Simulation and visualization of simple leapfrog advection scheme

ATMO 558

Term Project

Koichi Sakaguchi

Outline

1. Motivation
2. Method 2D
    a) Equations
    b) Model domain & grid setup
    c) Initial condition
    d) Integration
3. Result 2D
4. Method 3D
5. Result 3D
6. Conclusion

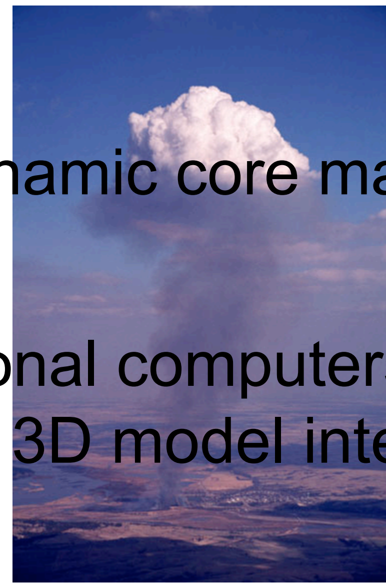
Motivation

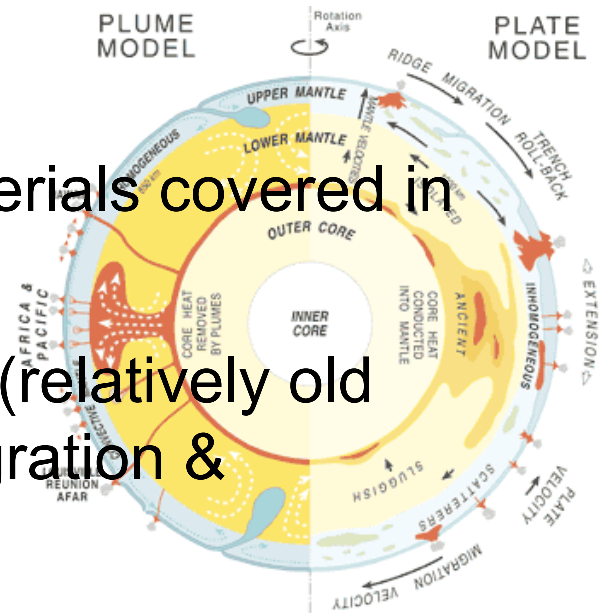
•Looks cool

•Sounds smart

•Good summary of the dynamic core materials covered in this course

•See how Matlab on personal computers (relatively old macs) can handle 2D and 3D model integration & visualization





- A priori experience: 10 > hours on EOF from global NCEP reanalysis data

-The materials presented are from Chapter 11 (Model Task #3) and Chapter 13 (Model Task #5) of Dr.Fovell’s class notes

-The visualization codes I wrote for this presentation are in Appendices for reference

Method - equations (2D)

$$\frac{\partial u}{\partial t} = -\frac{\partial uu}{\partial x} - \frac{1}{\overline{\rho}}\frac{\partial \overline{\rho} uw}{\partial z} - C_{pd}\overline{\theta}\frac{\partial \pi'}{\partial x}$$

$$\frac{\partial w}{\partial t} = -\frac{\partial uw}{\partial x} - \frac{1}{\overline{\rho}}\frac{\partial \overline{\rho} ww}{\partial z} - C_{pd}\overline{\theta}\frac{\partial \pi'}{\partial z} + g\frac{\theta'}{\overline{\theta}}$$

$$\frac{\partial \theta'}{\partial t} = -\frac{\partial u\theta'}{\partial x} - \frac{1}{\overline{\rho}}\frac{\partial \overline{\rho} w\theta'}{\partial z} - w\frac{d\overline{\theta}}{dz}$$

$$\frac{\partial \pi'}{\partial t} = -\frac{\overline{c}_s^2}{\overline{\rho} c_{pd}\overline{\theta}^2}\left[\overline{\rho}\overline{\theta}\frac{\partial u}{\partial x} + \frac{\partial \overline{\rho}\overline{\theta} w}{\partial z}\right]$$

- Everything as given in Dr.Fovell's note
- Matlab indexing is the same as that of Fortran
- The code has about 300 lines

•No moisture

•No adiabatic processes

•No diffusion/friction

•No Coriolis effect

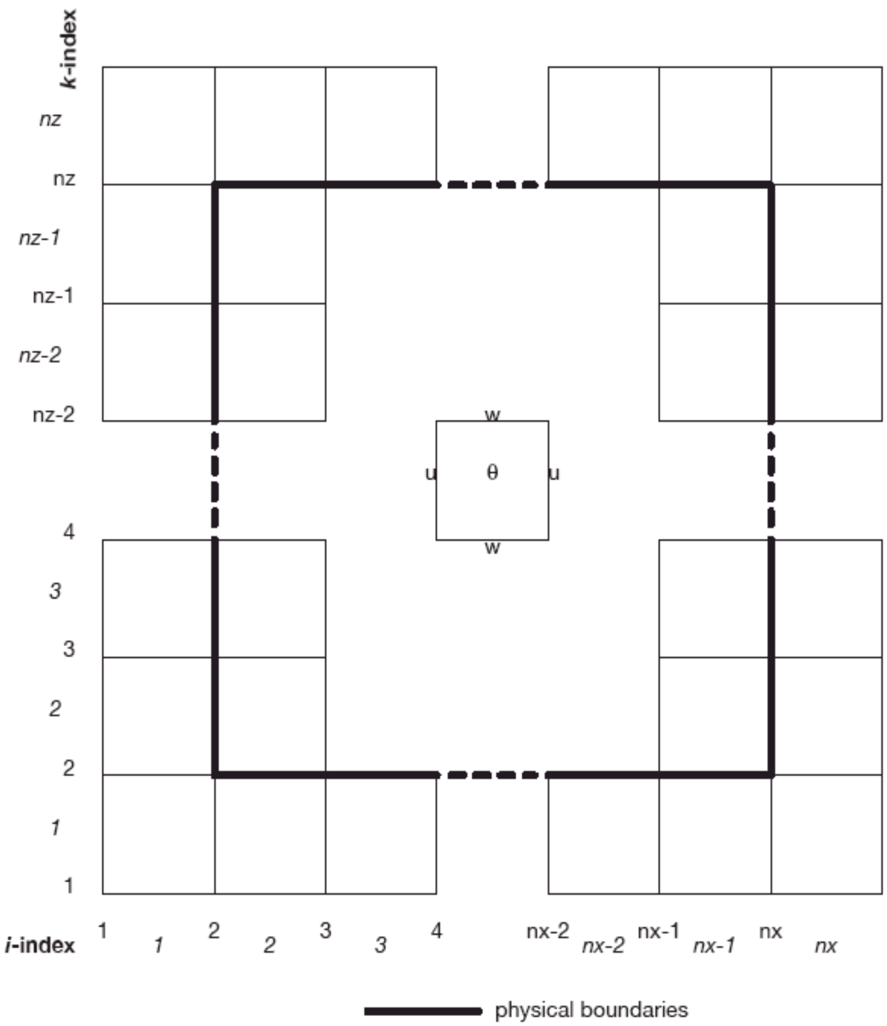
•Base-state is in hydrostatic balance

•Normalized pressure is used for pressure gradient force.

•Normalized pressure tendency eqn. instead of the continuity equation.

•Flux form instead of advection form (semi-anelastic atmosphere is assumed)

Model domain & grid setup (2D)



- Arakawa C grid staggering
- Dimensions:
  - 83 grid points in x direction (~33 km length)
  - 42 grid points in z direction (~17 km height)
- Grid size:
  - dx = dz = 400 m
- Boundary conditions:
  - Top & Bottom: Rigid, free-slip (w=0) and zero-gradient to other variables
  - Lateral: cyclic

Arakawa C grid & leapfrog scheme

Example: u-momentum

$$\frac{\partial u}{\partial t} = -\frac{\partial uu}{\partial x} - \frac{1}{\overline{\rho}}\frac{\partial \overline{\rho} uw}{\partial z} - C_{pd}\overline{\theta}\frac{\partial \pi'}{\partial x}$$

predict u at (i,k)

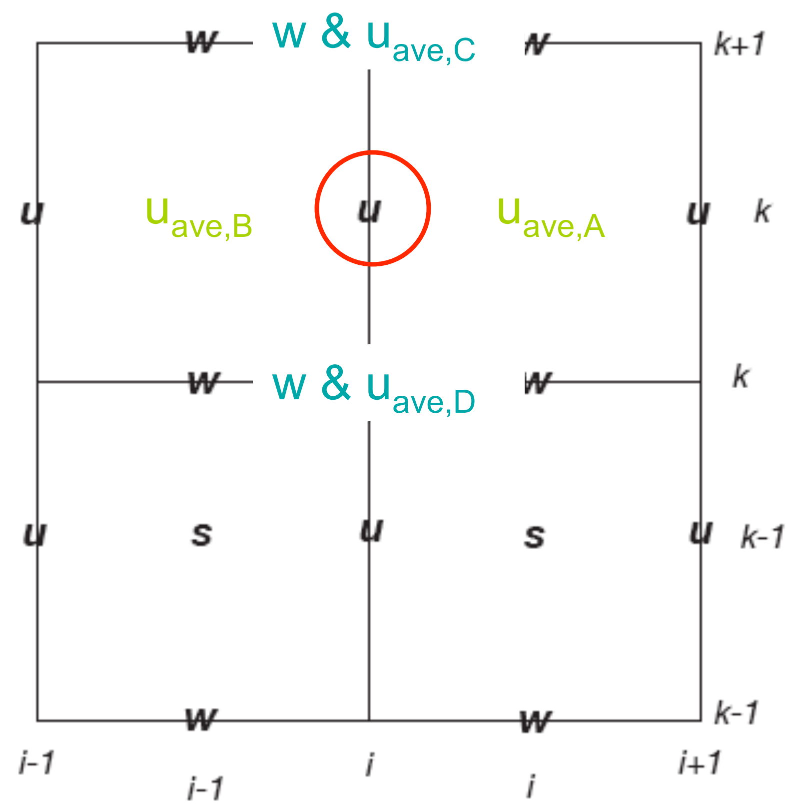
$$u_{i,k}^{n+1} = 2\Delta t\left[u_{i,k}^{n-1} - \frac{1}{\Delta x}\left(\left[\frac{u_{i+1,k}^{n} + u_{i,k}^{n}}{2}\right]^2 - \left[\frac{u_{i,k}^{n} + u_{i-1,k}^{n}}{2}\right]^2\right)\right.$$

$u_{ave,A}$ $u_{ave,B}$

$$-\frac{1}{\overline{\rho}_{u,k}\Delta z}\left(\overline{\rho}_{w,k+1}\frac{\left(w_{i,k+1}^{n} + w_{i-1,k+1}^{n}\right))}{2}\frac{\left(u_{i,k+1}^{n} + u_{i,k}^{n}\right)}{2} - \overline{\rho}_{w,k}\frac{\left(w_{i,k1}^{n} + w_{i-1,k}^{n}\right))}{2}\frac{\left(u_{i,k}^{n} + u_{i,k-1}^{n}\right)}{2}\right)$$

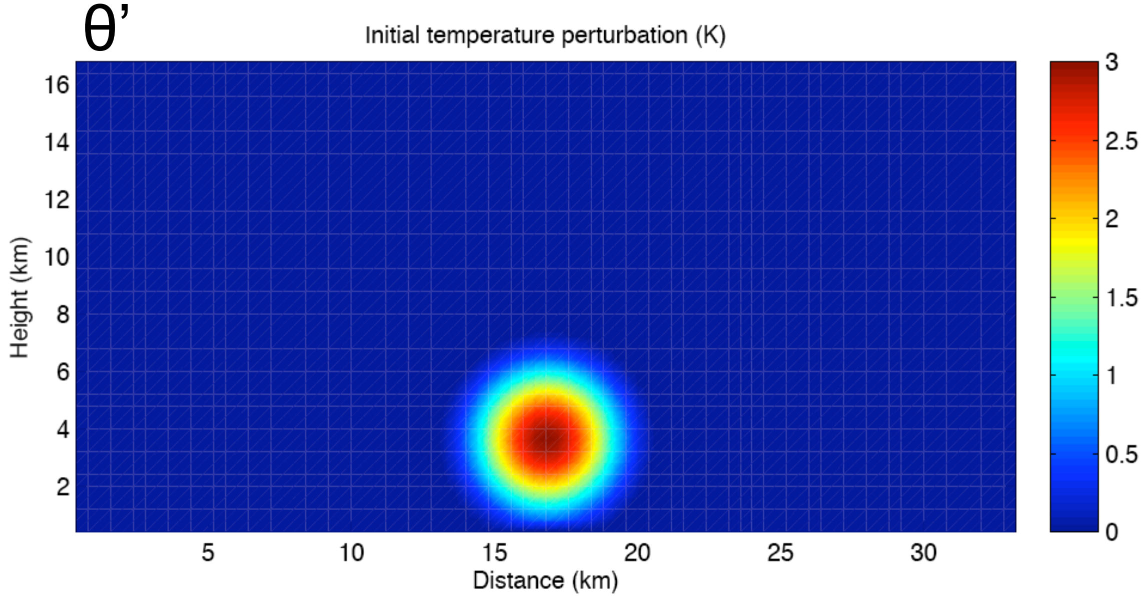
w & $u_{ave,C}$ w & $u_{ave,D}$

$$\left.-c_{pd}\overline{\theta}_k\frac{1}{\Delta x}\left[\pi_{i,k}^{n} - \pi_{i-1,k}^{n}\right]\right]$$



Initial condition (2D)

θ’

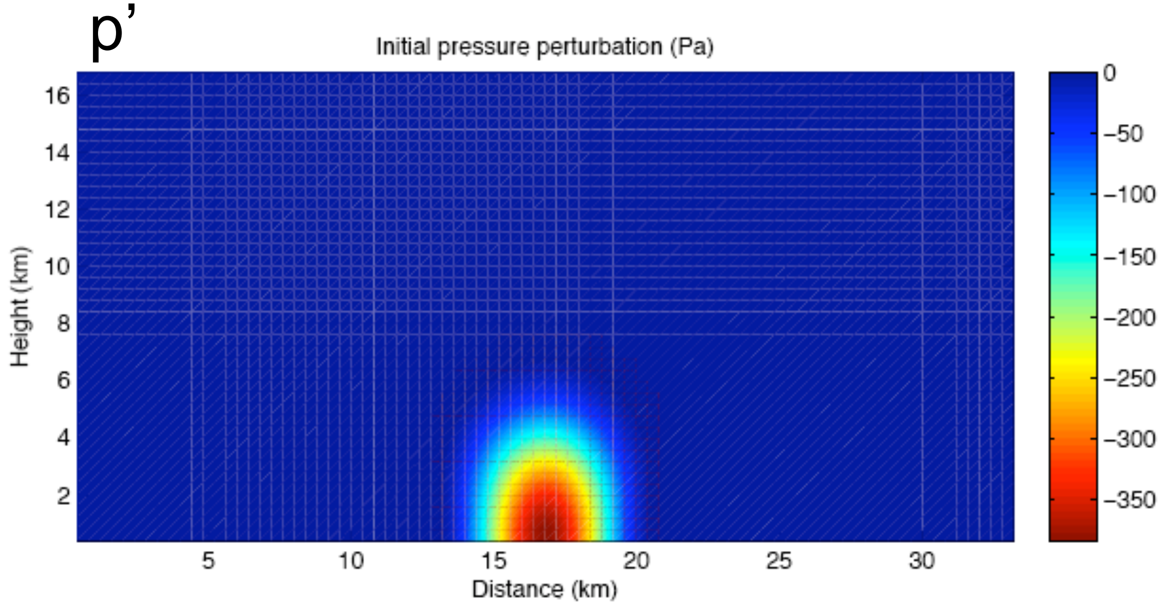


u,w = 0 (m/s)

Thermal perturbation:

- Radius of 4000m
- max θ’ = 3 (K)

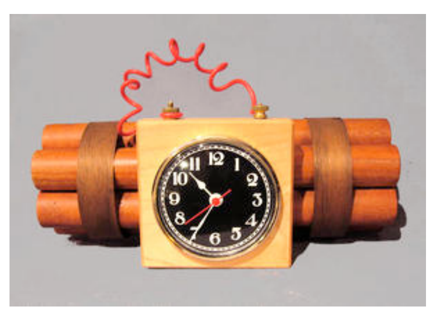
p’



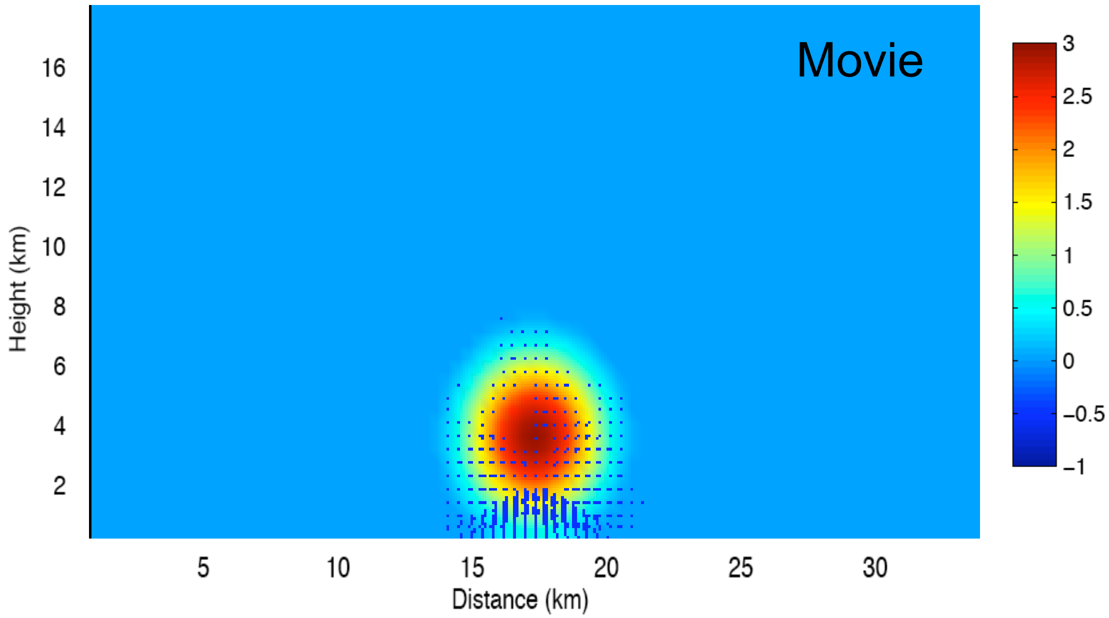
p’ field is adjusted to be in hydrostatic balance with θ’

Integration

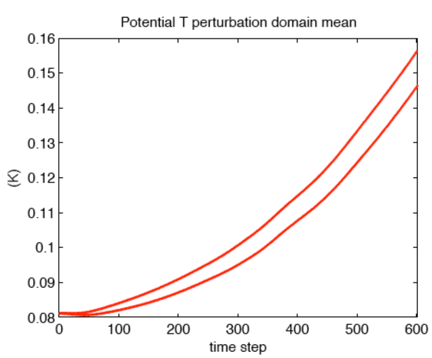
- 2 sec time step
- Integrated for 1200 sec
- Mean variables are function of height only:
  - $\overline{\theta}$ = 300 (K) at all height
  - $\overline{p}$ = 965 (mb) at the surface
- Machine: 3 years old imac: G5 power PC 1.6 GHz with 1.5 GB RAM



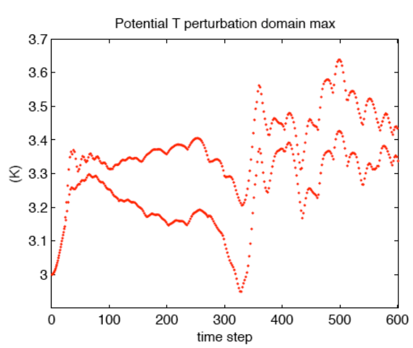
Results (2D)



θ' (K)

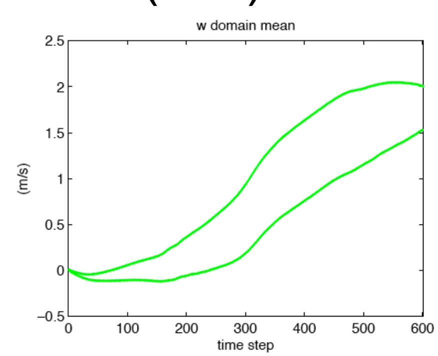


mean

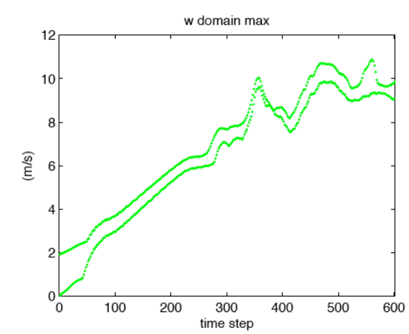


max

w (m/s)



mean

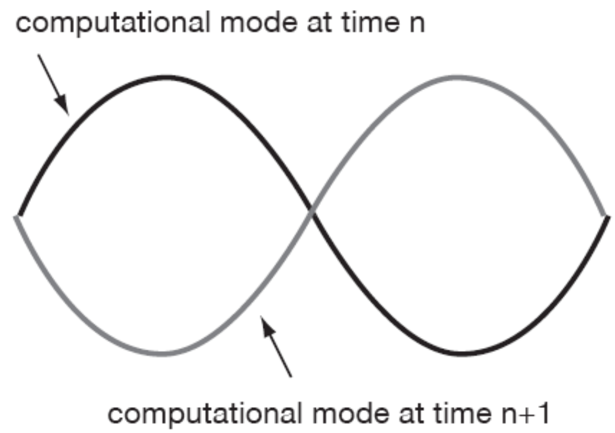


max

Computational Mode & Robert-Asselin filter

Leapfrog scheme produces two solutions:

$$u_j^n = Ae^{ikx}\left[\underbrace{\langle\frac{(1+\cos\theta)}{2\cos\theta}\rangle e^{-i\theta n}}_{\text{Physical mode}} - (-1)^n\underbrace{\langle\frac{(1-\cos\theta)}{2\cos\theta}\rangle e^{i\theta n}}_{\text{Computational mode}}\right]$$



From Dr.Fovell's note

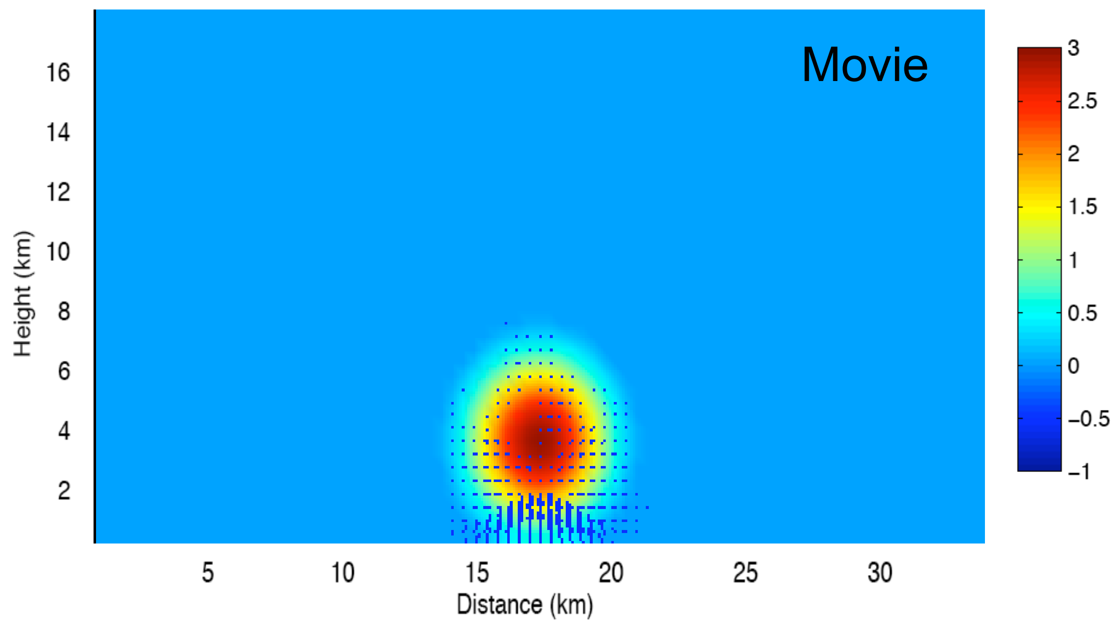
Robert-Asselin filter to cancel out the computational mode

$$u_j^{n+1*} = u_j^{n-1} - c'(u_{j+1}^{n*} - u_{j-1}^{n*})$$ — usual leapfrog scheme

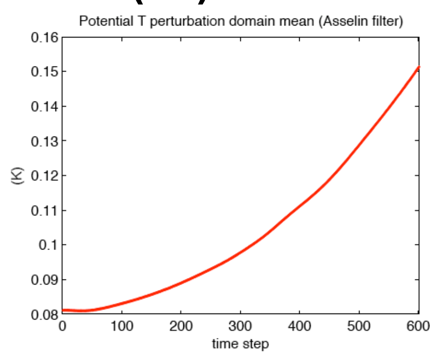
$$u_j^{n} = u_j^{n*} + \epsilon\left[u_j^{n+1*} - 2u_j^{n*} + u_j^{n-1}\right]$$ — Time-centered smoothing

Diffusion coefficient ε = 0.1

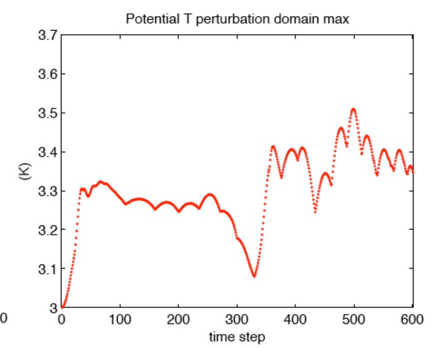
Results (2D): Asselin filtered



θ' (K)

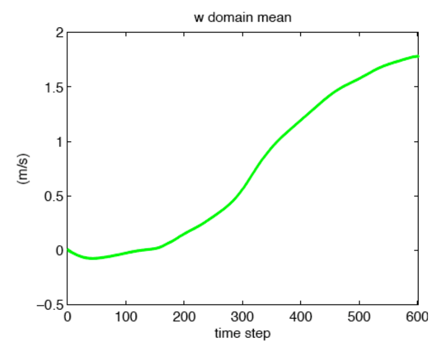


mean

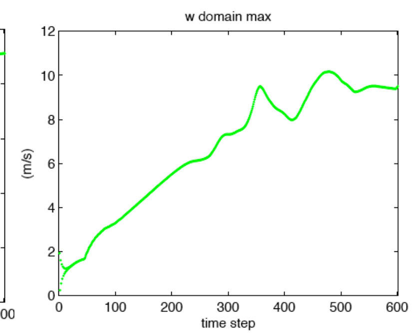


max

w (m/s)



mean



max

Results (2D)

Model integration:

•Output file size:

83(x) x 42(z) x 600(t) x 4(variables) + other constants:
44~70MB (.mat file)

•Integration time:

10 minutes

Model visualization:

•Machine:

2yrs old macbook:Intel Core Duo 1.83GHz with 2GB RAM
(upgraded for this assignment)

•2D Movie making:

5 minutes

File size about 200MB

Method - equations (3D)

$$\frac{\partial u}{\partial t} = -\frac{\partial uu}{\partial x} - \boxed{\frac{\partial uv}{\partial y}} - \frac{1}{\overline{\rho}}\frac{\partial \overline{\rho} uw}{\partial z} - C_{pd}\overline{\theta}\frac{\partial \pi'}{\partial x}$$

$$\boxed{\frac{\partial v}{\partial t} = -\frac{\partial uv}{\partial x} - \frac{\partial vv}{\partial y} - \frac{1}{\overline{\rho}}\frac{\partial \overline{\rho} vw}{\partial z} - C_{pd}\overline{\theta}\frac{\partial \pi'}{\partial y}}$$

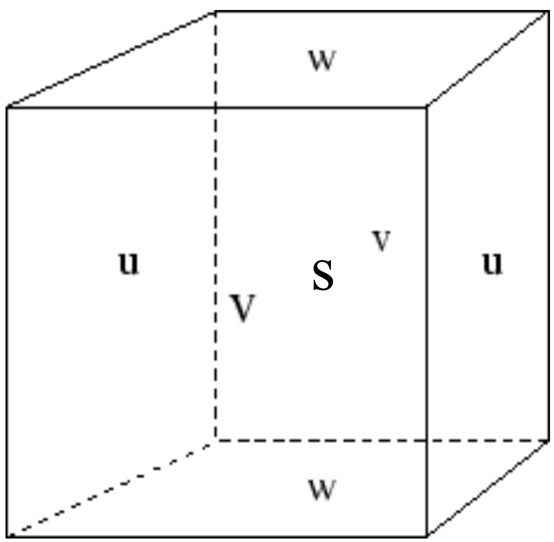
$$\frac{\partial w}{\partial t} = -\frac{\partial uw}{\partial x} - \boxed{\frac{\partial vw}{\partial y}} - \frac{1}{\overline{\rho}}\frac{\partial \overline{\rho} ww}{\partial z} - C_{pd}\overline{\theta}\frac{\partial \pi'}{\partial z} + g\frac{\theta'}{\overline{\theta}}$$

$$\frac{\partial \theta'}{\partial t} = -\frac{\partial u\theta}{\partial x} - \boxed{\frac{\partial v\theta'}{\partial y}} - \frac{1}{\overline{\rho}}\frac{\partial \overline{\rho} w\theta'}{\partial z} - w\frac{d\overline{\theta}}{dz}$$

$$\frac{\partial \pi'}{\partial t} = -\frac{\overline{c}_s^2}{\overline{\rho} c_{pd}\overline{\theta}^2}\left[\overline{\rho}\overline{\theta}\frac{\partial u}{\partial x} + \boxed{\overline{\rho}\overline{\theta}\frac{\partial v}{\partial y}} + \frac{\partial \overline{\rho}\overline{\theta} w}{\partial z}\right]$$

- Y-momentum conservation is added
- Y component is added on each equation

Model domain & grid setup (3D)



- Arakawa C grid staggering
- Grid size:
  - dx = dy = dz = 400 m
- Boundary conditions:
  - Top & Bottom: Rigid, free-slip (w=0) and zero-gradient to other variables
  - Lateral: cyclic

Same as 2D

83 grids in x
83 grids in y
42 grids in z
> 600 timesteps

→ "out of memory" error in matlab

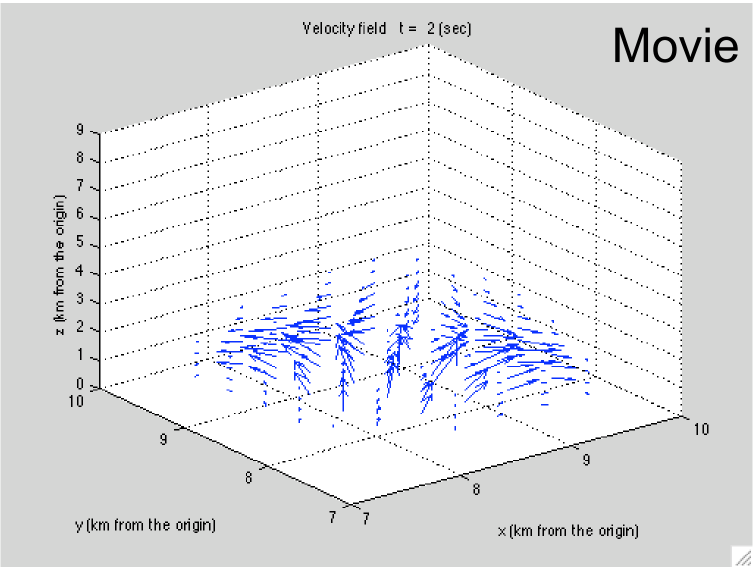
63 grids in x
63 grids in y
32 grids in z
300 timesteps

→ ~ 6hours for integration
~ 500 MB data size

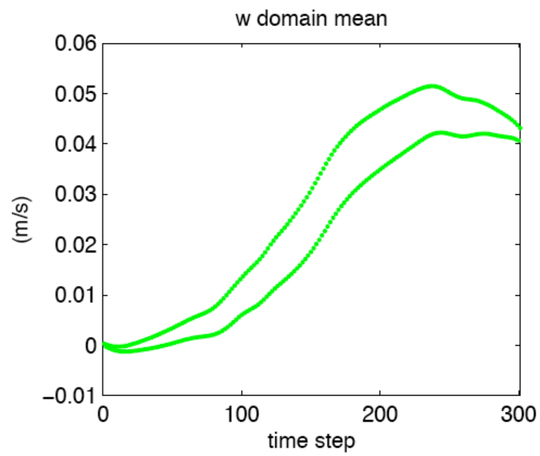
43 grids in x
43 grids in y
22 grids in z
300 timesteps

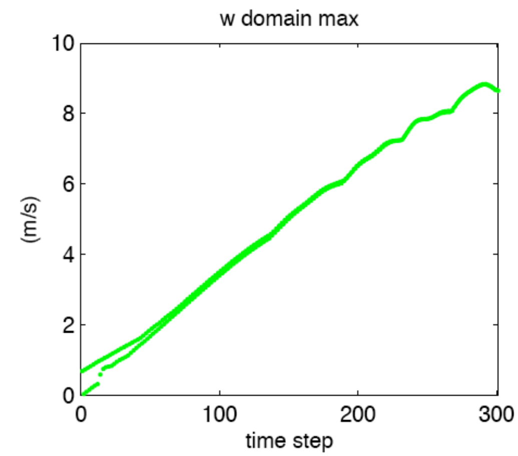
→ ~ 2hours for integration
~ 200 MB data size

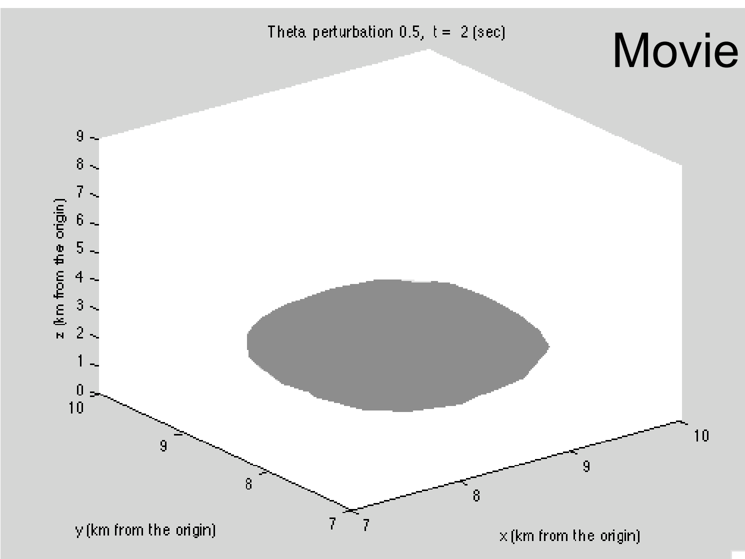
Results (3D): velocity and θ’

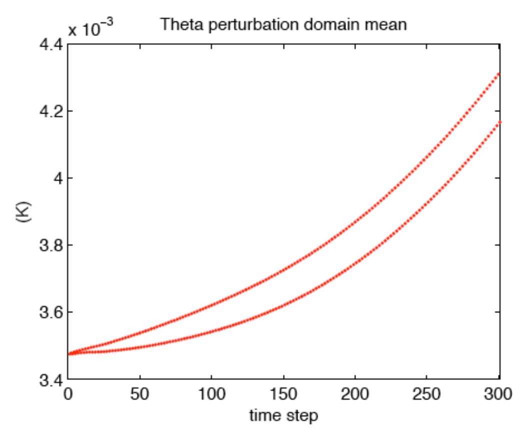


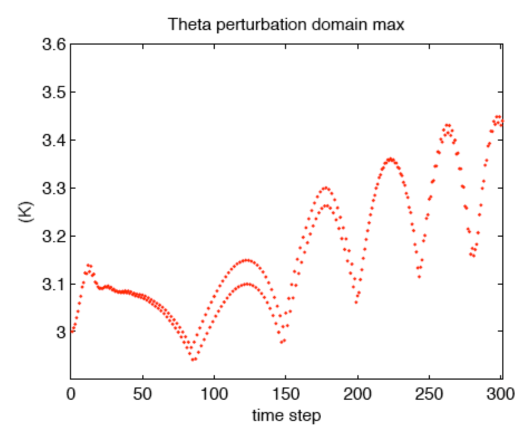
No filter



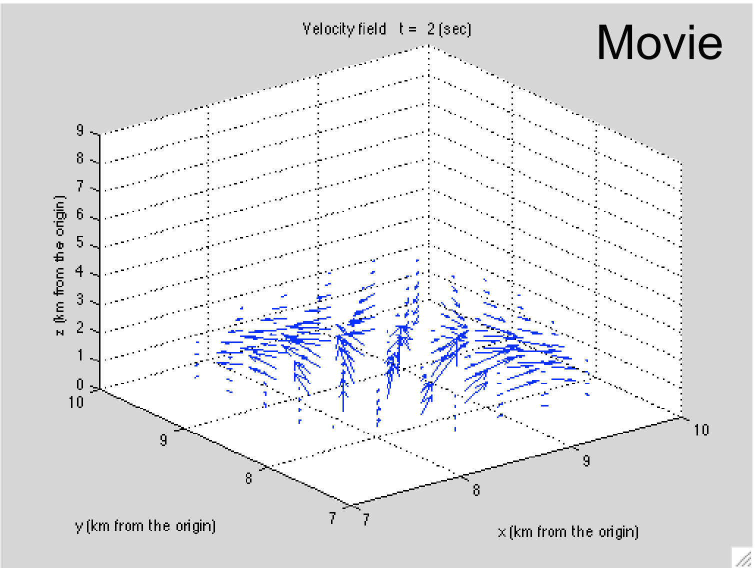


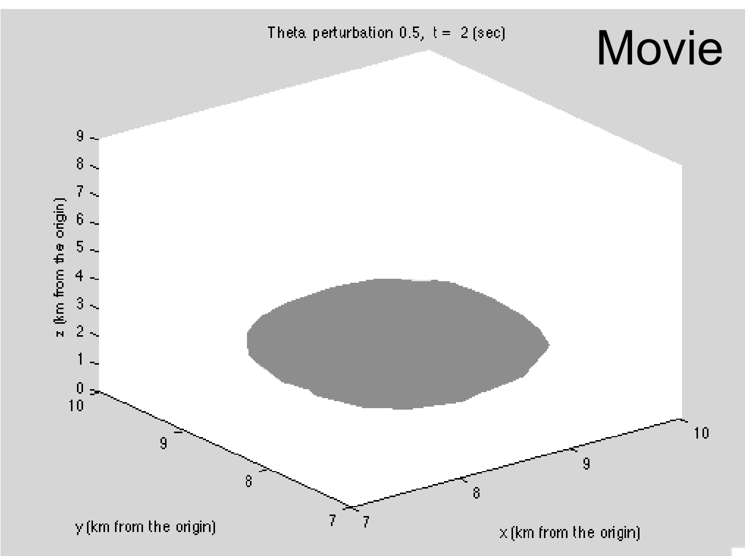




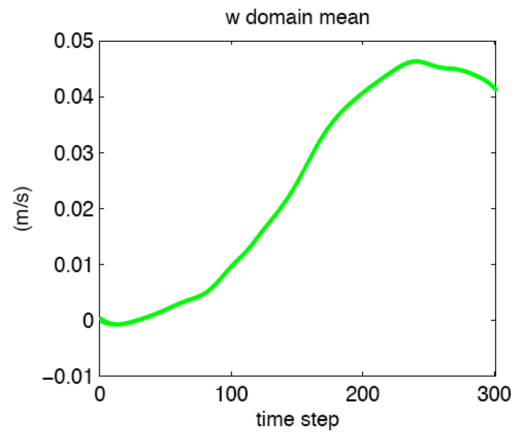


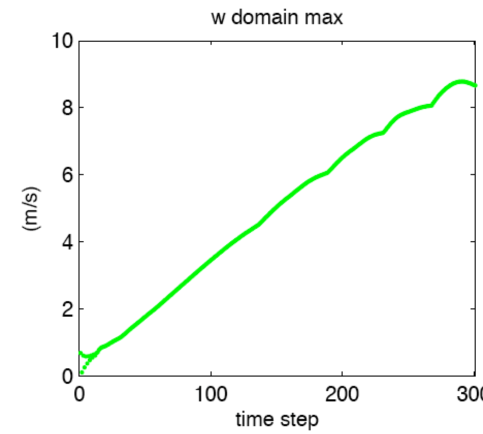
Results (3D): velocity and θ’

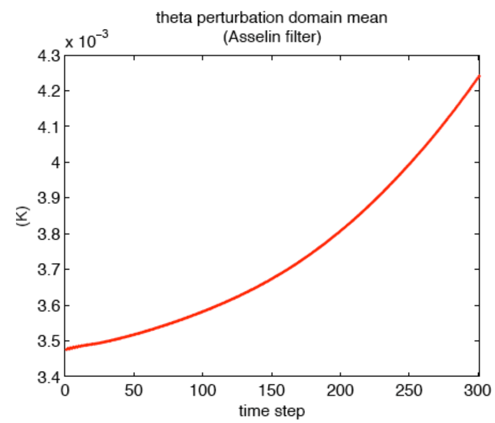


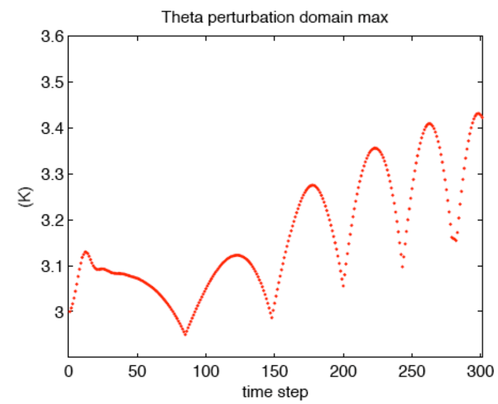


Robert-Asselin filter



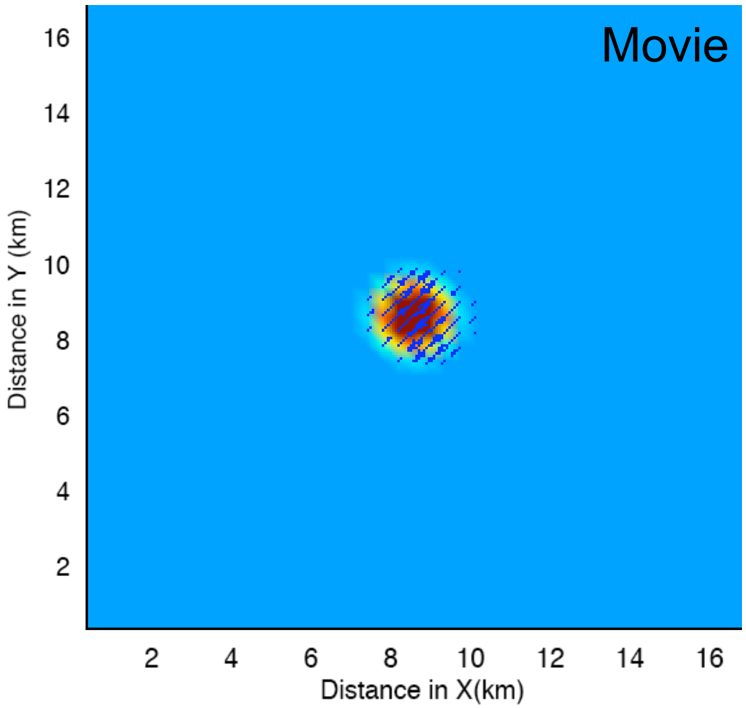




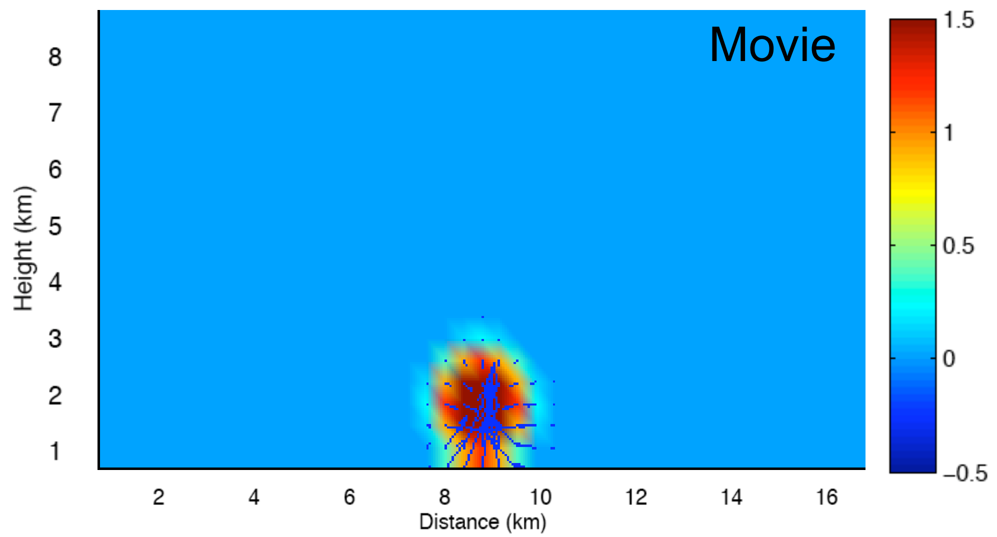


Results (3D): Dynamics

Velocity (u & v) and θ' at 1200 m height



Velocity (u & w) and θ' at perturbation center



Results (3D)

Model integration:

- Output file size:

  43(x) x 43(y) x 32(z) x 300(t) = 17,750,400 x 4(variables) + other constants: ~200 MB (.mat file)

- Integration time:

  2 hours

Model visualization:

- 3D Movie making:

  - Full domain

  >30 min for 3D quiver plot of one time slice

  -7(x) x 7(y) x 22(z) box at the center of the whole model domain

  5 minutes

  File size about 170MB

Conclusion

1. Simple 2D leapfrog model is successfully simulated (thermal circulation, waves)
2. Simple 3D leapfrog model was too much to integrate using Matlab. The numerical weather model are written in Fortran with good reasons
3. Robert-Asselin filter with $\varepsilon = 0.1$ successfully eliminated the computational mode
4. 3D visualization is not necessarily more useful than 2D plots: computing resource limit (domain size) & only one contour (e.g., isentropic surface)

Appendix A: Initialization plot (‘pcolor’)

```
pcolor(x*dx/1000,z*dz/1000,th(:,:,1))
axis image
shading interp
colorbar
title('Initial temperature perturbation (K)')
ylabel('Height (km)')
xlabel('Distance (km)')
```

Appendix B: 2D animation plot ('pcolor', 'quiver', 'getframe')

```
[X,Z] = meshgrid(x,z);                      %create 2D x and z grid from position vector
X = X';                                     %without transposing pcolor and quiver plot won't match up
Z = Z';
aviobj = avifile('uw_2d_filter.avi');        %create & open an .avi movie file
figure('position',[100 100 500 250])         %set the figure size

for i=1:nstep                                %start for loop to capture the  frame of each time step
   pcolor(x,z,w(:,:,i)')
   shading interp
   caxis([-3 8]) %for w                      %set the color limit to avoid getting sick or too high by
   %caxis([-1 1]) %for p (mb)                %watching the background color changing each frame
   %caxis([-1 3]) %for th
   colorbar
   hold on                                   %ready to superimpose another plot
   quiver(X,Z,u(:,:,i),w(:,:,i),'b')         %vector plot for velocity
   hold off
   title(['t =  ' num2str(i*dt)])
   pause(0.01)
   drawnow;
   F(i) = getframe(gca);                     %record the image of this loop on a matrix called "F"
   aviobj = addframe(aviobj,F(i));           %and put the frame into the movie file
end

aviobj = close(aviobj);                      %close and save the movie file
```

Appendix C: 3D animation plot (‘pcolor’, ‘quiver3’, ‘getframe’)

```
xmin = 18; xmax = 24;
ymin = 18; ymax = 24;
[X,Y,Z] = meshgrid(x(xmin:xmax),y(ymin:ymax),z);
aviobj = avifile('velocity_3d.avi');

close
for i=1:nstep
    quiver3(X*dx/1000,Y*dy/1000,Z*dz/1000,...
        u(xmin:xmax,ymin:ymax,:,i),v(xmin:xmax,ymin:ymax,:,i),...
        w(xmin:xmax,ymin:ymax,:,i),2.5)
    xlabel('x (km from the origin)')
    ylabel('y (km from the origin)')
    zlabel('z (km from the origin)')
    xlim([7 10])
    ylim([7 10])
    zlim([0 9])
    title(['Velocity field   t =  ' num2str(i*dt) ' (sec)'])
    drawnow
    F(i) = getframe(gcf);
    aviobj = addframe(aviobj,F(i));
end

aviobj = close(aviobj);
```