Estimating Centerline Velocity and Half-Width from Planar Jet Velocity Profiles

In the planar jet experiment, we choose different x-locations, and at each of these x-locations, we measure the dynamic pressure (using a pitot probe) at various y-locations across the jet. Flow velocity at these locations is then calculated from the equation for dynamic pressure, $\Delta p = \rho V^2/2$. The measured dynamic pressure and the corresponding calculated velocities usually display a scatter owing to uncertainties in experimental measurements. Figure 1 shows velocity distribution, obtained from measured data, along the width of the jet at an x-location past the end of the potential core (the data is taken from one of the datasets).

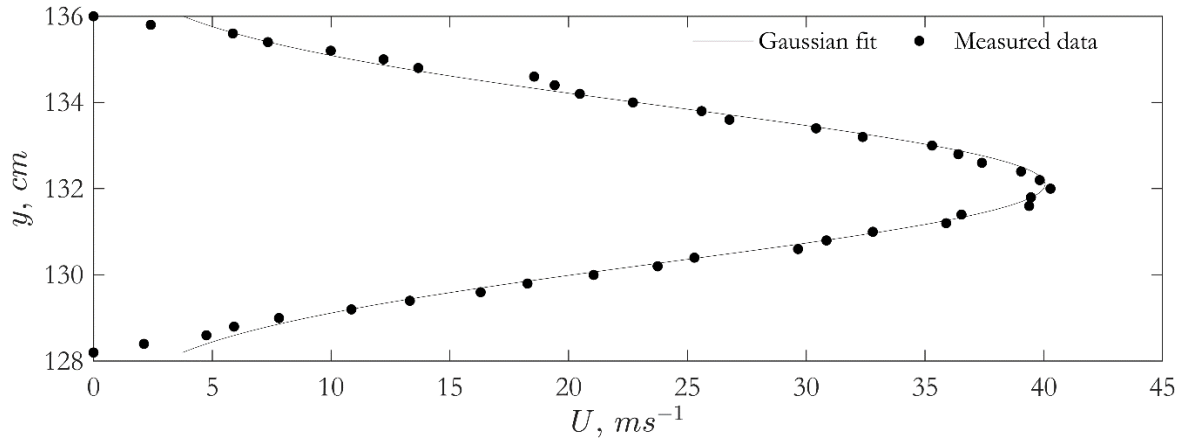

Figure 1: Velocity calculated from pressure measurements are fit to a Gaussian function

It can be seen from Figure 1 that the scattered data makes estimation of the peak velocity as well as the jet half-width difficult. However, we note that the velocity profile appears similar to a Gaussian function. Thus, we suggest that you fit the velocity profile data at each x-location (i.e., $U(x$=constant,$y)$) to a Gaussian function as given below, as long as you are past the end of the potential core:

$$U(y) = U_0 \exp\left\{-\ln 2 \frac{(y-y_0)^2}{b^2}\right\}.$$

Here, U_0 is the centerline velocity, b is the jet half-width (see that this works out by putting $U = U_0/2$ in the above expression), and y_0 is the position of the centerline. Note that you will obtain different values of these three parameters at each x-location. Although you may expect y_0 to be constant throughout x, it may not be so due to slight non-parallelism between the x-traverse axis and the jet axis.

The actual curve fitting can be done using the MATLAB function fit as follows

```
f = fit( y, U, 'gauss1').
```

An example MATLAB code is given for you. Note that the 'gauss1' built-in function is

$$U(y) = a_1 \exp\left\{-\left(\frac{y-b_1}{c_1}\right)^2\right\}.$$

Thus, a conversion is needed from the fit parameter c_1 returned by the fit function to obtain the actual half-width that is of interest to you. More details about MATLAB's fit function can be found in the link https://in.mathworks.com/help/curvefit/gaussian.html

If MATLAB is not accessible, the fitting can also be done in Python using the curve_fit function in the scipy.optimize module. This will also require scipy and numpy packages for Python. A tutorial for curve fitting in Python can be found in the link https://towardsdatascience.com/basic-curve-fitting-of-scientific-data-with-python-9592244a2509