The Aeroelastic Instability of a Cantilevered Beam in Two-dimensional Axial Flow

Gabriel Yam

University of New South Wales at the Australian Defence Force Academy

The aeroelastic instability of a cantilevered beam in two-dimensional axial flow is investigated as a representation of the dynamics of a folded rotor blade design concept. The fluid-structure interaction is numerically simulated using MATLAB and validated with FLUENT. The beam is assumed to be one-dimensional, uniform and obeys Euler-Bernoulli beam theory. The beam is discretized using a finite element method. The flow field is assumed to be two-dimensional, subsonic, incompressible, viscous, laminar and unsteady. A pressure correction technique is used to solve the Navier-Stokes equations at all points in the flow field, discretized by the Taylor series finite difference expansion equations. An attempt is made to optimize, verify and validate this numerical simulation.

Contents

I Introduction 3
   A Inspiration 3
   B Background 3
   C Initial Concept 4
II Problem Identification 4
   A Literature Review 6
   B Aims 8
III Methodology 8
   A Beam Mechanics 8
      1. Basic Parameters 8
      2. Transverse Vibrations on a Cantilevered Beam 9
      3. Analytical Solution 10
      4. Finite Element Method 11
      5. Damping 13
      6. Code Work-up 13
   B Fluid Mechanics 14
      1. Basic Parameters 14
      2. Nature of Partial Differential Equations 15
      3. Taylor Series Expansion 16
      4. Pressure Correction Technique 17
      5. Relaxation Technique 21
      6. Staggered Grid 22
      7. Boundary Conditions 23
      8. Implicit time stepping vs Explicit time stepping 23
      9. Optimization of Convergent Solution 24
     10. Code Work-up 25
   C Coupling 25
   D Grid 26
      1. Compressed Grid 26
      2. De-Conflicting Beam Nodes with the Flow Field Grid Points 27
      3. Reasons for not using a Regularly Spaced Grid 27
      4. Code Work-up 27
     E Operating Instructions 28
IV Results and Analysis 28
V Conclusions 33
VI Recommendations 34

1 Lieutenant (Republic of Singapore Air Force), School of Aerospace, Civil & Mechanical Engineering. ZACM 4049/4050, Aeronautical Engineering: Project, Thesis & Practical Work Experience A/B
## Nomenclature

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E$</td>
<td>Modulus of elasticity [GPa]</td>
</tr>
<tr>
<td>$\rho$</td>
<td>Density [kg/m$^3$]</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>Ratio of specific heats</td>
</tr>
<tr>
<td>$\mu$</td>
<td>Dynamic viscosity [N•s/m$^2$]</td>
</tr>
<tr>
<td>$\nu$</td>
<td>Kinematic viscosity [m$^2$/s]</td>
</tr>
<tr>
<td>$R$</td>
<td>Gas constant [J/kg•K]</td>
</tr>
<tr>
<td>$T$</td>
<td>Temperature [K]</td>
</tr>
<tr>
<td>$P_{atm}$</td>
<td>Atmospheric Pressure [Pa]</td>
</tr>
<tr>
<td>$Re$</td>
<td>Reynolds number</td>
</tr>
<tr>
<td>$A(x)$</td>
<td>Cross-sectional area of the beam [m$^2$]</td>
</tr>
<tr>
<td>$I_{zz}$</td>
<td>Moment of inertia [m$^4$]</td>
</tr>
<tr>
<td>$b$</td>
<td>Width of beam [m]</td>
</tr>
<tr>
<td>$h$</td>
<td>Height of beam [m]</td>
</tr>
<tr>
<td>$w$</td>
<td>Vertical displacement of the beam [m]</td>
</tr>
<tr>
<td>$C$</td>
<td>Damping co-efficient</td>
</tr>
<tr>
<td>$V(x,t)$</td>
<td>Distribution of the external forcing on the beam [N]</td>
</tr>
<tr>
<td>$X(\alpha)$</td>
<td>Spatial solution</td>
</tr>
<tr>
<td>$c$</td>
<td>Separation constant [m$^4$/s$^2$]</td>
</tr>
<tr>
<td>$\beta_{nl}$</td>
<td>Weighted frequency for nth mode</td>
</tr>
<tr>
<td>$\sigma_{n}$</td>
<td>Mode shape variable for nth mode</td>
</tr>
<tr>
<td>$\omega_{n}$</td>
<td>Natural frequency for nth mode [Hz]</td>
</tr>
<tr>
<td>$u_1, u_2$</td>
<td>Linear coordinates at the beam nodes</td>
</tr>
<tr>
<td>$V$</td>
<td>Flow velocity [m/s$^2$]</td>
</tr>
<tr>
<td>$M$</td>
<td>Mass matrix</td>
</tr>
<tr>
<td>$K$</td>
<td>Stiffness matrix</td>
</tr>
<tr>
<td>$K_{global}$</td>
<td>Global stiffness matrix</td>
</tr>
<tr>
<td>$B$</td>
<td>Forcing term co-efficient</td>
</tr>
<tr>
<td>$x0$</td>
<td>Array containing the initial conditions for the beam</td>
</tr>
<tr>
<td>$f(i)$</td>
<td>Value of function $f$ at coordinate $i$</td>
</tr>
<tr>
<td>$\Delta x, \delta x$</td>
<td>Distance between two adjacent grid points in the x-direction [m]</td>
</tr>
<tr>
<td>$u, v$</td>
<td>Velocity component in the x-direction [m/s$^2$]</td>
</tr>
<tr>
<td>$p$</td>
<td>Local pressure [Pa]</td>
</tr>
<tr>
<td>$\Delta y, \delta y$</td>
<td>Distance between two adjacent grid points in the y-direction [m]</td>
</tr>
<tr>
<td>$\Delta t, \delta t$</td>
<td>Time step [s]</td>
</tr>
<tr>
<td>$u^<em>, v^</em>, p^*$</td>
<td>Guess values</td>
</tr>
<tr>
<td>$u^{(N+1)}, v^{(N+1)}, p^{(N+1)}$</td>
<td>Next guess values for $u$ [m/s$^2$], $v$ [m/s$^2$], $p$ [Pa]</td>
</tr>
<tr>
<td>$p'^*$</td>
<td>Corrected pressure value at guess level $T$[Pa]</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Relaxation factor, alpha</td>
</tr>
<tr>
<td>$\Delta y_v$</td>
<td>Vertical distance between two adjacent grid points that capture $v$</td>
</tr>
<tr>
<td>$\Delta y_p$</td>
<td>Vertical distance between two adjacent grid points that capture $p$</td>
</tr>
<tr>
<td>$\Delta x_u$</td>
<td>Horizontal distance between two adjacent grid points that capture $u$</td>
</tr>
<tr>
<td>$\Delta x_p$</td>
<td>Horizontal distance between two adjacent grid points that capture $p$</td>
</tr>
<tr>
<td>$\omega_0$</td>
<td>Relaxation factor, omega</td>
</tr>
<tr>
<td>$\eta$</td>
<td>Corresponding compressed y-axis</td>
</tr>
<tr>
<td>$\xi$</td>
<td>Corresponding x-axis</td>
</tr>
</tbody>
</table>
I. Introduction

A. Inspiration

In December 2000, the Columbia TriStar Motion Picture Group released a movie named - *The 6th Day.* This movie featured a helicopter which exhibited Vertical Take-Off and Landing (VTOL) capabilities but was also capable of flying forward at supersonic speeds. This helicopter was able to stop its rotor blades in mid-flight and fold them approximately 30° backwards, before clamping the blades at the rotor hub. This effectively cantilevered the blades and transformed them into fixed wings, which then provided the primary lifting surface for the aircraft to sustain flight. In this configuration, the overall drag on the helicopter was significantly reduced and forward flight at supersonic speeds achievable. Figure 1 illustrates this helicopter whilst still in the helicopter configuration and Fig 2 illustrates it in the transformed fixed wing configuration. Inspired by the movie, this author wishes to see if this is physically possible.

![Figure 1. Helicopter in the rotor wing configuration (The 6th Day, 2000)](image1)

![Figure 2. Helicopter in the fixed wing configuration (The 6th Day, 2000)](image2)

B. Background

In the past century, there have been numerous attempts at combining the capabilities of VTOL and fast forward speeds. Traditionally, VTOL capabilities have been restricted to helicopters and aircrafts of similar rotor design. These designs lend additional stability to the aircraft when operating in VTOL, but inherently produce difficulties when greater forward speeds are required. To explain this briefly, when attempting to achieve a faster forward speed, helicopters increase the rotor blade rotational velocity in order to produce a greater amount of forward thrust. This increase in angular velocity implies that the advancing blade encounters local supersonic flow whilst the retreating blade stalls. This combination significantly increases the drag on the helicopter, making it difficult for faster forward speed to be achieved (Leishman, 2006 pp.223). Similarly, aircrafts that have been designed to operate at high forward speeds have been modified to also achieve VTOL. The AV-8 Harrier and the F-35 Joint Strike Fighter (JSF) are two such examples. Such aircrafts traditionally use thrust vectoring to achieve VTOL but have been found to be difficult to consistently operate without the use of computers. Suffice to say, using thrust vectoring to achieve VTOL also has its difficulties.

In 1970, a joint US Army, NASA and Sikorsky program designed and built a helicopter which could stop its rotor blades in mid-flight and still maintain sufficient lift. The S-72 “X-Wing” has extremely rigid rotor blades which allow the blades to be stopped with greater ease. Furthermore, because of the high rigidity in the blades, when in the stopped position these blades double as a secondary lifting surface. The S-72 also has fixed wings which produces the primarily lift for the aircraft. Refer to Appendix A for more information on related aircrafts. To date, the fastest helicopter in operation is the ZB500 Lynx, which can attain a forward speed of 333km/h or 92.5 m/s.

---

* Yahoo Website for movie reviews. [http://movies.yahoo.com/movie/1800420003/details](http://movies.yahoo.com/movie/1800420003/details) accessed on 23 Apr 08

Final Thesis Report 2008, UNSW@ADFA
C. Initial Concept

The initial concept, proposed by this author, was to fold the rotor blades fully backwards, inline with the tail boom of the helicopter. Lifting surfaces (wing and canard) would simultaneously be deployed, in an F-111 fashion, so that the helicopter would still be able to maintain flight. It was postulated that if the rotor blades were used to produce lift (like in the movie) they would have to be extremely rigid and thick, implying additional weight and design complexity. This additional weight and design complexity could eventually result in the design being less cost effective, negating the benefits of this modification in the first place. Therefore the rotor blades are completely folded backwards, attempting to keep them out of the airflow. By clamping the blades at the rotor hub, these blades are stored in this position for the duration of forward/cruise flight. Given that the helicopter now has no rotor blades to generate lift, thrust would be derived from the jet engine in a fighter aircraft configuration.

With regards to the folding of the blades, if we consider the physics of this design it becomes immediately evident that several difficulties will be encountered. Firstly, the reasonably flexible rotor blades will have to be stopped quickly in mid-flight and a folding mechanism utilized to fold the blades backwards for storage. Achieving this will undoubtedly result in additional weight and mechanical complexity at the rotor hub. Furthermore, the Fluid-Structure Interaction (FSI) between the flow and the blades, at the time of folding, may cause an unstable coupling and eventual structural failure. Needless to say, any analysis of the fluid-structure interaction while the blades are in the folding process must be three-dimensional and is beyond the scope of this undergraduate thesis. This author proceeds by choosing the simplest configuration to analyse: when the rotor blades have been folded completely backwards, cantilevered at the rotor hub and only experiences axial flow. It is noteworthy that this configuration is also the cruise configuration, when the helicopter is in forward flight, and is therefore one of the most common configurations. In its simplest configuration, if the rotor blades are found to exhibit an unstable vibrational response due to the fluid-structure interaction, then clearly the more complicated configurations do not need to be analyzed.

II. Problem Identification

As stated before, this author has chosen to analyse the configuration when the rotor blades are folded completed backwards, cantilevered at the rotor hub and is assumed to predominantly experience axial flow. The simplest configuration is chosen because it is a starting place for the analysis of this design concept and is arguably the configuration in which the aircraft will spend a significant portion of flight time. Furthermore, this allows the problem to be scaled to a level suitable for an undergraduate thesis whilst still maintaining the appropriate level of difficulty and challenge.

The problem described here and investigated in this thesis is a classic fluid-structures interaction (FSI) scenario. The axial flow over the beam sets up a differential pressure distribution over the upper and lower surfaces of the beam. This pressure distribution results in a force distribution on the beam which results in the beam distorting. This distortion causes a change in the flow parameters and a different pressure distribution is now set up on the beam. The new force on the beam serves to further distort it again and the cycle continues as time progresses. It is evident that two-way coupling exist between the flow field and the beam mechanics. At this point, this author would like to point out that there are several computer packages commercially available that claim to accurately model this problem. The ANSYS Mechanical and ANSYS CFX package is one such computer package. However, it is also noteworthy to point out that if one intends to fully understand the beam and flow mechanics of the problem, then using these ‘black box’ software packages will not be sufficient. It is for this reason, among many others, that this author has chosen to use MATLAB to create a numerical simulation that will attempt to model this fluid-structure interaction.

The detailed explanation of the beam and fluid mechanics, assumptions used and reasons why, are found in a later chapter of this thesis, therefore only a brief introduction of the problem will be provided here. The beam is assumed to obey Euler-Bernoulli beam theory and its distortions restricted to only the y-axis. This implies that any distortion in the beam will be a deflection in the transverse (y-axis) direction. The beam is also modeled to be one-dimensional, which implies an infinite beam width. The rotor blades have been modeled as a single beam, cantilevered at one end and free at the other, setting up a clamped-free boundary condition. Furthermore, the beam is discretized using a finite element method commonly used in the ANSYS software, and the deflections obtained by an ODE45 function in MATLAB.
For the purposes of clarity, the cantilevered end of the beam is now defined as the leading edge and the free end the trailing edge. This is because the cantilevered end first encounters the flow, given that the beam is assumed to predominantly experience an axial flow. The Cartesian coordinate system for the beam is illustrated in Fig 3 and the various axes are as indicated. The x-axis (green line) runs along the length of the beam from the leading edge to the trailing edge, while the y-axis (blue line) is positive upwards and the z-axis (red line) is positive to the right, as seen in the figure. The coordinate \( \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \) is denoted at the midpoint of the leading edge face.

The cross-sectional area of the beam is approximated to be rectangular and constant along the x-axis. The dimensions of the beam are 7m by 0.2m by 0.03m (length x width x height) respectively. For simplicity, the beam is also assumed to be of a uniform material, allowing the Modulus of Elasticity (E) to be kept constant. This author chooses an arbitrary material, stainless steel alloy 304, because the material properties are readily available in most common textbooks. The material properties can be subsequently adjusted to more realistic ones upon the completion of this thesis. Table 1 provides the material properties of stainless steel alloy 304 (Callister, D. William, 2003 Table B2).

<table>
<thead>
<tr>
<th>Stainless steel alloy 304</th>
<th>Material properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modulus of Elasticity, E</td>
<td>193 GPa</td>
</tr>
<tr>
<td>Density, ( \rho )</td>
<td>8.0 x 10^3 kg/m^3</td>
</tr>
<tr>
<td>Air</td>
<td></td>
</tr>
<tr>
<td>Ratio of Specific Heats, ( \gamma )</td>
<td>1.4</td>
</tr>
<tr>
<td>Density, ( \rho )</td>
<td>1.2250 kg/m^3</td>
</tr>
<tr>
<td>Dynamic Viscosity, ( \mu )</td>
<td>1.79 x 10^{-5} Ns/m^2</td>
</tr>
<tr>
<td>Kinematic Viscosity, ( \nu )</td>
<td>1.26 x 10^{-5} m^2/s</td>
</tr>
<tr>
<td>Gas constant, ( R )</td>
<td>2.869 x 10^5 J/kg*K</td>
</tr>
<tr>
<td>Temperature, ( T )</td>
<td>288.16 K</td>
</tr>
<tr>
<td>Atmospheric Pressure, ( P_{atm} )</td>
<td>1.01325 x 10^5 Pa</td>
</tr>
</tbody>
</table>

*Material properties of air taken at mean sea level
  (Munson, Young and Okiishi, 2006 Table 1.8)*

Table 1. Material Properties
Having now defined the beam model, the numerical simulation then models fluid flow over this beam. Figure 4 illustrates how the beam model is inserted into the flow field. The flow field is assumed to be only two-dimensional, running from the left to the right in the figure. The flow field is assumed to be predominantly axial, at least at the beginning of the flow grid, denoted by the plane \((y,0)\) in Fig 4. The beam is modeled in the centre of the flow field because the flow characteristics all around the beam are of interest to the analysis. Figure 4 illustrates a regularly spaced grid both before and after the beam. The flow regions above and below the beam have been compressed with more grid points because this is the region of special interest since the pressure distribution on the top and bottom surface of the beam is the end goal. This author wishes to point out that the actual flow grid used in the numerical simulation is a slightly modified version of Fig 4. Figure 4 is shown here only as an introduction to the flow field used. The actual flow field grid used is described in a later chapter of this thesis, and further illustrations can be found there. The red circles are grid points in the flow field where the flow parameters are captured and calculated, while the black line illustrates the position of the beam. The dimensions of the flow field grid are 50m by 40m (length by height). These values were chosen specifically because the focus of this simulation is to find the differential pressure distribution over the beam. This numerical simulation is built such that if the flow downstream of the beam was of interest, these dimension values can be changed accordingly.

This numerical simulation models the flow as subsonic, incompressible, viscous, laminar and unsteady. Given these particular flow assumptions, a pressure correction technique was chosen to capture and evaluate the flow parameters at each grid point. This method was also chosen because of the particular mathematical behavior of the Navier-Stokes equations under these assumptions. A more in-depth discussion into the mathematical behavior of partial differential equations can be found in a later chapter of this thesis. The fluid used in this numerical simulation is air and its flow properties can be found in Table 1. Figure 4 also illustrates the beam in its initial displaced condition (time = 0s). The beam is deflected downwards under its own weight because of gravitational forces. Again, the details for this calculation can be found in a later chapter. This author would like to point out that the effects of gravity was only used to obtain this initial deflection, and was not accounted for in all subsequent iterative calculations.

Now that the problem has been briefly described for the initial conditions, this numerical simulation then proceeds to obtain the pressure distribution over the beam in an iterative time-marching scheme. After obtaining the pressure distribution over the beam, a distributed forcing term on the surface of the beam can be developed and applied to the beam, thereby resulting in some sort of beam distortion. This newly distorted beam is then inserted into the flow field again and the iteration proceeds into the future. Within each iteration, if the time step is kept small enough, this iterative process can approximate a continuous interaction between the fluid and the beam.

A. Literature Review

A thorough literature review was conducted in order to fully understand the physics behind this problem as well as to avoid, as far as possible, re-inventing the wheel. The literature served to provide guidance into the methods commonly used when numerically simulating similar problems and these published results can serve to validate the numerical simulation produced through this thesis. It is in this author’s opinion that research work applicable to this problem is in the field of flow over a flexible plate, sheet or flag. It was found in the literature that flags or sheets are used to refer to structures with perfect flexibility. However, these terms can be considered interchangeable with plate when the bending stiffness, regardless of magnitude, is taken into account.
Haselgrove analyzed the motion of flags with zero mass and no bending stiffness. He found that instability occurred when the plate was subjected to small, periodic perturbations. In his analysis, the flag was observed to "roll up" (Haselgrove, 1973). Taneda was probably the first to conduct experiments on the motion of a flexible plate. He investigated flag flutter experimentally but considered only hanging configurations. This implied that the flags were under tensile loading due to gravitational forces in his experiments (Taneda, 1968). Datta and Gottenberg also conducted similar experiments on vertically hanging cantilevered beams and tried to theoretically predict the onset of flutter instability with critical flow velocity. Their research also utilized slender wing theory to obtain the theoretical aerodynamic loading on the cantilevered beam (Datta and Gottenberg, 1975). These papers prove that the beam mass and bending stiffness is important and must be considered in the numerical simulation.

Investigations by Huang on a cantilevered beam are made through the application of (Theodorsen, 1935) theory and addition of a wake behind the plate along the neutral axis to account for the boundary conditions. He used a predictor-corrector numerical method and found that a linear analysis was sufficient for the initial stage of flutter instability. Huang also considered the “droop” condition of the cantilevered plate, whereby the trailing edge of the plate is initially displaced downwards due to gravitational forces. His results show that, for a relatively thin plate, the dynamic characteristics are the same for when gravitational forces are ignored (Huang, 1995). This suggests reasonable supporting evidence that a constant gravitational force can be ignored in our analysis.

Shayo extended Kornecki’s theoretical work in a similar fashion but considered a three-dimensional stability analysis for flutter instability. He did this by investigating plates with a finite width. His results suggested that a flag of infinite width is more stable than a finite width one (Shayo, 1980). This result was later found to be contradictory to subsequent studies in (Datta and Gottenberg, 1975), (Lemaitre et al, 2005), (Lucey and Carpenter, 1993), (Argentina and Mahadevan, 2005) and (Eloy et al., 2007). All found that a finite width is always more stable than an infinite one. Lucey and Carpenter investigated theoretically the linear stability of finite aspect ratio panels embedded in a rigid wall. Note that this case bears a strong similarity with flutter effects of flags in axial flow (Lucey and Carpenter, 1993). Their results were found to be agreeable with Argentina and Mahadevan, who also considered the flutter instability of a simple two-dimensional flag model based on (Kornecki et al., 1976), with a three-dimensional numerical simulation. The model was then adjusted to account for the finite aspect ratio of the flag. A modified version of (Theodorsen, 1935) theory was used to obtain the aerodynamic loads on the flag. Both these papers seem to suggest that a finite width stabilizes the system from flutter instability (Argentina and Mahadevan, 2005). These papers seem to suggest that a finite width plate is more stable than an infinite width plate, implying that the results from an infinite width numerical simulation will be an overestimation of the aeroelastic instability. If this is true, then the real physical flow should induce a smaller instability than that predicted by the numerical simulation.

A numerical analysis conducted by Watanabe et al. combined the linear beam model with a two-dimensional solver based on the Navier-Stokes equation, with a time marching solution, to obtain their stability analysis (Watanabe et al., 2002a). Their investigation in the cantilevered beam model yielded numerical results that had good correlation to the analytical results of (Kornecki et al., 1976). Watanabe et al. also conducted an experimental analysis and found that the experimental values for the critical flow velocity to be at least two times that for numerical or analytical predictions, for all experimental parameters (Watanabe et al., 2002a). This discrepancy is also found in investigations done by Tang and Païdousis (Tang and Païdousis, 2007), who concluded that whilst theoretical results predict a supercritical onset of flutter, experimental results show a subcritical flutter onset, with strong hysteresis. Clearly these discrepancies show that while theoretical predictions are necessary, these predictions cannot be wholly accepted without experimental verification. The paper by Tang and Païdousis provides some useful suggestions as to why these discrepancies occur.

The analytical stability analysis carried out by Eloy et al. (2007) was based on a three-dimensional aerodynamic loading model obtained by a direct solution of the potential flow problem. Note that the three-dimensional model accounts for the finite width and length of the plate to calculate the surrounding flow. This aerodynamic load model was calculated in Fourier space and the stability analysis of the elastic plate in axial flow was obtained by the Galerkin method. Their stability analysis assumed that the flutter modes would always be two-dimensional (Eloy et al., 2007). This two-dimensional assumption is justified by results found in (Datta and Gottenberg, 1975), (Kornecki et al., 1976), (Shelley et al., 2005), (Souilliez et al., 2006), (Watanabe et al., 2002a) and (Yamaguchi et al., 2000). All these papers conclude that flutter modes observed at threshold are

---

2 Phenomenon where by the response of a physical system depends not only on the present external influences but also on the previous history of the system.
always two-dimensional. *These papers suggest that instability due to flutter will almost always be two-dimensional, even when considering a three-dimensional flow. The earlier restriction of only transverse deflection in a two-dimensional flow field seems to be well supported by the literature.*

Tang and Paidousis published two papers, both of which used a numerical analysis of the cantilevered plate in axial flow. They considered the plate clamped at the leading edge and assumed an infinite width. It follows on that there is no cross flow between the upper and lower surfaces of the plate along the side edges. The unsteady lumped-vortex model was used to obtain the unsteady fluid loads on the plate. This method accounts for the effect of tension in the plate and the longitudinal displacement of the trailing edge in the flow. Results found good correlation with literature only for large beam length values (Tang and Paidousis, 2006). The study also found that trailing edge wake has less of an influence on the stability of the fluid-structure interaction, when the beam length is large (Tang and Paidousis, 2007). Balint and Lucey considered a numerical analysis for two distinct conditions on a cantilevered uniform plate in axial flow. The first case considers flow on both the upper and lower surfaces of the plate, and produced results similar to the papers already presented above. The second case however considers flow only on one side of the plate. The analysis was done using a finite difference method to solve the Navier-Stokes equation. Their paper concludes that when there is flow on both sides of the plate, stability is first lost to the flutter mechanism. However, when there is only flow on one surface of the plate, instability is caused through a divergence mechanism (Balint and Lucey, 2005). In both cases, a critical flow velocity exists. It is evident that both the numerical and analytical approaches to flutter stability analysis are widely accepted. *This thesis will take a numerical approach in obtaining an analysis.*

**B. Aims**

This thesis aims to investigate the aeroelastic instability of a cantilevered beam in subsonic axial flow. It is expected that the instability onset will be attributed to flutter and hence this thesis will attempt to find the critical Reynolds number (Re) at which flutter instability occurs. This investigation will be attempted via a numerical simulation built in MATLAB, in accordance with the problem indentified earlier in this chapter. Furthermore, an in-depth understanding of the flow characteristics and equations as well as beam mechanics is expected upon completion of this thesis. This author wishes that this thesis will be a guide for future undergraduate students interested in fluid-structures interaction. Therefore, the methodology, derivation and code work-up used are detailed and fully provided.

Another aim of the thesis project is to instill valuable lessons learnt through research and project management into the undergraduates. These skills have proved invaluable in producing this thesis and have at times, ensured that the thesis was completed in a timely fashion. The management documents produced for the initial thesis report is attached in Appendix B.

**III. Methodology**

**A. Beam Mechanics**

1. **Basic Parameters**

The beam mechanics was obtained by first assuming a rectangular cross-sectional area, denoted by $A(x)$. The subscript $(x)$ indicates the position along the x-axis, from the origin. Figure 5 illustrates the beam cross-sectional area with the y-axis and z-axis denoted by the blue and red lines respectively. As indicated before, the Cartesian coordinate system used for defining the beam originates from the mid-point at the leading edge face of the beam. The dimensions of the cross-sectional area is 0.2m by 0.03m (width x height). Since $A(x)$ is modeled such that it does not vary with its position along the x-axis, the cross-sectional area of the beam is a fixed value and is now denoted by $A$.

A preliminary analysis of rotor blades was conducted and the results show that a long slender rotor blade is more likely to deform about the z-axis than the other two axes. Using simple calculations, it has been proven that the bending moment effects on a beam outweigh the shear force effects by at least a factor of 10. Without doing a fully worked calculation comparing the effects of torsion and bending, we first start by a simple
reasoning process to eliminate one of the two remaining forces. Since this problem is based on cantilevered rotor blades in axial flow. It is only logical to then accept that because conventional rotor blades are highly resistant to torsion, the effects of torsional forces on this problem are considerably reduced. Furthermore, the flow over the beam is predominantly axial and will therefore be more likely to set up an uneven force distribution across the length of the beam. In addition to that, the beam length to width ratio is calculated to be a value of 35. Statistically speaking, the probability that an uneven force distribution is set up across the width of the beam is therefore significantly reduced. This author concludes that even on the off chance that an uneven force distribution is set up over the width of the beam, the inherent resistance of the beam to torsion will negate any uneven force distributions. It is therefore realistic to model the numerical simulation of the beam only in one-dimension and restrict the nodal deformations to the y-axis. Given these stated assumptions, it follows that the cross-sectional area moment of inertia, \( I_{zz} \), is a fixed valued and can be found by Eq. (1).

\[
I_{zz} = \frac{bh^3}{12}
\]  

(1)

where \( b \) is the width and \( h \) is the height of the beam in meters. (Hibbeler, 2005)

The beam is also assumed to be made of a uniform material: stainless steel alloy 304. Even though conventional rotor blades are constructed using a variety of composites (glass epoxy), realistically modeling the variation in material properties along the length of the beam is expected to be complex and tedious. Furthermore, even if an effort was made to realistically model the material property changes, the potential benefit gained does not appear to outweigh the cost in time and effort. It was postulated that an equally accurate numerical model could be achieved, even if the beam was assumed to be a uniform material. The choice of stainless steel alloy 304 was simply because the material properties were readily available in the textbooks. The material properties for stainless steel alloy 304 can be found in Table 1.

The numerical simulation models the beam as cantilevered at the leading edge and free at the trailing edge, setting up a clamped-free beam condition. Because the leading edge is clamped, one would expect the displacement, \( w \), and slope \( \frac{dw}{dx} \) at the leading edge \( (x = 0) \), to be zero (Inman, 2001). These are the first two spatial boundary conditions of the beam. It is useful to point out that, \( w \), is defined as a distortion along the y-axis and is positive in the positive y-direction. Another note of possible contention is the terminology of this particular problem. As previously stated, the numerical simulation models the beam in only one-dimension, beam length. This would imply that the width of the beam is assumed to be infinite, changing the context of the problem from a beam to a plate. The author would like to clarify that despite the assumption that the beam width is infinite, classification of the problem as a cantilevered beam should not cause any confusion, as long as the set-up of the problem is clearly stated (as has been done here). For all extensive purposes, the terminology used in this thesis to describe the rotor blade shall remain as a beam.

2. Transverse Vibrations on a Cantilevered Beam

The numerical simulation assumes that the beam obeys the Euler-Bernoulli beam theory. This assumption is consistent throughout the majority of the literature, as seen in the literature review presented before (Tang and Paidousis, 2007). It also appears that several papers have achieved relatively good results with the Euler-Bernoulli beam theory. It seems both logical and reasonable that this thesis should also make this assumption. When obtaining the string equation shown below, the full derivation was conducted from initial principles so as to ensure that all assumptions made along the way were valid. Due to the nature of the derivation, some of the assumption validations had to be grouped in their relevant appendices. This derivation can be found in Appendix C. Having now obtained the full string equation, Eq. (2), we will use this determine the deflections along the beam (Inman, 2001 pp.460).

\[
\rho A \frac{\partial^2 w(x,t)}{\partial t^2} + C \left( \frac{\partial w(x,t)}{\partial t} \right) + \frac{\partial^2}{\partial x^2} \left[ EI \frac{\partial^2 w(x,t)}{\partial x^2} \right] = f(x,t)
\]  

(2)

where \( \rho \) denotes the density of the material, \( A \) is the cross-sectional area, \( C \) is the damping co-efficient, \( E \) is the modulus of elasticity of the material, \( I \) is the moment of inertia and \( f(x,t) \) is the distribution of the external forcing term.
3. Analytical Solution

Before the numerical simulation proceeds, it would be useful to first obtain an analytical solution of an undamped clamped-free beam undergoing free vibration. Free vibration implies that the forcing term is set to zero. If the analytical solution is completed with the appropriate material properties and initial conditions, the results should theoretically be more accurate than any numerical simulation. Since analytical solutions provide an exact solution for the deflection of the beam with respect to time, it will also be useful in determining the natural frequency and modes of the beam. This author hopes to achieve good correlating results between the analytical solution and the numerical simulation. Despite the fact that analytical solutions are exact, this approach cannot be taken when investigating the fluid-structure interaction over time. This is because the analytical solution requires the function of the forcing term to be known. In reality, the function of the forcing term on the beam is not known at every time step and tends to be tedious to determine. Furthermore, analytical solutions require a very precise mathematical structure before a well posed solution can be obtained, whereas numerical simulations overcome this difficulty by iterating numerous times to cut down on the error. This author chooses to proceed with the numerical simulation because it ultimately provides a more robust solution, despite the inherent errors with the method. The analytical solution is provided here to act as a yardstick for comparison.

In accordance with an undamped freely vibrating beam, both the damping coefficient and the forcing term are set to zero. We then assume a separation of variable solution of the form \( w(x,t) = X(x)T(t) \), and therefore Eq. (2) simplifies to Eq. (3) where \( \omega \) is the separation constant and \( c = \sqrt{\frac{EI}{\rho A}} \).

\[
c^2 \frac{X''}{X} = -\frac{T''}{T} = \omega^2 \tag{3}
\]

The derivation for the analytical solution obtained here is provided in Appendix D. Equation (4) is the characteristic equation for a clamped-free beam and is used to determine the weighted frequencies, \( \beta_nl \).

\[
\frac{-1}{\cos \beta l} = \cosh \beta l
\tag{4}
\]

The characteristic equation is graphed in Appendix D and illustrates how the weight frequencies are obtained. Table 2 provides the weighted frequencies for the first five mode shape as well as the respective natural frequencies of the undamped, clamped-free beam undergoing free vibration.

<table>
<thead>
<tr>
<th>Weighted Frequencies, ( \beta_nl )</th>
<th>Mode Shape, n</th>
<th>( \sigma_n )</th>
<th>( \omega_n ), Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.87510407</td>
<td>1</td>
<td>0.7341</td>
<td>3.05</td>
</tr>
<tr>
<td>4.69409113</td>
<td>2</td>
<td>1.0185</td>
<td>19.13</td>
</tr>
<tr>
<td>7.85475744</td>
<td>3</td>
<td>0.9992</td>
<td>53.56</td>
</tr>
<tr>
<td>10.9954073</td>
<td>4</td>
<td>1.0000</td>
<td>104.96</td>
</tr>
<tr>
<td>14.13716839</td>
<td>5</td>
<td>1.0000</td>
<td>173.50</td>
</tr>
</tbody>
</table>

Table 2. Weighted frequencies and natural frequencies

Equation (5) denotes the deflection shape of the beam for each mode shape, \( n = 1, 2, 3, \) etc

\[
X(x) = a_4 \left[ \cosh \beta_n x - \cos \beta_n x - \sigma_n \left( \sinh \beta_n x - \sin \beta_n x \right) \right]
\tag{5}
\]

The natural frequency for each mode shape is related to the weighted frequency by Eq. (6).

\[
\omega_n = \beta_n^2 \sqrt{\frac{EI}{\rho A}}
\tag{6}
\]

Equation (5) is graphed in Fig 6 to illustrate the different deflections of the cantilevered beam at varying mode shapes. It is evident that the higher the mode shapes, the larger the natural frequency and hence the shorter the period. This is also obvious from Table 2.
4. Finite Element Method

Now that we have discussed the analytical solution of the deflected beam, attention should be paid to the process of discretization of the beam. The beam was discretized using a finite element method, which assigns nodes to points along the beam. It follows that by tracking these nodes, a numerical solution will be able to determine the overall deflection of the beam over time. Consider a beam defined by a single element of length, \( l \), between two nodes (Node 1 and Node 2). The finite element method uses a coordinate system that denotes two linear coordinates (\( u_1 \) and \( u_3 \)) and two rotational coordinates (\( u_2 \) and \( u_4 \)). This implies that the finite element method models each node as having two degrees of freedom, one in the transverse direction and the other in the rotation direction. Figure 7 illustrates this point. The theory then stipulates that the transverse displacement must satisfy Eq. (7).

As was stated earlier, the values of \( E \) and \( I \) are constant as \( x \) varies along the length of the beam. This implies that Eq. (7) can be simply integrated four times to yield a general solution, Eq. (8).

\[
\frac{\partial^2}{\partial x^2} \left[ EI \frac{\partial^2 u_{(x,t)}}{\partial x^2} \right] = 0
\]  

(7)

where \( c_1, c_2, c_3 \) and \( c_4 \) are constants of integration. Applying the appropriate boundary conditions yields the four constants of integration. Substituting Eq. (8) into the equations for kinetic energy and strain energy yields the mass matrix, \( M \) and the stiffness matrix, \( K \). The derivation that determines these matrixes can be found in Appendix E.

The mass matrix, \( M \) and the stiffness matrix, \( K \), areas follows (Inman, 2001 pp.548-549):
Keeping in mind that these mass and stiffness matrices are only applicable if the beam was discretized with
one element, the numerical solution follows that when using a greater number of beam elements to describe the
deflection of the beam, a global mass and stiffness matrix must be obtained. If we now imagine that Fig 5 was
not a single element describing the entire length of the beam, but one element describing a small portion of the
beam, where \( l = dx \). Then we find that in order to describe the deflection of the entire beam, we simply need to
consider each single element individually and add the relevant joining ends appropriately. Here we assume that
the principle of superposition holds true. In physical terms, because the beam itself is a continuous material, this
assumption is a valid one and the discretization of the beam is merely a method of which we use to obtain the
overall beam deflection. When the entire beam is described with one element and two nodes, the size of the
mass and stiffness matrixes are 4 by 4. Going further, when the beam is described with two elements and 3
nodes, the size of the mass and stiffness matrixes are 6 by 6. This pattern holds true for however many nodes we
might want to use to describe the beam. We thus conclude that the size of both the mass and stiffness matrixes
are simply a symmetric matrix of size twice the amount of nodes used to describe the beam.

Assuming that the beam is discretized by three nodes, we therefore expect the global mass and stiffness
matrixes to be 6 by 6 in size. This following section shall obtain the global stiffness matrix for a 3 node
discretization. The detailed derivations that obtain the global stiffness matrix are provided in Appendix E. Only
the resultant global stiffness matrix is presented here in Eq. (11). For additional information, readers are
couraged to read the appendices and other textbooks for clarification.

\[
K_{\text{global}} = \frac{EI}{l^3} \begin{bmatrix}
12 & 6l & -12 & 6l & 0 & 0 \\
6l & 4l^2 & -6l & 2l^2 & 0 & 0 \\
-12 & -6l & 12+12 & -6l+6l & -12 & 6l \\
6l & 2l^2 & -6l+6l & 4l^2+4l^2 & -6l & 2l^2 \\
0 & 0 & 12 & -6l & 12 & -6l \\
0 & 0 & 6l & 2l^2 & -6l & 4l^2
\end{bmatrix}
\] (11)

Having obtained the global stiffness matrix for a 3 node cantilevered beam, the global mass matrix can also
be obtained in a similar fashion. From here, it is not a far stretch to then determine the global matrix for any
number of beam nodes. This numerical simulation models the beam with a total of 20 nodes. An official
optimization scheme of determining the ideal number of nodes to define the beam was not undertaken. Readers
should take note that, as always, this is a balance between accuracy and computational time required. Increasing
the number of nodes that define the beam will increase the accuracy of obtaining the beam deflection but will
also increase the amount of computational time required to obtain this solution. A balance must therefore be
struck between a minimum level of accuracy and a maximum time for a convergent solution. This author has set
the number of nodes along the beam to be 20 because it ensures a sufficiently accurate solution at an acceptable
computational time.

We are now ready to solve the string equation and determine the deflection of each node along the beam.
This numerical simulation achieves by using an ODE45 solver in MATLAB. ODE45 integrates the system of
differential equations, given a set of initial linear and rotational coordinates, for a specified time frame. ODE45
is considered one of the better solvers in MATLAB because of the variable time step it uses to reach its final value. The details of ODE45 will not be included here and can be easily found in any MATLAB help file. Consider a general case of a forced response on a damped linear system, Eq. (12).

\[ M \ddot{x} + C \dot{x} + K x = BF(x) \]  

(12)

where \( M \) is the global mass matrix, \( K \) is the global stiffness matrix, \( C \) is the global damping co-efficient, \( B \) is the co-efficient of the forcing term and \( F(x) \) is the global forcing term (Inman, 2001 pp.337). As stated before, the distortion of the beam is obtained using the ODE45 function, however, the global mass, damping and stiffness terms needs to be defined for the respective number of nodes in the beam. The functions which obtain these global matrixes are discussed in the Code Work-up section. Furthermore, the global forcing term acting on the beam is derived from the flow field around the beam. This forcing term is continuously updated for each iterative time step and a new function is fed to the ODE45 solver each time.

5. **Damping**

The original plan was to model the beam as being completely un-damped, thereby setting the damping co-efficient term, \( C \), to zero. After subsequent testing of the numerical simulation, this author decided that the damping co-efficient should be arbitrarily set at a value of 1 everywhere in the matrix. This was an attempt to model the stainless steel alloy 304 beam more accurately by artificially inducing damping into the beam mechanics.

6. **Code Work-up**

[beamconditions.m] is the function that determines the initial linear and rotational coordinates when time equals to 0, before the fluid is allowed to flow over the beam. As explained earlier, if the initial conditions of the beam are provided, ODE45 will obtain the final deflection of the beam for a given time frame. The initial conditions have been named, \( x0 \). These initial conditions include the linear and rotational coordinates at all nodes along the beam, as well as the time derivatives of these coordinates. This means that \( x0 \) defines four basic parameters at each node; linear deflection, rotational deflection, linear velocity and rotational velocity. At the initial condition, the linear and rotational coordinates are non-zero while all velocity terms are zero.

\[
\left( \frac{\partial u_1}{\partial t} = 0, \frac{\partial u_2}{\partial t} = 0 \right)
\]

The numerical simulation has been programmed to start the beam by initially displacing it downwards. The amount of deflection was obtained by simple static bending moments and shear force. The weight of the beam was calculated and a point load artificially induced at the centre of gravity of the beam. This point load is only applied at the initial condition and is not applied when the beam is interacting with the fluid. The weight of the beam is not accounted for in the fluid-structure interaction because the literature seems to suggest that the affect of this weight would be negligible. However, the initial ‘drooping’ condition of the beam is expected to initially generate the difference in pressure across the two surfaces of the beam. Subsequently, because of the fluid-structure interaction, a differential pressure distribution seems to be continually observed. The detailed calculations of the initial ‘droop’ condition can be found in Appendix F.

[beamnumericsolution.m] is the function that initiates the ODE45 solver. ODE45 requires a specific input code that will enable the solver to successfully call for the initial input variable, \( x0 \), as well as the input time frame, \( ts \). [beamnumericsolution.m] facilitates this process and ensures that ODE45 is functioning. This function also obtains and plots the deflection of the beam, at each iterative time step, when the iterative ODE45 process has been completed.

[f.m] is the input function that ODE45 calls when the global mass and stiffness matrixes are required. As mentioned before, ODE45 requires a very specific input syntax before it will run successfully. This implies that whilst [f.m] is the function that provides the input parameters, the syntax of [f.m] must not vary too much. For example, ODE45 must call [f.m] and will not work if [f.m] is renamed to allow for an updating of input parameters. In the function [f.m], the global mass and stiffness matrix is generated according to the number of nodes in the beam. In general, the number of nodes in the beam will not change from iteration to iteration, hence there is no real need to update these two parameters. However, because [f.m] also inputs the forcing term into the ODE45 solver, it is absolutely pertinent that the forcing term is continuously updated at each iterative step. This author has gotten around this difficulty by saving these two input parameters as a *.mat file at each
iterative time step. Subsequently, the \textit{[f.m]} function loads the *.mat file and updates these relevant input parameters.

**B. Fluid Mechanics**

1. **Basic Parameters**

   As stated earlier, this numerical simulation only evaluates a two-dimensional flow field at each iterative step. This was intentionally designed because the beam model is only one-dimensional. If the flow field was also designed to be one-dimensional, this author expects that the flow parameters at each grid point would be inaccurate to capture the flow over the beam. Keeping in mind that the real flow over the rotor blades is three-dimensional, each time a dimension is ignored by the flow field model, the simulation becomes progressively more inaccurate. Intuitively speaking, a one-dimensional flow field will clearly not be accurate enough to produce the differential pressure distribution over the beam. The flow field is also not three-dimensional because of the inherent complexity in adding the third dimension. The most obvious benefit in investigating this problem in three-dimensions would be obtaining the vortex shedding effects off the sides of the beam, when presented in axial flow. At the crux of this benefit is the assumption that the width of the beam is finite. However, this thesis models the beam in only one-dimension, inherently implying that the width of the beam is infinite. Clearly modeling the flow field in anything more than two-dimensions would only be adding complexity to the coding and not reaping any additional benefits. Furthermore, keeping the code as simple as possible is also a priority in the view of the author.

   The flow field is assumed to be axial along the boundaries because this problem is modeled when the helicopter is in forward/cruise flight. Even though there might be local variations of the direction of the flow due to the presence of the helicopter, for the purpose of simplicity, these variations are ignored. The assumption of only axial flow is only set along the four boundaries of the flow field grid. The assumption here is that the flow region far upstream and far downstream of the beam has returned to laminar axial flow. Likewise, the flow field is entirely axial along the top and bottom boundaries of the flow field grid. At all grid points within these boundaries, the flow is allowed to vary in the two-dimensional plane. These variations are captured in the simulation and their effects propagated throughout the flow field. This author points out that the no-slip condition applies along the boundaries of the beam. These boundaries are modeled as a wall and therefore the flow velocity in all directions are assumed to be zero. Implied in this assumption is that the flow is also not allowed to flow into, out of or pass these boundaries. This assumption is validated in the literature. (Munson, Young and Okiishi, 2006 pp.14)

   The beam model is centred in the middle of the flow field grid because the flow parameters both upstream and downstream of the beam affect the flow above and below the beam. This assumption is validated because the flow speeds investigated are considered subsonic; therein implying that “information” in the flow is able to propagate both upstream and downstream in the flow field. This author believes that centering the beam model in the flow field will produce results of greater accuracy. The dimensions of the flow field grid have also been chosen for a specific purpose. This investigation is primarily focused on ascertaining the pressure distribution over the beam so that the next iterative deflection of the beam can be obtained. Given that the flow both above and below the beam is affected by the flow upstream and downstream, the numerical simulation therefore accounts for sufficient allowances in both the upstream and downstream direction. Herein lies the balance that this author has taken when designing the dimensions of the flow field grid. If the numerical simulation accounted for the flow field far upstream and far downstream, whilst still maintaining the same grid point density ratio within the flow field, the amount of grid points required to capture the flow and therefore the computational time required will increase significantly. This author believes that 20m of allowance in all directions is sufficient to achieve accurate results.

   As stated earlier, the flow modeled to be subsonic, incompressible, viscous, laminar and unsteady. The numerical simulation was built to model subsonic flow speeds because this author believes these results are more pertinent to the overall design than any other flow speeds. Despite the fact that the mathematical behavior of partial differential equations governing subsonic flow is considered to be more unstable and inherently more difficult to code, any body intending to experience transonic or supersonic flow must first encounter subsonic flow. It is therefore logical to first examine the stability of the beam in subsonic flow before moving on to transonic and supersonic flow regimes. The flow is assumed to be incompressible because at Mach number values of 0.3 and less, air flow can be assumed to be relatively incompressible. Earlier in this thesis, the ZB500 Lynx helicopter was quoted as having a top speed of 92.5 m/s.
Mach number is given as:

\[ M = \frac{V}{\sqrt{\gamma RT}} \]  

where \( \gamma \) is the ratio of specific heat, \( R \) is the gas constant and \( T \) is the temperature (Anderson, 2005 pp.160).

Using the values provided in Table 1, and assuming the top speed of the Lynx helicopter yields a Mach number of 0.29. Figure 8 illustrates the variations of density with Mach number and clearly shows that for a Mach number below 0.3 the effects of compressibility can be neglected. Since this thesis is investigating the instability effects of the rotor beams in the helicopter transformed configuration, 92.5 m/s is taken as the maximum flow speed value. These results validate the incompressible flow assumption used in the numerical simulation.

In an attempt to model the real-life flow realistically, the viscous effects of the incompressible, subsonic flow are captured at each grid point. Viscous flow is analogous with flow within a boundary layer and this was precisely why this numerical simulation models a viscous fluid. As will be obvious in the recommendations of this thesis, an inviscid flow field outside of the boundary layer was also intended to be coded and added onto this simulation. It should be stated that the present numerical simulation models a fluid-structure interaction which is immersed in an unrealistically thick boundary layer. Similarly, the numerical simulation models a fully laminar flow field. This is done because the flow field, without the inserted beam, is initially fully laminar. After inserting the beam, we find that the flow is still laminar, at least until the local Reynolds number increases sufficiently that the flow transits to a turbulent one. Even in commercially available software, the turbulence in a flow is only approximated by turbulence models because of the difficulty in capturing turbulence. A Spalart-Allmaras turbulence model was intended to be coded as well but was not completed due to time constrains. Likewise, as the local Reynolds number and Mach number increases, the assumption of incompressibility is not true everywhere in the flow field. However, in order to ensure that the resultant code is not overly complex, this author has assumed that the flow is laminar and incompressible everywhere in the modeled flow field. It is clear that this numerical simulation is merely the first building block required to fully model this fluid-structure interaction.

Lastly, the flow is modeled as unsteady and is expected to change with time. This characteristic of the flow seems obvious since the flow parameters not only changes with space but also with time. It is, however, even more pertinent to assume an unsteady flow field because the pressure distribution over the beam at each time step is required in order to further distort the beam. If there, indeed, is a steady state solution for this interaction, then results from this numerical simulation should find this solution when the total time is taken towards infinity.

2. Nature of Partial Differential Equations

When dealing with the Navier-Stokes equations, it is impossible not to work with partial differential equations (PDEs). In fact, even when severe assumptions and limitations are placed upon the flow to simplify these equations, the remaining components from these equations still contain partial differentials. In addition to this, because these partial differential equations are then solved by an iterative process, the solutions of these equations take on the mathematical behavior of the partial differentials. It then becomes important to understand the mathematical behavior of these partial differential equations to fully understand the solutions of the Navier-Stokes equations. It should be pointed out that by taking various assumptions of the flow, the Navier-Stokes equations are therein simplified and therefore the remaining partial differential components take on a different mathematical behavior.
When discussing the impact of partial differential equations on computational fluid dynamics, there are generally three broad categories in which these partial differentials can be classified in: hyperbolic, parabolic, elliptic. There are times when the partial differential equations exhibit a mixed behavior of the above three mentioned categories, and this will be made clear in the following explanation. Since a solution commonly takes on the mathematical behavior of the partial differential equations the solution was derived from, it becomes pertinent that the solution obtained is not due to the mathematical behavior of these equations but accurately represents the flow parameters in reality. In addition to this, the different categories of partial differentials also impact on the overall stability of the eventual solution. Because of its mathematical behavior, the solution for certain flows are inherently more unstable than others, and therefore require special methods for obtaining the flow parameters. Herein lays the crux of the matter, when investigating a flow given a set of assumptions and limitation, the nature of the partial differential equations must be taken into account when determining the most ideal method for solving them.

Hyperbolic equations define the boundaries in which the solution is dependent on. In three dimensions, hyperbolic equations define a cone where every point downstream of the cone is dependent on the flow parameters at the tip of the cone. Parabolic equations also define the boundaries in which the solution is dependent on. In three dimensions, parabolic equations define a rectangular volume where points downstream of the flow are dependent on points upstream of the flow, within the rectangular boundaries. These two categories of partial differential equations lend themselves to marching solutions, whereby points downstream of the flow can be determined if points upstream of the flow are known quantities. Elliptic equations find that there are no boundaries where the solution is dependent on. Instead, in elliptic equations, the solution at a specific point in three-dimensional space is affected by all other points in that space. The opposite is also true, in that a change in the same specific point will affect all other points in that space. This implies that the solutions for elliptic equations must be obtained through solving simultaneously the equations at all points.

A thorough investigation into the partial differential equations is outside the scope of this thesis and these lengthy discussions can easily be found in most advanced mathematical textbooks. This section of the thesis serves only to introduce the partial differential equations applicable to the flow field under investigation. Given the assumptions already stated, it was found that the flow exhibited a mixed elliptical-parabolic mathematical behavior. Unfortunately this implies that the mathematical behavior was more prone to be unstable and tends to produce a divergent solution. The pressure correction technique is a method that deals with this particular type of mathematical behavior and is the technique that will be used to obtain a solution to this problem.

3. Taylor Series Expansion

The Taylor series expansion is one of the most common methods of achieving discretization in the flow field. This allows the replacement of partial derivatives with a suitable algebraic difference quotient, therein representing these partial derivatives with a finite-difference equivalent. The Taylor series expansion can be commonly found in most mathematical textbooks and will not be further explained here. The following section will only provide the finite-difference equivalent used in coding this numerical simulation.

\[
\frac{\partial f_{(i)}}{\partial x} = \frac{f_{(i)} - f_{(i-1)}}{\Delta x} + 0(\Delta x) \tag{14}
\]

\[
\frac{\partial f_{(i)}}{\partial x} = \frac{f_{(i+1)} - f_{(i)}}{\Delta x} + 0(\Delta x) \tag{15}
\]

\[
\frac{\partial^2 f_{(i)}}{\partial x^2} = \frac{f_{(i+1)} - 2f_{(i)} + f_{(i-1)}}{\Delta x^2} + 0(\Delta x^2) \tag{16}
\]

\[
\frac{\partial^2 f_{(i)}}{\partial x^2} = \frac{f_{(i+2)} - 2f_{(i+1)} + f_{(i)}}{\Delta x^2} + 0(\Delta x) \tag{17}
\]

\[
\frac{\partial^2 f_{(i)}}{\partial x^2} = \frac{f_{(i-2)} - 2f_{(i-1)} + f_{(i)}}{\Delta x^2} + 0(\Delta x) \tag{18}
\]
where \( f(i) \) is the value of \( f \) at the relative position \( i \), \( f(i+1) \) is the value of \( f \) at the relative position \( i+1 \) and \( \Delta x \) is the distance between these two positions in the \( x \)-direction. Taylor series expansions are considered an approximation because additional terms are assumed to have only a small impact on the result and are therefore neglected. For example, Eq. (14) is a first order accurate Taylor series expansion, which implies that all remaining terms which are multiplied by \( \Delta x \) or a higher order of \( \Delta x \) are neglected. Physically this means that the influence at grid points outside of \( (i) \) and \( (i+1) \) are ignored in the evaluation of \( f \) at grid point \( (i) \). This is a potential source of errors in the simulation and is termed as truncation errors. Obviously having a higher order representation of the partial derivative makes the solution more accurate, but will in turn add complexity and computational effort to the simulation. This numerical simulation being the first building block, this author feels that it is most appropriate to start simple and add complexity in later stages. Therefore, most of the Taylor series expansion representatives are only first order accurate.

Equations (14) and (15) are both used to represent a single partial derivative and are both first order accurate. Equation (14) is a backward difference and Eq. (15) is a forward difference. The remaining three equations are all used to replace second order partial derivatives where Eq. (16) is considered second order accurate. Equation (16) is a central difference, Eq. (17) is a forward difference and Eq. (18) is a backward difference. The direction of these replacements (i.e. forward, backward or central) determines which grid points are taken into account when evaluating the partial derivative. A different direction is chosen when there are constrains to the grid points in the flow field. For example, when evaluating a grid point along the boundary, there is obviously no data outside the flow field boundary that can be taken into account. Therefore a central difference replacement is not appropriate at this grid point and a forward or backward difference is used.

4. Pressure Correction Technique

The pressure correction technique was developed by Patankar and Spalding (Patankar and Spalding, 1972, pp.1787) and is embodied in an algorithm called Semi-Implicit Method for Pressure-Linked Equations (SIMPLE). For a two-dimensional flow field and where the fluid is assumed to be subsonic, incompressible, viscous, laminar and unsteady, the Navier-Stokes equations simplify to the following equations.

\[
\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0
\]

\[
\frac{\partial (\rho u)}{\partial t} + \frac{\partial (\rho u^2)}{\partial x} + \frac{\partial (\rho uv)}{\partial y} = -\frac{\partial p}{\partial x} + \mu \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)
\]

\[
\frac{\partial (\rho v)}{\partial t} + \frac{\partial (\rho v^2)}{\partial y} + \frac{\partial (\rho vu)}{\partial x} = -\frac{\partial p}{\partial x} + \mu \left( \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial x^2} \right)
\]

where \( u \) is the local velocity component in the \( x \)-axis, \( v \) is the local velocity component in the \( y \)-axis, \( \rho \) is the density of the fluid, \( p \) is the local pressure and \( \mu \) is the dynamic viscosity of the fluid. Note that both the density and dynamic viscosity of the fluid, and therein local temperature, are assumed to be constant. Equation (19) is derived from the continuity equation which ensures that no mass is lost or created and Eqs. (20) and (21) are the respective \( x \) and \( y \) components of the momentum equations. Because of the assumptions made to the flow field, the energy equation is completely decoupled from the continuity and momentum equations. This simplifies the code further and also provides some computational relief.

It was previously explained that the pressure correction technique needed to be implemented because of the mixed elliptic-parabolic nature of the mathematical equations. This section of the will validate the decision to choose the pressure correction technique. In computational fluid dynamics, one of the most straight forward and effective iterative method is the MacCormack time-marching technique. This technique has shown itself to be extremely useful when applied to both parabolic and hyperbolic partial differential equations. However, when applied to a mixed elliptic-parabolic problem, we find that the stability condition for an explicit solution when applying the MacCormack technique is as follows (Anderson, Dale, Tannehill and Pletcher, 1984):
\[
\Delta t \leq \frac{1}{\frac{|u|}{\Delta x} + \frac{|v|}{\Delta y} + a \sqrt{\frac{1}{(\Delta x)^2} + \frac{1}{(\Delta y)^2}}}
\]

(21a)

where \(\Delta t\) is the time step between each iteration and \(a\) is the speed of sound. Here we find that because the fluid is assumed to be incompressible, the value of the speed of sound is theoretically infinite, hence yielding a stability value for the time step to be zero. Stability conditions will be further discussed in a later section of this thesis. Furthermore, it has been found that even when a compressible fluid is analyzed using the MacCormack technique, the solution requires an exceedingly long amount of time to converge for low Mach numbers of 0.2 and below. These Mach numbers are exactly in the region of which this thesis wishes to investigate and is therefore problematic for the author. The pressure correction technique has produced good results for compressible flow at these Mach numbers and even better results for incompressible flows. This author is fairly confident that this will be the ideal approach to obtain a solution.

The pressure correction technique is a fairly complex technique to obtain the flow solution. It essentially combines a space-marching method with a time-march method and produces a steady solution for the flow field. This may seem like a contradiction to the above stated condition that unsteady flow fields were necessary for obtaining the fluid-structure interaction. This apparent contradiction shall be explained here. The pressure correction technique obtains a steady state solution for a given time step and therefore a pressure distribution over the beam is determined. The total time taken for the pressure correction technique to converge to a solution is denoted as the total time and is equal to the summation of each time step. In real terms, the flow has proceeded into the future by the total time and this is the first iteration. The forcing term from this pressure distribution is then imported into the beam model and the beam is deflected accordingly. The beam has now been deflected over the value given in the total time. The deflected beam is then re-modeled in the flow and the pressure correction technique obtains a new pressure distribution over the beam. This cyclic process continues on for however many iterations required. It is therefore possible, if the time step for the pressure correction technique is small enough, that when the solution finds a convergent solution at the first iteration, the total time taken would also be small enough for the fluid-structure coupling to be considered unsteady. It this explanation seems exceedingly complicated and there will be a more in-depth discussion in a later chapter.

The pressure correction technique is outlined step-by-step in the following:

i. The following procedure is applicable only to the internal grid points of the flow field grid. At points along the boundary, a separate procedure is used.

ii. Assign each grid point a value of \(u^*, v^*\) and \(p^*\). The * implies that these values of velocity or pressure is a guess value and is not the true value at that grid point. It should be noted that these values of velocity and pressure are not calculated at the same point in space. The pressure correction technique requires the use of a staggered grid system whereby \(u, v\) and \(p\) are all captured and calculated at separate grid points in space. A further explanation of this staggered grid system is provided in a subsequent chapter. All these values are assumed to exist at the initial condition, where time equals zero. In order to keep track of this, we denote these values as \(u^N\) where \(N\) corresponds to the current time step. The same is applied to both \(v\) and \(p\).

iii. Equations (19) to (21) are replaced by the Taylor series expansion to obtain the finite difference equations of (24), (25) and (32). These equations are provided at the conclusion of this section. Equations (24) and (25) are both derived from the momentum equations in the \(x\) and \(y\) direction respectively. These equations have components which are both spatial as well as temporal, enabling this author to then calculate the values of \(u\) and \(v\) for each grid point one time step into the future. Therefore, both \(u^{N+1}\) and \(v^{N+1}\) can be determined.

iv. These values of \(u\) and \(v\) (at time step \(N+1\)) are not the correct values at each grid point because the values of \(u\) and \(v\) at the previous time step (\(N\)) were not necessarily the correct values. Therefore, it becomes necessary to obtain an equation that checks and verifies when the values of \(u\) and \(v\) are actually correct. This is obtained when we substitute Eqs. (20) and (21) into Eq. (19). This is effectively forcing the values of \(u\) and \(v\) obtained through the momentum equations to also adhere to the continuity equation. If time is spent analyzing this step, this author finds that ensuring that both the momentum and continuity equations are true is only the logical step to take. Failing which implies that there is either mass being loss or produced in the flow field. Equation (32) is effectively checking to ensure that mass continuity is true at all internal grid points in the flow field. When this is true, then the values of \(u\) and \(v\) at all these points must be accurate.
v. Equation (32) yields the corrected pressure at each internal grid point, $p'$. Notice that the corrected pressure, $p'$, is dependent on its neighboring grid points and can be solved by using a relaxation technique. This relaxation technique will be explained in the next section. In Eq. (32), we find that there is a term denoted as $d$ and it is called the mass source term. The corrected pressure is initially assumed to be zero at all points in the grid. This relaxation technique will iteratively obtain the values of $p'$ at all internal grid points given the following formula:

$$p'_{T+1} = p^{T'} + \alpha p^T$$

(22)

where $T$ denotes the next guess level for $p$, $p''$ is the new guess of $p'$ and $\omega$ is the relaxation factor used. Again, an in-depth discussion of relaxation techniques is provided in the next section. An error margin is determined by the difference between $p'$ at the next guess level and the current guess level. This error margin is present at each individual grid point within the flow field. The relaxation technique will iteratively proceed until this error margin decreases to zero at all internal grid points. This implies that the next guess level of $p'$ is in fact the same as the current guess level of $p'$. Naturally when this convergent solution is achieved, the corrected pressure at all points will not be zero. Readers are reminded not to confuse the iterative guesses used here with the iterative time steps of the pressure correction technique. Furthermore, once the pressure correction technique reaches a convergent solution (total time = summation of all time steps taken), the iteration number for the fluid-structure interaction simulation is only considered to have advanced by one.

vi. Here the pressure at each point in the internal grid is now evaluated and obtained. This is determined by the following equation:

$$p^{N+1} = (p^*)^N + \alpha p'$$

(23)

where $\alpha$ is the relaxation factor. Equation (23) obtains the new pressure at each grid point for a single time step in the future.

vii. These new values of pressure, $p^{N+1}$, are now used to solve for new values of $u$ and $v$ for another time step into the future and the process returns to step (iii) and repeated until the maximum mass source term convergences to zero at all internal grid points. The mass source term is considered an indication when the pressure correction technique has reached a convergence solution. It is indicative of when the continuity equation is satisfied and is therefore zero when a steady state solution has been achieved. Readers are reminded here that this is only the convergent solution for the current iteration level and a single convergent pressure correction solution will only advance the iteration by one.

This author would like to pause here and clarify that there are three iterative processes that have been described. Each time a convergent pressure correction solution is obtained, the iterative number is advanced by one. This provides the pressure distribution over the beam so that the beam model can be distorted. Within each pressure correction solution, the corrected pressures are iteratively calculated until the mass source term approaches zero. In addition to this, each time a maximum mass source term is obtained, the corrected pressure at all internal grid points undergoes an iterative next-guess cycle to obtain a convergent $p'$ solution. This is the third iterative process.

As stated earlier, when replacing the continuity and momentum equations with finite difference expressions, the mathematical calculations yield the following:

$$\left(\rho u^*\right)_{i+0.5,j}^{N+1} = \left(\rho u^*\right)_{i+0.5,j}^N + A\Delta t \frac{\Delta t}{\Delta x} \left(p_{i+1,j}^{N+1} - p_{i,j}^{N+1}\right)$$

(24)

$$\left(\rho v^*\right)_{i,j+0.5}^{N+1} = \left(\rho v^*\right)_{i,j+0.5}^N + B\Delta t \frac{\Delta t}{\Delta y} \left(p_{i+1,j}^{N+1} - p_{i,j}^{N+1}\right)$$

(25)
\[
A = \left[ \frac{\rho u^* \nu^*_{i+1.5,j} - (\rho u^* \nu^*)_{j-0.5,i,j}}{2\Delta x} + \frac{(\rho u^* \nu^*_{j+0.5,j+1})}{2\Delta y} \right] \\
+ \mu \left[ \frac{u^*_{i+1.5,j}-2u^*_{i+0.5,j}+u^*_{i-0.5,j}}{(\Delta x)^2} \right]^N + \left[ \frac{u^*_{j+0.5,j+1}-2u^*_{j+0.5,j}+u^*_{j-0.5,j}}{(\Delta y)^2} \right]^N \right]^{(26)}
\]

where

\[
\nu^*_{i+0.5,j+1} = \frac{1}{2}(v^*_{i,j+0.5} + v^*_{i+1,j+0.5})^N \text{ and } \nu^*_{i+0.5,j-1} = \frac{1}{2}(v^*_{i,j-0.5} + v^*_{i+1,j-0.5})^N \quad (27) \text{ & } (28)
\]

\[
B = \left[ \frac{\rho v^* \nu^*_{i+1.5,j} - (\rho v^* \nu^*)_{j-0.5,i,j}}{2\Delta y} + \frac{(\rho v^* \nu^*_{j+0.5,j+1})}{2\Delta x} \right] \\
+ \mu \left[ \frac{v^*_{i+1.5,j+0.5}-2v^*_{i+0.5,j}+v^*_{i-1.5,j+0.5}}{(\Delta x)^2} \right]^N + \left[ \frac{v^*_{j+1.5,j+1}-2v^*_{j+0.5,j}+u^*_{j+1,j-1.5}}{(\Delta y)^2} \right]^N \right]^{(29)}
\]

where

\[
\nu^*_{i+1.5,j+0.5} = \frac{1}{2}(u^*_{i+0.5,j} + u^*_{i+0.5,j+1})^N \text{ and } \nu^*_{i-1.5,j+0.5} = \frac{1}{2}(u^*_{i-0.5,j} + u^*_{i-0.5,j+1})^N \quad (30) \text{ & } (31)
\]

By substituting Eqs. (24) and (25) into the continuity equation, the following equation is obtained:

\[
a(p^*_{i,j}) = b(p^*_{i+1,j} + p^*_{i-1,j}) + c(p^*_{i,j+1} + p^*_{i,j-1}) + d
\]

where

\[
a = 2\Delta t \left( \frac{\Delta y_p}{\Delta x_p} + \frac{\Delta x_p}{\Delta y_p} \right), \quad b = \Delta t \left( \frac{\Delta y_p}{\Delta x_p} \right), \quad c = \Delta t \left( \frac{\Delta x_p}{\Delta y_p} \right) \quad (33), (34) \text{ & } (35)
\]

\[
d = \Delta y_p \left( \rho u^*_{i-0.5,j} - \rho u^*_{i+0.5,j} \right)^N + \Delta x_p \left( \rho v^*_{i,j-0.5} - \rho v^*_{i,j+0.5} \right)^N \quad (36)
\]

where \(\Delta y_p\) is the vertical distance between two adjacent grid points that capture \(v\), \(\Delta x_p\) is the vertical distance between two adjacent grid points that capture \(u\), \(\Delta x_p\) is the horizontal distance between two adjacent grid points that capture \(p\). Equations (32) has been derived in this manner because the grid points in the flow field are not regularly spaced apart. Therefore, there is a difference between adjacent grid points in both the x-direction as well as the y-direction. Furthermore, because of the staggered grid required to use the pressure correction technique, grid points which capture and calculate \(u\), \(v\) and \(p\) are all different. These two points are the reasons why the equations provided above are slightly different than conventional equations used for the pressure correction technique. The author would also like to remind the reader that these equations only apply to all internal grid points and are not true for the boundaries. Note: boundaries are not only the outer edges of the flow field but also the surface of the beam, which is considered a wall. For clarification, it should be said that the equations above have notations that advance half a grid square in both the x and y direction because of the staggered grid used. Equation (24) applies the momentum equation in the x-direction about the grid point \((i+0.5,j)\). Equation (25) applies the
momentum equation in the y-direction about the grid point \((i,j+0.5)\). For further clarification, refer to the figures provided below.

Equations (24), (25), (26) and (29) are illustrated in Figs (9) and (10) shown below. These equations calculate the average respective component velocities from neighboring points for every internal grid point. This author has found it difficult to explain the notation in words and will therefore refer readers to Figs (9) and (10) for clarification.

5. Relaxation Technique

The relaxation technique is a finite difference method suited for solving elliptic partial differential equation. In an earlier section of this thesis stated that a relaxation technique would be used to obtain a convergent solution. Equation (22) in step \((v)\) of the pressure correction technique described above uses a point-iterative relaxation method to come to a convergent solution. The following describes this relaxation technique utilized in the numerical simulation. The following equation is used to determine a convergent solution for \(p'\).

\[
P'^{T+1} = p'^{T} + \omega p'^{T}
\]

where

\[
\left( p'^{T}_{i,j} \right) = \frac{b\left(p'_{i+1,j} + p'_{i-1,j} \right) + cp'_{i,j+1} + cp'_{i,j-1} + d}{a} \tag{37}
\]

and

\[
\left( p'^{T}_{i,j} \right) = \frac{b\left(p'_{i+1,j} + p'_{i-1,j} \right) + cp'_{i,j+1} + cp'_{i,j-1} + d}{a} - \left( p'^{T}_{i,j} \right) \tag{38}
\]

The omega term is called the relaxation factor and acts to either speed up or slow down the convergence. A slow speed of convergence helps prevent a divergent solution but requires additional computing power and requires more time before obtaining a solution. An omega value of one and below \((\omega < 1)\) is considered an under relaxation and is generally slower to obtain a solution. Conversely, a faster rate of convergence will take up less computing time but has a greater tendency for diverging. These correspond to omega values of greater than one \((\omega > 1)\). The range of values for omega ranges from \(0 < \omega < 2\). It was found through experimental optimization that the most appropriate value of omega was 1.2. This value has been subsequently used in all the numerical simulations produced in this thesis. Furthermore, the relaxation factor, \(\alpha\), which aids in obtaining a convergent solution in the mass source term is optimized at a value of 0.8. Values of alpha which were above 1.5 were found to be unstable and produced suspicious results. Likewise, values of alpha which were below 0.4 were deemed to be too slow in obtaining a convergent solution. Therefore, for the purpose of all remaining numerical simulations, the relaxation factor \(\alpha\) will remain at 0.8.
6. Staggered Grid

The staggered grid is required, when using the pressure correction technique on incompressible flows, because of a mathematical abnormality which produces a convergent solution that is impossible in reality. This impossible flow field is described as the checkerboard flow field and is illustrated in Fig 11. Figure 11 illustrates a regularly spaced grid and defines the boundary for a flow field. The values at each grid point are given and it is obvious that such a pressure field is impossible to achieve in real life. However, when applying the pressure correction technique, results have shown that a pressure field of this nature will satisfy the continuity equation. If compressibility effects had been taken into account, this type of pressure field would have been wiped out early on in the iterative process. In addition to this, the literature shows that this type of checkerboard abnormality is not only specific to the pressure field in an incompressible solution, but is also prevalent in both the velocity components too. Therein implying that if a velocity field had a checkerboard pattern, it would also satisfy the continuity equation. Looking at the mathematics of the pressure correction technique, it was determined that by using the central difference Taylor series expansion, these type of checkerboard patterns were not eradicated from the solution but were propagated instead. It was for these reasons that the staggered grid modification was added to this technique.

Figure 12 shows the regular grid used in this numerical simulation before the grid points were staggered. The grey circles are grid points where local flow pressure is captured and evaluated. These points are termed as the pressure grid points. The green vertical line denotes the y-axis and the red horizontal line denotes the x-axis. An example of the Cartesian coordinate system for the two-dimensional flow field grid is also included in the figure. The next figure, Fig 13, illustrates an enlarged version of the staggered grid. This grid was used in the computational calculations of this numerical simulation. The white circles represent the grid points at which the velocity components, $u$ and $v$ are captured. The author wishes to point out that the horizontal velocity component, $u$, is only captured at white circles which are immediately to the left and right of the pressure grid points. Likewise, the vertical velocity component, $v$, is only captured at white circles which are immediately above and below all pressure grid points. It quickly becomes obvious that when applying the staggered grid pressure correction technique, an effective method of bookkeeping is pertinent. This numerical simulation achieves this by running a MATLAB function called expand.m which expands the regularly spaced pressure grid to the staggered grid shown in Fig 13. The velocity component grids are then easily obtained from the newly staggered pressure grid. The important parameter that must be kept constant is the Cartesian coordinate system that marries up the two different grids. This coordinate system can be clearly seen in the enlarged Fig 13.
7. Boundary Conditions

The weakness in any numerical simulation solving the Navier-Stokes equations for a finite grid is at the boundaries of the flow field. Because of the finite difference method used to replace the partial differential equation, grid points at the boundaries find themselves unable to draw on data points outside the finite boundary of the flow field. In computational terms, this is literally due to the fact that everywhere outside the pre-defined grid does not exist. The simulation can continuously expand the grid size to accommodate the boundaries, but this will only re-define the boundary positions and the same problem is still faced at these new boundary locations. There are several methods of getting around this problem and one such method is termed the ghost point method. The ghost point method simply mirrors points outside the grid with points inside the grid. This allows the boundary points to be calculated and the solution can converge. This however induces an error into the simulation since grid points outside the boundary can at times be drastically different from points within the boundary. For example, consider moving from grid point to grid point in a certain direction and assume that we come to a boundary. Assume further that the partial differential equation under investigation is only spatial. Note that partial derivate terms are essentially nothing more than a gradient term from grid point to its adjacent grid point. At the boundary, if the partial derivative term is positive across the boundary, we expect the next grid point outside of the boundary to have a higher absolute value than the boundary. However, if the ghost point method is used and the point outside the boundary is mirrored with the point inside the boundary, we find that the solution is an averaging eventuality. This may just be an approximate solution and is certainly valid in its approach, but when dealing with drastic changes across the boundary, this method will certainly produce an increasing error margin.

Another method to determine the boundary conditions is to extrapolate from grid points within the boundary. This method seems logical and has proven to produce reasonable results at the boundary. However, if the gradient term changes drastically towards the boundary, then this method will also produce erroneous results. The last solution to evaluating points at the boundary is by changing the Taylor series expansion used to replace the partial derivatives. In the previous section, equations were provided that evaluated the flow field grid at all internal grid points. This was achieved by using central difference Taylor series expansion equations to replace the partial derivatives. This is considered the more accurate approach because not only is a central difference used; implying that the final solution is not skewed in any single direction, but it is also accurate to a second order. This numerical simulation is coded to use a central difference finite difference replacement equation, at all internal grid points, precisely for this reason. Now at the boundary locations, these central difference equations are no longer sufficient and are therefore changed to either a forward or backward difference scheme. This change allows points at the boundary to be evaluated and are calculated by taken into account only points that are within the grid boundary. This is arguably similar to the extrapolation method and indeed the two methods produce similar results. However, because this method preserves the original mathematics of the flow field, it is considered to be more accurate than simply extrapolating. It is, however, more tedious to code and thereby adds complexity to the simulation.

Coming back to the specifics of this numerical simulation, the boundary conditions assigned in this flow field were specifically chosen because it allowed for simple coding but also because it modeled the flow realistically. The inflow and outflow boundary conditions were modeled with a known pressure and horizontal velocity component. The vertical velocity component was set to zero. Likewise, the top and bottom boundary condition are exactly the same. These boundary conditions are further assumed to be steady and do not change with respect to time. This was done assuming that these grid points were far enough away from the beam, such that the beam would have no effect on the flow in these regions. As stated earlier, the distance of the boundaries to the beam is approximately 20m, which in the opinion of the author, is sufficient enough for this assumption. By not evaluating the grid points along the outer boundaries, the code is significantly simplified since a common central difference scheme can be applied at almost all the internal grid points.

The wall boundary conditions along the surface of the beam apply the no-slip condition. The local pressure is allowed to float, keeping both the velocity components at zero. When evaluating the grid points along the surface of the beam, the numerical simulation a forward difference for the upper surface and a backward difference for the bottom surface of the beam. Not only does this improve the accuracy of the solution (as opposed to using a ghost point method), it also provides a stabilizing affect since only using a central difference scheme propagates an incorrect checker board type solution.

8. Implicit time stepping vs Explicit time stepping

The decision between implicit time stepping and explicit time stepping is a balance between the added complexity in coding and the benefits gained in adding that complexity. Explicit time stepping predicts values in the future by using only current values and then stepping through time in an iterative process while implicit time
stepping predicts the future values by using future values. It is perhaps intuitive that the implicit time stepping scheme will involve additional complexity. However, the benefits gained from using an implicit time stepping scheme can sometimes outweigh the difficulty in coding associated with it.

In general, the explicit scheme has an inherent stability condition on the size of the time step used. In fact, this condition limits the size of the time step, causing the simulation to run for a large number of iterations before a minimum total time has elapsed. Explicit time stepping is significantly easier to code and is generally used for problems that want to investigate flows over a short total time frame. Truly unsteady flows fall into this category because extremely turbulent flows require an extremely small time step to fully capture the flow anyway. However, if the simulation was required to, say, obtain the flow field at a large time in the future. Using an explicit time stepping scheme would mean that the simulation would have to run for an impossibly large number of iterations and required an impossible amount of time. Conversely, implicit time stepping schemes have a large margin for stability and permits the use of larger time steps. This allows the simulation to run further ahead into the future and results have found these simulations have also found good convergence in their solutions. It then seems obvious that an implicit time stepping scheme is used when steady-state solutions are under investigations or when the unsteady flow has relatively low time dependence. As already mentioned, the main draw back in using an implicit time stepping scheme is the added complexity to the coding process.

The pressure correction technique is implemented through a SIMPLE framework which, as stated in the title, is semi-implicit. This author feels that the added complexity to the coding process is worth the benefit gained in the stability condition. Furthermore, given the problem under investigation, an unsteady analysis of the fluid-structure interaction only seems more appropriate. Even if the steady state solution of the interaction proved to be stable; it is the short term transient response of the rotor blades that would be of interest. Therein implying that a short time step would be taken anyway and supporting the decision that an implicit time stepping scheme is more apt. In addition to this, the mixed elliptic-parabolic behavior of the governing mathematical equations was inherently unsuited to a straight forward time-marching scheme. Therefore, this author concludes that the pressure correction technique seemed like the ideal approach to take.

9. Optimization of Convergent Solution

As mentioned earlier on, the mass source term is considered indicative of the progression of a solution and when this term tends to zero, a convergent solution is assumed to have been reached. Unfortunately, despite the continual and repetitive efforts of this author, the mass source term only appears to converge to zero after an unusually long computational time. When the numerical simulation is run, the mass source term is continually being printed so that the user can keep track of its value. According to the literature available, this mass source term is expected to have an oscillatory pattern before it eventually converges to zero. By converging to zero, it implies that the continuity equation is finally satisfied and the resultant solution is accurate. However, this takes an exceedingly long time to occur in the numerical simulation, much longer than the literature advises it should take. Instead, the oscillatory nature of the mass source value is observed to have an average value of approximately 1. This simulation has been exhaustively debugged but still no reason can be found as to why this occurs.

This author postulates that because the mass source term is dependant on the distance between individual grid points, if insufficient grid points are taken within the grid, the distance between each point approaches a critical value where the flow solution becomes divergent. This seems reasonable given that if only a few grid points are used to define the flow field, then it is also reasonable to expect that the flow parameters between adjacent grid points is drastically different. It is then, this large gradient value, which eventually prevents the solution from converging quickly enough. By increasing the number of grid points within the flow field, the distance between each point is effectively being decreased and there is no longer a larger jump in local parameters between adjacent grid points. After doubling the number of grid point in the flow, the result shows a decrease in the mass source term by a factor of 10. This result was initially welcomed because it implies that the calculated flow field was closer to satisfying the continuity equation and the hope was that it would converge onto zero faster. However, the overall trend of the mass source term continued to fluctuate around the same average value and still took a long time to converge towards zero. There is a limit as to how many grid points can be taken in the flow. As the number of grid points increases, so does the computational time required. Increasing the grid points seem to decrease the initial value of the mass source term, but if the resultant increase in grid points further extends the computational time required, then it is not beneficial to the overall solution.

In order to obtain the pressure distribution over the beam, this author has modified the code so that despite a fluctuating mass source term, an intermediate pressure distribution over the beam is obtained in a reasonable time frame and used to analyse the fluid-structure interaction. This modified code serves to track the value of
the mass source term. As each subsequent mass source term is generated, it is first compared to the previous value and the ratio between the two obtained. This ratio is termed as the mass ratio. The code then tracks this ratio and ensures that it does not reach a critical value of 2. This value was obtained over numerous runs of the simulation, which all seemed to show that while the mass ratio remained below 2, the mass source term would have an average value of approximately 1. If the mass ratio reaches 2 or more, this implies that the subsequent mass source term is two times or more than the current mass source value. Since the further away the mass source value is from zero represents an increase in the error, the code is programmed to stop the iteration at this point and use the previous flow parameters to determine the pressure distribution over the beam.

Furthermore, the code is also programmed to stop the iterative process if the mass source value drops below 0.2. It was found that for certain flow speeds, the mass source term would converge to a value close enough to zero in an acceptable time frame. Therefore, this author has chosen 0.2 to be the cut off value in the mass source term. Even though 0.2 is not zero, this author believes that the solution would have sufficiently converged and the flow field accurate enough for our purposes. Despite the mass source term not converging to zero, this modified method ensures that the best possibly flow field parameters are used in the analysis and only the most accurate iteration is used to produce the differential pressure distribution over the beam. This author considers this to be acceptable given that the mass source value fluctuates around a constant value and only converges to zero after an unacceptably long time period.

The code that provides this modification can be found in the MATLAB function [subsonic.m].

10. Code Work-up

Most of the code that deals with solving the Navier-Stokes equations for the flow field is enclosed within [subsonic.m]. The code is sufficiently commented so that the reader can easily follow the thought process and observe the design of the various functions. The code is attached in Appendix G and can also be found in the attached CDROM.

[expand.m] is a function expands the pressure and velocity component grids to a staggered grid. It also expands the matrices which track the Cartesian coordinate system for each grid point in the flow field. This allows the simulation to track the exact position of each grid point and determine the vertical and horizontal distance between each point and its neighboring points. This is important to the analysis of the problem and will be covered in a later chapter of this thesis. Suffice to say, because grid is not regularly spaced, when solving the Navier-Stokes equations the Cartesian coordinates of each grid point become important.

[boundcondition.m] is a function that serves to assign the boundary conditions to the flow field grid. These boundary conditions have already been explained earlier in this thesis.

[newUV.m] is a function that obtains the values of $u$ and $v$ at all points in the flow field grid. As far as possible, the internal grid points are evaluated using a central difference scheme so that the solution maintains a high level of accuracy. However, at the boundary points, $u$ and $v$ are evaluated using a forward or backward scheme, whichever is more appropriate according to the respective boundary locations.

[correctedP.m] is a function that obtains the corrected pressure values and its corresponding mass source term at all grid points within the flow field. Again, a central difference scheme is used where possible, and a forward or backward scheme used when the boundaries are evaluated.

[plotcontour.m] is a function that simply plots the pressure contours and velocity vector diagrams when the flow has converged to a sufficiently accurate solution. Because the grid has been staggered for the calculations, this function serves to re-compress the grid and returns them to their original form. It then obtains the pressure and velocity plots for the flow field and deflected beam.

[forces.m] is a function that can be found in [master.m], and serves to determine the distributed forcing term on the surface of the beam. This forcing term changes at each iterative interval and is therefore required to be continuously updated.

C. Coupling

The insertion of the beam model into the flow field has already been illustrated in Fig 4. The basic concept of first obtaining the distributed forcing term on the beam from the fluid mechanics and then importing this into the beam mechanics to obtain a deflection has also been introduced. This section of the thesis serves to provide the information that syncs the fluid mechanics to the beam mechanics. It seems obvious that the one common
parameter that can be used to syncs both mechanics is time. The flow field solution is first allowed to converge, given a certain time step; where by the total time taken is then the summation of all the time steps taken to obtain this solution. This total time is then used as an input parameter to determine the beam deflection through the ODE45 function. Once the new beam deflection has been obtained, the flow is once again allowed to find a convergent solution over the newly deflected beam. This process is repeated for as many iterative cycles required. In this fashion, the numerical simulation has effectively coupled the fluid mechanics with the beam mechanics, thereby ensuring that the fluid-structure interactions run concurrently into the future.

As stated earlier, the pressure correction technique is a method that produces a steady state solution for a given flow field. However, if the time step is sufficiently small, this steady state solution can be considered an unsteady flow over the beam. This author postulates that the reaction time scale for a fluid is several magnitudes higher than that for a solid. This implies that a fluid will change its flow parameters faster than a solid will take to distort. This is largely due to the high molecular forces in a solid which resist changes or distortions to the solid lattice structure. In fact, sufficient energy must be provided before distortion can occur. In a fluid, where the intermolecular forces are significantly weaker than in a solid, it can be reasonably assumed that any changes in the flow will have less resistive forces opposing it. Hence, even if the pressure correction technique provides a steady flow solution, if the total time taken is still small enough, the solution can be considered unsteady when compared to the distortion time scale of a solid beam.

\[ \text{[master.m]} \]
This is the main function of the numerical simulation. Nested in each of the loops is a single line that syncs the fluid mechanics to the beam mechanics using time as the common parameter.

D. Grid

1. Compressed Grid

Figure 4 was presented as an introduction to this problem. In reality, this numerical simulation uses a compressed grid to capture the flow field. This compressed grid is illustrated in Fig 14. The reason why the grid is compressed towards the beam is simply because this is where the flow field is of greatest interest. Since this problem is focused on determining the differential pressure distribution on the surface of the beam, it is only logical that the grid points are concentrated nearer the surface of the beam. Figure 14 illustrates this compression for a flow field with 20 grid points in the horizontal direction and 15 grid points in the vertical direction. In the vertical direction, the beam itself takes up one row of grid points. This is the reason why there is an odd number of grid points in the vertical direction. The code is written such that the number of grid points upstream and downstream of the beam can be individually changed. Likewise, the number of grid points above and below the beam can also be tailored. However, it is advised that an equal number of grid points above and below the beam are used. In the following results presented, the flow field grid was set with 80 grid points in the horizontal direction and 61 grid points in the vertical direction. This grid point density was finally settled on after optimizing the increase in flow capturing accuracy whilst maintaining an acceptable amount of computing time.

Furthermore, there is an increase in grid points both above and below the beam. This is simply because the flow in these regions is of particular interest to this investigation. Therefore, for the same grid density, if the grid points are concentrated where the flow is of interest, the simulation will require the same amount of computing time but produce more accurate results.

The grid points are compressed using the following equations:

\[ \eta = \ln(y + 1) \quad \text{and} \quad \xi = x \]  

(39) & (40)
where $\eta$ is the new y-axis and $\zeta$ corresponds to the x-axis (Anderson, 1995 pp.187). Note that there is no intentional compression in the x-axis and the compression in the y-axis is simply to concentrate the grid points nearer to the beam. Since there was no other constraints in the flow field, a simple natural log function was employed to achieve this compression.

The numerical simulation was optimized by using 80 grid points in the horizontal direction and 61 grid points in the vertical direction. As mentioned earlier, the mass source term was postulated to be unstable if too few grid points were used to capture the flow field. However, if too many grid points were used, this would inevitably result in an unacceptably long computational time required to obtain the solution. It was therefore for these reasons that this set of grid points were selected for the numerical simulations. In addition to using this specific grid point density ratio, because the grid points are concentrated at regions where the flow field is of particular interest, the results obtained are inherently more accurate for the same amount of computational time required.

2. De-Conflicting Beam Nodes with the Flow Field Grid Points

When coding this numerical simulation, it became increasingly obvious that there had to be some method of de-conflicting the grid points designated to model the beam nodes and the grid points designated to capture the flow field. Since the beam was expected to deflect in the flow, it was postulated that if the grid points that evaluated the flow did not move as well, there would be a highly probably chance that the simulation would produce results that indicated fluid flowing through the solid beam. Just as likely, results where flow field grid points coincided with beam nodal points were to be expected. Hence, the numerical simulation was written such that the flow field grid points are newly generated at each iterative step. This implies that the flow field grid points actually moves with the deflected beam and if these grid points were to be plotted against time, the results would show that they distort in a similar manner as the beam. When considering the physical aspect of this scenario, this author finds that this is a realistic manner in which the actual fluid will be moved by the deflected beam. The difficulty here was to track the flow parameters of the flow field grid point before and after it was moved by the beam. This author could not find a consistent solution to this problem and opted to simply move the flow field grid points and then re-evaluate the flow field from its boundary conditions. This seemed to solve the problem of conflicting beam nodes and flow field grid points in the same Cartesian space.

3. Reasons for not using a Regularly Spaced Grid

In the majority of the literature and textbooks reviewed, most of the computational fluid dynamic codes use a regularly spaced grid to capture the flow. This allows for simpler coding since the values of $\Delta x$ and $\Delta y$ will be constant across the grid. However, in this particular problem because the beam was expected to continuously deflect in the flow field, there had to be some method of de-conflicting the grid points designated to capture the flow with the grid points designated to model the beam nodes. Eventually, this author managed to overcome this problem by redefining the positions of the grid points capturing the flow each time the beam was deflected. This meant that the grid points that evaluated the flow field in the current iterative step would not be the same grid points used in the subsequent iterative step. Therefore, if the compressed grid points were converted to regularly spaced ones, this conversion would have to be done at each iterative step. Furthermore, because the deflection of the beam in the future was not yet known, the mathematics required for the grid conversion became impossibly difficult to predict.

Most conventional computational fluid dynamic schemes use a regularly spaced grid because the points are fixed in the flow field. However, since the pressure correction technique obtains a steady state solution, this author postulated that with an exceedingly small time step, it might be sufficient to assume that the flow parameters within the boundaries would always only depend on initial the boundary conditions. Therefore, the only change made to the flow field would be the newly distorted beam position. This seemed to imply that the relative positions of the grid points in the flow field did not matter too much on the accuracy of the solution. This author concludes that using a regularly spaced grid is purely a conventional practice and, in this problem, would not be of any benefit whatsoever. It is with these reasons, that this author has chosen to calculate the distance between each grid point, at each iterative step, whenever the Navier-Stokes equations are evaluated in the flow field. This is the why the Cartesian coordinate (in both the x and y directions) for all grid points have to be tracked for each iterative step (Roache, 1998 pp.220).

4. Code Work-up

[initialgrid.m] is a function that is only run once in the numerical simulation and serves to produce the initial x and y coordinates for all grid points. This function calculates the initial y coordinates of each beam node, accounting for the ‘droop’ of the beam.
[gridgeneration.m] is a function that serves to move the flow field grid points for each iteration, so that the beam nodes are de-conflicted with the grid points that evaluate the flow field.

E. Operating Instructions

The code is relatively easy and straightforward to use, especially if the reader is a seasoned MATLAB user. A brief version of the operating instruction will be provided here for readers that have not used MATLAB before. First ensure that all the *.m files are in the same working directory that MATLAB is using. The main code is termed [master.m] and this is the only function file that should be opened. Within [master.m] all the variables that define this problem are clearly commented and can be easily changed to suit the user. The more pertinent variables are the values of omega and alpha which denote the relaxation factors used in the program. To vary the flow speeds, change the values of U_ini and U_bound to the required flow speed. The number of grid points used can be easily changed in the GRID GENERATION INPUTS section. Lastly, the value of del_t is the time step used when determining each iterative value of d, the mass source term. Generally, the smaller this value is the more stable the simulation becomes.

Once all inputs parameters are set, the function is initiated by typing master() into the MATLAB command prompt. The simulation is programmed to run the given input parameters for one iterative step before prompting the user if additional iterative steps are required. This is the check phase and is used to tailor the remaining inputs accordingly. In this first iterative step, the program will produce all the default plots, namely: initial grid plot, beam deflection plot, lift variation plot, pressure contours, velocity vector plot, beam convergence plot, beam distortion plot with respect to beam origin and the grid generated for the next iterative step. The program will now prompt the user to specify the number of additional iterations required, and if every one of the plots stated above are required to be displayed again. An input variable of 1 is required to indicate if that specific plot should be plotted and no response is required if that plot is not needed to be graphed. This author advises that readers should leave all input entries blank and allow the program to run several more iterations. The pressure contour and velocity vector plots are the two more interesting plots to investigate.

IV. Results and Analysis

The ultimate aim of this thesis was to investigate the aeroelastic instability of a cantilevered rotor blade in axial flow. This was done by building a numerical simulation in MATLAB which models the unsteady fluid-structure interaction between the beam and the fluid. The following figures show the results obtained from this simulation, after 0.546 seconds have elapsed and a total number of 7 iterations have been run. The initial boundary conditions of the flow are as follows: \( u = 5 \text{ m/s}, P = P_{\text{atm}}, \nu = 0 \text{ m/s}, \omega = 1.2 \) and \( \alpha = 0.1 \). Figure 15 shows the beam deflection for each iterative time step on the same graph. The legend clearly shows which curve was the initial starting beam deflection and which is the current beam deflection. It would appear that the scale of the resulting beam deflection is realistic and within a reasonably expected range. Furthermore, the deflection of the beam seems to be in the positive y-axis. This is an expected result since the initial beam deflection was the ‘drooped’ condition.
One would therefore expect the flow to produce a region of high pressure below the beam and a region of low pressure above the beam, thereby producing an upward forcing term on the surface of the beam. Figure 16 shows the pressure contours of the beam at this instant. From Fig 16, a region of local high pressure seems to be growing at the leading edge of the beam while a region of low pressure seems to be concentrating at the trailing edge. Figure 17 is an enlarged velocity vector plot of the beam at this instant in time. It should be noted that the velocity components form streamlines which flow both above and below the beam. At no point, does the numerical simulation allow fluid to flow into and out of the beam. This seems to show that the no-slip condition and the wall boundary condition are true in this simulation. The flow field also seems to bend around the deflected beam and is somewhat distorted due to the presence of the beam in the flow. This is an expected result since the governing equations were mixed elliptic-parabolic, implying that information could be effectively transmitted both upstream and downstream in the flow field. Regardless of these observations, Fig 17 also shows that along the surface of the beam, the velocity component is zero (no-slip condition). However, because the boundary layer of the flow field was not effectively captured, this figure does not show the growing velocity profile commonly seen in boundary layers. Figure 18 is also an enlarged version of the beam in this flow. This figure shows the grid points used for this particular numerical simulation. It is evident that the grid points were compressed towards the beam, in an attempt to capture the flow parameters within the region. Note that because the flow field grid points are continually being shifted by the deflection of the beam, there are no conflicting grid points where the beam nodal points and the flow field grid points coincide.

Figure 19 shows the total lift force on the surface of the beam and its variation with time. The total lift force is obtained by the summation of the forcing term found on the beam. This author expects the lift variation to have some sort of sinusoidal response with time, but because 0.546s is not long enough to see this response, further iterations will have to be run. The overall lift on the beam does seem to be decreasing towards zero. This again, is an expected result as the originally deflected beam is being ‘straighten’ by the flow field. This is consistent with Fig 15, in that the deflection of the beam is approaching the y = 20m line. The next figure, Fig 20, is an indication of the stability condition of the beam mechanics. This figure plots the variation with time of each individual node in the beam model. If the result figure is a converging graph, then the resultant beam deflection is expected to be at an increasing rate. However, if Fig 20 is found to be converging, then the subsequent beam deflection is found to be at a decreasing rate. Therefore, Fig 20 is considered to be an indication of the rate of change in deflection of the beam, at each iterative time step.
The last figure in this set, Fig 21, also shows the beam deflection but is graphed from the perspective of the beam model. Therefore, it is much easier to determine the deflection of each node in the beam, since it can be readily read off Fig 21. This is termed as the beam distortion plot and only shows the beam distortion at the current iterative time step.

As mentioned in the Operating Instructions section of this thesis, the numerical simulation is programmed to plot all these figures for the very first iterative step. After the user has entered the relevant flow parameters and has fully defined the problem, this author advises that the program be allowed to run several iterative time steps without continually graphing these figures. The program will prompt the user for an input decision, as to whether these plots should be graphed at each and every time step. From experience, this author finds that graphing these plots at all time steps will heavily take up computing resources and slow the entire simulation down. Therefore, it is advised to first run the simulations for the required amount of iterations with graphing these figures, and then run a final iteration where a selection of these figures can be graphed.

The following section of this chapter will look at verification of the numerical simulation, specially scrutinizing the mathematical logic behind the coding of this simulation.

The most obvious indication of an incorrect numerical simulation is if there are areas in the mathematical process which are expected to have a certain response but instead do not follow this expectation. One such indication would be the mass source term used to determine when the continuity equation has been satisfied. This numerical simulation finds that the mass source term does follow the expected trend, where by its value tends towards zero as the number of iterations tends towards infinity. The governing flow equations are considered to be accurate when the mass source term is zero. However, as stated before in the explanation, due to the long time it takes for this value to converge to zero, this author has incorporated a cut off value whereby the simulation assumes that the flow is accurate enough and continues on with the iterative time stepping. This author believes that the justifications for this cut off value have been clearly shown in previous chapters and should not overly compromise the accuracy of the simulation. Furthermore, on a code verification note, because the mass source term tends towards a converging value of zero, this should show that the mathematics of the simulation is accurate and correct. Despite the fact that it takes a while before convergence is achieved, the agreeable trend of the mass source term should support the claim that this simulation is corrected verified.
The following figures have been produced in FLUENT, a commercial computational fluid dynamic software. As far as possible, the dimensions of the flow field grid and beam model are similar to that used in this numerical simulation. Furthermore, the flow parameters in FLUENT have also been set up to be similar to those used in this simulation. Because FLUENT does not account for the dynamic response of the beam in the fluid, variations in the two results are to be expected. However, this author is attempting to compare the flow field over a similar beam, to see if the flow field captured by this numerical simulation is accurate and can be then validated. Because the SIMPLE algorithm is a semi-implicit steady state solution, two different types of flows have been selected in FLUENT to be compared to these results. Figure 22 shows the pressure contours of a steady flow after one iteration in FLUENT. Figure 23 shows the same scenario but with an unsteady flow assumption.

These results are compared against the pressure contour obtained from this numerical simulation after one iteration. Figure 24 shows this plot. There seems to be a significant correlation between the three pressure contour plots in that a region of high pressure and low pressure is formed. It should be noted that the unsteady flow in FLUENT was obtained after 20 corrective iterations for a single time step and the steady flow converged to this solution after 13 corrective iterations. The pressure contour plot generated in Fig 24, was obtained after 25 corrective iteration. This author concludes that this comparison is insufficient to reasonably prove that the numerical simulation is validated.
The choice to only compare results after a single iterative time step was intentional because the transient response of the flow was important. However, because this comparison was inconclusive, perhaps a steady solution comparison would be more appropriate. Both the FLUENT models were run for 200 iterations and the results did not show much of a difference than with the initial flow fields already presented above. Therefore, this author suggests a comparison of the lift variation over 200 iterative time steps. Figure 25 plots the lift variation on the static beam for 200 iterative time steps in steady flow. Figure 26 plots the lift variation for half the time frame but assumes an unsteady flow. The unsteady flow is only run for half the time period because the resultant lift variation is a constant with respect to time.

Figure 27 is the corresponding lift variation graph obtained from this numerical simulation. We observe that all three graphs are expectedly markedly different. Both Figs 25 and 26 are based on a static beam which cannot be deflected in the flow. The lift variation from the steady flow shows a steady state response to time which appears to be gradually damped out as the iteration progresses. This same steady state response is also observed in the lift variation plot derived from this numerical simulation. The steady state response also seems to be gradually damped out with more iteration. However, in addition to this steady state oscillatory response, Fig 27 also shows a short term fluctuation in the lift variation. As the iteration proceeds into time, the frequency of this transient oscillatory response seems to be increasing. This is not seen in either Fig 25 or 26. This author postulates that this might be the coupling effect of the fluid-structure interaction and therefore requires sufficient time to build up momentum. The y-axis in Fig 25 and 26 may seem to be several magnitudes out from the one depicted in Fig 27. However, this is because Fig 25 and 26 are plotting the lift co-efficient value against time/iteration number.
Corresponding to the current iterative time step of the simulation shown in Fig 27, Fig 28 shows the beam deflection history with respect to time. It should be pointed out that the deflection of the beam matches up with the corresponding lift on the beam at the specific iterative time step. Figure 28 seems to imply that a steady oscillatory response should be expected if the simulation was left to iterate further into the future. The beam deflection is also analogous to the first mode shape of an analytical solution on cantilevered beams in free vibration. Figure 29 is taken from a paper by Balint and Lucey who conducted investigations on the stability of a flexible cantilevered plate in viscous channel (Balint and Lucey, 2005). This author has found that the results produced from this numerical simulation show some decent trends with published literature and commercial available fluid analysis software. However, it is perhaps still premature to say that this numerical simulation can be validated for the flow field presently being investigated.

The aeroelastic instability of the beam is now ready to be determined, assuming that this numerical simulation passes both the verification and validation test. This author had planned to test the numerical simulation by running the simulation for subsonic flow speeds in increasing increments up to 95 m/s. This would generate a graph whereby the critical velocity at which aeroelastic instability occurs can be determined. Unfortunately, due to time constrains, this part of the analysis was not yet completed.

V. Conclusions

This author set out to design and build a numerical simulation in MATLAB that would solve the Navier-Stoke equations as well as Euler-Bernoulli’s beam theory. This simulation was successfully built and the results presented in the earlier chapter. This simulation provided the fluid-structure interaction that would make an aeroelastic instability analysis on a beam possible. However, before used for analysis, this simulation was first optimized, verified and validated.

The comparisons presented in the previous chapter of this thesis have shown that the results obtained from this numerical simulation have a good chance of being validated. Additional work is required to fully validate this simulation and exhaustively test the code. This author believes that, within the given time frame, this simulation has been optimized to the best of the author’s ability and the results show relatively good co-relation to commercial software like FLUENT. In fact, this numerical simulation is in some ways better than the FLUENT software because the beam is modeled as a dynamic structure. Therefore the fluid-structure interaction can truly be captured and investigated.

Furthermore, because the numerical simulation was built from scratch in MATLAB, the process has taught the author numerous lessons and has ensured that the concepts and fundamental lessons have all been picked up. The author believes that this may not have been achieved if a ‘black box’ program had been used to obtain the numerical simulation. In coding the simulation from scratch, it forced this author is go back to the initial assumptions for each equation and individually verify them so that the final solution was certain to be accurate. Even though a conventional uncertainty analysis was not conducted in this thesis, the process of validating the individual assumptions and then debugging the final code made use of several, if not all, of the skills important
in conducting an uncertainty analysis. In some ways, the step-by-step validations of the assumptions used in this code can also be considered an uncertainty analysis.

Due to time constrains, the aeroelastic instability analysis was not able to be conducted using this numerical simulation.

When the numerical simulation was completed, it has becoming blatantly obvious that the realm of computational fluid dynamics is both vast as it is complicated. However, this author believes that all tertiary education should be challenging and should aim to push students to achieve something outside their comfort zone. This was one of the main reasons why this author decided to undertake this task of building a fluid-structure interaction in MATLAB from scratch.

VI. Recommendations

Given the limited time frame of the academic year, this author was unable to complete certain portions of this thesis that was initially planned to be accomplished. Specifically speaking, the damping matrix of the beam needs to be modeled accurately and not arbitrarily set as was done in this investigation. Furthermore, a study into the added mass phenomenon due to the fluid flow over the beam should be conducted to see if this will affect the damping term in the beam model. Also, turbulence modeling was initially expected to be completed by the end of this investigation. This author postulated that using a Spalart-Allmaras turbulence model would be the most apt approach to take, given that it is a single-equation model and should be easily incorporated into the simulation. This could be an avenue for future work to be done. Furthermore, a more realistic boundary layer flow can be simulated over the distorting beam, within which the flow is modeled to be viscous. Outside the boundary layer, the flow should be modeled as inviscid (given the flow speeds under investigation).

In addition to this, flow speeds in the transonic and supersonic regimes can also be investigated and a stability analysis obtained for the cantilevered beam undergoing such a fluid-structure interaction. Also, in most conventional helicopters, the rotor blades are simply connected at the rotor hub. This implies a pin joint at the leading edge and no moment bearing capability. This scenario could also be investigated.

Acknowledgements

This author wishes to acknowledge the support of the University of New South Wales at the Australian Defence Force Academy, in particular Dr John Milthorpe and Mr Murat Tahtali for their guidance and support. Special mention must be given to family and friends who have continually supported this author in his academic endeavor.

References


Yahoo Website for movie reviews. http://movies.yahoo.com/movie/1800420003/details accessed on 23 Apr 08