Combustion Theory Final Take Home Exam

Nicholas Malaya
Department of Mechanical Engineering
University of Texas at Austin

Turbulent diffusion flames

In this problem, I want you to assume that the system is turbulent and that you know the turbulence mass diffusivity (D_T) is 10 times the laminar value. Assume that the fluctuation squared of the mixture fraction is equal to the gradient of the mean mixture fraction squared multiplied by the characteristic diffusion length scale squared, i.e.

$$z^{\bar{\prime}}z' = \frac{1}{2}(\nabla \bar{z})^2 \frac{D_T L_x}{u} \tag{1}$$

a) Write down the solution for the mean mixture fraction field with the turbulent diffusivity.

We start with the steady state species equation,

$$\rho u \frac{\partial Y_i}{\partial x} = \rho D \frac{\partial^2 Y_i}{\partial y^2} \pm \omega_i. \tag{2}$$

We note that,

$$\omega = \frac{\omega_i}{\nu_i W_i} = \frac{\omega_F}{\nu_F W_F} = \frac{\omega_O}{\nu_O W_O}. \tag{3}$$

This hints at a conserved scalar form of the species equation, where with,

$$\beta = \frac{\omega_F}{\nu_F W_F} - \frac{\omega_O}{\nu_O W_O} \tag{4}$$

then our reaction is decoupled from the convection-diffusion of a conserved scalar quantity. In particular,

$$\mathcal{L}(\beta) = \mathcal{L}\left(\frac{\omega_F}{\nu_F W_F} - \frac{\omega_O}{\nu_O W_O}\right) \Rightarrow \rho u \frac{\partial \beta_i}{\partial x} - \rho D \frac{\partial^2 \beta_i}{\partial y^2} = 0. \tag{5}$$

Now, we construct z,

$$z = \frac{\beta - \beta_O}{\beta_F - \beta_O} \tag{6}$$

Here, the boundary conditions are that $z = 1$ for all $y > 0$ and $x < 0$ (e.g. the fuel reserve) and $z = 0$ for $y < 0$ and $x < 0$ (e.g. the oxygen reserve).

Thus, we are solving,

$$\rho u \frac{\partial z}{\partial x} - \rho D \frac{\partial^2 z}{\partial y^2} = 0. \tag{7}$$

and,

$$\rho u \frac{\partial \bar{z}}{\partial x} - \rho D_T \frac{\partial^2 \bar{z}}{\partial y^2} = 0. \tag{8}$$

Where the first equation is from the laminar flow, and the latter case is the favre-averaged mean field. We now need to discretize this equation, in order to solve it numerically (It looks like it would be a trainwreck to solve analytically!). We will completely wimp out, and only use finite difference methods,

$$\frac{\partial \bar{z}}{\partial x} = \frac{\bar{z}_{i+1} - \bar{z}_i}{\Delta x} \tag{9}$$

$$\frac{\partial^2 \bar{z}}{\partial y^2} = \frac{\bar{z}_{j+1} - 2\bar{z}_j + \bar{z}_{j-1}}{\Delta y^2} \tag{10}$$

The boundary conditions are $\bar{z} = 1 \; \forall x < 0, y > 0$, $\bar{z} = 0 \; \forall x < 0, y < 0$. I additionally imposed a Neuman (zero flux) boundary condition on the top and bottom of the box, essentially forcing the derivatives to zero at $\pm\infty$. As with the previous examination, we are now in a position to instantiate this on a computer using python to solve for the $\bar{z}(x, y)$ field.

As an interesting implementation detail, because the solution is only first order in x, we only need one boundary condition for that direction, as shown above. This is equivalent to an initial condition on a first order (in time) ODE. Thus, we do not actually need to store our entire x-domain in memory, but can simply solve for all y-values at our particular x-coordinate, and then step forward in space and solve for the next grid location.

b) Plot the fluctuation and mean value of the mixture fraction at 5 cm, 30 cm, and 50 cm.

The method to arrive at the mean value mixture fraction was described above. The value of the mixture fraction at several locations is plotted in figure 1. The flow at $x = 0$ cm is a sharp interface between the fuel and the oxidizer. As you move downstream, the fuel and oxidants mix, which will create a mixing layer that diffuses out and increases in y-width as a function of distance downstream.

The fluctuation $z^{\bar{\prime}}z'$ must also be determined. Normally, this would require solving another differential equation, and potentially using submodels for the scalar dissipation rate as well. However, we were given a simplified expression (model) for the variance, namely,

$$z^{\bar{\prime}}z' = \frac{1}{2}(\nabla \bar{z})^2 \frac{D_T L_x}{u}. \tag{11}$$

Variations in y will be much larger than variations in x, and this expression can be simplified to be,

$$z^{\bar{\prime}}z' = \frac{1}{2}(\frac{\partial \bar{z}}{\partial y})^2 \frac{D_T L_x}{u}. \tag{12}$$

This expression is a model for the variance of the mixture fraction. Intuitively, this expression is reasonable, as we expect the variance (in some sense, our uncertainty) of the value of the mixture fraction to be largest in regions with large gradients. In the mixing layer, the gradient will be quite large near the layer ($y = 0, x = 0$) and expanding outward at larger x.

We discretize equation 2 using the finite difference scheme shown in equation 10, however, now the indicies are changed from i to j, to reflect the different

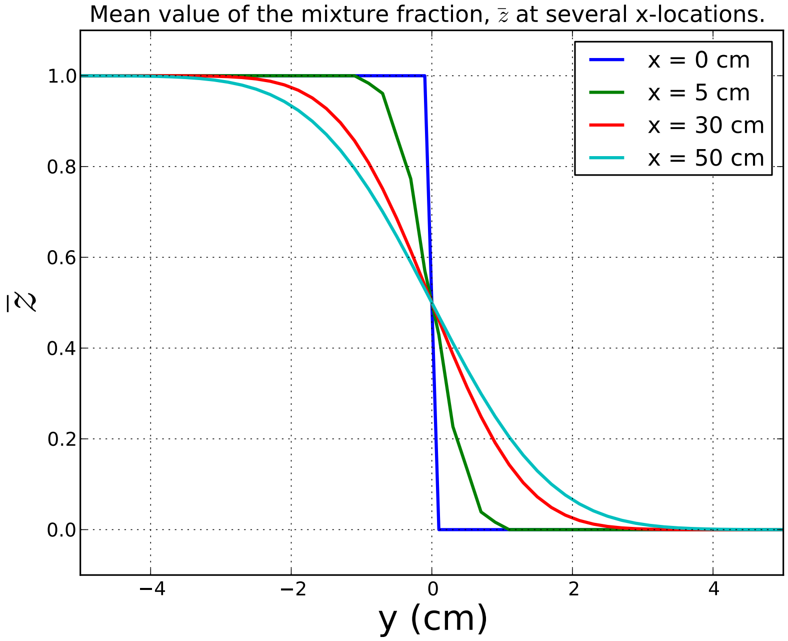

Figure 1: The mixture fraction plotted as a function of y.

direction of the derivative. However, upon running the code, I found that a forward finite difference was very numerically noisy for the solution at $x = 5$ cm. This was because the interface at this distance is still quite sharp, and so squaring the derivative “blew-up” the noise. I therefore switched a centered finite difference scheme, namely:

$$\frac{\partial \bar{z}}{\partial y} = \frac{\bar{z}_{i+1} - \bar{z}_{i-1}}{2\Delta y}. \tag{13}$$

The results are plotted in figure 2. The results of this figure display that at anything but low values of x, the variance is nearly zero. It is only near the sharp interface that the flow is turbulent. Outside of just a few centimeters, the variance is very nearly zero, implying essentialy laminar flow. As we move downstream, the pdf opens up, as the mixing layer grows and entrains more fluid around it. The peak also grows, implying that the turbulence grows to a higher reynolds number, with larger fluctuations.

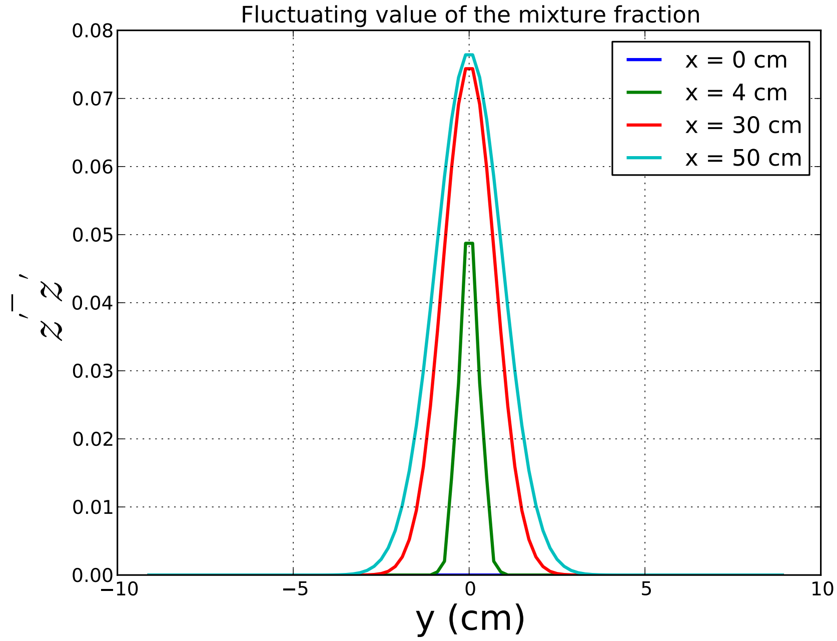

Figure 2: The mixture fraction variance plotted as a function of y.

c) Plot the PDF of the mixture fraction at two points, $y = 0$ cm, $x = 30$ and at $y = 15$ cm, $x = 30$ cm.

$P(z)$ = probability of z in $z + \Delta z$. If we knew the joint probability density function we could calculate it directly. However, we don't know P() (the PDF). Instead, we will use what Peters calls the "Presumed Shape PDF Approach". We must pick a probability distribution. We are limited to two parameter distributions for the model to be closed, because we only possess $\bar{z}$ and $z^{\bar{\prime}}z'$. Essentially, our choice is between a clipped Gaussian and the Beta function distribution. We will use the Beta distribution, as it should have much more appropriate limit behaviour for the mixing layer. In particular, due to the effect of intermittency, we expect the edges of the mixing layer to act like a delta function. This effect can be captured by the Beta distribution. The distribution function pdf has the form,

$$P(\bar{z}) = \frac{\bar{z}^{\alpha-1}(1-\bar{z})^{\beta-1}}{\Gamma(\alpha)\Gamma(\beta)}\Gamma(\alpha+\beta) \tag{14}$$

We further define,

$$\alpha = \bar{z}\gamma \tag{15}$$

$$\beta = (1 - \bar{z})\gamma \tag{16}$$

Where the variable γ is defined as,

$$\gamma = \frac{\bar{z}(1 - \bar{z})}{z'\bar{\,}z'^2} - 1 \geq 0. \tag{17}$$

For the two locations, we can already predict what the distributions will look like. The first, at $y = 0$, will be roughly gaussian, with a mean centered around the expected value of the mixture fraction at that location. Given that we expect the mixture fraction to be one half at that location, the mean should be around that value as well. For the location off-center, we expect a distribution that is skewed towards the side with more fuel or oxidizer. This is the oxidizer side, so we expect the mixture fraction pdf to have a mean less than zero. Furthermore, we note from figure 1 that this is far away from the mixing layer. We expect there to be essentially no mixing this far away, and therefore, the distribution should be essentially completely oxidizer, e.g. a distribution very sharply pushed up against zero.

The results of this when instantiated numerically are shown in figure 3. The results are precisely what we expected. The distribution at $x = 30, y = 0$ is not exactly at 0.5, but it is nearly so (0.53). The distribution is not skewed to either the fuel or oxidizer side. I expect that this is accurate. One might note that the beta distribution, despite how sharp it is near zero, is still giving non-trivial weight to less than zero mixture fractions. Personally, I am skeptical this is not an over-estimate, and it is likely a weakness of the assumed beta distribution for this edge case. However, note that assuming a Gaussian pdf would break down even more severely in this case.

d) Plot the laminar and mean turbulent temperature distributions at $x = 30$ cm.

We already know that the laminar profile will have a linear profile, from the Burke-Schumann solution. This takes the form,

$$az + b = T(z). \tag{18}$$

This is equivalent to saying that at each grid point, we expect the temperature distribution to be a delta function around the temperature predicted by the mixing fraction at that location.

The turbulent system is more complicated. Now, we expect the non-zero variance in mixing fraction concentration to impact the temperature. In order to account for the turbulent fluctuations, we need to integrate over the probability density, e.g.

$$\tilde{T} = \int P(\bar{z})T(\bar{z})dz \tag{19}$$

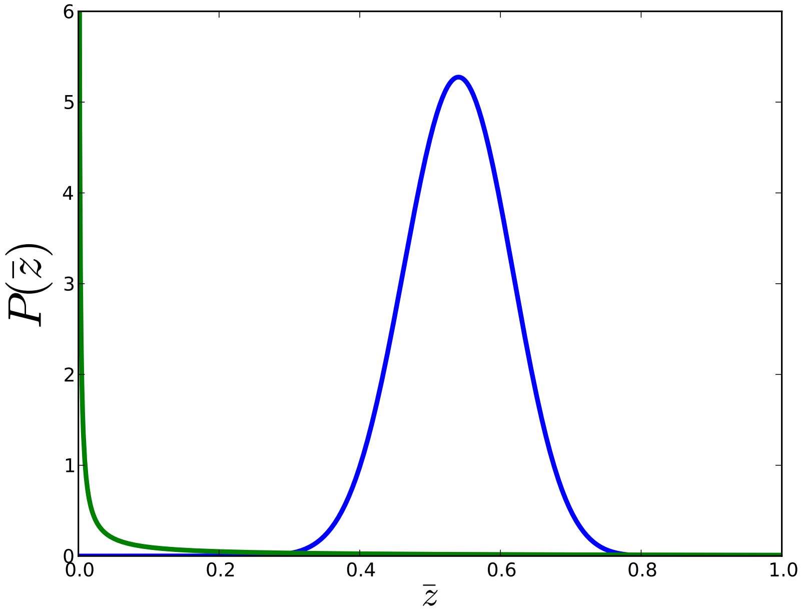

Figure 3: The distribution of the mixture fraction, using the assumed pdf approach (with an assumed β distribution). The green line on the right is at $x = 30, y = 0$, and the blue distribution is at $x = 30, y = 15$.

Using equation 18, this is equivalent to,

$$\tilde{T} = \int P(\bar{z})(a\bar{z} + b)dz \tag{20}$$

Essentially, this “blurs” the temperature profile by averaging each location with its neighboring temperatures.

These results are plotted in figure 4. These profiles are qualitatively similar to what we expected. The peak is lower for the turbulent profile, but the temperature peak is wider and the temperature is elevated farther away from the mixing layer, due to the increased mixing from the turbulence. I was surprised that the slope is not perfectly linear in the case of the laminar profile.

e) Discuss the results.

The entire formulation utilized chemical equilibrium models to find the long time stable solution. The model cannot predict extinction or ignition. This also assumes that the chemical time scales are much smaller than the turbulence time scales, e.g. that the Damkohler number is large.

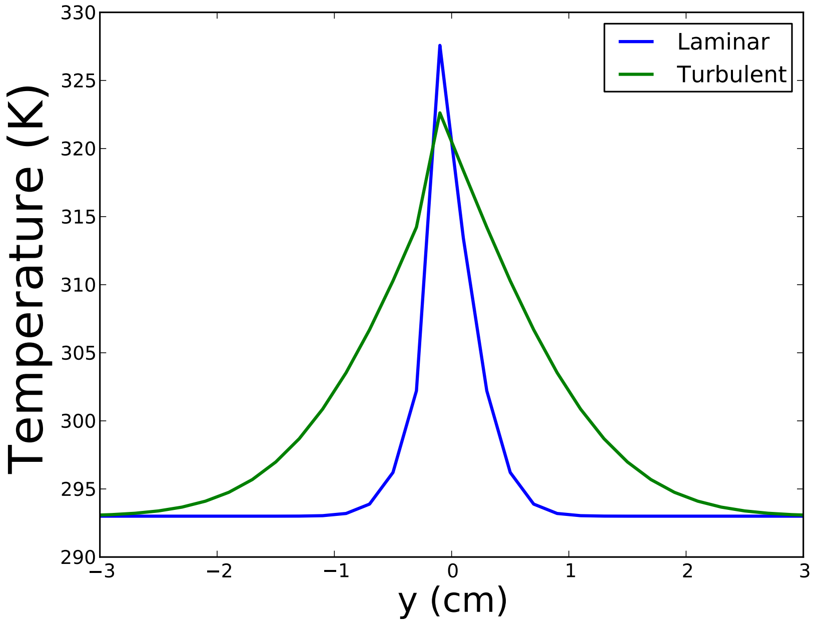

Figure 4: The temperature profiles for the laminar (blue) and turbulent (green) simulations at $x = 30$.

We are also ignoring instantaneous turbulence effects, on account of using a Favre-Averaging scheme, instead of resolving the fine turbulence scales. So these results are certainly not expected to be accurate instanteously, but only represent mean quantities. Furthermore, they may be far from realistic for some time, as the flow transitions to turbulence and large eddies form downstream of the splitter.

Finally, we either "turned-on" or "turned-off" the turbulence (e.g. it was fully developed turbulence, or completely laminar). In reality, the flow might be intermittent or laminar away from the mixing layer at $y = 0$ and turbulent near it. So a mixing of the models might be more appropriate. Ideally, the flow would be fully turbulent near the centerline, intermittent at the edges of the layer, and laminar outside. This could be more accurately modeled using the "composite PDF approach for intermittency".

Given these assumptions, we would have to be careful using the results of this model in a predictive context. Those caveats aside, the results have the correct qualitative character for a flame, and so are certainly much more useful than only expert opinion or conjecture. This is not to be too negative, this is an extremely complex system we are simulating, and I am able to generate all the plot used in this report in a few seconds on my computer.

All the work contained in this report was entirely my own.
Thank you for the class!

References:
“Turbulent Combustion”, Norbert Peters
“Combustion Physics”, Chung K. Law

Code

I wrote these routines entire from scratch, using python 2.X. The only libraries necessary to run these routines should be Numpy, SciPy and Matplotlib.

```
#!/bin/py
#
# Combustion Theory Final Exam: Turbulent Diffusion Flames
#
import sys
import numpy as np
import pylab
import scipy.special as ss

def beta(a, b, mew):
    e1 = ss.gamma(a + b)
    e2 = ss.gamma(a)
    e3 = ss.gamma(b)
    e4 = mew ** (a - 1)
    e5 = (1 - mew) ** (b - 1)
    return (e1/(e2*e3)) * e4 * e5

def plot_beta(a, b):
    Ly = []
    Lx = []
    mews = np.mgrid[0:1:1000j]
    for mew in mews:
        Lx.append(mew)
        Ly.append(beta(a, b, mew))
    pylab.plot(Lx, Ly, linewidth=3.0)

#
# main function
#
```

```
if __name__ == "__main__":
    """ run a simulation, and plot everything up"""
    #
    # define field
    #
    nx = 50
    ny = 91

    # sanity check
    if(ny%2 != 1):
        print 'ny must be odd!'
        print ny
        print sys.exit(1)

    # initialize
    z = np.zeros(ny)

    #
    # enforce b.c. ( x < 0, y = 0 => z=1)
    #
    z[0:ny/2 +1] = 1.0

    # spacing
    dx = 1 # centimeters
    dy = 0.2 # centimeters

    # Diffusivities
    DL = 0.176 # (cm^2/sec)
    DT = 10*DL

    u = 100 # 100 centimeters = 1 m/s

    # ---------------------------------------------------------
    #
    # spatial iteration loop!
    #
    # ---------------------------------------------------------
    zf = []
    zf.append(z)

    zlf = []
    zlf.append(z)

    zzf = []
    zzf.append(np.zeros(ny))
```

```
zl = z

for i in xrange(nx):

    zt = np.zeros(ny)
    zt[0] = 1.0

    zlt = np.zeros(ny)
    zlt[0] = 1.0

    for j in xrange(1,ny-1):
        zt[j]  = DT*dx*(z[j+1]-2*z[j]+z[j-1])/(u*dy*dy) + z[j]
        zlt[j] = DL*dx*(zl[j+1]-2*zl[j]+zl[j-1])/(u*dy*dy) + zl[j]

    #
    # update mean field  and save state
    #
    z = zt
    zl = zlt

    zlf.append(zl)
    zf.append(z)

    #
    # calculate fluctuation \bar (z'z')
    #
    zzt = np.zeros(ny)
    for j in xrange(1,ny-1):
        zzt[j] = 0.25*((z[j+1]-z[j-1])/dy)**2 * DT * (i*dx)/u

    zzf.append(zzt)

# ------------------------------------------------------------------
#
# plot solution of mean field
#
# ------------------------------------------------------------------

y = np.arange(-dy*ny/2., dy*ny/2., dy)
ind=0
t='x = '+str(ind*dx)+' cm'
pylab.plot(y,zf[ind], linewidth=2.0, label=t)
pylab.xlabel('y (cm)',size=22.0)
pylab.ylabel(r'$\bar z$', size=30.0)
pylab.title(r'Mean value of the mixture fraction, $\bar z$ at several x-loca
pylab.grid(True)
```

```
#
# 5 cm, 30 cm and 50 cm
#
ind=5
t='x = '+str(ind*dx)+' cm'
pylab.plot(y,zf[ind], linewidth=2.0, label=t)

ind=30
t='x = '+str(ind*dx)+' cm'
pylab.plot(y,zf[ind], linewidth=2.0, label=t)

ind=50
t='x = '+str(ind*dx)+' cm'
pylab.plot(y,zf[ind], linewidth=2.0, label=t)


#
# plot mean field
#
pylab.xlim([-5,5])
pylab.ylim([-0.1,1.1])
pylab.legend()
pylab.savefig('mean.pdf')
pylab.close()

# ---------------------------------------------------------------
#
# calculate and plot fluctuation!
#
# ---------------------------------------------------------------

ind=0
t='x = '+str(ind*dx)+' cm'
pylab.plot(y,zzf[ind], linewidth=2.0, label=t)
pylab.xlabel('y (cm)',size=22.0)
pylab.ylabel(r'$\bar{z^{\prime}z^{\prime}}$', size=30.0)
pylab.title(r'Fluctuating value of the mixture fraction')
pylab.grid(True)

ind=4
t='x = '+str(ind*dx)+' cm'
pylab.plot(y,zzf[ind], linewidth=2.0, label=t)

ind=30
t='x = '+str(ind*dx)+' cm'
```

```
pylab.plot(y,zzf[ind], linewidth=2.0, label=t)

ind=50
t='x = '+str(ind*dx)+' cm'
pylab.plot(y,zzf[ind], linewidth=2.0, label=t)

#
# plot fluctuation
#
pylab.legend()
pylab.savefig('fluc.pdf')
pylab.close()

# ---------------------------------------------------------------
#
# calculate and plot PDF of mixture fraction!
#
# ---------------------------------------------------------------

#
# y = 0, x = 30 (should look gaussian)
#
ind = 30
yloc = ny/2.
zbar  = zf[ind][yloc]
zzbar = zzf[ind][yloc]
gamm = (zbar * (1 - zbar ) / (zzbar*zzbar) ) - 1
if(gamm < 0):
    gamm = 0
alph = zbar * gamm
bet  = (1-zbar)*gamm
plot_beta(alph, bet)

#
# plot at y = 15, x = 30 (should look like a delta function)
#
ind = 30
yloc = ny-1
zbar  = zf[ind][yloc]
print zbar
zzbar = zzf[ind][yloc]
gamm = (zbar * (1 - zbar ) / (zzbar*zzbar) ) - 1
alph = zbar * gamm
bet  = (1-zbar)*gamm
plot_beta(alph, bet)
```

```
#
# generic plot options
#
pylab.xlabel(r'$\bar z$',size=22.0)
pylab.ylabel(r'$P(\bar z)$', size=30.0)

pylab.xlim(0.0, 1.0)
pylab.ylim(0.0, 6.0)
pylab.legend()
pylab.savefig('pdf.pdf')
pylab.close()

# ——————————————————————————————————————————
#
# calculate and plot temperature profiles for laminar and turbulent profiles
#
# ——————————————————————————————————————————

# laminar:
#
tl  = zlf
tu  = 293. # room temp
Q   = 55

# CP = 1.00 kJ/kg.K
cp  = 1.00
nuf = 1
wf  = 1

#
# location = 30 cm
#
ind = 30
trl = tu + zlf[ind]*Q/(cp*nuf*wf)
tll = tu + (1−zlf[ind])*Q/(cp*nuf*wf)
tll[ny/2:]=trl[ny/2:]
pylab.plot(y,tll, linewidth=2.0, label='Laminar')


#
# turbulent
#
tr = tu + zf[ind]*Q/(cp*nuf*wf)
tl = tu + (1−zf[ind])*Q/(cp*nuf*wf)
tl[ny/2:]=tr[ny/2:]
pylab.plot(y,tl, linewidth=2.0, label='Turbulent')
```

```
    #
    # plot
    #
    pylab.xlim(-3.0, 3.0)
    pylab.xlabel('y (cm)',size=22.0)
    pylab.ylabel('Temperature (K)', size=30.0)
    pylab.legend()
    pylab.savefig('temperature.pdf')

#
# nick
# 5/9/14
#
```