

MEA712: Mesoscale Modeling

Class mesoscale model (CMM) project, assignment 4

Due at 5 PM on Wednesday, 4 November.

In-class supervised work period on 27 October (and 3 November if needed).

We will be adding Asselin filtering and computational diffusion to CMM. When you are done, the processes for each time step should occur in this order...

- variables are updated using equations implemented in assignment 3
- apply computational diffusion
- apply Asselin filtering
- apply boundary conditions
- write new variable values to your “history” file

1. Implement an Asselin filter in the model. Recall that the formulation for Asselin filtering is given by:

$$u(i,k) = u(i,k) + \text{asscoef} * (u_p(i,k) - 2 * u(i,k) + u_m(i,k))$$

where I have used Fovell’s convention for u_p , u , and u_m . This should be applied after each time step to all variables on all physical points. For starters, use a value of `asscoef=0.1`.

2. Implement computational diffusion in the model. Recall from class that the basic formulation for diffusion is:

$$u_p(i,k) = u_p(i,k) + d2t * (\& \\ + \text{kmixh} * (u_m(i+1,k) - 2.0 * u_m(i,k) + u_m(i-1,k)) / (dx * dx) \& \\ + \text{kmixv} * (u_m(i,k+1) - 2.0 * u_m(i,k) + u_m(i,k-1)) / (dz * dz))$$

where I have used Fovell’s convention for u_p , and u_m . This should be applied as a part of each time step. You should apply it to all predicted variables **except** π' . *Important:* horizontal diffusion should be applied on all physical points from $k=2$ to $nz-1$, **but vertical** diffusion should be applied on all physical points from $k=3$ to $nz-1$. This is done to prevent the model from diffusing perturbations through the bottom boundary (i.e. the ground).

The mixing coefficients should be computed in your diffusion code as follows:

$$\text{kmixh} = \text{cmixh} * dx * dx / dt \\ \text{kmixv} = \text{cmixv} * dz * dz / dt$$

Sidebar: In class, we defined the Fourier number as:

$$\gamma = \frac{K \Delta t}{(\Delta x)^2}$$

In the above sample code, kmixh and kmixv are the horizontal and vertical representations of what we called K in the class lectures. These are the values that are directly used in the diffusion scheme. Meanwhile, cmixh and cmixv are the horizontal and vertical representations of γ . As in class, to ensure stability of a 2D diffusion scheme we require $\gamma \leq 0.125$. In practice, we choose some fraction of this value to ensure stability and to select how strongly the filter will damp. We then back K (kmixh and kmixv) out from γ (cmixh and cmixv) for use in the code, as shown above.

Initially, we will set c_{mixh} and c_{mixv} to be the same value, 0.003. Based on our current model set-up, this will yield $k_{hdif}=k_{vdif}=240 \text{ m}^2/\text{s}$. The tunable parameters are c_{mixh} and c_{mixv} , which are measures of how close we are to the limit for stability (recall that it is 0.125, so we are starting at about 1/40th of that value). Multiplying by dx^2 (or dz^2) and dividing by dt then gives the coefficient proper units and ensures that it can be stably applied to any grid spacing.

3. Rerun the assignment 3 simulations for 1800 s, turning only one of the two new features on at a time. Make plots (i.e. the plots we made for assignment 3) corresponding to $t=1200$ s and $t=2400$ s for each of your two runs. Then, comment briefly on the impacts of each new feature that we have added. You may wish to animate your various simulations in order to observe different transient behaviors in the model.

4. Play around with the coefficients for the filters. Rerun your 2400 s simulation (with both filters turned on) and report on the nature of the experiments you've tried and any unusual behaviors observed.

5. Rerun the above simulation (with both filters) on a much wider domain (i.e. 400 points or so), and with a cold bubble instead of a warm bubble. In other words, change the temperature perturbation of the bubble from +3K to -3K. Run the model for 1500 s and make a plot of θ' with wind vectors (the u and w components) superposed. Be sure to zoom in on the area of interest. In GrADS, here is the method:

```
set x 150 250
d ptpvt
set gxout vector
display u:w
```