We will be following a set of notes developed by Dr. Rob Fovell of UCLA. Our first four tasks will involve setting up the basic model state and implementing our four governing equations. After that, we will depart from Dr. Fovell’s notes and add some other things to the model on our own.

1. Model task: read Fovell’s chapter 9, complete model task 1, and turn in a printout similar to Fovell’s p. 105–106. You should set up the skeleton for your code, declare all of the variables mentioned in chapter 9, and define their vertical profiles as instructed.

We will deviate from Fovell’s task as follows: we will use a form of the initial sounding (which is based on the famous Weisman and Klemp 1982 convection simulation sounding) that is more general than what Fovell gives in Chapter 9 (Fovell simply hardwires the default values). WK82 actually had a form that allowed the user to tune the temperature and humidity profiles by setting a few key parameters. This will enable you more easily to do some experimenting later on. *Note: because we are making some small modifications to the mid-level humidity (to avoid a subtle problem later on in the semester), the values in your table will not be identical to Fovell’s page 105-106 (but they should be close).*

Here are some excerpts from my code that show you how to modify Fovell’s examples:

```fortran
! parameters to describe the sounding
real tsurf ! potential temperature at the surface (K)
parameter(tsurf=300.0) ! default is 300.0 K
real qsurf ! mixing ratio at the surface (kg/kg)
parameter(qsurf=0.0161) ! default is 0.0161
real q4km ! mixing ratio at 4 km AGL (kg/kg)
parameter(q4km=0.001) ! default is 0.001
real ztr ! height of the tropopause (m)
parameter(ztr=12000.0) ! default is 12000.0 m
real temptr ! temperature at the tropopause (K)
parameter(temptr=213.0) ! default is 213.0 K
real ttr ! pot. temperature at the tropopause (K)
parameter(ttr=343.0) ! default is 343.0 K

! assign mixing ratios
if(zu(k).le.4000.0) then
  qb(k) = qsurf - (qsurf-q4km)*zu(k)/4000.0
elseif(zu(k).le.8000.0) then
  qb(k) = q4km - q4km*(zu(k)-4000.0)/4000.0
else
  qb(k) = 0.0
endif
```

(Continues on next page)
! assign potential temperatures
    if(zu(k).le.ztr) then
      tb(k) = tsurf + (ttr-tsurf)*(zu(k)/ztr)**1.25
    else
      tb(k) = ttr*exp(grav*(zu(k)-ztr)/(cp*temptr))
    endif

2. Review the first two pages of the supplemental handout and briefly report (in your own words) on where Fovell’s equation for \( q_{vis} \) on the bottom of p. 103 comes from.

3. Model task: read Fovell’s chapter 10, complete model task 2, and turn in a printout similar to Fovell’s p. 112–113. Be sure to diagnose the LCL, LFC, EL, and sum up the CAPE. Start your parcel at 350 m AGL (the first u/scalar point).

4. Review the third page of the supplemental handout and briefly report (in your own words) on the origins of the first two equations on page 110 of Fovell’s notes (the equations for \( C \) and \( \phi \)). What is \( \phi \) anyhow?
Chapter 9

Model Task #1: Setting up the base state

9.1 Vertical grid arrangement

Our model will have five prognostic variables: horizontal velocity $u$, vertical velocity $w$, potential temperature $\theta$, water vapor mixing ratio $q_v$, and nondimensional pressure $\pi$. First, we will set up the initial environmental values (“base state”) for each of these variables, plus air density $\rho$. The base state will vary only in the vertical direction and will be assumed to be in hydrostatic balance. MKS values will be used at all times.

In finite differencing, we subdivide the model domain into a set of grid volumes (or grid areas in 2D). Once accomplished, it remains to specify values of the variables for each grid volume. Where in space should the variables be defined, or thought of as existing? The simplest procedure might be to declare all variables to represent points residing at the center of each grid box. In this arrangement, each variable value is thought of as representing a grid volume average, and the most logical place to think of this average existing is at the center.

Instead, our model will employ a staggered grid arrangement – specifically, Arakawa’s “C” grid – which is a natural for mesoscale models. The scalar variables ($\theta$, $\pi$, $q_v$) are still placed at the grid center but the velocity components are arrayed along the volume or area edges (see Fig. 1). In this arrangement, the velocities represent flows across the boundaries.
The grid box width and depth will be called $\Delta x$ and $\Delta z$, respectively. Note that $u$ and the scalar variables are at the same physical height levels, while $w$ is displaced $0.5\Delta z$ above and below. These two height levels will be called the $u$ and $w$ heights, respectively. In this arrangement, we only really need boundary conditions in the vertical direction on $w$, and we will assume the boundaries are rigid, flat plates such that $w = 0$ there.

The only density our model will need is base state density — at least explicitly. Density, a scalar variable, is most naturally defined at the grid box center with the other scalars (the $u$ height level). However, since density is so important, we will find it useful to also define density at the $w$ height levels. The base state densities at the $u$ and $w$ height levels will be called RHOU and RHOW, respectively, for coding purposes. The base states for the other variables will be called UB (for $\bar{u}$), TB (for $\bar{\theta}$), QB (for $\bar{q}_v$), and PIB (for $\bar{\pi}$). Mean $w$ is zero.

The vertical grid index will be termed $k$, and all base state arrays will be vectors with dimension $NZ$, as there are a total of $NZ$ grid points (see Figs. 1 and 2). Since the grid is staggered, note that a given $k$-index value, say “$k = 5$”, refers to different physical heights for the $w$ and $u$/scalar variables! To facilitate coding, we will include one fictitious grid volume at the top and bottom of each model column. It will be seen that this permits simpler and more efficient coding of our equations at a small cost of increased storage. (Fictitious grids will be added along the horizontal boundaries as well, for the same reason.)

In Fortran (Fig. 1), the index for an array dimensioned $NZ$ begins at index 1 and runs to $NZ$, inclusive. Thus, the real boundaries for $w$ will be located at $k = 2$ and $NZ$, and $w$ will be zero at both. Therefore, the lowest and highest physical $u$ and scalar points are located at $k = 2$ and $NZ-1$, respectively. The scalar/$u$ grid points at $k = 1$ and $NZ$ are fictitious, as is the $w$ point for $k=1$.

In C++ (Fig. 2), array indices start at 0 and thus the maximum index for an array dimensioned $NZ$ is actually $NZ-1$. So, we will enforce $w$ to be zero at $k = 1$ and $NZ-1$ and the lowest and highest $u$ and scalar points are located at $k = 1$ and $NZ-2$. The scalar/$u$ grid points at $k = 0$ and $NZ-1$ are fictitious, as is $w$ at $k=0$. *Note that in an array dimensioned $NZ$ the point $k=NZ$ does not exist — at least, not within the range of the array under examination. The attempt to access array
location \( NZ \) in an array dimensioned \( NZ \) is the single most common coding error I’ve seen.

### 9.2 Base state temperature and moisture

The base state environment described similar to that employed by Weisman and Klemp (1982, MWR, p. 504) and represents conditions common to the Midwestern United States during the spring season. Let \( z_{TR} \) and \( T_{TR} \) be the height level (12000 m) and temperature (213 K) of the tropopause, with \( \theta_{TR} \) being the potential temperature there (343 K). Let \( z_T \) be the distance of a given scalar location above the model surface. In **Fortran**, the model surface is the \( k = 2 \) \( w \) point in Fortran and so the physical height of the first scalar point is \( (k - 1.5)\Delta z \). In **C++**, the model surface is the \( k = 1 \) \( w \) point and so the physical height of the first scalar point is \( (k - 0.5)\Delta z \). The Weisman and Klemp vertical mean potential temperature (\( \bar{\theta} \)) is:

\[
\bar{\theta} = \begin{cases} 
300 + 43 \left[ \frac{z_T}{z_{TR}} \right]^{1.25} & z_T \leq z_{TR}; \\
\theta_{TR} \exp \left[ \frac{g(z_T - z_{TR})}{c_p T_{TR}} \right] & z_T > z_{TR}.
\end{cases}
\]

where \( g = 9.8 \) m s\(^{-1} \), and \( c_p = 1004 \) J kg\(^{-1} \) K\(^{-1} \). Let the base state mean vapor mixing ratio, \( \bar{q}_v \) (kg kg\(^{-1} \)), be given by:

\[
\bar{q}_v = \begin{cases} 
0.0161 - 0.000003375z_T & z_T \leq 4000 \text{ m}; \\
0.0026 - 0.00000065(z_T - 4000) & 4000 < z_T \leq 8000 \text{ m}; \\
0 & z_T > 8000 \text{ m}.
\end{cases}
\]

Take \( NZ=40 \) and \( \Delta z = 700 \) m and compute both. Then use these data to create virtual potential temperatures, which should also be contained in an array of dimension \( NZ \) and positioned at the scalar heights, using \( \bar{\theta}_v = \bar{\theta} [1. + 0.61\bar{q}_v] \). I’ll call this array TBV.
Figure 9.1: Vertical grid arrangement for Fortran code. The first and last real $w$ points are at $k = 2$ and $NZ$, respectively. The first and last real scalar/$u$ points are at $k = 2$ and $NZ-1$. Note presence of two fictitious grid boxes, located just above and below the model’s vertical boundaries.
Figure 9.2: Vertical grid arrangement for C++ code. The first and last real \( w \) points are at \( k = 1 \) and \( NZ-1 \), respectively. The first and last real scalar/\( u \) points are at \( k = 1 \) and \( NZ-2 \). Note presence of two fictitious grid boxes, located just above and below the model’s vertical boundaries.
9.3 Derived quantities

Next, the base state nondimensional pressure ($\bar{\pi}$) has to be computed\(^1\). Pressure is nondimensionalized as:

$$\pi = \left[ \frac{p}{p_0} \right]^{\frac{R_d}{\gamma}} \, ,$$

where $R_d = 287 \text{ J kg}^{-1}$ and $p_0 = 100000 \text{ N m}^{-2}$ (1000 mb). The hydrostatic equation, written in terms of $\pi$ is:

$$\frac{d\bar{\pi}}{dz} = -\frac{g}{c_p\theta_v} \, .$$

We will obtain $\bar{\pi}$ as a function of height by integrating this equation upwards from the surface, starting with a supplied surface pressure which will be taken to be 96500 N m$^{-2}$ (965 mb). This surface pressure (PSURF) is recorded at the lowest real $w$ point ($k = 2$ in Fortran, $k = 1$ in C++). The first task is then to compute $\bar{\pi}$ at the first real scalar point, located 0.5$\Delta z$ above the surface. The mean virtual potential temperature, $\bar{\theta}_v$, will be taken to be its value at the first real scalar point and presumed constant below that point. This can be coded in Fortran as:

```fortran
xk = rd/cp
pisfc = (psurf/p0)**xk
pib(2) = pisfc - grav*0.5*dz/(cp*tbv(2))
```

In C++, the code might look like this:

```cpp
xk = rd/cp;
pisfc = pow((psurf/p0),xk);
pib[1] = pisfc - grav*0.5*dz/(cp*tbv[1]);
```

Now we need to integrate the hydrostatic equation up through the rest of the column. Each step will be of height $dz$ as we traverse from one scalar point to the next. From the hydrostatic equation, we know that the pressure at the top of a designated layer depends on two things: the pressure at the layer bottom, and the mean virtual temperature of the layer. To get the mean $\bar{\theta}_v$ for a layer between two scalar heights, we need to average the TBV values at the layer top and bottom.

\(^1\)The advantage of avoiding dimensional pressure will be seen later.
In Fortran, we’re integrating from \( k = 3 \) to \( NZ-1 \) because \( NZ-1 \) represents the uppermost real scalar point (one-half \( \Delta z \) below the model top at \( w \) position \( k = NZ \)). (If the desired DO loop increment is unity, it need not be explicitly coded.) In C++, we traverse from \( k = 2 \) to \( NZ-2 \), inclusive.

```fortran
  do k = 3, nz-1
    tbvavg = 0.5*(tbv(k)+tbv(k-1))
    pib(k) = pib(k-1) - grav*dz/(cp*tbvavg)
  enddo
```

```c++
  for(k = 2; k <= nz-2; k++){
    tbvavg = 0.5*(tbv[k]+tbv[k-1]);
    pib[k] = pib[k-1] - grav*dz/(cp*tbvavg);
  }
```

Once \( \bar{\pi} \) is obtained, mean state density \( \bar{\rho} \) has to be computed from a form of the ideal gas law (with pressure nondimensionalized):

\[
\bar{\rho} = \frac{p_0 \pi \bar{\rho}_d}{R_d \bar{\theta}_v}
\]

where \( c_v = c_p - R_d \). Density at the scalar height levels (RHOU) is easily obtained. You get the surface value of RHOW (rhow(2) or rhow[1]) by knowing the surface pressure, and RHOWs farther aloft by averaging pairs of RHOU values to the \( w \) height level. From Fig. 1, it is seen this is written as:

\[
rhow(k) = 0.5*(rhou(k) + rhou(k-1))
\]

Finally, compute the relative humidity at each scalar height level, using a form of Teten’s equation to get the saturation mixing ratio for the base state, \( \bar{q}_{vs} \):

\[
\bar{q}_{vs} = \frac{380}{\bar{\rho}} \exp \left[ \frac{17.27(\bar{T} - 273.1)}{(\bar{T} - 36.1)} \right]
\]

where \( \bar{\rho} \) and \( \bar{T} \) are base state pressure (N m\(^{-2}\)) and temperature (K). Since you are carrying nondimensional pressure and potential temperature instead, these variables need to be converted. Using the definition of \( \pi \):
\[ p = p_0 \pi^{\frac{\theta}{\pi_d}} \]

and the relationship between \( T \) and \( \theta \) is:

\[ T = \theta \pi. \]

Compute \( \bar{q}_{es} \) for each scalar grid point and use it to calculate the relative humidity.
### 9.4 Results for some fields

<table>
<thead>
<tr>
<th>z (km)</th>
<th>tb (K)</th>
<th>qb (g/kg)</th>
<th>rhou (kg/m^3)</th>
<th>rel. hum (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.35</td>
<td>300.52</td>
<td>14.92</td>
<td>0.108854E+01</td>
<td>88.77</td>
</tr>
<tr>
<td>1.05</td>
<td>302.05</td>
<td>12.56</td>
<td>0.102338E+01</td>
<td>96.03</td>
</tr>
<tr>
<td>1.75</td>
<td>303.88</td>
<td>10.19</td>
<td>0.960102E+00</td>
<td>99.87</td>
</tr>
<tr>
<td>2.45</td>
<td>305.90</td>
<td>7.83</td>
<td>0.899168E+00</td>
<td>98.58</td>
</tr>
<tr>
<td>3.15</td>
<td>308.08</td>
<td>5.47</td>
<td>0.840753E+00</td>
<td>89.01</td>
</tr>
<tr>
<td>3.85</td>
<td>310.38</td>
<td>3.11</td>
<td>0.784929E+00</td>
<td>65.95</td>
</tr>
<tr>
<td>4.55</td>
<td>312.79</td>
<td>2.24</td>
<td>0.731065E+00</td>
<td>62.75</td>
</tr>
<tr>
<td>5.25</td>
<td>315.30</td>
<td>1.79</td>
<td>0.679760E+00</td>
<td>66.72</td>
</tr>
<tr>
<td>5.95</td>
<td>317.89</td>
<td>1.33</td>
<td>0.631160E+00</td>
<td>67.25</td>
</tr>
<tr>
<td>6.65</td>
<td>320.56</td>
<td>0.88</td>
<td>0.585195E+00</td>
<td>60.80</td>
</tr>
<tr>
<td>7.35</td>
<td>323.30</td>
<td>0.42</td>
<td>0.541789E+00</td>
<td>40.88</td>
</tr>
<tr>
<td>8.05</td>
<td>326.11</td>
<td>0.00</td>
<td>0.500848E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>8.75</td>
<td>328.97</td>
<td>0.00</td>
<td>0.462178E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>9.45</td>
<td>331.90</td>
<td>0.00</td>
<td>0.425832E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>10.15</td>
<td>334.88</td>
<td>0.00</td>
<td>0.391712E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>10.85</td>
<td>337.91</td>
<td>0.00</td>
<td>0.359725E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>11.55</td>
<td>340.99</td>
<td>0.00</td>
<td>0.329775E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>12.25</td>
<td>346.95</td>
<td>0.00</td>
<td>0.299408E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>12.95</td>
<td>358.26</td>
<td>0.00</td>
<td>0.267705E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>13.65</td>
<td>369.94</td>
<td>0.00</td>
<td>0.239361E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>14.35</td>
<td>382.00</td>
<td>0.00</td>
<td>0.214020E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>15.05</td>
<td>394.45</td>
<td>0.00</td>
<td>0.191364E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>15.75</td>
<td>407.31</td>
<td>0.00</td>
<td>0.171109E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>16.45</td>
<td>420.59</td>
<td>0.00</td>
<td>0.152998E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>17.15</td>
<td>434.30</td>
<td>0.00</td>
<td>0.136807E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>17.85</td>
<td>448.46</td>
<td>0.00</td>
<td>0.122330E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>18.55</td>
<td>463.08</td>
<td>0.00</td>
<td>0.109386E+00</td>
<td>0.00</td>
</tr>
<tr>
<td>19.25</td>
<td>478.17</td>
<td>0.00</td>
<td>0.978136E-01</td>
<td>0.00</td>
</tr>
<tr>
<td>19.95</td>
<td>493.76</td>
<td>0.00</td>
<td>0.874663E-01</td>
<td>0.00</td>
</tr>
<tr>
<td>20.65</td>
<td>509.85</td>
<td>0.00</td>
<td>0.782146E-01</td>
<td>0.00</td>
</tr>
<tr>
<td>Temperature</td>
<td>Value</td>
<td>Error</td>
<td>Probability</td>
<td>Confidence</td>
</tr>
<tr>
<td>-------------</td>
<td>-------</td>
<td>-------</td>
<td>-------------</td>
<td>------------</td>
</tr>
<tr>
<td>21.35</td>
<td>526.48</td>
<td>0.00</td>
<td>0.699425E-01</td>
<td>0.00</td>
</tr>
<tr>
<td>22.05</td>
<td>543.64</td>
<td>0.00</td>
<td>0.625461E-01</td>
<td>0.00</td>
</tr>
<tr>
<td>22.75</td>
<td>561.36</td>
<td>0.00</td>
<td>0.559327E-01</td>
<td>0.00</td>
</tr>
<tr>
<td>23.45</td>
<td>579.66</td>
<td>0.00</td>
<td>0.500194E-01</td>
<td>0.00</td>
</tr>
<tr>
<td>24.15</td>
<td>598.55</td>
<td>0.00</td>
<td>0.447319E-01</td>
<td>0.00</td>
</tr>
<tr>
<td>24.85</td>
<td>618.07</td>
<td>0.00</td>
<td>0.400040E-01</td>
<td>0.00</td>
</tr>
<tr>
<td>25.55</td>
<td>638.21</td>
<td>0.00</td>
<td>0.357764E-01</td>
<td>0.00</td>
</tr>
<tr>
<td>26.25</td>
<td>659.02</td>
<td>0.00</td>
<td>0.319961E-01</td>
<td>0.00</td>
</tr>
</tbody>
</table>
Chapter 10

Model Task #2: Assessing convective instability

The present task is to assess the convective instability of the sounding we created in Task #1. This instability, called Convective Available Potential Energy (CAPE), is usually gauged for an insulated parcel originating in the lower troposphere that is subsequently raised without mixing with — nor experiencing resistance from — the environment. If the parcel can become positively buoyant, it will convert the potential energy represented by its buoyancy (CAPE) into kinetic energy of motion. In actuality, the parcel will experience some degree of mixing with its surroundings, and has to push air residing above it out of the way. Thus, the CAPE is an overestimate of the true amount of kinetic energy that might be realized by the parcel.

Figure 1 sets up the problem and presents a hypothetical situation. In the grid view we are defining a parcel at the lowest real scalar grid point by specifying its potential temperature and water vapor mixing ratio. Call these $\theta_p$ and $q_{vp}$. (We’re using these properties because they are conserved, at least until the parcel saturates.) Once defined, we will raise the parcel, grid point by grid point, until we reach the model’s topmost scalar point. There are two parts to this problem: first, we compute potential temperature and vapor content along the parcel’s path, and then we assess its instability (if any) relative to its surroundings. We use the fundamental air parcel assumption that the parcel’s pressure equals that of its surroundings at all times.

Call the lowest scalar point index $k_1$; this is $k = 2$ in Fortran and $k = 1$ in C++. When we raise the parcel to the next grid point up ($k = k_1 + 1$), its $\theta_p$ and $q_{vp}$ are
Figure 10.1: Grid and Stuve diagram views of an ascending parcel.
preserved. Figure 1 also sketches what this would look like on a thermodynamic chart like a Stuve diagram. We have ascended one grid point along a dry adiabat. At this juncture, we test to see if the parcel is saturated (that is: is the parcel saturated now, or did it become saturated somewhere along the path?). This is done by computing the parcel’s saturation mixing ratio, $q_{vs}$, a function of its potential temperature $\theta_p$ and (nondimensional) pressure $\bar{\pi}$ (the environment value at the parcel’s height) given by:

$$q_{vs} = \frac{380}{p_0 \bar{\pi}} \exp \left[ \frac{17.27(\theta_p \bar{\pi} - 273.1)}{(\theta_p \bar{\pi} - 36.1)} \right],$$

where it is recalled that $T = \theta \pi$. In the example shown, the parcel is assumed to have just become saturated at $k = k_1+1$, so $q_{vs} = q_{vp}$, so the lifting condensation level (LCL) is at this height level.

Further lifting should now be moist adiabatic, with the parcel continually condensing sufficient vapor to prevent supersaturation at any and every infinitesimal vertical “step”. Note that since pressure varies along the parcel path, this is impossible to handle with precise accuracy in a model (the model’s framework is Eulerian, not Lagrangian, anyway). However, it is not terribly inaccurate to first raise the saturated parcel dry adiabatically – which of course produces a supersaturated parcel) – and then adjust the parcel back to 100% relative humidity at the new height level. This effectively discretizes the saturated adiabatic process into two separate steps, dry adiabatic expansion followed by isobaric saturation adjustment, as shown on the Stuve view in Fig. 1. This approximation gets worse as the vertical grid interval gets larger, of course.

We appreciate that if the parcel is supersaturated, i.e., $q_{vp} > q_{vs}$, the amount of condensation actually realized is less than the difference $q_{vp} - q_{vs}$. This is because of the temperature dependence of the saturation mixing ratio. The parcel is supersaturated because it carries more vapor than it can hold at its temperature. However, as the supersaturated parcel’s vapor condenses, heat is released that raises the parcel’s temperature, increasing its ability to hold vapor. Thus, during the process of condensation, $q_{vp}$ decreases (as vapor condenses) but $q_{vs}$ increases (owing to latent heating).

---

1In avoiding carrying both dimensional and nondimensional pressure, the equation got more complicated. You may decide to create a vector array for both.
We compute the isobaric saturation adjustment in the following way: The amount of vapor that can be condensed from a supersaturated parcel starting with pre-adjustment values of potential temperature $\theta_p$, vapor mixing ratio $q_{vp}$ and saturation mixing ratio $q_{vs}$ is:

$$
C = \frac{q_{vp} - q_{vs}}{1 + \phi}
$$

where $\phi$ is defined as:

$$
\phi = q_{vs}\left[\frac{17.27 \cdot 237 L_v}{c_{pd}(\theta_p \pi - 36.)^2}\right]
$$

and $L_v$ is the latent heat of vaporization ($2.5 \times 10^6$ J kg$^{-1}$). The vapor loss due to condensation is then

$$
q_{vp} = q_{vp} - C,
$$

and the heat released raises the parcel’s potential temperature according to:

$$
\theta_p = \theta_p + \frac{L_v}{c_{pd} \pi} C.
$$

Finally, we wish to compute the sounding’s CAPE. This is an integrated property, defined as:

$$
CAPE = \int_{z_{LFC}}^{z_{EQL}} g \left[\frac{\theta_{pv} - \bar{\theta}_v}{\bar{\theta}_v}\right] \, dz
$$

and represents the “positive area” on a thermodynamic diagram when both the environmental sounding and the parcel path are plotted. As the parcel rises, it may find itself less dense than the environment at some point. The height where this first occurs is called the level of free convection (LFC). Eventually, the parcel will become more dense than the surrounding environment; this height is termed the equilibrium level (abbreviated EQL) and is the first guess at cloud top. CAPE is computed between these two levels.

Note that the formula uses virtual potential temperatures for the parcel and environment ($\theta_{pv}$ and $\bar{\theta}_v$, respectively). These are defined as:

$$
\theta_{pv} = \theta_p (1 + 0.61 q_{vp}),
$$

and

$$
\bar{\theta}_v = \bar{\theta}(1 + 0.61 \bar{q}_v).
$$
Using virtual temperatures allows us to incorporate the effect of moisture on parcel density.

The units of CAPE are J kg\(^{-1}\) or, equivalently, m\(^2\) s\(^{-2}\). The latter suggests that CAPE is a (vertical) velocity squared. In our discussion of gravity waves, we rewrote the vertical equation of motion into a form similar to that below (though now with moisture):

\[
\frac{dw}{dt} = -\frac{1}{\bar{\rho}} \frac{\partial p'}{\partial x} + \frac{\theta'_v}{\theta_v}. 
\]

The hydrostatically balanced mean state has already been removed and the primed values represent the difference between the parcel and its surrounding environment. Given the air parcel assumption of mechanical equilibrium, \(p' = 0\), and the equation is revised to:

\[
\frac{dw}{dt} = g \frac{\theta_p - \bar{\theta}}{\theta}. 
\] (10.1)

Note the right hand side already looks like CAPE, all it needs is to be integrated between the LFC and EQL height levels. If the motion is strictly one-dimensional (in the vertical) and steady with time, then

\[
\frac{dw}{dt} = \frac{dw}{dz} \frac{dz}{dt} = w \frac{dw}{dz} = \frac{1}{2} d\left(\frac{w^2}{dz}\right). 
\]

Integrate (10.1), using the rightmost expression above to replace the left hand side of (10.1). So, after rearranging:

\[
CAPE = \frac{1}{2} \int_{z_{LFC}}^{z_{EQL}} d(w^2) = \frac{1}{2} w_{EQL}^2 - w_{LFC}^2 = \frac{1}{2} w_{EQL}^2. 
\] (10.2)

where it has been assumed that \(w = 0\) at the LFC (not a bad assumption). Thus, the vertical velocity at the estimated cloud top (EQL) depends on the vertically integrated parcel positive buoyancy such that:

\[
w_{EQL} = \sqrt{2CAPE}.
\]

For a CAPE of 2000 J kg\(^{-1}\), such as might be found in a late spring Midwestern environment during the afternoon, this predicts a parcel would strike cloud top with a
vertical velocity of over 63 m s\(^{-1}\). Two things are unrealistic about this value. First, it is too large. Second, maximum vertical velocity is not likely to be found at cloud top, but rather somewhere farther below. (Can you explain why?) Indeed, by the time a real parcel approaches cloud top, its velocity will be much reduced from the largest value it had previously attained, and may even be reasonably close to zero. (Again, can you explain why?)

Here are results assuming an initial parcel potential temperature equal to that of the environment at \(k = 2\), and an initial vapor mixing ratio of 11.5 g kg\(^{-1}\) or 0.0115 kg kg\(^{-1}\). (Use only kg kg\(^{-1}\) in doing your model calculations...). “Diff” is the virtual potential temperature difference between the parcel and environment.

<table>
<thead>
<tr>
<th>(z) (km)</th>
<th>Parcel PT (K)</th>
<th>Parcel Qv (g/kg)</th>
<th>Diff (K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.05</td>
<td>300.52</td>
<td>11.50</td>
<td>-1.73</td>
</tr>
<tr>
<td>1.75</td>
<td>304.11</td>
<td>10.15</td>
<td>.23</td>
</tr>
<tr>
<td>2.45</td>
<td>307.85</td>
<td>8.79</td>
<td>2.13</td>
</tr>
<tr>
<td>3.15</td>
<td>311.51</td>
<td>7.48</td>
<td>3.82</td>
</tr>
<tr>
<td>3.85</td>
<td>315.07</td>
<td>6.24</td>
<td>5.30</td>
</tr>
<tr>
<td>4.55</td>
<td>318.49</td>
<td>5.08</td>
<td>6.25</td>
</tr>
<tr>
<td>5.25</td>
<td>321.70</td>
<td>4.01</td>
<td>6.85</td>
</tr>
<tr>
<td>5.95</td>
<td>324.65</td>
<td>3.07</td>
<td>7.10</td>
</tr>
<tr>
<td>6.65</td>
<td>327.26</td>
<td>2.25</td>
<td>6.97</td>
</tr>
<tr>
<td>7.35</td>
<td>329.48</td>
<td>1.57</td>
<td>6.41</td>
</tr>
<tr>
<td>8.05</td>
<td>331.27</td>
<td>1.04</td>
<td>5.37</td>
</tr>
<tr>
<td>8.75</td>
<td>332.62</td>
<td>.65</td>
<td>3.78</td>
</tr>
<tr>
<td>9.45</td>
<td>333.58</td>
<td>.38</td>
<td>1.76</td>
</tr>
</tbody>
</table>

(LCL, LFC between 1.05 and 1.75 km)

<table>
<thead>
<tr>
<th>(z) (km)</th>
<th>Parcel PT (K)</th>
<th>Parcel Qv (g/kg)</th>
<th>Diff (K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.15</td>
<td>334.21</td>
<td>.21</td>
<td>-6.63</td>
</tr>
<tr>
<td>10.85</td>
<td>334.59</td>
<td>.11</td>
<td>-3.30</td>
</tr>
<tr>
<td>11.55</td>
<td>334.81</td>
<td>.05</td>
<td>-6.17</td>
</tr>
<tr>
<td>12.25</td>
<td>334.93</td>
<td>.02</td>
<td>-12.02</td>
</tr>
<tr>
<td>12.95</td>
<td>334.98</td>
<td>0.01</td>
<td>-23.28</td>
</tr>
<tr>
<td>13.65</td>
<td>335.01</td>
<td>0.00</td>
<td>-34.93</td>
</tr>
<tr>
<td>14.35</td>
<td>335.02</td>
<td>0.00</td>
<td>-46.98</td>
</tr>
<tr>
<td>15.05</td>
<td>335.03</td>
<td>0.00</td>
<td>-59.43</td>
</tr>
<tr>
<td>15.75</td>
<td>335.03</td>
<td>0.00</td>
<td>-72.28</td>
</tr>
<tr>
<td>16.45</td>
<td>335.03</td>
<td>0.00</td>
<td>-85.56</td>
</tr>
<tr>
<td>17.15</td>
<td>335.03</td>
<td>0.00</td>
<td>-99.27</td>
</tr>
<tr>
<td>17.85</td>
<td>335.03</td>
<td>0.00</td>
<td>-113.43</td>
</tr>
<tr>
<td>18.55</td>
<td>335.03</td>
<td>0.00</td>
<td>-128.05</td>
</tr>
<tr>
<td>19.25</td>
<td>335.03</td>
<td>0.00</td>
<td>-143.14</td>
</tr>
<tr>
<td>19.95</td>
<td>335.03</td>
<td>0.00</td>
<td>-158.73</td>
</tr>
<tr>
<td>20.65</td>
<td>335.03</td>
<td>0.00</td>
<td>-174.82</td>
</tr>
<tr>
<td>21.35</td>
<td>335.03</td>
<td>0.00</td>
<td>-191.44</td>
</tr>
</tbody>
</table>

(EQL between 9.45 and 10.15 km)
Vertically integrated CAPE is $1206.65 \text{ J/kg}$