CSU – RAMS

HUCM-SPECTRAL-BIN MICROPHYSICS MODEL

This document contains details regarding the Hebrew University Binned Microphysics model that was interfaced into CSU-RAMS version 6.1.20 by Adele Igel.

Updated by:

Adele Igel & Stephen Saleeby
Department of Atmospheric Science
Colorado State University

Last updated: 12 September, 2016

Bin Model Implementation in RAMS
Source files for the bin microphysics are found in the folder micro_bin

First, a little about the HUCM bin scheme. There are seven hydrometeor groups that are each represented by 33 mass doubling bins. These are (with model names in parentheses:

- liquid water (FFCD)
- columns (FFIC)
- plates (FFIP)
- dendrites (FFID)
- snow (FFSN, what RAMS calls aggregates),
- graupel (FFGL)
- hail (FFHL).

The three crystal types, (FFIC, FFIP, and FFID) and their properties are always combined into common arrays in the source code. For example, the fall velocities of the three crystal types are all stored in a 33x3 matrix called 'VR2' whereas all other hydrometeor types have their own 33x1 vector for fall velocity (e.g. 'VR1' for liquid water). The masses are the same for all hydrometeor types, but they still are stored in separate variables should they be changed in the future – XL, XI (all three crystal types), XS, XG, and XH.

The aerosol distribution is also represented by 33 mass doubling bins. It is called FNCN with bin masses in XCCN. A separate variable for ice nuclei is used for the CNT nucleation scheme called FFIN which also uses XCCN for the mass for each bin.

The number of bins is stored in the variable NKR. This technically should be a grid size variable, and perhaps defined in mem_grid with the other grid variables. I have left it in micro_prm in order to keep the bin code as separate as possible from the rest of the code. As a result, "use micro_prm, only: nkr" shows up in a lot of subroutines, such as those for memory allocation and others which need to know the grid size.

The nucleation and condensation schemes are called multiple times in one time step. The number of times that these schemes are called per time step is controlled by NCOND. Rather than using just the current values of temperature and water vapor, these quantities are stepped with each iteration from their previous values at the end of the last microphysics call, to their current values, accounting for changes that occur due to the condensation or evaporation that is occurring along the way. Since the old values of temperature and water vapor are needed, there are new variables T_OLD and RV_OLD in the microphysics table. These are written to the output files as they are necessary in order to do a history restart.

The original code calculated the advection of supersaturation for use in the Meyers ice nucleation scheme. See Khain et al. 2004 EQ 4. They use a semi-Lagrangian

approach that requires a change in supersaturation. They calculate this change as (assuming NCOND=1, that is only one iteration, for simplicity) the difference in the current and old value of supersaturation at the grid point, plus the change due to advection. This does not make sense to me, because the difference of the current and old values of supersaturation already has advection taken into account. Therefore, I have removed the calculation of supersaturation advection, which was not done in the same way as RAMS advection anyway, and only use the current and old values of supersaturation in the Meyers ice nucleation scheme.

A little more about NCOND. The variables that are actually stepped are RHw and RHi. After these variables are stepped, the current water vapor mixing ratio and temperature are solved for. When NCOND > 1 and RHw and RHi have decreased since the last time step, you can sometimes have a situation where RHw > RHi after stepping. This is unphysical and will result in negative water vapor. I don't see any satisfactory way to deal with this. In the WRF implementation, if this issue arises then the rest of condensation is simply skipped. I don't think that just skipping a physical process during a time step is acceptable. My advice is just to always use NCOND=1. In this case, T_OLD and RV_OLD are unnecessary.

The code details:

-There are three source files in the folder micro_bin:

- **micro_prm.f90**: constants and parameters used by the bin scheme
- **microphysics.f90**: contains the main bin microphysics driver, microphysics initialization routines, and any functions that I custom wrote for the scheme including the precipitation routine and several functions for summing the bin distributions to get total mass and number
- **module_hujisbm.f90**: contains the routines for all microphysics processes including condensation, nucleation, collision-coalescence, etc.

-Additionally, data needed by the scheme are found in a folder called 'sbm_input' which is located in the folder 'etc'.

-Bin microphysics is activated by RAMSIN option LEVEL=4.

-The flag ICEPROCS is used to determine if ice species are activated in a run.

-This is linked to the IPRIS, ISNOW, etc. namelist options. If any of them are set to a value greater than zero, then ice processes are activated

-There is currently one aerosol mode that is initialized corresponding to CCN.

- The number concentration is set by CCN_MAX in RAMSIN.
- The median radius and width (σ) of the distribution are set by rp_g and sig_g, respectively in micro_prm.f90. The median radius is set in RAMSIN (AERO_MEDRAD) and sig_g is hard coded to be 1.8.
- The number of ions (ions) and molecular weight (mwaero) of the aerosol can correspond to either ammonium sulfate or sodium chloride. The choice of aerosol type is given by IAERO_CHEM in RAMSIN

- There is no GCCN mode, but one could be easily added to the initialization routines. Note that the maximum aerosol size though is 1 micron diameter.
- The initial vertical profile of aerosol concentration is the same as for the RAMS bulk micro scheme.
- There is a flag diagCCN that gives the option of having a diagnostic rather than prognostic aerosol scheme. This flag is hard coded in micro_prm to use the prognostic scheme.

-There are two ice nucleation options. IIFN = 1 for Meyer's nucleation. IIFN = 2 for Comstock's classical nucleation. Only IIFN = 2 requires IN. IN are initialized using CIN_MAX. Within the source code, IIFN = ICEFLAG + 1. I could have just replaced ICEFLAG everywhere with IIFN, but I wanted to preserve the original code as much as possible.

-Jiwan Fan added CCN and IN regeneration schemes. I don't know how well tested these routines are. They are currently commented out in microphysics.f90

-A precipitation scheme compatible with the RAMS grid structure needed to be written. I have adapted Steve Saleeby's sedimentation scheme that allows for the sedimentation of number and mass separately for this purpose. This being a 1M bin scheme, only the sedimentation of mass portion needed to be retained (and modified for the bin species).

-The hydrometeor species have been interfaced with the radiation scheme. Since the current radiation lookup tables are designed for a bulk scheme, the bin scheme is treated as a bulk scheme for the purposes of radiation. The total number concentration and mixing ratio for every subset of bins that corresponds to a RAMS bulk microphysics variable is used to find the characteristic diameter. The namelist parameter GNU is used for values of nu for calculating this characteristic diameter. Power law relationships between the mass per particle and the diameter were found by empirically fitting the mass and diameter values found in the sbm_input folder. The relationships are as follows:

Bin variable (corresponding bulk variable)	Mass = cf *diam^pw		Notes
	cf	pw	
Liquid (cloud and rain)	523.4	3	
Small columns (pristine ice)	22.1	3	There is an obvious discontinuity in the mass data for this category. The two fits correspond to these two sections of the data
Large columns (snow)	2.924	2.841	
All plates (pristine ice and snow)	0.884	2.474	
Small dendrites	1.124	2.523	Again there is a discontinuity in the

(pristine ice)			data. The first fit is for bins 1-20. The second is for bins 21-33.
Large dendrites (snow)	0.023	2.038	
Snow (aggregates)	4.843	2.584	
Graupel	209.4	3	Density = 400 kg/m3
Hail	471.2	3	Density = 900 kg/m3

The cutoff between pristine ice and snow in RAMS, and hence also in this table, is 125 microns. The first bin with a size greater than 125 microns for all three crystal types is stored in the parameter KRPRIS. There is also a parameter KRDROP that corresponds to the first rain bin in the liquid hydrometeor type. This is set to also correspond to the RAMS definition of rain. Another parameter could be implemented if users wanted to define a drizzle category.

-A negative adjustment scheme has been written. It has also been adapted from that used for the bulk scheme in RAMS. Any bin with a mixing ratio less than 1.e-12 kg/kg of water is set equal to zero. This is the same threshold used for the bulk scheme. Positive bins are adjusted to preserve the total mixing ratio in the grid box. A similar approach should be used for aerosol, but currently these species are simply set to zero where they are less than 1.e-12 kg/kg. This scheme is called adj1_bin in micro_bin/microphysics.f90. It is called from negadj in micro/mic_misc.f90. Adding the call to adj1_bin in negadj is the only change to the code in the "micro" folder.

-A new scheme was also needed in rthrm.f90 to diagnose rv and theta. This is called wetthrm3_bin.

- Added new variables to mem_micro. New bin variables are 4D. These are the first 4D variable to occupy the atmosphere (some soil variables are 4D but have a different number of vertical levels). I assigned these variables idim_type=7 for use in anal_write.f90

-In anal_extra.f90, I have included additional "extra" variables for writing. These are the traditional mixing ratio and number concentration fields that are written automatically for LEVEL =3. I suppose these could be moved to REVU, but it is rather convenient to have them automatically output by RAMS.

-Added the new bin variables to the scalar table. Each bin variable is 4D, but each index in the 4th dimension (each bin) is really a separate variable that needs to be treated as such for advection, diffusion, etc. Therefore, I have included in the scalar table a separate pointer for each index in the 4th dimension. These are continuous chunks of data in each variable, so I do not anticipate major slow downs by setting the pointers this way. Treating the bin variables in this way for the scalar table means that the scalars in the table are all still 3D.

RAMSIN parameters to worry about:

LEVEL = 4 to turn on bin microphysics

IMBUDGET – same meaning for options 1 and 2. Option 3 is not used since the bin scheme does not include dust.

For FULL-SBM: IPRIS ≥4, IGRAUP≥1, IHAIL≥1

For LIQUID-ONLY-SBM: IPRIS, ISNOW, IAGGR, IGRAUP, and IHAIL all equal 0.

For FAST-SBM:

There are a lot of options here. Only IPRIS, IGRAUP, and IHAIL are needed:

1. Traditional FAST-SBM. Only liquid, aggregates, and graupel. Set IPRIS=0, IHAIL=0, and IGRAUP ≥1
2. FAST-SBM-HAIL. Like traditional FAST-SBM, but uses hail instead of graupel. Set IPRIS=0, IGRAUP=0, and IHAIL≥1
3. FAST-SBM-ONE-CRYSTAL-TYPE. The full SBM uses three ice crystal types, namely, columns, plates, and dendrites. To use just one crystal type, set IPRIS=1 for columns, IPRIS=2 for plates, or IPRIS=3 for dendrites.
    - 3a. For graupel and hail, both IGRAUP and IHAIL >0
    - 3b. For just graupel, IGRAUP ≥1, IHAIL = 0
    - 3c. For just hail, IGRAUP = 0, IHAIL ≥1

Either graupel or hail must be turned on. Aggregates are always used.

GNU for shape parameters used by the radiation code

CCN_MAX

CIN_MAX

IAEROHIST

AERO_MEDRAD

IAERO_CHEM

IIFN: 1 for Meyers, 2 for CNT

None of the other microphysics parameters are linked to the bin scheme!

New parameters:

NDTCOLL – collisions called every NDTCOLL time steps. For example, NDTCOLL = 3 means that collision-coalescence will only be called every third time step.

Complete list of files that have been modified and a summary of the changes:

core/rthrm.f90: rv and theta diagnosis (new routine wetthrm3_bin).

core/rtimh.f90: add if level=4 call to micro_bin. Also, only call some routines if LEVEL is not equal to 4.

init/rdint.f90: if level=4 call to bin micro_init where necessary

io/anal_extra.f90: add mixing ratio and number concentration variables to “extra” write-out vars

For FAST-SBM: added new variable max_num_extra. With the flexibility of the FAST-SBM code, we will need to actively skip extra variables – they simply cannot be ordered in a way that makes this

easy. Then, in anal_extra_comp, added an extra argument, skip, to determine if the extra variable actually needs to be written

io/anal_write.f90: instructions for idim_type=7
For FAST-SBM. Changed the extra variable write loop to do 1 to max_num_extra

io/history_start.f90: modify maxarr to include max size of bin array in history_start; add case(7) for unarranged of in vars in hist_read

io/opspec.f90: New checks on RAMSIN options that comply with bin needs. Set iceprocs to 1 if necessary.

io/rname.f90: change allowable range for LEVEL to 0-4

memory/mem_all.f90: added module micro_prm to the list

memory/mem_micro.f90: added bin micro variables, including new precip vars accpic, accpip, accpid, etc. and old budget vars nucicect, nuccldct, inuchomct, inucifnct.
For FAST-SBM: included IF statements based on IPRIS, IGRAUP, and IHAIL to only allocate species if they are wanted.

memory/alloc.f90: alloc_micro has number of bins passed in.

memory/mem_scratch.f90: size of scratch arrays scr1, scr2 increased to nxp*nyp*nzp*nkr only if bin micro is on; size of vt3dp (used primarily in parallel code) increased to nxp*nyp*nzp*nkr if level=4

memory/mem_tend.f90: bin micro tendency arrays added; calls to vtables_scalar added for them; npts added to filltab_tend subroutine call

micro/mic_misc.f90: call to bin negative adjustment scheme in negadj

mpi/mpass_cyclic.f90 : For nbuffsend_cyc and nbuffrecv_cyc, multiply max(2,npvar) by nkr.

mpi/mpass_lbc.f90: added code for idim_type=7 in order to pass bin vars. Added routines mklbcbuff4 and exlbcbuff4

mpi/para_init.f90: allocate bigger buffers – added npvar7 which is used analogously to npvar2, npvar3, for allocating buffers

mpi/mpass_full.f90: added code for idim_type=7 for send and receive of bin vars

mpi/mpass_init.f90: Add send and receive of iceprocs to broadcast_config() for dm

mpi/node_mod.f90: only for dm: added variable mmxyzbp, which is number of bin points, but it isn't actually used anywhere

radiate/rad_driv.f90: new routine cloud_prep_lev4, and new code for level=4 in cloud_opt and IF statement to call cloud_prep_lev4, some places where if level≤2 changed to level≤2 or level==4
For FAST-SBM: included IF statements based on IPRIS, IGRAUP, and IHAIL in cloud_opt

turb/turb_k.f90: Added code to get rcp for subroutine bruvais.

lib/hdf5_utils.f90: shdf5_set_hs_select – add case for idimtype = 7. Changed options for current 7 and 8 to 8 and 9, respectively.

isan/isan_io.f90: where it appears change shdf5_set_hs_select(7, … to (8, … and same for (8, … to (9, …

objects.mk: add new files that need to be compiled

Changes to the bin code itself:

-Found an error in ONECOND2!! Wasn't doing condensation/evaporation for the snow category (RAMS aggregates)
-Some parts of main code moved to initialization routine
-Added pcpg, qpcpg, and dpcpg for use in surface routines
-Added regular RAMS precip variables accp*, pcpp*, and pcpv*
-Added budget variables analogous to the RAMS bulk micro budget variables
-Removed calculation of dsupice_xyz
-Simplified CCN initialization. The original code had three observed distributions or something. Didn't really understand what it was doing. Deleted this code, but left the theoretical part. It is now a lognormal distribution. Initial concentration is determined in the same way as for bulk.
-New routines sum_bins, sum_bins_conc to calculation mixing ratio and number concentration
-New routines sum_snow, sum_pris for doing budget variables
-New sedimentation scheme sedim_bin (see above). Sedimentation scheme is called before any other processes
-New routine update_thermo to update thp and rtp after precip.
-Some code simplification without changing logic or physics
-variable dthalf is supposed to be the collision-coalescence time step and was originally set to be DT or DT/2 depending on the value of NCOND (why it was related to NCOND, I don't know). However, later in the code, collisions were only called every third time step, which is inconsistent with DT or DT/2. I removed dthalf and replaced it with dtcoll which is equal to DT*NDTCOLL.
-routine coll_xyyz was producing negative liquid water. Added an if statement to prevent this issue.

Concerns and Ideas for future improvement:

-Vapor variable is specific humidity, while we use mixing ratio. I do not know if this is original to the code, or a change made by SAM developers, if it even matters much
-Instantaneous melting
-Meyers ice nucleation for contact nuclei not implemented correctly. But the contact nucleation is zero right now, so it doesn't matter.
-Uses base state pressure. Consider implementing full 3d pressure fields
-Diagnose the shape parameter for the radiation scheme
-Make aerosols radiatively active

SBM Bin Distributions

Bin limits are mass doubling.

$$F\ ***= \frac{d(mixing\ ratio)}{d\ln(radius)}$$

Where F*** is FFCD, FFIP, FFIC, FFID, FFSN, FFGL, FFHL, or FNCN

$$d\ln(radius) = \frac{\ln(2)}{3} = \frac{d\ln(mass)}{3}$$

--proof

$$d\ln(mass) = \ln(2) = d\ln\left(\frac{4}{3}\pi(radius)^3\right)$$

$$\ln(2) = d\ (\quad ^3)$$

$$\ln(2) = 3\,d\ln(radius)$$

Therefore:

$$Total\ mixing\ ratio\ (kg/kg) = \frac{\ln(2)}{3}\sum F\ ***$$

$$Total\quad concentration\ (\#/g) = \frac{\ln(2)}{3}\sum \frac{F\ ***}{x*}$$

$$diameter = \left(\frac{mixing\ ratio(\frac{kg}{kg})}{N\left(\frac{\#}{\quad}\right) * \rho water(\frac{kg}{m^3})} * \frac{6}{\pi}\right)^{1/3}$$

where x* is the mass (g) per particle and is given by HUCM variables XL, XI, XS, XG, and XH.

In the microphysics subroutine, the variables stored in memory (FFCD, etc.) are converted to new units. FFCD → FF1R, FFSN →FF3R, etc. Specifically:

$FF\#R = \frac{F***\rho}{3x*^2}$ I think this is needlessly complicated. The units work out to be d(number concentration)/d(ln(mass))/mass per particle = #/cm3_air/g_water. Everywhere these variables are used, they must first multiply by the bin mass (e.g. XL) in order to get back to useful units. This adds up to a lot of needless computations. But it would be a big hassle to change this.