Schedule

1. **What are models?**
   Exercise: Python Skills

2. **Energy budget of the Earth**
   Exercise: Simple Energy Balance Model

3. **Nonlinearity, Feedback, and Predictability**
   Exercise: Nonlinearity and Feedbacks
   Exercise: Revised Energy Balance Model

4. **Parametrization and Sensitivity**

5. **Radiative budget**
   Exercise: 1-layer greenhouse model
   Exercise: 2-layer greenhouse model

6. **Introduction to fluid dynamics**
   Exercise: Analytical katabatic flow model

`

7. **Finite difference method**
   Exercise: Heat Equation
   Exercise: Advection-Diffusion Equation
   Exercise: Boundary layer Evolution
   Exercise: Numerical katabatic flow model

8. **Implicit finite difference methods**
   Exercise: Boundary layer evolution

9. **Optimization problem**
   Exercise: Surface energy balance
   Exercise: Sublimation

10. **COSIPY snow model**
    Exercise: Simulations with COSIPY

11. **Introduction to PALM**
    Exercise: Simulations with PALM

12. **How to write an article**

Exercise: Advection-diffusion equation

$$\frac{\partial c}{\partial t} = -\left(u\frac{\partial c}{\partial x} + v\frac{\partial c}{\partial y} + w\frac{\partial c}{\partial z}\right) + S_c - \frac{\partial \overline{u'c'}}{\partial x} - \frac{\partial \overline{v'c'}}{\partial y} - \frac{\partial \overline{w'c'}}{\partial z}$$

Accumulation | Advection | Source term | Turbulent mixing

$$\frac{\partial c}{\partial t} = -u\frac{\partial c}{\partial x} - \frac{\partial \overline{u'c'}}{\partial x}$$

Parametrization using eddy-diffusivity closure:

$$\overline{u'c'} = -K\frac{\partial c}{\partial x}$$

$$\frac{\partial \overline{u'c'}}{\partial x} = \frac{\partial}{\partial x}K\left(\frac{\partial c}{\partial x}\right) = K\frac{\partial^2 c}{\partial x^2}$$

$$\boldsymbol{\frac{\partial c}{\partial t} = -u\frac{\partial c}{\partial x} + K\frac{\partial^2 c}{\partial x^2}}$$

Exercise: Advection-diffusion equation

$$\frac{\partial c}{\partial t} = -u\frac{\partial c}{\partial x} + K\frac{\partial^2 c}{\partial x^2}$$

Let's discretize the equation:

$$c_i^{n+1} = c_i^n + \left[-u\frac{c_{i+1}^n - c_{i-1}^n}{2\Delta x} + K\frac{c_{i+1}^n + c_{i-1}^n - 2c_i^n}{(\Delta x)^2}\right]\Delta t,$$

which can be rewritten as

$$c_i^{n+1} = (1-2d)c_i^n + \left(d - \frac{c}{2}\right)c_{i+1}^n + \left(d + \frac{c}{2}\right)c_{i-1}^n,$$

where we introduce the dimensionless parameters

$$d = \frac{K\Delta t}{(\Delta x)^2} \quad \text{with} \quad c = \frac{u\Delta t}{\Delta x}$$

Exercise: Advection-diffusion equation

Task 4: Solve the Advection-Diffusion equation, with the following initial and boundary conditions: at t= 0 , $c_0 = 0$; for all subsequent times, $c = 0$ at x $= 0$, $c = 1$ at $x = L = 1$, $u = 1.0$, K $= 0.1$, and 40 grid points. Integrate over 0.05 s with a $\Delta t = 0.0028$. Plot the results and the dimensionless time scales. Increase gradually Δt and analyse the results. Once you understand what is happening, set again $\Delta t = 0.0028$ and gradually increase the wind speed. Discuss the results.

WHAT HAPPENED?

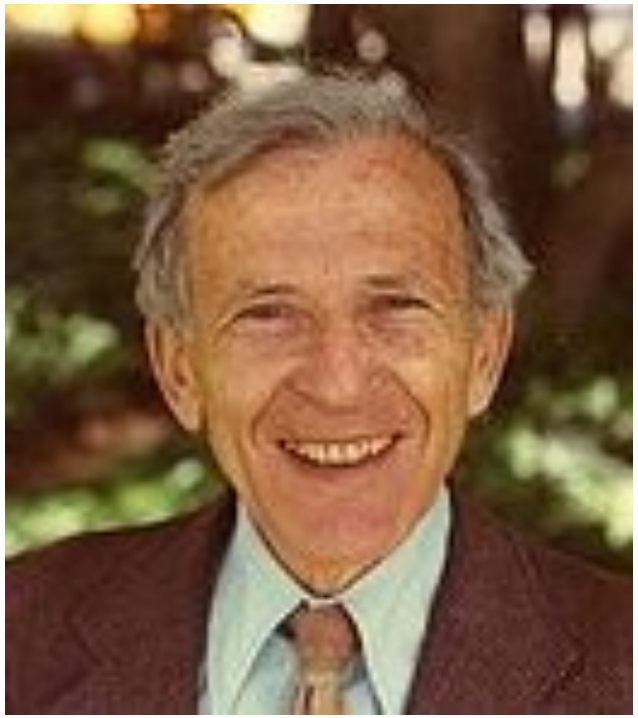
Exercise: Advection-diffusion equation

Characteristic diffusion time: $$d = \frac{K\Delta t}{(\Delta x)^2}$$

Courant number: $$c = \frac{\mathrm{u}\Delta t}{\Delta x}$$

Mr. Courant

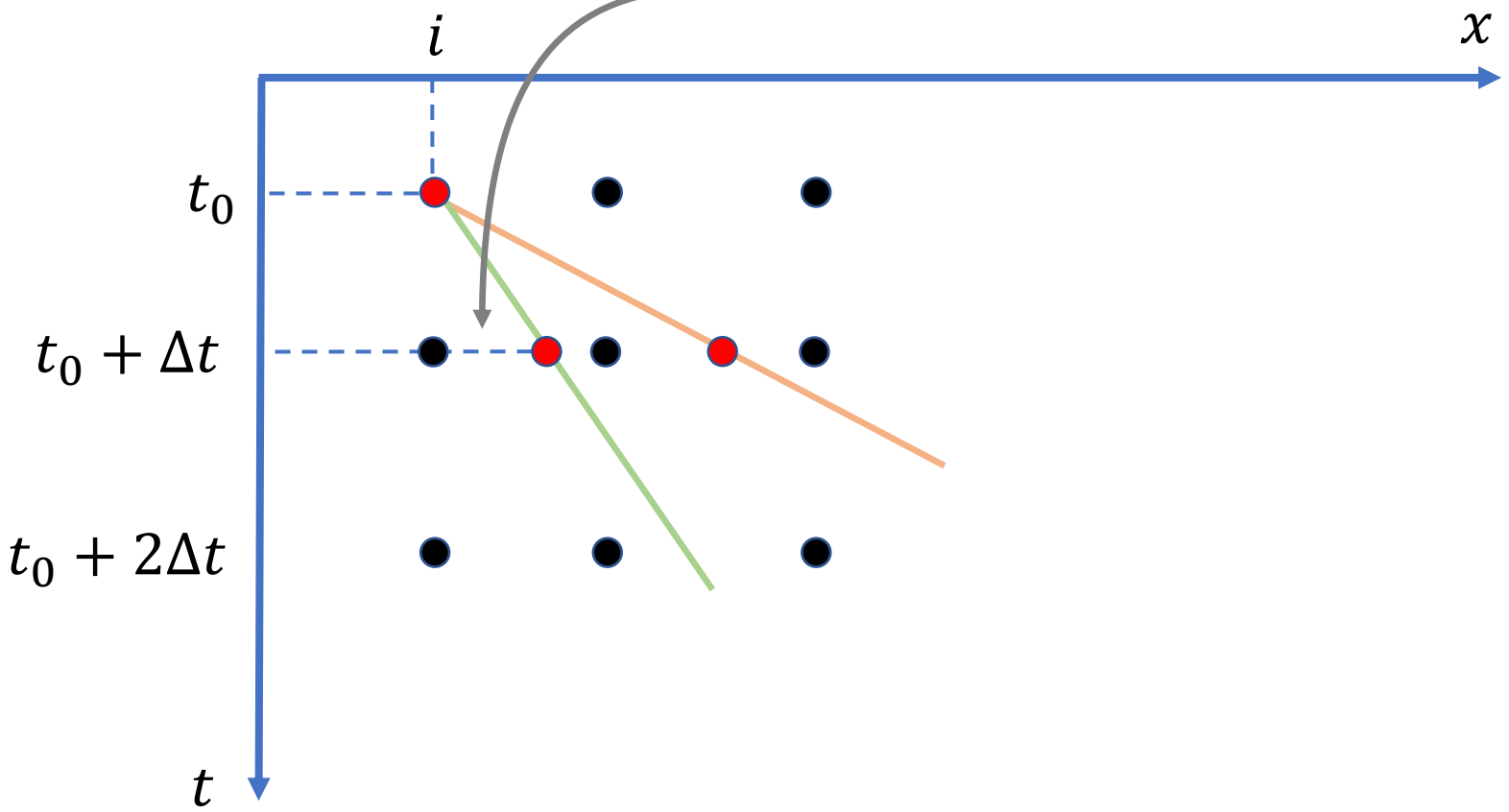
Mr. Friedrichs



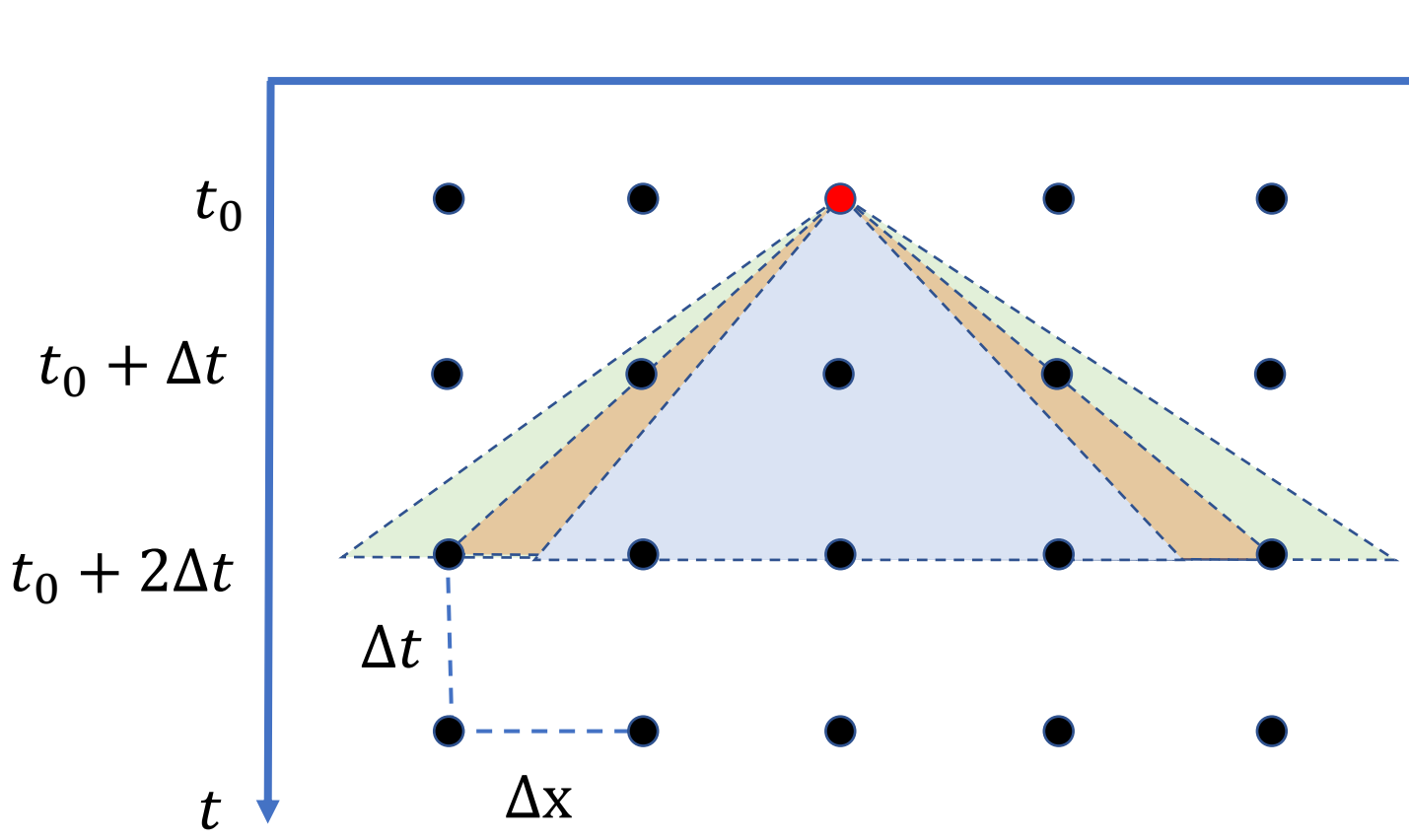
Mr. Lewi

Definition of velocity

$$u = \frac{\Delta x}{\Delta t} \qquad \Delta x = u\Delta t$$



Courant-Friedrichs-Lewy condition (CFL)

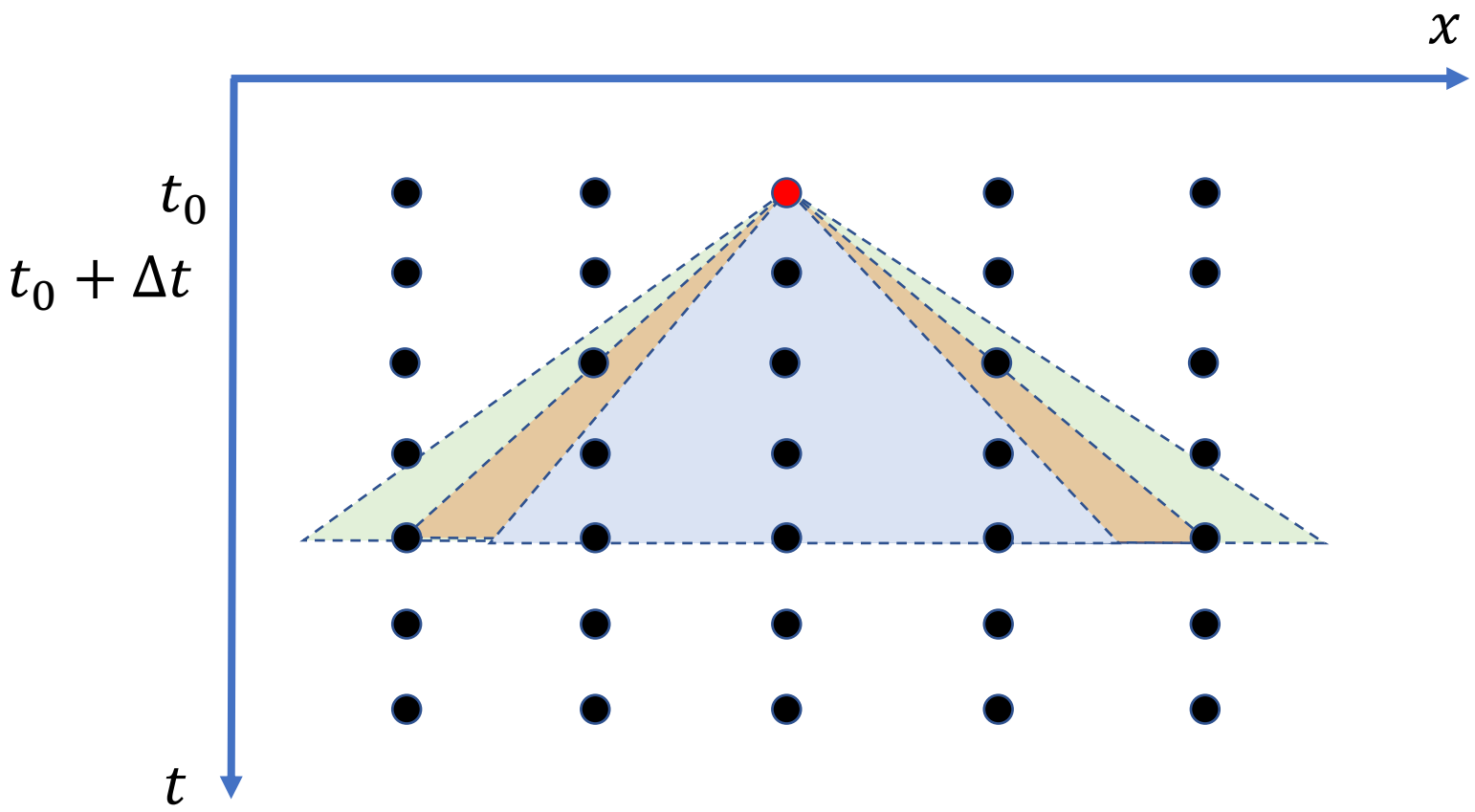


$$\frac{u\Delta t}{\Delta x} = 1$$

$$\frac{u\Delta t}{\Delta x} > 1$$

$$\left.\frac{u\Delta t}{\Delta x} < 1\right\} \quad \frac{u\Delta t}{\Delta x} < C$$

Courant-Friedrichs-Lewy condition (CFL)



$$\frac{u\Delta t}{\Delta x} = 1$$

$$\frac{u\Delta t}{\Delta x} > 1$$

$$\left.\frac{u\Delta t}{\Delta x} < 1\right\} \quad \frac{u\Delta t}{\Delta x} < C$$

$$\Delta t < \frac{C\Delta x}{u}$$

Courant-Friedrichs-Lewy condition (CFL)

For velocity with units of u being $\left[\frac{m}{s}\right]$: $$\Delta t \leq \frac{C\Delta x}{u}$$

For diffusion with units of α being $\left[\frac{m^2}{s}\right]$: $$\Delta t \leq \frac{C\Delta x^2}{\alpha}$$

Exercise: Advection-diffusion equation

Task 5: Solve the Advection-Diffusion equation form x=0 to L, with the following initial and boundary conditions:

$$c(x,0) = e^{\left(\frac{x-10}{2}\right)^2}$$

$$c(0,t) = 0$$

$$c(L,t) = \frac{\partial c}{\partial x} = 0$$

Integrate the equation with K=0.1, u=1.0 over 0.05 s with a Δt=0.0028. Plot the results and the dimensionless time scales. Increase gradually Δt and plot and analyze the results for different integration times.

Exercise: Boundary layer evolution

$$\frac{\partial \theta}{\partial t} = -\left(u\frac{\partial \theta}{\partial x} + v\frac{\partial \theta}{\partial y} + w\frac{\partial \theta}{\partial z}\right) - \frac{1}{\rho c_p}\left(\frac{\partial Q_x^*}{\partial x} + \frac{\partial Q_y^*}{\partial y} + \frac{\partial Q_z^*}{\partial z}\right) - \frac{L_v E}{\rho c_p} - \frac{\partial \overline{u'\theta'}}{\partial x} - \frac{\partial \overline{v'\theta'}}{\partial y} - \frac{\partial \overline{w'\theta'}}{\partial z}$$

Task 6: Starting with the heat equation above simplify the equation to model the temperature evolution in the boundary layer from the surface up to H=2 km height. Assume a fair-weather condition with a subsidence of -0.001 m s^{-1}. Also assume horizontal homogeneity. Parameterize the heat flux using the eddy-diffusivity closure with K=0.25 m s^{-2}. Solve the simplified equation using the following initial and boundary conditions:

$$\theta(z,0) = 290\ K \qquad \theta(0,t) = 290 + 10 \cdot \sin\left(\frac{2\pi \cdot t}{86400}\right) K$$

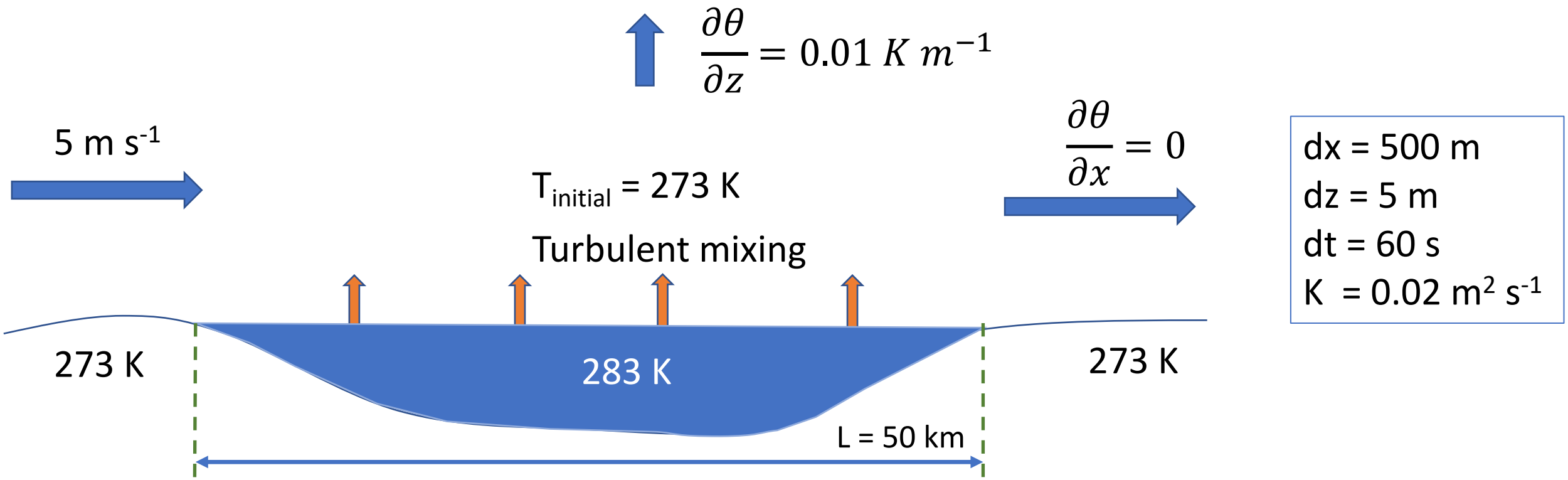
$$\overline{w'\theta'}\,(z,0) = 0\ W\ m^{-2} \qquad \theta(H,t) = \frac{\partial \theta}{\partial z} = 0.01\ K\ m^{-1}$$

- What happens when you increase the subsidence to -0.01 m s^{-1}?
- Plot the kinematic heat flux.
- What is the maximum heat flux in W m^{-2}? Is this a realistic value for a fair-weather condition?
- Calculate the heating rate in K per hour.

Lake effect of Lake Erie

Exercise: Lake-effect

Task 7: Intense boundary layer convection may develop when cold air masses are advected over relatively warm surfaces. Develop a simple model for this by assuming that the time evolution of the boundary layer is determined by the vertical turbulent heat transport and the horizontal heat advection. Make the following assumptions: [Hint: use the eddy-diffusivity closure and the upwind scheme for the advection flux]



Exercise: Lake-effect

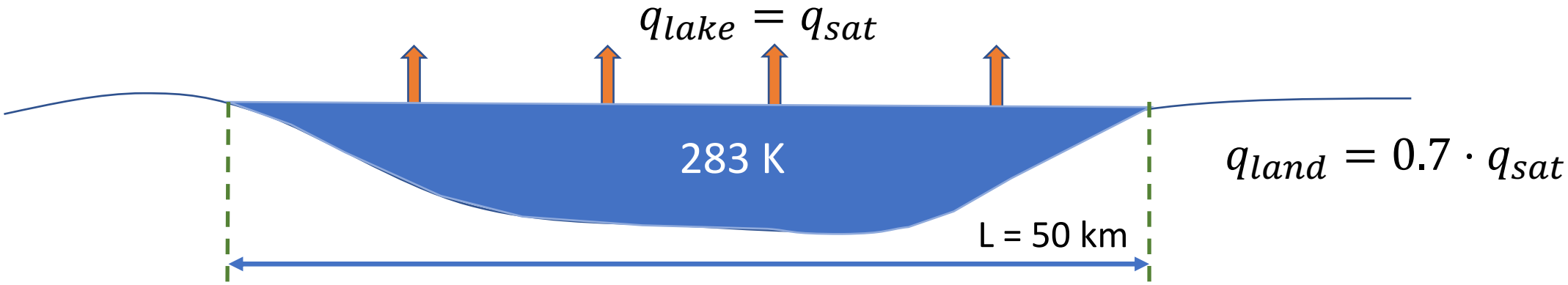
Task 8: Extend the Lake-effect model by adding the moisture transport equation. Assume that the top millimetres above the water surface are saturated. Assume that the relative humidity in the atmosphere decreases linearly from 70 to 20 %. Calculate the relative humidity at each grid cell to identify where condensation takes place [Note: Convert the potential temperature to normal temperature (by adding a lapse rate) and calculate the mixing ratio at each cell. Then calculate the relative humidity.]

dx = 500 m
dz = 5 m
dt = 60 s
K = 0.1 m^2 s^{-1}

$$\frac{\partial q}{\partial z} = 0.0\ g\ kg^{-1}\ m^{-1}$$

$$\frac{\partial q}{\partial x} = 0.0\ g\ kg^{-1}\ m^{-1}$$

Relative humidity from 70% at surface to 20%



Exercise: Lake-effect

Step 1: Write three auxiliary functions to calculate the saturation water vapor, hypsometric equation, and the mixing ratio

Saturation water vapor [hPa]

$$E = 6.122 \cdot \exp^{\left(\frac{17.67 \cdot (T - 273.16)}{T - 29.66}\right)}$$

Hypsometric equation [hPa]

$$p = \left(\frac{p_0}{\exp^{\frac{g \cdot z}{R \cdot Tv}}}\right)$$

Mixing ratio [g kg^{-1}]

$$q = 0.622 \cdot \frac{E}{(p - E)}$$

Exercise: Lake-effect

Step 1: Write three auxiliary functions to calculate the saturation water vapor, hypsometric equation, and the mixing ratio

```
4  # ------------------------
5  # Auxiliary functions
6  # ------------------------
7  def saturation_water_vapor(T):
8      """ Calculates the saturation water vapor pressure [Pa]"""
9      return ( 6.122*np.exp( (17.67*(T-273.16))/(T-29.66) ) )
10
11 def hypsometric_eqn(p0, Tv, z):
12     """Hypsometric equation to calculate the pressure at a certain height
13        when the surface pressure is given
14        p0 :: surface pressure [hPa]
15        Tv :: mean virtual temperature of atmosphere [K]
16        z  :: height above ground [m]
17     """
18     return(p0/(np.exp((9.81*z)/(287.4*Tv) )))
19
20 def mixing_ratio(theta, p0, Tv, z):
21     """ Calculates the mixing ratio from
22         theta :: temperature [K]
23         p0    :: surface pressure [hPa]
24         Tv    :: mean virtual temperature of atmosphere [K]
25         z     :: height [m]
26     """
27     return(622.97 * (saturation_water_vapor(theta)/(hypsometric_eqn(p0,Tv,z)-saturation_water_vapor(theta))))
```

Exercise: Lake-effect

Step 2: Initialize temperature and add the lapse rate to the difference equation approximation.

```
# -------------------------
# Initial temperature field
# -------------------------
# Neutral stratification with lapse rate of 0.01 K/m
# Create a 1D-array with the vertical temperature distribution
# Surface = 268 K, decreasing according to the dry-adiabative lapse rate 0.01 K/m
lapse_rate = -0.01
theta_vec = np.array([268 + lapse_rate * (dz * z) for z in range(Nz)])
theta = np.array([theta_vec,] * Nx).transpose()
```

```
for x in range(1,Nx-1):
    # Update temperature NEW::lapse rate added
    theta[m,x] = theta[m,x] + ((K*dt)/(dz**2))*(old[mu,x]+old[md,x]-2*old[m,x]) + lapse_rate
```

Exercise: Lake-effect

Step 3: Initialize the moisture field.

```
61      # Make height grid
62      height = np.array([np.arange(0,Nz*dz,dz),] * Nx).transpose()
```

```
83      # ------------------------
84      # Initialize moisture fields
85      # ------------------------
86      # Init saturation mixing ratio array
87      qsat = mixing_ratio(theta, 1013, 270, height)
88
89      # Use qsat to derive mixing ratio at each grid cell assuming a
90      # decreasing relative humidity from 70% at the surfacee to 20% at the top
91      q = (qsat.T * np.linspace(0.7, 0.2, Nz)).T
92
93      # The lower moisture boundary needs to be updated where there is the lake
94      # Here, were set the moisture at the lower boundary from the grid cell 50
95      # to 150 to a mixing ratio of 0.9 times the saturation mixing ratio
96      q[0, lake_from:lake_to] = 0.9 * qsat[0, lake_from:lake_to]
```

Exercise: Lake-effect

Step 3: Add transport equation for moisture.

```
138         for x in range(1,Nx-1):
139             # Update temperature NEW::lapse rate added
140             theta[m,x] = theta[m,x] + ((K*dt)/(dz**2))*(old[mu,x]+old[md,x]-2*old[m,x]) + lapse_rate
141             # Moisture transport
142             q[m,x] = q[m,x] + ((K*dt)/(dz**2))*(old_q[mu,x]+old_q[md,x]-2*old_q[m,x])
```

```
149         for z in range(1,Nz-1):
150             # Update temperature
151             theta[z,k] = theta[z,k] - ((u*dt)/(dx))*(old[z,k]-old[z,kl])
152             # Update moisture
153             q[z,k] = q[z,k] - ((u*dt)/(dx))*(old_q[z,k]-old_q[z,kl])
```

Exercise: Lake-effect

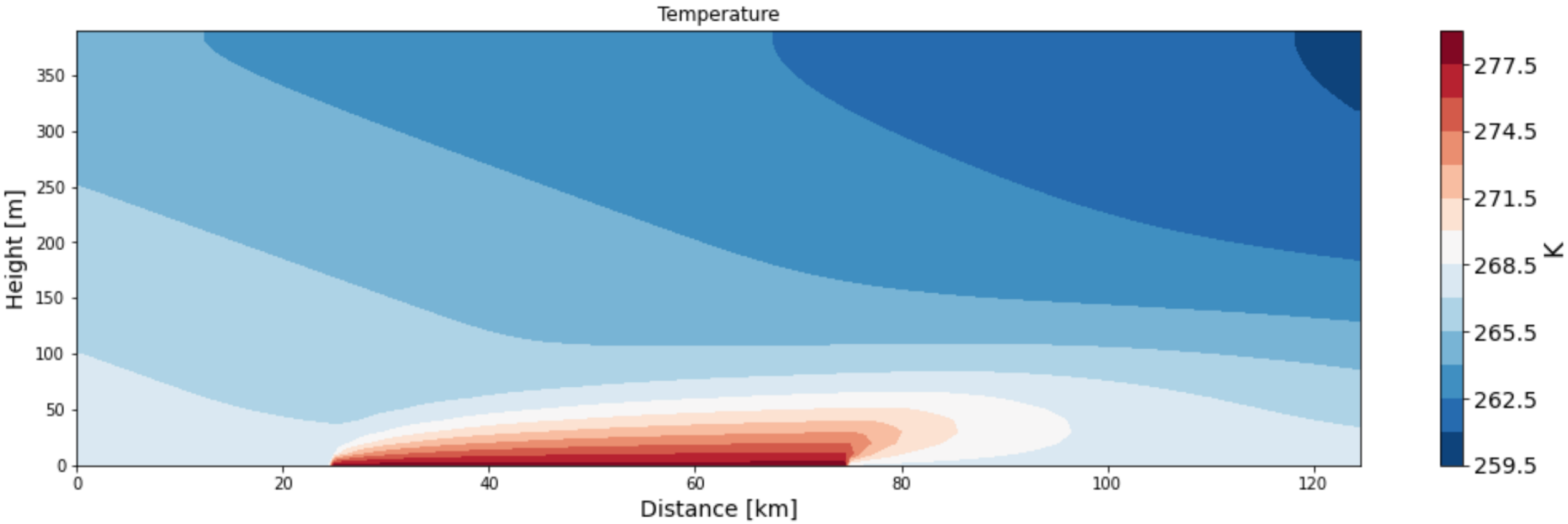
Step 4: At the end of the simulation caculate the new saturation mixing ratio and derive the relative humidity.

```
158         # Calculate new saturation mixing ratio
159         qsat = mixing_ratio(theta, 1013, 270, height)
160
161         # Then the relative humidity using qsat
162         # Limit the relative humidity to 100 %
163         rH = np.minimum(q/qsat, 1)
```

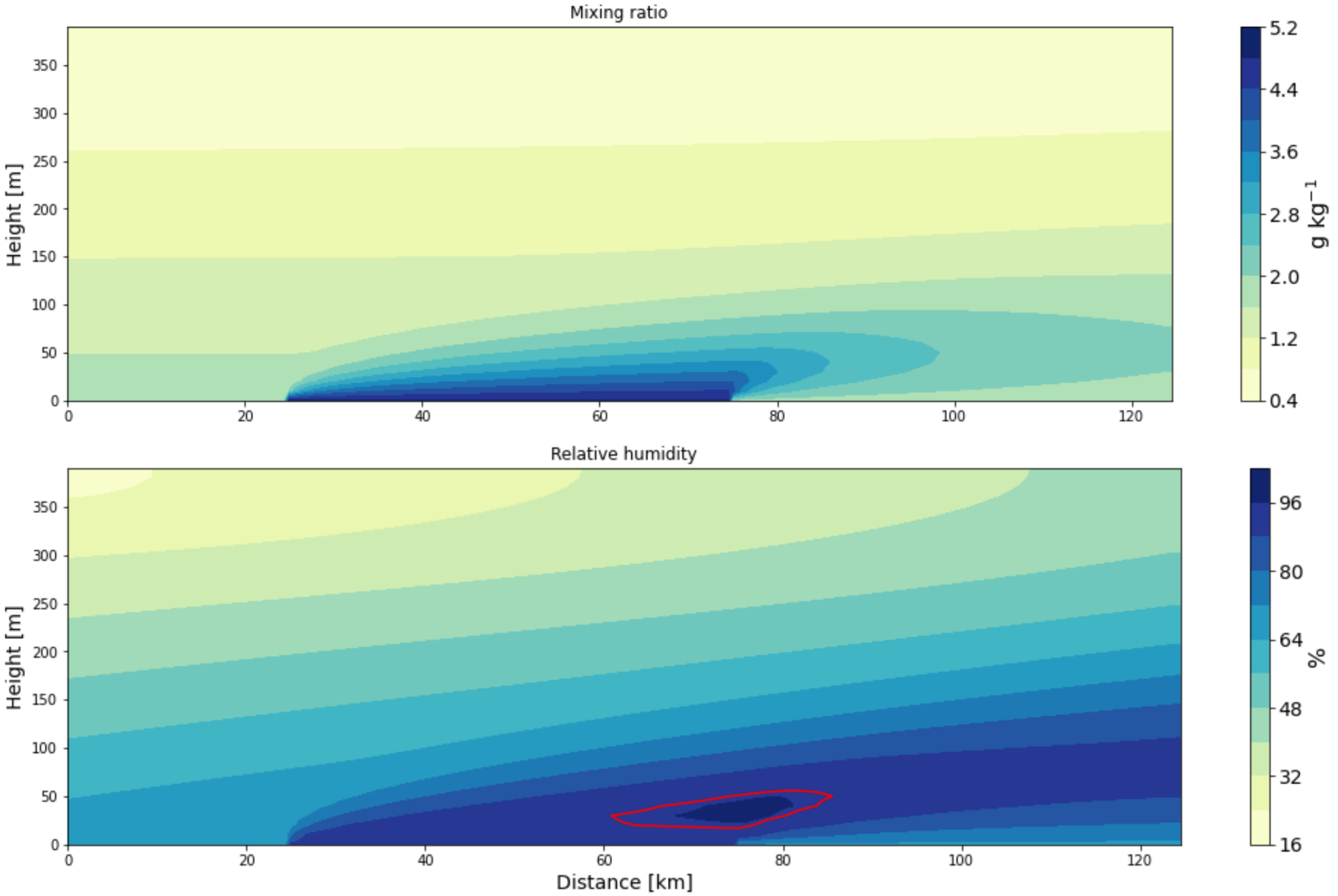
Exercise: Lake-effect

Step 5: Run the model.

```
# Run the model
theta, q, qsat, rH, cov, adv, c, d, x, z = boundary_layer_evolution_moisture(u=5, K=0.2, dx=500, dz=10,
                                                                             Nx=250, Nz=40, hours=24, dt=60)
```



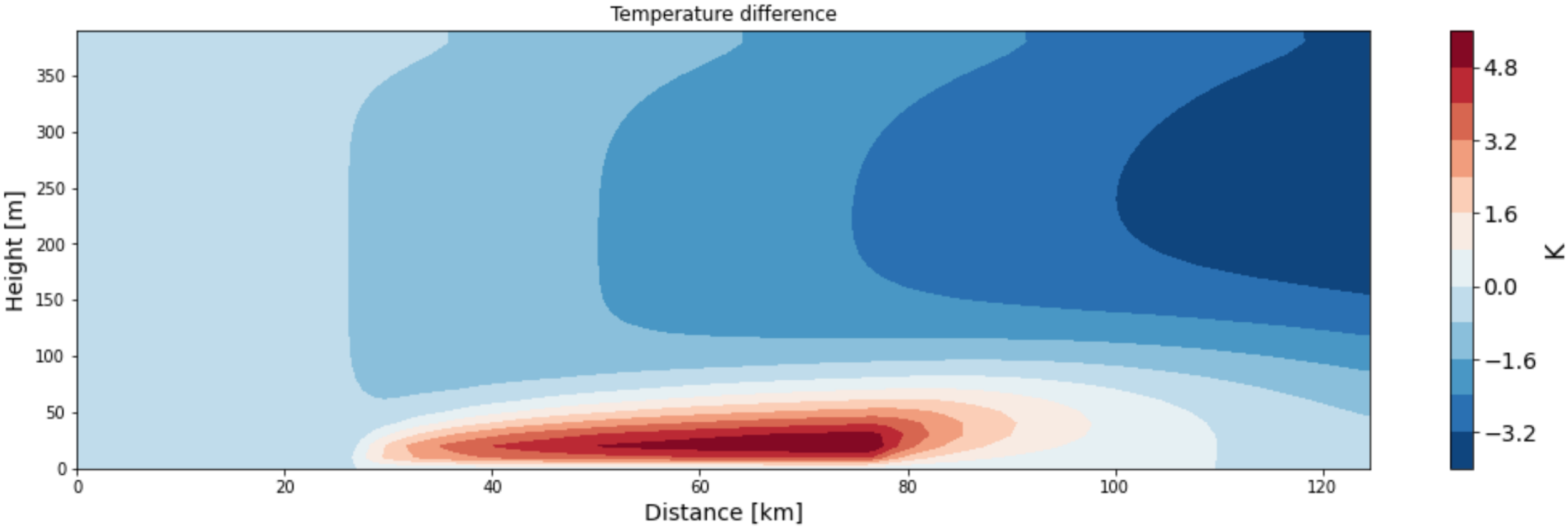
Mixing ratio and relative humidity



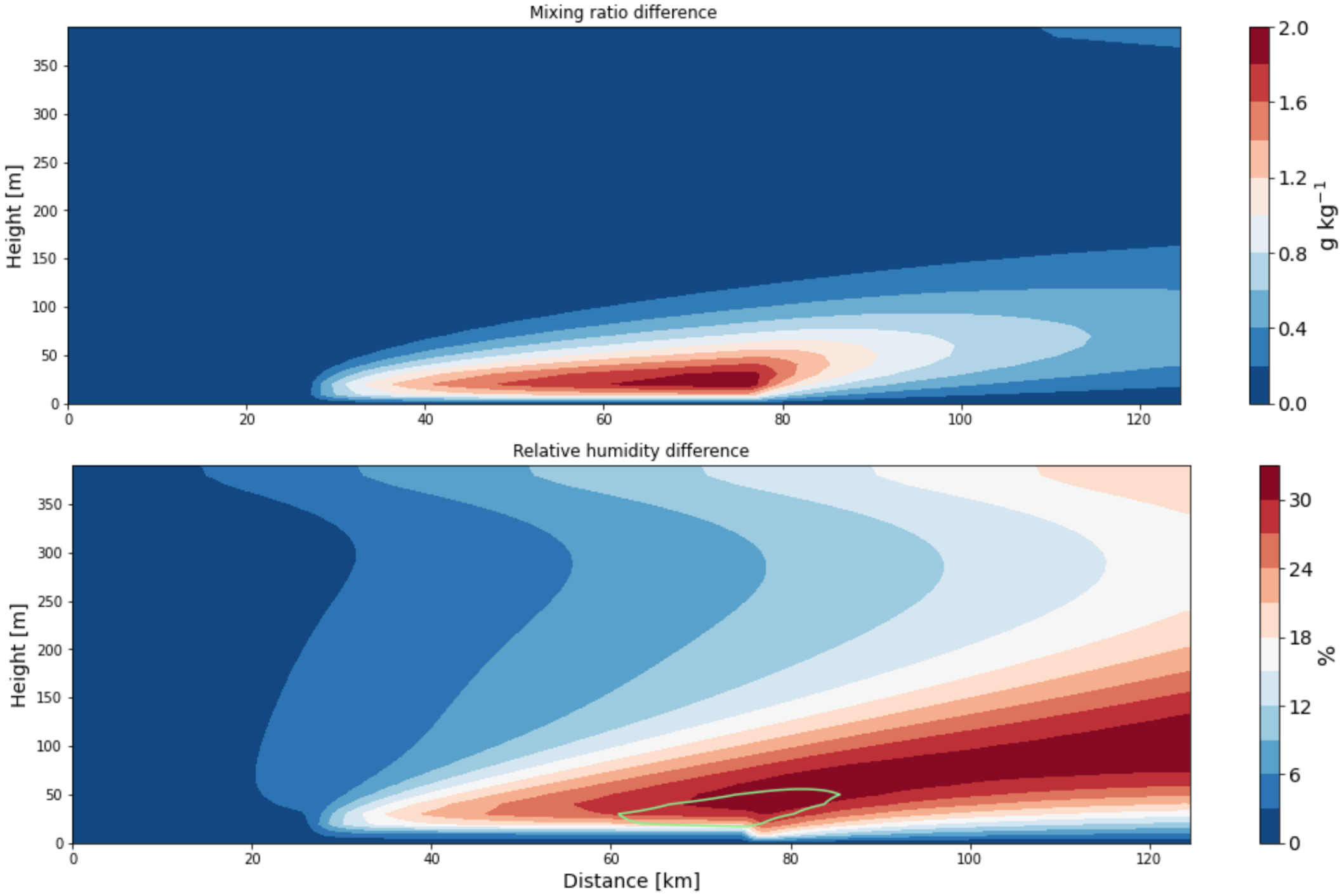
Exercise: Lake-effect

Step 6: Run the model again with a very short time step to obtain a reference run and plot the differences between the two runs.

```
# Base run
theta0, q0, qsat0, rH0, cov0, adv0, c0, d0, x0, z0 = boundary_layer_evolution_moisture(u=5, K=0.2, dx=500, dz=10,
                                                                                       Nx=250, Nz=40, hours=0.1, dt=60)
```



Mixing ratio and relative humidity



Exercise: Lake-effect

Step 7: Modify the code to account for different lapse rates, e.g. when saturated the lapse rate decreases to a moist adiabatic lapse rate of 0.006 K/m.

```
64      # Make lapse rate array
65      Gamma = -0.01 * np.ones((Nz, Nx))
```

```
142             # Update temperature NEW::lapse rate added
143             theta[m,x] = theta[m,x] + ((K*dt)/(dz**2))*(old[mu,x]+old[md,x]-2*old[m,x]) + Gamma[m,x]
```

```
167         # Correct lapse rates where rH==100% (moist adiabatic lapse rate)
168         Gamma[rH==1] = -0.006
```