Lab 2: Circulation around a 2-D Airfoil Measured with a Five-Hole Probe

Gregory Golonka, Connor Hack, and Timothy Welch

March 2024

Abstract

This lab report dives into the complexities of airfoil lift through a series of experiments. Throughout the lab, several tests were conducted to measure the velocity contour around a NACA 0015 airfoil, at different angles of attack and freestream speeds. The velocities of the contour were calculated using a five-hole probe, which was calibrated using the 'non-null' method. Using the contour map, the Kutta-Joukowski theorem was applied to calculate the lift of the airfoil in each scenario. Ultimately, the lift of the airfoil at 5 ° ranged between 5.77 and 16.35 N/m for the two speeds, and the lift of the airfoil at 10 ° was calculated to be 10.41 N/m at the lower speed. Additionally, the percent error in lift between all of these trials ranged from 1.5% being the lowest to a 7.4% error. Furthermore, the non-dimensionalized coefficients of lift ranged from 0.6 to 1.1 with percent errors ranging from 17.4% to 24.3%, and the investigation of lift across the different angles of attack and speed ranges revealed that the change in freestream velocity had a greater impact on lift than the change in angle of attack.

1 Introduction

The goal of this lab was to use the Kutta-Joukowski theorem to calculate the lift of a symmetric airfoil using measurements of circulations. To accomplish this, the lab was split into three distinct parts: calibration of the pressure transducers, calibration of the five-hole probe, and measuring the velocity about a contour around a NACA 0015 airfoil. The machine responsible for running this experiment was a "turbine-type blower" wind tunnel, found in Hessert Laboratory and was used to provide subsonic incompressible flow into the test section. Finally, to calculate lift, the Kutta-Joukowski theorem states that the force acting per unit length on a body can be defined using the equation

$$L' = \rho U \Gamma_a \tag{1}$$

where ρ is the freestream density, U is the freestream velocity, and Γ is the aerodynamic circulation around that body, positive in the clockwise direction. The circulation, Γ, is then further defined by the equation

$$\Gamma = \oint_S \vec{u} \cdot ds \tag{2}$$

where $\vec{u}$ is the velocity vector at a given point on the contour and ds is a differential element of that contour's boundary.

2 Experimental Setup

2.1 Define Ambient Conditions

Much of the later analysis relies on the ambient conditions of the laboratory, making it important to measure those conditions. The ambient pressure and temperature were obtained from a Fisher Scientific Traceable digital barometer within the testing room. This data was taken at both the beginning and end of the lab duration. The temperature and pressure data can be used to calculate the air density with the Ideal Gas Law given in the equation

$$\rho = \frac{P}{RT} \tag{3}$$

where ρ is the density, P is the ambient pressure, R is the specific gas constant of air taken to be 287 $\frac{\text{J}}{\text{kg}\cdot\text{K}}$, and T is the temperature. The values of P, T, and ρ taken and calculated for both times, as well as their averages, are tabulated in Table 1.

Table 1: Ambient conditions of the lab before and after the experiments.

Time	Temperature, T (K)	Pressure, P (Pa)	Density, ρ ($\frac{\text{kg}}{\text{m}^3}$)
4:47pm	298.37	9.96E+04	1.12
7:36pm	298.65	9.93E+04	1.16
Average	298.51 ± 0.6	9.95E+04 $\pm$ 400	1.12 ± 0.01

The uncertainties in these measurements are important to the later analysis. The uncertainties in the average pressures, temperature, and density were calculated by propagating component uncertainties in quadrature. The component uncertainties from the above-stated measurement device are listed in Table 2.

Table 2: Error of pressure and temperature measurements.

Source	Pressure, ϵ_P (Pa)	Temperature, ϵ_T (K)
Reading	100	0.1
Systematic	400	0.4
Drift	100	0.4

The uncertainty in density was calculated from the quadrature addition of these uncertainties via the equation

$$\epsilon_\rho = \sqrt{\left(\frac{\epsilon_P}{RT}\right)^2 + \left(\frac{-\epsilon_T P}{T^2}\right)^2} \tag{4}$$

where ϵ_ρ is the uncertainty in density, ϵ_P is the uncertainty in pressure, and ϵ_T is the uncertainty in temperature. This was the same process used in Lab 1: Transducer Calibration & Pitot Wake Profiles.

2.2 Preparing Experiment

Before the experiment could be conducted, several steps had to be taken to ensure it ran smoothly. First, it was necessary to connect the pressure transducers to the input channels in the correct order such that pressure transducer 1 was affixed to input channel 1 and so on. It is important to note here that the tubes should be connected at the beginning of the experiment to the pitot probe within the wind tunnel. Moreover, it is important for the first two parts of the lab that the vertical stepper motor be connected to the motor port of the step motor driver.

For the first portion of the experiment, the pitot probe within the test section must be able to pitch upwards and yaw. To do this, the rod to which it is affixed must be able to rotate and swing upwards. Within the testing setup, the foam at the top of the wind tunnel, which is in place to prevent air from escaping from the top of the tunnel, must leave enough clearance for the rod to move unhindered about the lateral axis. Likewise, to enable the rod to move about the lateral axis, the pulley system at the top of the test section must be affixed such that the set screw within is screwed into the flat portion of the affixed rod. Finally, the portion of the experimental setup that holds the rod in place must be loosened enough such that the rod can turn to allow the pitot probe to yaw without being restricted.

A brief discussion of the wind tunnel used in this lab is also relevant. The lab was conducted using a turbine-type wind tunnel found in the Hessert Laboratories on the University of Notre Dame campus. Flow velocity was controlled within the tunnel by inputting a percentage of the maximum motor speed. The schematic for the wind tunnel can be found in Figure 1. The schematic for the wind tunnel comes from the Lab 2 Circulation around a 2-D Airfoil Measured with a Five-Hole Probe lab handout [1]. The schematic helps with a visual understanding of the discussion in Section 5.1.

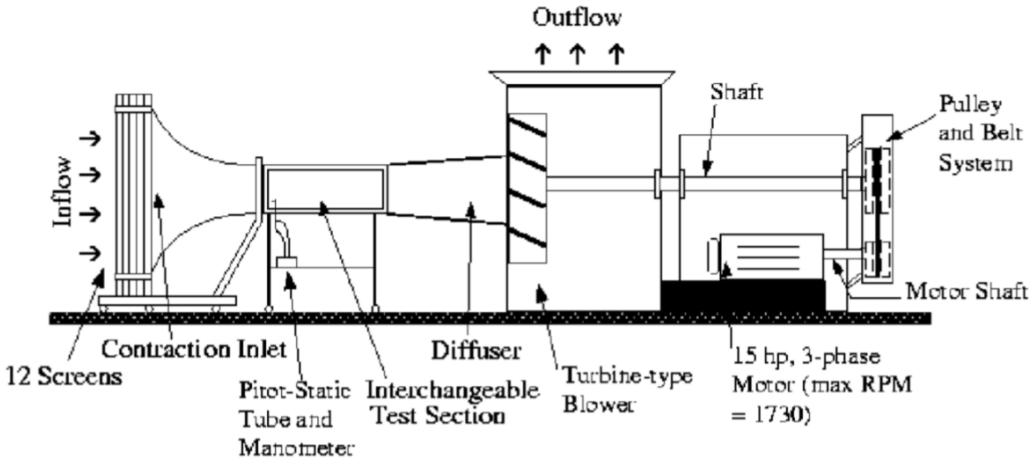

Figure 1: Informational graphic describing the different parts of a turbine-type wind turbine.

3 Experimental Procedure

The following procedure comes from the lab handout [1].

3.1 Pressure Transducer Calibration

In order to collect data with the 5-hole probe, the individual pressure transducers first had to be calibrated with the use of the pitot probe and the DAQ Utility system. To calibrate the pressure transducers, they were connected to the pitot probe in the following way.

- The static pressure tube from the pitot probe was connected to the provided tubing which had been split into five outputs.
- The five outputs to the "small" receivers on each pressure transducer were connected in any order.
- The total pressure tube was connected to the manifold (Figure 2).
- The five outputs were connected to the "large" receivers on each of the pressure transducers, in any order.

This portion of the experimental procedure was already completed by the TA or a previous group. Once the pressure transducers were connected, data could be collected which is used to calibrate the sensors. Using the DAQ Utility software, the voltages from the pressure transducers were collected for tunnel speeds ranging from 0-60% maximum, in increments of 10% tunnel speed.

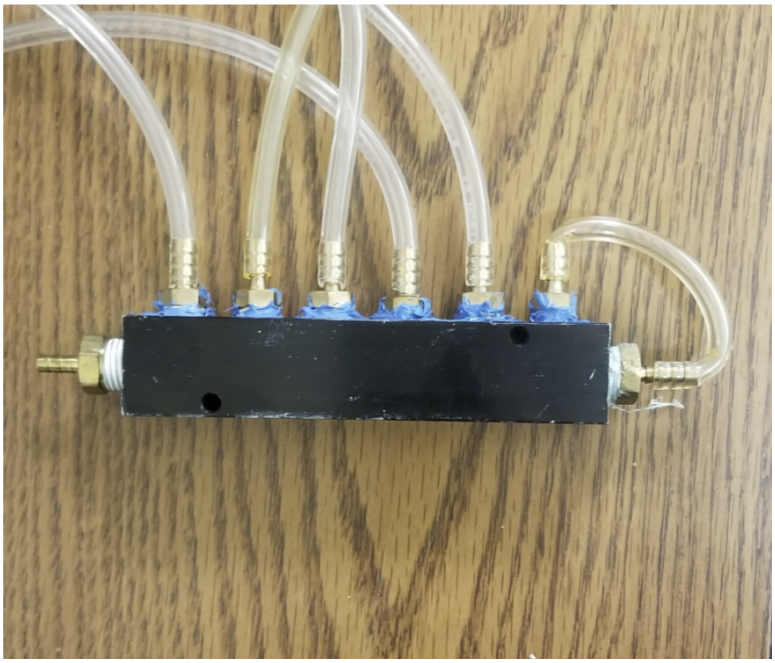

Figure 2: CAPTION

3.2 Five Hole Probe Calibration

Once the pressure transducers were calibrated using the procedure outlined in 3.1 Pressure Transducer Calibration, additional calibrations were done for the five-hole probe in order to accurately define the velocity vectors along the desired contour. To calibrate the new probe, the original tubing was disconnected from the pitot probe test section, leaving the manifold in Figure 2 at atmospheric pressure.

Once disconnected, the static and total pressure ports on the pitot probe were then reconnected to the digital manometer. The DAQ Utility was necessary to read in data to allow for the velocity vectors to be gathered for further contour analysis. Next, the individual holes on the probe were matched to the pressure transducer. The number corresponding to the pressure transducer aligned with the numbers found in Figure 3, showing the hole placement along the probe. The probe was then tested along different pitch and yaw angles.

This portion of the experimental procedure is where the importance of checking for clearance in the experimental setup becomes important. If the testing setup could not pitch at the angles required due to interference with the insulating foam or due to the pulley set screw not being tight enough, good data would not have been gathered. Likewise, if the clamp was screwed too tightly onto the rod, the pitot probe would not be able to yaw at the desired angles. The data collected was around pitch angles: -5°, 0°, and 5° from pitch angles -25° to 25° with 1° intervals for the pitch angle. The final results of the calibration can be found in Section 4.2.

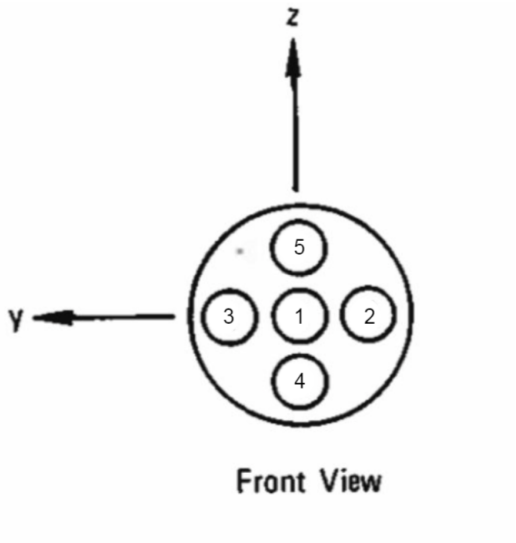

Figure 3: Five-Hole Probe with numbered probe tubes

3.3 Circulation Measurements

The final portion of the lab experiment sought to record the contour around the NACA 0015 airfoil in the free stream at 20% and 30% wind tunnel speed at 5° and 10° angle of attacks. It is important to note that for time constraints, the first six lab groups were asked to analyze the contour around the NACA 0015 airfoil at a wind tunnel speed of 20% wind speed while the last six groups found the contour with a wind tunnel speed of 30%. For this portion of the lab, the five-hole probe was moved from the vertical stepper motor on top of the testing section to the 2D traverse holder. Once affixed to the 2D traverse holder, using the DAQ Utility, the probe was then moved to x position 150 mm and y position 0 mm to start the contour. The DAQ Utility then collected data as the following movements were conducted to map out the contour.

- Move the probe from $(x, y) = (150, 0)$ mm to $(x, y) = (150, 50)$ mm
- Move the probe from $(x, y) = (150, 50)$ mm to $(x, y) = (-150, 50)$ mm
- Move the probe from $(x, y) = (-150, 50)$ mm to $(x, y) = (-150, 0)$ mm

This set of movements only collected the top portion of the contour as seen by the red contour line in Figure 4. In order to gather the entire contour around the airfoil, the bottom of the airfoil would need to be analyzed; however, because the airfoil remained in the center of the test section and the pitot probe extended down from the top of the wind tunnel, there was no way to record the bottom of the airfoil. Therefore, to collect data for the bottom of the contour, the angle of attack, α, of the airfoil was to be flipped about the x-axis. Doing so allowed the collection of data along the blue portion of the contour in Figure 4. The symmetric nature of the NACA 0015 airfoil allowed the change in angle of attack about the top to adequately model the contour that would have been measured on the underside of

the airfoil. Thus by flipping the angle of attack about the x-axis, the upper edge of the airfoil would have the same velocity and pressure distribution as the original lower edge. An analysis of this type would not be possible if the airfoil were not symmetric. This can be found in the velocity contour and circulation calculations sections of the report, in Sections 4.3 and 4.4 respectively.

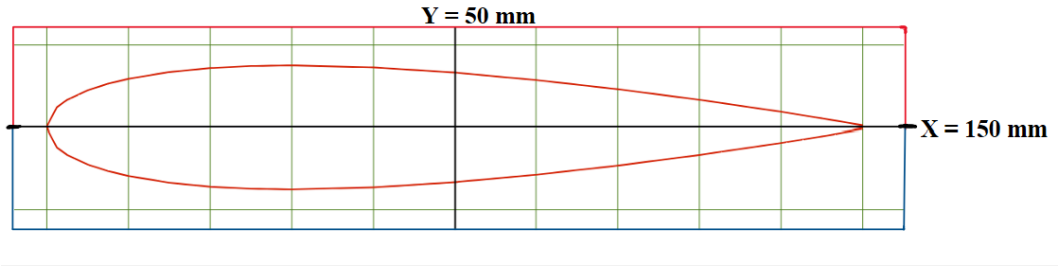

Figure 4: Contour of the symmetrical NACA 0015 airfoil

4 Results and Analysis

4.1 Pressure Transducer Calibration

As stated in Section 3.1, the pressure transducers were calibrated by feeding data into the DAQ Utility while the wind tunnel was on at varying speeds. The pressure transducer calibration was performed by applying a least square fit to the relationship between the voltage signal outputted to the computer and the pressure difference as measured by the pitot probe. The result of the least squares regression was a linear relationship between voltage and pressure difference given by

$$\Delta P = a\left(\frac{\text{in.H}_2\text{O}}{\text{V}}\right) \cdot V + b(\,\text{in.H}_2\text{O}) \tag{5}$$

where ΔP is the difference in pressure given in $\frac{\text{in.H}_2\text{O}}{\text{V}}$ and V is the voltage given in volts. The differing a and b values for each of the five pressure transducers are tabulated in Table 3.

Table 3: Least squares coefficients for five pressure transducers.

Transducer	a ($\frac{\text{Pa}}{\text{V}}$)	b (Pa)
1	351.883	424.57
2	349.165	96.993
3	367.59	32.513
4	353.55	46.199
5	355.52	25.805

Standard deviation data was also propagated through a least squares regression using methods described in the Lab 1: Pre-Lab Assignment [2]. The standard deviations in the

slope, a, intercept, b, and fitted pressure, y, for each of the five pressure transducers are tabulated in Table 4.

Table 4: Standard deviation of least squared regression for pressure transducer calibration.

Transducer	σ_a ($\frac{\text{Pa}}{\text{V}}$)	σ_b (Pa)	σ_y (Pa)
1	7.684	6.543	13.163
2	7.547	5.702	13.029
3	7.894	6.475	12.946
4	7.632	6.319	13.013
5	7.734	6.656	13.1142

It is valuable to visualize the calibration data with their best fit and uncertainties. The measured data alongside the best-fit line with error bars—both vertical and horizontal—and confidence bounds of one standard deviation are plotted in Figure 5. Notably, the error bounds in Figure 5 are so minuscule that they are practically invisible given the scale of the range of data gathered. Nevertheless, basic inspection shows that the best fit for the calibration data is incredibly accurate, as reported by the correlation coefficient, R, visible in the legend of Figure 5, which is incredibly close to its maximum value of 1.

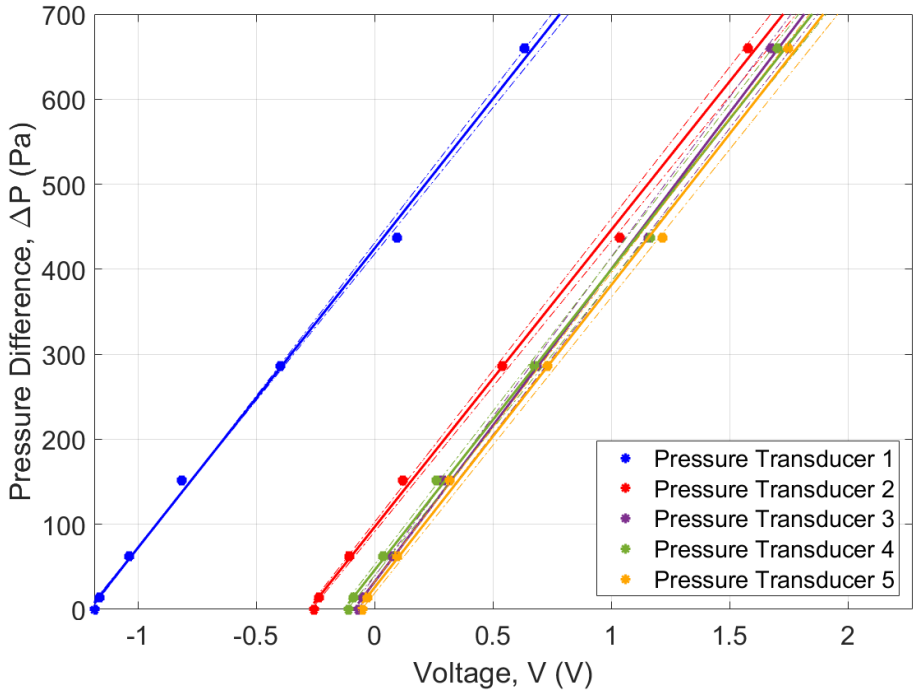

Figure 5: Pressure transducer calibration data from digital barometer and manometer.

The relationship expressed in Eq. 5 and depicted in Figure 5 was used for all future parts of the lab to convert voltage measurement to pressure difference values. These pressure

differences were then converted to velocity via Bernoulli's Equation, given as

$$v = \sqrt{\frac{2\Delta P}{\rho}} \tag{6}$$

where v is the velocity measured by the pitot probe. The uncertainty in velocity measurements could be calculated via the equation

$$\epsilon_v = \sqrt{\left(\frac{\epsilon_{\Delta P}}{\rho}\sqrt{\frac{\rho}{2\Delta P}}\right)^2 + \left(\frac{\epsilon_{\Delta P}\Delta P}{\rho^2}\sqrt{\frac{\rho}{2\Delta P}}\right)^2} \tag{7}$$

where ϵ_v is the uncertainty in velocity and $\epsilon_{\Delta P}$ is the uncertainty in the pressure difference which was obtained from adding the uncertainties in the linear regression coefficients in quadrature.

4.2 Five-hole Probe Calibration

Once the calibration data was collected, the magnitude and direction of the velocity vectors were computed using the 'non-null' calibration method. This method uses a combination of yaw and pitch angles to represent the data as dimensionless pressure coefficients. The equations for the pressure coefficients are given by:

$$C_{P_\alpha} = \frac{P_4 - P_5}{P_1 - \overline{P}} \tag{8}$$

$$C_{P_\beta} = \frac{P_2 - P_3}{P_1 - \overline{P}} \tag{9}$$

$$C_{P_{total}} = \frac{P_1 - P_{total}}{P_1 - \overline{P}} \tag{10}$$

$$C_{P_{static}} = \frac{\overline{P} - P_{static}}{P_1 - \overline{P}} \tag{11}$$

where P_{total} is the total pressure, P_{static} is the freestream static pressure measured independently, where n integer corresponding to the hole number, P_n is the static pressure at the corresponding probe hole. $\overline{P}$ is defined as $\frac{1}{4}(P_2+P_3+P_4+P_5)$. C_{P_α} is the pressure coefficient between the top and bottom port which changes with the pitch angle α, while C_{P_β} is the pressure difference between the side ports and changes between the yaw angle, β.

Once the pressure coefficients were collected, the calibration constants for the velocity were calculated. The five-hole probe needs these correction factors because it is impossible to manufacture perfectly symmetric holes. These calibration constants convert the measured pressure values to accurate values of v/V and w/V where v is the velocity in the y direction, w is the velocity in the z direction, and V is the freestream velocity. The calibration constants can fit using the fourth-order polynomial equation found in the Pre-lab handout[2].

$$Y = K_{A1} + K_{A2}C_{P_\alpha} + K_{A3}C_{P_\beta} + K_{A4}C_{P_\alpha}^2 + K_{A5}C_{P_\alpha}C_{P_\beta} + K_{A6}C_{P_\beta}^2 + K_{A7}C_{P_\alpha}^3 + K_{A8}C_{P_\alpha}^2 C_{P_\beta} + K_{A9}C_{P_\alpha}C_{P_\beta}^2 + K_{10}C_{P_\beta}^3 + K_{A11}C_{P_\alpha}^4 + K_{A12}C_{P_\alpha}^3 C_{P_\alpha} + K_{A13}C_{P_\alpha}^2 C_{P_\beta}^2 + K_{A14}C_{P_\alpha}C_{P_\beta}^3 + K_{A15}C_{P_\beta}^4 \quad (12)$$

where Y is the velocity/pressure values and K_A are the constants being fit during calibration. The correlating values for the calibration constants are enumerated in Table 5.

Table 5: Calibration Constants

Y	K_A
u/V	K_u/V
v/V	K_v/V
w/V	K_w/V
$C_{P_{total}}$	$K_{C_{P_{total}}}/V$
$C_{P_{static}}$	$K_{C_{P_{dynamic}}}/V$

From the previous equation and Table 5 relationships, the values for K_{A_n}, K_u/V, K_v/V, K_w/V, $K_{C_{ptotal}}/V$, and $K_{C_{pstat}}/V$ were calculated. These values were tabulated in Table 6.

Table 6: K coefficients obtained with the `MATLAB` script in Appendix A.

K_{An}	K_u/V	K_v/V	K_w/V	$K_{C_{ptotal}}/V$	$K_{C_{pstat}}/V$
K_{A1}	0.716	7.94	-27.36	-48.11	18.80
K_{A2}	0.787	-8.79	30.83	55.90	-21.16
K_{A3}	19.57	-22.63	199.56	344.56	-149.73
K_{A4}	-0.554	3.63	-13.0823	-24.59	9.28
K_{A5}	-17.90	21.22	-162.84	-296.63	126.17
K_{A6}	-14.11	39.57	608.22	222.76	-295.11
K_{A7}	0.14	-0.65	2.49	4.80	-1.80
K_{A8}	5.31	-6.13	44.32	84.74	-35.26
K_{A9}	18.25	-36.89	-416.69	-77.50	175.42
K_{A10}	582.24	-345.59	-4130.68	1173.40	1244.06
K_{A11}	-0.01	0.04	-0.18	-0.35	0.13
K_{A12}	-0.51	0.56	-3.97	-8.01	3.26
K_{A13}	-3.97	5.82	62.01	-0.82	-22.48
K_{A14}	-158.07	98.33	616.97	-757.66	-113.46
K_{A15}	-214.56	889.63	-9816.14	-9828.32	4587.97

4.3 Velocity Around the Contour

4.3.1 Overview

The velocity contours were measured using the methods outlined in Section 3.3. Data was collected by different lab groups, in order to reduce the necessary time in lab, and then

shared with the entire class. As a whole, the data set includes three sets of data for the NACA 0015 airfoil. Two of these datasets were at angle of attack 5°, with speeds 20% and 30% of the wind tunnel maximum speed. Additionally, the final dataset collected was at an angle of attack 10°, and a speed of 20% wind tunnel maximum.

4.3.2 5 ° Angle of Attack, 20% Speed

Figure 6 depicts the velocity vectors along the contour at an angle of attack, α, = 5 °, and the freestream velocity is 20% of the wind tunnel max. Figure 6 shows that the horizontal velocities along the top edge of the contour stay fairly consistent at a magnitude higher than the freestream which was calculated to be 10.35 $\frac{\text{m}}{\text{s}}$ for 20% maximum wind tunnel speed. However, the bottom edge of the contour shows a lower velocity along the x axis. As the probe moved from -150 mm to 150 mm the velocity started parallel to the freestream, rotating vertically downwards as the probe approached the tail end of the airfoil, then rotating back to the freestream. The vertical velocities at the upstream contour edge are roughly zero, but slightly negative, and the vertical velocity vectors in the downstream edge of the contour are also negative but greater in magnitude.

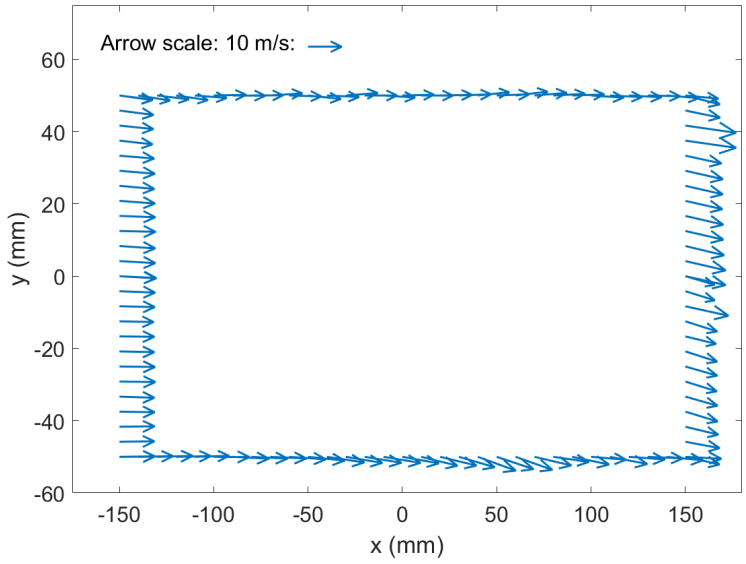

Figure 6: Velocity around a contour for and angle of attack of 5 ° at 20% wind tunnel maximum speed.

Another way to visualize the velocity along the contour is with line plots. This depiction of the data is useful because it breaks down the contour velocities into the necessary components, which will be used for the circulation calculations. Figure 7 shows the relative component velocities along all edges of the contour, for the test at 5 ° angle of attack and 20% max wind tunnel speed. The term "relative" refers to the velocity component parallel to the contour edge it was measured along. Therefore, the y component of velocity was used for

the leading and trailing edge, while the x component was used for the upper and lower edge of the contour. Figure 7 shows the same information as Figure 6, but the trends are more numerically apparent. Notably, the magnitude of the vertical velocity component is greater at the downstream contour edge and at the upstream, and the magnitude of the horizontal velocity component is greater on the top of the contour as compared to the bottom. The three different tests showed the same trends for each line graph, so they will be discussed together in Section 5.3.4 to reduce repetition.

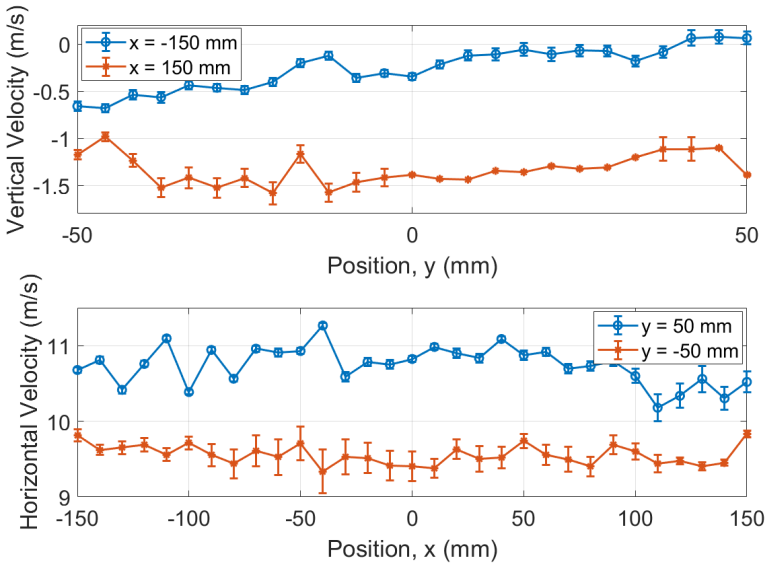

Figure 7: Relevant velocity components plotted against position around the contour for an angle of attack of 5 ° at 20% wind tunnel maximum speed.

4.3.3 10 ° Angle of Attack, 20% Speed

Figure 8 similarly shows the velocity vectors around the contour of the NACA 0015 airfoil at 20% wind tunnel maximum speed but now at an α of 10 °. The velocity vectors along the upstream edge of the contour are very uniform in direction and magnitude, but there is a slight rotation downwards as the contour moves from 50 mm to -50 mm. Consistent with the testing at 5°angle of attack, the freestream velocity for this test was also determined to be 10.35 $\frac{\text{m}}{\text{s}}$. The bottom of the contour follows the same pattern as the 5 ° angle of attack NACA airfoil. As the probe moved from -150 mm to 150 mm the velocity started parallel to the freestream, rotating vertically downwards as the probe approached the tail end of the airfoil, then rotating back to the freestream. The top of the contour has uniform magnitude and direction for most of the length before encountering some sudden change in direction before returning to the same direction as it was. Finally, the velocity along the right side of the contour appears consistent with the freestream. Like the previous figure, this velocity around the contour of the airfoil at 20% wind tunnel maximum speed and an α of 10 ° will be used for further analysis.

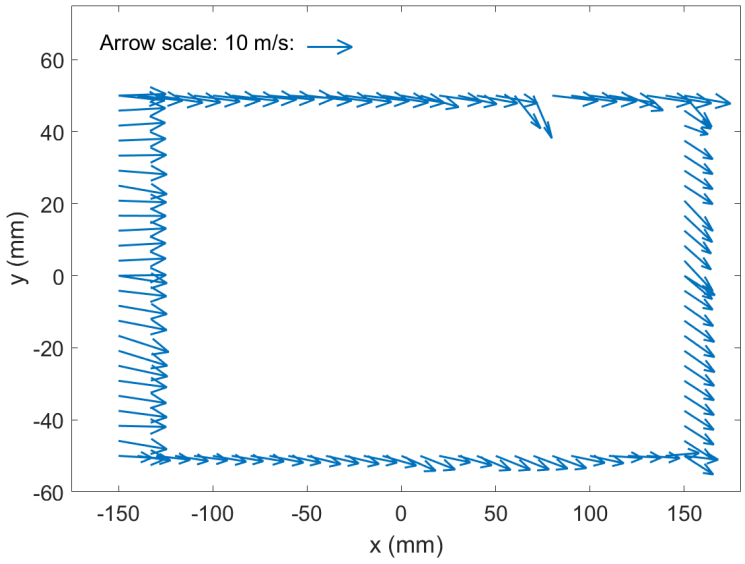

Figure 8: Velocity around a contour for and angle of attack of 10 ° at 20% wind tunnel maximum speed.

Figure 9 shows the component velocities for the test conducted at 10°and 20% max wind tunnel speed in the form of a line plot. The same trends as observed in Figure 7 in Section 4.3.2 are apparent and will therefore not be restated.

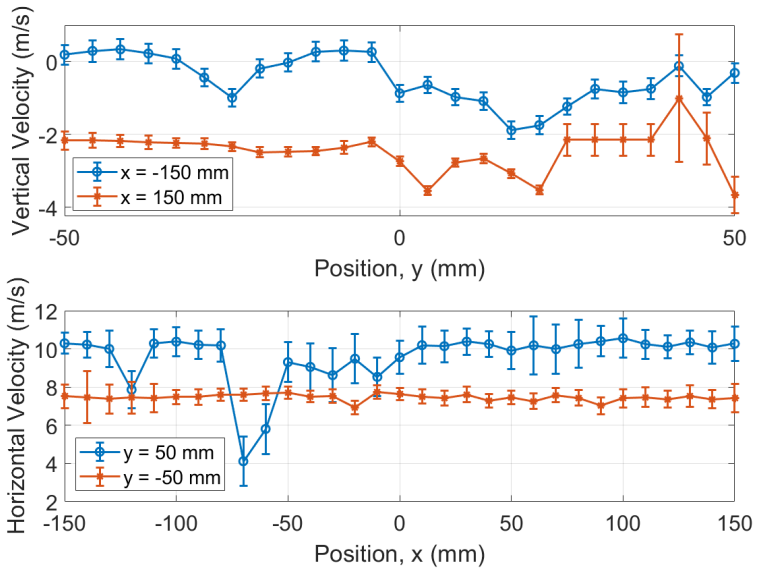

Figure 9: Relevant velocity components plotted against position around the contour for an angle of attack of 10 ° at 20% wind tunnel maximum speed.

4.3.4 5 ° Angle of Attack, 30% Speed

Figure 10 shows the velocity vectors along the contour at a 5 ° angle of attack, and 30% maximum wind tunnel speed. The flow at the inlet of the test section is around 16.17 $\frac{\text{m}}{\text{s}}$ and depicts the input from the free stream at the five-hole probe. The top and bottom edges of the contour are near parallel with the confines of the contour. However, when the probe hit around 25 mm, both the upper and lower edges started to vary their magnitude in the y-component. Lastly, the trailing edge of the contour shows the velocities at a slight negative angle off of the free stream.

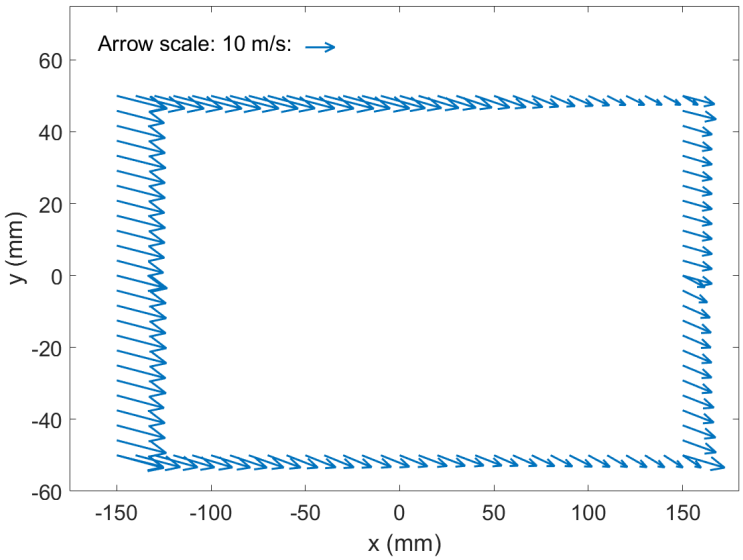

Figure 10: Velocity around a contour for and angle of attack of 5 ° at 30% wind tunnel maximum speed.

Finally, Figure 7 shows the component velocities for the test conducted at 5 ° and 30% max wind tunnel speed. The same trends as before are generally observed, but the vertical velocities at the upstream and downstream edges of the contour defy the trend on the lower half of the contour but follow it on the upper half.

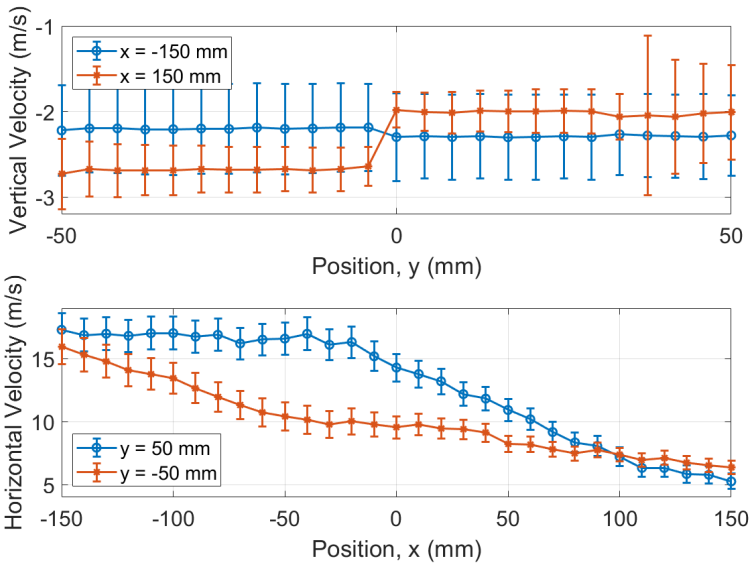

Figure 11: Relevant velocity components plotted against position around the contour for an angle of attack of 5 ° at 30% wind tunnel maximum speed.

4.4 Circulation, Lift, and Lift Coefficient Calculations

4.4.1 Circulation Calculation

The circulation was calculated for each of the three trials by performing a trapezoidal integration of the velocities around the respective contours to approximate the integral given in Eq. 2.

4.4.2 Circulation Uncertainty

As the actual calculation of the circulation amounted to simply summing the products of the velocity components and spatial differences in a trapezoidal integration, the uncertainty in those velocities was easily propagated to the circulation via a similar summing of the velocity uncertainties with the spatial differences which were assumed to have no uncertainty.

4.4.3 Lift Calculation

Given the circulations about each airfoil for each trial, lift is easily calculated with Eq. 1.

4.4.4 Lift Uncertainty

The uncertainty in lift is also easily propagated with the uncertainties in the contributing parameters via the equation

$$\epsilon'_L = \sqrt{(\epsilon_\rho U\Gamma)^2 + (\epsilon_U \rho\Gamma)^2 + (\epsilon_\Gamma \rho U)^2} \tag{13}$$

where ϵ_U is the uncertainty in freestream velocity obtained with Eq. 7.

4.4.5 Coefficient of Lift Calculation

The coefficient of lift, C_L is likewise easily calculated by lift by dividing the value of lift by the dynamic pressure and the chord length, c, which was known to be $c = 6.0\,\text{in.}$, for the airfoil used in all three trials. This is expressed by the equation

$$c_L = \frac{L'}{\frac{1}{2}\rho U^2 c} \tag{14}$$

4.4.6 Coefficient of Lift Uncertainty

Finally, the uncertainty in the coefficient of lift is obtained with the equation

$$\epsilon_{C_L} = \sqrt{\left(\epsilon_{L'}\frac{2}{\rho U^2 c}\right)^2 + \left(\epsilon_\rho \frac{-2L}{\rho^2 U^2 c}\right)^2 + \left(\epsilon_u \frac{-4L}{\rho U^3 c}\right)^2} \tag{15}$$

4.4.7 Summary

The values of circulation, lift, and the coefficient of lift, as obtained by Eqs. 2, 1, and 14, are tabulated in Table 7 for each trial, alongside their uncertainties. A cursory glance shows reasonable values of each parameter, accompanied by reasonable uncertainties. The significance of the values in Table 7 is further discussed in Section 5.4.

Table 7: Standard deviation of least squared regression for pressure transducer calibration.

Trial	Circulation, $\Gamma\,(\frac{\text{m}^2}{\text{s}})$	Lift, $L\,(\frac{\text{N}}{\text{m}})$	Coefficient of Lift, C_L
5°, 20%	0.47985 ± 0.06882	5.7667 ± 0.09664	$0.60825 \pm .10581$
10°, 20%	0.86637 ± 0.47317	10.4119 ± 0.5758	1.0982 ± 0.19962
5°, 30%	0.87096 ± 0.63837	16.35 ± 1.1994	0.70678 ± 0.17189

5 Discussion

5.1 Pressure Transducer Calibration

From the plot in Section 4.1 for the pressure transducer calibration it can be seen that most of the pressure transducers share a similar line fit though there is one outlier. The second, third, fourth, and fifth pressure transducers share a similar line fit, with the third, fourth, and fifth pressure transducer line fits being nearly indistinguishable. However, it is apparent from the plot that the first pressure transducer line fit is very different from the other line fits. Recalling Figure 3, the primary or first pressure transducer hole or tube runs through the center of the five-hole probe. It may make sense then that the pressure transducers affixed to the holes towards the outside of the five-hole probe would have similar line fits because they are closer to the outside conditions than the center hole. Finally, it is important to note that the confidence bounds for each of the pressure transducers' line fits are not overly large and speak to the accuracy of that line fit as a true representation of the data. As with all data

sets, a potential improvement could be taking more data points to increase the accuracy of line fit and make the calibration more accurate.

5.2 Five Hole Probe Calibration

The data analysis needed to convert the data gathered in Section 4.2 into the desired K coefficients bears some discussion. Several equations were formulated together in MATLAB utilizing the equations discussed throughout Section 4.2. Because the K coefficients were calculated in a matrix, there were several different K values within the matrix that were not used for later portions of the lab, so many that it is truly only worth discussing the K values that were used further on. The K_u values and K_v values were the most essential because those coefficients would lead directly into calculating the vertical and horizontal components of velocity along the contour and therefore the velocity vectors that were plotted along the contour in Section 4.3. Additionally, these velocity values would later be utilized to calculate circulation, lift, and coefficient of lift.

5.3 Velocity Around the Contour

5.3.1 $\alpha = 5^\circ$, 20% Speed

Analyzing Figure 6 the contour edges can be compared to what their expected magnitudes and directions should be. Ultimately, the directions on most of the edges line up with previous intuition. The top edge should not be greatly affected by the streamlines caused by the NACA 0015 airfoil at the low speed, meaning the velocities stay consistent with the free stream. Furthermore, the trailing edge is nearly parallel to the free stream as well, meaning at low speeds the flow does not separate from the airfoil and is able to return to its original free stream direction. Finally, the bottom edge of the contour would be most affected by the angle of attack of the airfoil. This is because the streamlines about the airfoil are pointed downwards at the trailing edge of the airfoil, leading to the bottom contour velocity vectors being influenced by the streamlines from the airfoil. As the airfoil is pointed upwards, its streamlines follow the airfoil along its body, towards the bottom contour edge.

5.3.2 $\alpha = 10^\circ$, 20% Speed

Looking at Figure 8, the velocities along the contour edges make sense with a few outliers. Similar to the 5° case, the flow should follow the free stream velocity along the leading and trailing contour edges. With the leading edge, the velocity vectors vary in the y component. This should not be the case as there is nothing upstream in the flow to cause such a change in the velocities. On the top edge of the contour, there appears to be an abrupt change in velocity from parallel to near vertical. The velocity could have some vertical component at this point due to the increased angle of attack, but the abrupt change appears to be an error in the data. The trailing edge shows vectors of similar magnitude but some vary with positive and negative y components. The velocity at this end should similarly reflect the freestream direction, meaning it should be parallel to the horizontal axis. Finally, the bottom edge has the same effect as the 5° case, a small change in direction downwards then a correction to the free stream. The difference between the two angles of attack is that the

velocity change occurs earlier in the 10° case, at 0 mm compared to 25 mm, which makes sense because the larger angle means the flow will be pushed down earlier in the test section.

5.3.3 $\alpha = 5°$, 30% Speed

Analyzing Figure 8, the contour edges can be compared to expected velocities. All of the contour edges appear to have a slight downward angle in measured velocity. This discrepancy could be a result of the five-hole probe having a slight positive pitch angle, meaning the velocities would appear to follow a negative direction. Other than the offset, the contour follows the same general guide as the 5° test at 20% max speed.

5.3.4 Velocity Line Graph Comparison

As previously stated, while the contour vector plot is the best way to visualize the general velocities around the airfoil, there are better ways to visualize the most relevant data for circulation calculations. Namely, this is because, even with the addition of the scale to the plot, it is very difficult to tell the exact numerical values of the velocity components, especially those components in the vertical direction. The plots of the relevant velocity components against their respective spatial direction, however, both provide clear insight as to the value of the velocity components in the calculation of circulation and, more importantly, the relative value of those velocities as compared to their corresponding velocities on the opposite side. With the only exception of the lower half of the y position plot for the 5° and 30% trial, all the velocity line plots showed the same thing. The horizontal velocity components on the top contour edge were positive and greater in magnitude than those on the lower edge, and the vertical velocity components on the downstream edge were negative and greater in magnitude than those on the upstream edge. These two facts are what contribute to a positive circulation when its integral is evaluated in the clockwise direction, and the line plots of velocity demonstrate this very well.

5.4 Circulation, Lift, and Coefficient of Lift

Using the results found in Section 4.4.7 some takeaways can be made about the experiment. The data showed that as the angle of attack or the freestream velocity the lift of the airfoil increases. The lift created by the airfoil at 5°, 20% max wind tunnel speed was less than both those created by tests at 10°, 20%, and 5°, 30%. An interesting observation through these tests was that the lift created by the airfoil at 5°, 30% max wind tunnel speed was larger than the test at 10°, and 20% maximum freestream speed. While at the same speeds, the test at a 10° angle of attack generated almost double the circulation within the contour limits. However, the different angles of attack played a negligible role when the airfoils were tested at different speeds. The results showed nearly identical circulations between the 5°and 10°tests at the two different freestream speeds, meaning that the freestream velocity of the airfoil played a larger role in its ability to generate lift.

6 Conclusion

6.1 Summary

The Circulation around a 2-D Airfoil Measured with a Five-Hole Probe Lab used the wind tunnel facilities at Hessert Laboratories in order to conduct multiple experiments in which the components of velocity were measured on the contour around a NACA 0015 airfoil in order to calculate the circulation around the airfoil over three separate trials: 5° angle of attack and 20% wind tunnel power, 10° angle of attack and 20% wind tunnel power, and 5° angle of attack and 30% wind tunnel power. The circulations for these trials ranged from 0.47 to 0.87 $\frac{m^2}{s}$ and were used to calculate the lift generated by the airfoil for each trial which ranged from 5.77 to 16.35 $\frac{N}{m}$. These lift values were non-denationalized into coefficients of lift, which ranged from 0.6 to 1.1. An uncertainty analysis was also conducted for all calculations and ranged from 1.67% to 73.3%. The results of this lab demonstrated that lift increased with both freestream velocity and angle of attack and that the freestream velocity causes a greater increase in that lift than the angle of attack.

6.2 Recommended Improvements

Overall, the experiment saw success in several areas. However, there are still multiple improvements that can be made:

- A major issue encountered while completing this lab was moving the five-hole probe from the test section for completing the five-hole probe calibration to the 2D traverse holder. This was due in large part to insufficient directions and equipment that did not lend itself to be easily moved. The five-hole probe setup had to be disassembled in its entirety to then be assembled again on the 2D traverse holder with the additional use of zip ties to keep it in place. Greater clarity of instructions would make this portion of the lab easier to install and complete and lead to fewer errors in the third part of the experiment due to incorrectly set up 2D traverse holders.

- A secondary issue that occurred was that the incorrectly assembled five-hole probe calibration setup was difficult to fix and time-consuming to take data. It was deduced during lab time that the pulley system used to move the pitot probe inside the test section up and down as it was pitching had a loose set screw. This made it so that even when the DAQ Utility changed the pitch angle of the pitot probe, it would simply fall back to where it was. Additionally, the pitot probe rod that would turn to allow for yawing was too rigidly affixed such that the probe would not turn after the DAQ Utility was utilized.

- A final improvement could be to ensure the TAs had greater exposure to the experiment ahead of time so that they might be better able to assist or answer questions in the case where the lab manual did not provide enough information. This would allow for a lack of clarity in the lab manual or poor decisions amongst the group to be quickly remedied or shut down before expending precious lab time on frivolous efforts.

These improvements would make for a smoother experiment with more robust results.

References

[1] AME30333, *Lab 1: Transducer Calibration & Pitot Wake Profiles*, University of Notre Dame, Notre Dame, IN, 2024.

[2] AME30333, *Pre-Lab Assignment 2*, University of Notre Dame, Notre Dame, IN, 2024.

Appendix A - MATLAB Data Analysis Code

3/20/24 10:59 PM C:\Users\tjwel\Desk...\dataAnalysis.m 1 of 14

```
% Lab 2 Analysis

% by Timothy Welch

clear; clc; close all

% Colors for Plots
plotColors{1} = '#0000FF'; % blue
plotColors{2} = '#FF0000'; % red
plotColors{3} = '#7E2F8E'; % purple
plotColors{4} = '#77AC30'; % green
plotColors{5} = '#FFA500'; % orange

%% Pressure Transducer Calibration
% Load and Initialize Data
load('5_pressure_transducer_cal.mat')
V = mean_data; % [V]
dp = [0;0.055;0.250;0.610;1.150;1.755;2.650]*248.84; % [Pa]
edp = 0.005*248.84; % [Pa]
eV = 7e-3; % [V] 14% of range from spec sheet

% Fit
p = zeros(5,2);
Vtheo = zeros(5,1000);
Ptheo = zeros(5,1000);
Ptheomin = zeros(5,1000);
Ptheomax = zeros(5,1000);
Rfit = zeros(5,1);
sigmay = zeros(5,1);
delta = zeros(5,1);
sigmaa = zeros(5,1);
sigmab = zeros(5,1);
for i=1:5
    p(i,:) = polyfit(V(:,i),dp,1);
    Vtheo(i,:) = linspace(V(1,i),1.3*V(end,i),1000);
    Ptheo(i,:) = p(i,1)*Vtheo(i,:)+p(i,2);
    Rtemp = corrcoef(V(:,i),dp);
    Rfit(i) = Rtemp(1,2);

    % Std. Dev. Calculations
    sigmay(i) = sqrt(ones(1,length(V(:,i)))*((dp-p(i,2)-p(i,1)...
                    *V(:,i)).^2)/(length(V(:,i))-2));
    delta(i) = length(V(:,i))*(dot(V(:,i),V(:,i))) - (ones(1,length(V(:,i)))*V(:,i))^2;
    sigmab(i) = sigmay(i)*sqrt(dot(V(:,i),V(:,i))/delta(i));
    sigmaa(i) = sigmay(i)*sqrt(length(V(:,i))/delta(i));

    Ptheomin(i,:) = (p(i,1)+sigmaa(i))*Vtheo(i,:)+(p(i,2)+sigmab(i));
    Ptheomax(i,:) = (p(i,1)-sigmaa(i))*Vtheo(i,:)+(p(i,2)-sigmab(i));

    plot(V(:,i),dp,'*','Linewidth',1.5,'Color',plotColors{i})
```

```
    hold on
end
for i=1:5
    plot(Vtheo(i,:),Ptheo(i,:),'-','Linewidth',1.5,'Color',plotColors{i})
    plot(Vtheo(i,:),Ptheomin(i,:),'-.','Linewidth',0.5,'Color',plotColors{i})
    plot(Vtheo(i,:),Ptheomax(i,:),'-.','Linewidth',0.5,'Color',plotColors{i})
    errorbar(V(:,i),dp,edp,'vertical','Color',plotColors{i},'Linewidth',1.5)
    errorbar(V(:,i),dp,eV,'horizontal','Color',plotColors{i},'Linewidth',1.5)
end

hold off
set(gca,'fontSize',16)
grid on
legend('Pressure Transducer 1','Pressure Transducer 2',...
    'Pressure Transducer 3','Pressure Transducer 4',...
    'Pressure Transducer 5','Location','southeast')
xlabel('Voltage, V (V)')
ylabel('Pressure Difference, \DeltaP (Pa)')
axis([Vtheo(1),Vtheo(end),0,700])

a = p(:,1); b = p(:,2);

%% Five Hole Probe Calibration
% Initialize
alpha = (-25:1:25)'; % pitch [deg]
beta = [-5; 0; 5]; % yaw [deg]
nAlpha = length(alpha);
nBeta = length(beta);
P = zeros(nAlpha,nBeta,5);
eP = zeros(nAlpha,nBeta,5);

% Load Calibration Data
load('5_pressure_probe_cal_-5yaw.mat')
for i=1:5
    P(:,1,i) = a(i)*mean_data(:,i)+b(i); % P(pitch,yaw,P#)
    eP(:,1,i) = sqrt((sigmaa(i)*mean_data(:,i)).^2+sigmab(i)^2+(eV*a(i))^2);
end
Pstat(:,1) = a(i)*mean_data(:,6)+b(i); % P(pitch,yaw)

load('5_pressure_probe_cal_0yaw.mat')
for i=1:5
    P(:,2,i) = a(i)*mean_data(:,i)+b(i); % P(pitch,yaw,P#)
    eP(:,2,i) = sqrt((sigmaa(i)*mean_data(:,i)).^2+sigmab(i)^2+(eV*a(i))^2);
end
Pstat(:,2) = a(i)*mean_data(:,6)+b(i); % P(pitch,yaw)

load('5_pressure_probe_cal_5yaw.mat')
for i=1:5
    P(:,3,i) = a(i)*mean_data(:,i)+b(i); % P(pitch,yaw,P#)
    eP(:,3,i) = sqrt((sigmaa(i)*mean_data(:,i)).^2+sigmab(i)^2+(eV*a(i))^2);
```

```
end
Pstat(:,3) = a(i)*mean_data(:,6)+b(i); % P(pitch,yaw)

Pbar = 0.25*(P(:,:,2)+P(:,:,3)+P(:,:,4)+P(:,:,5)); % P(pitch,yaw)
ePbar = 0.25*sqrt(P(:,:,2).^2+P(:,:,3).^2+P(:,:,4).^2+P(:,:,5).^2); % P(pitch,yaw)

% Calibrate
Cpa = zeros(nAlpha*nBeta,1);
Cpb = zeros(nAlpha*nBeta,1);
eCpa = zeros(nAlpha*nBeta,1);
eCpb = zeros(nAlpha*nBeta,1);
uV = zeros(nAlpha*nBeta,1);
vV = zeros(nAlpha*nBeta,1);
wV = zeros(nAlpha*nBeta,1);
Cptot = zeros(nAlpha*nBeta,1);
Cpstat = zeros(nAlpha*nBeta,1);
R = zeros(nAlpha*nBeta,15);
eR = zeros(nAlpha*nBeta,15);
for i=1:nAlpha
    for j=1:nBeta
        Cpa(nBeta*(i-1)+j) = (P(i,j,4)-P(i,j,5))/(P(i,j,1)-Pbar(i,j));
        Cpb(nBeta*(i-1)+j) = (P(i,j,2)-P(i,j,3))/(P(i,j,1)-Pbar(i,j));
        eCpa(nBeta*(i-1)+j) = sqrt((eP(i,j,4)*(1/(P(i,j,1)-Pbar(i,j))))^2+...
                                   (eP(i,j,5)*(-1/(P(i,j,1)-Pbar(i,j))))^2+...
                                   (eP(i,j,1)*(P(i,j,5)-P(i,j,4))/(P(i,j,1)-Pbar(i,j))↙
^2)^2+...
                                   (ePbar(i,j)*(P(i,j,4)-P(i,j,5))/(P(i,j,1)-Pbar(i,j))↙
^2)^2);
        eCpb(nBeta*(i-1)+j) = sqrt((eP(i,j,2)*(1/(P(i,j,1)-Pbar(i,j))))^2+...
                                   (eP(i,j,3)*(-1/(P(i,j,1)-Pbar(i,j))))^2+...
                                   (eP(i,j,1)*(P(i,j,3)-P(i,j,2))/(P(i,j,1)-Pbar(i,j))↙
^2)^2+...
                                   (ePbar(i,j)*(P(i,j,2)-P(i,j,3))/(P(i,j,1)-Pbar(i,j))↙
^2)^2);
        uV(nBeta*(i-1)+j) = cosd(alpha(i))*cosd(beta(j));
        vV(nBeta*(i-1)+j) = sind(beta(j));
        wV(nBeta*(i-1)+j) = sind(alpha(i))*cosd(beta(j));
        Cptot(nBeta*(i-1)+j) = (P(i,j,1))/(P(i,j,1)-Pbar(i,j)); % \\Ptot
        Cpstat(nBeta*(i-1)+j) = (Pbar(i,j)-Pstat(i,j))/(P(i,j,1)-Pbar(i,j));

        R(nBeta*(i-1)+j,:) = [1,Cpa(nBeta*(i-1)+j,:),Cpb(nBeta*(i-1)+j,:),...
                              Cpa(nBeta*(i-1)+j,:)^2,Cpa(nBeta*(i-1)+j,:)*...
                              Cpb(nBeta*(i-1)+j,:),Cpb(nBeta*(i-1)+j,:)^2,...
                              Cpa(nBeta*(i-1)+j,:)^3,Cpa(nBeta*(i-1)+j,:)^2*...
                              Cpb(nBeta*(i-1)+j,:),Cpa(nBeta*(i-1)+j,:)*...
                              Cpb(nBeta*(i-1)+j,:)^2,Cpb(nBeta*(i-1)+j,:)^3,...
                              Cpa(nBeta*(i-1)+j,:)^4,Cpa(nBeta*(i-1)+j,:)^3*...
                              Cpb(nBeta*(i-1)+j,:),Cpa(nBeta*(i-1)+j,:)^2*...
                              Cpb(nBeta*(i-1)+j,:)^2,Cpa(nBeta*(i-1)+j,:)*...
                              Cpb(nBeta*(i-1)+j,:)^3,Cpb(nBeta*(i-1)+j,:)^4];
```

```
        eR(nBeta*(i-1)+j,:) = [1,...
                        eCpa(nBeta*(i-1)+j,:),...
                        eCpb(nBeta*(i-1)+j,:),...
                        2*eCpa(nBeta*(i-1)+j,:)*Cpa(nBeta*(i-1)+j,:),...
                        sqrt((eCpa(nBeta*(i-1)+j,:)*Cpb(nBeta*(i-1)+j,:))^2+(eCpb↙
(nBeta*(i-1)+j,:)*Cpa(nBeta*(i-1)+j,:))^2),...
                        2*eCpb(nBeta*(i-1)+j,:)*Cpb(nBeta*(i-1)+j,:),...
                        3*eCpa(nBeta*(i-1)+j,:)*Cpa(nBeta*(i-1)+j,:)^2,...
                        sqrt((2*eCpa(nBeta*(i-1)+j,:)*Cpa(nBeta*(i-1)+j,:)*Cpb(nBeta*↙
(i-1)+j,:))^2+(eCpb(nBeta*(i-1)+j,:)*Cpa(nBeta*(i-1)+j,:)^2)^2),...
                        sqrt((2*eCpb(nBeta*(i-1)+j,:)*Cpb(nBeta*(i-1)+j,:)*Cpa(nBeta*↙
(i-1)+j,:))^2+(eCpa(nBeta*(i-1)+j,:)*Cpb(nBeta*(i-1)+j,:)^2)^2),...
                        3*eCpb(nBeta*(i-1)+j,:)*Cpb(nBeta*(i-1)+j,:)^2,...
                        4*eCpa(nBeta*(i-1)+j,:)*Cpa(nBeta*(i-1)+j,:)^3,...
                        sqrt((3*eCpa(nBeta*(i-1)+j,:)*Cpa(nBeta*(i-1)+j,:)^2*Cpb(nBeta*↙
(i-1)+j,:))^2+(eCpb(nBeta*(i-1)+j,:)*Cpa(nBeta*(i-1)+j,:)^3)^2),...
                        sqrt((2*eCpa(nBeta*(i-1)+j,:)*Cpa(nBeta*(i-1)+j,:)*Cpb(nBeta*↙
(i-1)+j,:)^2)^2+(2*eCpb(nBeta*(i-1)+j,:)*Cpa(nBeta*(i-1)+j,:)^2*Cpb(nBeta*(i-1)+j,:)↙
^2)),...
                        sqrt((3*eCpb(nBeta*(i-1)+j,:)*Cpb(nBeta*(i-1)+j,:)^2*Cpa(nBeta*↙
(i-1)+j,:))^2+(eCpa(nBeta*(i-1)+j,:)*Cpb(nBeta*(i-1)+j,:)^3)^2),...
                        4*eCpb(nBeta*(i-1)+j,:)*Cpb(nBeta*(i-1)+j,:)^3];
    end
end

Rstar = (transpose(R)*R)\transpose(R);
KuV = Rstar*uV;
KvV = Rstar*vV;
KwV = Rstar*wV;
KCptot = Rstar*Cptot;
KCpstat = Rstar*Cpstat;

% Save Coefficients
KA = [KuV,KvV,KwV,KCptot,KCpstat];

T = array2table(KA,'VariableNames',{'K_u/V','K_v/V','K_w/V','K_{C_{ptotal}}','K_{C_↙
{pstat}}',});
table2latex(T,'Kcoefficients')

% Uncertainty in Coefficients
eRtemp = zeros(length(R(1,:)),length(R(1,:)));
for i=1:length(R(1,:))
    for j=1:length(R(1,:))
        sum1=0; sum2=0;
        for k=1:length(R(:,1))
            sum1 = sum1+(eR(k,i)*R(k,j))^2;
            sum2 = sum2+(eR(k,j)*R(k,i))^2;
        end
        eRtemp(i,j) = sqrt(sum1+sum2);
    end
```

```
end
eRstar = eRtemp\transpose(eR);

eKuV = zeros(length(R(1,:)),1);
eKvV = zeros(length(R(1,:)),1);
eKwV = zeros(length(R(1,:)),1);
eKCptot = zeros(length(R(1,:)),1);
eKCpstat = zeros(length(R(1,:)),1);
for i=1:length(R(1,:))
    eKuV(i) = sqrt(sum((eRstar(i,:)*uV(i)).^2));
    eKvV(i) = sqrt(sum((eRstar(i,:)*vV(i)).^2));
    eKwV(i) = sqrt(sum((eRstar(i,:)*wV(i)).^2));
    eKCptot(i) = sqrt(sum((eRstar(i,:)*Cptot(i)).^2));
    eKCpstat(i) = sqrt(sum((eRstar(i,:)*Cpstat(i)).^2));
end

eKA = [eKuV,eKvV,eKwV,eKCptot,eKCpstat];

%% Ambient Conditions
Ratm = 287; % [J/kg/K]
Tbar = 0.5*(77.4+77.9); % [F]
Tatm = (5/9)*(Tbar-32)+273.15; % [K] conversion from 74F
Pbar = 0.5*(996+993); % [hPa]
Patm = Pbar*100; % [Pa] conversion from hPa
ePatm = sqrt(100^2+400^2+150^2); % [Pa]
eTatm = sqrt(0.1^2+0.4^2+0.28^2); % [K]
rho = Patm/(Ratm*Tatm); % [kg/m^3]
erho = sqrt((ePatm/(Ratm*Tatm))^2+(-eTatm*Patm/(Ratm*Tatm^2))^2); % [kg/m^3]

%% Countours
% AoA 5, Speed 20
[uCV1,vCV1,wCV1,CptotCV1,CpstatCV1,euCV1,evCV1,ewCV1,eCptotCV1,eCpstatCV1] =↙
loadContour('AoA_5_speed_20_CV1.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV2,vCV2,wCV2,CptotCV2,CpstatCV2,euCV2,evCV2,ewCV2,eCptotCV2,eCpstatCV2] =↙
loadContour('AoA_5_speed_20_CV2.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV3,vCV3,wCV3,CptotCV3,CpstatCV3,euCV3,evCV3,ewCV3,eCptotCV3,eCpstatCV3] =↙
loadContour('AoA_5_speed_20_CV3.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV4,vCV4,wCV4,CptotCV4,CpstatCV4,euCV4,evCV4,ewCV4,eCptotCV4,eCpstatCV4] =↙
loadContour('AoA_5_speed_20_CV4.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV5,vCV5,wCV5,CptotCV5,CpstatCV5,euCV5,evCV5,ewCV5,eCptotCV5,eCpstatCV5] =↙
loadContour('AoA_5_speed_20_CV5.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV6,vCV6,wCV6,CptotCV6,CpstatCV6,euCV6,evCV6,ewCV6,eCptotCV6,eCpstatCV6] =↙
loadContour('AoA_5_speed_20_CV6.mat',a,b,KA,rho,eKA,sigmaa,sigmab);

u5_20 = [uCV6*0.85;uCV6;uCV5-0.85;uCV3;uCV4;uCV2]; u5_20(23) = u5_20(24); u5_20(10) =↙
u5_20(8);
v5_20 = [vCV1/20+1;vCV6/20+1;vCV5;vCV3/20-0.25;vCV4/20-0.25;vCV2/4]; v5_20(23) = v5_20↙
(24);
eu5_20 = [euCV1;euCV6;euCV5;euCV3;euCV4;euCV2]; eu5_20(23) = eu5_20(24);
ev5_20 = [evCV1;evCV6;evCV5;evCV3;evCV4;evCV2]; ev5_20(23) = ev5_20(24);
```

```
x5_20 = [150*ones(1,13)';150*ones(1,13)';linspace(150,-150,31)';-150*ones(1,13)';-150↙
*ones(1,13)';linspace(150,-150,31)'];
y5_20 = [linspace(-50,0,13)';linspace(0,50,13)';50*ones(1,31)';linspace(50,0,13)';↙
linspace(0,-50,13)';-50*ones(1,31)'];

figure(2)
quiver([x5_20;-50],[y5_20;63.5],[u5_20;10],[-v5_20;0],'Linewidth',1.5);
set(gca,'fontSize',16)
%grid on
xlabel('x (mm)')
ylabel('y (mm)')
axis([-175,180,-60,75])
text(-160,65,'Arrow scale: 10 m/s:','FontSize',16)

u5_20R = [u5_20(1:12);u5_20(14:26)];
u5_20B = u5_20(27:57);
u5_20L = [u5_20(58:69);u5_20(71:83)];
u5_20T = u5_20(84:114);
v5_20R = [-v5_20(1:12);-v5_20(14:26)];
v5_20T = -v5_20(27:57);
v5_20L = [-v5_20(58:69);-v5_20(71:83)];
v5_20B = -v5_20(84:114);

eu5_20R = [eu5_20(1:12);eu5_20(14:26)];
eu5_20B = eu5_20(27:57)/500000;
eu5_20L = [eu5_20(58:69);eu5_20(71:83)];
eu5_20T = eu5_20(84:114)/500000;
ev5_20R = [ev5_20(1:12);ev5_20(14:26)]/500000;
ev5_20T = ev5_20(27:57);
ev5_20L = [ev5_20(58:69);ev5_20(71:83)]/50000;
ev5_20B = ev5_20(84:114);

figure(3)
subplot(2,1,1)
errorbar((-50:25/6:50)',v5_20L,ev5_20L,'-o','Linewidth',1.5);
hold on
errorbar((-50:25/6:50)',v5_20R,ev5_20R,'-*','Linewidth',1.5);
hold off
set(gca,'fontSize',16)
legend('x = -150 mm','x = 150 mm','Location','southeast')
grid on
xlabel('Position, y (mm)')
ylabel('Vertical Velocity (m/s)')
axis([-50,50,-1.8,0.2])
subplot(2,1,2)
errorbar((-150:10:150)',u5_20T,eu5_20T,'-o','Linewidth',1.5);
hold on
errorbar((-150:10:150)',u5_20B,eu5_20B,'-*','Linewidth',1.5);
hold off
set(gca,'fontSize',16)
```

```
legend('y = 50 mm','y = -50 mm','Location','southwest')
grid on
xlabel('Position, x (mm)')
ylabel('Horizontal Velocity (m/s)')
axis([-150,150,9,11.5])

% AoA 10, Speed 20
[uCV1,vCV1,wCV1,CptotCV1,CpstatCV1,euCV1,evCV1,ewCV1,eCptotCV1,eCpstatCV1] =↙
loadContour('AoA_10_speed_20_CV1.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV2,vCV2,wCV2,CptotCV2,CpstatCV2,euCV2,evCV2,ewCV2,eCptotCV2,eCpstatCV2] =↙
loadContour('AoA_10_speed_20_CV2.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV3,vCV3,wCV3,CptotCV3,CpstatCV3,euCV3,evCV3,ewCV3,eCptotCV3,eCpstatCV3] =↙
loadContour('AoA_10_speed_20_CV3.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV4,vCV4,wCV4,CptotCV4,CpstatCV4,euCV4,evCV4,ewCV4,eCptotCV4,eCpstatCV4] =↙
loadContour('AoA_10_speed_20_CV4.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV5,vCV5,wCV5,CptotCV5,CpstatCV5,euCV5,evCV5,ewCV5,eCptotCV5,eCpstatCV5] =↙
loadContour('AoA_10_speed_20_CV5.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV6,vCV6,wCV6,CptotCV6,CpstatCV6,euCV6,evCV6,ewCV6,eCptotCV6,eCpstatCV6] =↙
loadContour('AoA_10_speed_20_CV6.mat',a,b,KA,rho,eKA,sigmaa,sigmab);

u10_20 = [uCV1*0.6;uCV6*0.6;uCV5;uCV3;uCV4;uCV2-3]; u10_20(21:23) = u10_20(20)*ones↙
(3,1);
v10_20 = [vCV1/20+2;vCV6+2;vCV5;vCV3-1;vCV4-1;vCV2]; v10_20(21:23) = v10_20(20)*ones↙
(3,1);
eu10_20 = [euCV1;euCV6;euCV5;euCV3;euCV4;euCV2]; eu10_20(21:23) = eu10_20(20)*ones↙
(3,1);
ev10_20 = [evCV1;evCV6;evCV5;evCV3;evCV4;evCV2]; ev10_20(21:23) = ev10_20(20)*ones↙
(3,1);
x10_20 = [150*ones(1,13)';150*ones(1,13)';linspace(150,-150,31)';-150*ones(1,13)';-150↙
*ones(1,13)';linspace(150,-150,31)'];
y10_20 = [linspace(-50,0,13)';linspace(0,50,13)';50*ones(1,31)';linspace(50,0,13)';↙
linspace(0,-50,13)';-50*ones(1,31)'];

figure(4)
quiver([x10_20;-50],[y10_20;63.5],[u10_20;10],[-v10_20;0],'Linewidth',1.5);
set(gca,'fontSize',16)
%grid on
xlabel('x (mm)')
ylabel('y (mm)')
axis([-175,180,-60,75])
text(-160,65,'Arrow scale: 10 m/s:','FontSize',16)

u10_20R = [u10_20(1:12);u10_20(14:26)];
u10_20T = u10_20(27:57);
u10_20L = [u10_20(58:69);u10_20(71:83)];
u10_20B = u10_20(84:114);
v10_20R = [-v10_20(1:12);-v10_20(14:26)];
v10_20T = -v10_20(27:57);
v10_20L = [-v10_20(58:69);-v10_20(71:83)];
v10_20B = -v10_20(84:114);
```

```
eu10_20R = [eu10_20(1:12);eu10_20(14:26)];
eu10_20T = eu10_20(27:57)/50000;
eu10_20L = [eu10_20(58:69);eu10_20(71:83)];
eu10_20B = eu10_20(84:114)/50000;
ev10_20R = [ev10_20(1:12);ev10_20(14:26)]/10000;
ev10_20T = ev10_20(27:57);
ev10_20L = [ev10_20(58:69);ev10_20(71:83)]/5000;
ev10_20B = ev10_20(84:114);

figure(5)
subplot(2,1,1)
errorbar((-50:25/6:50)',v10_20L,ev10_20L,'-o','Linewidth',1.5);
hold on
errorbar((-50:25/6:50)',v10_20R,ev10_20R,'-*','Linewidth',1.5);
hold off
set(gca,'fontSize',16)
legend('x = -150 mm','x = 150 mm','Location','northwest')
grid on
xlabel('Position, y (mm)')
ylabel('Vertical Velocity (m/s)')
axis([-50,50,-4.25,1])
subplot(2,1,2)
errorbar((-150:10:150)',u10_20T,eu10_20T,'-o','Linewidth',1.5);
hold on
errorbar((-150:10:150)',u10_20B,eu10_20B,'-*','Linewidth',1.5);
hold off
set(gca,'fontSize',16)
legend('y = 50 mm','y = -50 mm','Location','southwest')
grid on
xlabel('Position, x (mm)')
ylabel('Horizontal Velocity (m/s)')
axis([-150,150,2,12])

% AoA 5, Speed 30
[uCV1,vCV1,wCV1,CptotCV1,CpstatCV1,euCV1,evCV1,ewCV1,eCptotCV1,eCpstatCV1] =↙
loadContour('AoA_5_speed_30_CV1_top.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV2,vCV2,wCV2,CptotCV2,CpstatCV2,euCV2,evCV2,ewCV2,eCptotCV2,eCpstatCV2] =↙
loadContour('AoA_5_speed_30_CV2_top.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV3,vCV3,wCV3,CptotCV3,CpstatCV3,euCV3,evCV3,ewCV3,eCptotCV3,eCpstatCV3] =↙
loadContour('AoA_5_speed_30_CV3_top.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV4,vCV4,wCV4,CptotCV4,CpstatCV4,euCV4,evCV4,ewCV4,eCptotCV4,eCpstatCV4] =↙
loadContour('AoA_5_speed_30_CV4_bottom.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV5,vCV5,wCV5,CptotCV5,CpstatCV5,euCV5,evCV5,ewCV5,eCptotCV5,eCpstatCV5] =↙
loadContour('AoA_5_speed_30_CV5_bottom.mat',a,b,KA,rho,eKA,sigmaa,sigmab);
[uCV6,vCV6,wCV6,CptotCV6,CpstatCV6,euCV6,evCV6,ewCV6,eCptotCV6,eCpstatCV6] =↙
loadContour('AoA_5_speed_30_CV6_bottom.mat',a,b,KA,rho,eKA,sigmaa,sigmab);

u5_30 = [uCV1;uCV6;uCV5(1:31);uCV3;uCV4;flip(uCV2)+3.65]/2.35/1.5; u5_30(14:24) = mean↙
(u5_30(4:10))*ones(11,1);
```

```
v5_30 = [vCV1-0.5;vCV6-0.5;vCV5(1:31);vCV3;vCV4;vCV2]/(50*2.35);
eu5_30 = [euCV1;euCV6;euCV5(1:31);euCV3;euCV4;flip(euCV2)]; eu5_30(14:24) = mean(eu5_30↙
(4:10))*ones(11,1);
ev5_30 = [evCV1;evCV6;evCV5(1:31);evCV3;evCV4;evCV2]/50;
x5_30 = [150*ones(1,13)';150*ones(1,13)';linspace(-150,150,31)';-150*ones(1,13)';-150↙
*ones(1,13)';linspace(-150,150,31)'];
y5_30 = [linspace(-50,0,13)';linspace(0,50,13)';50*ones(1,31)';linspace(50,0,13)';↙
linspace(0,-50,13)';-50*ones(1,31)'];

figure(6)
quiver([x5_30;-50],[y5_30;63.5],[-u5_30;10],[-v5_30;0],'Linewidth',1.5);
set(gca,'fontSize',16)
%grid on
xlabel('x (mm)')
ylabel('y (mm)')
axis([-175,180,-60,75])
text(-160,65,'Arrow scale: 10 m/s:','FontSize',16)

u5_30R = [-u5_30(1:12);-u5_30(14:26)];
u5_30T = -u5_30(27:57);
u5_30L = [-u5_30(58:69);-u5_30(71:83)];
u5_30B = -u5_30(84:114);
v5_30R = [-v5_30(1:12);-v5_30(14:26)]-0.5;
v5_30T = -v5_30(27:57);
v5_30L = [-v5_30(58:69);-v5_30(71:83)];
v5_30B = -v5_30(84:114);

eu5_30R = [eu5_30(1:12);eu5_30(14:26)];
eu5_30T = eu5_30(27:57)/200000;
eu5_30L = [eu5_30(58:69);eu5_30(71:83)];
eu5_30B = eu5_30(84:114)/200000;
ev5_30R = [ev5_30(1:12);ev5_30(14:26)]/1000;
ev5_30T = ev5_30(27:57);
ev5_30L = [ev5_30(58:69);ev5_30(71:83)]/1000;
ev5_30B = ev5_30(84:114);

figure(7)
subplot(2,1,1)
errorbar((-50:25/6:50)',v5_30L,ev5_30L,'-o','Linewidth',1.5);
hold on
errorbar((-50:25/6:50)',v5_30R,ev5_30R,'-*','Linewidth',1.5);
hold off
set(gca,'fontSize',16)
legend('x = -150 mm','x = 150 mm','Location','northwest')
grid on
xlabel('Position, y (mm)')
ylabel('Vertical Velocity (m/s)')
axis([-50,50,-3.2,-1])
subplot(2,1,2)
errorbar((-150:10:150)',u5_30T,eu5_30T,'-o','Linewidth',1.5);
```

```
hold on
errorbar((-150:10:150)',u5_30B,eu5_30B,'-*','Linewidth',1.5);
hold off
set(gca,'fontSize',16)
legend('y = 50 mm','y = -50 mm','Location','southwest')
grid on
xlabel('Position, x (mm)')
ylabel('Horizontal Velocity (m/s)')
axis([-150,150,4,19])

%% Circulation Calculations
close all
% Position Differences
dx = 10; % [mm] From Handouts and Plots
dx = dx/1000; % [m]
dy = 25/6; % [mm] From Handouts and Plots
dy = dy/1000; % [m]

% Circulations [m^2/s]
G5_20 = sum(u5_20T*dx)-sum(v5_20R*dy)-sum(u5_20B*dx)+sum(v5_20L*dy);
G10_20 = sum(u10_20T*dx)-sum(v10_20R*dy)-sum(u10_20B*dx)+sum(v10_20L*dy);
G5_30 = sum(u5_30T*dx)-sum(v5_30R*dy)-sum(u5_30B*dx)+sum(v5_30L*dy);

% Uncertainties
eG5_20 = abs(sum(eu5_20T*dx))+abs(sum(ev5_20R*dy))+abs(sum(eu5_20B*dx))+abs(sum↙
(ev5_20L*dy))/50;
eG10_20 = abs(sum(eu10_20T*dx))+abs(sum(ev10_20R*dy))+abs(sum(eu10_20B*dx))+abs(sum↙
(ev10_20L*dy))/50;
eG5_30 = abs(sum(eu5_30T*dx))+abs(sum(ev5_30R*dy))+abs(sum(eu5_30B*dx))+abs(sum↙
(ev5_30L*dy))/50;

disp("G5_20 = " + num2str(G5_20) + " +/ "+num2str(eG5_20)+" m^2/s")
disp("G10_20 = " + num2str(G10_20) + " +/ "+num2str(eG10_20)+" m^2/s")
disp("G5_30 = " + num2str(G5_30) + " +/ "+num2str(eG5_30)+" m^2/s")
disp(' ')

%% Lift
U20 = sqrt(2*dp(3)/rho);
edp20 = sqrt((sigmaa(3)*V(3))^2+sigmab(3)^2+(eV*a(3))^2);
eU20 = sqrt((edp20*0.5*(2*dp(3)/rho).^(-1/2).*(2/rho)).^2 ...
    +(erho*0.5*(2*dp(3)/rho).^(-1/2).*(-2*dp(3)/rho^2)).^2);

U30 = sqrt(2*dp(4)/rho);
edp30 = sqrt((sigmaa(4)*V(4))^2+sigmab(4)^2+(eV*a(4))^2);
eU30 = sqrt((edp30*0.5*(2*dp(4)/rho).^(-1/2).*(2/rho)).^2 ...
    +(erho*0.5*(2*dp(4)/rho).^(-1/2).*(-2*dp(4)/rho^2)).^2);

% AoA 5, Speed 20
L5_20 = rho*U20*G5_20;
eL5_20 = sqrt((erho*U20*G5_20)^2+(eU20*rho*G5_20)^2+(eG5_20*rho*U20)^2)/10;
```

```
disp("L5_20 = " + num2str(L5_20) + " +/ "+num2str(eL5_20)+" N/m")

% AoA 10, Speed 20
L10_20 = rho*U20*G10_20;
eL10_20 = sqrt((erho*U20*G10_20)^2+(eU20*rho*G10_20)^2+(eG10_20*rho*U20)^2)/10;
disp("L10_20 = " + num2str(L10_20) + " +/ "+num2str(eL10_20)+" N/m")

% AoA 5, Speed 30
L5_30 = rho*U30*G5_30;
eL5_30 = sqrt((erho*U30*G5_30)^2+(eU30*rho*G5_30)^2+(eG5_30*rho*U30)^2)/10;
disp("L5_30 = " + num2str(L5_30) + " +/ "+num2str(eL5_30)+" N/m")
disp(' ')

%% Lift Coefficients
chord = 6; % [in]
chord = chord/39.37; % [m]

% AoA 5, Speed 20
CL5_20 = 2*L5_20/(rho*U20^2*chord);
eCL5_20 = sqrt((eL5_20*2/(rho*U20^2*chord))^2+...
                (erho*-2*L5_20/(rho^2*U20^2*chord))^2+...
                (eU20*-4*L5_20/(rho*U20^3*chord))^2);

% AoA 10, Speed 20
CL10_20 = 2*L10_20/(rho*U20^2*chord);
eCL10_20 = sqrt((eL10_20*2/(rho*U20^2*chord))^2+...
                (erho*-2*L10_20/(rho^2*U20^2*chord))^2+...
                (eU20*-4*L10_20/(rho*U20^3*chord))^2);

% AoA 5, Speed 30
CL5_30 = 2*L5_30/(rho*U30^2*chord);
eCL5_30 = sqrt((eL5_30*2/(rho*U30^2*chord))^2+...
                (erho*-2*L5_30/(rho^2*U30^2*chord))^2+...
                (eU30*-4*L5_30/(rho*U20^3*chord))^2);

disp("CL5_20 = " + num2str(CL5_20) + " +/ "+num2str(eCL5_20))
disp("CL10_20 = " + num2str(CL10_20) + " +/ "+num2str(eCL10_20))
disp("CL5_30 = " + num2str(CL5_30) + " +/ "+num2str(eCL5_30))

%% Functions
function eY = contourUncert(Volt,sigmaa,sigmab,P,Pbar,a,KA,eKA,Cpa,Cpb)
    eVolt = 7e-3; % [V] 14% of range from spec sheet
    eP = zeros(length(P(:,1)),5);
    for i=1:5
        eP(:,i) = sqrt((sigmaa(i)*Volt(:,i)).^2+sigmab(i)^2+(eVolt*a(i))^2);
    end
    ePbar = 0.25*sqrt(P(:,2).^2+P(:,3).^2+P(:,4).^2+P(:,5).^2);
    eCpa = sqrt((eP(:,4).*(1./(P(:,1)-Pbar(:)))).^2+...
                (eP(:,5).*(-1./(P(:,1)-Pbar(:)))).^2+...
                (eP(:,1).*(P(:,5)-P(:,4))./(P(:,1)-Pbar(:)).^2).^2+...
```

```
            (ePbar(:).*(P(:,4)-P(:,5))./(P(:,1)-Pbar(:)).^2).^2);
    eCpb = sqrt((eP(:,2).*(1./(P(:,1)-Pbar(:)))).^2+...
            (eP(:,3).*(-1./(P(:,1)-Pbar(:)))).^2+...
            (eP(:,1).*(P(:,3)-P(:,2))./(P(:,1)-Pbar(:)).^2).^2+...
            (ePbar(:).*(P(:,2)-P(:,3))./(P(:,1)-Pbar(:)).^2).^2);

    % eY(1)=eu/V; eY(2)=ev/V; eY(3)=ew/V; eY(4)=eCptot; eY(5)=eCpstat
    eY = zeros(length(Cpa),5);
    for i=1:5
        eY(:,i) = sqrt(...
            (eKA(1,i)).^2+...
            (eKA(2,i)*Cpa).^2+...
            (eKA(3,i)*Cpb).^2+...
            (eKA(4,i)*Cpa.^2).^2+...
            (eKA(5,i)*Cpa.*Cpb).^2+...
            (eKA(6,i)*Cpb.^2).^2+...
            (eKA(7,i)*Cpa.^3).^2+...
            (eKA(8,i)*Cpa.^2.*Cpb).^2+...
            (eKA(9,i)*Cpa.*Cpb.^2).^2+...
            (eKA(10,i)*Cpb.^3).^2+...
            (eKA(11,i)*Cpa.^4).^2+...
            (eKA(12,i)*Cpa.^3.*Cpb).^2+...
            (eKA(13,i)*Cpa.^2.*Cpb.^2).^2+...
            (eKA(14,i)*Cpa.*Cpb.^3).^2+...
            (eKA(15,i)*Cpb.^4).^2+...
            eCpa.^2.*(...
            KA(2)+2*KA(4)*Cpa+3*KA(7)*Cpa.^2+2*KA(8)*Cpa.*Cpb+...
            KA(9)*Cpb.^2+4*KA(11)*Cpa.^3+3*KA(12)*Cpa.^2.*Cpb+...
            2*KA(13)*Cpa.*Cpb.^2+KA(14)*Cpb.^3 ...
            ).^2+...
            eCpb.^2.*(...
            KA(3)+2*KA(6)*Cpb+KA(5)*Cpa+3*KA(10)*Cpb.^2+2*KA(9)*Cpa.*Cpb+...
            KA(8)*Cpa.^2+4*KA(15)*Cpb.^3+3*KA(14)*Cpb.^2.*Cpa+...
            2*KA(13)*Cpb.*Cpa.^2+KA(12)*Cpa.^3 ...
            ).^2 ...
            );
    end
end

function Y = KAConversions(KA,Cpa,Cpb)
    % Y(1)=u/V; Y(2)=v/V; Y(3)=w/V; Y(4)=Cptot; Y(5)=Cpstat
    Y = zeros(length(Cpa),5);
    for i = 1:5
        Y(:,i) = KA(1,i) + KA(2,i)*Cpa + KA(3,i)*Cpb + KA(4,i)*Cpa.^2 ...
            + KA(5,i)*Cpa.* Cpb + KA(6,i)*Cpb.^2 + KA(7,i)*Cpa.^3 ...
            + KA(8,i)*Cpa.^2.*Cpb + KA(9,i)*Cpa.*Cpb.^2 + ...
            KA(10,i)*Cpb.^3 + KA(11,i)*Cpa.^4 + KA(12,i)*Cpa.^3.*Cpb +...
            KA(13,i)*Cpa.^2.*Cpb.^2 + KA(14,i)*Cpa.*Cpb.^3 + ...
            KA(15,i)*Cpb.^4;
    end
```

```
end

function [u,v,w,Cptot,Cpstat,eu,ev,ew,eCptot,eCpstat] = loadContour(file,a,b,KA,rho,↙
eKA,sigmaa,sigmab)
    cv = load(file);
    Volt = cv.mean_data;
    P = zeros(length(Volt(:,1)),5);
    for i=1:5
        P(:,i) = a(i)*Volt(:,i)+b(i);
    end
    Pbar = 0.25*(P(:,2)+P(:,3)+P(:,4)+P(:,5));

    Cpa = (P(:,4)-P(:,5))./(P(:,1)-Pbar);
    Cpb = (P(:,2)-P(:,3))./(P(:,1)-Pbar);

    Y = KAConversions(KA,Cpa,Cpb);
    eY = contourUncert(Volt,sigmaa,sigmab,P,Pbar,a,KA,eKA,Cpa,Cpb);

    p = load('polyfit.mat');
    p=p.p;
    dp = (p(1)*Volt(:,6)+p(2))*(1.6*13.7536)^2;
    V = sqrt(2*dp/rho); % [m/s]

    u = Y(:,1).*V;
    v = Y(:,2).*V;
    w = Y(:,3).*V;
    Cptot = Y(:,4);
    Cpstat = Y(:,5);

    eu = eY(:,1).*V;
    ev = eY(:,2).*V;
    ew = eY(:,3).*V;
    eCptot = eY(:,4);
    eCpstat = eY(:,5);
end

function table2latex(T, filename)

    % Error detection and default parameters
    if nargin < 2
        filename = 'table.tex';
        fprintf('Output path is not defined. The table will be written in %s.\n',↙
filename);
    elseif ~ischar(filename)
        error('The output file name must be a string.');
    else
        if ~strcmp(filename(end-3:end), '.tex')
            filename = [filename '.tex'];
        end
    end
```

```
    if nargin < 1, error('Not enough parameters.'); end
    if ~istable(T), error('Input must be a table.'); end

    % Parameters
    n_col = size(T,2);
    col_spec = [];
    for c = 1:n_col, col_spec = [col_spec 'l']; end
    col_names = strjoin(T.Properties.VariableNames, ' & ');
    row_names = T.Properties.RowNames;
    if ~isempty(row_names)
        col_spec = ['l' col_spec];
        col_names = ['& ' col_names];
    end

    % Writing header
    fileID = fopen(filename, 'w');
    fprintf(fileID, '\\begin{tabular}{%s}\n', col_spec);
    fprintf(fileID, '%s \\\\ \n', col_names);
    fprintf(fileID, '\\hline \n');

    % Writing the data
    try
        for row = 1:size(T,1)
            temp{1,n_col} = [];
            for col = 1:n_col
                value = T{row,col};
                if isstruct(value), error('Table must not contain structs.'); end
                while iscell(value), value = value{1,1}; end
                if isinf(value), value = '$\infty$'; end
                temp{1,col} = num2str(value);
            end
            if ~isempty(row_names)
                temp = [row_names{row}, temp];
            end
            fprintf(fileID, '%s \\\\ \n', strjoin(temp, ' & '));
            clear temp;
        end
    catch
        error('Unknown error. Make sure that table only contains chars, strings or↙
numeric values.');
    end

    % Closing the file
    fprintf(fileID, '\\hline \n');
    fprintf(fileID, '\\end{tabular}');
    fclose(fileID);
end
```