One-dimensional turbulence (ODT) model with cloud microphysics

Documentation v1

Jan 2024

by
Mani Rajagopal

1. Introduction

This document is intended to help run the One-Dimensional Turbulence (ODT) model with cloud microphysics to simulate cloudy Rayleigh-Bernard convection in the Pi cloud chamber at the Michigan Technological University (MTU). Section 1 will provide a brief introduction to the Pi chamber, ODT model, and microphysical processes newly implemented. Section two will give an overview of the program structure of the ODT model and program files. Commands to compile and run the model are provided in Section 3. Input parameters and output files are described in Sections 4 and 5, respectively. Finally, Section 6 list the python program files to read and plot the output files.

1.1 The Pi convection cloud chamber

The Pi convection cloud chamber produces clouds in a laboratory setup and has been instrumental in understanding the effect of turbulence and aerosol on the cloud's microphysical properties. The clouds in the chamber are produced through Rayleigh-Bernard convection (RBC) when the warm air from the bottom plate mixes with the cold air from the top plate. Such clouds are called mixing clouds, as opposed to clouds formed from the adiabatic cooling of a rising air parcel.

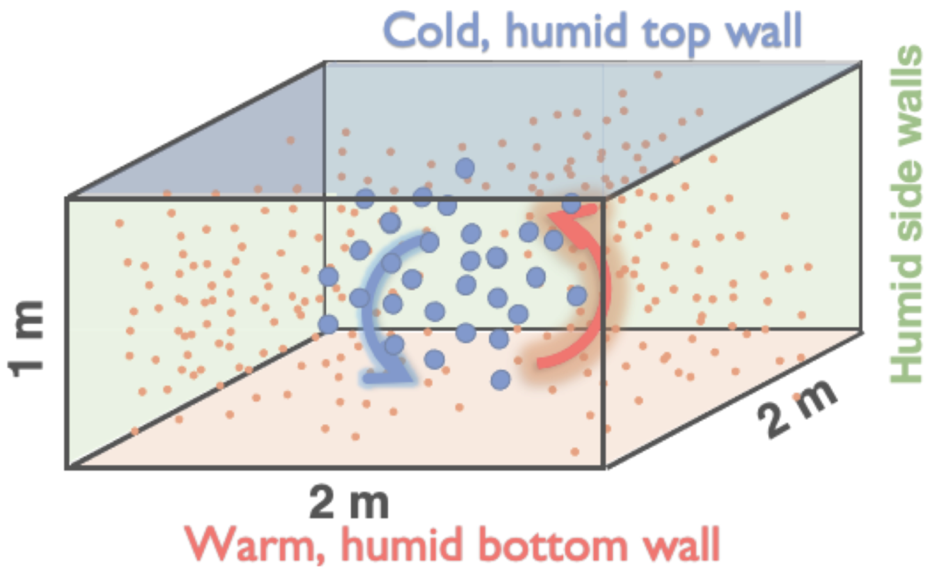

Fig.1. Schematic of cloudy Rayleigh-Bernard Convection in the Pi Cloud Chamber at the Michigan Tech University. Blue dots represent the cloud droplets, and orange dots represent the aerosol particles (From Steve Krueger)

The Pi cloud chamber has bottom and top plates with dimensions of 2 meters x 2 meters. Both plates are temperature-controlled and set at saturated conditions. The chamber is one meter tall and has various inlets, outlets, portholes, and instrumentation. The sidewalls are also temperature-controlled, but the saturation values of the sidewall due to the droplet condensation and sidewall fluxes are currently unknown. This is an active area of research to understand the sidewall fluxes and their effect on the supersaturation fluctuations in the chamber's core. The Pi chamber can either be parallelepiped with a volume of 4 m^3 or have a cylinder insert to produce a cylindrical volume of pi m^3 (Fig. 2) Hence referred to as a pi chamber. For more details on Pi Chamber, please refer to Chang et al. (2016)

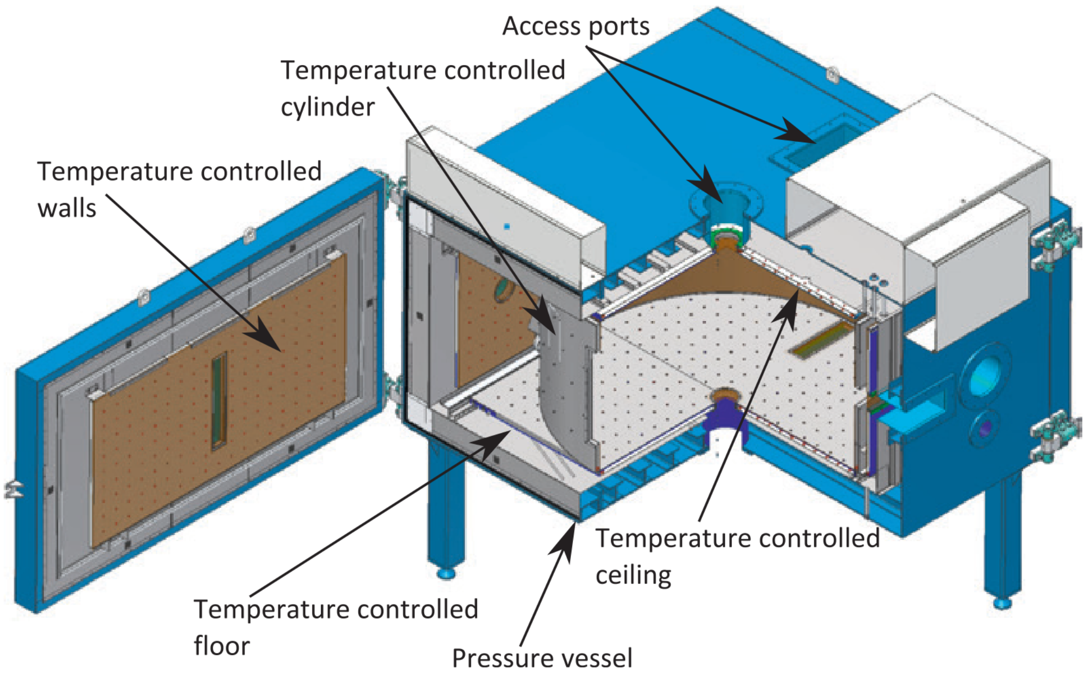
Access ports
Temperature controlled cylinder
Temperature controlled walls
Temperature controlled ceiling
Temperature controlled floor
Pressure vessel

Fig. 2. Three-dimensional view of the Pi chamber with temperature-controlled walls, cylindrical insert, and access ports (Chan et al. 2016)

1.2 Numerical simulation of pi chamber

The Pi chamber has been crucial in understanding the effects of aerosol and turbulence on the droplet size distribution. However, the measurements of supersaturation, turbulence, and particle size distribution are limited in space. Numerical models complement laboratory observations to understand the processes in the chamber and provide a high spatial and temporal sampling of variables, including supersaturation. Many groups have carried out direct numerical simulations (DNS) of the pi chamber. The DNS resolves turbulence but does not resolve the microphysics; hence, particle microphysics is parameterized. The DNS runs are also computationally expensive and limit the number of experiments that can be performed. One-dimensional models are computationally inexpensive and helpful for exploring the parameter space (boundary conditions and various aerosol properties).

1.3 Dry One-dimensional turbulence model (ODT)

Though many one-dimensional models exist, the one-dimensional turbulence (ODT) model has been demonstrated to simulate the dry Rayleigh Bernard convection (RBC) for very high Rayleigh numbers. Wunch and Kerstein (2005) developed the ODT model and tuned model parameters to match the laboratory observations of the Nusselt number, Sherwood number, and scaling of these non-dimensional quantities in high Rayleigh number experiments.

The ODT model is a stochastic turbulence model that randomly picks an eddy size from an eddy size distribution determined from the RBC boundary conditions. The effect of an eddy is implemented through triplet mapping and numerical diffusion. Triplet mapping rearranges grid cells within an eddy of length, l, such that the scalar field has a sinusoidal variation. This increases the gradients between adjacent cells and increases the scalar diffusion.

For the RBC convection, the steady state variability is along the vertical direction. Therefore, the model is set up in the vertical direction. Fig.3. shows how two eddies, large and small, modify the temperature field. Triplet mapping rearranges 1-D array cells to produce sinusoidal variations but doesn't change the scalar values. Therefore, all the statistical moments within the domain are conserved. The intermediate diffusion steps between triplet mapping change the scalar values. Together, eddy mapping and diffusion mimic the effect of turbulence in mixing any variability introduced into the domain. The triplet mapping increases the local gradient, and diffusion reduces it.

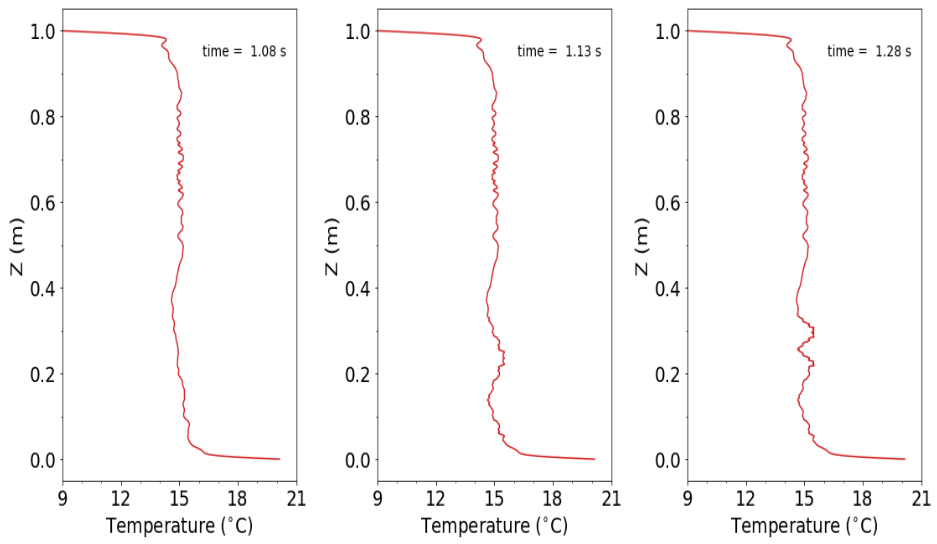

Fig. 3. The effect of eddies is implemented through triplet remapping that rearranges the grid cell to produce sinusoidal variations in a scalar field.

1.4 Moist One-dimensional turbulence model (ODT)

Chandarkar et al. (2019) modified the dry ODT model to include water vapor. The moist RBC convection also takes into account the effect of water vapor on buoyancy. This model can be used to study the mixing process and supersaturation PDF for given boundary conditions without aerosol or other microphysical processes.

1.5 One-dimensional turbulence model (ODT) with microphysics

The moist ODT model has water vapor but no aerosol or cloud droplets. Mani Rajagopal added the various microphysical processes such as particle injection, condensational growth, gravitational settling, and particle displacement in eddies. The new model allows for various particles, such as dry aerosol, haze droplets (interstitial aerosols), and cloud droplets, to be injected into the domain. Though dry aerosol can be injected, they quickly become haze droplets, even in sub-saturated conditions. Based on the trajectory that a particle takes, it is exposed to different local supersaturations, which might make it grow to a size greater than the critical radius (depending on dry aerosol radius) to become a cloud droplet. If the particle radius is less than the critical radius, it is a haze droplet or interstitial aerosol. In this ODT version, the microphysical processes are implemented particle by particle in a Lagrangian framework as opposed to being parameterized in the DNS. Each particle is tracked separately as it moves with eddies, grows, evaporates according to its local supersaturation, and falls due to gravity.

The implementation of four microphysical processes, 1) particle injection, 2) particle movement by eddies, 3) condensational growth, and 4) gravitational settling, is briefly described here. In a real chamber, the particles are injected at a point source. In this ODT model, the particle injection is mimicked by placing an injection blob (configurable volume) within the domain. This implementation assumes injected particles are quickly mixed within a certain volume (injection volume) around the injection point. The particles are randomly placed within this injection volume. For the second process of moving particles in eddies, the triplet mapping is extended to the particles. When the triplet mapping rearranges grid cells with an eddy length, l, to mimic the eddy effects, particles present in those grid cells are moved with it by updating their new position. In this process, a particle's local supersaturation will be the same since it is in the same grid cell. The third process, condensational growth, is implemented through subprograms from the Explicit Mixing Parcel Model (EMPM). Each particle is run through this condensational growth code to determine if it grows or evaporates in its local environment. If a droplet grows, it increases the temperature and decreases the water vapor of its grid cell. The vice versa happens when the droplet evaporates. The final process of gravitational settling is based on the stokes law to determine the terminal velocity of a particle based on its size. A particle's new position is calculated using this terminal velocity and the model time step. The gravitational settling causes a particle to fall through from one grid cell to the next exposing it to a new environment.
There are three ways that a particle experiences a change in its environment: 1) the diffusion of temperature and water vapor, 2) gravitational settling into a grid cell below, and 3) If there is more than one particle in the grid cell, the particle that is first in the list (injected first) will be grow or evaporate changing the local supersaturation. Ideally, all particles should grow/evaporate at the same time, but the implementation here is sequential.

2. ODT program structure

In the previous section, various implementations of ODT (dry, moist, and microphysics) are discussed briefly. The new version of ODT with microphysical processes brings together implementation from two numerical models – ODT and EMPM. The turbulence processes are from ODT, and the microphysical processes are similar to EMPM. The microphysical processes have some modifications compared to EMPM, particularly the gravitational settling and fallout of a particle when it reaches a bottom boundary. In this version of ODT, some of the program and subprogram files from moist ODT and EMPM are used with little or no modifications. The microphysical processes are implemented as a separate module to minimize changes to ODT program files. This made the code organization complex.

2.1 Dry and moist ODT programs

The dry and moist ODT programs are similar. They have two program files, Labinit.f, and LabExp.f. The program file, Labinit.f, has input parameters for ODT described in Section 4.

Program files	Description
`Labinit.f`	Set fluid properties (viscosity, thermal diffusivities, vapor diffusivity), boundary conditions (height, temperature, supersaturation for top and bottom plates), and model tuning parameters.
`LabExp.f`	Implementation of ODT model

Unlike many other numerical models that take input through a name list, ODT model parameters are provided through a Fortran program file. These dimensional model parameters are converted to non-dimensional model parameters and written to a file LabExppar.dat, along with initial profiles for temperature, water vapor, and wind. The default initial profiles for temperature and water vapor vary linearly from bottom to top boundaries.

2.2 ODT with microphysics

The new version of ODT with microphysics has the following files from the EMPM model in addition to the files from the ODT model listed in the previous section. These files have subroutines or functions that implement the numerical integration of condensational growth equations.

Program files	Description
`array.f90`	Parameters used in condensational growth program
`const.f90`	Physical constants for the fluid and others
`rand1.f90`	Create a random value between 0 and 1 that follow uniform distribution
`rkqs.f90`	Used in numerical integration
`fcknb.f90`	Determines rate of change of particle’s radius, temperature and humidity with solute and curvature effects
`fcnkb0.f90`	Determines rate of change of particles’radius, temperature and humidity without solute and curvature effects
`rkck.f90`	Used in numerical integration
`ew.f90`	Calculate the vapor pressure at a given temperature
`odeint.f90`	Integrate ordinary differential equations for condensational growth. Uses all the above subprograms
`size_dis.f90`	Produce gamma particle size distribution

The “**microphysics.f90,” or the microphysics module**, implements all the microphysical processes described in section 1.5 and uses the “odeint.f90” for condensational growth. The microphysics module has global variables (for particle properties) and subprograms that implement the microphysical processes or compute statistics. This module is well-commented, and microphysical processes are broken into small manageable subprograms.

The “**LabExp_mphy.f90**” is a modified version of the ODT model with minimal code changes. After each diffusion step, three microphysical processes (injection, condensational growth, and gravitational settling) are called through the microphysics module. After each triplet mapping step, another microphysics module’s function is called to move particles to new positions.

3. Compiling and running the program

All the program files are located in the following folder and need to be copied to your working folder.
“`/uufs/chpc.utah.edu/common/home/u1147793/bin/fortran/odt_kt/102_mrbc_mphy/`”.

The shell script “**compile_commands.sh**” in this folder has the commands used to compile, configure, and run the ODT model. They are described below as steps, a through d

a)Compile and run model input

```
gfortran -O3 Labinit_mphy.f  -o Labinit_mphy.exe
./Labinit_mphy.exe
```

The ODT model input parameters are in the Fortran file “Labinit_mphy.f”. Therefore, any changes to parameters need to be changed in the Fortran file, compiled, and run to generate the input file and initial profiles. The ”Labinit_mphy.f” is modified to write these files to the “./input/default” folder to keep them organized.

b) Run ODT to reach turbulence steady-state (spin-up)

The initial profiles that ODT have temperature and water vapor increase linearly from the bottom to top plates. The ambient wind profiles are set to zero for the pi chamber. The model takes nearly 40 seconds to reach a turbulence steady state. The aerosol injection should start only after the turbulence reaches a steady state. The non-dimensional simulation time is set in the Labinit_mphy.f file. The default value of 0.001 in non-dim time translates to ~70 seconds in dimensional time, when scaled by time for viscous diffusion across the domain length of one meter.

In many experiments, the same boundary conditions for turbulence are chosen, but only microphysical parameters are changed. To save turbulence spin-up time, the steady-state profiles are saved, and all microphysical experiments can start from the same steady state. The following commands are used to compile the ODT model and run it. The modified ODT model executable (LabExp_mphy.exe) takes four arguments, 1) an subfolder inside “./input” directory which has input parameters and initial files, 2) a microphysics name list file located in the input subfolder, 3) a subfolder in the “./output” directory to store the output file, and 4) a random seed. The random seed will be used to generate random numbers that will randomly pick eddy size, eddy position, and particle position when injected. A different realization of the simulation can be obtained by choosing a different random seed for the same input parameters (i.e. same boundary conditions) and initial profiles.

```
#compile ODT with microphysics
gfortran -O3 -g -fbacktrace array.f90 const.f90 rand1.f90 rkqs.f90 fcnkb.f90
fcnkb0.f90 rkck.f90 ew.f90 odeint.f90 size_dis.f90 microphysics.f90
LabExp_mphy.f90 -o LabExp_mphy.exe

#spinup to steady state
nohup ./LabExp_mphy.exe default nml_mpy steady_state 001 &> steady.log &
```

After the model reaches a steady state, the profiles are copied to the input directory. Unlike the ODT model parameters, the input parameters for microphysical processes are stored in a name list. The default name list "`nml_mphy`" in the parent folder need to be copied to the subfolder in the input directory.

```
#copy steady state profile and input params
mkdir -p input/steady_state
cp  output/steady_state/001/*   input/steady_state/
cp  nml_mphy input/steady_state/
```

c) Turn on microphysics and edit the simulation time and

The simulation in the previous steps did not include microphysical processes but only turbulence. The input parameter file "LabExppar.dat" generated from Labinit.f under folder "input/steady_state" has to be changed to turn on microphysics. The last line in the file should be changed from "0" to "1" to include microphysical processes. Otherwise, the ODT model will run as moist RBC without any microphysics. In the same file, line 4 column 2 has the simulation time in non-dimensional format. This should be changed to the desired value. For example, The simulation time of one hour (3600 seconds) is approximately 0.052 in non-dimensional time and the simulation time of 70 seconds is approximately 0.001 in non-dimensional time.

d) run experiments using ODT with microphysics

To run experiments with microphysics, execute the following command. The modified ODT model takes four arguments as described in the section above. The following command will run the ODT model with microphysics based on files in subfolder steady_state under input directory, uses parameters from "nml_mphy" for microphysics, and stores the output in the subfolder "test" under output directory. This run uses a random seed of "001"

```
#run odt with microphysics
nohup ./LabExp_mphy.exe steady_state nml_mphy test 001 &> 001.log &
```

Sometimes, we need to run multiple realizations of the same ODT experiment. The shell script "odt_multi_realz.sh" will help run any number of realizations as needed only by changing the random seed from 1 to n. The variable "n_realization" in the shell script file specifies required number of realizations. It has a default value of five. The script may be executed as follows to create five realizations.

```
#run multiple realization of a ODT experiment
./odt_multi_realz.sh steady_state nml_mphy test &> multi_realz.log &
```

4. Input

Only important input parameters are listed in this section. For a complete list, please check the comments in the following files.

Config files	Description
`Labinit.f`	The ODT model uses a fortran file to configure input parameters instead of a name list. The parameters include fluid properties (viscosity, thermal diffusivities, vapor diffusivity), boundary conditions (height, temperature, supersaturation for top and bottom plates), and model tuning parameters. The file is located in the program directory. When compiled and executed, it creates a LabExppar.dat with non-dimensional model parameters in the “input/default” folder and initial profiles for temperature and water vapor that increases linearly with height
`nml_mph`	It is a name list file with configurable parameters for microphysical processes. This file should be located in the subfolder under input directory. The name of this subfolder should be passed as an argument to the ODT model executable (LabExp_mphy.exe). For example, the steady state input files and nml_mphy are present in “input/steady_state”

The following parameters are set in the Labinit.f

Parameter	Value	Description
`Tdif`	12	Temperature difference between bottom and top plate
`T_o`	15	Reference temperature in Celsius or mid value of bottom and top. T_bot = T_o + Tdif/2; T_top = T_o – Tdif/2
`H`	1	Height of the chamber
`p`	1e5	Pressure in the chamber. Assumed to be same everywhere within the chamber.
`N`	6000	Number of grid cells
`Lo`	25	1/3rd of smallest eddy size
`Lp`	100	1/3rd of highly probable eddy size
`Lm`	2000	1/3rd of largest eddy
`tmax`	0.52e-1	Simulation time in non-dimensional form; Simulation time in seconds is normalized by timescale for viscous diffusion through depth of the chamber. timescale for

		viscous diffusion, tscale_vis_diff = H^2 /viscosity; The value 0.52e-1 is approximately 3600 sec.
`has_microphy`	0	1 -> no microphysics; 1 -> run with microphysics

Currently, the bottom and top plates are set to be saturated. If they are sub-saturated, the parameters pv_b (for the bottom plate) and pv_t (for the top plate) should be set accordingly. The ODT model parameters (C^2, and ZC^2) have been tuned to match lab and DNS results and doesn't require change. The following table lists the important microphysical parameters set in the name list "nml_mphy"

Parameter	Value	Description
`width_dom`	0.001	Length in x and y direction
`area_frac`	1	Shrink or expand the horizontal area (x*y), and hence the volume. It is much easier to control the domain area or volume by a single factor. The area or volume can be reduced to half or doubled easily
`max_prtcl_dom`	10000	maximum particle in the domain; used for array allocation
`max_prtcl_gcell`	100	maximum particle in a grid cell; used for array allocation
`add_prtcl_mthd`	1	1 -> mono disperse particle size distribution 2 -> Gamma function-based particle size distribution 3 -> read particle size distribution from a file
`nbins_prtcl_dist`	49	number of bins in particle size distribution used in method 2 (gamma dis) and 3 (read from a file)
`r0_prtcl`	0.0630e-6	particle radius in meters
`lwc_prtcl_sum`	1.51731E-5	liquid water content in kg m^{-3}; used as parameter in gamma particle size distribution
`m0_aero`	2.212E-18	solute mass in kg; used in monodisperse method (1) and gamma size distribution method (2). Though particles in gamma size distribution are of different sizes, they have same aerosol mass or dry aerosol radius.
`dis_prtcl_file`	"dis_prtcl.dat"	The file with the particle size distribution
`inject_stime`	0.1d-3	Injection start time (in non-dimensional format)
`inject_etime`	1.0E12	Injection end time (in non-dimensional format); set to a very large value if injection ends only when simulation ends
`inject_onetime`	0	inject one time (1) or continuously (0)

`n_prtcl_uvol_onetime`	400e6	number of particles to be injected one time which happens instantaneously.
`n_prtcl_uvol_utime`	10e6	number of particles to be injected continuously in unit volume in unit time
`is_inj_blob_rnd`	0	Place the injection blob's center at a random location in the domain (1-> true 0-> false). This parameter is used in both one-time and continuous injection. If false, the injected blob will be placed at the location given by parameter "inject_at"
`inject_at`	0.5	if `is_inj_blob_rnd` = 1 then this parameter value is ignored else then inject blob is centered at this position. This value for position should be > 0 + inject_vol/2 and < 1-inject_vol/2; else error will be thrown. This parameter is used in both one-time and continuous injection
`inj_vol`	1.0	injection blob vol as fraction of domain volume used in both one-time and continuous injection.
`aero_type`	1	aerosol type 1-> NaCl, 2-> $(NH4)_2(SO4)$, 3-> (NH4)HSO4,
`has_curv_sol_eff`	1	include curvature and solute effects in the particle growth. 1 -> yes, 0 -> no
`r_min_instru`	1.5e-6	Minimum particle size that an instrument can measure. This parameter is used to generate particle statistics (mean, number concentration, size distribution) that an instrument would see.
`term_vel_factor`	1.0	The implementation assumes that particle follows stokes formula for terminal velocity but if the particles doesn't then this factor can be modified accordingly.

5. Output

The output files are written under a subfolder passed as an argument to the ODT model executable (LabExp.f). The output files can be grouped into four categories – evolution, profiles, PDF, and Lagrangian particle history. Files under the first three categories have a prefix of "evol", "prof", and "pdf", respectively. The particle history is stored under the sub-sub-folder "hist_prtcl." The output files from the original ODT model related to turbulence have a file extension of ".dat," whereas the microphysical processes-related output files have a file extension of "*.txt". The following output files are from the original ODT model.

Filename	Description
`EddyDiagram.dat`	Eddy size and location information. The file has three columns: non-dim time, non-dim eddy location, and non-dim eddy length
`eu.dat`	Kinetic energy profile from different wind components. Columns – non-dim height, u comp, v comp, w comp, and total
`LabExppar.dat`	The non-dim input parameter for ODT model executable. This file is generated from the Labinit.f; Copying input parameters to output is useful future reference since the input file can be changed for another experiment
`Nu.dat`	Nusselt number and Sherwood number; These values are computed from gradient of time averaged temperature and water vapor at bottom and top grid cell
`out_ODT.dat`	Average stats on eddy: number of eddies, average time between eddies, acceptance rates, etc.
`PDF_pv.dat`	PDF of water vapor values at five vertical levels through simulation time. Five levels are, N/2, N/4, N/8, 10th grid cell from bottom, and 25th grid cell from the bottom
`PDF_s.dat`	Same as above but for supersaturation
`PDF_T.dat`	Same as above but for temperature
`PL.dat`	
`pv.dat`	Profile of water vapor statistics. The value is calculated at each grid cell. The file has four columns: non-dim height, instantaneous water vapor value at the last time step (end of simulation), mean value over simulation time, variance over simulation time
`s.dat`	Same as above but for supersaturation
`T.dat`	Same as above but for temperature
`T_qv_covar.dat`	Covariance profile for temperature and water vapor. The covariance computed at each grid through simulation time
`U.dat, V.dat, W.dat`	Same as pv.dat but for u, v, and w velocity component
`Warnings.dat`	Error or warning messages

The output files related to microphysical processes have a ".txt" file extension. Each file has a header describing the columns in that file and their units.

Filename	Description
`evol_prtcl_prop.txt`	This file has the statistical properties derived from all particles **at each time step,** thereby providing the evolution. Some of the statistics are count, mean radius, min radius, max radius, and liquid water content. For a complete list check the file header.
`evol_cdplt_prop.txt`	The particles in the domain are classified as dry aerosol (aero), haze droplet(hdplt), and cloud droplet(cdplt). This file has the statistical properties of cloud droplets at each time step. The file has same columns/statistics as evol_prtcl_prop.txt
`evol_aero_prop.txt`	Same as `evol_cdplt_prop.txt` but for dry aerosols
`evol_hdplt_prop.txt`	Same as `evol_cdplt_prop.txt` but for haze droplets
`evol_instru_prop.txt`	Same as evol_cdplt_prop.txt but for particles that are detectable by an instrument. Minimum particle size that this instrument can detect is set in the name list (parameter name: r_min_instru)
`evol_fallout_prop.txt`	Statistical properties of particles that fallout at the bottom of the chamber
`evol_mphy_prop.txt`	Evolution of microphysical properties in terms of number concentration of particles injected, activated to cloud droplet, deactivated, fallout, and particle currently in the domain **at each time step,** thereby providing the evolution. see the file header for the complete list of properties
`evol_dom_scalar.txt`	Evolution of temperature, water vapor, and supersaturation. The statistical properties such as mean, min, max, and variance, is computed from values at all grid cells in the domain **at each time step,** thereby providing the evolution**.**
`evol_bulk_scalar.txt`	Same as above but statistical value is computed from grid cells in the bulk or core area (non-dim height of 0.2 to 0.8) of the domain. The height range for bulk region is hard coded in the program. To choose a different region, edit the program or this can be made as a parameter in the name list in the future code change.
`injection_prtcl.txt`	Injection information – non-dim time, and number of particles injected
`pdf_bulk_qv.txt`	PDF of water vapor from grid cells in the bulk region (height = 0.2 to 0.8) over a short time interval (~10 seconds). The time interval is stored separately in pdf_time.txt. After the file header, the first line represents the bin edges with n+1 columns. The lines two to end of files have time step weighted frequency at n bins. Note that these lines have one column less (n bins) than bin edges. The frequency at each bin is multiplied by the time step because ODT has unequal timesteps and needs to be weighted by respective timestep.

`pdf_time.txt`	Start and end of time interval (~10 seconds) when a PDF is computed. Excluding the header, the line 1 corresponds to PDF values in line 2 of pdf_*_*.txt. To obtain a PDF over certain time interval, say line 91 to 100, add the weighted frequency in each bin from rows say 91 to 100 in pdf_*_*.txt, then divide each bin by sum of non-dim time intervals from 90 to 99 in pdf_time.txt. Since the pdf_*_*.txt files have bin edges in the first line, the corresponding time interval in pdf_time.txt is staggered by a line
`pdf_bulk_ss.txt`	PDF of supersaturation in the bulk region during various time interval given in pdf_time.txt
`pdf_bulk_T.txt`	PDF of temperature in the bulk region during various time interval given in pdf_time.txt
`pdf_dom_qv.txt`	PDF of water vapor in the entire domain; The frequency is computed using all the grid cells in the domain during various time interval given in pdf_time.txt
`pdf_dom_ss.txt`	Same as above but for supersaturation
`pdf_dom_T.txt`	Same as above but for temperature
`pdf_radius_aero.txt`	PDF of dry aerosol radius in the domain during various time intervals given in pdf_time.txt
`pdf_radius_cdplt.txt`	PDF of cloud droplet radius in the domain during corresponding time interval in pdf_time.txt
`pdf_radius_hdplt.txt`	Same as "`pdf_radius_cdplt.txt`" but for haze droplet
`pdf_radius_instru.txt`	Same as "`pdf_radius_cdplt.txt`" but for instrument measured particles
`pdf_radius_prtcl.txt`	Same as "`pdf_radius_cdplt.txt`" but for all particles
`prof_cdplt_nsum.txt`	cloud droplets concentration varies with height. Creating a profile at each grid cell is computationally expensive. Also there are only few or no cloud droplets at each grid cell to obtain a reliable statistical value. Therefore, the grid cells are grouped into vertical bins. After the file header, the first line has the bin edges of vertical bins in non-dim z values. The bins are small close to bottom and top boundary whereas they are larger in the bulk. The lines 2 to end of file has number of cloud droplets in each vertical bin over a time interval. The respective time interval is stored in the prof_time.txt.
`prof_time.txt`	Start and end of time interval (~10 seconds) when a profile is computed. After the file header, the line 1 in this file corresponds to profile values in line 2 of files prof_*_*.txt.

	For example, to obtain a profile over certain time interval, say line 91 to 100, add the value in each bin from rows 91 to 100 in prof_*_*.txt, then divide each bin by sum of non-dim time intervals from 90 to 99 in prof_time.txt. Since the prof_*_*.txt files have bin edges in the first line, the corresponding time interval in prof_time.txt is staggered by a line
`prof_cdplt_rsum.txt`	Profile of sum of radius of cloud droplets in that vertical bin. To obtain mean radius at a vertical bin, divide each bin by cdplt_nsum. To obtain mean value over a time interval: add bin values in prof_cpldt_rsum over a time interval, add bin values in prof_cdplt_nsum over the same time interval, then divide rsum by nsum at respective bins. The numerator and denominator don't need to be divided by total time interval because they would cancel out.
`prof_cdplt_r2sum.txt`	Same as above but the statistic is sum of radius squared for cloud droplets in each vertical bin during various time intervals given in prof_time.txt. To obtain mean r-squared value it needs to be divided by nsum at each vertical bin
`prof_cdplt_r3sum.txt`	Same as above but the statistic is sum of radius cubed for cloud droplets in each vertical bin during various time intervals given in prof_time.txt. To obtain mean r-cubed value it needs to be divided by nsum at each vertical bin

`prof_hdplt_nsum.txt,`
`prof_hdplt_rsum.txt,`
`prof_hdplt_r2sum.txt,`
`prof_hdplt_r3sum.txt,`
	Same as above but for haze droplets

`prof_prtcl_nsum.txt,`
`prof_prtcl_rsum.txt,`
`prof_prtcl_r2sum.txt,`
`prof_prtcl_r3sum.txt,`
	Same as above but for all particles (dry aerosol, haze droplets, and cloud droplets

`prof_actvd_nsum.txt`	Used to compute profile of particles that activated to cloud droplets. A particle is considered activated if r_particle > critical radius (depends on aerosol mass/ the dry aerosol radius inside that particle). The profile is computed as described in `prof_time.txt`
`prof_deactvd_nsum.txt`	Used to compute profile of particles that are deactivated i.e., the r_particle < critical radius due to evaporation. The profile is computed as described in `prof_time.txt`
`prof_inj_nsum.txt`	Profile of injected particles. The profile is computed as described in `prof_time.txt`. It should match the injection

	parameters defined in the name list. Within the inject location and volume, the particle concentration should be equal.
`prof_qvsum.txt`	Used to compute the profile of water vapor at each grid cell over a short time interval (~10 seconds). Unlike particle profile, these values are computed at each grid cell. The profile is computed as described in `prof_time.txt`.
`prof_qv2sum.txt`	Same as above but for qv-squared
`prof_sssum.txt`	Same as "prof_qvsum.txt" but for supersaturation
`prof_ss2sum.txt`	Same as "prof_qvsum.txt" but for supersaturation-squared
`prof_Tsum.txt`	Same as "prof_qvsum.txt" but for temperature
`prof_T2sum.txt`	Same as "prof_qvsum.txt" but for temperature-squared.

The particle history is stored under the sub-sub-folder "hist_prtcl." The particle properties and its environment are sampled every second and stored in a buffer to reduce frequent writing to a file. The frequent file writing slows the model execution and takes longer to complete the simulation. Therefore, the particle histories are written to a file from the buffer every 30 seconds. Both the sampling time and write interval can be set in the name list (parameters' name: `rec_intvl_prtcl_hist, write_intvl_prtcl_hist`)

6. Python code to plot output

Python programs to plot output files are in the folder "/uufs/chpc.utah.edu/common/home/u1147793/bin/python/odt_kt/". These are jupyter notebook files; so you need a jupyter notebook server up and running. Copy these files to your jupyter notebook directory to read and run them. The following table briefly describes Python program files.

Python files	Description
`plot_comp_evol_microphy.ipynb`	For various injection rates or other experiments, compare the evolution of particle statistics: particle conc., cloud droplet concentration, haze droplet concentration, mean radius, total cloud liquid water water, and domain mean supersaturation and variance.
`plot_comp_prtcl_prof.ipynb`	For various injection rates or other experiments, compare particle profile (number concentration, and mean radius, mean radius-squared) at steady state.
`plot_comp_psd.ipynb`	For various injection rates or other experiments, compare particle size distribution at steady state.
`plot_comp_scalar_prof.ipynb`	For various injection rates or other experiments, compare the profile of temperature, water vapor, and supersaturation at the steady state.

`plot_comp_ss_pdf.ipynb`	For various injection rates or other experiments, compare the PDF of supersaturation values at the steady state.
`plot_comp_zpos_scalar_pdf.ipynb`	For various injection rates or other experiments, compare PDF of temperature, water vapor, and supersaturation at five vertical levels. Five levels are, N/2, N/4, N/8, 10th grid cell, and 25th grid cell
`plot_eddy_diagram.ipynb`	Plot eddy size distribution, location, and inter-eddy time
`plot_evol_mrbc_wo_mphy.ipynb`	Plot evolution of temperature variance for moist RBC (ODT without microphysics)
`plot_evol_scalar.ipynb`	Plot evolution of temperature, water vapor, and supersaturation for ODT with microphysics
`plot_prof_mrbc_wo_mphy.ipynb`	Plot profile of temperature, water vapor, and supersaturation for moist RBC (ODT without microphysics).

There are two steady states: 1) steady state in turbulence and 2) steady state in microphysics. The steady state in turbulence implies that for a given boundary condition and injection rate, the domain variance for temperature, water vapor, and supersaturation becomes nearly constant. Without microphysics, the model reaches a steady state in turbulence with a spin-up time of 30-40 seconds. The injection of particles and condensation affects the buoyancy and turbulence in the chamber; it takes a few minutes (Figs. 4c and 4f) after injection to reach a steady state in turbulence. In contrast, the steady state in microphysics occurs when the number of particles injected is balanced by a number of cloud droplets that fall out. This would result in a nearly constant number of cloud droplets in the domain (Fig 4b.). The steady state for microphysics depends on the injection rate because, for a high injection rate, the mean cloud droplet radius is small and takes longer to fall out. This leads to high cloud droplet concentration in the domain while also taking long time to reach the steady state(Fig. 4b).

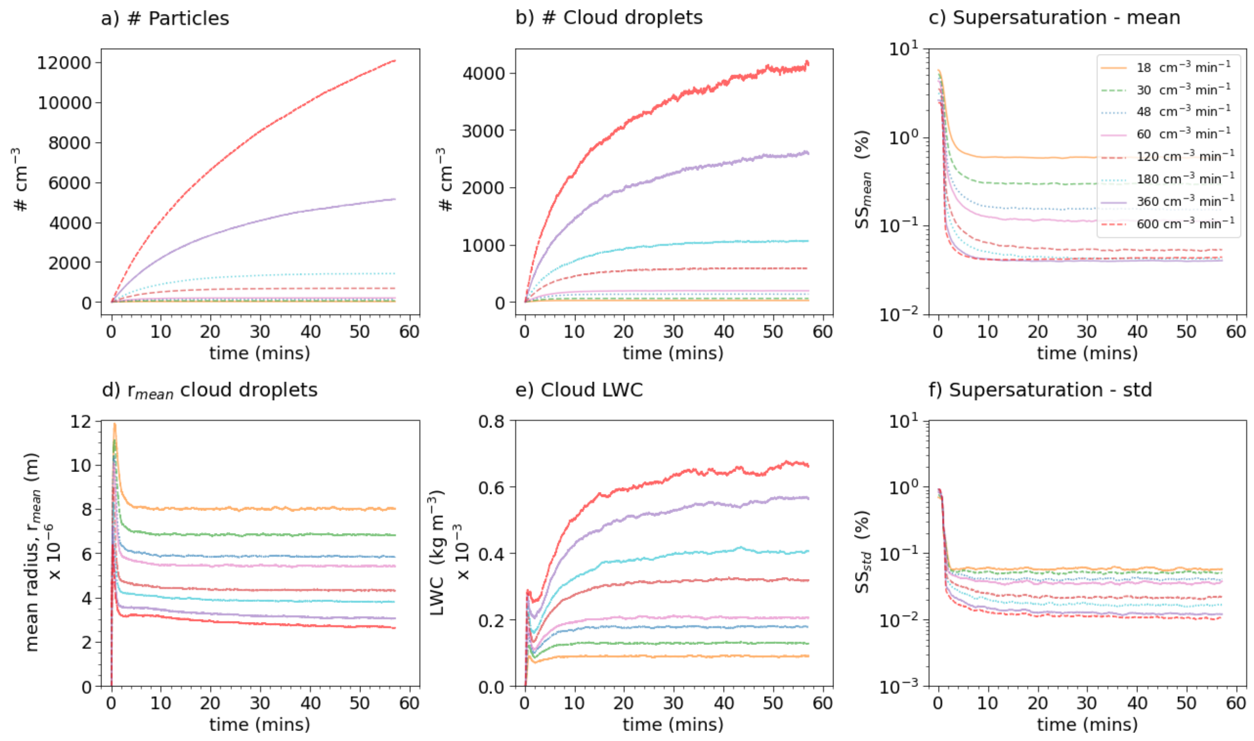

Fig. 4. The evolution of microphysical properties and domain supersaturation