User Guide

Goddard Satellite Data Simulator Unit (G-SDSU)

Version 3.5.1

Document Revision: 5
20 Jun 2016

Authors: Toshi Matsui[1&2], Eric Kemp[1&3]

[1]NASA Goddard Space Flight Center, Greenbelt, MD
[2]ESSIC UMCP, College Park, MD
[3]Science Systems And Applications Inc, Lanham, MD

Table of Contents

1. Overview **2**
  1.1 Enhancements and Updates **2**
  1.2 Current Components **3**

2. Using the Software **7**
  2.1 Structure of the Source Code **7**
  2.2 Source Code Structure and Building **7**
  2.3 Running G-SDSU for Checking I/O **8**
  2.4 Configuration of Running Satellite Simulator **13**
    2.4.1 Running Microwave Simulator 13
    2.4.2 Running Radar Simulator 16
    2.4.3 Running Visible-IR Simulator 18
    2.4.4 Running Lidar Simulator 20
    2.4.5 Running Broadband Simulator 21
    2.4.6 Running GV Simulator 22
    2.4.7 Running SARTA Simulator 23
    2.4.8 Additional Settings 24
  2.5 Known Issues and Limitations **26**

3. Citations **27**

1. Overview

Modern multi-sensor satellite observations provide a more complete view of land, cloud, precipitation, and aerosols processes; meanwhile, it is becoming a challenge for remote sensing and modeling communities to harness these observations simultaneously due to inconsistent physics assumptions and spatial scales between satellite retrievals and model physics. To this end, a unified system of multi-sensor simulators, the Goddard Satellite Data Simulator Unit (G-SDSU), has been developed through multi-institutional collaborations [*Matsui et al.,* 2013; *Matsui,* 2013; *Matsui et al.,* 2014] upon the initial version of HyARC SDSU [*Masunaga et al.,* 2010]. The G-SDSU is the end-to-end satellite simulator, and computes satellite-consistent Level-1 (L1) measurements (e.g., radiance/brightness temperature or backscatter), from outputs of the NASA-Unified Weather Research and Forecasting (NU-WRF) through various simulators:

- Microwave simulator: SSMI, SSMIS, TMI, GMI, AMSR, AMSU, MSU, etc.
- Radar simulator: PR, DPR, CPR, etc.
- Visible IR simulator: AVHRR, MODIS, VIRS, GLI, GOES APR, etc.
- LIDAR simulator: CALIOP, CATS, etc.
- Broadband simulator: ERBE, CERES, etc.
- SARTA simulator: AIRS.

G-SDSU-simulated L1 signals can be directly compared with the satellite-observed L1 signals; therefore G-SDSU bridges the model and satellite remote sensing through the following paths: i) radiance-based model evaluation and development [*Matsui et al.,* 2009; *Li et al.,* 2010], ii) an operator of radiance-based data assimilation system [*Zupanski et al.,* 2011; *Zhang et al.,* 2013], iii) development of synthetic satellite observations for future satellite missions [*Matsui et al.,* 2013] and satellite-retrieval database [Kidd et al., 2015]. Users should cite the following paper as the main reference in G-SDSU V3.5.1.

Matsui, T., J. Santanello, J. J. Shi, W.-K. Tao, D. Wu, C. Peters-Lidard, E. Kemp, M. Chin, D. Starr, M. Sekiguchi, and F. Aires, (2014): Introducing multisensor satellite radiance-based evaluation for regional Earth System modeling, *Journal of Geophysical Research*, *119, 8450–8475, doi:10.1002/2013JD021424.*

1.1 Enhancements and Updates

G-SDSU version 3.5.1 was released along the release of NU-WRF Version 8 “Bjerknes”. Upgrades from the previous version V3.3.3 include: i) a global IR emissivity database from the University of Wisconsin [*Seemann et al.,* 2008]; ii) modified size boundaries for precipitating and non-precipitating particles (200µm diameter threshold consistently) for computing microwave/visible-IR optical properties; iii) a UV-VIS-IR non-spherical ice scattering database for the visible and IR simulator from Texas A&M University [*Yang et al.,* 2013]; and iv) new Thompson microphysics options [*Thompson et al.,* 2008]. The new emissivity map i) will affect IR brightness temperature especially around desert regions, where land-surface emissivity greatly varies. The release version contains only one monthly database due to its large size. User may individually request IR emissivity data from CIMSS/University of Wisconsin (http://cimss.ssec.wisc.edu/iremis/). Correction ii) changes the cloud emission in microwave simulator, especially around 37GHz. It tends to increase cloud-related absorption/emission of microwave radiance, hence the overall impact is to increase TOA microwave Tb where cloud particles are present. This also changes the visible-IR radiance for some specific overlapping of non-precipitation and precipitating particles. New database iii) will affect visible-IR radiance and broadband radiation flux over ice-particle clouds, and a few options are available for sensitivity studies. New database iii) also enables computing the de-polarization ratio (P22 component of polarized lidar backscatter) in the lidar simulator, which is now included in the lidar-simulator output automatically. New options iv) allow Thompson-microphysics users to employ a microphysics-consistent treatment in calculating single-scattering properties for various satellite simulators.

1.2 Current Components

- **Input Model options**: G-SDSU can directly read the NU-WRF's native output format (NetCDF 4), including surface fields, vertical profiles of pressure, geopotential height, water vapor, temperature, vertical velocity, and variety of condensates and aerosol particles. Surface properties include geolocation (latitude and longitude), elevation, land-cover, vegetation fraction, soil temperature, soil moisture and snow depth. (G-SDSU additionally reads data from the Goddard Cumulus Ensemble (GCE) model, NASA Multi-Modeling Framework (MMF), and NASA Goddard Earth Observing System Version 5 (GEOS-5).

- **Orbit and Scanning options:** The G-SDSU scanning module accounts for satellite orbit and sensor scanning patterns and associated geolocations with respect to NU-WRF model coordinates. Satellite orbit is predicted by Keplerian parameters and Kozai's first-order perturbation parameters. Scanning geolocations can be tracked by scan type, rotation speed and angles, and sampling rates. A limited number of satellites and sensors are pre-registered in the G-SDSU. For unregistered instruments, users must provide specific information on their characteristics.

- **Cloud Microphysics options**: One/two-moment bulk schemes (e.g., Goddard scheme, RAMS scheme, Morrison scheme) as well as spectra-bin (HUCM SBM scheme) microphysics are supported. Consistent assumption of particle size distributions (PSDs), density, and hydrometeor classes are used to calculate optical properties.

- **Aerosol Microphysics options**: G-SDSU currently supports only the GOCART aerosol scheme.

- **Microwave optical properties**: Microwave optical properties (extinction, single scattering albedo, asymmetry, and backscattering coefficient) are computed through the Lorenz-Mie solution with an *a priori* database of the complex refractive indices. Mixture of the refractive indices can be estimated through either the Maxwell-Garnet approximation or the effective-medium approximation. Non-spherical single-scattering properties of ice crystals can be optionally obtained from the SCATDB [*Liu,* 2008]; however, the database is generally limited to small-size particles. Recently, the T-matrix database was added [*Meneghini and Liao*, 1996]. Gaseous extinction and absorption (H_2O, N_2, and O_2) are parameterized by the empirical fitting parameters [*Rosenkranz,* 1993]. These microwave optical properties are computed identically for both the microwave and the radar simulators.

- **Microwave simulator**: Top-of-atmosphere (TOA) emergent microwave brightness temperature (Tb) is derived from two-stream radiative transfer with Eddington's second approximation along the model column [*Kummerow,* 1993]. More realistic pseudo 3D (1D slant radiance path) calculations [*Olson et al.,* 2006] are available with the microwave scanning option, which account for satellite orbit and scanning along the microwave simulator. Horizontal and vertical polarization is affected by surface emissivity only. If horizontal grid spacing of the NU-WRF simulation is finer than the footprint size of microwave, simulated microwave Tb is convoluted within the effective field of view (EFOV). There is also an option that computes the bottom-of-atmosphere (BOA) downwelling microwave Tb for a ground-based radiometer.

- **Radar simulator**: Attenuated and non-attenuated radar reflectivity and Doppler velocity profiles are derived through a single scattering process along the model column [*Masunaga and Kummerow,* 2005]. More realistic pseudo 3D (1D slant radiance path) calculation [*Meneghini and Kozu,* 1990] is also available with the radar scanning option, which accounts for satellite orbit and instrumental scanning along the radar simulation and surface clutter impact. If horizontal grid spacing of the NU-WRF simulation is finer than the footprint size of microwave, simulated reflectivity are convolved for range resolutions and instantaneous field of view (IFOV). There is also an option that computes radar reflectivity from the ground.

- **Visible-IR optical properties**: Visible-IR optical properties (extinction, single scattering albedo, phase function, and backscattering coefficient) are computed through the Lorenz-Mie kernel with an *a priori* database of the complex refractive indices of clouds and aerosols. Gaseous extinction and absorption (H_2O, CO_2, O_3, N_2O, CO, CH_4, and O_2) are parameterized by the k-distribution method using the HITRAN 2004 database [*Sekiguchi and Nakajima,* 2008]. Since NU-WRF, without WRF-CHEM, generally predicts only water vapor (H_2O), vertical climatological profiles of various gaseous constituents are interpolated to model levels. These visible-IR optical properties are treated identically between the visible-IR simulator and the lidar simulator.

- **Visible-IR simulator**: TOA emergent visible radiance and IR Tb are derived from a discrete-ordinate radiative transfer scheme along the model column [*Nakajima and Tanaka,* 1986] with different phase-function calculation modes [*Nakajima and Tanaka,* 1988]. The number of streams can modulate accuracy of multiple scattering processes; the default is six streams, which result in reasonable convergence of accuracy in treating multiple scattering processes. Because of this, the visible-IR simulator takes several times more computational time than the microwave simulator.

- **Lidar simulator**: Attenuated and non-attenuated lidar backscattering profiles are computed from single scattering processes along the model column [*Platt,* 1973]. Backscatter is computed at each range resolution. Since lidar instruments have finer horizontal resolution IFOV than the NU-WRF simulation, there is no backscatter convolution process. Multiple scattering and instrumental random noises are not currently calculated in the lidar module.

- **Broadband optical properties**: Broadband optical properties (extinction, single scattering albedo, and asymmetry parameter) are computed from line-by-line integration using the visible-IR optical properties. Molecular absorptions are parameterized by the correlated K method, and have not been integrated between broadband and visible-IR optical properties yet.

- **Broadband simulator**: TOA broadband fluxes and profiles of radiative heating rate are computed through the delta-Eddington two-stream adding approximation [*Chou and Suarez,* 1999; *Chou et al.,* 2001]. The shortwave scheme accounts for the absorption due to water vapor, O_3, O_2, CO_2, clouds, and aerosols. The longwave scheme accounts for the absorption due to major gases and most of the minor trace gases, as well as clouds and aerosols. Accuracy of the parameterized broadband fluxes is within 1% in comparison with the high spectral-resolution line-by-line calculations.

- **SARTA simulator**: Stand-alone *AIRS* Radiative Transfer Algorithm (SARTA) is developed at UMBC Atmospheric Spectral Lab (http://asl.umbc.edu/pub/packages/sarta.html). AIRS hyper spectral radiance/brightness temperatures are computed through the very fast radiative transfer model for clear-sky conditions [*Stow et al.* 2003].

- **Surface Properties**: Over-land microwave surface emissivity is computed by either a dynamic or a static module. The dynamic module is the CRTM microwave emissivity model [*Weng et al.,* 2001], while the static module is TELSEM [*Aires et al.,* 2011]. For both schemes, at a strong-rainfall pixel, the surface is treated as a water body that mimics surface inundation. Over-ocean microwave emissivity is a function of ocean salinity, temperature, and wind speed [*Wilheit,* 1979]. Visible surface albedo is derived from the MODIS satellite product and stored mean values of different surface land-cover type in look-up tables. There is no consideration of the bi-directional reflectance distribution function (BRDF) over land. Over ocean, visible-IR albedo/emissivity is calculated through a matrix formulation that accounts for BRDF in response to sensor-solar cone angle and wind speed effect. Land IR emissivity is the MODIS-based empirical global map (0.05deg) [*Seemann et al.,* 2007].

- **GV simulator**: This simulator is specifically designed for GPM Ground-Validation instruments: the aircraft- (*2D-P* and *2D-C*) and surface-based (*Parsivel*) microphysics probes. Currently, it has been only applied to HUCM spectral-bin microphysics scheme [*Iguchi et al.,* 2012a; *Iguchi et al.,* 2012b]. The GV module integrates and re-samples HUCM SBM bin-by-bin microphysics information, and generates aircraft 2D- or ground Parsivel-observable bulk microphysics properties.

- **MPI**: The G-SDSU can optionally utilize the Message Passing Interface (MPI) library [*Aoyama and Nakano,* 1999] for parallel computation on multi-core super computers. Two MPI options are available. The first option is file-number decomposition, which has greater advantage to process a large number of model inputs. The second option is domain decomposition, which assign different tiles of NU-WRF domains for different processors. The second option also involves memory decomposition; thus it is advantageous when processing a large-domain model file (or complex bin microphysics).

2. Using the Software

2.1 Structure of the Source Code

G-SDSU V3.5.1 is included within the package of NU-WRF V8 “Bjerknes”. After opening the NU-WRF package, you will find GSDSU directory and the following sub-directories:

`SRC: Source codes of G-SDSU (Fortran.F programs).`

`QRUN: Default directory (**** could be anything) to run G-SDSU.`

`INPUTS: Default directory to store model (NU-WRF) input files.`

`OUTPUTS: Default directory to store the simulated satellite signals.`

`DATAFILES: various run-time input files.`

`SSLUT: Storage of single-scattering look-up tables.`

`REFFERENCE: References for G-SDSU.`

`QUICK_START_GSDSU_V3.5.1.txt is the quick-start guide.`

2.2 Source Code Structure and Building

G-SDSU is built upon object-oriented Fortran90 programming (and some old Fortran codes) and C programming. The C-preprocessor is used for the MPI library and their call routines. Dynamic allocation allows better memory management for large memory data. Many computational intensive calculations use pre-computed look-up tables that can boost the computational speed (> 10x) without losing accuracy. The G-SDSU can optionally utilize the Message Passing Interface (MPI) library for parallel simulations on a multi-core computer. Thus, compilation of the G-SDSU requires 1) a Fortran compiler, 2) a MPI library (optional), 3) the NetCDF library, 4) a C-compiler, 5) a C-Preprocessor, and 6) the Make utility in the standard Unix-flavor machine (UNIX, LINUX, Mac OSX, and etc.). Thus, users of G-SDSU require basic knowledge of these computing environments.

For building the G-SDSU, you can use NU-WRF compiler script (./build.sh gsdsu) in the top NU-WRF directory. Otherwise, you can take following steps.

1. Go to SRC directory.

2. Modify `makefile` according to the compiler and library options.

- Make sure the QRUN directory (running directory) is consistent to what you defined.

  ```
  QRUN      = ./../QRUN
  ```

- Modify compiler options, library, and flags. (Some parameters are available for the NCCS Discover and the NAS Pleiades as default.) Binary data is assumed in little endian format (in case of Intel Fortran `-assume byterecl`), and you must put an appropriate compiler flag for this.

  ```
  CPP     = /lib/cpp -C -w
  CF      = ifort
  CFFLAGS = -O3 -ip -assume byterecl
  CC      = icc
  INC_NETCDF = /nasa/netcdf/3.6.0/intel/include
  LD_NETCDF = /nasa/netcdf/3.6.0/intel/lib
  LINK_MPI = -lmpi
  ```

3. Modify define_CPP.h file depending on your library (HDF or MPI).

- The MPI option can be either MPI 0 - no mpi, 1 - file decomposition, or 2- domain decomposition. Note that MPI=1 option is most efficient to process many input files, but it often creates memory issues with large-domain inputs. File MPI=2 option is more memory friendly, since it assigns sub-domain on each CPU memory.

  ```
  # define MPI 2
  ```

- HDF option is for the synthetic GPM simulator, and generally not used: 0 - no HDF library, 1 - HDF library.

  ```
  # define HDF 0
  ```

4. Compile the source codes by typing

```
>make
```

This will take 10 seconds to a few minutes, depending on the Fortran compiler flags. Executable (GSDSU.x) will be copied to the QRUN directory.

2.3 Running G-SDSU for Checking I/O

For the first time, it is important to check if G-SDSU running without any I/O problems.

1. Make necessary changes for run-time parameters in Configure_SDSU.F to have a consistent I/O directory. First, modify the I/O direction, and try running G-SDSU without changing any other options.

Turn off (.false.) all simulator switches for the 1st time (we just want to check I/O process).

```
$simulator_switch
  micro  = .false.     ! Microwave simulator switch; on when .true.
  radar  = .false.     ! radar simulator switch; on  when .true.
  visir  = .false.     ! visible/IR simulator switch; on when .true.
  lidar  = .false.     ! Lidar simulator switch; on when .true.
  broad  = .false.     ! Broad-band simulator switch; on when .true.
  sarta  = .false.     ! SARTA simulator switch; on when .true.
  GV     = .false.      ! GV simulator switch; on when .true.
$end
```

Here is the default setting of io_option for first-time running.

```
$io_options
  sdsu_dir_sslut= './../SSLUT/'  ! directory for the single-scattering LUTs
  sdsu_dir_data = './../DATAFILES/ ! directory for various datafiles
  sdsu_io_name  = 'inpfile_wrf' ! name of model-input-list file
  verbose_SDSU  = .false.   ! if true, print out more comments during run.
  write_surface = .false.   ! if true, write out surface properties
  write_opt     = .false.   ! if true, write out single scattering properties
  write_CRM3D   = .true.   ! if true, write out CRM 3D file in GrADS format
  write_CRM2D   = .true.   ! if true, write out CRM 2D file in GrADS format
  write_grads_ctl = .true.  ! if true, write grads control files (logical)
  write_cftext  =  .false.  ! if true, write climate-format text    output for Super Cloud
Library (logical)
  cf_init_time  = '2011-11-23_00:00:00' ! if write_cftext is true, you must specify initial time
in this format.
  write_psdmmt  = .false.   ! if true, write out PSD moments from HUCM SBM
  output_suffix = '_test'   !it can be any, if you don’t want just black ‘’

  minlat        =  58.0  ! min longitude of lat-lon output ( -90 ~  90) [deg]
  maxlat        =  62.0  ! max latitude  of lat-lon output ( -90 ~  90) [deg]
  minlon        =  21.0  ! min longitude of lat-lon output (-180 ~ 180) [deg]
  maxlon        =  28.0  ! max longitude of lat-lon output (-180 ~ 180) [deg]
  res_latlon    =  0.05  ! grid spacing  of lat-lon output [deg]

$end
```

`write_CRM3D` and `write_CRM2D` are generally activated for diagnostic purpose. In general, `write_grads_ctl` must be always activated so that G-SDSU will write GrADS control files. `minlat`, `maxlat`, `minlon`, and `maxlon` are specific latitude-longitude output domain, and they should be close to the input NU-WRF domain. `res_latlon` could be close to the resolution of satellite footprint.

Input NU-WRF information must be specified within

```
$crm_options
 …….
 ……
 …..
$end
```

User must specify correct information of `sdsu_dir_input`, `sdsu_dir_output`, `mxgridx`, `mxgridy`, `mxlyr`, `gridsize`. For processing NU-WRF information, `sim_case` must be always `'WRF'`. Note that `mxgridx` and `mxgridy` can be smaller than actual maximum dimension of the NU-WRF file, but it cannot be larger than that. Also, it is very important that mxgridx and mxgridy are NOT for the staggered grid; they must be for the **UN-staggered** grid (e.g., east-west, north-south grid of WRF).

```
  sim_case  = 'WRF'  ! NU-WRF (character*10)
  sdsu_dir_input  = './../INPUTS/' ! G-SDSU input (character*200)
  sdsu_dir_output = './../OUTPUTS/'  ! output directory (character*200)
  mxgridx   = 423    ! max grid # in horizontal x direction  (integer)
  mxgridy   = 411    ! max grid # in horizontal y direction (integer)
  mxlyr     = 60     ! max grid # in vertical direction  (integer)
  gridsize  = 6.e0   ! horizontal grid spacing [km] (real)
```

Make sure the microphysics options are consistent to your NU-WRF simulation.

```
cloud_microphysics = 'GMP_4ICE'  ! Cloud Microphysics Type (character*20)
                             ! GOD: Goddard 1-mmt scheme [Tao et al. 2003]
                             ! GOD10: Goddard scheme 2010 [Lang et al. 2010]
                             ! GMP_4ICE: Goddard 4ICE scheme [Lang et al. 2013]
                             ! LIN: LIN bulk 1-mmt scheme  [Lin et al. ]
                             ! WSM: WRF-Single-Moment 6-Class Scheme [Hong et al. 2004]
                             ! RAMS1: RAMS 1-mmt scheme [Walko et al., 1995]
                             ! RAMS2: RAMS 2-mmt scheme [Meyers et al., 1997]
                             ! HUCM_SBM: HUCM spectra-bin microphysics scheme [Khain et al. 2007]
                             ! HUCM_SBM43: HUCM spectra-bin microphysics 43 bin scheme [Khain et
al. 2010]
                             ! MORR: Morrison two-moment scheme.

 gsfc_hail = .false.         ! GSFC hail options (graupel --> hail)
```

Below is a special setting for idealized forward simulations. For NU-WRF case, users won't activate these parameters.

```
clear_sky_scene = .false.   ! if .true., zero out all condensates (cloud-precip) to create clear
sky.

 uniform_surface = .false.             ! When it is true, this option assigns spatially uniform
                                       ! surface characters over the entire domain.
                                       ! (When sim_case='GCE', this must be always .true.,
because GCE input
                                       !  does not have these surface parameters.)

 idealized_surface%lat       = -12.75e0 ! latitude [deg]
 idealized_surface%lon       = 131.5e0  ! lon      [deg]
 idealized_surface%frac_veg  =   0.e0  ! vegetation fraction [%] (optional for WRF)
 idealized_surface%albedo    =   0.05  ! surface SW albedo [-]
 idealized_surface%h2o_snow  =   0.e0  ! snow water equivalent [kg m-2]
 idealized_surface%h2o_soil  =   0.9e0 ! soil moisture fraction [0-1]
 idealized_surface%elev      =   0.e0  ! surface elevation [m]
 idealized_surface%dhgt_snow =   0.e0  ! snow depth [m]
 idealized_surface%iland     =   2     ! 1-land,  2-water
 idealized_surface%igbp_typ  =   0     ! IGBP land-cover type (dominant vegetation type )
```

If you are running GOCART aerosols with WRF-CHEM, you may activate `account_aerosol` in order to account for GOCART aerosol mass concentrations.

```
 account_aerosol  = .false.   ! if true, account aerosol particles (logical)
```

2. Prepare the input file list `inpfile_wrf`, which is a simple text file that contains a list of input file names. This can be easily done by going to your NU-WRF output directory, and using the command

```
>ls wrf*d02* > inpfile_wrf
```

This means that all WRF domain 2 output will be written in inpfile.

Your `inpfile_wrf` looks like,

```
> vi inpfile_wrf
wrfout_d02_2011-05-20_10:00:00
```

Then, copy inpfile into your QRUN directory.

```
>cp inpfile $GSDSU/QRUN_####
```

Problem reading wrfoutput?

Note that G-SDSU only handle WRF instantaneous output files. E.g., one netcdf file contains one time output. For example, in your NU-WRF run, namelist.input must be specified as one:

```
frames_per_outfile = 1, 1, 1
```

If you put this parameter one than one, G-SDSU will only read first set of times frame in the netcdf file.

3. Running G-SDSU executable.

Go to the QRUN directory..

a. Running on your desk-top machine,

- For the single CPU option, simply type

```
>./GSDSU.x
```

- For the multiple CPU (MPI) option,

```
>mpirun -n## GSDSU.x
```

b. Running in Discover or Pleiades Super computer.

- For the single CPU option, simply type

```
>./GSDSU.x
```

- For the multiple CPU (MPI) option, you must submit Q_script:

  ```
  >qsub Qbatch***.sh
  ```

  or For Discover SLURM commands

  ```
  >sbatch Sbatch***.sh
  ```

  Before submitting a Q script, you must modify the Q script (or S script) for your own setting and resource requirements. Details can be seen on the NCCS (http://www.nccs.nasa.gov/primer/) or NAS websites (http://www.nas.nasa.gov/hecc/support/kb/).

c. Determining the number of CPUs in MPI runs.

- For MPI=1 (file decomposition) options, MPI will subdivide the number of input files into the discrete number of CPUs. For example, with 24 input files (e.g., NU-WRF netcdf output files), if you set 24 (2) CPU, each CPU will process 1 (12) file(s). Thus, it can be any number between 1 and the total input number (N); however, it would be ideal that the CPU number can divide N without residuals. E.g., if N=24, the number of CPUs could be 2, 3, 4, 6, 8, 12.

- For MPI=2 (domain decomposition) options, MPI will subdivide the input domain (i-j-k domain) into small tiles (di-dj-k domains). The CPU number should be any integer that can divide the domain without residual (equal sub-domain size). For example, with 10x10 (ixj) domain, the CPU number should be 2 (two 5x10 subdomains), 4 (four 5x5 subdomains), 5 (five 2x10 subdomains), 10 (10 1x10 subdomain), and so on. These number can be derived from running the Fortran executable

  ```
  > ./HOW_MANY_CPU_GSDSU
  ```

  This asks your domain size, and provides the combination of node number and total number of CPUs. (*Tips. For better configuration of node and threads combination, you may reduce the size of WRF domains. E.g., 113x111 horizontal domain can be reduced to 112 x 110 domain, and you will find more combinations of CPU-threads numbers*.)

In the default configuration, all simulators are inactivated (.false.) so that G-SDSU essentially reads NU-WRF netcdf files for checking I/O processes. You must make sure all of I/O processes works right. If there are problems, it could be due to wrong I/O information in the Configure_SDSU.F file.

2.4 Configuration of Running Satellite Simulator

Once you successfully run G-SDSU for initial I/O checking, you are ready to run G-SDSU for simulating satellite signals. The following sections provide brief instructions of running each simulator.

2.4.1 Running Microwave Simulator

Activate microwave simulator switch.

```
$simulator_switch
  micro  = .true.      ! microwave simulator switch; on when .true.
  radar  = .false.     ! radar simulator switch; on  when .true.
  visir  = .false.     ! visible/IR simulator switch; on when .true.
  lidar  = .false.     ! Lidar simulator switch; on when .true.
  broad  = .false.     ! Broad-band simulator switch; on when .true.
  sarta  = .false.     ! SARTA simulator switch; on when .true.
  GV     = .false.     ! GV simulator switch; on when .true.
$end
```

Options of microwave simulator are configured within

```
$micro_options
…………
…………
…………
…………
$end
```

There are two types of microwave simulator configuration.

a. Column Simulator (traditional): This option compute radiative transfer along the model column (normal to surface) or east-west slant-path (recommended) at every single NU-WRF grid with fixed sensor view angle, and simulated microwave Tb is convolved with simple convolution techniques that mimic Gaussian weighting. Users must provide following information (example of GPM Microwave Imager [GMI]).

```
slant_path_micro = .true. ! if .true., it account two-way slantpath for microwave RT in east-west
direction.


micro_sensor = 'GMI'   !sensor name (GPM Microwave Imager)
ground_micro = .false.    !=.true. for ground based; =.false. for satellite based
ona_angle_micro = 48.5      ! off-nadir angle [deg]
mxfreq_micro = 10                ! The number of microwave-radiometer channels
freq_micro = 10.65, 18.7, 23.8, 36.5, 89.0, 166., 176.31, 180.31, 186.31, 190.31  ! Channel
frequencies [GHz]
fov_ct_micro =  19.4, 11.2, 9.2, 8.6, 4.4, 4.4, 4.4, 4.4, 4.4, 4.4       ! cross-track FOV
fov_dt_micro =  32.2, 18.3, 15.0, 14.4, 7.3, 7.3, 7.3, 7.3, 7.3, 7.3     ! down-track FOV
beamwidth    =  1.75, 1.00, 0.90, 0.90, 0.40, 0.4, 0.4, 0.4, 0.4, 0.4
sma_micro    = 6776.14    ! semi-major axix of GPM Core satellite [km]
nf_L2        = 4          ! main frequency for L2 parameter
```

There are a number of different sensors specified in the configure file, which are masked out by “`!`” at the beginning of the row. Output is 2D microwave Tb data and associated geophysical parameters in the WRF-coordinate map.

b. Scanning Simulator (advanced): This option computes radiative transfer along satellite positions and sensor-pointing surface geolocations through rigorous orbit-scanning simulations. FOV-convolution is based on half-power beam-width angles with Gaussian gain functions including side lobe impact. Currently, only pre-registered satellites and sensors can be chosen from the configuration files. Any combination of the satellite type and sensor type technically works, but it should be consistent to the existing combination (e.g., TRMM_POST and TMI, DMSP_F17 and SSMIS, or Aqua and AMSR).

```
  scan_micro = .true.  ! If true, scan / orbit simulator will run. (logical)
                       ! If true, specify satellite/instrument name below.
                       ! if true, all of above column setups are ignored.

  satellite_micro = 'GPM'    ! Available satellite type (character*20)
                             ! GPM: GPM core satellite
                             ! TRMM_PRE: TRMM satellite before orbit boost
                             ! TRMM_POST: TRMM satellite after orbit boost
                             ! AQUA: Aqua satellite
                             ! DMSP_F16: DMSP satellite F16
                             ! DMSP_F17: DMSP satellite F17

  scan_type_micro = 'GMI_LF'  ! Available sensor scan type
  ! GMI_LF : GMI low-frequency channels  (10.65, 18.7, 23.8, 36.5, 89.0GHz)
  ! GMI_HF : GMI high-Frequency channels (166, 183pm1, 183pm3, 183pm6GHz)
  ! TMI_LF : TMI low-frequency channels  (10.65, 19.35, 21.3, 37.0GHz)
  ! TMI_HF : TMI high-Frequency channels (85.5GHz)
  ! AMSR_E_LF : AMSR-E low-freq channels  (6.925, 10.65, 18.7, 23.8, 36.5)
  ! AMSR_E_HF : AMSR-E high-freq channels (89.0 GHz)
  ! GMI_LF37 : just test purpose
  ! SSMIS : SSMIS imager channels (19.35, 37, 91.655, 150, 183pm3,
183pm6.6GHz)

 inpfile_overpass_micro  = 'overpass_satellite'  ! name of model-input-list
file  (character)
```

It is important to set up `inpfile_overpass_micro`. You must prepare this file similar to `inpfile`, but contains satellite overpassing information.

e.g.,

`> vi overpass_satellite`

```
wrfout_d02_2011-05-20_10:00:00    35.75    -97.75    A   ORBIT
```

1st column is the NU-WRF file name, consistent to `inpfile`.
2nd column is the satellite overpassing latitude (-90 ~ 90deg)
3rd column is the satellite overpassing longitude (-180 ~ 180deg)
4rh column is "A" or "D", which is ascending or descending overpass, respectively.

5th column is the satellite-overpassing tag, which may be obtained from the actual satellite L1B file or you can specify whatever you want.

Satellite overpassing latitude and longitude can be derived from the existing satellite orbit data (available from satellite L1B data products), or they can be any latitude and longitude for synthetic database as long as they are within the NU-WRF domain.

Once successfully run with the scan_microwave option, G-SDSU provides a set of outputs in your specified output directory `sdsu_dir_output`.

This GrADS binary output file (and control file) contains satellite overpass information in a global lat-lon map.

```
wrfout_d02_2011-05-20_10:00:00.FOV_GPM_GMI_LF_TEST.bin
wrfout_d02_2011-05-20_10:00:00.FOV_GPM_GMI_LF_TEST.ctl
```

This NetCDF output file contains partial-orbit simulated orbital and sensor geolocation information as well as simulated microwave Tb only over the NU-WRF domain. This is closest to the satellite L1B orbital data format.

```
wrfout_d02_2011-05-20_10:00:00.GMI_LF.ORBITAL.ORBIT_TEST.nc
```

This NetCDF output file contains full-orbit simulated orbital and sensor geolocation information.

```
wrfout_d02_2011-05-20_10:00:00.GMI_LF.GEOLOCATION.ORBIT_TEST.nc
```

This GrADS binary output file (and control file) contains microwave Tb and associated geophysical parameters just over the NU-WRF domain in a regional lat-lon (specified in `minlat`, `maxlat`, `minlon`, and `maxlon`) map.

```
wrfout_d02_2011-05-20_10:00:00.GMI_LF.ORBITAL.ORBIT_TEST.latlon.bin
wrfout_d02_2011-05-20_10:00:00.GMI_LF.ORBITAL.ORBIT_TEST.latlon.ctl
```

Another GrADS binary output file (and control file) contains footprint-imaged microwave Tb data over the NU-WRF domain in the WRF-coordinate map, which is useful for presentation purpose. (Tip: you must use GrADS with the “gxout grfill” option.)

```
wrfout_d02_2011-05-20_10:00:00.GMI_LF.ORBIT_TEST_fovimage.bin
wrfout_d02_2011-05-20_10:00:00.GMI_LF.ORBIT_TEST_fovimage.ctl
```

2.4.2 Running Radar Simulator

Activate the radar simulator switch.

```
$simulator_switch
  micro  = .false.     ! microwave simulator switch; on when .true.
  radar  = .true.     ! radar simulator switch; on  when .true.
  visir  = .false.     ! visible/IR simulator switch; on when .true.
  lidar  = .false.     ! Lidar simulator switch; on when .true.
  broad  = .false.     ! Broad-band simulator switch; on when .true.
  sarta  = .false.     ! SARTA simulator switch; on when .true.
  GV     = .false.      ! GV simulator switch; on when .true.
$end
```

Options of radar simulator can be configured within

```
$radar_options
…………
…………
…………
…………
$end
```

There are two types of radar simulator configuration.

a. Column Simulator (traditional): This option computes radiative transfer along the model column (normal to surface) at every single NU-WRF grid point with fixed sensor view angle, and simulated radar reflectivity and Doppler velocity are convolved with a simple convolution technique that mimics Gaussian weighting. Users must provide following parameters (example of GPM DPR).

```
radar_output_L2 = .F.     !if .true. it also output radar-range-sampled geophysical parameter.

radar_sensor = 'DPR'  !sensor name (Dual-frequency Precipitation Radar)
ground_radar = .false.    !=.true. for ground-based sensor; =.false. for satellite-based sensor
mxfreq_radar = 2              !The number of channels
min_echo = 17.                     !minimal_detactable echo [dBZ]
inc_angle_radar = 0.            !incident angle [deg]
k2 = -999.,-999.  !dielectric constant |k^2| defaults (if not known -> -999.)
freq_radar = 13.6, 35.5     !Channel frequencies [GHz]
fov_ct_radar =  5.0,5.0       ! Spatial resolution for cross-track FOV
fov_dt_radar =  5.0,5.0       ! Spatial resolution for down-track FOV
mxhgt_radar  = 19.0       ! maximum height of measurementit (above sea level) [km]
range_radar= 0.25    ! radar measurement range resolution [km]
```

There are a number of different sensors specified in the configure file, which are masked out by "`!`" at the beginning of the row. Output is 3D radar reflectivity and associated geophysical parameters in the WRF-coordinate map. If radar_output_L2 is specified, it dumps geophysical parameters sampled in radar range for diagnostics (but the output size tends to be very large.)

b. Scanning Simulator (advanced): This option computes radiative transfer along satellite positions and sensor-pointing surface geolocations through rigorous orbit-scanning simulations. FOV-convolution is based on half-power beam-width angles with Gaussian gain functions including side lobe impact. Currently, only pre-registered satellite and sensors can be chosen from the configuration files. Any combination of the satellite type and sensor type technically works, but it should be consistent with the existing combination (e.g., CLOUDSAT and CPR, TRMM_POST and PR, or GPM and DPR_Ku).

```
  scan_radar = .true.  ! If true, scan / orbit simulator will run.
                       ! Slow but more accurate geometry. (logical)
                       ! If true, specify satellite/instrument name below.
                       ! if true, all of above sensor setups are ignored.

  satellite_radar = 'GPM'    ! Available satellite type (character*20)
                             ! GPM: GPM core satellite (2013~)
                             ! TRMM_PRE: TRMM satellite before orbit boost (1998~~2002)
                             ! TRMM_POST: TRMM satelite after orbit boost (2002~2011)
                             ! CLOUDSAT: CloudSat satellite

  scan_type_radar = 'DPR_Ku'  ! Available sensor scan type
                              ! DPR_Ku : GPM Dual-Frequency Radar Ku band (13.6GHz)
                              ! DPR_Ka : GPM Dual Frequency Radar Ka band (35.5GHz)
                              ! PR     : TRMM Precipitation Radar (13.6GHz)
                              ! CPR    : CloudSat CPR (94GHz)

 inpfile_overpass_radar  = 'overpass_satellite'  ! name of model-input-list file  (character)
                          ! file must contains lat, lon, ascend/descend, orbit_TAG for each CRM
input
```

It is important to set up `inpfile_overpass_radar`. You must prepare this file similar structure to `inpfile_overpass_microwave` (Section 2.4.1).

Once successfully run with the scan_radar option, G-SDSU provides the following sets of output in your specified output directory `sdsu_dir_output`.

This GrADS binary output file (and control file) contains satellite overpass information in a global lat-lon map.

```
wrfout_d02_2011-05-20_10:00:00.FOV_GPM_DPR_Ku_TEST.bin
wrfout_d02_2011-05-20_10:00:00.FOV_GPM_DPR_Ku_TEST.ctl
```

This NetCDF file contains partial-orbit simulated sensor geolocation information as well as CPR reflectivity just over the NU-WRF domain. This is closest to the satellite L1B data format.

```
wrfout_d02_2011-05-20_10:00:00.DPR_Ku.ORBITAL.ORBIT_TEST.nc
```

This NetCDF file contains full-orbit simulated sensor geolocation information over the globe.

```
wrfout_d02_2011-05-20_10:00:00.DPR_Ku.GEOLOCATION.ORBIT_TEST.nc
```

This GrADS binary output file (and control file) contains partial-orbit reflectivity profiles and sensor geolocations just over the NU-WRF domain.

```
wrfout_d02_2011-05-20_10:00:00.DPR_Ku.ORBITAL.ORBIT_TEST.bin
wrfout_d02_2011-05-20_10:00:00.DPR_Ku.ORBITAL.ORBIT_TEST.ctl
```

2.4.3 Running Visible-IR Simulator

Activate visible-IR simulator switch.

```
$simulator_switch
  micro  = .false.     ! microwave simulator switch; on when .true.
  radar  = .false.     ! radar simulator switch; on  when .true.
  visir  = .true.     ! visible/IR simulator switch; on when .true.
  lidar  = .false.     ! Lidar simulator switch; on when .true.
  broad  = .false.     ! Broad-band simulator switch; on when .true.
  sarta  = .false.     ! SARTA simulator switch; on when .true.
  GV     = .false.      ! GV simulator switch; on when .true.
$end
```

Options of visible-IR simulator can be configured within

```
$visir_options
…………
…………
…………
…………
$end
```

There are two types of visible-IR simulator configuration.

a. Column Simulator (traditional): This option computes radiative transfer along the model column (normal to surface) at every single NU-WRF grid point with fixed sensor view angle, and simulated radiance and/or infrared Tb are convolved with a simple convolution technique that mimics Gaussian weighting. However, convolution may not be used, because visible-IR sensors tend to have higher resolution IFOV than NU-WRF grids. Users must provide following information (example of MODIS three IR channels). (Remember that the vis-IR simulator is the slowest among all simulators. Please be patient.)

```
  visir_sensor = 'MODIS_IR'   ! sensor name  (IR channel for T3EF) (character*20)
  znth_slr = 0.      ! solar zenith angle [deg] (if -999. coszen depends on model time.) (real)
  znth_obs = 12.13   ! viewing zenith angle [deg] (real)
  azmth    = 0.      ! azimuth angle between the sun and sensor [deg] (real)
  mxwavel = 3        ! The number of channels (real)
  wavel = 7.32, 11.e0, 12.0   ! Channel wavelengths  [micron] (real,dimension(mxwavel))
  fov_ct_visir =  5.0,5.0,5.0 ! Spatial resolution for cross-track FOV (adjusted for PR FOV)
(real,dimension(mxwavel))
  fov_dt_visir =  5.0,5.0,5.0 ! Spatial resolution for down-track FOV  (adjusted for PR FOV)
(real,dimension(mxwavel))
```

There are also a number of different sensors specified in the configure file, which is masked out by "`!`" at the beginning of the row. Output is 3D MODIS radiance/Tb and associated geophysical parameters in the WRF-coordinate map.

b. Scanning Simulator (advanced): This option computes radiative transfer along NU-WRF each grid column, but swath width and sensor viewing angle are specified for each grid through rigorous orbit-scanning simulations. Convolution is not used, because usually visible-IR sensors have higher resolution than the NU-WRF simulation. Because input parameters are complex, only pre-registered satellite and sensors can be chosen from the configuration files. So far, the list of sensors is very limited due to a lack of sensor mechanics information. If `scan_visir = .false.`, the traditional column simulator will run.

```
scan_visir = .true.  ! If true, scan / orbit simulator will run to account more accurate scan
geometry.(logical)
                      ! If true, specify satellite/instrument name below.
                      ! if true, all of above sensor setups are ignored.

satellite_visir = 'AQUA'      ! Available satellite type (character*20)
                              ! AQUA: Aqua satellite (2002 May ~)
                              ! TERRA: Terra satellite (2000 Jan ~ )
                              ! TRMM_PRE: TRMM satellite before orbit boost
                              ! TRMM_POST: TRMM satellite after orbit boost

scan_type_visir = 'MODIS_IR'  ! Available sensor scan type
                              ! MODIS_IR : MODIS sensor IR channel (11um )

inpfile_overpass_visir  = 'overpass_satellite'  ! name of model-input-list file  (character)
```

It is important to set up `inpfile_overpass_visir`. You must prepare this file with a similar structure to `inpfile_overpass_microwave` (Section 2.4.1).

After a successful run, G-SDSU provides the following set of outputs in your specified output directory `sdsu_dir_output`.

This GrADS binary output file (and control file) contains satellite overpass information in a global lat-lon map.

```
wrfout_d02_2011-05-20_10:00:00.FOV_AQUA_MODIS_IR_TEST.bin
wrfout_d02_2011-05-20_10:00:00.FOV_AQUA_MODIS_IR_TEST.ctl
```

This GrADS binary output file (and control file) contains IR Tb and associated geophysical parameters just over the NU-WRF domain on the WRF domain grid.

```
wrfout_d02_2011-05-20_10:00:00.MODIS_IR.ORBITAL.ORBIT_TEST.bin
wrfout_d02_2011-05-20_10:00:00.MODIS_IR.ORBITAL.ORBIT_TEST.ctl
```

This GrADS binary output file (and control file) contains simulated IR Tb just over the NU-WRF domain in regional lat-lon coordinates (specified in `minlat`, `maxlat`, `minlon`, and `maxlon`) map.

```
wrfout_d02_2011-05-20_10:00:00.MODIS_IR.ORBITAL.ORBIT_TEST.latlon.bin
wrfout_d02_2011-05-20_10:00:00.MODIS_IR.ORBITAL.ORBIT_TEST.latlon.ctl
```

2.4.4 Running Lidar Simulator

Activate the lidar simulator switch.

```
$simulator_switch
  micro  = .false.     ! microwave simulator switch; on when .true.
  radar  = .false.     ! radar simulator switch; on  when .true.
  visir  = .false.     ! visible/IR simulator switch; on when .true.
  lidar  = .true.     ! Lidar simulator switch; on when .true.
  broad  = .false.     ! Broad-band simulator switch; on when .true.
  sarta  = .false.     ! SARTA simulator switch; on when .true.
  GV     = .false.     ! GV simulator switch; on when .true.
$end
```

Options for the lidar simulator can be configured within

```
$lidar_options
…………
…………
…………
…………
$end
```

There are two types of lidar simulator configuration.

a. Column Simulator (traditional): This option computes radiative transfer along the model column (normal to surface) at every single NU-WRF grid point at specified range resolution. Convolution is not available, because lidar sensors usually have higher resolution than the NU-WRF grid. Users must provide following information (example of CALIOP).

```
  lidar_sensor = 'CALIOP'         ! sensor name in three character (character*20)
  ground_lidar = .false.          !=.true. for ground-based sensor; =.false. for satellite-based
sensor (logical)
  MS_Correct = 0.7                ! multiple scattering correction factor (real)
  mxwavel_lidar = 1               ! The number of channels (integer)
  wavel_lidar = 0.532             ! Channel wavelengths  [micron] (real,dimension(mxwavel_lidar))
  inst_profile_lidar = .true.     ! = .true. for instrument-defined profile (must define
mxhgt_lidar,range_lidar)
  mxhgt_lidar  =  20.0            ! maximum height of measuremenit (above sea level) [km]
  range_lidar  =  0.3             ! lidar measurement range resolution [km]
```

There are a number of different sensors specified in the configure file, which are masked out by "`!`" at the beginning of the row. Output is 3D lidar backscattering and associated geophysical parameters in the WRF-coordinate map.

b. Scanning Simulator (advanced): This option computes radiative transfer along each NU-WRF column at the satellite orbit track through rigorous orbit-scanning simulations. Thus, computational speed may be faster than traditional approach. So far, the list of sensors is very limited due to lack of sensor mechanics information. If `scan_lidar = .false.`, the traditional column simulator will be run.

```
scan_lidar = .true.  ! If true, scan / orbit simulator will run to account more accurate scan
geometry.(logical)
                       ! If true, specify satellite/instrument name below.
                       ! if true, all of above sensor setups are ignored.

satellite_lidar = 'CALIPSO'    ! Available satellite type (character*20)
                              ! CALIPSO: CALIPSO satellite

scan_type_lidar = 'CALIOP'  ! Available sensor scan type
                            ! CALIOP : CALIOP (532nm, 1064nm)

inpfile_overpass_lidar  = 'overpass_overpass'  ! name of model-input-list file  (character)
```

It is important to set up `inpfile_overpass_lidar`. You must prepare this file with a similar structure to `inpfile_overpass_microwave` (Section 2.4.1).

After a successful run, G-SDSU provides a following set of outputs in your specified output directory `sdsu_dir_output`.

This GrADS binary output file (and control file) contains satellite overpass information in a global lat-lon map.

```
wrfout_d02_2011-05-20_10:00:00.FOV_CALIPSO_CALIOP_TEST.bin
wrfout_d02_2011-05-20_10:00:00.FOV_CALIPSO_CALIOP_TEST.ctl
```

This GrADS binary file (and control file) contains lidar backscatter profiles and associated geophysical parameters just over the NU-WRF domain.

```
wrfout_d02_2011-05-20_10:00:00.CALIOP.ORBITAL.ORBIT_TEST.bin
wrfout_d02_2011-05-20_10:00:00.CALIOP.ORBITAL.ORBIT_TEST.ctl
```

This GrADS binary file (and control file) contains the backscatter-color ratio joint PDF diagram in GrADS format.

```
wrfout_d02_2011-05-20_10:00:00.CALIOP.Joint_PDF.ORBIT_TEST.bin
wrfout_d02_2011-05-20_10:00:00.CALIOP.Joint_PDF.ORBIT_TEST.ctl
```

2.4.5 Running Broadband Simulator

Activate the microwave simulator switch.

```
$simulator_switch
  micro   = .false.     ! microwave simulator switch; on when .true.
  radar   = .false.     ! radar simulator switch; on  when .true.
```

```
  visir  = .false.     ! visible/IR simulator switch; on when .true.
  lidar  = .false.     ! Lidar simulator switch; on when .true.
  broad  = .true.     ! Broad-band simulator switch; on when .true.
  sarta  = .false.     ! SARTA simulator switch; on when .true.
  GV     = .false.      ! GV simulator switch; on when .true.
$end
```

Options of broadband simulator are configured within

```
$broad_options
…………
…………
…………
…………
$end
```

There are two types of broadband simulator configuration.

a. Column Simulator (traditional): This option computes radiative transfer along the model column (normal to surface) at every single NU-WRF grid point with fixed sensor view angle, and simulated energy budgets and radiative heating rate profiles are convolved with a simple convolution technique that mimics Gaussian weighting. Users must provide the following information.

```
broad_scheme = 'goddard'   ! goddard - Goddard Radition (CliRad) scheme

heating_rate = .false. ! write out 3D broadband SW/LW heating rate [K/day]
                       ! in addition to the default energy budget output.

fov_ct_broad =  10.e0    ! Spatial resolution for cross-track FOV (CERES)
fov_dt_broad =  10.e0    ! Spatial resolution for down-track FOV  (CERES)
```

Output are 2D energy budgets at top-of-atmosphere, bottom-of-atmosphere, and atmosphere levels.

b. Scanning Simulator (advanced): Currently, this option is not available for the broadband simulator.

2.4.6 Running GV Simulator

The GV simulator is only designed for WRF-SBM, a special version of the WRF with HUCM SBM scheme, which has not yet been incorporated to NU-WRF. The following options can generate aircraft- or ground-based probe-observable microphysics parameters from the WRF-SBM simulations.

Activate the GV simulator switch.

```
$simulator_switch
  micro  = .false.     ! microwave simulator switch; on when .true.
```

```
  radar   = .false.      ! radar simulator switch; on  when .true.
  visir   = .false.      ! visible/IR simulator switch; on when .true.
  lidar   = .false.      ! Lidar simulator switch; on when .true.
  broad   = .false.      ! Broad-band simulator switch; on when .true.
  sarta   = .false.      ! SARTA simulator switch; on when .true.
  GV      = .true.       ! GV simulator switch; on when .true.
$end
```

Specify the following parameters:

```
particle_shape = 1  ! 0 - sphere
                    ! 1 - irregular (assumption from SnowFake and 2DVD measurements)

aircraft_on = .true.  ! true - simulate aircraft 2D-probe measurable parameters,
                        ! then dump output file (***.GV3D.bin), including         following
parameters.
                        ! bulk water content [g/m3]
                        ! bulk effective radius [micron]
                        ! bulk particle volume [cm3/m3]
                        ! bulk density [g/cm3]
                        ! liquid water fraction [-]

aircraft_ice = .true. ! if true, it only sample ice particle
                        ! if .false. it only sample liquid particle

parsivel_on = .true.  ! true - simulate ground-based Persivel measurable parameters
                        ! , then dump output (***.GV2D.bin), including following parameters
                        ! Parsivel 5min rainfall accumulation [mm]
                        ! Geonor 5min rainfall accumulation [mm]
                        ! bulk effective density [g/cm3]
                        ! bulk effective radius [micron]

parsivel_liq_cutoff = .true. !if True, Parsivel parameters are only accounted for diameter less
than 8mm.


  dump_psd = .false.   ! true - output full 33-bin PSD (***.GV3D_PSD.bin or ***.GV2D_PSD.bin).

  zonal_sampling_on = .false. ! true - sample Aircraft- or PARSIVEL-simulator parameters in
specific zone
                              ! and dump zonal statistical output (mean normalized PSDs)

                             ! if zonal_sampling_on is true, define sampling zone below
  min_lat = 0.e0             ! minimum latitude [deg] (Aircraf and Parsivel)
  max_lat = 90.e0            ! maximum latitude [deg] (Aircraf and Parsivel)
  min_lon = -180.e0          ! minimum longitude [deg]  (Aircraf and Parsivel)
  max_lon =  0.e0            ! maximum longitude [deg]  (Aircraf and Parsivel)
  min_alt = 0.e0             ! minimum altitude [km]
  max_alt = 5.e0             ! maximum altitude [km]
```

2.4.7 Running SARTA Simulator

The SARTA simulator is developed for simulating AIRS spectral radiance that accounts for AIRS channels specifications, developed at ASL, UMBC. This version of SARTA computes cloud-less sky radiance (does not account for cloud optical properties.)

Activate the SARTA simulator switch.

```
$simulator_switch
  micro   = .false.      ! microwave simulator switch; on when .true.
  radar   = .false.      ! radar simulator switch; on  when .true.
```

```
  visir  = .true.     ! visible/IR simulator switch; on when .true.
  lidar  = .false.     ! Lidar simulator switch; on when .true.
  broad  = .false.     ! Broad-band simulator switch; on when .true.
  sarta  = .false.     ! SARTA simulator switch; on when .true.
  GV     = .false.      ! GV simulator switch; on when .true.
$end
```

The following options specify either to simulate all (2784) channels or specific channels. Output units can be also chosen.

```
$sarta_options

  all_channel_airs = .true.    ! if true, it simulates default all AIRS (2784) channels (ichan
will be ignored).
                               ! if false, you must specify specific channels.

  nchan_airs  = 6         ! number of airs channels
  ichan_airs  = 1, 101, 501, 1001, 1501, 2001 ! specific airs channel ID numbers

  sol_zen_airs  = 0.e0   ! solar zenith angle [deg] (if -999. coszen depends on model time.)
  view_ang_airs = 0.0   ! veiwing zenith angle [deg] (should be between -49.6 and49.6)
  airs_unit = 'tb'       !'tb' gives output in brightness temperature.
                         !'rad' gives output in radiance.

$end
```

Output GrADS binary files contains spectral Tbs. (Note that the Z-coordinate in the GrADS control file is used for channel dimension.) The output does not contain geophysical parameters. For comparison between Tb and geophysical parameters (surface skin temperature and air temperature profiles), you may also dump CRM 3D and 2D outputs. E.g.,

```
 $io_options
.
.
.
  write_CRM3D   = .true.   ! if true, write out CRM 3D file in GrADS format
  write_CRM2D   = .true.   ! if true, write out CRM 2D file in GrADS format
.
.
.
.

$end
```

2.4.8 Additional Settings

With each simulator setting, Configure_SDSU.F has the following options for single-scattering properties. Default settings are recommended unless you really understand optical physics or want to investigate sensitivity/uncertainties of forward simulations. Blue-highlighted text is new addition in V3.5.1.

```
$single_scatter_options
```

```
  lut_micro    = .true.    ! Particle single-scattering LUT options for micro/radar simulator
(logical).
                           !.true.  : Use LUTs for microwave opt. Very Fast. (recommended)
                           !.false. : Full solution of Mie routine. Slow, but accurate.
                           ! This will be false for HUCM_SBM or HUCM_SBM43 microphysics.

  lut_visir    = .true.     ! Particle single-scattering LUT option for visir simulator (logical)
                            !.true.  : Use LUTs for microwave opt. Very fast. (recommended)
                            !.false. : Full solution of Mie routine Slow, but accurate.
                            ! This will  be .false. for HUCM_SBM or HUCM_SBM43 microphysics.

  lut_replace = .true.   ! Replace existing single-scattering LUT, if you modify single-
scattering routines (logical).
                          !.true.  : Replace single-scattering LUTs.
                          !.false. : Use existing Mie LUTs data.

  ss_opt_micro = 4        ! Single scattering calculation options for Microwave/Radar simulator
(integer)
                          ! If you change this option, you must set lut_replace = .true.
                          ! 1 - Mie      (flufy sphere)
                          ! 2 - SCATDB   (non-spherical ice crystals via DDSCAT: G. Liu @ FSU)
                          ! 3 - OpenSSP  (non-spherical ice crystals, snow aggregate via DDSCAT)
, not available
                          ! 4 - Tmatrix  (oblate/spheroid aggregate, garupel, hail via Tmatrix:
L. Liao) (recommended)

  scatdb_ice_type = 8     ! SCATDB ice crystal shape index (if ss_opt_micro = 2, you must choose)
                          ! 0-hexl, 1-hexs, 2-hexb, 3-hexf, 4-hexp, 5-ros3,
                          ! 6-ros4, 7-ros5, 8-ros6 (recommended), 9-sstr, 10-sden

  ice_refraction_func = 1 ! Effective refraction functions for frozen particles for
Microwave/Radar simulator (integer)
                          ! for Mie - maxwell-garnett combination (if ss_opt_micro == 1 or 2)
                          ! 1: Oblique Maxwell-Garnett function that assumes ice inclusion within
air matrix. (recommneded)
                          ! 2: Oblique Maxwell-Garnett function that assumes air inclusion within
ice matrix.
                          ! 3: Effective-Medium function that assumes homogeneous mixing.


  melt_opt = 0   ! Effective refraction functions for melting particles for Microwave/Radar
simulator (integer)
                 ! 0: Does not account melting particle
                 ! 1: Oblique Maxwell-Garnett function that assumes ice inclusion within water
matrix.
                 ! 2: Oblique Maxwell-Garnett function that assumes water inclusion within ice
matrix.
                 ! 3: Oblique Maxwell-Garnett function averaging option 1 and 2 (recommended)
                 ! 4: Effective-Medium function that assumes homogeneous mixing.

  ss_opt_visir = 1        ! Visible-IR Ice cyrytals single scattering calculation options
(integer)
                          ! , which affect visir simulator and lidar simulator
                          ! If you change this option, you must set lut_replace = .true.
                          ! 0 - Sphere    (Mie)
                          ! 1 - Ping Yang Non-spherical IceScattering database (recommended)

  land_emiss_micro =  2   ! Microwave land-surface emissivity scheme
                          ! 0: simple scheme
                          ! 1: NESDIS land emissivity model (V1)
                          ! 2: TELSEM emissivity database  (recommended)

  input_bf_emiss   = 'global_emis_inf10_monthFilled_MYD11C3.A2006182.nc'  !IR land-surface
emissvity database
                    ! file name for IR spectrum emissivity for visir simulator (char)
                    ! you must have these files under DATAFILE/FILLED_IR_EMISS/DATABASE/
                    ! if not, download it from http://cimss.ssec.wisc.edu/iremis/

$end
```

2.5 Known Issues and Limitations

- G-SDSU is designed for high-reoslution meteorological modeling with grid spacing less than ~5km. For coarse-resolution model, user must be very careful for interpret outputs.

- It should be able to run multiple simulators simultaneously. However, it requires more computational memory and often causes unusual crashes in the middle of the simulations.

- The orbital GPM DPR and LIDAR simulators require larger memory due to the size of orbital output. When you submit a MPI run via batch script on the Pleiades or Discover machines, it should be safe to request a half (or less) number of total CPU per node E.g., Haswell node has 24~28 CPUs total in one node. So you can specify 12 or less CPUs per node. Otherwise, the simulation may stop due to insufficient memory.

- CloudSat CPR and CALIPSO CALIOP simulations do not include the multiple-scattering effect. Users must be careful when investigating backscattering signals within optically thick targets.

3. Citations

Aires, F., Prigent, C., and Bernardo, F., Jimenez, C., Saunders, R., and Brunel, P,, (2011), A Tool to Estimate Land-Surface Emissivities at Microwave frequencies (TELSEM) for use in numerical weather prediction, Quarterly Journal of the Royal Meteorological Society, Volume 137, Issue 656, pages 690–699.

Aoyama, Y., and J. Nakano (1999), *RS/6000 SP: Practical MPI programming*, IBM Poughkeepsie, New York.

Chou, M.-D., and Suarez, M. J. (1999), A solar radiation parameterization for atmospheric studies, *NASA Tech. Memo*, *104606*40.

Chou, M.-D., Suarez, M. J., Liang, X.-Z., and Yan, M. M.-H. (2001), A thermal infrared radiation parameterization for atmospheric studies, *NASA Tech. Memo*, *19*56.

Iguchi, T., Matsui, T., Shi, J. J., Tao, W., Khain, A. P., Hou, A., Cifelli, R., Heymsfield, A., and Tokay, A. (2012a), Numerical analysis using WRF-SBM for the cloud microphysical structures in the C3VP field campaign: Impacts of supercooled droplets and resultant riming on snow microphysics, *Journal of Geophysical Research: Atmospheres (1984–2012)*, *117*(D23).

Iguchi, T., Matsui, T., Tokay, A., Kollias, P., and Tao, W. (2012b), Two distinct modes in one-day rainfall event during MC3E field campaign: Analyses of disdrometer observations and WRF-SBM simulation, *Geophysical Research Letters*, *39*(24).

Kidd, C., T. Matsui, J. Chern, K. Mohr, C. Kummerow, and D. Randall: Physically-based precipitation retrievals from cross-track passive microwave sensors for the Global Precipitation Measurement mission. *Journal of Hydrometeorology*, (submitted).

Kummerow, C. (1993), On the accuracy of the Eddington Approximation for radiative transfer in the microwave frequencies, *J. Geophys. Res.*, *98*2757-2765.

Li, X., Tao, W.-K., Matsui, T., Liu, C., and Masunaga, H. (2010), Improving a spectral bin microphysical scheme using long-term TRMM satellite observations, *Quart. J. Roy. Meteor. Soc*, *136*382-399.

Liao, L. and R. Meneghini, 2013: Examination of Effective Dielectric Constants of Nonspherical Mixed-Phase Hydrometeors. *J. Appl. Meteor. Climatol.*, **52**, 197–212. doi: http://dx.doi.org/10.1175/JAMC-D-11-0244.1

Liu, G. (2008), A database of microwave single-scattering properties for nonspherical ice particles, *Bulletin of the American Meteorological Society*, *89*(10), 1563-1570.

Masunaga, H., and Kummerow, C. (2005), Combined radar and radiometer analysis of precipitation profiles for a parametric retrieval algorithm, *J. Atmos. Ocean. Tech.*, *22*909-929.

Masunaga, H., Matsui, T., Tao, W.-K., Hou, A. Y., Kummerow, C. D., Nakajima, T., Bauer, P., Olson, W. S., Sekiguchi, M., and Nakajima, T. Y. (2010), Satellite data simulator unit: A multisensor, multispectral satellite simulator package, *Bulletin of the American Meteorological Society*, *91*(12), 1625-1632.

Matsui, T. T. Iguchi, X. Li, M. Han, W.-K. Tao**,** W. Petersen, T. L'Ecuyer**,** R. Meneghini, W. Olson, C. D. Kummerow, A. Y. Hou, M. R. Schwaller, E. F. Stocker, J. Kwiatkowski (2013), GPM satellite simulator over ground validation sites, *Bull. Amer. Meteor. Soc.*, **94**, 1653–1660. doi: http://dx.doi.org/10.1175/BAMS-D-12-00160.1

Matsui, T. (2013), Chapter 12. Mesoscale Modeling and Satellite Simulator, *Mesoscale Meteorological Modeling. 3rd Edition*, R. A. Pielke Sr. Ed. *Academic Press*, 760 p, ISBN: 9780123852373.

Matsui, T., J. Santanello, J. J. Shi, W.-K. Tao, D. Wu, C. Peters-Lidard, E. Kemp, M. Chin, D. Starr, M. Sekiguchi, and F. Aires, (2014): Introducing multisensor satellite radiance-based evaluation for regional Earth System modeling, *Journal of Geophysical Research*, *119, 8450–8475, doi:10.1002/2013JD021424.*

Matsui, T., Zeng, X., Tao, W.-K., Masunaga, H., Olson, W. S., and Lang, S. (2009), Evaluation of long-term cloud-resolving model simulations using satellite radiance observations and multifrequency satellite simulators, *Journal of Atmospheric and Oceanic Technology*, *26*(7), 1261-1274.

Meneghini, R., and Kozu, T. (1990), Spaceborne weather radar, *Norwood, MA, Artech House, 1990, 208 p., 1*.

Nakajima, T., and Tanaka, M. (1986), Matrix formulations for the transfer of solar radiation in a plane-parallel scattering atmosphere, *Journal of Quantitative Spectroscopy and Radiative Transfer*, *35*(1), 13-21.

Nakajima, T., and Tanaka, M. (1988), Algorithms for radiative intensity calculations in moderately thick atmospheres using a truncation approximation, *Journal of Quantitative Spectroscopy and Radiative Transfer*, *40*(1), 51-69.

Olson, W. S., Kummerow, C. D., Yang, S., Petty, G. W., Tao, W.-K., Bell, T. L., Braun, S. A., Wang, Y., Lang, S. E., and Johnson, D. E. (2006), Precipitation and latent heating distributions from satellite passive microwave radiometry. Part I: Improved method and uncertainties, *Journal of applied meteorology and climatology*, *45*(5), 702-720.

Platt, C. (1973), Lidar and radioinetric observations of cirrus clouds, *Journal of the atmospheric sciences*, *30*(6), 1191-1204.

Rosenkranz, P. W. (1993), Absorption of microwaves by atmospheric gases, *Atmospheric remote sensing by microwave radiometry(A 95-14701 02-46), New York, NY, John Wiley & Sons, Inc., 1993,* 37-90.

Seemann, W. S., E. E. Borbas, R. O. Knuteson, G. R. Stephen, and H.-L. Huang, 2008: Development of a Global Infrared Land Surface Emissivity Database for Application to Clear Sky Sounding Retrievals from Multispectral Satellite Radiance Measurements. *J. Appl. Meteor. Climatol.*, **47**, 108–123. doi: http://dx.doi.org/10.1175/2007JAMC1590.1

Sekiguchi, M., and Nakajima, T. (2008), A k-distribution-based radiation code and its computational optimization for an atmospheric general circulation model, *Journal of Quantitative Spectroscopy and Radiative Transfer*, *109*(17), 2779-2793.

Strow, L. S. E. Hannon, S. D. Souza-Machado, H. E. Motteler, and D. Tobin (2003), An Overview of the AIRS Radiative Transfer Model, *IEEE Transactions On Geoscience And Remote Sensing*, **41**(2), 2003

Thompson G., P. R. Field, R. M. Rasmussen, and W. D. Hall (2008), Explicit forecasts of winter precipitation using an improved bulk microphysics scheme. Part II: Implementation of a new snow parameterization, Mon. Wea. Rev., 136, 5095–5115.

Yang, P., L. Bi, B.A. Baum, K.-N. Liou, G.W. Kattawar, M.I. Mishchenko, and B. Cole (2013) Spectrally consistent scattering, absorption, and polarization properties of atmospheric ice crystals at wavelengths from 0.2 to 100 µm. *J. Atmos. Sci.*, **70**, 330-347, doi:10.1175/JAS-D-12-039.1.

Weng, F., Yan, B., and Grody, N. C. (2001), A microwave land emissivity model, *Journal of Geophysical Research: Atmospheres (1984–2012)*, *106*(D17), 20115-20123.

Wilheit, T. T. (1979), A model for the microwave emissivity of the ocean's surface as a function of wind speed, *Geoscience Electronics, IEEE Transactions on*, *17*(4), 244-249.

Zhang, S. Q., Zupanski, M., Hou, A. Y., Lin, X., and Cheung, S. H. (2013), Assimilation of precipitation-affected radiances in a cloud-resolving WRF ensemble data assimilation system, *Monthly Weather Review*, *141*(2), 754-772.

Zupanski, D., Zhang, S. Q., Zupanski, M., Hou, A. Y., and Cheung, S. H. (2011), A prototype WRF-based ensemble data assimilation system for dynamically downscaling satellite precipitation observations, *Journal of Hydrometeorology*, *12*(1), 118-134.