Shape Sensitivity Analysis for Coupled Fluid-Solid Interaction Problems

Koorosh Gobal

December 9, 2015

Contents

List of Figures

List of Tables

Abstract

In this paper, a robust continuum sensitivity formulation for the shape sensitivity analysis of weakly coupled aero-structural systems is derived. In this method, the solid boundaries are modelled using the immersed boundary approach. This simplifies the grid generation for the complex and deforming geometries since the computational mesh does not need to conform to the boundaries. The sensitivity analysis consists of differentiating the continuum form of the governing equations where the effect of the solid boundaries are modelled as additional forcing terms in these equations. By differentiating the governing equations, it is possible to reuse the operators utilized for solving the governing equations. Therefore, there is no need to develop new solvers for the solution of sensitivity response. This methodology is applied to different demonstration problems including flow over a cylinder and a simplified aeroelastic model of a wing. The wing structure is modelled as a beam with the lifting surface mounted at the tip where the load is transfered to the structure through the mounting point. The sensitivity results with this approach compares well with the complex step method results. Moreover, it is shown that the methodology is capable of handing complex shapes with high Reynolds numbers.

Chapter 1

Introduction

1.1 Motivation

Fluid-structure interaction (FSI) problems play important role in many scientific and engineering fields, such as automotive, aerospace, and biomedical industry. Despite the wide application, a comprehensive study of FSI systems still remains a challenge due to their strong nonlinearity and multidisciplinary nature. For most FSI problems, analytical solutions to are not available, and physical experiments are limited in scope. Therefore, to get more insight in the physics involved in the complex interaction between fluids and solids, numerical simulations are used. The numerical solutions are conducted based on Computational Fluid Dynamics (CFD) models for the flow and Finite Element Analysis (FEA) for the structural response. Nevertheless, the prohibitive amount of computations has been one of the major issues in the design optimization of such coupled multidisciplinary systems. The other bottle neck is generating an appropriate computational domain that represents the fluid and solid regions. The effort and time required to take a geometry from a CAD package, clean up the model, and generate a mesh is usually a large portion of the overall human time required for the simulation. This cannot be automated for complex and moving domains. The Immersed Boundary (IB) method, reduces the amount of time needed for the fluid flow simulations and provides fast results by directly addressing the challenges associated with this issue.

Due to the large amount of computations involved in the FSI simulation, the gradient based methods are the best candidates for design optimization of such problems. Sensitivity analysis is the integral part of gradient based methods. Although there are analytical techniques for efficient and accurate sensitivity calculation, they have not yet implemented in commercial CFD packages. Therefore, most gradient optimization techniques relay on finite difference method for sensitivity calculation when solving FSI problem that are prone to errors.

The motivation for the research proposed in this document is in two areas. First, we want to have sensitivity analysis capabilities that can treat the solvers as black-box. This means that we can solve both the governing equations and sensitivity response using the same code. Second, a robust analysis technique for the coupled FSI system based on IB method is formulated. The current approach of IB is not suited for the sensitivity analysis due to the discontinuities in its formulation. This will be explained in more details in the following Chapters.

1.2 Literature Review

1.2.1 Sensitivity Analysis

Sensitivity analysis consists of computing the derivatives of solution of the governing equations, i.e. displacement, velocity, or pressure, with respect to one of several independent design variables, i.e. shape of boundaries or size of elements. There various applications for sensitivity information such as improving the accuracy of surrogate models as in gradient enhanced Kriging [1] or uncertainty quantification [2]. However, our main motivation is the use of this information in gradient-based optimization. The calculation of gradients is often the most expensive step in the optimization cycle therefore, using efficient methods for accurate calculation of sensitivities are vital to the optimization process. As shown in Figure 1.1, methods for sensitivity calculation can be put into three methods: i) numerical, ii) analytical, and iii) automatic differentiation.

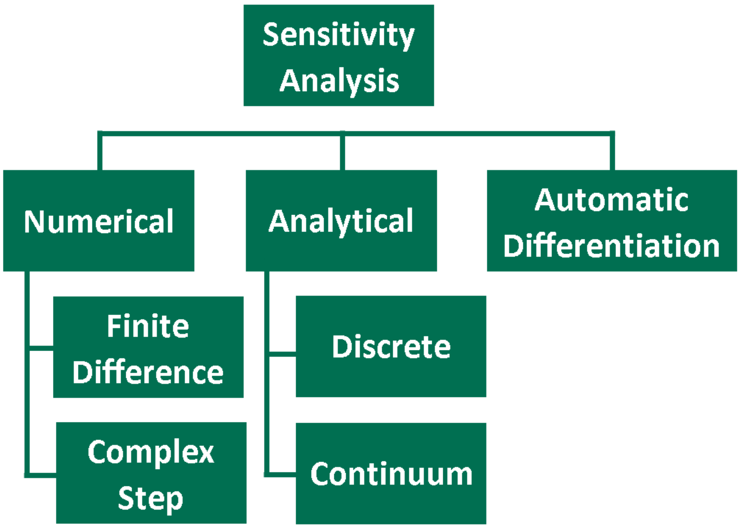


Figure 1.1: Sensitivity calculation techniques.

The Finite Difference (FD) method is probably the easiest method to implement for calculating the sensitivity of a variable. The fact that they can be implemented even when a given computational model is a black box makes most gradient based optimzation algorithms perform finite differences by default when the used does not provide the required gradients. However, the computational cost associated with finite difference for large systems can become very large. For a system with n number of design variables, the analysis needs to be done $n + 1$ time to calculate the design sensitivities. Furthermore, to ensure the accuracy, convergence study needs to be done for selecting the appropriate step size for finite difference. The inaccuracy of finite differencing could result in convergence difficulties and inaccurate optimum results. On the other hand, Complex Step (CS) method avoids the loss of precision in finite differences approximation of sensitivities by employing complex arithmetic [3]. The complex step derivative is calculated as shown in Equation (1.1).

$$\mathcal{F}'(u; b) = \frac{\mathrm{Im}\left[\mathcal{F}(u; b + ih)\right]}{h} \tag{1.1}$$

This means that we perturb the design variable by an imaginary value of ih and then look at the imaginary portion of the resulting response. Using the complex step method, we can choose a small step size for h without loosing accuracy. However, many commercial packages such as ANSYS or Nastran cannot handle complex arithmetic which makes the implementation of complex step method infeasible. Moreover, the high cost of finite difference is still associated with the complex method as well.

Automatic differentiation (AD) is based on the systematic application of the differentiation to computer programs [4]. In the AD approach, the chain rule of differentiation is applied to every line in the program. This assumes that the computer program consists of a sequence of explicit functions that act successively on some variables. Therefore, by differentiating each of these functions and applying the chain rule, it is possible to calculate the sensitivities.

There has been many research on utilizing AD for optimization. Bischof et al. used AD for calculating the sensitivities using a CFD solver. They used ADIFOR for differentiating the source code of their CFD code (TLNS3D) which later used for calculating the sensitivity of a transonic flow to change in the boundary conditions. Hascout et al., also used AD for a sonic boom reduction under a supersonic aircraft. However, in all of these works, it is needed to have access to the source code and modify the solver extensively to calculate the sensitivities. This make the use of this method for general purpose codes infeasible since the source code is usually not available.

The short comings of numerical and AD techniques, fuels the research for more sophisticated methods for sensitivity calculation which are generally known as analytical methods. Formulation of the analytical sensitivities requires derivation of analytic sensitivity equations. These are obtained by differentiating the governing equations with respect to design variables such as shape of the boundaries. Analytical methods can be further categorized based on how the sensitivity equations derived and solved. Typically the continuum equations are solved using an approximate method which discretizes the governing equations. The Discrete Sensitivity Analysis (DSA) technique, differentiates the discretized governing equations with respect to design variables to get the analytical sensitivity equations [5]. This system of equations is later solved for analytical sensitivities.

The discrete sensitivity analysis has been historically the method of choice to calculate the high-accurate sensitivity when the details of analysis are available [6]. This method has been adopted by the structural optimization community, and been applied to fluid-solid interaction problems as well. Reuther et al., used the discrete method for aerodynamic shape optimization of a complex aircraft configuration [7]. They used Euler flow as the aerodynamics theory where they optimized different configurations for transonic and supersonic regimes. However, they only focused on the flow. Martins et al., developed an adjoint method for sensitivity analysis for an aero-structural aircraft design framework where the sensitivities were computed using a coupled adjoint approach. The framework was used on a supersonic business jet to calculate the sensitivity of drag with respect to the shape (OML) of the aircraft. In their work, the discretization details of the solver needs to be known to calculate the sensitivities. As a matter of fact, source code modification is essential in all the other related research in the area as well [8, 9, 10].

As mentioned before, sensitivity calculation is the essential part of gradient based optimization. However, many general purpose solvers do not have this capability. This is even more prominent in the fluid mechanics community where none of the available commercial software packages such as ANASYS Fluent or CFX has analytical sensitivity analysis capabilities. In optimization loop, they relay on finite difference as a method for sensitivity calculation. As mentioned in the previous paragraph, for the discrete sensitivity methods, intimate knowledge and access to the source codes of these software is needed to calculate the sensitivities. However, the source code is usually inaccessible and very complex. Therefore, there is a great interest in sensitivity calculation techniques which require minimum knowledge of the analysis code. This can be achieved using the sensitivity formulation that operate on the governing equations before they are discretized. These method are commonly known as Continuum Sensitivity Methods [11].

The Continuum Sensitivity Analysis (CSA), involves solving a set of partial differential equations named the continuum sensitivity equations (CSEs) to get the analytical sensitivities. When deriving the CSEs, the governing equations can either be differentiated in local or total form. The difference between local and total differentiation depends on the Lagrangian or Eulerian representation of the governing equations. In the general case of continuum mechanics, displacement and velocity are vector valued functions. I any application, we have the choice of writing these vectors as functions of the position of material particles before deformation.

This is called the Lagrangian description of motion and is really helpful for visualizing the deformations. This technique is mostly adopted in solid mechanics where we track the material points as the deformations are usually assumed to be small. However, in the fluid flow problems, since it is generally hard to identify a reference configuration and the deformations are large, it is preferable to write the displacement and velocities as functions of the deformed position of the particles. These quantities are now defined for a particular point in space that does not move with the particles. This is called the Eulerian description of motion. As CSA was matured over the year, the computational fluids community adopted the local CSEs, since its formulation is consistent with the governing equations [12, 13]. The structural optimization community adopted the total formulation for the sensitivities since its formulation was consistent with the Lagrangian formulation of structural mechanics. Nevertheless, the total and local formulations for the sensitivities can be converted to each other. This will be explained in more details in next chapter.

Overally, the sensitivity analysis in general is more matured is the concept of structural optimization than for problems of fluid dynamics. Nevertheless, neither of these disciplines do not typically employ CSA for the sensitivity calculation. Aurora and Haug [14], followed by Dems and Mroz [15], were among the first to develop the concept of CSA for structural optimization. They modeled the sensitivities as functionals therefore they were able to convert the sensitivity integrals over the entire domain to the integrals over the boundaries. Although using the approach it is possible to reduce the cost of the simulation, accurate values of functionals are required at the boundaries. This is not always achievable especially for finite element analysis where the solution accuracy drops near the boundaries. The lack of applicability of CSA in structural optimization is due to the complicated definition of the boundary conditions and maturity of discrete methods for the sensitivity calculation.

The first application of CSA in optimization problems for fluid dynamics is the work by Borggaard and Burns [16] for shape sensitivity analysis of inviscid supersonic flows over rigid bodies. Stanley and Stewart [17] applied CSA in a fluid mechanics discipline with a goal for aerodynamic design. Pelletier and Etienne have applied CSA to numerous fluid-structure interaction (FSI) problems [18] focused mainly on sensitivities of fluid flow parameters near the structure. Liu and Canfield have employed CSA for shape optimization of nonlinear structures subject to an aeroelastic gust response [19]. They used the finite element method to solve the potential flow around an airfoil and applied CSA to find the airfoil pressure coefficient sensitivity with respect to the maximum camber.

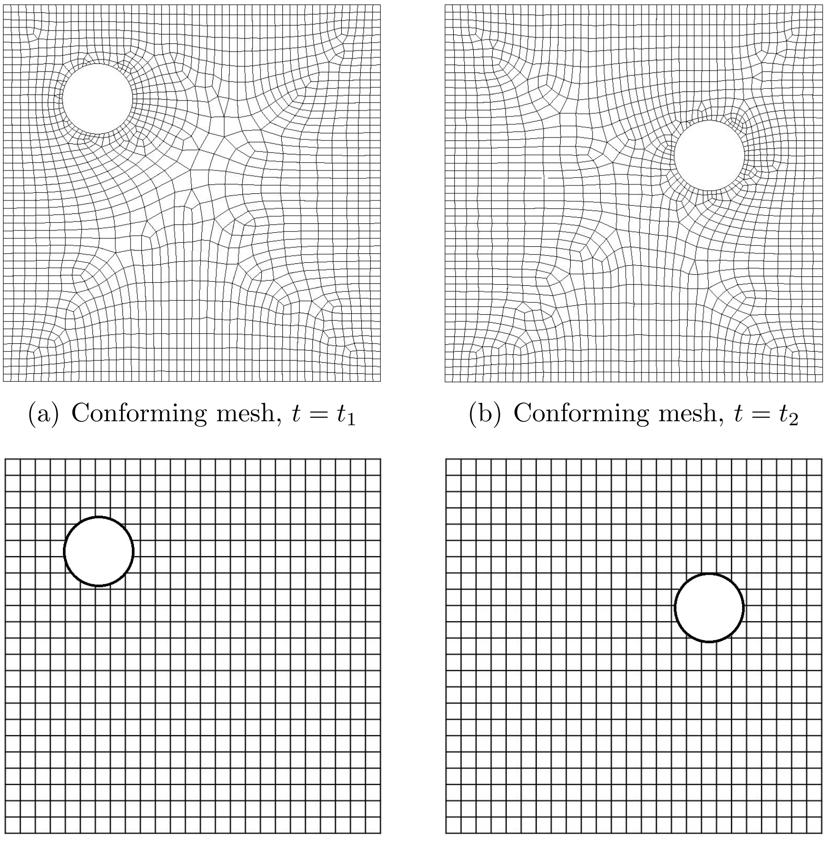
In almost all of the research done on sensitivity calculation of flow field to shape design variable, body conformal grids were used. The conforming mesh methods consider the interface conditions as physical boundary conditions, which treat the interface location as part of the solution and requires meshes that conform to the interface. Using this approach it is possible to represent the solid boundary shape with good accuracy. However, due to the coupling of fluid mesh topology and solid boundary shape, with the movement and/or deformation of the solid structure, re-meshing (or mesh-updating) is needed. Although conforming mesh methods have been widely used in many FSI problems, they are cumbersome, if not impossible, to apply to problems with large deformations [20]. The other shortcoming of body-conformal grids, is the effect of mesh deformation on the sensitivity analysis. Since the computational domain is affected by change in the shape of the boundaries, it is required to calculate the mesh sensitivities as well [21]. This adds to the computational cost of calculating sensitivities.

The shortcoming of a robust grid generation and the additional cost of calculating mesh sensitivities motivated an important research effort to develop a method that does not require fluid domain mesh modification for the optimization iterations. On of the possible candidates to achieve this goal, is the use of Immersed Boundary (IB) methods to decouple the fluid mesh from the shape of solid domain.

1.2.2 Immersed Boundary Method

Traditionally the simulation methods for flow over complex bodies are based on a body-fitted multi-block or unstructured grid methods. However, in the last decade another class of techniques, called immersed boundary methods, have been introduced to model the flow around solid boundaries. The term immersed boundary is first used by Peskin [22] to simulate the blood flow through heart valves. What distinguishes this method from the other methods of representing the solid boundaries is that the flow is solved on the Cartesian grid that does not conform to the solid boundaries. Therefore, the mesh generation is greatly simplified and in case of moving/deforming boundaries, there is no need to update the mesh during the simulation. A separate formulation is used to impose the effect of the boundaries on the solution of the equations.

Consider the simulation of flow past a circular cylinder as shown in Figure 1.2. Generating structured or unstructured grids is achieved in two steps. First, a surface grid covering the boundaries is generated. This is then used as a boundary condition to generate a grid in the volume occupied by the fluid. The differential form of the governing equations is then transformed to a curvilinear coordinate system aligned with the grid lines [23]. If a finite-volume technique is employed, then the integral form of the governing equations is discretized and the geometrical information regarding the grid is incorporated directly into the discretization. If an unstructured grid is employed, then either a finite-volume or a finite-element methodology can be used.



(a) Conforming mesh, $t = t_1$

(b) Conforming mesh, $t = t_2$

(c) Nonconforming mesh, $t = t_1$

(d) Nonconforming mesh, $t = t_2$

Figure 1.2: Example of conforming and nonconforming meshes.

Non-body conformal Cartesian grid can also be utilized for this simulation as shown in Figure 1.2. In this approach the IB would still be represented through some means such as a surface grid, but the Cartesian volume grid would be generated with no regard to this surface grid. Thus, the solid boundary would cut through this Cartesian volume grid. Because the grid does not conform to the solid boundary, incorporating the boundary conditions would require modifying the equations in the vicinity of the boundary. Assuming that such a procedure is available, the governing equations would then be discretized using a finite-difference, finite-volume, or a finite-element technique without resorting to coordinate transformation or complex discretization operators.

Depending on how the effect of solid boundaries are imposed, IB methods can be divided into four categories: i) Continuous forcing, ii) discrete forcing, iii) penalization, and iv) cut-cell method.

Continuous forcing method

This is the originaly used by Peskin [22] and later developed by others researchers [24, 25, 26]. In this approach, the boundary configuration is described by a curve $x(s,t)$ (Lagrangian nodes) where the location of each point on this boundary is governed by its equations of motion. The forces that the curve $x(s,t)$ exerts on the fluid is calculated buy the constitutive law of the curve $x(s,t)$ that relates the displacements to stress values. These stress values are transferred to the Navier-Stokes (NS) (Eulerian nodes) for the fluid by means of a Dirac delta function. Practical implimentation of this method resets in representing the Dirac delta function as a discrete function that has the same properties. This method is applied to variety of problems, including Cardiac blood flow [27], animal locomotion [28], multiphase flows [29], and particle sedimentation [30].

Peskin's method is well suited for the elstic boundaries. For stiff boundaries, the constitutive laws of the solid boundaries will result in instabilities in the solution of governing equations [31]. The virtual boundary method of Goldstein [32] enables us to handle these rigid boundaries. The main idea of this approach is the same as Peskin's method where the solid boundary is treated as a virtually existing surface embedded in the fluid. The force on this surface is calculated by the requirement that the fluid velocity should satisfy the no-slip condition. Since this body force is not known a priori, it is calculated in a feedback way.

Discrete forcing method

This discrete formulation of IB method was first introduced by Mohd-Yusof [33] for addressing the time penalties associated with the simulation involving moving boundaries. The main idea of this technique is same as the virtual boundary method where a force term is added to the momentum equations to represent the solid boundary. However, in the current approach the momentum equation manipulation is done in the discrete manner. In the work of Mohd-Yusof [33] and Verzicco et al. [34] this is done by modifying the discretization stencil in the vicinity of the boundary curve so that the velocity values on this boundary satisfies a pre-defined condition. The major advantage of this approach is the lack of user specified parameters however, its implementation depends strongly on the discretization approach used for the analysis. Therefore, it cannot easily implemented in commercial codes. This implementation of discrete forcing method has be applied to many different problems such as turbulent flow inside an internal combustion engine [34], flow past 3D bluff bodies [35], and cylindrical stirred tank [36].

Penalization method

The penalization method is based on the Brinkman equation that describes the flow of a fluid through a porous medium. The Brinkman equation is analogous to Fourier's laws in heat conduction and Ohm's law in the field of electrical engineering [37]. This approach was first proposed by Arquis and Caltagirone where they imposed the boundary conditions by adding the penalization terms to the momentum equations [38]. The main idea of this approach is to model the solid obstacles as a porous medium with the porosity of $\mathcal{K}$. The porosity is a measure of the void spaces in a material. Therefore, by selecting low values for porosity, the porous domain acts as a solid wall. It has be argued that the solution of the penalized incompressible NS equations converges to the exact solution as the porosity approaches zero [39] however in the practical implementation, extreme low values for porosity will result in systems with a very large stiffness and numerically unstable. To apply the porosity within the solid domain, a Heaviside function is used by different researchers [40, 41].

Cut-cell method

In the cut-cell methods, the grid cells are cut by the solid boundary and reshaped so that they form a boundary-conforming, unstructured gird. This is shown in Figure 1.3.

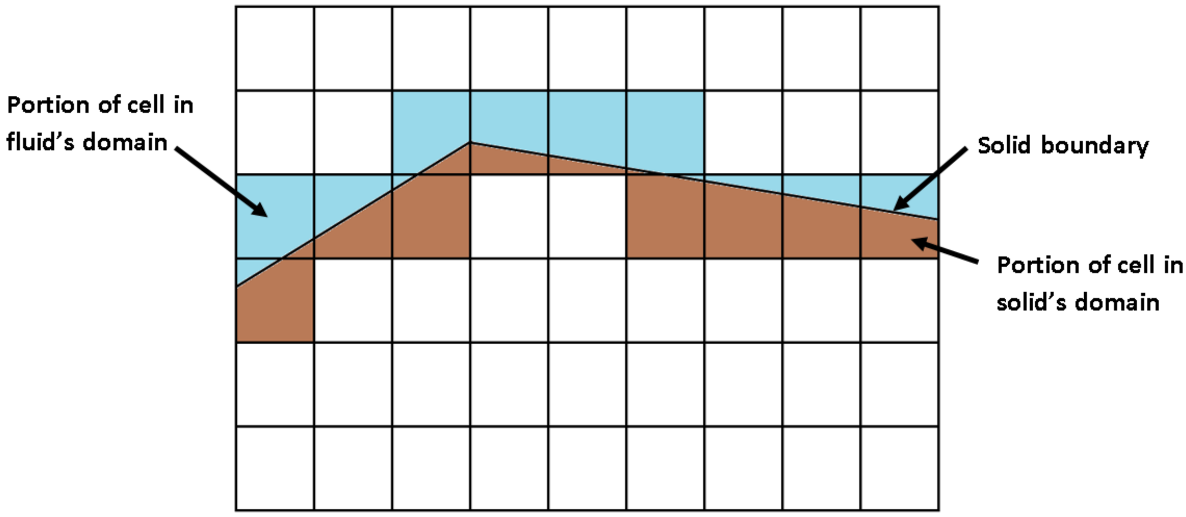


Figure 1.3: Modified mesh near the solid boundary for cut cell method.

The cut-cell method was first introduced Clarke [42] for inviscid flows. This method have been applied to both collocated and staggered grids [43] however, most of the applications were focused on 2D flows [44, 45]. This is because of the many possibilities for the geometrical shape of the cut-cell, that makes the flux calculation and therefore numerical implementation extremely difficult.

1.2.3 Sensitivity analysis for IB method

The body of the research done in the immersed boundary community has been concentrated on the improvement of the method and resolving the stability issues. However, there has not been much effort in the sensitivity calculation of a IB formulation. Most of the sensitivity analysis research is developed for the penalization framework. Borrvall and Petersson applied the penalization method for topology optimization of fluids in Stokes flow [46]. They used the discrete sensitivity analysis to calculate the sensitivity of fluid pressure to the shape of the boundaries. They used this technique to optimize the shape of a diffuser and also a pipe bend. They verified their methodology for outflow problems as well, where they optimized the shape of solid domain to maintain the least possible pressure drop. They also had constraint on the volume of domain that can be penalized. Challis and Guest investigated the level set formulation for the topology optimization of fluids in Stokes flow [47]. They used the penalization technique to formulate their problem where they were able to accurately model the no-slip condition on the solid boundaries. They implemented the Topological sensitivities in their solver where they used power dissipation minimization to optimize the shape of a diffuser and a connecting pipe in 3D.

The penalization method is probably the easiest of the IB methods to implement. This is because it does not have a free parameter such a virtual boundary method and does not depend on the discretization such as discrete forcing methods. Moreover, no interpolation is needed for applying penalization method to a computational domain. In the penalization method, the fluid and solid domain are differentiated based a scalar function (porosity) which is very similar to the density based approaches in the topology optimization community [48]. This is probably the biggest reason for researchers to use penalization technique for shape optimization using IB method since the available techniques from topology optimization community can be reused [49, 50]. Similar to topology optimization, the shortcoming of penalization technique is in its accuracy for representing the boundaries. Since the nodes are assigned the porosity values, it is extremly difficult if not impossible to control the exact location of immersed boundary if it does not coincide with the computational nodes. Moreover, the current demonstration problems for the penalization technique are only applicable to low Reynolds numbers flow.

1.3 Research Contribution

Due to the shortcomings of the penalization method, in this research we develop a senstivitiy analysis based on continuous forcing IB method. Since the sensitivity analysis is done using the continuum sensitivity analysis, only continuum formulations of IB are possible in this research. The continuum sensitivity formulation enables us to reuse the differential operators that we already developed for solving the governing equations. The IB method, removes the need to mesh deformation so that same computational mesh can be reused throughout the analysis. Moreover, since the mesh nodes do not move, the local and total form of the sensitivities are the same. Therefore, no extra computation is needed to converting the local to total form of the sensitivities.

Chapter 2

Design Sensitivity Analysis

In this chapter, the concept behind discrete and continuum sensitivity formulation is discussed and the general approach for deriving the sensitivity equations is presented. The two sensitivity analysis techniques are applied to a heat transfer benchmark problem where the sensitivity of response to the shape of the domain is calculated. This problem is also used in the next chapter for implementation of different immersed boundary methods. The difference between local and total formulation of the sensitivity response is discussed and finally the independence of continuum sensitivity formulation to discretization method is proven. This enables use to reuse the solver of governing equation to calculate the sensitivity response. This is typically not possible for discrete sensitivity formulation.

2.1 General formulation

The general computational domain is defined as shown in Figure **??**. The response variable on this domain can be from the fluid, i.e. pressure or velocity, or the solid, i.e. displacements. Nevertheless, the response is calculated using a governing equation subject to boundary conditions. In this work, the governing equation and boundary conditions are represented in the functional form as shown in Equation (2.1).

$$\mathbf{A}(\mathbf{u}, t; \mathbf{b}) = \mathbf{f}(\mathbf{x}, t; \mathbf{b}) \quad \text{on} \quad \Omega \tag{2.1a}$$

$$\mathbf{B}(\mathbf{u}, t; \mathbf{b}) = \mathbf{g}(\mathbf{x}, t; \mathbf{b}) \quad \text{on} \quad \Gamma \tag{2.1b}$$

where $\mathbf{A}$ is the governing equation such as Navier-Stokes equations or elastic equations and $\mathbf{B}$ is the boundary condition definition. $\mathbf{u}$ is the response variable such as displacement or pressure. t is time, $\mathbf{b}$ is the design variable such as shape or size, and $\mathbf{x}$ is the spatial coordinate. $\mathbf{f}$ and $\mathbf{g}$ is the value of governing equation and boundary condition. The sensitivity of response variable with respect to i-th design variable b_i, $\partial \mathbf{u}/\partial b_i$, can be calculate using some sensitivity analysis technique. This is done in the following sections.

The total sensitivity of response variable, $\mathbf{u}$ with respect to the i-th design variable is written as

$$\frac{D\mathbf{u}}{Db_i} = \frac{\partial \mathbf{u}}{\partial b_i} + \frac{\partial \mathbf{u}}{\partial \mathbf{x}} \cdot \frac{\partial \mathbf{x}}{\partial b_i} \tag{2.2}$$

The total derivative is known as material derivative in continuum mechanics [51]. This total sensitivity defines the change of response variable, $\mathbf{u}$, subjected to design variable and space dependent changes. The material derivative consists of the local derivative, $\partial \mathbf{u}/\partial b_i$, plus the convective term, $\partial \mathbf{u}/\partial \mathbf{x} \cdot \partial \mathbf{x}/\partial b_i$. The local derivative is the measure of change in the response variable due to change in the design parameter. Whereas, the convective term accounts for the movement of this point in space due to change in the design variable. This is specially applicable to shape sensitivity calculating where the change in design variable, will cause the material points to move [52]. The convective term consists of two separate gradients: i) $\partial \mathbf{u}/\partial \mathbf{x}$ which represents the spatial gradient of the response variable in the domain, and ii) $\partial \mathbf{x}/\partial b_i$ which defines the sensitivity of the spatial domain with respect to the change in design variable. The later defines how the computational nodes moves as the design variable changes. The response variable spatial gradient term can be calculated using the analysis results, using finite difference approach or derivative of shape functions in FEA formulation. Calculation of domain sensitivity, $\partial \mathbf{x}/\partial b_i$, requires more attention.

A common approach to calculate the domain sensitivity, is to use the same techniques used to deform the body-conformal mesh in a CFD simulation. These methods are usually based on representing the computational grid as a system of springs that connected to each other at the nodes. This system can be modeled and solved using structural analysis techniques, where the sensitivities can be easily added to its formulation. This is effectively a shape sensitivity analysis for the structural analyses [53]. This step can be removed from the analysis if the computational domain does not affected by design variable since $\partial x/\partial b$ is equal to zero for this case. This is removes the extra step of doing the structural shape sensitivity analysis for the mesh and also response variable gradient calculation. As mentioned in Chapter 1, by using the immersed boundary method the mesh definition is decoupled from the boundary shape. Therefore, domain sensitivity is equal to zero [**?**]. This is one of the reasons to use the immersed boundary calculation since it reduces the cost of the simulation. This is discussed in more details in Chapter **??**.

2.2 Benchmark case

To compare the discrete and continuum sensitivity analysis we chose the one-dimensional heat transfer analysis in a rod. The temperature in governing by the Laplace equation as shown in Equation (2.3).

$$\frac{\partial^2 T}{\partial x^2} = 0 \tag{2.3}$$

where T is the temperature and x is spatial variable. The boundary conditions are defined as constant temperatures at the two ends of the domain as shown in Figure 2.1. The length of domain is selected as L.

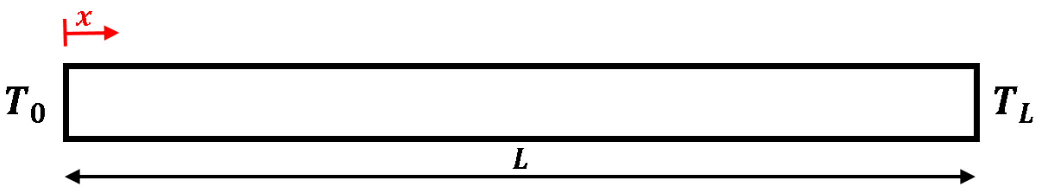


Figure 2.1: One dimensional domain with heat conduction.

The analytical solution of Equation (2.3) can be written as follows

$$T = \frac{T_L - T_0}{L} x + T_0 \tag{2.4}$$

The analytical sensitivity of the temperature with respect to beam's length can be calculated by differentiating Equation (2.4) with respect to L.

$$\frac{\partial T}{\partial L} = -\frac{T_L - T_0}{L^2} x \tag{2.5}$$

Since the analytical derivatives are known, we can compare it to the discrete and continuous sensitivity results.

2.3 Discrete sensitivity formulation

To formulate the discrete sensitivity equation, we start by discretizing the governing equation (2.3) using finite difference method. It should be noted that the finite difference is used for discretization of continuum governing equation and not sensitivity calculation. The design variable effects the shape of the domain which is the distance between the nodes in the discrete manner. Therefore, for the sake of sensitivity analysis it is required to keep the nodal distances in the discretized solution as well. We use 6 nodes to discretize the domain as shown in Figure 2.2.

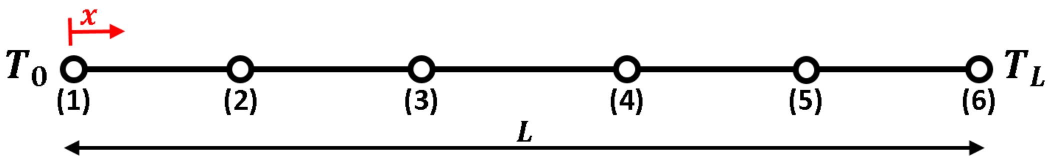


Figure 2.2: One dimensional computational domain for heat conduction.

To discretize Equation (2.3), the first step is to approximate the second derivative. This is done by writing the Taylor series expansion at at arbitrary location x_i.

$$T(x) = T(x_0) + \left.\frac{\partial T}{\partial x}\right|_{x_i}(x - x_i) + \left.\frac{\partial^2 T}{\partial x^2}\right|_{x_i}(x - x_i)^2 + \mathcal{H.O.T}$$

To approximate the second derivative at the arbitrary node x_i using central method we need to use the neighbouring nodes. The second order approximation for the second order derivative of central difference method is shown in Equation (2.6). The discretized equations are differentiated with respect to shape design variable. This requires the nodal distances to be included in the discretized equations. To maintain the generality, we assume that distance of node T_i to T_{i+1} is Δ_i and the distance of node T_i to T_{i-1} is Δ_{i-1}.

$$\frac{\partial^2 T}{\partial x^2} = \frac{T_{i-1}\Delta_{iL} - T_i(\Delta_{iL} + \Delta_{iR}) + T_{i+1}\Delta_{iR}}{\frac{1}{2}\left[\Delta_{iL}\Delta_{iR}^2 + \Delta_{iL}^2\Delta_{iR}\right]} \tag{2.6}$$

The approximation of the second derivative in Equation (2.3) is done using definitions in Equation (2.6) for each node excluding the boundary nodes. As mentioned in the previous section, the nodal values at boundary nodes, (1) and (2) are known. The discretized governing equations is written in the matrix form as shown in Equation (2.7).

$$\begin{bmatrix} \frac{-2}{\Delta_1\Delta_2} & \frac{2}{\Delta_1\Delta_2+\Delta_1^2} & 0 & 0 \\ \frac{2}{\Delta_3^2+\Delta_2\Delta_3} & \frac{-2}{\Delta_2\Delta_3} & \frac{2}{\Delta_2\Delta_3+\Delta_2^2} & 0 \\ 0 & \frac{2}{\Delta_4^2+\Delta_3\Delta_4} & \frac{-2}{\Delta_3\Delta_4} & \frac{2}{\Delta_3\Delta_4+\Delta_3^2} \\ 0 & 0 & \frac{2}{\Delta_5^2+\Delta_4\Delta_5} & \frac{-2}{\Delta_4\Delta_5} \end{bmatrix} \begin{bmatrix} T_2 \\ T_3 \\ T_4 \\ T_5 \end{bmatrix} = -\begin{bmatrix} \frac{2T_1}{\Delta_2^2+\Delta_1\Delta_2} \\ 0 \\ 0 \\ \frac{2T_6}{\Delta_5\Delta_4+\Delta_5^2} \end{bmatrix} \tag{2.7}$$

To verify the discretization process, we compare the analytical solution of this problem with the result of Equation (2.7) in Figure 2.3. For this problem we choose $T_1 = 0$, $T_6 = 1$, and $L = 1$. As shown in this figure, the discrete and continuum results match very well.

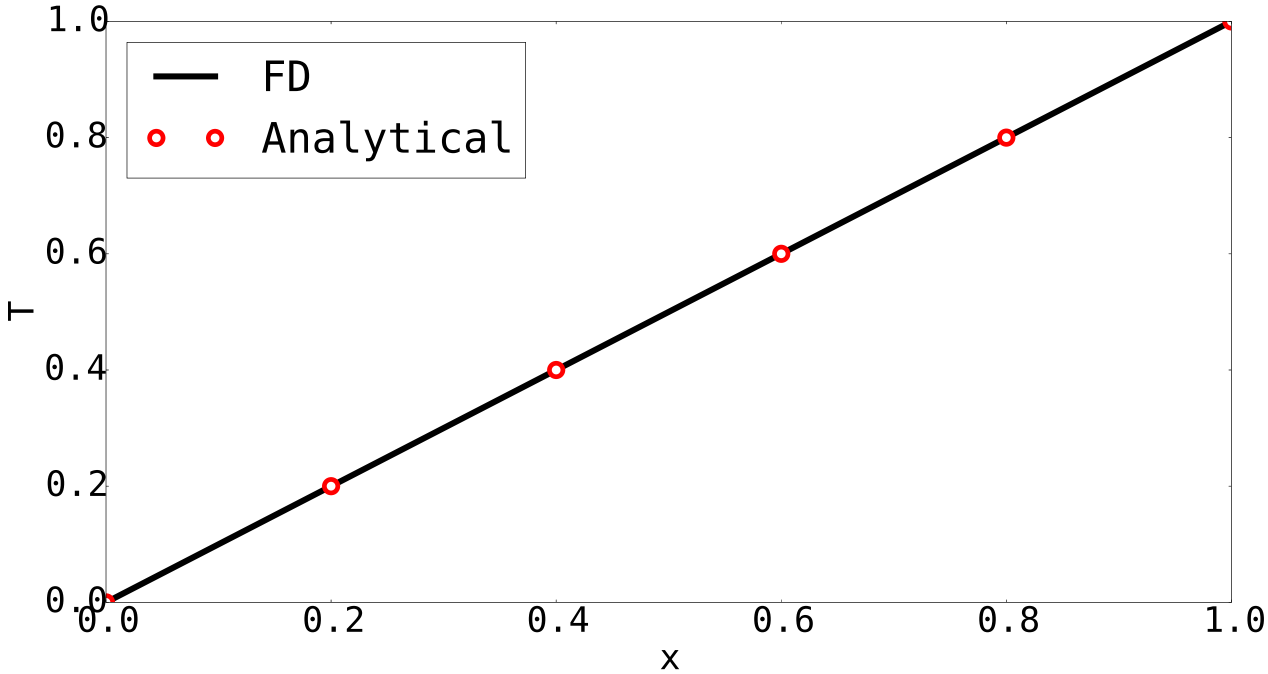


Figure 2.3: Comparison between the analytical and finite difference solution for 1D heat equation.

The sensitivity equations are derived by differentiating the discretized governing equation of (2.7) with respect to the length of the domain. We assume that the change in the length, only affects the nodal distance between the last two nodes and the rest remain unchanged. This means that only the node next to the boundary will move and the rest of nodes will be stationary. This is required to make sure that the sensitivity at each of the degrees of freedom is only a function of change in shape not movement of material nodes.

To calculate the sensitivities, It is required to calculate the derivative of nodal distances in Equation (2.7) with respect to L. For a equally space grid, the nodal distance is written as

$$\Delta x = \frac{L}{n-1}$$

where L is the length of the domain, and n is the number of nodes chosen to discretize the domain. The sensitivity of nodal distances to the length of the domain is calculated as

$$\frac{\partial \Delta x}{\partial L} = \frac{1}{n-1} \tag{2.8}$$

The discretized equation is differentiated as shown in Equation (2.9). It should be noted that since only the adjacent node to the boundary is affected by shape change, only that element in the matrix derivative has a value.

$$\begin{bmatrix} -2 & 1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -2 \end{bmatrix} \begin{bmatrix} T_2' \\ T_3' \\ T_4' \\ T_5' \end{bmatrix} = \frac{1}{2}\frac{\partial \Delta}{\partial L}\frac{1}{\Delta} \begin{bmatrix} 0 \\ 0 \\ 0 \\ T_6 \end{bmatrix} - \underbrace{\frac{1}{2}\frac{\partial \Delta}{\partial L}\frac{1}{\Delta} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -3 & 1 \end{bmatrix} \begin{bmatrix} T_2 \\ T_3 \\ T_4 \\ T_5 \end{bmatrix}}_{\text{effect of shape change}} \tag{2.9}$$

In order to better study above we mention the general sensitivity equation in the discrete form. Assume the the continuum governing equation, can be written in the discrete form as

$$[K]\,[U] = [F]$$

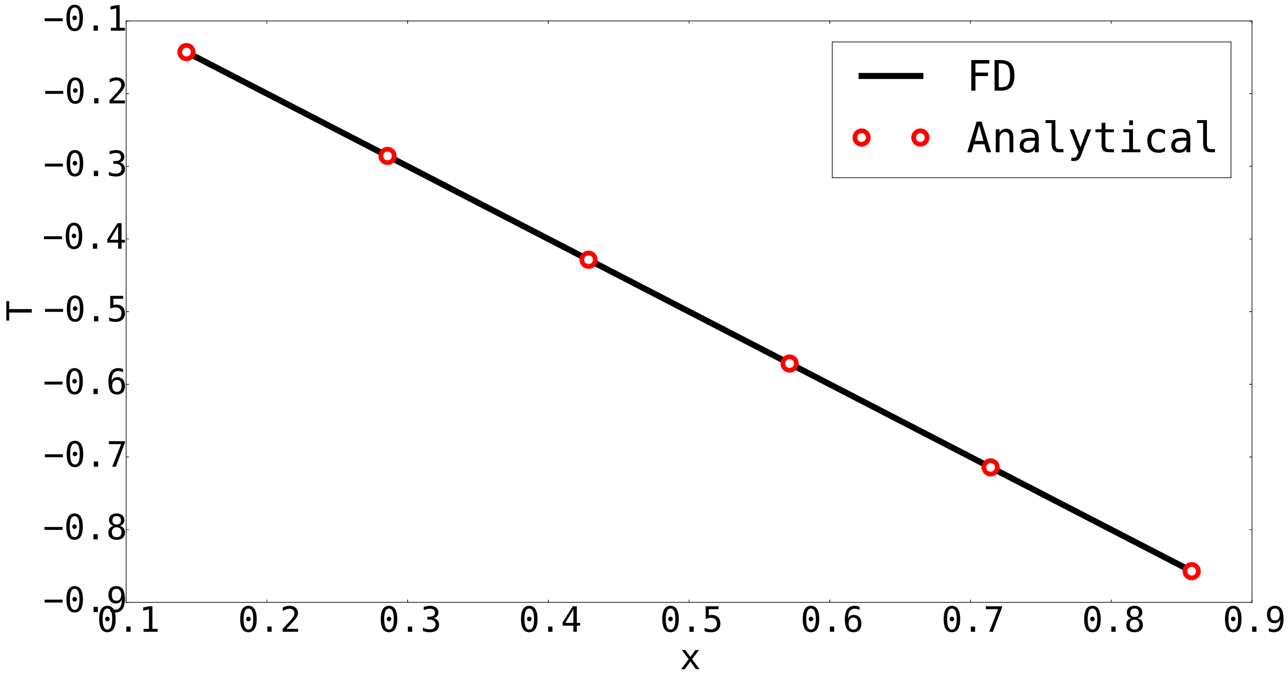


Figure 2.4: Sensitivity analysis verification.

where $[K]$ is the discrete operator that represents the governing equation, $[U]$ is the vector of response variable defined at each of the degrees of freedom, and $[F]$ is the vector of the boundary conditions. The sensitivity of this system with respect to design variable b is derived as

$$[K]\left[\frac{\partial U}{\partial b}\right] = \left[\frac{\partial F}{\partial b}\right] - \left[\frac{\partial K}{\partial b}\right][U]$$

By comparing above equation with Equation (2.9), we can see that the last term represents the change in the stiffness matrix. This depends on how the governing equations are discretized. Therefore, to do the sensitivity analysis it is required to know how this matrix is put together so it can be differentiated. This is not possible in most cases since the details of discretization is not known for the commertial software packages.

To verify the results of Equation (2.9), we compared it with the analytical sensitivity results are shown in Figure 2.4.

2.4 Continuum sensitivity formulation

For a continuum system, the governing equations and boundary conditions can be written as

$$A(u, t; b) = 0 \quad \text{on } \Omega \tag{2.10a}$$
$$B(u, t; b) = g(x, t; b) \quad \text{on } \Gamma \tag{2.10b}$$
$$\tag{2.10c}$$

where u is the response variable, i.e. displacement or pressure, t is time, x is the spatial coordinate, and b is the design variable that can be used to control the solution. A and B are continuum functions that define the governing equations and boundary conditions respectively. It should be noted that the governing equation is written in the residual form where its value needs to be equal to zero when u is the solution at time t. g is the value of the boundary condition of the defined system. To calculate the sensitivity of response, it is required to calculate the total derivative of Equation (2.10). This is broken into calculating the local sensitivity of the system of governing equation followed by converting the local sensitivities to their total form using Equation (2.2).

The governing equation of (2.10) is differentiated as follows

$$A(u', t; b) + A'(u, t; b) = 0 \quad \text{on } \Omega \tag{2.11a}$$
$$B(\dot{u}, t; b) + \dot{B}(u, t; b) = \dot{g}(x, t; b) \quad \text{on } \Gamma \tag{2.11b}$$
$$\tag{2.11c}$$

where u' and $\dot{u}$ are local and total derivative which are defined as follows

$$(\,)' = \frac{\partial(\,)}{\partial b}$$
$$\dot{(\,)} = \frac{D(\,)}{Db}$$

It should be noted that the governing equations are differentiated in the local form and the boundary conditions are differentiated in the total form. Local differentiation of the governing equations enables us to treat the solver nonintrusively. However, in order to capture the effect of shape change, the boundary conditions need to be differentiated in total form. The boundary condition is further simplified by assuming linearly. This is a valid assumption for many practical cases such structural analysis or computational fluid dynamics. The boundary conditions are usually in the form of known gradient or value, i.e. outflow and free-slip wall for CFD and predefined displacement for structural analysis, at the boundary. It should be noted that this assumption is not valid for problems such as contacts. The boundary condition is written as

$$\begin{aligned} &B(\dot{u}, t; b) = \dot{g}(x, t; b) \Rightarrow \\ &B(u', t; b) = \dot{g}(x, t; b) - B(\frac{\partial u}{\partial x} \cdot \frac{\partial x}{\partial b}, t; b) \end{aligned} \tag{2.13}$$

To study the differentiated governing equation of Equation (2.11), we will assume two situations: i) linear analysis, ii) nonlinear analysis.

For the linear analysis, the governing of (2.11) is written as

$$A(u', t; b) = 0 \tag{2.14}$$

This is valid since for the linear case, the differential operators of the governing equation does not depend on the response variable, u. To explain this concept, we look at the transient heat conduction in 1D domain. The governing equations are defined as

$$\frac{\partial^2 T}{\partial x^2} = \frac{1}{\alpha}\frac{\partial T}{\partial t} \tag{2.15}$$

where T is the temperature, x is spatial coordinate, t is time, and α is is the thermal diffusivity $(k/\rho c_p)$. This equation can be differentiated with respect to design variable b as shown in the following equation.

$$\frac{\partial}{\partial b}\left(\frac{\partial^2 T}{\partial x^2}\right) = \frac{\partial}{\partial b}\left(\frac{1}{\alpha}\frac{\partial T}{\partial t}\right)$$

Due to linearity of the differential operators we can change the order of differentiation as follows

$$\frac{\partial^2}{\partial x^2}\left(\frac{\partial T}{\partial b}\right) = \frac{1}{\alpha}\frac{\partial}{\partial t}\left(\frac{\partial T}{\partial b}\right) \tag{2.16}$$

By comparing Equations (**??**) and (**??**) we can see that the differential operators, $\partial^2/\partial x^2$, and $\partial/\partial t$ remain unchanged. Therefore, same analysis can be used for solving this system of governing equations for the sensitivity variable, $\partial T/\partial b$.

For nonlinear problems, the differential operators are functions of response variable as well. For example, the incompressible Euler's equation for a 2D flow is derived as

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + \frac{\partial p}{\partial x} = 0 \tag{2.17a}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + \frac{\partial p}{\partial y} = 0 \tag{2.17b}$$

where u and v are the velocity components in x and y directions respectively. p is the pressure, t is time, x, and y are spatial coordinates. We can rewrite Equation (**??**) in terms of differential operators too.

$$\mathcal{T}u + \mathcal{C}u + \mathcal{G}_x p = 0$$

$$\mathcal{T}v + \mathcal{C}v + \mathcal{G}_y p = 0$$

where $\mathcal{T}$ is the time derivative operator ($\partial/\partial t$), $\mathcal{C}$, is the convective operator ($u\partial/\partial x + v\partial/\partial y$). $\mathcal{G}_x$ and $\mathcal{G}_y$ are gradient operators in x and y respectively ($\partial/\partial x$ and $\partial/\partial y$). The gradient and time derivative operators are linear, therefore they can be treated as shown in the previous paragraphs. On the other hand, the convective operator is nonlinear and it should be differentiated alongside response variables as well. As a result, the sensitivity equations can be written in the operator form as

$$\mathcal{T}u' + \mathcal{C}'u + \mathcal{C}u' + \mathcal{G}_x p' = 0 \tag{2.19a}$$

$$\mathcal{T}v' + \mathcal{C}'v + \mathcal{C}v' + \mathcal{G}_y p' = 0 \tag{2.19b}$$

where

$$\mathcal{C}' = u'\frac{\partial}{\partial x} + v'\frac{\partial}{\partial y}$$

Several interesting properties of CSA can be explained using Equation (**??**). First of all, although the original Euler equation is nonlinear due to multiplication of response variables and their derivatives (u and $\partial u/\partial x$), the resulting sensitivity equation is linear. This means that the sensitivity equations are easier to solve both in terms of algorithms and simulation time compared to the original equations. The challenging expression that needs to be calculated in Equation (**??**) is the convective term.

The first step in solving the sensitivity equations is to get the solution of the governing equations. This enables us to calculate the convective operator, $\mathcal{C}$, at each step of sensitivity solution based on the analysis data. $\mathcal{C}'$ is also calculated at each step of the solution of sensitivity equations based on the solution as previous time step. For a simple predictorcorrector method used to solving the original governing equations, this can be written as follows for sensitivity equation in x direction.

$$\bar{u}' = u'(i) - \Delta t\,[\mathcal{C}'(i)u(i) + \mathcal{C}(i)u'(i) + \mathcal{G}p'(i)] \quad \text{predictor}$$

$$u'(i+1) = u'(i) + \frac{\Delta t}{2} - \left[\mathcal{C}'(i)u(i) + \mathcal{C}(i)u'(i) + \mathcal{G}p'(i) + \bar{\mathcal{C}}(i)u(i) + \mathcal{C}(i)\bar{u} + \mathcal{G}p'(i)\right] \quad \text{corrector}$$

In above equation, $\bar{\mathcal{C}}$ in the convective operator evaluated using $\bar{u}$. It should be noted that in order to generate the operator we do not need to know the details of how it has been put together. We only supply the required material for generating the convective operator and the black-box solver will generate it for us. This input is response variable when solving the governing equation, and is the sensitivity of response variable for sensitivity analysis. Therefore, the solver can still be considered as a black box that we do not modify.

As for the discrete sensitivity case, we use the continuum sensitivity formulation for calculating the sensitivity of temperature with respect to length of a 1D domain. The domain is defined in Figure 2.1. The temperature in the domain is governed by Equation (2.3). The boundary conditions are defined as T_0 at $x = 0$ and T_L at $x = L$.

To get the governing equation for the sensitivity analysis, we need to differentiate the governing equations in the local form and boundary conditions in total form. Because the differential operator is not a function of design variable, the differentiated governing equation and boundary conditions. The boundary conditions for this problem are written as follows to be consistent with the general formulation of the boundary conditions.

$$\begin{cases} BT = T_0 & \text{at x = 0} \\ BT = T_L & \text{at x = L} \end{cases}$$

where $\mathcal{B}$ is the operator acting on the boundary. For this problem, it is equal 1. Above equation is differentiated in the total form since the boundaries are moving. This results in

$$\begin{cases} \dot{B}T + B\dot{T} = \dot{T}_0 & \text{at x = 0} \\ \dot{B}T + B\dot{T} = \dot{T}_L & \text{at x = L} \end{cases}$$

Bibliography

[1] Han, Z.-H., Görtz, S., and Zimmermann, R., "Improving variable-fidelity surrogate modeling via gradient-enhanced kriging and a generalized hybrid bridge function," *Aerospace Science and Technology*, Vol. 25, No. 1, 2013, pp. 177–189.

[2] Pettit, C. L., "Uncertainty quantification in aeroelasticity: recent results and research challenges," *Journal of Aircraft*, Vol. 41, No. 5, 2004, pp. 1217–1229.

[3] Martins, J. R., Sturdza, P., and Alonso, J. J., "The complex-step derivative approximation," *ACM Transactions on Mathematical Software (TOMS)*, Vol. 29, No. 3, 2003, pp. 245–262.

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

[5] Choi, K. K. and Kim, N.-H., *Structural sensitivity analysis and optimization 1: linear systems*, Springer Science & Business Media, 2006.

[6] Arora, J. S. and Haug, E. J., "Methods of design sensitivity analysis in structural optimization," *AIAA journal*, Vol. 17, No. 9, 1979, pp. 970–974.

[7] Reuther, J. J., Jameson, A., Alonso, J. J., Rimllnger, M. J., and Saunders, D., "Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers, part 2," *Journal of aircraft*, Vol. 36, No. 1, 1999, pp. 61–74.

[8] Gamboa, P., Vale, J., P. Lau, F., and Suleman, A., "Optimization of a morphing wing based on coupled aerodynamic and structural constraints," *AIAA journal*, Vol. 47, No. 9, 2009, pp. 2087–2104.

[9] Pandya, M. J. and Baysal, O., "Gradient-based aerodynamic shape optimization using alternating direction implicit method," *Journal of aircraft*, Vol. 34, No. 3, 1997, pp. 346–352.

[10] Kim, H.-J., Sasaki, D., Obayashi, S., and Nakahashi, K., "Aerodynamic optimization of supersonic transport wing using unstructured adjoint method," *AIAA journal*, Vol. 39, No. 6, 2001, pp. 1011–1020.

[11] Haftka, R. T. and Adelman, H. M., "Recent developments in structural sensitivity analysis," *Structural optimization*, Vol. 1, No. 3, 1989, pp. 137–151.

[12] Borggaard, J. and Burns, J., "A PDE sensitivity equation method for optimal aerodynamic design," *Journal of Computational Physics*, Vol. 136, No. 2, 1997, pp. 366–384.

[13] Hristova, H., Etienne, S., Pelletier, D., and Borggaard, J., "A continuous sensitivity equation method for time-dependent incompressible laminar flows," *International Journal for numerical methods in fluids*, Vol. 50, No. 7, 2006, pp. 817–844.

[14] Arora, J. and Haug, E., "Methods of Design Sensitivity Analysis in Structural Optimization," *AIAA Journal*, Vol. 17, No. 9, 1979, pp. 970–974.

[15] Dems, K. and Mroz, Z., "Variational approach to first- and second-order sensitivity analysis of elastic structures," *International Journal for Numerical Methods in Engineering*, Vol. 21, No. 4, 1985, pp. 637–661.

[16] Borggaard, J. and Burns, J., *A sensitivity equation approach to shape optimization in fluid flows*, Springer, 1995.

[17] Stanley, L. G. D. and Stewart, D. L., *Design Sensitivity Analysis: Computational Issues of Sensitivity Equation Methods*, Frontiers in applied mathematics, Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 2002.

[18] Etienne, S. and Pelletier, D., "A general approach to sensitivity analysis of fluid–structure interactions," *Journal of Fluids and Structures*, Vol. 21, No. 2, 2005, pp. 169–186.

[19] Liu, S. and Canfield, R. A., "Equivalence of continuum and discrete analytic sensitivity methods for nonlinear differential equations," *Structural and Multidisciplinary Optimization*, Vol. 48, No. 6, 2013, pp. 1173–1188.

[20] Sahin, M. and Mohseni, K., "An arbitrary Lagrangian–Eulerian formulation for the numerical simulation of flow patterns generated by the hydromedusa Aequorea victoria," *Journal of Computational Physics*, Vol. 228, No. 12, 2009, pp. 4588–4605.

[21] Liu, S. and Canfield, R. A., "Boundary velocity method for continuum shape sensitivity of nonlinear fluidstructure interaction problems," *Journal of Fluids and Structures*, Vol. 40, No. 0, 2013, pp. 284 – 301.

[22] Peskin, C. S., "Numerical analysis of blood flow in the heart," *Journal of computational physics*, Vol. 25, No. 3, 1977, pp. 220–252.

[23] Anderson, J. D. and Wendt, J., *Computational fluid dynamics*, Vol. 206, Springer, 1995.

[24] Saiki, E. and Biringen, S., "Numerical simulation of a cylinder in uniform flow: application of a virtual boundary method," *Journal of Computational Physics*, Vol. 123, No. 2, 1996, pp. 450–465.

[25] Zhu, L. and Peskin, C. S., "Interaction of two flapping filaments in a flowing soap film," *Physics of Fluids (1994-present)*, Vol. 15, No. 7, 2003, pp. 1954–1960.

[26] Beyer, R. P. and LeVeque, R. J., "Analysis of a one-dimensional model for the immersed boundary method," *SIAM Journal on Numerical Analysis*, Vol. 29, No. 2, 1992, pp. 332–364.

[27] Peskin, C. S. and McQueen, D. M., "A three-dimensional computational method for blood flow in the heart I. Immersed elastic fibers in a viscous incompressible fluid," *Journal of Computational Physics*, Vol. 81, No. 2, 1989, pp. 372–405.

[28] Fauci, L. J. and Peskin, C. S., "A computational model of aquatic animal locomotion," *Journal of Computational Physics*, Vol. 77, No. 1, 1988, pp. 85–108.

[29] Kempe, T., Lennartz, M., Schwarz, S., and Fröhlich, J., "Imposing the free-slip condition with a continuous forcing immersed boundary method," *Journal of Computational Physics*, Vol. 282, 2015, pp. 183–209.

[30] Uhlmann, M., "An immersed boundary method with direct forcing for the simulation of particulate flows," *Journal of Computational Physics*, Vol. 209, No. 2, 2005, pp. 448–476.

[31] Mittal, R. and Iaccarino, G., "Immersed boundary methods," *Annu. Rev. Fluid Mech.*, Vol. 37, 2005, pp. 239–261.

[32] Goldstein, D., Handler, R., and Sirovich, L., "Modeling a no-slip flow boundary with an external force field," *Journal of Computational Physics*, Vol. 105, No. 2, 1993, pp. 354–366.

[33] Mohd-Yusof, J., "Combined immersed-boundary/B-spline methods for simulations of ow in complex geometries," *Annual Research Briefs. NASA Ames Research Center= Stanford University Center of Turbulence Research: Stanford*, 1997, pp. 317–327.

[34] Verzicco, R., Mohd-Yusof, J., Orlandi, P., and Haworth, D., "LES in complex geometries using boundary body forces," *Center for Turbulence Research Proceedings of the Summer Program, NASA Ames= Stanford University*, 1998, pp. 171–186.

[35] Verzicco, R., Fatica, M., Iaccarino, G., Moin, P., and Khalighi, B., "Large eddy simulation of a road vehicle with drag-reduction devices," *AIAA journal*, Vol. 40, No. 12, 2002, pp. 2447–2455.

[36] Iaccarino, G. and Verzicco, R., "Immersed boundary technique for turbulent flow simulations," *Applied Mechanics Reviews*, Vol. 56, No. 3, 2003, pp. 331–347.

[37] Durlofsky, L. and Brady, J., "Analysis of the Brinkman equation as a model for flow in porous media," *Physics of Fluids*, Vol. 30, No. 11, 1987, pp. 3329–3341.

[38] Arquis, E. and Caltagirone, J., "Sur les conditions hydrodynamiques au voisinage dune interface milieu fluide-milieu poreux: applicationa la convection naturelle," *CR Acad. Sci. Paris II*, Vol. 299, 1984, pp. 1–4.

[39] Angot, P., "Analysis of singular perturbations on the Brinkman problem for fictitious domain models of viscous flows," *Mathematical methods in the applied sciences*, Vol. 22, No. 16, 1999, pp. 1395–1412.

[40] Gazzola, M., Chatelain, P., Van Rees, W. M., and Koumoutsakos, P., "Simulations of single and multiple swimmers with non-divergence free deforming geometries," *Journal of Computational Physics*, Vol. 230, No. 19, 2011, pp. 7093–7114.

[41] Kevlahan, N. K.-R. and Ghidaglia, J.-M., "Computation of turbulent flow past an array of cylinders using a spectral method with Brinkman penalization," *European Journal of Mechanics-B/Fluids*, Vol. 20, No. 3, 2001, pp. 333–350.

[42] Clarke, D. K., Hassan, H., and Salas, M., "Euler calculations for multielement airfoils using Cartesian grids," *AIAA journal*, Vol. 24, No. 3, 1986, pp. 353–358.

[43] Kirkpatrick, M., Armfield, S., and Kent, J., "A representation of curved boundaries for the solution of the Navier–Stokes equations on a staggered three-dimensional Cartesian grid," *Journal of Computational Physics*, Vol. 184, No. 1, 2003, pp. 1–36.

[44] Hu, X., Khoo, B., Adams, N. A., and Huang, F., "A conservative interface method for compressible flows," *Journal of Computational Physics*, Vol. 219, No. 2, 2006, pp. 553–578.

[45] Udaykumar, H., Mittal, R., and Shyy, W., "Computation of solid–liquid phase fronts in the sharp interface limit on fixed grids," *Journal of computational physics*, Vol. 153, No. 2, 1999, pp. 535–574.

[46] Borrvall, T. and Petersson, J., "Topology optimization of fluids in Stokes flow," *International journal for numerical methods in fluids*, Vol. 41, No. 1, 2003, pp. 77–107.

[47] Challis, V. J. and Guest, J. K., "Level set topology optimization of fluids in Stokes flow," *International journal for numerical methods in engineering*, Vol. 79, No. 10, 2009, pp. 1284–1308.

[48] Deaton, J. D. and Grandhi, R. V., "A survey of structural and multidisciplinary continuum topology optimization: post 2000," *Structural and Multidisciplinary Optimization*, Vol. 49, No. 1, 2014, pp. 1–38.

[49] LeVeque, R. J. and Li, Z., "Immersed interface methods for Stokes flow with elastic boundaries or surface tension," *SIAM Journal on Scientific Computing*, Vol. 18, No. 3, 1997, pp. 709–735.

[50] Pingen, G., Evgrafov, A., and Maute, K., "Topology optimization of flow domains using the lattice Boltzmann method," *Structural and Multidisciplinary Optimization*, Vol. 34, No. 6, 2007, pp. 507–524.

[51] Mase, G. T., Smelser, R. E., and Mase, G. E., *Continuum mechanics for engineers*, CRC press, 2009.

[52] Cross, D. M., *Local Continuum Sensitivity Method for Shape Design Derivatives Using Spatial Gradient Reconstruction*, Virginia Tech, 2014.

[53] Haftka, R. T. and Grandhi, R. V., "Structural shape optimizationa survey," *Computer Methods in Applied Mechanics and Engineering*, Vol. 57, No. 1, 1986, pp. 91–106.