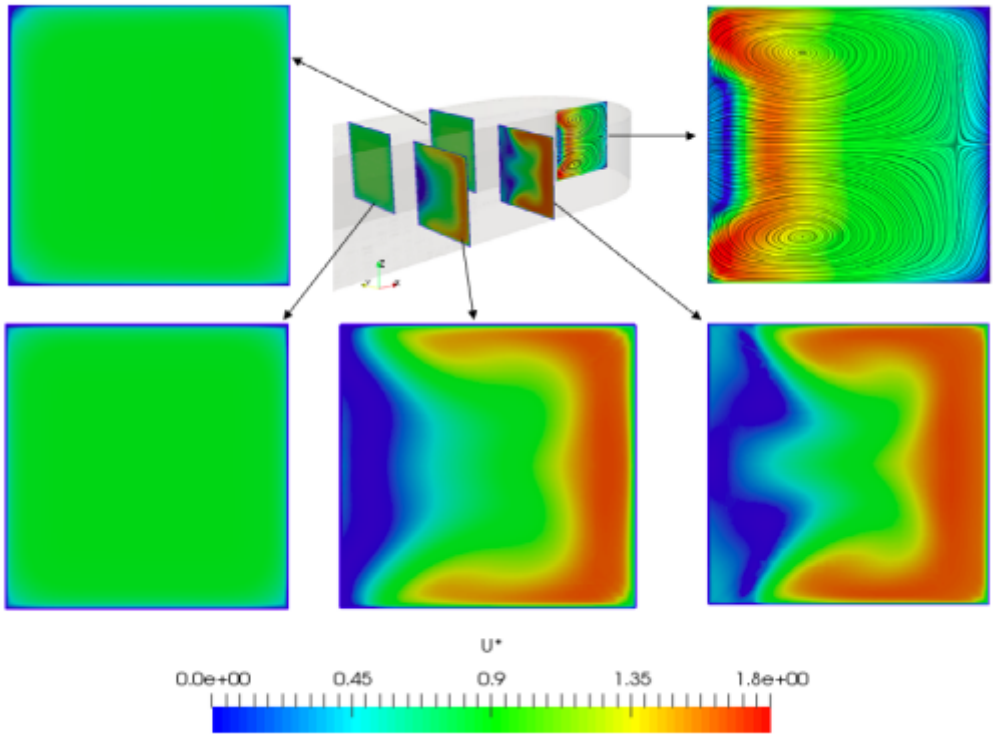


Figure 6.23: Flow field of the baseline geometry

able to reduce the effects of secondary flows and flow separation near the turn and exit channel of the U-Bend geometry.

6.5.3 Static NSPCC

Global refinement of control points on the design surface

Before testing the proposed adaptive parameterisation method, the influence of global refinement of control points on the regularity of the shapes captured in shape optimisation and its impact on the rate of convergence of the optimisation is investigated. Aerodynamic shape optimisation of internal turbine cooling channel is performed by using three levels of parameterisation coarse (L1), medium (L2) and fine (L3). Each optimisation is performed independently and control point distribuition on the design surface remain fixed throughout the optimisation. For the first two levels steepest descent method is used as the optimiser for the entire design process. However, for the third level the optimiser is switched to BFGS when the rate of convergence of SD algorithm becomes poor. Finer level parameterisation does not constrain the design, however it may provide multiple local minima in the

design process. Therefore for the finer level, when the rate of convergence of SD algorithm become too poor, it may converge to a sub-optimal solution, hence the optimiser has been switched to BFGS to search for a better solution. Figure 6.24 shows a comparison of the convergence histories of the objective function for all three levels of optimisation.

The value of SVD cut-off frequency σ_C determines the rank r' of the constraint matrix $\mathbf{C}$, hence influence the resultant modes and the number of design variables (N_{DV}). In this work, a value of $\sigma_C = 10^{-10}$ is used for all the optimisation runs. Each control points are allowed to deform in x, y, z direction and weights are remain fixed throughout the optimisation. Effect of varying σ_C is investigated in [94]. Table 6.1 shows the percentage drop in total pressure loss achieved in each optimisation and the number of design variables offered by the static NSPCC approach to each control net level. As the design space dimension increases, performance of the shape improves further. However this is not applicable to every case as the finer parameterisation may contain more local minima hence the gradient-based optimiser may converge to any local minima [133].

Figure 6.25 shows the comparison of optimised shapes obtained using three different parameterisation. Opt-L1 represents the optimum shape obtained using coarse control net level (L1) as a baseline shape. Similarly, opt-L2 and opt-L3 represents the optimum shapes obtained using L2 and L3 control net level respectively. One of the interesting features of the opt-L3 design is the presence of surface undulations that the other two levels failed to capture. This is mainly due to the availability of strong local shape control on L3 parameterisation which generates high frequency shape modes during the design process whereas other two levels handle low-frequency shape modes than L3 in the design space. This can be clearly seen in the Fig. 6.4 which shows L3 parameterisation exhibit a very small region of influence for a control point over the computational mesh than other two levels. Furthermore, the optimised geometries obtained using global refinement almost preserves the symmetric nature of the baseline shape which can be seen in Fig. 6.26 which compares the inner U-Bend region of the optimised geometries obtained using static NSPCC approach.

Hradil et al. [59] pointed out that finer parameterisation may capture surface

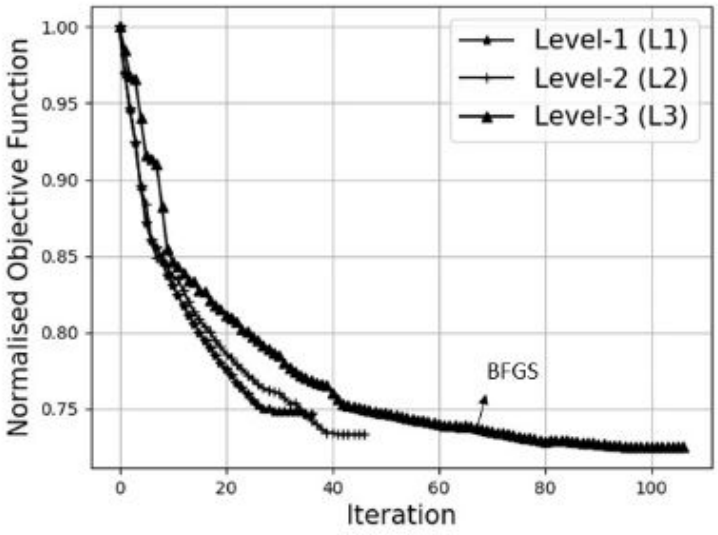


Figure 6.24: Convergence of objective function: Static NSPCC

Level	**Control net dimension**	**Total number of control points (3N)**	**Size of the design space** (N_{DV})	**Percentage drop in total pressure loss**
L1	4×6	864	442	-25.34%
L2	6×8	1728	1082	-26.67%
L3	8×10	3456	2100	-27.52%

Table 6.1: Optimisation results obtained using static NSPCC

wiggles even if the shape parameterisation generate smooth shapes in the design space which may also affects the rate of convergence of the gradient-based shape optimisation process. When compared with the baseline shape, the optimum shape obtained using level 3 parameterisation shows better performance improvement than other levels. However, the design space richness and the presence of surface wiggles affects the rate of convergence of the optimisation. For the current application, control point refinement upto L3 is sufficient, further refinement might generate very high surface undulations in the design space which may not be resolved by the CFD solver hence additional surface regularisation or gradient smoothing might be needed.

6.5.4 Adaptive Parameterisation

In the adaptive refinement study, optimisation starts with the coarser control net (L1) and they are refined locally using knot insertion algorithm without modifying

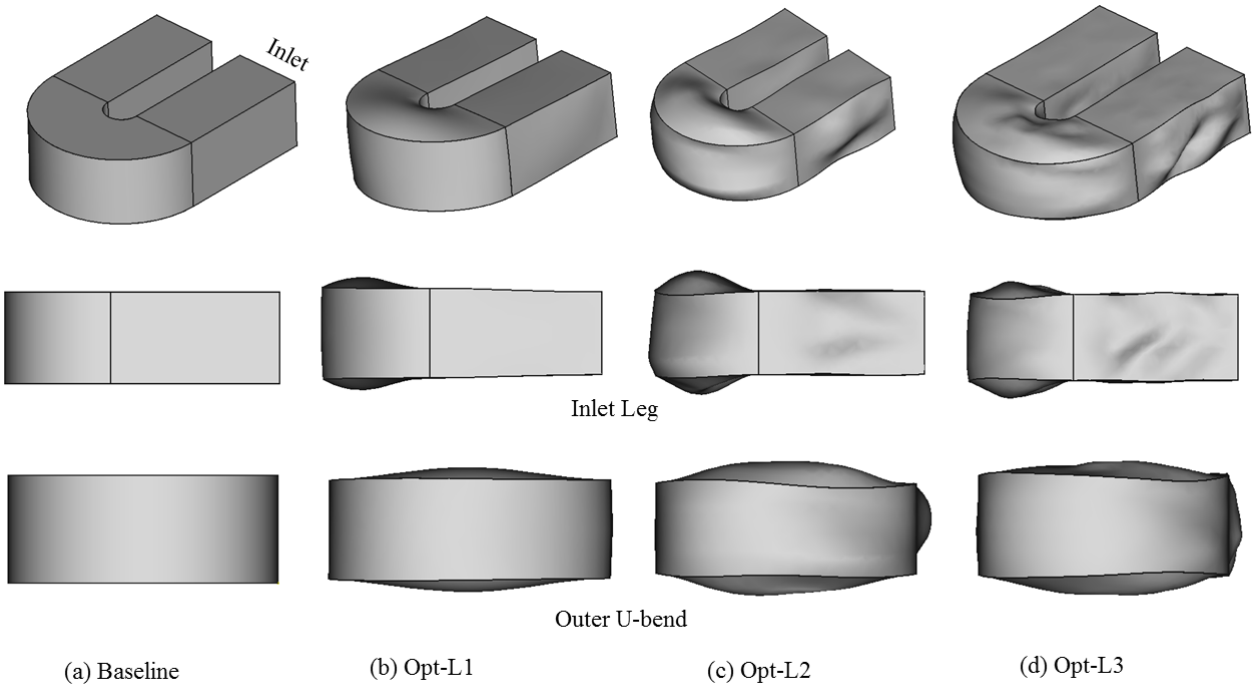


Figure 6.25: Comparison of optimised geometries: Static-NSPCC

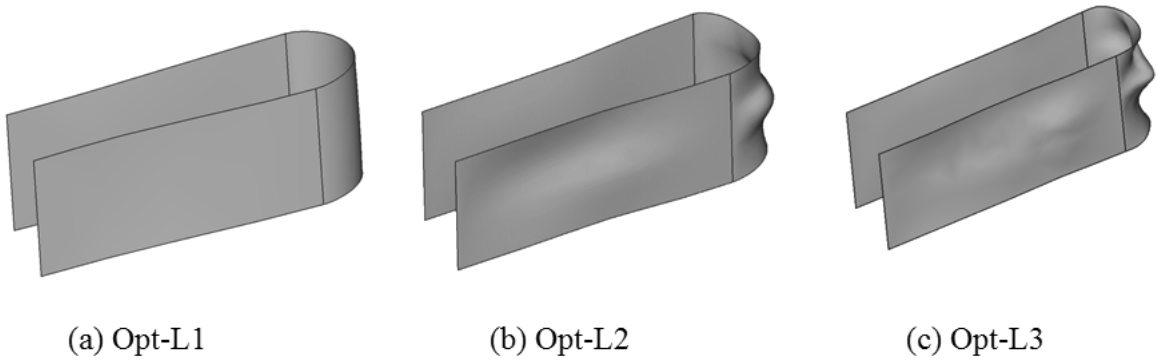


Figure 6.26: Comparison of inner U-Bend shape: Static NSPCC

the shape. The refinement criteria is driven by the smoothed node-based adjoint sensitivity information and the control nets of all the 12 design NURBS patches are refined only in the region where the gradient modes are high. During the optimisation, the design space has been refined three times. Figure 6.27 compares the convergence history of the objective function value between static and adaptive NSPCC methods. Table 6.2 shows the comparison of number of design variables and the performance improvement achieved between static and adaptive NSPCC method. Shape optimisation using the refined design space is able to reduce the objective function value further and has the potential to capture important shape modes outside the fixed envelope offered by the static design space. However, no appropriate improvement can be made with the fourth design space level. Figure 6.28 shows the comparison of optimum geometries obtained using static and adaptive NSPCC method. Figure 6.29 shows the convergence history of the control points distribution on the design surfaces. Adaptive refinement does not preserve the symmetric nature of the control net hence it has more potential to capture asymmetric shape modes in the design process. This can be clearly seen in Fig. 6.30 which shows the evolution of the inner U-Bend region in the design process.

Figure 6.31 shows the comparison of the convergence history of the objective function between adaptive NSPCC and adaptive node-based parameterisation methods. In the adaptive node-based parameterisation method, optimisation starts with $n = 30$ smoothing iterations and with each refinement trigger, the number of smoothing iterations is reduced by 5. When compared with the node-based method, NSPCC offers better convergence as the shape deformation modes are orthogonal to each other which may not be the case for the node-based method as the shape modes are not independent of each other. For the current application, the adaptive-node based method is converged to local optima. Furthermore, the resulting optimal shape still contain certain high-frequency shape modes. This can be seen in Fig. 6.32 which shows the optimised mesh obtained using adaptive-node based method.

6.5.5 Flowfield of the optimised geometry

When compared with the baseline configuration, optimised geometries obtained using both the parameterisation methods suppress flow separation near the inner wall

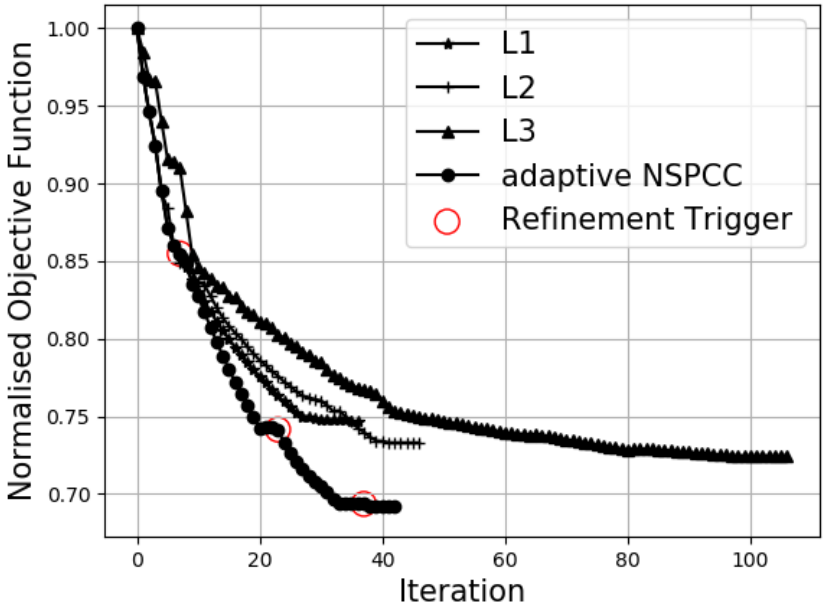
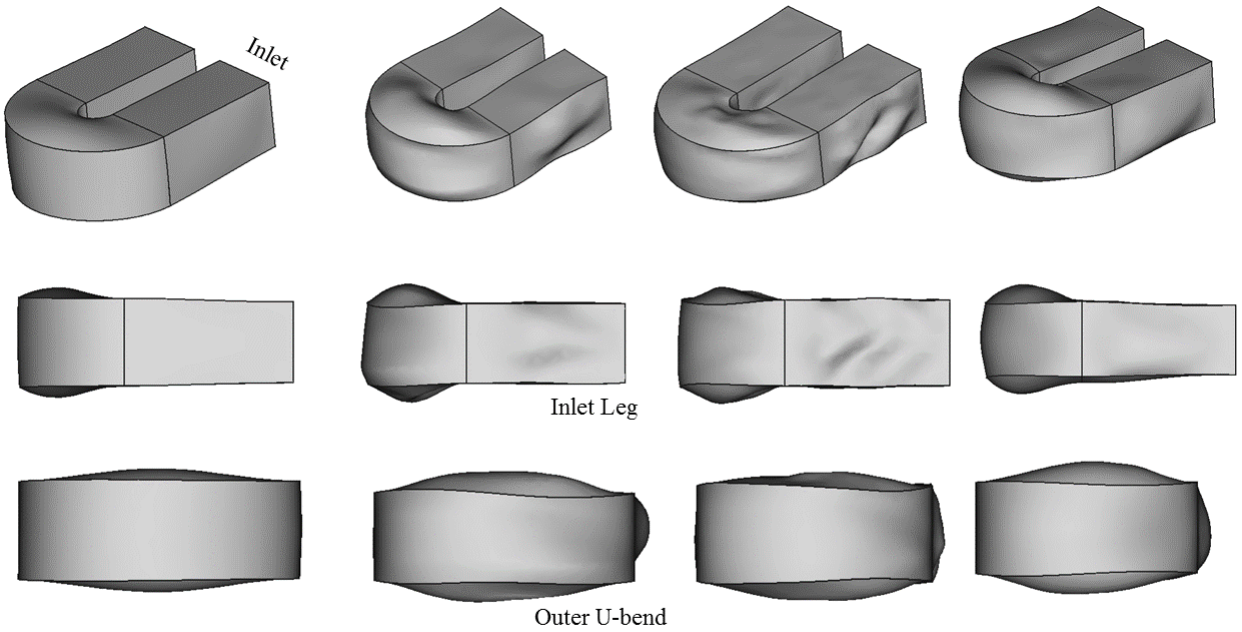


Figure 6.27: Convergence of objective function: Static vs Adaptive NSPCC

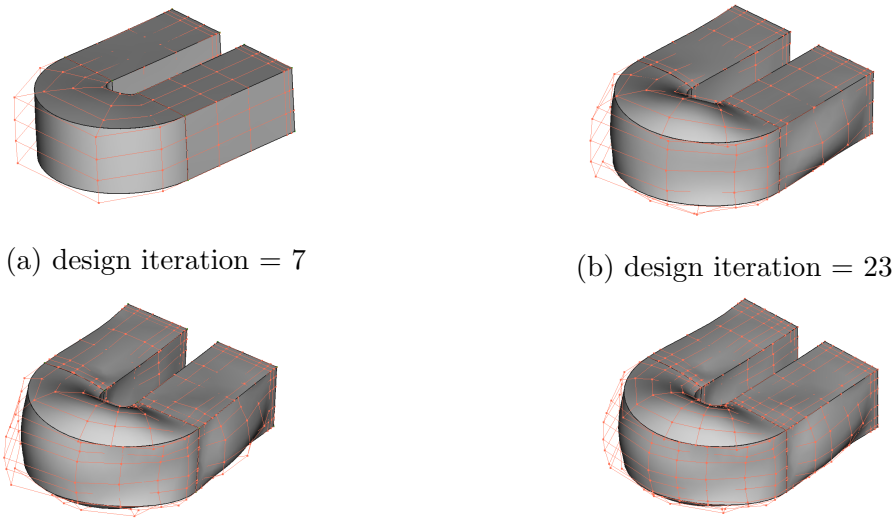
Level	**Control net dimension**	**Total number of control points (3N)**	**Size of the design space**	**Percentage drop in total pressure loss**
L1	4×6	864	442	-25.34%
L2	6×8	1728	1082	-26.67%
L3	8×10	3456	2100	-27.52%
Adaptive NSPCC				
Iter 1-7	4×6	864	574	-14.51%
Iter 8-23	5×7	1260	736	-25.68%
Iter 24-37	6×8	1728	1026	-30.63%
Iter 42	7×9	2268	1402	-30.8%

Table 6.2: Optimisation results: static NSPCC vs Adaptive NSPCC



(a) Opt-L1 (b) Opt-L2 (c) Opt-L3 (d) Adaptive

Figure 6.28: Comparison of optimised geometries: Static NSPCC vs Adaptive NSPCC



(a) design iteration = 7 (b) design iteration = 23

(c) design iteration = 37 (d) design iteration = 42

Figure 6.29: Convergence of control points distribution: Adaptive NSPCC

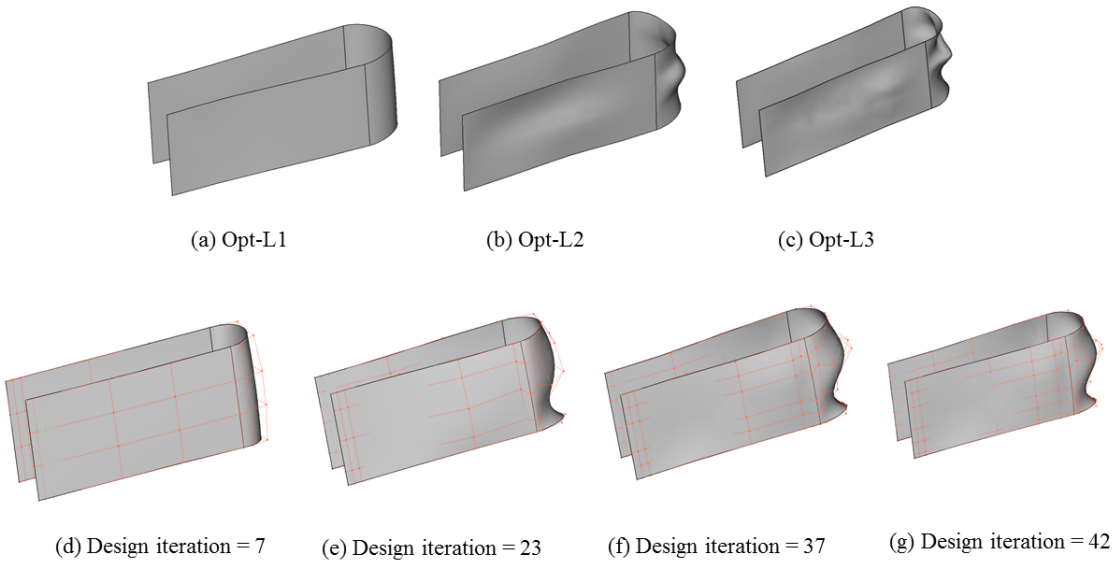


Figure 6.30: Comparison of inner U-Bend region: Static NSPCC vs Adaptive NSPCC

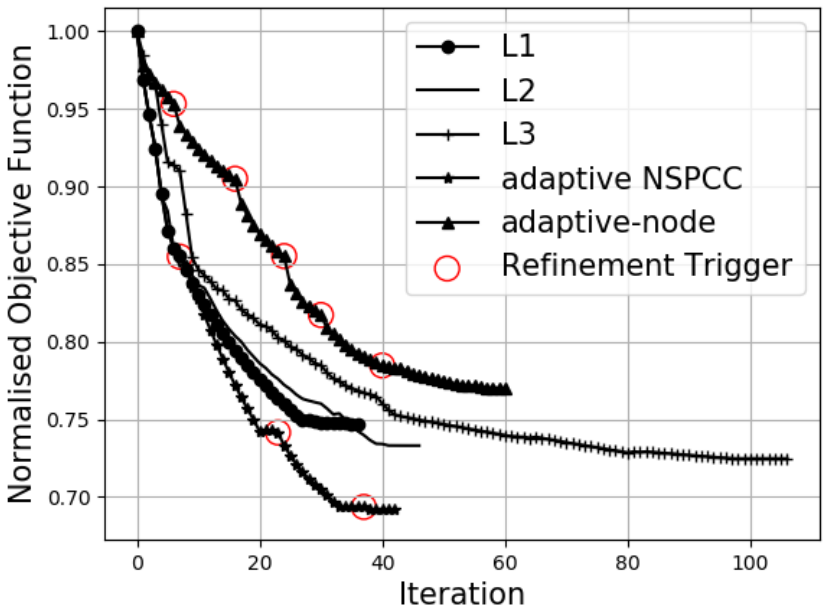


Figure 6.31: Convergence of objective function: Adaptive NSPCC vs adaptive node-based

Figure 6.32: Optimised mesh: Adaptive node-based

of the exit channel and reduces wall shear stress significantly. Figure 6.33 shows the comparison of velocity magnitude taken at mid symmetry plane between baseline and optimised geometries. The reason for the design improvement is threefold. Firstly, all levels of parameterisation altered the radius of the inner U-Bend, this can be clearly seen in Fig 6.30 which shows the comparison of inner U-Bend region between baseline and optimised geometries. For incompressible and irrotational flow the velocity gradient normal to the streamline is proportional to the curvature of the streamline. Hence the optimised geometries with enhanced radius of curvature reduces the required radial pressure gradient and hence the streamwise adverse pressure gradient, resulting in a smaller separation zone.

Secondly, the duct section is considerably enlarged for all the optimised geometries, this can be clearly seen in Fig. 6.34 which compares the CS area taken at the 90° turn region, all the optimum geometries exhibit larger CS area than the baseline geometry. Hence reducing the velocity in the bend which, similar to the radius increase, reduces the required centripetal forces, hence the required radial pressure gradient, hence the separation zone. When compared with the other geometries, the optimised shape obtained using adaptive NSPCC shows larger cross-sectional area which further reduces the pressure gradient normal to the stream line. Lower normal pressure gradient at the turn generates weaker secondary vortex which significantly reduces the associated diffusion loss. In addition, weaker secondary vortex reduces velocities near the outer wall of the exit channel which reduce the wall shear stress further.

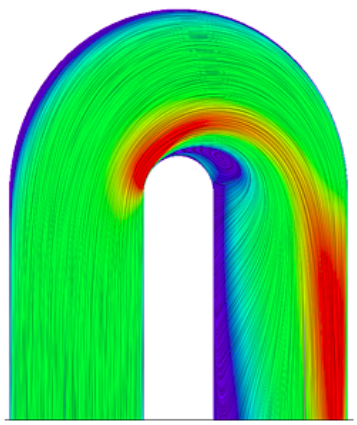
Finally, a third contribution is obtained by the formation of strake like shape along the vertical direction of the inner U-Bend such as widely used on airplanes and pipelines. At the inner part of the U-Bend, a low velocity region was observed which is shown in the Fig. 6.34. The size of this region is more for baseline geometry

Computation	**Percentage over total time**		
	L1	**L2**	**L3**
Primal	44.99	44.87	44.74
Adjoint	54.75	54.59	54.43
Surface Mesh Mapping	0.16	0.22	0.33
SVD null space	0.05	0.22	0.34
CAD perturbation	0.01	0.01	0.02
Constraint recovery	0.02	0.05	0.11

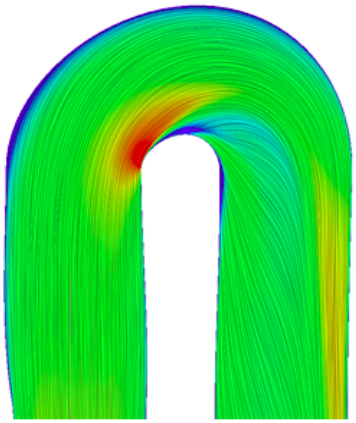
Table 6.3: Computational time breakdown for a single design step

than others. From Fig. 6.34c and Fig. 6.34d it is interesting to note that inner U-Bend region of the opt-L3 captures strong convex or hump-like shape mode along vertical direction however the opt-L2 and adaptive NSPCC geometry exhibit weak convex like shape whereas opt-L1 and adaptive node-based shape doesn't include this local shape mode. This strake like shape mode formed at the center of the inner U-Bend that splits the counter-rotating vortices hence re-energies and reduces the low velocity region in the optimum geometries.

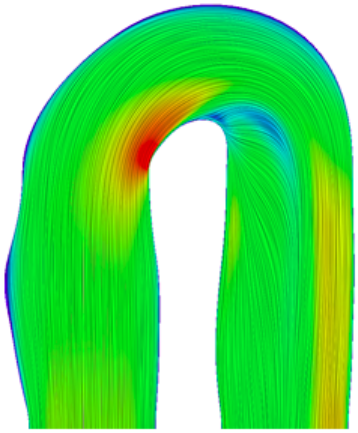
The costs associated with each of these steps for three levels of parameterisation at first design step in which primal and adjoint CFD solves upto full convergence are shown in Table 6.3.



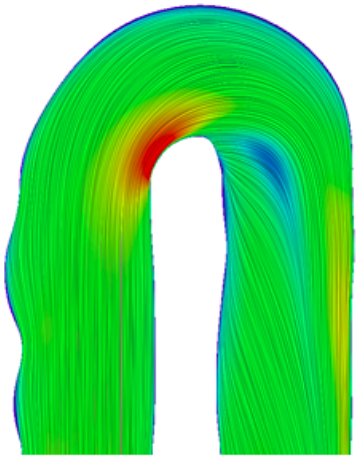
(a) Baseline



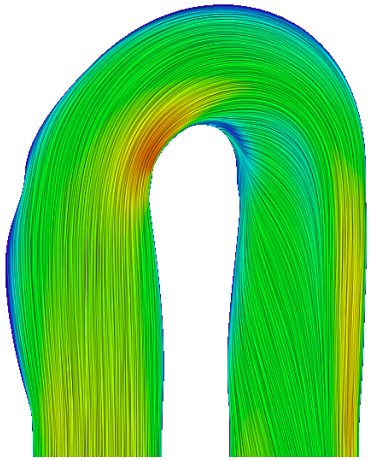
(b) opt-L2



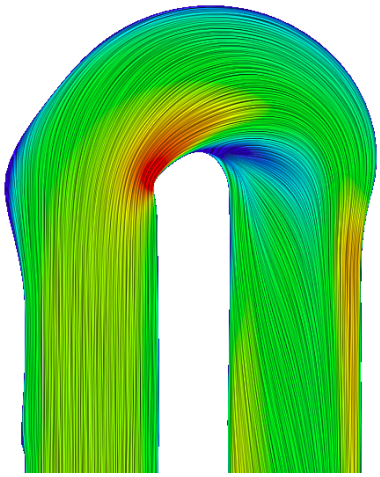
(c) opt-L2



(d) opt-L3

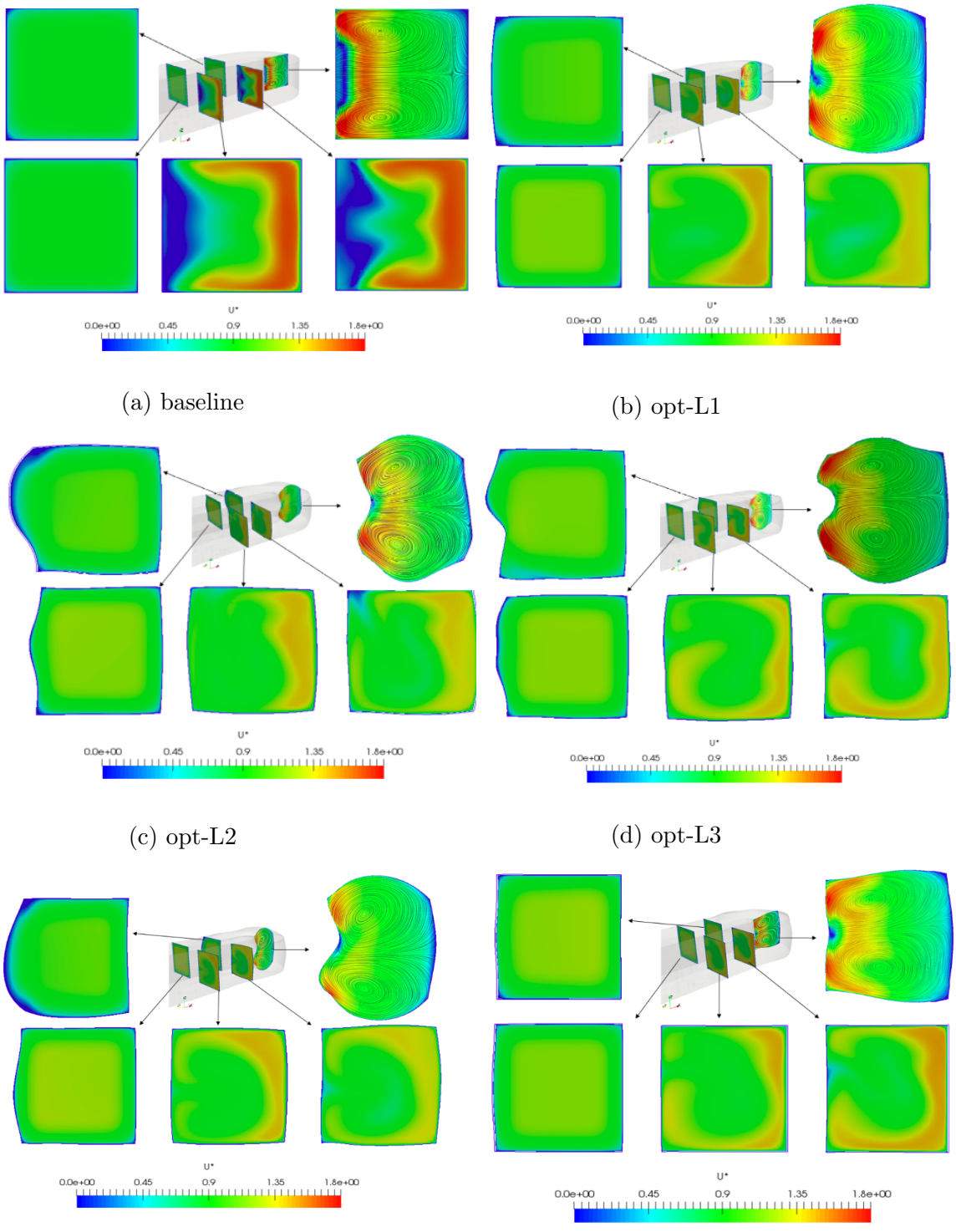


(e) adaptive NSPCC



(f) adaptive node-based

Figure 6.33: Comparison of velocity magnitude between optimised geometries. Cross section taken at middle plane.



(a) baseline

(b) opt-L1

(c) opt-L2

(d) opt-L3

(e) adaptive NSPCC

(f) adaptive node-based.

Figure 6.34: Comparison of secondary flow structure between optimised geometries. Cross-section taken at 90° turn region

Chapter 7

Conclusion and Future Work

The aim of this thesis work is to develop a NURBS-based adaptive parameterisation method for shape optimisation. NURBS have become the de-facto industry standard for boundary representation(BRep) and data exchange between different CAD systems. The BRep in the typical standardised STEP format represents the geometry as a collection of NURBS patches. One can use the control points of the NURBS patches to deform geometry in the shape optimisation. However a finite displacement of control points near or at the patch interface leads to the violation of geometric constraints such as $G_0 - G_2$ between patches. Researchers from the CFD optimisation group at QMUL have developed the NURBS-based parameterisation method with Complex Constraints (NSPCC) method for shape optimisation. NSPCC approach has many advantages. They are:

1. **Local shape control:** NSPCC offers wide range of shape modes in the design space though the use of local shape modification property of NURBS.

2. **Robustness**: Surface mesh mapping is done via parametric coordinates this not only preserves the characteristics of original surface mesh but also handles large CAD and surface mesh deformation without any failure.

3. **Efficient constraint handling:** Both geometric and manufacturing constraints are handled simultaneously via test point approach. Based on the smoothness requirements different geometric continuities such as G_0, G_1 and G_2 can be maintained at each NURBS patch interfaces.

4. **CAD sensitivities:** NSPCC CAD kernel bas been differentiated using source

transformation tool TAPENADE in forward mode which provide exact CAD sensitivities for shape optimisation.

5. **Portability:** Preserves CAD geometry in the design loop hence easy to transfer between disciplines for MDAO based applications and/or manufacturing. Furthermore, a valid CAD model is obtained in each design step hence it can be used for inspection and/or remeshing during premature termination of the design loop.

Shape optimisation often required a large number of design variables to capture important shape modes in the design process. Gradient-free methods are computationally expensive when large number of design variables are employed in the design process. Therefore gradient-based methods are required to handle rich design space. However the adjoint method is essential when using gradient-based optimiser in the loop which computes the gradient of the objective function independent to the number of design variables. The major contributions of this thesis are as follows:

7.0.1 Reverse differentiation of the entire design chain:

In this work, CFD sensitivities are computed by using in-house flow and discrete adjoint solver named STAMPS. Previously, NSPCC CAD kernel has been differentiated in a forward mode [142, 143, 65] where computational costs for computing CAD sensitivity is proportional to the number of design variables. In this present work, the NSPCC CAD kernel is differentiated in reverse mode, therefore the entire design chain in now differentiated using source transformation AD tool in reverse mode which computes adjoint sensitivities in an efficient manner.

In Section 6.3.2, performance of the reverse differentiation of the NSPCC CAD kernel is tested by using six different control net levels of the U-Bend geometry L1, L2, L3, L4, L5 and L6 with the total number of control points $288, 576, 1152, 2304, 4752$ and 9360 respectively. The results of the performance analysis shows that forward mode differentiation of the NSPCC CAD kernel affects the overall performance when a large number of control points are employed in the design process

7.0.2 Influence of control points distributions on the shape optimisation process:

Using NURBS one can represent a shape with a wide range of control net distributions starting from a coarser level that fits a shape within a suitable minimum level of tolerance to a finer parameterisation. However, it is a daunting task to determine the suitable parameterisation a-prior to the shape optimisation process. Therefore it is important to investigate the influence of the parameterisation on the shape optimisation process.

In this present work, aerodynamic shape optimisation of internal turbine blade cooling channel is performed using three levels of control points distribution L1, L2, and L3 with total number of control points $288, 576$, and 1152 respectively (Section 6.5.3). Control points are refined globally on the design surface and each optimisation is performed independently with fixed design space. Results of the shape optimisation shows that, finer parameterisation not only improves the objective function but also creates surface undulations even if the shape parameterisation method mostly captures smooth shapes. Furthermore, the optimisation performed using finer parameterisation is driven by high-frequency shape modes which causing the optimisation to stall and may converge to any local minima. This behavior is widely observed with finer parameterisation [70, 133] and gradient smoothing may be required to damp out high-frequency shape modes in the design space [61] to accelerate convergence.

7.0.3 Adaptive Design Space

In this present work, NSPCC approach is extended to include adaptive design space for shape optimisation by refining the control points locally on the design surface using knot insertion algorithm. The proposed adaptive parameterisation method has been used to reduce the pressure loss of the internal turbine cooling channel U-Bend (Section 6.5.4). Optimisation begins in a coarser design space focusing on low-frequency shape modes and then automatically refining the parameterisation to include high-frequency shape modes only in the regions where significant high-frequency surface sensitivities are detected. Design space enrichment is performed by

inserting knots in both parameter directions of the NURBS patches. The refinement procedure is fully automatic and minimal user input is required to setup the design space for the optimisation.

This approach is both efficient and complete by eliminating the arbitrary trade-off between the dimension and distribution of the design variables in the design space. The optimised geometry obtained using adaptive parameterisation outperforms static parameterisation. In addition, the resultant geometry is smoother than the optimum geometry obtained using finer parameterisation (L3). This is due to the fact that early design stages are driven by low-frequency shape modes and high-frequency shape modes are permitted as the design approaches the minimum. This characteristics behavior is found desirable for shape optimisation which captures better shape modes in the design process and also reduce the stiffness of the optimisation [11, 83, 65]. In this work, a simple explicit adaptive surface regualarisation method is also proposed in which optimisation is started with large amount of smoothing iterations and as the design evolves amount of smoothing is reduced to achieve better convergence rate. When compared with adaptive node-based method, both static and adaptive NSPCC method shows better convergence and also converge to better solution.

7.1 Recommendations and Future Work

This thesis presents the successful integration of reverse differentiated adaptive CAD-based shape parameterisation into gradient-based aerodynamic shape optimisation process which further opened the research direction for future work.

- Needs to investigate the efficiency of the proposed parameterisation method with other shape optimisation benchmark test cases. Currently work is in progress by the author to test the robustness of the adaptive parameterisation in the shape optimisation of constrained drag reduction of Onera M6 wing.

- The adaptive refinement criteria presented in this work needs to be investigated further to include large number of candidates for the selection process.

- The proposed adaptive NSPCC method has been tested in shape optimisa-

tion based on single discipline. As presented above, adaptive NSPCC method preserves CAD model in the loop hence this can be extended to handle multi-objective functions. Currently other PhD student from the same research work is testing the efficiency of the method in the shape optimisation of turbine blade cooling channel including both fluid flow and conjugate heat transfer analysis.

- The proposed parameterisation is suitable to handle untrimmed and conformal patch topology which has topologically four edges. The extended NSPCC to handle trimmed and intersecting NURBS patches in the shape optimisation are in close to completion by other team members in CFD-optimisation group at QMUL.

- Fortran-based NSPCC CAD kernel needs to be coupled with open source CAD engine such as openCASCADE for reading and writing STEP files. Therefore boundary representation of any complex geometry can be extracted. Current version of the CAD kernel depends on python-based STEP parser developed by the author which can read and write STEP file which has conformal patch topology.

- Fortran-based NSPCC CAD kernel needs to be coupled with other open source NURBS library such as SISL-The SINTEF Spline Library. The SISL library is written in C and offers wide range of NURBS handling algorithms such as surface-surface intersection computation and NURBS fitting for curves and surfaces. The work is in initial stage by the author and some initial results by the author can be found in [106]. To authors knowledge reverse differentiation of the intersection algorithm is not yet performed. Reverse differentiation of the design chain with intersection would be very useful to handle shape optimisation of intersecting components such as wing and fuselage.

- Currently CFD sensitivities are computed using in-house flow and discrete adjoint solver STAMPS, current version of the solver does not support different turbulence models for handling complex test case. The developed methodology is a stand-alone tool which can be coupled with other open source solvers such as SU2, openFOAM etc.

- NSPCC method needs to be extended to include multi-level parameterisation in the design process and needs to tested with the adaptive NSPCC method.

Appendix A

Author's Publications and Presentations

A.1 Journal Papers

1. **Rejish Jesudasan**, Mateusz Gugala and Jens-Dominik Müeller *'CAD-Based Parameterisation Framework for Aerodynamic Shape Optimisation Using Adjoint Sensitivities'*. Journal: Structural and Multi-diciplinary Optimisation (To be submitted)

2. **Rejish Jesudasan**, Mateusz Gugala and Jens-Dominik Müeller *'CAD-based Adaptive Shape Optimisation Using Adjoint Sensitivities'*. Journal: Advances in Engineering Software (To be submitted)

3. **Rejish Jesudasan**, Mateusz Gugala and Jens-Dominik Müeller *'CAD-free vs CAD based parameterisation method in adjoint based aerodynamic shape optimisation'*. Journal: Numerical Method in Fluids (In preparation)

A.2 Conference Papers

1. **Rejish Jesudasan**, Xingchen Zhang and Jens-Dominik Müeller *'Adjoint Optimisation of Internal Turbine Cooling Channel Using NURBS-Based Automatic And Adaptive Parameterisation Method'*. ASME-2017, Gas Turbine India. 07-08 December 2017, Bangalore, India.

2. Xingchen Zhang, **Rejish Jesudasan** and Jens-Dominik Müeller *"Adjoint-based Aerodynamic Optimisation of Wing Shape Using Non-Uniform Rational B-splines"*, EUROGEN-2017. 13-15 September 2017, Madrid, Spain.

A.3 Conference Presentations

1. **Rejish Jesudasan**, Pavanakumar Mohanamuraly and Jens-Dominik Müller *'CAD-based adaptive parameterisation framework for aerodynamic shape optimisation'*.ECCM ECFD IODA mini-symposium 13th June 2018 Glasgow

2. **Rejish Jesudasan**, Orest Mykhaskiv and Jens-Dominik Müeller *'Surface Mesh Adaptation Method for Handling Wing-Fuselage Intersections'* EUROGEN-2017. 13-15 September 2017, Madrid, Spain.

3. **Rejish Jesudasan**, Orest Mykhaskiv, Mateusz Gugala and Jens-Dominik M̆eller *'Adjoint Optimisation of Internal Turbine Cooling Channel Using Node and CAD-Based Automatic Parameterisation Methods'* NAFEMS-2016, European Conference: Simulation Based Optimisation. 12-13 October 2016, Manchester UK.

4. **Rejish Jesudasan**, Mateusz Gugala, Orest Mykhaskiv and Jens-Dominik Müeller *'A comparison of node based and CAD-based parameterisations in shape optimisation'* International Conference on Numerical Optimisation Methods for Engineering Design (NOED2016) ;TU Munich, Germany, July 2016.

5. **Rejish Jesudasan**, Xingchen Zhang , Mateusz Gugala and Jens-Dominik Müeller *'CAD-free vs CAD based parameterisation method in adjoint based aerodynamic shape optimisation'*. 7th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS Congress 2016) ; Crete Island, Greece, June 2016.

Bibliography

[1] https://confluence.cornell.edu/display/simulation/fluent

[2] https://mdolab.engin.umich.edu/wiki/aerodynamic-design-optimization-workshop.html.

[3] https://www.grc.nasa.gov/www/wind/valid/m6wing/m6wing.html. Technical report.

[4] D. Agarwal, S. Marques, T. T. Robinson, C. G. Armstrong, and P. Hewitt. Aerodynamic shape optimization using feature based cad systems and adjoint methods. In *18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, page 3999, 2017.

[5] D. Agarwal, T. T. Robinson, C. G. Armstrong, and C. Kapellos. Enhancing cad-based shape optimisation by automatically updating the cad model's parameterisation. *Structural and Multidisciplinary Optimization*, pages 1–16, 2018.

[6] P. Aidala, W. DAVIS, JR, and W. Mason. Smart aerodynamic optimization. In *Applied Aerodynamics Conference*, page 1863, 1983.

[7] G. Alessi, T. Verstraete, L. Koloszar, and J. P. A. J. van Beeck. Comparison of large eddy simulation and reynolds-averaged navier–stokes evaluations with experimental tests on u-bend duct geometry. *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy*, 234(3):315–322, 2020.

[8] C. B. Allen, D. J. Poole, and T. C. Rendall. Wing aerodynamic optimization using efficient mathematically-extracted modal design variables. *Optimization and Engineering*, 19(2):453–477, 2018.

[9] C. B. Allen and T. C. Rendall. Cfd-based optimization of hovering rotors using radial basis functions for shape parameterization and mesh deformation. *Optimization and Engineering*, 14(1):97–118, 2013.

[10] G. Anderson, M. Aftosmis, and M. Nemec. Parametric deformation of discrete geometry for aerodynamic shape design. In *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, page 965, 2012.

[11] G. R. Anderson and M. J. Aftosmis. Adaptive shape parameterization for aerodynamic design. *Nat. Aeronaut. Space Admin., Ames Res. Center, Moffett Field, CA, USA, NAS Tech. Rep. NAS-2015-02*, 2015.

[12] M. Andreoli, J. Ales, and J.-A. Désidéri. *Free-form-deformation parameterization for multilevel 3D shape optimization in aerodynamics.* PhD thesis, INRIA, 2003.

[13] C. G. Armstrong, T. T. Robinson, H. Ou, and C. Othmer. Linking adjoint sensitivity maps with cad parameters. *Evolutionary methods for design, optimization and control*, pages 234–239, 2007.

[14] S. Arnout, G. Lombaert, G. Degrande, G. De Roeck, and K. Bletzinger. Optimal design of shells with fe based parameterization. In *Proceedings of the 9th National Congress on Theoretical and Applied Mechanics NCTAM*, volume 2012, 2012.

[15] S. Auriemma. Applications of differentiated cad kernel in gradient-based aerodynamic shape optimisation. In *2018 Joint Propulsion Conference*, page 4828, 2018.

[16] J. Q. Bai and S. Chen. Aerodynamic optimization design of airfoil based on directly manipulated ffd technique. In *Applied Mechanics and Materials*, volume 390, pages 121–128. Trans Tech Publ, 2013.

[17] T. Bai, J. Liu, W. Zhang, and Z. Zou. Effect of surface roughness on the aerodynamic performance of turbine blade cascade. *Propulsion and Power Research*, 3(2):82–89, 2014.

[18] M. Banovic, O. Mykhaskiv, S. Auriemma, A. Walther, H. Legrand, and J.-D. Müller. Automatic differentiation of the Open CASCADE Technology CAD system and its coupling with an adjoint cfd solver. *Optimization Methods and Software*, 33(4-6):813–28, 2017.

[19] M. Banović, O. Mykhaskiv, S. Auriemma, A. Walther, H. Legrand, and J.-D. Müller. Algorithmic differentiation of the open cascade technology cad kernel and its coupling with an adjoint cfd solver. *Optimization Methods and Software*, 33(4-6):813–828, 2018.

[20] K.-U. Bletzinger. A consistent frame for sensitivity filtering and the vertex assigned morphing of optimal shape. *Structural and Multidisciplinary Optimization*, 49(6):873–895, 2014.

[21] K.-U. Bletzinger, M. Firl, J. Linhard, and R. Wüchner. Optimal shapes of mechanically motivated surfaces. *Computer methods in applied mechanics and engineering*, 199(5-8):324–333, 2010.

[22] W. Brock, C. Burdyshaw, S. Karman, V. Betro, B. Hilbert, K. Anderson, and R. Haimes. Adjoint-based design optimization using cad parameterization through capri. In *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, page 968, 2012.

[23] A. Bueno-Orovio, C. Castro, F. Palacios, and E. Zuazua. Continuous adjoint approach for the spalart-allmaras model in aerodynamic optimization. *AIAA journal*, 50(3):631–646, 2012.

[24] P. Castonguay and S. Nadarajah. Effect of shape parameterization on aerodynamic shape optimization. In *45th AIAA Aerospace Sciences Meeting and Exhibit*, page 59, 2007.

[25] I.-C. Chang, F. J. Torres, C. Tung, U. A. Aviation, T. Command, et al. Geometric analysis of wing sections. 1995.

[26] J. Chen, B. Cao, Y. Zheng, L. Xie, C. Li, and Z. Xiao. Automatic surface repairing, defeaturing and meshing algorithms based on an extended b-rep. *Advances in Engineering Software*, 86:55–69, 2015.

[27] F. Christakopoulos. *Sensitivity computation and shape optimisation in aerodynamics using the adjoint methodology and Automatic Differentiation.* PhD thesis, Queen Mary University of London, 2012.

[28] F. Christakopoulos, D. Jones, and J.-D. Müller. Pseudo-timestepping and verification for automatic differentiation derived cfd codes. *Computers & Fluids*, 46(1):174–179, 2011.

[29] F. Coletti, T. Verstraete, J. Bulle, T. Van der Wielen, N. Van den Berge, and T. Arts. Optimization of a u-bend for minimal pressure loss in internal cooling channels part ii: Experimental validation. *Journal of Turbomachinery*, 135(5):051016, 2013.

[30] P. Cusdin. *Automatic sensitivity code for Computational Fluid Dynamics.* PhD thesis, School of Aeronautical Engineering, Queen's University Belfast, 2005.

[31] P. Cusdin and J.-D. Müller. Deriving linear and adjoint codes for cfd using automatic differentiation. *Preprint submitted to AIAA J., www. ea. qub. ac. uk/jmueller*, 2003.

[32] F. Daoud, M. Firl, and K. Bletzinger. Filter techniques in shape optimization with cad-free parameterization. In *Proceedings of 6th World Congress of Structural and Multidisciplinary Optimization*, 2005.

[33] D. Dasgupta and Z. Michalewicz. *Evolutionary algorithms in engineering applications.* Springer Science & Business Media, 2013.

[34] K. Deb. Multi-objective optimisation using evolutionary algorithms: an introduction. In *Multi-objective evolutionary optimisation for product design and manufacturing*, pages 3–34. Springer, 2011.

[35] T. DeRose, M. Kass, and T. Truong. Subdivision surfaces in character animation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 85–94. Citeseer, 1998.

[36] J. Desideri and A. Dervieux. Hierarchical methods for shape optimization in aerodynamics i: Multilevel algorithms for parametric shape optimization. *Lecture Series-Von Kármán Institute for Fluid Dynamics*, 3:10, 2006.

[37] J.-A. Désidéri, B. A. El Majd, and A. Janka. Nested and self-adaptive bézier parameterizations for shape optimization. *Journal of Computational Physics*, 224(1):117–131, 2007.

[38] B. A. El Majd, J.-A. D 'e sid é ri, and R. 'e. g. Duvigneau. Multilevel strategies for parametric shape optimization in aerodynamics. *European Journal of Computational Mechanics / European Review of Digital Mechanics*, 17(1-2):149–168, 2008.

[39] M. Firl. *Optimal shape design of shell structures*. PhD thesis, Technische Universität München, 2010.

[40] A. Forrester, A. Sobester, and A. Keane. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons, 2008.

[41] R. Forsythe. *A partitioned approach to fluid-structure interaction for artificial heart valves*. PhD thesis, Queen's University of Belfast, 2006.

[42] S. Forth, P. Hovland, E. Phipps, J. Utke, and A. Walther. *Recent advances in algorithmic differentiation*, volume 87. Springer Science & Business Media, 2012.

[43] D. Fudge, D. Zingg, and R. Haimes. A cad-free and a cad-based geometry control system for aerodynamic shape optimization. In *43rd AIAA Aerospace Sciences Meeting and Exhibit*, page 451, 2005.

[44] M. Gallo and T. Astarita. 3d reconstruction of the flow and vortical field in a rotating sharp u turn channel. *Experiments in Fluids*, 48(6):967–982, 2010.

[45] S. Ghoman, Z. Wang, P. Chen, and R. Kapania. A pod-based reduced order design scheme for shape optimization of air vehicles. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 20th AIAA/ASME/AHS Adaptive Structures Conference 14th AIAA*, page 1808, 2012.

[46] K. C. Giannakoglou and M. Karakasis. Hierarchical and distributed metamodel-assisted evolutionary algorithms. *LECTURE SERIES-VON KARMAN INSTITUTE FOR FLUID DYNAMICS*, 3:6, 2006.

[47] R. Giering. Tangent linear and adjoint model compiler. *User manual, TAMC version*, 4, 1997.

[48] A. Griewank, D. Juedes, and J. Utke. Algorithm 755: Adol-c: a package for the automatic differentiation of algorithms written in c/c++. *ACM Transactions on Mathematical Software (TOMS)*, 22(2):131–167, 1996.

[49] A. Griewank and A. Walther. *Evaluating derivatives: principles and techniques of algorithmic differentiation*, volume 105. Siam, 2008.

[50] M. Gugala. *Output-based mesh adaptation using geometric multi-grid for error estimation.* PhD thesis, School of Engineering and Materials Science, Queen Mary University of London UK., 2018.

[51] R. Haimes and G. J. Follen. Computational analysis programming interface. In *Numerical grid generation in computational field simulations*, pages 663–672. Proceedings of the 6th International Conference on Numerical Grid Generation in Computaional Fluid Simulations, 1998.

[52] X. Han and D. W. Zingg. An adaptive geometry parameterization for aerodynamic shape optimization. *Optimization and Engineering*, 15(1):69–91, 2014.

[53] L. Hascoet and V. Pascual. The tapenade automatic differentiation tool: Principles, model, and specification. *ACM Transactions on Mathematical Software (TOMS)*, 39(3):20, 2013.

[54] S. Hazra and A. Jameson. One-shot pseudo-time method for aerodynamic shape optimization using the navier–stokes equations. *International Journal for Numerical Methods in Fluids*, 68(5):564–581, 2012.

[55] X. He, J. Li, C. A. Mader, A. Yildirim, and J. R. Martins. Robust aerodynamic shape optimization—from a circle to an airfoil. *Aerospace Science and Technology*, 87:48–61, 2019.

[56] R. M. Hicks and P. A. Henne. Wing design by numerical optimization. *Journal of Aircraft*, 15(7):407–412, 1978.

[57] M. Hojjat. *Node-based parameterization for shape optimal design*. PhD thesis, Technische Universität München, 2015.

[58] M. Hojjat, E. Stavropoulou, and K.-U. Bletzinger. The vertex morphing method for node-based shape optimization. *Computer Methods in Applied Mechanics and Engineering*, 268:494–513, 2014.

[59] J. Hradil. *Adaptive Parameterisation for Aerodynamic Shape Optimisation in Aeronautical Applications*. PhD thesis, Brno University of Technology, 2015.

[60] A. Hubeli and M. H. Gross. A survey of surface representations for geometric modeling. *CS technical report*, 335, 2000.

[61] A. Jameson. Optimum aerodynamic design using CFD and control theory. In *12th Computational Fluid Dynamics Conference*, page 1729, 1995.

[62] A. Jameson. Aerodynamic shape optimization using the adjoint method. *Lectures at the Von Karman Institute, Brussels*, 2003.

[63] A. Jameson and S. Kim. Reduction of the adjoint gradient formula in the continuous limit. In *41st Aerospace Sciences Meeting and Exhibit*, page 40, 2003.

[64] A. Jaworski and J.-D. Müller. Toward modular multigrid design optimisation. In *Advances in Automatic Differentiation*, pages 281–291. Springer, 2008.

[65] R. Jesudasan, X. Zhang, and J.-D. Mueller. Adjoint optimisation of internal turbine cooling channel using nurbs-based automatic and adaptive

parametrisation method. In *ASME 2017 Gas Turbine India Conference*, pages V001T02A009–V001T02A009. American Society of Mechanical Engineers, 2017.

[66] M. Khurana, H. Winarto, and A. Sinha. Airfoil geometry parameterization through shape optimizer and computational fluid dynamics. In *46th AIAA Aerospace Sciences Meeting and Exhibit*, page 295, 2008.

[67] H.-J. Kim, S. Koc, and K. Nakahashi. Surface modification method for aerodynamic design optimization. *AIAA journal*, 43(4):727–740, 2005.

[68] J. Kröger and T. Rung. Cad-free hydrodynamic optimisation using consistent kernel-based sensitivity filtering. *Ship Technology Research*, 62(3):111–130, 2015.

[69] B. M. Kulfan. Universal parametric geometry representation method. *Journal of aircraft*, 45(1):142–158, 2008.

[70] W. Li and S. Padula. Using high resolution design spaces for aerodynamic shape optimization under uncertainty. Technical Report NASA/TP-2004-213003, L-18355, NASA Langley Research Center; Hampton, VA, United States, 2004.

[71] J. Luo and E. H. Razinsky. Analysis of turbulent flow in 180 deg turning ducts with and without guide vanes. *Journal of Turbomachinery*, 131(2):021011, 2009.

[72] J. Luo, J. Xiong, F. Liu, and I. McBean. Three-dimensional aerodynamic design optimization of a turbine blade by using an adjoint method. *Journal of Turbomachinery*, 133(1):011026, 2011.

[73] T. Lyche and K. Mørken. Knot removal for parametric b-spline curves and surfaces. *Computer Aided Geometric Design*, 4(3):217–230, 1987.

[74] Z. Lyu, G. K. Kenway, and J. R. Martins. Aerodynamic shape optimization investigations of the common research model wing benchmark. *AIAA Journal*, 53(4):968–985, 2014.

[75] Z. Lyu, G. K. Kenway, C. Paige, and J. R. Martins. Automatic differentiation adjoint of the reynolds-averaged navier-stokes equations with a turbulence model. In *21st AIAA Computational Fluid Dynamics Conference*, page 2581, 2013.

[76] M. J. Martín Burgos. *NURBS-Based Geometry Parameterization for Aerodynamic Shape Optimization*. PhD thesis, Telecomunicacion, 2015.

[77] M. Martinelli and F. Beux. Multi-level gradient-based methods and parameterisation in aerodynamic shape design. *European Journal of Computational Mechanics/Revue Européenne de Mécanique Numérique*, 17(1-2):169–197, 2008.

[78] J. R. Martins and J. T. Hwang. Review and unification of methods for computing derivatives of multidisciplinary computational models. *AIAA journal*, 51(11):2582–2599, 2013.

[79] J. R. Martins, P. Sturdza, and J. J. Alonso. The complex-step derivative approximation. *ACM Transactions on Mathematical Software (TOMS)*, 29(3):245–262, 2003.

[80] D. Masters, D. Poole, N. Taylor, T. Rendall, and C. Allen. Influence of shape parameterization on a benchmark aerodynamic optimization problem. *Journal of Aircraft*, 54(6):2242–2256, 2017.

[81] D. A. Masters. *Evaluation of parameterisation methods and development of hierarchical subdivision techniques for aerodynamic optimisation*. PhD thesis, University of Bristol, 2017.

[82] D. A. Masters, N. J. Taylor, T. Rendall, and C. B. Allen. *Three-Dimensional Subdivision Parameterisation for Aerodynamic Shape Optimisation*.

[83] D. A. Masters, N. J. Taylor, T. Rendall, and C. B. Allen. A locally adaptive subdivision parameterisation scheme for aerodynamic shape optimisation. In *34th AIAA Applied Aerodynamics Conference*, page 3866, 2016.

[84] D. A. Masters, N. J. Taylor, T. Rendall, and C. B. Allen. Progressive subdivision curves for aerodynamic shape optimisation. In *54th AIAA Aerospace Sciences Meeting*, page 0559, 2016.

[85] M. McHale, J. Friedman, and J. Karian. Standard for verification and validation in computational fluid dynamics and heat transfer. *The American Society of Mechanical Engineers, ASME V&V*, pages 20–2009, 2009.

[86] S. Menzel, M. Olhofer, and B. Sendhoff. Direct manipulation of free form deformation in evolutionary design optimisation. In *Parallel Problem Solving from Nature-PPSN IX*, pages 352–361. Springer, 2006.

[87] B. Mohammadi and O. Pironneau. Shape optimization in fluid mechanics. *Annu. Rev. Fluid Mech.*, 36:255–279, 2004.

[88] A. L. Moigne and N. Qin. Variable-fidelity aerodynamic optimization for turbulent flows using a discrete adjoint formulation. *AIAA journal*, 42(7):1281–1292, 2004.

[89] A. Morris, C. Allen, and T. Rendall. Cfd-based optimization of aerofoils using radial basis functions for domain element parameterization and mesh deformation. *International Journal for Numerical Methods in Fluids*, 58(8):827–860, 2008.

[90] A. Morris, C. Allen, and T. Rendall. High-fidelity aerodynamic shape optimization of modern transport wing using efficient hierarchical parameterization. *International journal for numerical methods in fluids*, 63(3):297–312, 2010.

[91] A. Morris, C. Allen, and T. S. Rendall. Domain-element method for aerodynamic shape optimization applied to modern transport wing. *AIAA journal*, 47(7):1647–1659, 2009.

[92] J. Mueller, M. Banovic, S. Auriemma, P. Mohanamuraly, A. Walther, H. Legrand, et al. Nurbs-based and parametric-based shape optimisation with differentiated cad kernel. *Computer-Aided Design and Applications*, 2017.

[93] J.-D. Mueller, J. Hueckelheim, and O. Mykhaskiv. Stamps: a finite-volume solver framework for adjoint codes derived with source-transformation ad. In *2018 Multidisciplinary Analysis and Optimization Conference*, page 2928, 2018.

[94] J.-D. Müller, X. Zhang, S. Akbarzadeh, and Y. Wang. Geometric continuity constraints of automatically derived parametrisations in CAD-based shape optimisation. *International Journal of Computational Fluid Dynamics*, pages 1–17, 2019.

[95] O. Mykhaskiv, P. Mohanamuraly, J.-D. Mueller, S. Xu, and S. Timme. Cad-based shape optimisation of the nasa crm wing-body intersection using differentiated cad-kernel. In *35th AIAA Applied Aerodynamics Conference*, page 4080, 2017.

[96] H. Namgoong, P. Ireland, and C. Son. Optimisation of a 180° u-shaped bend shape for a turbine blade cooling passage leading to a pressure loss coefficient of approximately 0.6. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 230(8):1371–1384, 2016.

[97] U. Naumann. *The art of differentiating computer programs: an introduction to algorithmic differentiation*, volume 24. Siam, 2012.

[98] E. J. Nielsen and W. K. Anderson. Recent improvements in aerodynamic design optimization on unstructured meshes. *AIAA journal*, 40(6):1155–1163, 2002.

[99] J. Nocedal and S. Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[100] L. Osusky and D. Zingg. A novel aerodynamic shape optimization approach for three-dimensional turbulent flows. In *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, page 58, 2012.

[101] P. S. Patel, D. L. Marcum, and M. G. Remotigue. Automatic cad model topology generation. *International journal for numerical methods in fluids*, 52(8):823–841, 2006.

[102] O. Pfeifle. Aerodynamic investigation of the exit guide vane followed by curved duct. Master's thesis, Delft University.

[103] L. Piegl and W. Tiller. *The NURBS book*. Springer Science & Business Media, 2012.

[104] O. Pironneau. Optimal shape design for elliptic systems. In *System Modeling and Optimization*, pages 42–66. Springer, 1982.

[105] D. J. Poole, C. B. Allen, and T. C. Rendall. Metric-based mathematical derivation of efficient airfoil design variables. *AIAA Journal*, 53(5):1349–1361, 2015.

[106] O. M. Rejish Jesudasan and J.-D. Mueller. Surface mesh adaptation method for handling wing-fuselage intersections. In *EUROGEN-2017.*, 2017.

[107] G. Robinson and A. Keane. Concise orthogonal representation of supercritical airfoils. *Journal of Aircraft*, 38(3):580–583, 2001.

[108] T. T. Robinson, C. G. Armstrong, and H. S. Chua. Determining the parametric effectiveness of a cad model. *Engineering with Computers*, 29(1):111–126, Jan 2013.

[109] T. T. Robinson, C. G. Armstrong, H. S. Chua, C. Othmer, and T. Grahs. Optimizing parameterized cad geometries using sensitivities based on adjoint functions. *Computer-Aided Design and Applications*, 9(3):253–268, 2012.

[110] F. Salmoiraghi, A. Scardigli, H. Telib, and G. Rozza. Free-form deformation, mesh morphing and reduced-order methods: enablers for efficient aerodynamic shape optimisation. *International Journal of Computational Fluid Dynamics*, 32(4-5):233–247, 2018.

[111] J. Samareh. Aerodynamic shape optimization based on free-form deformation. In *10th AIAA/ISSMO multidisciplinary analysis and optimization conference*, page 4630, 2004.

[112] J. A. Samareh. Survey of shape parameterization techniques for high-fidelity multidisciplinary shape optimization. *AIAA journal*, 39(5):877–884, 2001.

[113] I. Sanchez Torreguitart, T. Verstraete, and L. Mueller. Cad kernel and grid generation algorithmic differentiation for turbomachinery adjoint optimization. In *Proceedings of the 7th European Congress on Computational Methods in Applied Sciences and Engineering, Crete Island, Greece*, pages 5–10, 2016.

[114] S. Schmidt, C. Ilic, N. Gauger, and V. Schulz. Shape gradients and their smoothness for practical aerodynamic design optimization. *Optim. Eng.(20 08) Preprint-Number SPP1253-10-03*, 2008.

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

[116] Y.-D. Seo, H.-J. Kim, and S.-K. Youn. Shape optimization and its extension to topological design based on isogeometric analysis. *International Journal of Solids and Structures*, 47(11-12):1618–1640, 2010.

[117] P. A. Sherar, C. P. Thompson, B. Xu, and B. Zhong. An optimization method based on b-spline shape functions & the knot insertion algorithm. In *World congress on engineering*, pages 862–866. Citeseer, 2007.

[118] D. Sieger, S. Menzel, and M. Botsch. A comprehensive comparison of shape deformation methods in evolutionary design optimization. In *Proceedings of the International Conference on Engineering Optimization*. Citeseer, 2012.

[119] D. Sieger, S. Menzel, and M. Botsch. Constrained space deformation for design optimization. *Procedia Engineering*, 82:114–126, 2014.

[120] O. Sigmund and K. Maute. Topology optimization approaches. *Structural and Multidisciplinary Optimization*, 48(6):1031–1055, 2013.

[121] O. Sigmund and J. Petersson. Numerical instabilities in topology optimization: a survey on procedures dealing with checkerboards, mesh-dependencies and local minima. *Structural optimization*, 16(1):68–75, 1998.

[122] S. Skinner and H. Zare-Behtash. State-of-the-art in aerodynamic shape optimisation methods. *Applied Soft Computing*, 2017.

[123] A. Sóbester and A. I. Forrester. *Aircraft aerodynamic design: geometry and optimization*. John Wiley & Sons, 2014.

[124] H. Sobieczky. Parametric airfoils and wings. In *Recent development of aerodynamic design methodologies*, pages 71–87. Springer, 1999.

[125] W. Squire and G. Trapp. Using complex variables to estimate derivatives of real functions. *SIAM-Review*, 10(1):110–112, 1998.

[126] A. Stück and T. Rung. Adjoint rans with filtered shape derivatives for hydrodynamic optimisation. *Computers & Fluids*, 47(1):22–32, 2011.

[127] A. Stueck and T. Rung. Filtered gradients for adjoint-based shape optimisation. In *20th AIAA Computational Fluid Dynamics Conference*, page 3072, 2011.

[128] E. S. Tashnizi, A. A. Taheri, and M. H. Hekmat. Investigation of the adjoint method in aerodynamic optimization using various shape parameterization techniques. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 32(2):176–186, 2010.

[129] S. Thomas, O. Carsten, et al. Adjoint optimization for vehicle external aerodynamics. *International Journal of Automotive Engineering*, 7(1):1–7, 2016.

[130] D. Toal, N. Bressloff, and A. Keane. Geometric filtration using pod for aerodynamic design optimization. In *26th AIAA Applied Aerodynamics Conference*, page 6584, 2008.

[131] J. Utke, U. Naumann, M. Fagan, N. Tallent, M. Strout, P. Heimbach, C. Hill, and C. Wunsch. Openad/f: A modular open-source tool for automatic differentiation of fortran codes. *ACM Transactions on Mathematical Software (TOMS)*, 34(4):18, 2008.

[132] A. J. t. J. C. VASSBERG. Studies of alternative numerical optimization methods applied to the brachistochrone problem. 2000.

[133] J. Vassberg and A. Jameson. Influence of shape parameterization on aerodynamic shape optimization. *__, April*, 2014.

[134] T. Verstraete. Cado: a computer aided design and optimization tool for turbomachinery applications. In *2nd Int. Conf. on Engineering Optimization, Lisbon, Portugal, September*, pages 6–9, 2010.

[135] T. Verstraete. The VKI U-Bend optimization test case. Technical report, The von Karman Institute for Fluid Dynamics, 2016.

[136] T. Verstraete, F. Coletti, J. Bulle, T. Vanderwielen, and T. Arts. Optimization of a u-bend for minimal pressure loss in internal cooling channels part i: Numerical method. *Journal of Turbomachinery*, 135(5):051015, 2013.

[137] B. Walther and S. Nadarajah. Constrained adjoint-based aerodynamic shape optimization of a single-stage transonic compressor. *Journal of turbomachinery*, 135(2):021017, 2013.

[138] D. Whitley. An overview of evolutionary algorithms: practical issues and common pitfalls. *Information and software technology*, 43(14):817–831, 2001.

[139] M. Wintzer and I. Ordaz. Under-track cfd-based shape optimization for a low-boom demonstrator concept. In *33rd AIAA Applied Aerodynamics Conference*, page 2260, 2015.

[140] S. Wright and J. Nocedal. Numerical optimization. *Springer Science*, 35(67-68):7, 1999.

[141] H.-Y. Wu, S. Yang, F. Liu, and H.-M. Tsai. Comparisons of three geometric representations of airfoils for aerodynamic optimization. In *16th AIAA computational fluid dynamics conference*, page 4095, 2003.

[142] S. Xu, W. Jahn, and J.-D. Müller. Cad-based shape optimisation with cfd using a discrete adjoint. *International Journal for Numerical Methods in Fluids*, 74(3):153–168, 2014.

[143] S. Xu, D. Radford, M. Meyer, and J.-D. Müller. Cad-based adjoint shape optimisation of a one-stage turbine with geometric constraints. In *ASME Turbo Expo 2015: Turbine Technical Conference and Exposition*, pages V02CT45A006–V02CT45A006. American Society of Mechanical Engineers, 2015.

[144] S. Xu, D. Radford, M. Meyer, and J.-D. Müller. Stabilisation of discrete steady adjoint solvers. *Journal of Computational Physics*, 299:175–195, 2015.

[145] W. Yamazaki, S. Mouton, and G. Carrier. Geometry parameterization and computational mesh deformation by physics-based direct manipulation approaches. *AIAA journal*, 48(8):1817–1832, 2010.

[146] G. Yu, J.-D. Müller, D. Jones, and F. Christakopoulos. Cad-based shape optimisation using adjoint sensitivities. *Computers & Fluids*, 46(1):512–516, 2011.

[147] Y. Yu, Z. Lyu, Z. Xu, and J. R. Martins. On the influence of optimization algorithm and initial design on wing aerodynamic shape optimization. *Aerospace Science and Technology*, 75:183–199, 2018.

[148] Q. Zhang, J.-D. Lee, and J.-H. Wendisch. An adjoint-based optimization method for helicopter fuselage backdoor geometry. 2010.

[149] X. Zhang. *CAD-based geometry parameterisation for shape optimisation using Non-uniform Rational B-splines*. PhD thesis, Queen Mary University of London, 2018.

[150] X. Zhang, R. Jesudasan, and J.-D. Müller. Adjoint-based aerodynamic optimisation of wing shape using non-uniform rational b-splines. In *Evolutionary and Deterministic Methods for Design Optimization and Control With Applications to Industrial and Societal Problems*, pages 143–158. Springer, 2019.

[151] F. Zhu. *Geometric parameterisation and aerodynamic shape optimisation*. PhD thesis, University of Sheffield, 2014.