Conical flow simulation project.

In this assignment you will create a matlab code that accepts the following input parameters:

- upstream Mach number
- shock angle (in deg)

These parameters will be used to evaluate the appropriate cone angle and the pressure, temperature, and density along the cone.

You will choose a point on one of the curves in Figure 10.5. From the curve, you know the freestream Mach number and you can estimate the shock angle. Assume atmospheric values of pressure, temperature, and density in the freestream. It is your job to show that one can compute the associated cone angle in Figure 10.5 by solving the system of ODEs using the marching procedure discussed in class. Once you have your solution, compare it to the solution for flow past a wedge with the same half-angle as the cone half-angle. Discuss the differences in Mach number, pressure, temperature, and density at the body surface You must do your own work on this project. You may discuss the logic (i.e. the flowchart) needed to create program with your peers, but the programming should reflect your work alone. Make sure to comment your code!

The steps you should follow the follow for this project are:

- Find the appropriate value of δ from the θ-β-M relation, and the associated normal Mach number just downstream of the shock and the jump values across the oblique shock. You can use functions that were supplied for the last assignment, but you will need to program your own temperature jump and density jump equations.
- Compute the Mach number just downstream of the shock.
- Compute V' from the downstream Mach number.
- Compute the values of V_r' and V_θ' at θ_s (remember that V_θ' should be negative).
- Perform the iteration to find where V_θ' changes from negative to positive as discussed in class.
- Perform the iteration using a smaller change in θ to check for "convergence" of the scheme.
- Use the **quiver** command to visualize the velocity as it changes. quiver accepts vectors of: the x-locations, the y-locations, and the x and y components of velocity. It then produces arrows at the x and y locations that are proportional to the velocity. You can use the varying θ values and a set r value to get the locations as you iterate. You must convert the r and θ components of velocity to x and y components. Again, the process for doing the conversions can be discussed in groups, but the coding should be done individually.

The milestones you should use to check your program are:

- Use the program to redo the problem that we worked in class to make sure it is working.
- Apply the program to your chosen point from Figure 10.5.
- Make the computation for the wedge, and discuss the results.