SOAC Project: Cloud: parcel? Model?

Chenxi Qiu (5854814), Edward Groot (6077587)

November 9, 2018

1 Introduction

We model a rising parcel of moist air that is exposed to entrainment under a background atmospheric profile based on observations. These observations are from August 26th 2010 in Essen (Germany) [1]; we investigate the properties of an air parcel that forms a convective cloud in this situation and simulate consequent cloud droplets and precipitation. Aforementioned observations are slightly modified. The model described in this report is Lagrangian and simulates the cloud, including relatively simple parameterisations of microphysical processes. The aim is to use the model to illustrate sensitivity of the produced convective clouds for entrainment processes, phase changes, initial conditions and the environment of the cloud parcel. We also want to clarify the important limitations of the parcel approach to a convective cloud.

2 Methods

2.1 Model setup: state variables

The model that we use for the simulations was inspired by a book of Pruppacher & Klett (2011) [2]. As a starting point, we have the differential equation for vertical velocity

$$\frac{dW}{dt} = \frac{g}{1+\gamma}\left(\frac{T_v - T_v'}{T_v'} - w_L - w_i\right) - \frac{\mu}{1+\gamma}|W|W \tag{1}$$

in which the first term on the right hand side represents the effect of buoyancy ($T_v = T(\frac{1+\epsilon^{-1}w_v}{1+w_v})$ [3][1]), liquid water (w_L) and solid water (w_i) contents on the vertical acceleration of the parcel. The term $\mu|W|W$ represents the effect of entrainment and μ^{-1} can be interpreted as length scale of entrainment (see (6)); the γ term represents the effect of surrounding mass being accelerated due to the displacement of the cloud parcel, which has a negative effect on the acceleration of the cloud parcel itself. The entrainment effect is a mixing effect (trapping of surrounding air by the convective cloud), occurring in the other budget equations as well.
The time derivative of parcel temperature can be expressed as

$$\frac{dT}{dt} = -\frac{gW}{c_{pa}} + \frac{L_v}{c_{pa}}[C_\phi - E_\phi] + \frac{L_s F(\phi)}{c_{pa}} - \mu(T - T')|W| \tag{2}$$

Apart from phase changes (see 2.3) and entrainment, the cooling/warming due to adiabatic vertical displacements is found in the temperature budget equation ($-\frac{gW}{c_{pa}}$). In the equations, ϕ is a vector containing the state variables, for which differential equations are used.
Water vapour mixing ratio of the parcel can be expressed as

$$\frac{dw_v}{dt} = E_\phi - C_\phi - \mu(w_v - w_v')|W| \tag{3}$$

Liquid phase cloud content mixing ratio can be expressed as

$$\frac{dw_L}{dt} = C_\phi - E_\phi - F_\phi - D_\phi - \mu w_L|W| - P_{w,\phi} \tag{4}$$

Solid phase cloud content mixing ratio can be expressed as

$$\frac{dw_i}{dt} = F_\phi + D_\phi - M_\phi - P_{c,\phi} \tag{5}$$

Details on the phase changes that act as source and sink terms in these equations are found in section 2.3, where our deposition equation which is a parameterisation of the term D_ϕ after Rotstayn et al. (2000) [4] is also discussed.
Additionally, we keep track of the mass of the parcel and its pressure (assuming hydrostatic equilibrium,

[1]Some other thermodynamic equations from this reference were also used in the model.

which may not be the most accurate, since the parcel is accelerated during the simulation, but this value is only necessary for calculating saturation mixing ratio of water vapour).

$$\mu m = |\frac{dm}{dz}| = \frac{dm}{dt}|\frac{dt}{dz}| => \frac{dm}{dt} = m\mu|W| \tag{6}$$

$$\frac{dp}{dt} = \frac{dp}{dz}\frac{dz}{dt} = -\rho g W \tag{7}$$

Here, atmospheric density ρ is a function of virtual temperature and pressure (equation of state for an ideal gas).

2.2 Numerical scheme

To calculate environmental values of temperature and water vapour mixing ratio at any level, radio sounding observations are linearly interpolated between two nearest observation levels in the model and extrapolated beyond the range of observations by assuming constant values outside this range.
In the programme, any of the state variables is integrated using an Runge-Kutta 4^{th} order scheme.
We integrate our equations with the Runge-Kutta 4^{th} order scheme (developed by C. Runge and M. Kutta). If we express the differential equation as $\frac{d\phi}{dt} = f(t, \phi)$, the numerical scheme goes as

$$k_1 = \Delta t f(t_n, \phi_n), \tag{8}$$

$$k_2 = \Delta t f\left(t_n + \frac{\Delta t}{2}, \phi_n + \frac{k_1}{2}\right), \tag{9}$$

$$k_3 = \Delta t f\left(t_n + \frac{\Delta t}{2}, \phi_n + \frac{k_2}{2}\right), \tag{10}$$

$$k_4 = \Delta t f(t_n + \Delta t, \phi_n + k_3). \tag{11}$$

Next, we can calculate ϕ for the next time step from the values above

$$\phi_{n+1} = \phi_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4), \tag{12}$$

$$t_{n+1} = t_n + \Delta t \tag{13}$$

Here, n indicates the time step identity number. In our specific case, the functions are not explicitly time-dependent, such that we omit these arguments. All state variables are simultaneously integrated in a function that was build with this purpose.
Pressure is integrated assuming that the density is constant during a time step, which is not problematic, since pressure is only used to calculate w_{vs} and $e_{s,i}$ (see 2.3) and therefore the model is not so sensitive to pressure changes (so only Δt shouldn't be chosen very large to avoid problems, but this holds in general).

2.3 Phase changes and precipitation production

The various processes are all parameterised differently, with two different levels of complexity. Most processes are assumed to be linear and are therefore modelled as exponentially decaying over time, but are in reality essentially non-linear due to interactions and different substeps that dominate other substeps in an ordered way (resulting in non-linearity in the real world and parameterisations of substeps, e.g. Liu & Daum, 2004 [5]; Beheng, 1994 [6]). Our assumption means that we basically simplify many processes to something occurring at a specified time scale, which can be fitted to the model.
An overview is given below.
Condensation is calculated with (14), provided that an air parcel is supersaturated ($w_{vs}(T, p) < w_v$)

$$C_\phi = (w_v - w_{vs}(T, p))(1 - exp\left(\frac{\Delta t}{\tau_c}\right)) \tag{14}$$

Evaporation is calculated with (15), provided that the parcel contains cloud water ($w_L > 0$) and that its relative humidity is lower than 100%.

$$E_\phi = c_{evap} w_L (w_{vs}(T,p) - w_v)(1 - exp\left(\frac{\Delta t}{\tau_e}\right)) \tag{15}$$

Here, c_{evap} is implemented to make evaporation really occur at the time scale τ_e that we define in the model (since $w_L << 1$). Based on a typical w_L, we can give a value to c_{evap} and our typical value is 0.7 g/kg, which means that $C_{evap} = 1400 \approx \frac{1}{0.0007}$.
Production of warm precipitation is evaluated with (16) if $w_L > w_{L,t}$

$$P_{w,\phi} = (w_L - w_{L,t})(1 - exp\left(\frac{\Delta t}{\tau_w}\right)) \tag{16}$$

Similarly, cold precipitation is produced if the cloud ice content is larger than a defined threshold and this is calculated with (17)

$$P_{c,\phi} = (w_i - w_{i,t})(1 - exp\left(\frac{\Delta t}{\tau_{pc}}\right)) \tag{17}$$

Cloud liquid water could play a role for cold precipitation production, but the usual pathway for this will be taken into account via the deposition of cloud liquid water on ice crystals (see 19 - 23). Alternatively, precipitating ice may accrete with liquid water at positive temperatures when the cloud contains melting ice that precipitates, but this is cannot be taken into account, since our model only allows one parcel temperature for each time step.
The conversion between cloud liquid water and cloud ice content is dependent on temperature. If the temperature of the cloud parcel reaches 235 K or -38 °C, any liquid water present always freezes (instantaneously $\tau_f = 0$). Similarly, any cloud ice occurring at temperatures above 0 °C or 273.15 K, melts completely (also instantaneously). At intermediate temperatures, a mixed phase cloud may exist. Conversion between cloud liquid water and cloud ice is parameterised after Rotstayn et al. (2000) [4]. They use a parameterisation for the specific cloud ice content (q_i), whereas we use the cloud ice mixing ratio (w_i). These differ by a factor $\frac{\rho_d}{\rho_d + \rho_v}$, which can be assumed to be exactly 1.0 with an error much smaller than the mean error of a cloud ice content parameterisation. Therefore, neglecting this difference and interchanging the symbols is not incorrect. Hereto, we use the following equations to parameterise ice crystals in our cloud (given that the temperature is in the range where mixed phase clouds occur, 235 to 273.15 K):

$$D_\phi = \frac{w_{i,D}(t + \Delta t) - w_i(t)}{\Delta t} \tag{18}$$

$$w_{i,D}(t + \Delta t) = (\frac{2}{3} c_{vd}(T,\rho,p)\Delta t + w_i(t)^{\frac{2}{3}})^{\frac{3}{2}} \tag{19}$$

$$c_{vd}(T,\rho,p) = 7.8 c_{conv} \frac{N_i(T,p)}{\rho}^{\frac{2}{3}} \frac{e_{s,L}(T) - e_{s,i}(T,p)}{\rho_i^{\frac{1}{3}} (A(T) + B(T,p)) e_{s,i}(T,p)} \tag{20}$$

$$N_i(T,p) = 1000 exp(12.96 \frac{e_{s,L}(T) - e_{s,i}(T,p)}{e_{s,i}(T,p)} - 0.639) \tag{21}$$

$$A(T) = \frac{L_s}{K_a T}(\frac{L_s}{R_v T} - 1) = \frac{L_s}{0.024T}(\frac{L_s}{R_v T} - 1) \tag{22}$$

$$B(T,p) = R_v T \chi(p) e_{s,i}(T,p) = R_v T \frac{2.21}{p} e_{s,i}(T,p) \tag{23}$$

In these equations, $e_{s,i}(T,p)$ is the saturation vapour pressure over ice and $e_{s,L}(T)$ is the saturation vapour pressure over liquid water (see list of symbols in the Appendix A, table 2).
Because Rotstayn et al. (2000) [4] studied stratocumulus clouds that do not extend so deep vertically (shallow stratocumulus) and they argue that the conversion to ice would be faster in clouds with larger vertical extend, their c_{vd} is multiplied with an adjustable c_{conv} which we set by default to 10.0. The higher

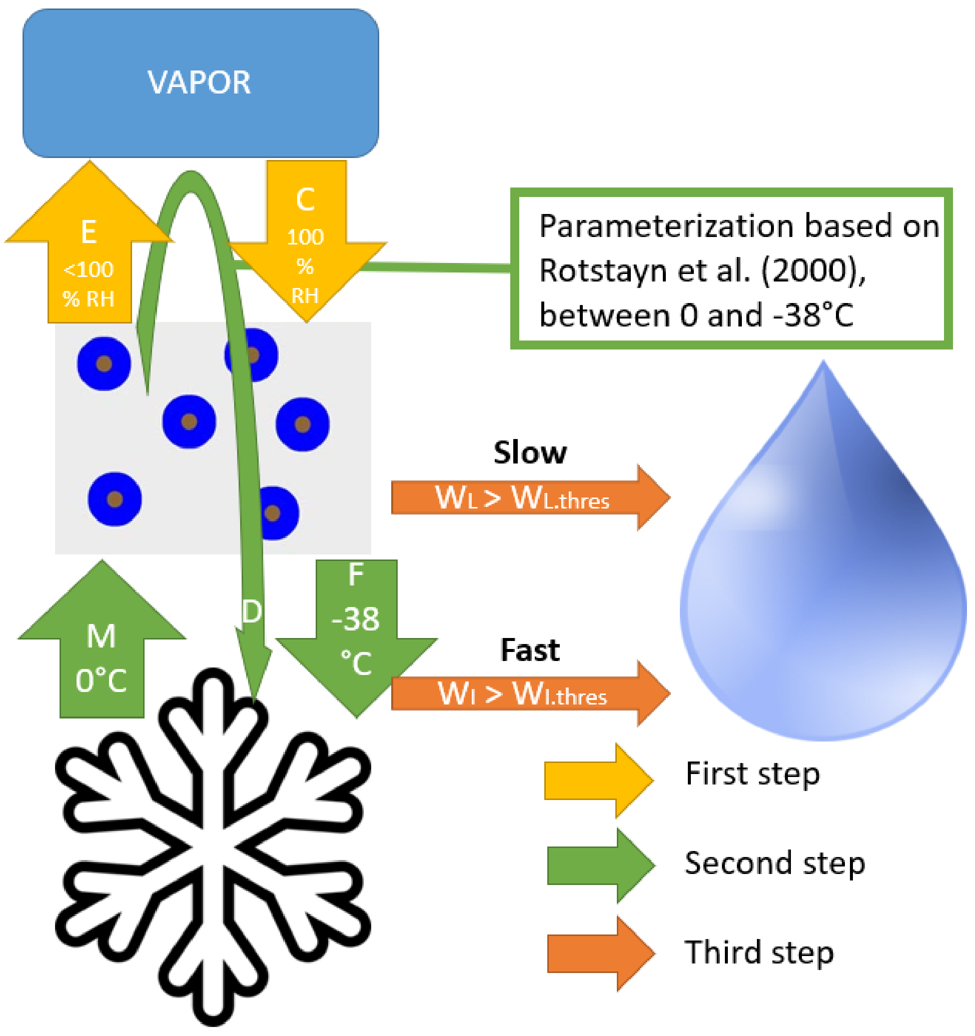


Figure 1: Phase diagram of the cloud parcel model

conversion rate for vertically deeper clouds with higher vertical velocities is consistent with intuition, because larger and vertically more extensive gradients in buoyancy will lead to the tendency of cloud parcels to rise faster which will usually lead to an extra production of turbulence in deep convective clouds compared to shallow and layered clouds.

2.4 Entrainment

The cloud parcel model was extended by assuming that μ (see (1)-(6)) is a function of the cloud parcel size (and thus parcel mass). Alternatively, it can be set to any (non-negative) constant value in the model code (which we did initially). Pruppacher & Klett [2] note that the interaction with environmental parcels at the cloud top is much stronger than elsewhere at the edge of the cloud. Therefore, the relation used was:

$$\mu(R) = \frac{c_{inv}}{R_{eq}} + \mu_0 \tag{24}$$

$\frac{C_{inv}}{R_{eq}}$ was based on E. Kessler (1969) [7], R.A. Anthes (1977) [8] and Pruppacher & Klett (2011) [2]. In reality, our cloud is not a spheroid or any other easily describable geometrical object (but deforms all the time) and in the model it is limited one point in space by representing it with one value for each state variable (instead of a field within and around the cloud), but at the same time, it is assumed for entrainment that it can be represented by such a geometrical shape with at least an equivalent radius (R_{eq}) on which the entrainment is based. The variables R_{eq} and m can be inter-converted assuming a spherical shape of the cloud and using ρ (see appendix B, code of the model).

2.5 Initialisation of the parcel

The parcel is initialised as being situated at a certain height and may obtain the state variables of observations at this point plus an additional disturbed part. But its state variables may also be obtained based on the mean of a certain layer that is initially mixed, plus disturbance. Besides this, no cloud (ice content and liquid water content) content is assumed to exist initially. The parcel can be forced to a certain vertical velocity as initial value and it is also given a certain initial equivalent radius, which implies a certain mass.
To initialise pressure, hydrostatic equilibrium ($\frac{dp}{dz}$ in 7) is integrated by calculating ρ' based on T_v' that is linearly interpolated from observations of water vapour and temperature. A triangular upward integrating scheme from the observation point below the parcel is used in this case, calculating density at half-levels and with $\Delta z = 0.1$ m (by default).
Before the integration is really started, T' and w_v' have to be initiated as well and w_{vs} is calculated to obtain an initial μ (via ρ' and R_{eq}), condensation and evaporation (0, since $w_L = 0$) value. Ice production is also initiated at zero.
The Essen observations [1] are slightly modified in our reference profile. A convective precipitation event during the evening of August 26th 2010 in the Dutch Acherhoek region was used as an inspiration for this modified profile. The reason for the modification is that we assume that pre-existing cumulus clouds have transported significant amounts of moisture to the dry layer at 2500-3000 m after the sounding was done (12 UTC), removing a layer with (approximately) dry adiabatic lapse rates at that height.[2]
The initial parcel conditions in our experiment were used from the same observations and slightly disturbed by warming of 0.4 K and moistening of 0.2 g/kg, with initial parcel height at approximately ground level in Essen (150 m). These conditions should be fairly representative for the pre-convective atmosphere in the Dutch Achterhoek region (only 70 km from Essen).

2.6 Structure of integration step

During one step of the integration, the starting point is calculating local ρ of the atmosphere and current mass and R_{eq} of the parcel. These values lead to a μ. After this, all state variables in ϕ are simultaneously integrated using the method mentioned in 2.2. Subsequently, the values are stored and based on these, the parcels' T' and w_v' are the first ones to be updated. Subsequently, the saturation mixing ratio for water vapour is updated to calculate condensation and evaporation and then ice may be produced based on the new water content in the cloud. Having produced ice, the final step to be done is calculation of warm and cold precipitation production, which then is subtracted from both cloud contents.
In principal, the model is integrated over 5 hours with time steps (with steps of 0.1 s). Most interesting processes will play on shorter time scales of 1-2 hours, and it has also been checked that nothing interesting would happen after these five hours for some simulations which were still not in equilibrium on relatively long time scales among simulations that were done. However, in reality, processes not taken into account will usually become relevant within these 5 hours (on a scale of maybe 1-2 hours; non-stationary environment for example).
After the model has run, precipitation production is also calculated. This regards an areal average over the region over which the parcel is located considering its R_{eq} in the final state. This final size depends strongly on integration time, as air is always entrained, but also on initial size, covered path and final parcel height (through density, which is used to calculate R_{eq}).

[2]The moisture flux towards that level turns out to be reproducible with certain model simulations, which is a justification for this modification.

3 Results & discussion

3.1 First model evaluation

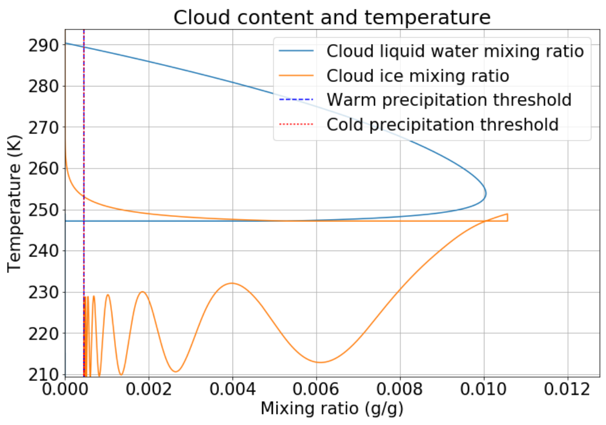
During the model development, we kept track of total water (fallen as precipitation, cloud water, cloud ice and vapour) in 'no entrainment runs' to see that the model is conserving water. During the development, it indicated that the model is physically not consistent at some point, but then we could resolve the error. In the final version of the model, water was indeed conserved for no-entrainment runs. However, water conservation only does not imply that other aspects of the model are also certainly correct.
Moreover, some simulations (e.g. when using a mixed layer parcel instead of an surface based parcel) revealed large sensitivity to time step in, for example, total precipitation and the timing of certain phase change events. This was largely related to the phase of the cloud content as the exact result of the equilibrium temperature differed by a few K (around 273 K) and due to resulting cloud content, precipitation would also be influenced significantly. Changing the time step from 0.4 to 0.1 seconds revealed a large sensitivity for the reference run with a mixed layer, but the 0.01 and 0.1 seconds runs were very similar. The fact that the model is very sensitive to initial conditions and small adjustments in parameter values illustrates the nature of such system (and the real world) where small changes in condition can lead to huge difference later in time.

3.2 Role of entrainment

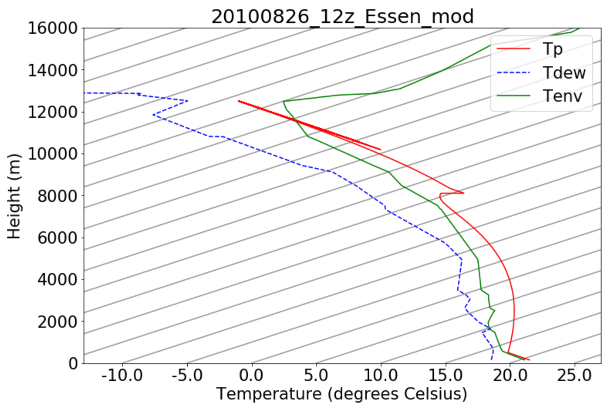
In figure 2, the no entrainment solution of this cloud parcel model is visualised, containing the vertical profile (similar to a Skew-T diagram) and the phase of water in this cloud. This simulation can be compared to constant entrainment in figure 3 and the dynamic entrainment model in figure 4. First of all, we can notice that the cloud top lowers significantly when entrainment is included. This is easily understood: entrainment dilutes strongly positively (or negatively) buoyant air parcels with (approximately) neutrally buoyant air parcels, which decreases the parcels acceleration and resulting speed and thus acts in a way similar to friction.
The dynamic entrainment run of which the results are shown in figure 4 is our reference run in subsequent sections.
Additionally, a big difference between the three simulations is the equilibrium level around which the parcel will oscillate after several hours. In figure 2, this is at $\approx 220K$; for figure 3, this is at $\approx 229K$ and in figure 4 this is around $\approx 260K$. In addition to the effect of buoyancy mentioned earlier, a large portion of this difference is because the former two simulations get an strong extra impulse by heat of fusion due to ice formation, while the ice content in the dynamic entrainment run remains low and therefore heat of fusion doesn't push the parcel so much. Therefore it stays warmer than about 258 K. As such, the intensity of entrainment and heat of fusion can importantly feedback on the (vertical) destination of the cloud.
In figure 2, the parcel makes it to the tropopause in one go, whereas in figure 3, the parcel temporarily descends due to a large waterload (and iceload; see 1) which overwhelms the buoyancy acceleration; subsequently, upon the release of heat of fusion by ice formation, which happens at high rates for temperatures of 254-262 K, extra warmth is released and abundant buoyancy with respect to environment will push it up again; this is also related to the water that leaves the parcel by the formation of precipitation (reducing the water- and iceload). In reality, the formation of precipitation will still contribute to the momentum change of the parcel (equation 1; not in our model), but precipitation will relative to the air parcel fall as well (mutual influence that gradually diminishes when the fall velocity of precipitation increases). This can not be included in our parcel model and is an inherent limitation of our approach: our approach assumes a cloud mass with coherent vertical velocity that is a "parcel". However, the consequences of this may be that in reality the horizontal dynamics (wind shear) either deforms the parcel such that the downward motion caused by waterload is separately sorted in space from the engine of the convective cell (strongly buoyant air), or it may not be the case and then the precipitation formation will partly draw along the buoyant air that is feeding the convective cloud at some point (see also Wallace and Hobbs, 2006 [3]). Additionally, the buoyant air will become less buoyant due to cooling by evaporating precipitation below cloud base and this will result eventually in dissipation of the convective cloud (that is no longer fed).
In figure 4, this waterload and iceload effect allows the parcel to oscillate around the 269 K temperature

level, due to larger cloud water contents at the end of an ascending episode which brings the parcel down; still having some buoyancy, which becomes more abundant at the 270-273 K level due to a relatively low environmental temperature there, it starts rising again a bit below that level and subsequently, the parcel undergoes a damped oscillation. In the meanwhile, precipitation is formed at exponential rates bringing cloud content to $w_{L,thres}$ (or $w_{i,trhes}$, if mainly ice content) but the parcel is also diluted slightly by entrainment which neutralises buoyancy; both contributions slightly different and therefore, the parcel still gradually ascents (net change; buoyancy is reduced less efficiently than water- or iceload). This leads to ice formation in the end and cold precipitation bringing the cloud to threshold ice content.
Another interesting feature of the constant entrainment simulation (figure 3) is that around 6000 m height, the parcel undergoes an (approximately) dry-adiabatic lapse rate. This is because the parcel (at 259-262 K temperature) becomes unsaturated due to downward motions caused by waterload and cloud water has all been converted into ice. In reality, this would lead to sublimation, but this is not included in our model (see figure 1). Saturation vapour pressure over ice ($e_{s,i}$, which is lower than over water) should be used to model this and it would slightly modify the path of the parcel. The parcel would follow an almost saturated (with respect to ice) adiabatic lapse rate. However, in the simulation the downward (more or less) dry-adiabatic part is neutralised by an upward (more or less) dry-adiabatic part and having passed the condensation level again, the only improvement to make it more realistic would be adding sublimation. A consequence of this (close to) dry-adiabatic lapse rate motion is a vertical momentum gain that is stronger than would occur in reality.

The values of C_{inv} and μ_0 were also changed in some experiments. Upon setting $\mu_0 = 0.00$, the simu-



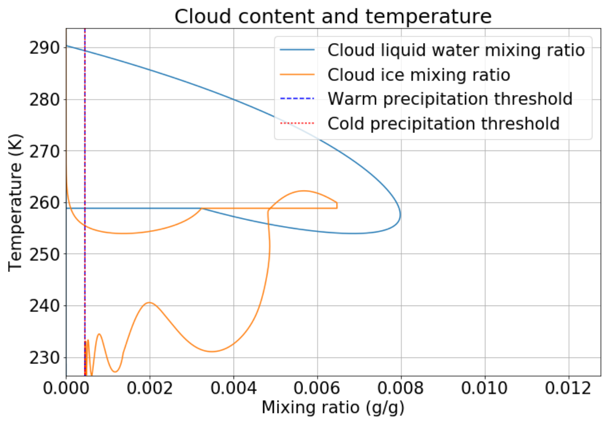
(a) Cloud content as function of temperature.



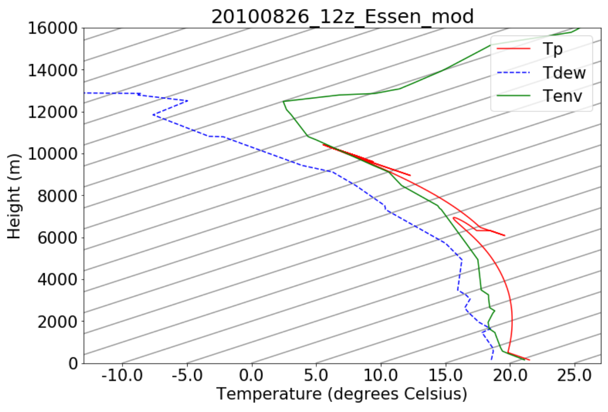
(b) Temperatures as function of z (grey isotherms are tilted).

Figure 2: Result of the cloud parcel model integration without entrainment.

lation was very similar to the constant entrainment case in figure 3. However, the choice of intermediate values lead to different results. For $\mu_0 = 3.5e-5$, the result was already very similar to figure 4, but at $2e-5$, the cloud top goes to an intermediate level (about 9 km) due to intermediate dilution. The total precipitation is slightly less at 4.3 mm, which is related to a combined effect of drier environmental air and more entrainment due to a larger vertical path, but also more water content condensation and resulting ice formation in the model due to lower saturation values in the region where the parcel ends up. In this range of μ_0, precipitation is very sensitive in our model.
Putting C_{inv} to 0.08 gives a result very similar to figure 5. However, changing this value to 0.12 seriously affects the outcome of the cloud parcel model. As can also be seen in 5, the ice formed at about 270 K almost gets to melting point during the short descend in the model in the similar 0.08-run, due to a 2 K temperature increase by heat of fusion. In the 0.12-run, it actually melts. Net it costs heat, but the parcel still decelerates downward and accelerates upward subsequently. After this, oscillations around 268 K are done with net ascend and in the end upon heat released by refreezing as it net ascends above 265 K, brings the parcel towards 256 K finally. This complicated behaviour just illustrates how complicated a simulation by such a simplified model can be. The illustrated behaviour will not exactly be observed in reality, as was described earlier in this section.

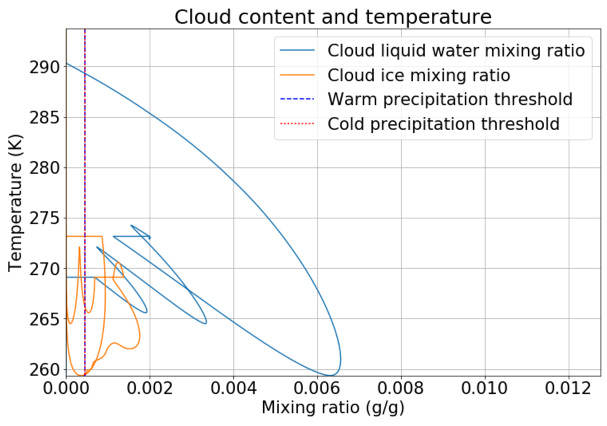


(a) Cloud content as function of temperature.

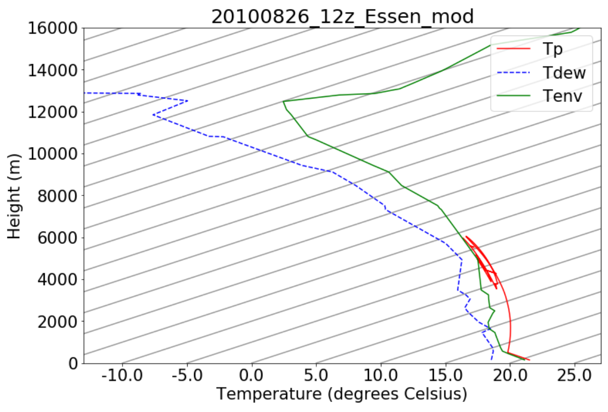


(b) Temperatures as function of z (grey isotherms are tilted).

Figure 3: Result of the cloud parcel model integration with constant, $\mu_0 = 5e^{-5}$

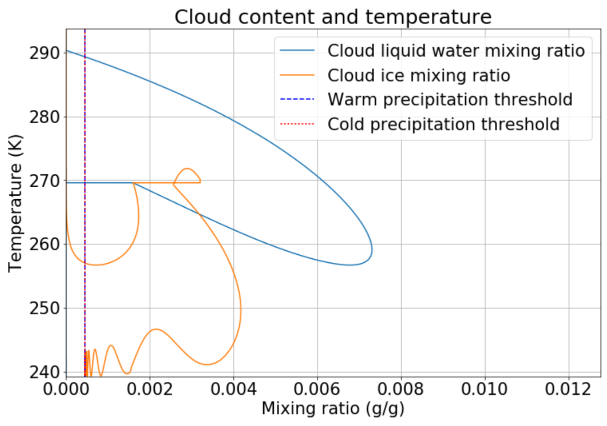


(a) Cloud content as function of temperature.

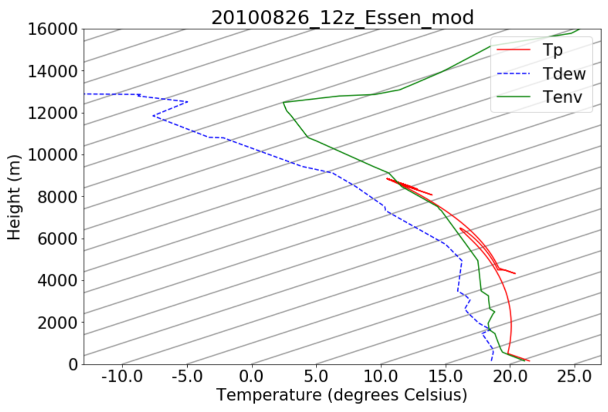


(b) Temperatures as function of z (grey isotherms are tilted).

Figure 4: Result of the cloud parcel model integration with constant, $\mu = \mu(R_{eq})$

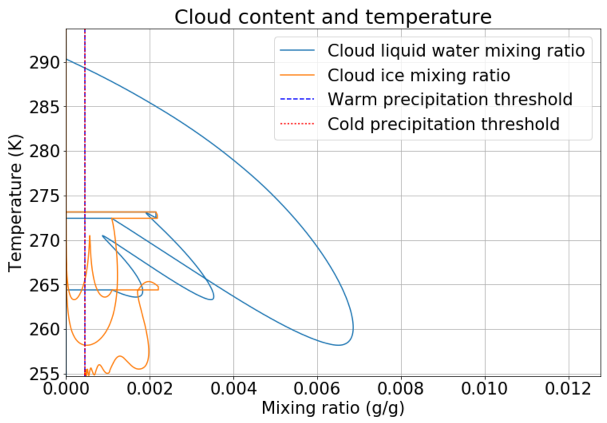


(a) Cloud content as function of temperature.

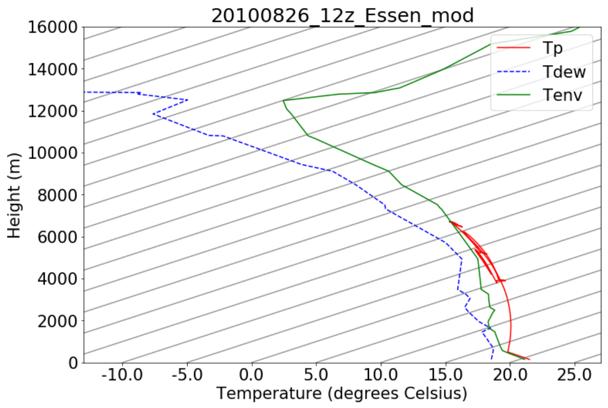


(b) Temperatures as function of z (grey isotherms are tilted).

Figure 5: Result of the cloud parcel model integration with $\mu_0 = 2e - 5$



(a) Cloud content as function of temperature.

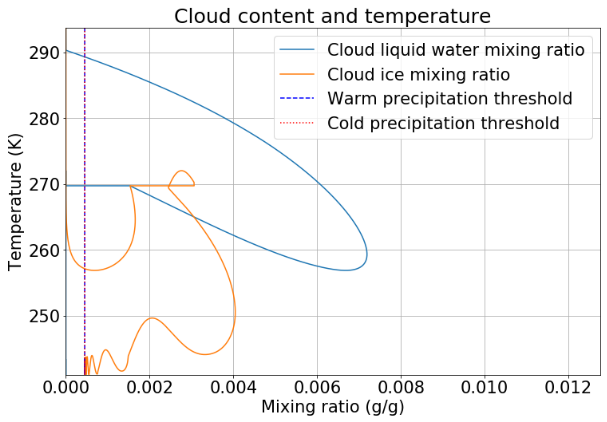


(b) Temperatures as function of z (grey isotherms are tilted).

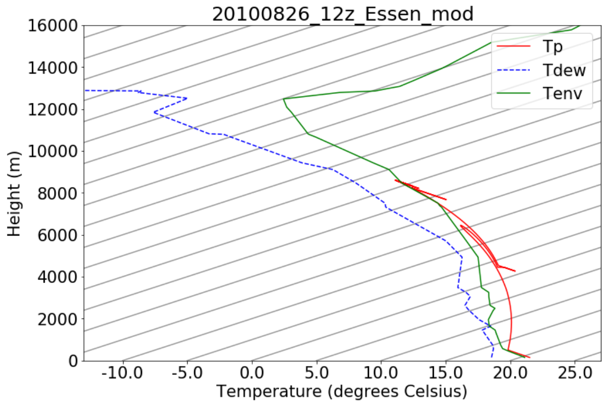
Figure 6: Result of the cloud parcel model integration with $C_{invr} = 0.12$

3.3 Parcel size

When R_{eq} is initially set to 10 m, which is a size for an intermediate to large boundary layer eddy, no cloud is produced. For a value of 100 m, the produced cloud is negligible in size (with cloud top at 279 K, whereas surface temperature is 293 K) and the potential of a deep convective cloud is not realised. A small (warm) shower is produced when we initialise the parcel at 500 m, but very little precipitation is formed and ice is insignificant, as the minimal parcel temperature only reaches 268 K.
The result for the inital radius at 2500 m was already shown in figure 4, since this is the reference run. In figure 7, we can see that an initially larger parcel of 5000 m leads to a colder and higher cloud top, with after some oscillations (noting that the parcel would behave differently in reality as has been pointed out in 3.2) reaches another, higher, equilibrium level at 242 K temperature (near 8000 m height). This new run produces much more precipitation over the 5 hours simulation below the parcel: on average 9.2 mm, compared to 5.0 mm in the reference run. A larger parcel naturally limits the influence of environmental air (magnitude dilution of the "initial air") and the water content halfway the ascent is still higher with an approximately 10% difference between figures 4 and 7. After ice conversion, the difference with aforementioned runs increases at constant temperature (or equivalently saturation vapor pressure). Additionally, the parcel itself grows less in time. In other words, in the smaller parcel case, the water content produced is distributed over a larger region than the source area, but this effect is smaller for larger parcels.



(a) Cloud content as function of temperature.

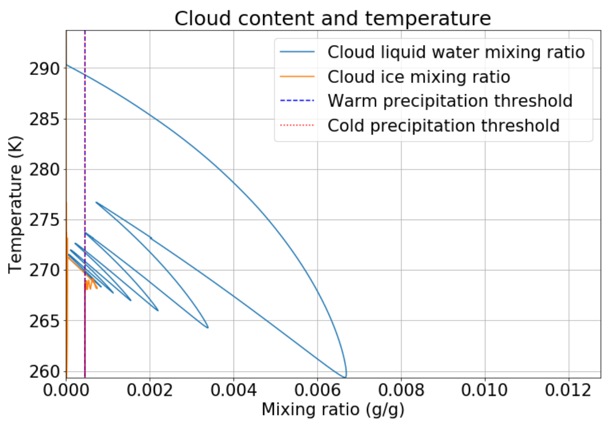


(b) Temperatures as function of z (grey isotherms are tilted).

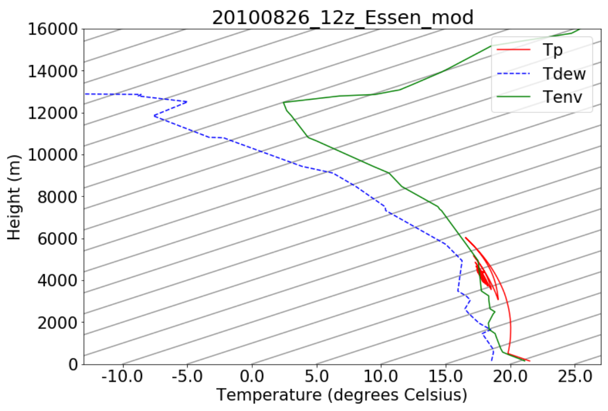
Figure 7: Result of the cloud parcel model integration with $R_{eq,init} = 5000m$

3.4 Ice-water conversion

When ice forms just as in a shallow stratocumulus layer with $C_{conv} = 1.00$ (Rotstayn et al., 2000 [4]), and not much faster in our deep convective cloud compared to aforementioned type of cloud (see 2.3), the time scale of ice formation is multiple hours. This is because the cloud is only slightly below freezing point (260-273 K) and most of the time near 270 K (figure 8), where the parcel is approximately neutrally buoyant. This limits the precipitation formation almost completely to warm processes and leads to only 1.6 mm precipitation. It is a good illustration that deep convective clouds and resulting showers are highly dependent on the ice formation processes. However, as argued by Rotstayn et al. and in 2.3, our approach is probably not the best possible ice parameterisation for a convective cloud. Still being useful for illustrative experiments, changing C_{conv} to 2.5 or 5.0 already has a large impact; the latter result is shown in figure 9. A change from 5.0 to 10.0 lifts the neutral buoyancy level from approximately 267 K to 259-260 K (figure 4) and doubles precipitation from 2.5 to 5.0 mm. This is mainly due to the release of some heat of fusion during the ascend.
Increasing the value of C_{conv} to 25.0 lifts the cloud top further and almost removes the downward motion (and oscillations) which will in reality also not occur (see section 3.2). Once the parcel reaches a level at which it has a temperature of about 265 K, ice formation is done in a less than a kilometre (vertically). The cloud top equilibrium level is now situated at about 242 K. However, precipitation appears to be very insensitive to this change: 5.1 mm compared to 5.0 mm in the reference run.



(a) Cloud content as function of temperature.

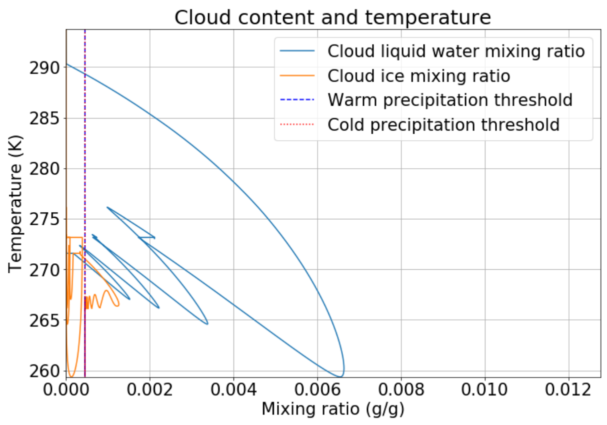


(b) Temperatures as function of z (grey isotherms are tilted).

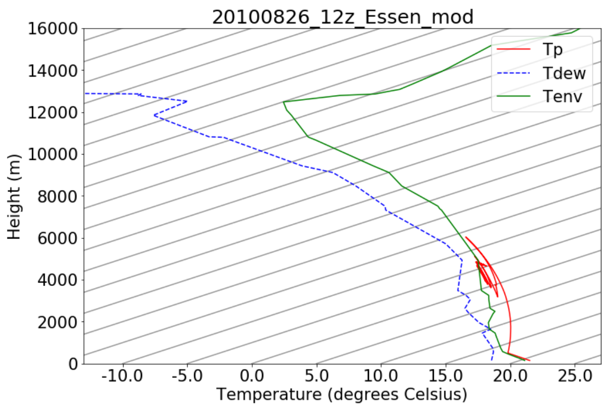
Figure 8: Result of the cloud parcel model integration with $C_{conv} = 1.0$

3.5 Upper air drying

So far, we have looked at the parameter sensitivity of the model. But the initial conditions and environment of the parcel are most interesting to look at, because these vary between different atmospheric conditions. The atmospheric profile used in this report largely originates from a flooding event in the Netherlands, as mentioned in 2.5. Therefore, it is interesting to look at how sensitive the model is to changes in the parcel and its environmental atmosphere; this section regards the environment.
We have adjusted the environmental water vapour profile above $z = 2000m$ in experiments and compared these to the control experiment. At a 5% smaller environmental water vapour content, the outcome of the model is already seriously affected. The cloud top reaches a vertically lower equilibrium level at about 268 K, due to less buoyancy (compare to the difference between figures 4 and 9). This warmer cloud top is due to a larger sink of moisture in the parcel by entrainment of drier ambient air, hereby loosing a little buoyancy. But this moisture is essential for the top height and precipitation. Precipitation is reduced from 5.0 to 2.3 mm (-53%), which results from the smaller maximum ice content (less than 1 g/kg in this run as compared to almost 2 g/kg as maximum in figure 4).
Decreasing the water vapour content by 25%, does affect the maximum elevation of the parcel only a bit

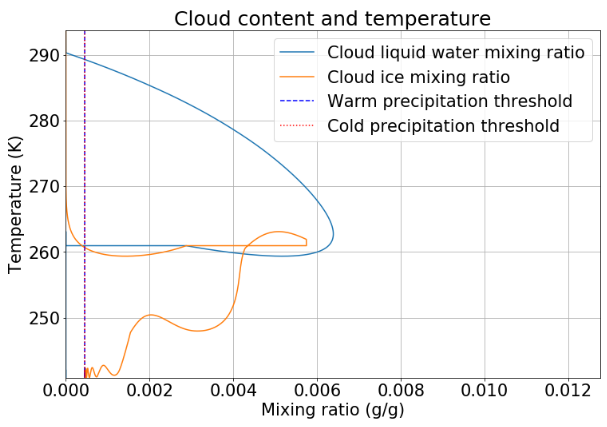


(a) Cloud content as function of temperature.

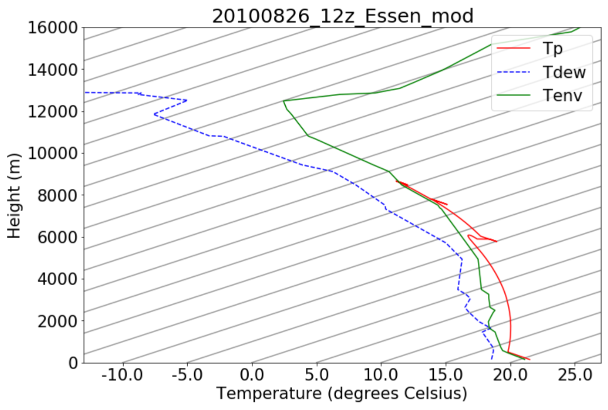


(b) Temperatures as function of z (grey isotherms are tilted).

Figure 9: Result of the cloud parcel model integration with $C_{conv} = 5.0$



(a) Cloud content as function of temperature.



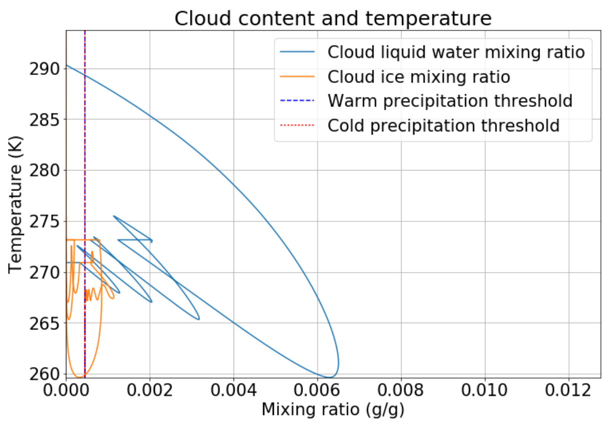
(b) Temperatures as function of z (grey isotherms are tilted).

Figure 10: Result of the cloud parcel model integration with $C_{conv} = 25.0$

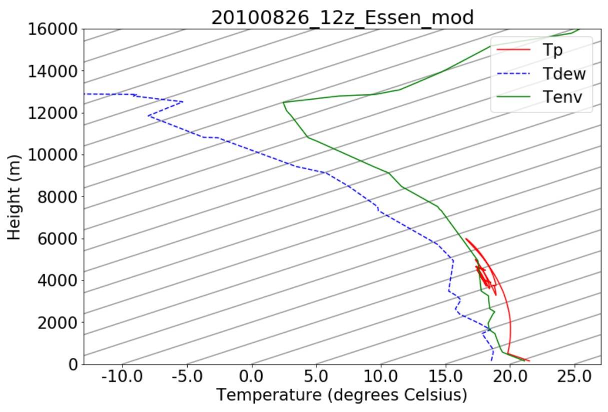
Table 1: Final cloud temperature and areal average precipitation after a full simulation with a dried environmental moisture profile.

Fractional drying (-)	Precipitation (mm)	"Equilibrium" cloud temperature (K)
0.00	4.99	260
0.01	4.70	261
0.02	4.02	264
0.03	3.44	265
0.04	2.50	267
0.05	2.32	268
0.25	0.34	277
0.50	0.18	278

(figure 12), but ice content only slightly exceeds threshold for cold precipitation (shortly). Then, due to dilution by entrainment, the parcel will reach an equilibrium at temperatures of 273-280 K and the shower stops soon. It has also lost cloud water at some points, with the parcel experiencing (approximately) dry adiabatic lapse rates oscillating around a level just below 278 K. This simulation leads to only 0.3 mm precipitation.
In short, we can show with the model that potential showers are very sensitive to moisture in the middle troposphere due to the entrainment processes.
Other results for experiments in which we applied drying of the upper air above $z = 2000m$ are shown in table 1.



(a) Cloud content as function of temperature.

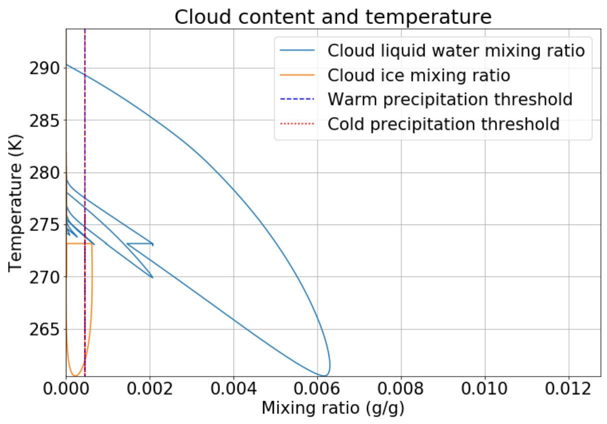


(b) Temperatures as function of z (grey isotherms are tilted).

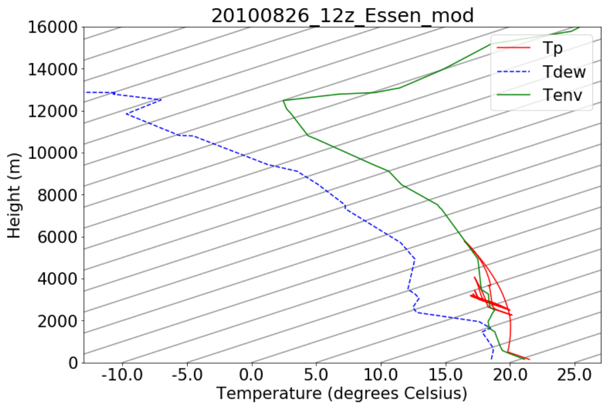
Figure 11: Result of the cloud parcel model integration with w'_v multiplied by 0.95

3.6 Other parameters & initial conditions

With other initial parcel conditions, namely a temperature disturbance of 1.0 K instead of 0.4 K (maintaining the water vapour disturbance), the final cloud top reaches a similar equilibrium height and temperature as in figure 7, due to enhanced (positive) buoyancy if the initial temperature is higher, compared to the reference simulation. However, the way it gets there is different and compares well to 3, but with slightly higher μ. This leads to lower moisture contents compared to both other simulations mentioned during the ascend. Total precipitation is more than in both of the two simulations, which seems to be partly related to the faster onset of precipitation. However, one would expect the lower entrainment value (μ) in the case with constant entrainment to be more important for precipitation that results (inituitively leading to less precipitation due to less water vapor in a case with more entrainment). Part of this counter-intuitive and seemingly contradictory difference is caused by the fact that even with lower μ, the assumed final



(a) Cloud content as function of temperature.



(b) Temperatures as function of z (grey isotherms are tilted).

Figure 12: Result of the cloud parcel model integration with w'_v multiplied by 0.75

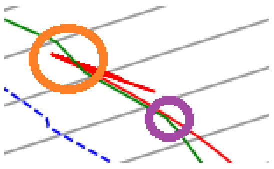


Figure 13: Zoom-in on figure 5

area over which the parcel floats (based on the last value of R_{eq}) is larger in the constant entrainment simulation of figure 3 (compared to this simulation) as could be; see section 2.6.
Increasing the water vapour content disturbance from 0.2 g/kg to 0.5 g/kg, we see almost the same result as for increasing the temperature disturbance from 0.4 to 1.0 K. This is because initial buoyancy is affected in both simulations in the same way. However, numerically, the differences in initial buoyancy between both initial conditions are not exactly the same.
By giving one of the two disturbances a sufficiently negative value (without changing the other), we can suppress convective clouds and parcels would barely be lifted outside the planetary boundary layer or at some point not be lifted outside this layer.

3.7 Sensitivity

In short, we can show non-linearity of convective clouds and related sensitivity to initial conditions, phase changes, precipitation formation and cloud environment. An important point is that environmental lapse rates are in certain regions usually accelerating the parcel a mostly, but decelerating in other localised regions (most notably, if a convective cloud reaches the tropopause, there and possibly at the location that caps the boundary layer). As a result, cloud may be after passing a region like the one marked in purple in figure 13 just not be able to penetrate into the layer above and sink, but it also may just penetrate and then be able to go to the region marked by an orange circle in the figure. In the case we studied, convective clouds are often attracted to a level of 240-242 K, but also to 260 K and rarely to levels in between. In the visual appearance of these convective clouds, this may lead to an anvil. This is the most important non-linearity in convective showers, because it can in reality often determine their existence and how deep they will become.

4 Conclusion

We have shown that the model in this report allows us to describe the evolution of deep convective clouds in detail. It allows us to study important aspects of the non-linearity, which were illustrated for an example

case and mainly with precipitation and cloud top height: we have demonstrated some sensitivities of the model to initial conditions, phase changes, precipitation formation and the environmental atmosphere. Precipitation from a convective cloud may be highly sensitive on all these and sometimes change counter-intuitively. An important feature that can be demonstrated with the model includes that the convective cloud has certain vertical levels that are more attractive as cloud top than other levels (see section 3.7). Moreover, we saw that in addition to latent heat release, but to a much smaller extent, release of heat of fusion can be an important push for convective clouds. However, the parcel approach has some limitations, most notably:
- The influence of horizontal dynamics can not be taken into account in a satisfying way.
- Once waterload or iceload is converted into precipitation, it can not affect the vertical evolution of the cloud anymore as it leaves the cloud, although it would in reality (gradually decreasing decoupling of the parcels air reservoir and the precipitation).
- Interrelations between co-existing or subsequent cloud cells which occur in reality can not be taken into account in our simplified model in which we assume environmental air to be stationary.

5 Appendix A

A list of variables is provided in table 2.

Table 2: List of symbols

variables	name	unit	value
A	Heat conduction term in ice deposition parameterisation		see equation 22
B	Vapour diffusion term in ice deposition parameterisation		see equation 23
C_{conv}	Rate of ice formation in deep convective cloud over that of a shallow stratocumulus cloud		10
C_{evap}	Constant for proportionality of evaporation		1400
C_{inv}	Considering convective cloud as jet: twice the sine of angle at which the jet entrains air (if $\mu_0 = 0.0$)		0.16
c_{pa}	Heat capacity of dry air	J/(kg·K)	1005
C_{vd}	Rate of water vapour deposition on cloud ice	$(\mathrm{kg/kg})^{\frac{2}{3}}\mathrm{s}^{-1}$	
C_{ϕ}	Condensation rate in the parcel	kg/(kg·s)	
$\frac{d}{dt}$	Material time derivative in differential equation	$[\ ]\mathrm{s}^{-1}$	
$\frac{d}{dz}$	Material vertical derivative in differential equation	$[\ ]\mathrm{m}^{-1}$	
D_{ϕ}	Deposition rate of cloud ice in the parcel	kg/(kg·s)	
$e_{s,i}$	Saturation vapour pressure over ice surface	Pa	$e_{s,i} = e_{s,i}(T)$
$e_{s,L}$	Saturation vapour pressure over liquid water surface	Pa	$e_{s,L} = e_{s,L}(T)$
E_{ϕ}	Evaporation rate in the parcel	kg/(kg·s)	
F_{ϕ}	Freezing rate in the parcel	kg/(kg·s)	
g	Gravitational acceleration	$\mathrm{m/s^2}$	9.81
$k_{1...4}$	Runge-Kutta constants within integration step		
K_a	Thermal conductivity of air	w/K	2.4×10^{-2}
L_s	Latent heat of fusion	J/kg	3.35×10^{5}
L_v	Latent heat of vaporisation	J/kg	$L_v = L_v(T) \approx 2.5 \times 10^{6}$
m	Parcel mass	kg	
M_{ϕ}	Melting rate in the parcel	kg/(kg·s)	
N_i	Number density of ice crystals in a cloud		
p	Pressure	Pa	
P_c	Cold precipitation formation at time step	mm	
P_w	Warm precipitation formation at time step	mm	
R_d	Gas constant for dry air	J/(kg·K)	287.05
R_{eq}	Radius parcel, assuming it would be spherical	m	
R_v	Gas constant for water vapour	J/(kg·K)	461.5
T	Temperature of the air parcel	K	
T'	Temperature of the environment	K	
T_v, T_v'	Virtual temperature/density temperature	K	$T_v(T, w_v) = T(\frac{1+\epsilon^{-1}w_v}{1+w_v})$
W	Vertical velocity	m/s	
w_i	Ice mixing ratio of the parcel	kg/kg	
$w_{i,D}$	Ice mixing ratio after the deposition substep of the phase changes (see figure 1)	kg/kg	
$w_{i,t}$	Ice mixing ratio threshold for precipitation	kg/kg	4.5×10^{-4}
w_L	Liquid water mixing ratio of the parcel	kg/kg	
w_v	Water vapour mixing ratio of the parcel	kg/kg	
w_v'	Water vapour mixing ratio of the environment.	kg/kg	
w_{vs}	Saturated water vapour mixing ratio	kg/kg	
$w_{L,t}$	Liquid water mixing ratio threshold for precipitation	kg/kg	4.5×10^{-4}
z	Parcel elevation	m	
γ	Induced extra mass (environmental air) inertia coefficient		
Δt	Time step for numerical integration	s	0.1
Δz	Vertical step for initial parcel pressure integration	m	0.1
ϵ	Molar mass ratio water and dry air		0.622
μ	Dynamical entrainment rate		see equation 24
μ_0	Start value entrainment rate (if $C_{inv} = 0$)		
ρ	Density of the air parcel	$\mathrm{kg/m^3}$	

ρ'	Density of the environment	kg/m^3	
ρ_d	Density of dry air (air parcel excluding water vapour)	kg/m^3	
ρ_i	Density of cloud ice	kg/m^3	
ρ_v	Density of water vapour in air	kg/m^3	
τ_c	Condensation time scale	s	5
τ_e	Evaporation time scale	s	5
τ_f	Freezing time scale	s	0
τ_{pc}	Cold precipitation time scale	s	720
τ_{pw}	Warm precipitation time scale	s	5400
χ	Diffusivity of water vapour	m^2/s	$\frac{2.21}{p}$
ϕ	Vector containing state variables (see 2.1): T, w_v, w_L, w_i, m, p	see above list	

6 Appendix B: python code of cloud parcel model

```
# -*- coding: utf-8 -*-
"""
Created on Fri Oct 05 08:12:20 2018

@author: Edward and Chenxi
"""
import numpy as np
import matplotlib.pyplot as pl
import matplotlib as mpl
mpl.rcParams.update({'font.size': 21})

#input of the model
#constants
g=9.81 #gravitational acceleration
cp=1005. #specific heat per kilogram of dry air
T0=273.15 #zero Celsius Kelvin reference temperature
Rv=461.5 #gas constant water vapor
Rd=287.05 #gas constant dry air
Lf = 3.35e5 #latent heat of fusion
es0=610.78 #reference saturation vapor pressure
T1=273.16 #tripel point of water
T2=235. #upper bound for removing water vapor into ice
es1=611.20 #saturation vapor pressure over ice at tripel point
epsilon=0.622 #molar mass ratio water and dry air
Ka = 2.4e-2 #Thermal conductivity of air
rhoi = 700. #density of ice cristal, kg/m3

#pseudoconstants
def chi(p): #diffusivity of water vapor
    return 2.21/p
def A(T): #see Rotstayn (2000)
    return Ls(T)/Ka/T*(Ls(T)/(Rv*T)-1)
def Lv(T):#latent heat of vaporization water
    return (2.501 - 2.361e-3*(T-T0))*1e6
def Ls(T): #latent heat of sublimation water
    return Lf+ Lv(T)

#time space
tend=18000. #end of the simulation, s; 5 hours
dt=0.1 #time step, s
t1=np.linspace(0.0,tend,int(tend/dt))
dz=0.1 #vertical step for pressure initiation of parcel

#initial parcel characterstics
Riniteq=4000. #initial CP radius
parcel_bottom=150. #initial condition
ntop=0.0 #for parcel top; if exactly spherical n=2.0
parcel_top=parcel_bottom+ntop*Riniteq
Tdis=0.4 #temperature disturbance = initial condition
wvdis=0.2e-3 #water vapor disturbance = initial condition
winit=0. #initial condition vertical velocity
```

```
#parameters
gamma=0.5 #induced relation with environmental air, inertial
#mu=0.9e-4 #entrainment of air: R.A. Anthes (1977) gives 0.183/radius as its value
tau_cond = 5. #time scale for condensation, s
tau_evap = 5. #time scale for evaporation, s
tau_warmpc = 90.*60 #time scale for the formation of warm precipitation, s, 1000 s in A
tau_coldpc = 12.*60 #time scale for the formation of cold precipitation, 700 s in ECMWF
C_evap=1400. #rate constant for evaporation
wLthres=4.5e-4 # threshold for precip based on ECMWF documentation; 5e-4 in Anthes (197
withres=wLthres #threshold for precip form from ice
Cconv = 10. #assumed constant for increased rate in deposition in convective clouds com
Cinvr=0.16
mu0=5e-5
#entrainment parameterization
def mu_calc(R):
    #this is based on reading in the Pruppacher & Klett, 2010, chapter 12
    return Cinvr/R+mu0 #to switch off if no entrainment
    #return 0.00 #to switch on if no entrainment

#profile drying constants experiment , 1.00 in any layer means no drying and zint is the
Cdry=np.array([1.00,1.00])
zint=2000.0
i=0

#%%
#read background data from 20090526_00z_De_Bilt
fn='20100826_12z_Essen_mod.txt'
f=open(fn,'r')
p_d = np.array([])
z = np.array([])
T = np.array([])
wv = np.array([])
for line in f:
    line=line.split(';')
    p_d = np.append(p_d, float(line[1])*100.) #read pressure and convert to Pa
    z = np.append(z, float(line[2])) #read height in meters
    if z[-1] > zint:
        i=1
    T = np.append(T, float(line[3])+T0) #read temperature and convert to Kelvin
    wv = np.append(wv, Cdry[i]*float(line[6])/1000.) #read water vapor mixing ratio and
f.close()

#%%
#arrays for data in the environment and in the parcel, p:parcel env:environment
sat = np.zeros(len(t1))
zp = np.zeros(len(t1))
Tp = np.zeros(len(t1))
w = np.zeros(len(t1))
wvp = np.zeros(len(t1))
wvenv = np.zeros(len(t1))
p = np.zeros(len(t1))
Tenv = np.zeros(len(t1))
wL = np.zeros(len(t1))
wi = np.zeros(len(t1))
```

```
total_prec = np.zeros(len(t1))
sat = np.zeros(len(t1))
C = np.zeros(len(t1))
E = np.zeros(len(t1))
total_water=np.zeros((len(t1)))
Rp = np.zeros(len(t1))
mup = np.zeros(len(t1))
Mp = np.zeros(len(t1))

#%% envirmental profiles used
#interpolate T and wv profiles, linear interpolation y=a*x+b where a = d/dz of the resp
def find_nearest(array, value):
    array = np.asarray(array)
    idx = (np.abs(array - value)).argmin()
    return idx

def Tenvcalc(h):
    if h<=z[0]:
        Tenv = T[0]
    elif h>=z[-1]:
        Tenv = T[-1]
    else:
        i = find_nearest(z,h)
        if h == z[i]:
            Tenv = T[i]
        elif h > z[i]:
            dTdz=(T[i+1]-T[i])/(z[i+1]-z[i])
            Tenv = T[i]+(h-z[i])*dTdz
        else:
            dTdz=(T[i]-T[i-1])/(z[i]-z[i-1])
            Tenv = T[i]+(h-z[i])*dTdz
    return Tenv

def wvenvcalc(h):
    if h<=z[0]:
        wvenv = wv[0]
    elif h>=z[-1]:
        wvenv = wv[-1]
    else:
        i = find_nearest(z,h)
        if h == z[i]:
            wvenv = wv[i]
        elif h > z[i]:
            dwvdz=(wv[i+1]-wv[i])/(z[i+1]-z[i])
            wvenv = wv[i]+(h-z[i])*dwvdz
        else:
            dwvdz=(wv[i]-wv[i-1])/(z[i]-z[i-1])
            wvenv = wv[i]+(h-z[i])*dwvdz
    return wvenv

def p0(zloc,dz):
    #locate layer in which parcel is
    i=0
    while zloc > z[i+1]:
        i+=1

    #get properties at the base of this layer (lower bound, pressure & height) and layer
    zval=z[i]
    pref=p_d[i]

    while zval < zloc:
        #integrate hydrostatic equilibrium with EF and given dz
        zval+=0.5*dz
        Tloc=Tenvcalc(zval)
        wvloc=wvenvcalc(zval)
        Tvloc=Tvcalc(Tloc,wvloc)#*(1+(wvloc)/epsilon)/(1+wvloc) #from Aarnouts lecture 
        rho = pref/(Rd*Tvloc)
        dpdz=-rho*g
        pref+=dpdz*dz
        zval+=0.5*dz
    return pref

#%%
#initial conditions
def meanenvcalc(bottom,top,name):
    levels=np.linspace(bottom,top+1e-12,51)
    values=np.zeros(len(levels))
    for i in range(len(levels)):
        if name=='Tenv':
            values[i]=Tenvcalc(levels[i])
        elif name=='wvenv':
            values[i]=wvenvcalc(levels[i])
    return np.mean(values)

#put initial conditions provided above in arrays of result values
zp[0] = parcel_bottom+0.5*(parcel_top-parcel_bottom) #initial height of parcel, m
Tp[0] = meanenvcalc(parcel_bottom,parcel_top,'Tenv')+Tdis #initial temperature of parcel
w[0] = winit #initial velocity of parcel, m/s
wvp[0] = meanenvcalc(parcel_bottom,parcel_top,'wvenv')+wvdis #mixing ratio of water vapo
wL[0] = 0. #cloud content
total_prec[0] = 0.
p[0] = p0(zp[0],dz)

#%%
#differential equations
def dwdt(w,Tp,Tenv,wvp,wvenv,wL):
    Tvp=Tvcalc(Tp,wvp)
    Tvenv=Tvcalc(Tenv,wvenv)
    return 1./(1.+gamma)*(g*((Tvp-Tvenv)/Tvenv-wL-wi[i])-mu*abs(w)*w)

def dTpdt(w,Tp,Tenv,zp,C,E,dwidt):
    return -g*w/cp-mu*abs(w)*(Tp-Tenv)+Lv(Tp)/cp*(C-E)+dwidt*Lf/cp

def dwvpdt(w,wvp,wvenv,C,E):
    return -mu*(wvp-wvenv)*abs(w)-C+E

def dpdt(rho,w):
    return -rho*g*w
```

```
def dwLdt(w,C,E,wL):
    return C-E-mu*wL*abs(w)

def dmdt(mu,w,m):
    return mu*np.abs(w)*m

def func(phi,procarg,rho):#C,E, warm_precip,rho,Tenv,wvenv,t):#phi = [p,w,zp,Tp,wvp,wL]
    #extract values
    m,w,zp,Tp,wvp,wL=phi[0],phi[2],phi[3],phi[4],phi[5],phi[6]
    C,E,Tenv,wvenv,dwidt=procarg[0],procarg[1],procarg[2],procarg[3],procarg[4]

    #do the diff eqs
    dm=dmdt(mu,w,m)*dt
    dp=dpdt(rho,w)*dt
    dw=dwdt(w,Tp,Tenv,wvp,wvenv,wL)*dt
    dzp=w*dt
    dTp=dTpdt(w,Tp,Tenv,zp,C,E,dwidt)*dt
    dwvp=dwvpdt(w,wvp,wvenv,C,E)*dt
    dwL=dwLdt(w,C,E,wL)*dt
    return np.array([dm,dp,dw,dzp,dTp,dwvp,dwL])

#%% thermodynamic equilibria over water/ice surfaces
#equations from lecture notes by Van Delden (2017/2018) retrieved from http://www.staff.
def escalc(T):
    diffT=(1./T0-1./T)
    difflnes=Lv(T)/Rv*diffT
    lnes=difflnes+np.log(es0)
    es=np.exp(lnes)
    return es

def wvscalc(T,p):#calculation of water vapor saturation mixing ratio
    es=escalc(T)
    wvsat=epsilon*(es/(p-es))
    return wvsat

def esicalc(T,p):#calculation of water vapor saturation mixing ratio
    diffT=(1./T1-1./T)
    difflnesi=Ls(T)/Rv*diffT
    lnesi=difflnesi+np.log(es1)
    esi = np.exp(lnesi)
    return esi

def Tvcalc(T,wv):
    return T*(1+(wv)/epsilon)/(1+wv)

#%%
#processes: phase changes
def condensation(wv,wvs):
    if wv > wvs:
        return (wv-wvs)*(1-np.exp(-dt/tau_cond))/dt
    else:
        return 0.00
```

```
def evaporation(wv,wvs,wL):
    if wvs > wv and wL>0:
        return C_evap*wL*(wvs-wv)*((1-np.exp(-dt/tau_evap)))/dt
    else:
        return 0.00

#deposition of cloud water to solid phase: Rotstayn et al 2000, multiplied by Cconv
def B(T,p):
    return Rv*T*chi(p)*esicalc(T,p)
def Ni(T,p):
    return 1e3*np.exp(12.96*(escalc(T)-esicalc(T,p))/esicalc(T,p)-0.639)
def cvd(T,p,rho):
    return Cconv*7.8*(((Ni(T,p)/rho)**(2./3))*(escalc(T)-esicalc(T,p)))/(rhoi**(1./3)*(A
def Wi_depmeltfreez(T,p,rho,wL,dt):
    if T > T2 and T < T0:
        result=(2./3*cvd(T,p,rho)*dt+wi[i]**(2./3))**(3./2)
        if result < wL:
            return result
        else:
            return wi[i]+wL
    elif T < T2:
        return wi[i]+wL
    else:
        return 0.00

#%% precipitation processes
#warm precipitation (mainly autoconversion simulation)
def warm_precip(wL,Tp):
    if wL > wLthres:
        return (wL-wLthres)*(1-np.exp(-dt/tau_warmpc))
    else:
        return 0.0

#cold precipitation
def cold_precip(wL,wi):
    result1=(wi-withres)*(1-np.exp(-dt/tau_coldpc))
    if wi > withres:
        return result1
    else:
        return 0.00

#%%Integration procedure
t=t1[0]
Tenv[0] = Tenvcalc(zp[0])
wvenv[0] = wvenvcalc(zp[0])
sat[0] = wvp[0]/wvscalc(Tp[0],p[0])
C[0] = condensation(wvp[0],wvscalc(Tp[0],p[0]))
E[0] = evaporation(wvp[0],wvscalc(Tp[0],p[0]),0)
Rp[0] = Riniteq
mup[0] = mu_calc(Rp[0])
Tv = Tvcalc(Tp[0],wvp[0])
rho = p[0]/(Rd*Tv) #ideal gas law
Mp[0] = (4./3*Rp[0]**3)*rho
dwidt=0.
```

```
for i in range(len(t1)-1):
    #do the gass law and hydrostatic equilibrium to calculate pressure and saturation
    Tv = Tvcalc(Tp[i],wvp[i])
    rho = p[i]/(Rd*Tv) #ideal gas law
    Rp[i]=(3./4*Mp[i]/rho)**(1./3)
    mu=mu_calc(Rp[i])
    mup[i]=mu
    #Runge- Kutta numerical scheme
    processargs=np.array([C[i],E[i],Tenv[i],wvenv[i],dwidt])
    phi=np.array([Mp[i],p[i],w[i],zp[i],Tp[i],wvp[i],wL[i]])
    k1,k2,k3,k4=np.zeros(7),np.zeros(7),np.zeros(7),np.zeros(7)
    k1[:]=func(phi, processargs,rho)
    k2[:]=func((phi+0.5*k1), processargs,rho)
    k3[:]=func((phi+0.5*k2), processargs,rho)
    k4[:]=func((phi+k3), processargs,rho)

    #update values and save them in resulting array that includes time
    phi=phi+np.array((1./6)*(k1+2*k2+2*k3+k4),dtype='float64')
    t=t1[i+1]
    Mp[i+1]=phi[0]
    p[i+1]=phi[1]
    w[i+1]=phi[2]
    zp[i+1]=phi[3]
    Tp[i+1]=phi[4]
    wvp[i+1]=phi[5]
    wL[i+1]=phi[6]

    #update parcel environment
    Tenv[i+1] = Tenvcalc(zp[i+1])
    wvenv[i+1] = wvenvcalc(zp[i+1])

    #calculate saturation values
    wvs = wvscalc(Tp[i+1],p[i+1]) #water vapor saturation mixing ratio
    sat[i+1] = wvp[i+1]/wvs

    #then do condencsation, evaporation, freezing, melting, deposition/Findeisen-Wegene
    C[i+1]=condensation(wvp[i+1],wvs)
    E[i+1]=evaporation(wvp[i+1],wvs,wL[i+1])
    wi[(i+1)]=Wi_depmeltfreez(Tp[i+1],p[i+1],rho,wL[i+1],dt)
    dwidt=(wi[i+1]-wi[i])/dt
    dwi=(wi[i+1]-wi[i])
    wL[i+1]=wL[i+1]-dwi

    #precipitation process of the clouds and remove the cold precip from ice parcels
    warm_prec=warm_precip(wL[i+1],Tp[i+1])
    cold_prec=cold_precip(wL[i+1],wi[i+1])
    wi[i+1]=wi[i+1]-cold_prec
    wL[i+1]=wL[i+1]-warm_prec
    total_prec[i+1]=total_prec[i]+warm_prec+cold_prec #update total precipitation
#integrate precipitation and divide by areal extent
total_prec_mm=np.round(np.dot((total_prec[1:]-total_prec[:-1]),Mp[:-1])/(np.pi*Rp[-2]**
#%% visualization of results
#plot temerature profile
Tgamma=0.0050 #skew T visualzation constant
```

```
def Tdew(T,wv,p):
    #approximate dew point working above -50 degrees C, calculation retrieved from http:
    Tdew=np.ones(len(T))
    wvsloc=wvscalc((T+T0),p)

    #prevent run towards minus infinity for the log number
    wv[wv==0.]=1e-7

    #continue calculations
    relhum=wv[wv>0.]/wvsloc
    relhum=relhum*100.
    H=(np.log10(relhum)-2.)/0.4343+(17.62*T)/(243.12+T)
    Tdew=243.12*H/(17.62-H)
    return Tdew

xticks=np.array([])
z_plot=np.arange(0,18000,1000)
pl.figure(figsize=(12,8))
for i in range(183,310,5):
    pl.plot(i*np.ones(len(z_plot))+Tgamma*z_plot,z_plot,c=(0.6,0.6,0.6))
    if i > 260 and i < 300:
        xticks=np.append(xticks,np.array([i]))
pl.plot((Tp+Tgamma*zp),zp,c='r',label='Tp')
dew=(Tdew((T-T0),wv,p_d)+Tgamma*z+T0)
pl.plot(dew,z,c='b',label='Tdew',ls='--')
pl.plot((T+Tgamma*z),z,c='g',label='Tenv')
pl.title(fn[:-4])
pl.xlim(260,300)
pl.xticks(xticks,(xticks-273))
pl.legend(loc=1)
pl.ylim(0,16000)
pl.xlabel('Temperature (degrees Celsius)')
pl.ylabel('Height (m)')
pl.show()

#rain event evolution
pl.figure(figsize=(12,8))
pl.title(fn[:-4]+' precipitation produced by CPM')
pl.plot(t1,total_prec)
pl.xlabel('Time (s)')
pl.ylabel('Cumulative precipitation mixing ratio (g/g)')
pl.grid()
maxaxis=np.round(0.5+1000*total_prec[-1]*1.2,0)/1000.
pl.ylim(0,maxaxis)
pl.xlim(0,np.max(t1))
pl.text(0,-.16*maxaxis,'Areal mean total precipitation: '+str(total_prec_mm)+' mm')
pl.show()

#cloud composition as function of temperature
pl.figure(figsize=(12,8))
pl.plot(wL,Tp,label='Cloud liquid water mixing ratio')
pl.plot(wi,Tp,label='Cloud ice mixing ratio')
pl.plot(np.ones(len(Tp))*wLthres,Tp,ls='--',c='b',label='Warm precipitation threshold')
pl.plot(np.ones(len(Tp))*withres,Tp,ls=':',c='r',label='Cold precipitation threshold')
pl.legend(loc=1)
pl.title('Cloud content and temperature')
pl.xlabel('Mixing ratio (g/g)')
pl.ylabel('Temperature (K)')
pl.xlim(0,np.max(wv))
pl.ylim(np.min(np.min(Tp)),np.max(Tp))
pl.grid()
pl.show()

#cloud composition as function of time
pl.figure(figsize=(12,8))
pl.plot(t1,wL,label='Cloud liquid water mixing ratio')
pl.plot(t1,wi,label='Cloud ice mixing ratio')
pl.plot(t1,wi+wL,label='Ice + liquid water mixing ratio',c=(0.4,0.4,0.4))
pl.plot(t1,np.ones(len(Tp))*wLthres,ls='--',c='b',label='Warm precipitation threshold')
pl.plot(t1,np.ones(len(Tp))*withres,ls=':',c='r',label='Cold precipitation threshold')
pl.legend(loc=1)
pl.title('Cloud content and temperature')
pl.ylabel('Mixing ratio (g/g)')
pl.xlabel('time (s)')
pl.ylim(0,np.max(wv))
pl.xlim(0,np.max(t1))
pl.grid()
pl.show()

#velocity of the parcel
pl.figure(figsize=(12,8))
pl.title('Vertical velocity evolution')
pl.plot(t1,w)
pl.xlabel('time (s)')
pl.ylabel('w (m/s)')
pl.xlim(0,np.max(t1))
maxvel=np.round(1.1*np.max(np.abs(w))+0.5)
pl.ylim(-maxvel,maxvel)
pl.text(0,-1.36*maxvel,r'Effective maximum kinetic energy: '+str(np.round(0.5*np.max(w**
pl.grid()
pl.show()
```

References

[1] Atmospheric Soundings. Accessed October 20, 2018. http://weather.uwyo.edu/upperair/sounding.html.
[2] Pruppacher, Hans R., and James D. Klett. Microphysics of Clouds and Precipitation. Dordrecht: Springer, 2011.
[3] Wallace, John M., and Peter Victor. Hobbs. Atmospheric Science: An Introductory Survey. Amsterdam: Elsevier Acad. Press, 2011.
[4] Rotstayn, Leon D., Brian F. Ryan, and Jack J. Katzfey. "A Scheme for Calculation of the Liquid Fraction in Mixed-Phase Stratiform Clouds in Large-Scale Models." Monthly Weather Review 128, no. 4 (2000): 1070-088. doi:10.1175/1520-0493(2000)1282.0.co;2.
[5] Liu, Yangang, and Peter H. Daum. "Parameterization of the Autoconversion Process. Part I: Analytical Formulation of the Kessler-Type Parameterizations" Journal Of The Atmospheric Sciences 61, no. 13 (2004): 1539-1548. doi: 10.1175/1520-0469(2004)061¡1539:POTAPI¿2.0.co;2.
[6] Beheng, Klaus D. "A parameterization of warm cloud microphysical conversion processes" Atmospheric Research 33 no. 1-4 (1994): 193-206. doi: 10.1016/0169-8095(94)90020-5.
[7] Kessler, Edwin. "On the Distribution and Continuity of Water Substance in Atmospheric Circulations." On the Distribution and Continuity of Water Substance in Atmospheric Circulations, 1969, 1-84. doi:10.1007/978-1-935704-36-2.
[8] Anthes, Richard A. "A Cumulus Parameterization Scheme Utilizing a One-Dimensional Cloud Model." Monthly Weather Review 105, no. 3 (1977): 270-86. doi:10.1175/1520-0493(1977)1052.0.co;2.