FV3-LAM / SRW App

(FV3-Limited Area Model / Short Range Weather Application)

- Guide for Master/Develop Branch -

Chan-Hoo Jeon, Ph.D.

Scientific Programmer/Analyst

Engineering and Implementation Branch (EIB)

NOAA/NWS/NCEP/EMC

March 16, 2021

Contents

.

Chapter 1

Quick Start Guide of SRW App

1.1 Stable Tag of Regional Workflow for EMC

1. Retireve the 'master' branch of the SRW App from the GitHub repository :

```
git clone -b master https://github.com/ufs-community/ufs-srweather-app
cd ufs-srweather-app
```

2. Specify the stable hashes of the external components in 'External.cgf':

```
vim Externals.cfg
(check and modify hashes)
```

where the latest five hashes of the external components are as follows:

Component	Hash				
	03/12/21	03/05/21	01/20/21	-	-
regional_workflow	5315e22	3586e11	6f0e868	-	-
ufs_utils	005f9a0a	005f9a0a	005f9a0a	-	-
ufs_weather_model	ea8a7aa	ea8a7aa	63591b6	-	-
EMC_post	9fa1e088	9fa1e088	9fa1e088	-	-

Table 1.1: Hashes of the external components

3. Check out the submodules for the SRW App:

```
(module load python/3.6.3 on WCOSS Dell/Cray)
(module load python/3.7.5 on Orion)
./manage_externals/checkout_externals
```

4. Set up the build environment:

```
source env/build_[machine]_[compiler].env
```

   where the '[machine]' is 'wcoss_dell_p3', 'wcoss_cray', 'hera', or 'orion', and the '[compiler]' is 'intel'.

5. Build executables:

```
mkdir build
cd build
cmake .. -DCMAKE_INSTALL_PREFIX=..
make -j 8  >& build.out &
```

6. Set up experiment configuration:

```
cd ../regional_workflow/ush
cp config.community.sh config.sh  (or, cp config.nco.sh config.sh)
# Edit as needed
```

7. Load *Python* environment for the workflow:

```
source ../../env/wflow_[machine].env
```

   where the '[machine]' is 'wcoss_dell_p3', 'wcoss_cray', 'hera', or 'orion'.

8. Generate the workflow:

```
./generate_FV3LAM_wflow.sh
```

9. Run the workflow:

```
cd ../../../expt_dirs/{EXPTDIR: experiment directory}]
./launch_FV3LAM_wflow.sh
```

For automatic re-submission of the workflow using 'cron':

(a) Automatically:

Set the parameters in the configuration file 'config.sh' (Step 6) as follows:

```
USE_CRON_TO_RELAUNCH="TRUE"
CRON_RELAUNCH_INTVL_MNTS="5"
```

Note) If you set the above parameters in 'config.sh', you will not need to run Step 9 separately. The 'cron' will run the workflow automatically.

(b) Manually:

- WCOSS Dell:

```
vim /u/$USER/cron/mycrontab
# put command lines:
*/5 * * * * cd [path to 'expt_dirs']/test_CONUS_25km_GFSv15p2 && ./
    launch_FV3LAM_wflow.sh
```

- Hera, WCOSS Cray, or Orion:

```
crontab -e
# put command lines:
*/5 * * * * cd [path to 'expt_dirs']/test_CONUS_25km_GFSv15p2 && ./
    launch_FV3LAM_wflow.sh
```

1.2 Release Branch: release/public-v1

1. Retireve the 'release/public-v1' branch of the SRW App from the GitHub repository :

```
git clone -b release/public-v1 https://github.com/ufs-community/ufs-
    srweather-app
```

2. Check out the submodules for SRW App:

```
cd ufs-srweather-app
(module load python/3.6.3 on WCOSS Dell/Cray)
(module load python/3.7.5 on Orion)
./manage_externals/checkout_externals
```

3. Set up the build environment:

```
source env/build_[machine]_[compiler].env
```

   where the '[machine]' is 'hera', 'wcoss_dell_p3', 'wcoss_cray', 'orion', or 'cheyenne', and the '[compiler]' is 'intel' or 'gnu'.

4. Build executables:

```
mkdir build
cd build
cmake .. -DCMAKE_INSTALL_PREFIX=..
make -j 8  >& build.out &
```

5. Set up experiment configuration:

```
cd ../regional_workflow/ush
cp config.community.sh config.sh  (or, cp config.nco.sh config.sh)
# Edit as needed
```

6. Load *Python* environment for the workflow:

```
source ../../env/wflow_[machine].env
```

   where the '[machine]' is 'hera', 'wcoss_dell_p3', 'wcoss_cray', 'orion', 'jet', or 'cheyenne'.

7. Generate the workflow:

```
./generate_FV3LAM_wflow.sh
```

8. Run the workflow:

```
cd ../../../expt_dirs/{EXPTDIR: experiment directory}
./launch_FV3LAM_wflow.sh
```

For automatic re-submission of the workflow using 'cron':

- WCOSS Dell: **NOT** working correctly (not finishing with SUCCESS)
- Hera, WCOSS Cray, or Orion:

```
crontab -e
# put command lines:
*/3 * * * * cd [path to 'expt_dirs']/test_CONUS_25km_GFSv15p2 && ./
    launch_FV3LAM_wflow.sh
```

or, set the parameters in the configuration file 'config.sh' (Step 5) as follows:

```
USE_CRON_TO_RELAUNCH="TRUE"
CRON_RELAUNCH_INTVL_MNTS="3"
```

Chapter 2

Pre-workflow with UFS Short Range Weather Application

2.1 Overall Procedure of SRW App

The UFS Short-Range Weather Application (SRW App) is an umbrella repository that contains the tool to check out external components required for the regional workflow system. Once the build process is complete, all the files and executables necessary for a regional workflow experiment are located in its sub-directories. Users can utilize the pre-defined domains or build their own domain. In either case, users must create/modify the case-specific and/or grid-specific configuration files. The overall procedure of the regional workflow in SRW App is illustrated in Figure 2.1. Its steps are as follows:

1. Clone the SRW App from GitHub.
2. Check out the external components.
3. Build the regional workflow system using *cmake* / *make*.
4. Check the grid-specific configuration file 'set_predef_grid_param.sh'.
5. Modify the case-specific configuration file 'config.sh'.
6. Load the appropriate *python* environment for the regional workflow
7. Generate a regional workflow experiment.

8. Run the regional workflow repeatedly as needed.

Each step will be described in detail in the following sections.

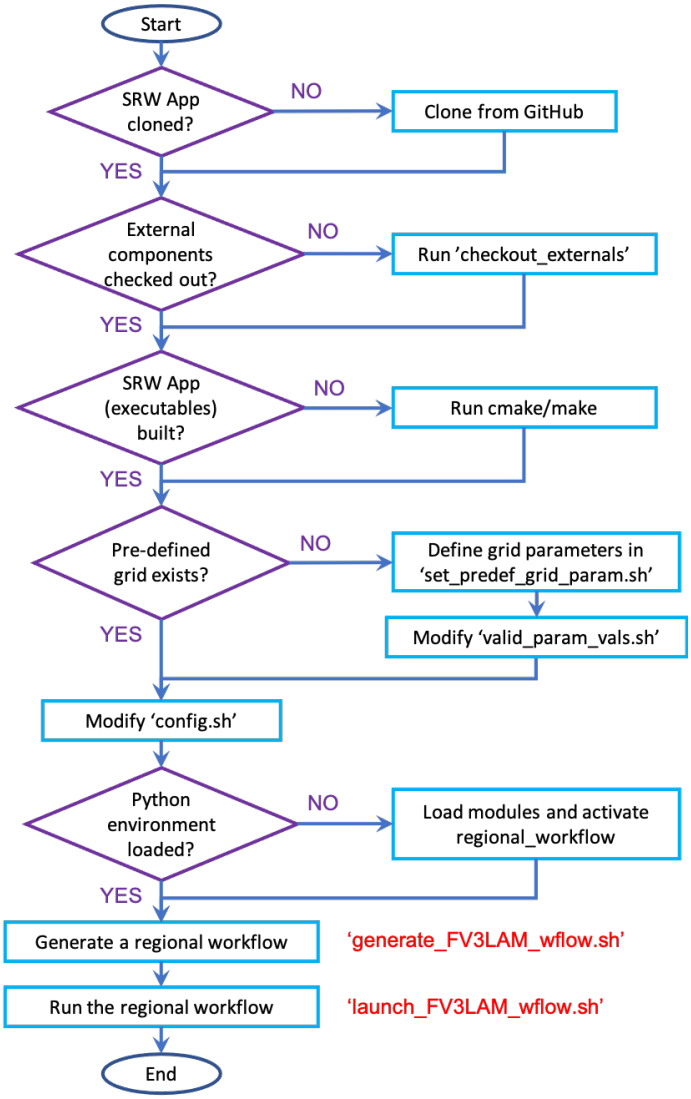

Figure 2.1: Overall procedure of Short Range Weather App

2.2 Download from GitHub

Retrieve the 'UFS Short Range Weather Application (SRWeather App)' repository from the specific GitHub repository to use the stable version of the regional workflow as follows:

```
git clone -b master https://github.com/ufs-community/ufs-srweather-app
```

Note) You should load the 'git' module to clone a GitHub repository on WCOSS-Cray:

```
module use -a /usrx/local/dev/modulefiles
module load git/2.18.0
```

The cloned repository contains the following sub-repositories and files:

Directory/file name	Description
docs	Release notes, documentation, users' guide
env	Machine-specific environment for set-up and workflow
manage_externals	Method for checking out external components
src	Build scripts are initially located, and the external components will be cloned in this directory.
CMakeLists.txt	Main *cmake* file for UFS Short Range Weather Application
Externals.cfg	GitHub repositories/branches for the external components
LICENSE.md	(empty)
README.md	Quick users' guide
ufs_srweather_app_meta.h.in	Meta information for UFS Short Range Weather Application which can be used by other packages
ufs_srweather_app.settings.in	UFS Short Range Weather App configuration summary
regional_workflow	Regional workflow (This directory will be created by running 'checkout_externals' in Section 2.3).
bin	Executables for workflow (This directory will be created by running 'cmake/make' in Section 2.4.2.)
include, lib, share	Modules, library, and input files for post-processing (This directory will be created by running 'cmake/make' in Section 2.4.2.)

Table 2.1: Sub-directories of the short range weather application

2.3 External Components

For the purpose of maintenance on WCOSS, the stable tag of the regional workflow for EMC utilizes the specific hash numbers for the external components as shown in Table 1.1 rather than branch names.

```
cd {HOME}
vim Externals.cfg
(Update the hash numbers to the latest)
```

Check out the sub-modules such as regional_workflow, ufs_weather_model, ufs_utils, and emc_post for UFS Short Range Weather Application:

```
./manage_externals/checkout_externals
```

where '{HOME}' is the path to the 'ufs-srweather-app' repository. For example, it can be '/scratch2/NCEPDEV/fv3-cam/{LOGNAME}/ufs-srweather-app' on Hera.

Note)

- It should be run from the {HOME} directory where the 'Externals.cfg' file exists to avoid the root error saying 'Model description file does not exist at path'.
- On WCOSS-Cray, you should load Python correctly:

```
module load python/3.6.3.  (2.7.xx or higher should be loaded)
```

This step will use the configuration file 'Externals.cfg' in the '{HOME}' directory to clone the specific branches (or hashes) of the external components as listed in Table 2.2. As a result, the regional workflow are cloned in the {HOME} directory while the external components are cloned into the target sub-directories in the '{HOME}/src' directory as shown in Figure 2.2. The directories and sub-directories that will be created later by running the scripts are presented in parentheses.

Component	GitHub repository (branch)	Target sub-directory
regional_workflow	NOAA-EMC/regional_workflow (develop)	regional_workflow
ufs_utils	NOAA-EMC/UFS_UTILS (develop)	src/UFS_UTILS

ufs_weather_model	ufs-community/ufs-weather-model (develop)	src/ufs_weather_model
EMC_post	NOAA-EMC/EMC_post (develop)	src/EMC_post

Table 2.2: GitHub repository for the external components

Notes)

- The target sub-directories are not shown in the {HOME} or '{HOME}/src' directory until the 'checkout_externals' is executed.
- The source codes of the executables such as 'chgres_cube', 'global_equiv_resol', 'orog', and 'shave' are located in '{HOME}/src/UFS_UTILS/sorc/' ('ufs_utils' component).

Workflow with Another Hash or HEAD

If you want to run the workflow with another hash or with the latest version (HEAD) of a given repository/branch (e.g. stable version/tag of the regional workflow), you will have to modify the 'Externals.cfg' file before issuing the above './checkout_externals' command.

1. Check the current hash or HEAD of the external component in 'Externals.cfg':

```
[ufs_utils]
protocol = git
repo_url = https://github.com/NOAA-EMC/regional_workflow
# Specify either a branch name or a hash but not both.
#branch = develop
hash = 6891c8a
local_path = regional_workflow
required = True
```

   Note that the 'branch' property in the above block (for 'regional_workflow') is commented out (using the # character), and instead the 'hash' property is specified.

2. To have 'manage_externals' obtain a different hash, simply specify a different hash in the line 'hash=6891c8a' in the block above.

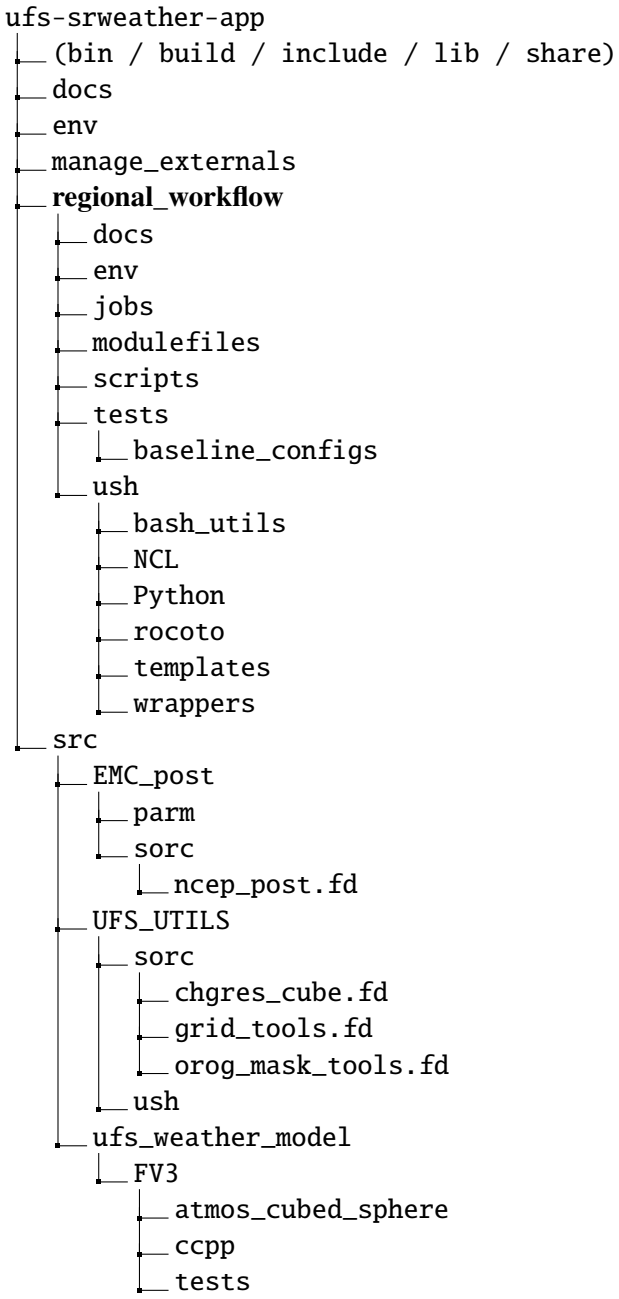

Figure 2.2: Structure of UFS Short Range Weather App

3. To obtain the HEAD of the 'develop' branch, uncomment the line specifying the branch and

comment out the line specifying the hash as follows:

```
branch = develop
#hash = 6891c8a
```

The regional workflow contains the sub-directories and files as shown in Table 2.3:

```
cd {HOMErrfs}
```

where '{HOMErrfs}' is the path to the 'regional_workflow' repository. For example, it can be 'scratch2/NCEPDEV/fv3-cam/{LOGNAME}/ufs-srweather-app/regional_workflow' on Hera.

Directory/file name	Description
docs	Release notes, documentation, users' guides
env	Machine-specified environment variables
jobs	J-job scripts which are launched by *Rocoto*
modulefiles	Module files required for running the model
scripts	Run scripts launched by the J-jobs
tests	Baseline experiment configurations
ush	Utility scripts called by various run scripts
environment.yml	
README.md	Quick user's guide to run a FV3-LAM workflow
update_fork.pl	

Table 2.3: Initial sub-directories and files of the regional workflow

2.4 Building the Executables for Workflow

2.4.1 Environment Set-up

Load the required modules and set the environment variables for 'BASH' users in the 'build_env_[machine]_[compiler].sh' file as:

```
cd {HOME}/env/
source build_[machine]_[compiler].env
module list
```

where '[machine]' is 'hera', 'wcoss_dell_p3', 'wcoss_cray', or 'orion', and '[compiler]' is 'intel'.

Notes)

- The above file ('build_[machine]_[compiler].env') is sourced from the script '{HOMErrfs}/ush/load_modules_run_task.sh' when a task is launched. Therefore, if you want to change the file name and its path, you should modify the script (lines 156–157):

```
155 machine=${MACHINE,,}
156 env_fn="build_${machine}_${COMPILER}.env"
157 env_fp="${SR_WX_APP_TOP_DIR}/env/${env_fn}"
```

- If you use 'TCSH' on Hera, you should change the environment variables in the last four lines as follows:

```
setenv CMAKE_C_COMPILER mpiicc
setenv CMAKE_CXX_COMPILER mpiicpc
setenv CMAKE_Fortran_COMPILER mpiifort
setenv CMAKE_Platform hera.intel
```

2.4.2 Building the Executables

Build a regional workflow system including the preprocessing utilities, forecast model, and post-processor:

```
cd {HOME}
mkdir build
cd build

cmake .. -DCMAKE_INSTALL_PREFIX=..
make -j 8 >& build.out &
```

where '-DCMAKE_INSTALL_PREFIX' specifies the location where the bin, include, lib, and share directories containing various components of SRW App will be created, and its recommended value '..' denotes one directory up from the build directory. In the next line for the *make* call, '-j 8' means the parallel run with 8 threads.

When the build process completes, there should exist the forecast model executable 'ufs_model' and eleven pre- and post-processing executables in '{HOME}/bin/', as shown in Table 2.4:

Name	Description	Detail
chgres_cube	Creates IC/LBC	Section 7.3
filter_topo	Filters the orography	Section 6.4
global_equiv_resol	Computes the global equivalent resolution for ESG grids	Section 5.1.3
make_hgrid	Creates the grid files which contain geo-referencing records for the model grid including global uniform and GFDL stand-alone regional grids	Section 6.2
make_solo_mosaic	Creates the mosaic files	Sections 3.4.1 & 6.2
ncep_post	Post-processing	Section 8.1
ufs_model	UFS weather model executable	Chapter 9
orog	Computes land mask, orography and gravity wave drag	Section 6.3
regional_esg_grid	Creates the stand-alone ESG grid files	Section 5.1.4
sfc_climo_gen	Creates surface climatological fields such as soil type	Sections 5.3 & 6.6
shave	Shaves the excess halo rows down to what is required for the LBCs in the orography and grid files	Section 6.5
vcoord_gen	Generate hybrid coordinate interface profiles	-

Table 2.4: List of executables.

2.5 Generating a Regional Workflow Experiment

2.5.1 Steps to Generate a New Regional Workflow Experiment

First, check if a pre-defined grid exists in the grid-specific configuration file (see the detail in Sections 2.5.2 and 2.5.3):

```
cd {HOMErrfs}/ush
vim set_predef_grid_param.sh
vim valid_param_vals.sh
```

Second, create a user-specified experiment configuration file ('{HOMErrfs}/ush/config.sh') and specify the values of your experiment parameters in it (see the detail in Sections 2.5.5 and 2.5.6):

```
cd {HOMErrfs}/ush
cp config.[option].sh config.sh
vim config.sh
# Edit parameters as needed.
```

where 'config.[option].sh' is described in Section 2.5.6. For users' convenience, two example configuration files are provided in '{HOMErrfs}/ush/': 'config.community.sh' and 'config.nco.sh'. The first is an example for creating and running an experiment in the 'community' mode (i.e. with the parameter 'RUN_ENVIR' set to "community"), while the second is an example of creating and running an experiment in the 'NCO' (operational) mode (i.e. with 'RUN_ENVIR' set to "nco").

Third, check templates of the input files and modify them as needed (Section 2.6).

Fourth, modify the 'set_cycle_dates.sh' script if you want to change the frequency of cycles. The frequency is set to one day in the script, and it is not able to be adjusted in the 'config.sh' file.

```
cd {HOMErrfs}/ush
vim set_cycle_dates.sh
```

Change the number of '# days' (default: 1) on Line 109 as follows:

```
105  all_cdates=()
106  date_crnt="${date_start}"
107  while [ "${date_crnt}" -le "${date_end}" ]; do
108    all_cdates+=( $( printf "%s " ${cycle_hrs[@]/#/${date_crnt}} ) )
109    date_crnt=$( date -d "${date_crnt} + 1 days" +%Y%m%d )
110  done
```

Fifth, check additional steps only for the '**nco**' mode:

1. Check if the FIX files necessary for running the 'nco' mode exist in the designated directory, which is specified in 'config.sh', as described in Section 2.5.7.

2. Create mosaic halo files ('mosaic.halo[3,4,6].nc'):

   If the mosaic halo files named 'mosaic.halo[3,4,6].nc' do not exist in the above designated directory, you should create them as described in Section 3.4.1.

Sixth, load the appropriate *Python* environment for the workflow. The workflow requires *Python3* with the packages 'PyYAML', 'Jinja2', and 'f90nml' available.

```
cd {HOME}/env/
source wflow_[machine].env
```

This file includes machine-specific *Python* environment as follows:

- On Hera:

```
module use -a /contrib/miniconda3/modulefiles
module load miniconda3
conda activate regional_workflow
```

- On WCOSS_dell_p3:

```
module load python/3.6.3
module use /usrx/local/nceplibs/dev/modulefiles
module load srw-app-python/1.0.0
```

- On WCOSS_cray:

```
module unload python/2.7.14
module load python/3.6.3
module use /usrx/local/nceplibs/modulefiles
module load srw-app-python/1.0.0
```

- On Orion:

```
module use -a /apps/contrib/miniconda3-noaa-gsl/modulefiles
module load miniconda3
conda activate regional_workflow
```

Seventh, generate the workflow:

```
cd {HOMErrfs}/ush
./generate_FV3LAM_wflow.sh
```

The flowchart for generating a new regional workflow is shown in Figure 2.3.

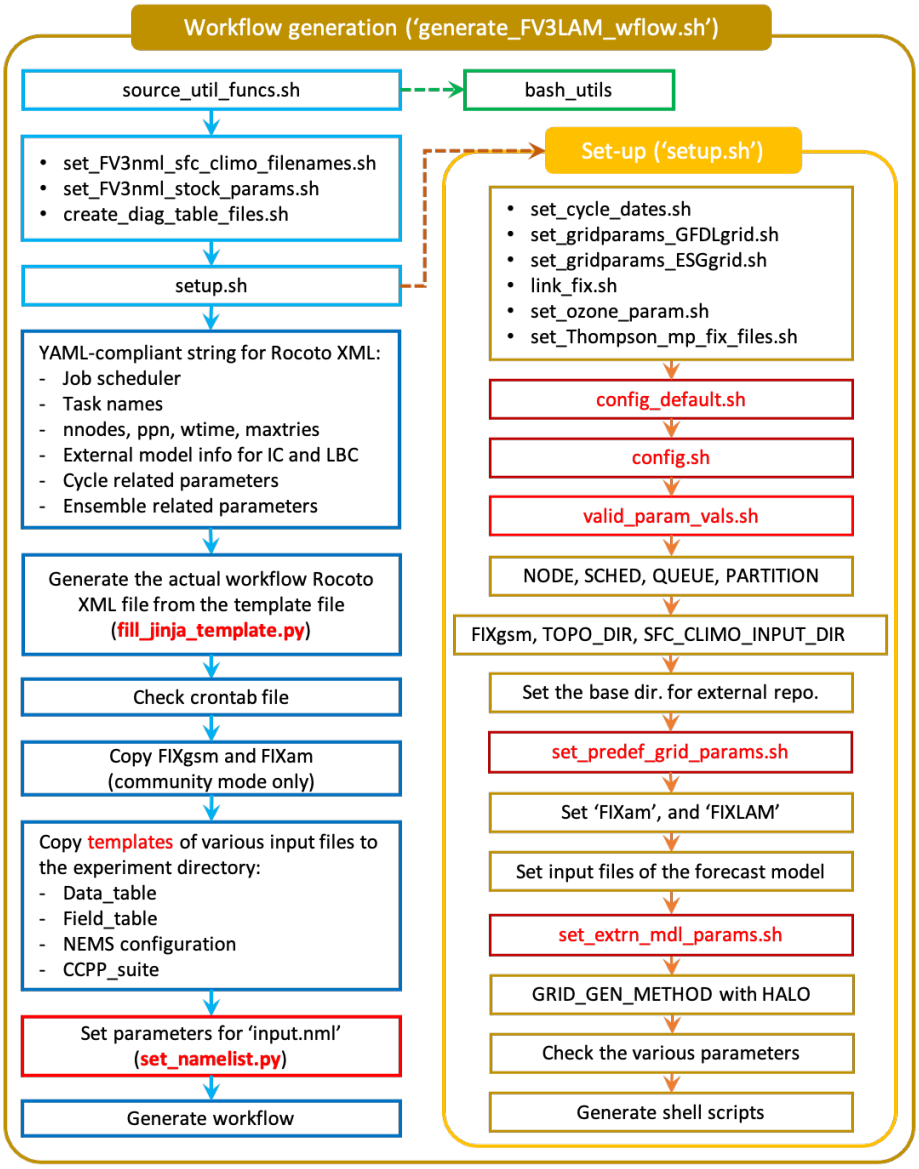

Figure 2.3: Structure of workflow generation

- The default configuration file ('{HOMErrfs}/ush/config_default.sh') assigns default values to the experiment parameters. Some of these default values are intentionally invalid in order to force the user to assign them valid values in the user-specified configuration file. There is usually no need for a user to modify the default configuration file. Note that the default

configuration file also contains documentation describing the experiment parameters.

- Parameter settings in the user-specified configuration file ('{HOMErrfs}/ush/config.sh') will override settings in the default configuration file because the user-specified configuration file is sourced after the default one.

- As shown in the flowchart, the grid(domain)-specific parameters such as the GFDL grid refinement or ESG (JP) grid, write-component parameters when 'QUILTING'="TRUE", 'DT_ATMOS', 'LAYOUT_X(Y)', and 'BLOCKSIZE', are separately set in '{HOMErrfs} /ush/set_predef_grid_params.sh'. If you want to use other regional domains that are not defined in this file, you should modify this file.

2.5.2 Pre-defined Grid Parameters: 'set_predef_grid_params.sh'

This file defines grid parameters for the specific pre-defined grids. The list of the pre-defined grids can be found in Table 2.8 ('PREDEF_GRID_NAME'). The grid parameters, defined in this file, are as follows:

1. Pre-defined grids:

No.	Grid name	Description	Method(s)
1	RRFS_CONUS_25km	GSD's 25*km* CONUS domain	ESGgrid
2	RRFS_CONUS_13km	GSD's 13*km* CONUS domain	ESGgrid
3	RRFS_CONUS_3km	GSD's 3*km* CONUS domain	ESGgrid
4	RRFS_SUBCONUS_3km	GSD's 3*km* sub-CONUS domain	ESGgrid
5	RRFS_AK_13km	GSD's 13*km* HRRR Alaska grid	ESGgrid
6	RRFS_AK_3km	GSD's 3*km* HRRR Alaska grid	ESGgrid
7	CONUS_25km_GFDLgrid	GFDL 25*km* CONUS domain	GFDLgrid
8	CONUS_3km_GFDLgrid	GFDL 3*km* CONUS domain	GFDLgrid
9	EMC_AK	EMC's 3*km* Alaska grid	ESGgrid
10	EMC_HI	EMC's 3*km* Hawaii grid	ESGgrid
11	EMC_PR	EMC's 3*km* Puerto Rico grid	ESGgrid
12	EMC_GU	EMC's 3*km* Guam grid	ESGgrid
13	GSL_HAFSV0.A_25km	HAFS v0.A grid at 25*km*	ESGgrid
14	GSL_HAFSV0.A_13km	HAFS v0.A grid at 13*km*	ESGgrid

15	GSL_HAFSV0.A_3km	HAFS v0.A grid at 3*km*	ESGgrid
16	GSD_HRRR_AK_50km	GSD's 50*km* HRRR Alaska grid	ESGgrid
17	GSD_RAP13km	GSD's RAP grid	ESGgrid

Table 2.5: Pre-defined grids

2. Parameters for the GFDL grid refinement (Table 2.23) or ESG grid (Table 2.24).

3. Parameters for configurations in 'input.nml' and 'model_configuration' (Table 2.25).

4. Parameters for the write components in case of 'QUILTING'="TRUE":

   (a) 'regional_latlon' or 'rotated_latlon':

Parameter name	Description (Unit: in degrees)
WRTCMP_cen_lon	Central longitude of the output domain
WRTCMP_cen_lat	Central latitude of the output domain
WRTCMP_lon_lwr_left	Longitudinal distance from the center to the bottom-left corner
WRTCMP_lat_lwr_left	Latitudinal distance from the center to the bottom-left corner
WRTCMP_lon_upr_rght	Longitudinal distance from the center to the top-right corner
WRTCMP_lat_upr_rght	Latitudinal distance from the center to the top-right corner
WRTCMP_dlon	Longitudinal distance between two adjacent output-grid points
WRTCMP_dlat	Latitudinal distance between two adjacent output-grid points

Table 2.6: Parameters for 'regional_latlon' or 'rotated_latlon'

   (b) 'lambert_conformal':

Parameter name	Description
WRTCMP_cen_lon	Central longitude of the output domain (in degrees)
WRTCMP_cen_lat	Central latitude of the output domain (in degrees)
WRTCMP_stdlat1	Latitude of the first standard parallel (in degrees)
WRTCMP_stdlat2	Latitude of the second standard parallel (in degrees)
WRTCMP_nx	Total number of grid points along x-axis
WRTCMP_ny	Total number of grid points along y-axis
WRTCMP_lon_lwr_left	Longitude of the bottom-left corner of output (in degrees)

WRTCMP_lat_lwr_left	Latitude of the bottom-left corner of output (in degrees)
WRTCMP_dx	Grid cell size in *x* direction (in meters)
WRTCMP_dy	Grid cell size in *y* direction (in meters)

Table 2.7: Parameters for 'lambert_conformal'

Notes) As shown in Figure 2.3, The script 'set_predef_grid_params.sh' is issued later than 'config.sh'. Therefore, if any parameters are set in both scripts, the parameters in 'set_predef_grid_params.sh' will be finally used in the model.

2.5.3 Valid Values for Configuration Parameters: 'valid_param_vals.sh'

The valid values for the experiment's configuration parameters:

	Parameter name	Valid values
1	RUN_ENVIR	"nco", "community"
2	VERBOSE	"TRUE", "YES", "FALSE", "NO"
3	MACHINE	"WCOSS_CRAY", "WCOSS_DELL_P3", "HERA", "ORION", "JET", "ODIN", "CHEYENNE", "STAMPEDE"
4	SCHED	"slurm", "pbspro", "lsf", "lsfcray", "none"
5	PREDEF_GRID_NAME	"RRFS_CONUS_25km", "RRFS_CONUS_13km", "RRFS_CONUS_3km", "RRFS_SUBCONUS_3km", "RRFS_AK_13km", "RRFS_AK_3km, "CONUS_25km_GFDLgrid", "CONUS_3km_GFDLgrid", "EMC_AK", "EMC_HI", "EMC_PR", "EMC_GU", "GSL_HAFSV0.A_25km", "GSL_HAFSV0.A_13km", "GSL_HAFSV0.A_3km", "GSD_HRRR_AK_50km", "GSD_RAP13km"
6	CCPP_PHYS_SUITE	"FV3_CPT_v0", "FV3_GFS_2017_gfdlmp_regional", "FV3_GSD_SAR", "FV3_GSD_v0", "FV3_GFS_v15p2", "FV3_GFS_v16beta", "FV3_RRFS_v1beta", "FV3_RRFS_v1alpha", "FV3_HRRR"
7	GFDLgrid_RES	"48", "96", "192", "384", "768", "1152", "3072"
8	EXTRN_MDL_NAME_ICS	"GSMGFS", "FV3GFS", "RAP", "HRRR", "NAM"
9	EXTRN_MDL_NAME_LBCS	"GSMGFS", "FV3GFS", "RAP", "HRRR", "NAM"
10	FV3GFS_FILE_FMT_ICS	"nemsio", "grib2", "netcdf"
11	FV3GFS_FILE_FMT_LBCS	"nemsio", "grib2", "netcdf"
12	GIRD_GEN_METHOD	"GFDLgrid", "ESGgrid"
13	PREEXISTING_DIR_ METHOD	"delete", "rename", "quit"
14	GTYPE	"regional"
15	WRTCMP_output_grid	"rotated_latlon", "lambert_conformal", "regional_latlon"
16	RUN_TASK_MAKE_GRID	"TRUE", "YES", "FALSE", "NO"
17	RUN_TASK_MAKE_OROG	"TRUE", "YES", "FALSE", "NO"
18	RUN_TASK_MAKE_SFC_ CLIMO	"TRUE", "YES", "FALSE", "NO"

19	QUILTING	“TRUE”, “YES”, “FALSE”, “NO”
20	PRINT_ESMF	“TRUE”, “YES”, “FALSE”, “NO”
21	USE_CRON_TO_RELAUNCH	“TRUE”, “YES”, “FALSE”, “NO”
22	DOT_OR_USCORE	“.”, “_”
23	NOMADS	“TRUE”, “YES”, “FALSE”, “NO”
24	NOMADS_file_type	“TRUE”, “YES”, “FALSE”, “NO”
25	DO_ESEMBLE	“TRUE”, “YES”, “FALSE”, “NO”
26	USE_CUSTOM_POST_CONFIG_FILE	“TRUE”, “YES”, “FALSE”, “NO”
27	DO_SHUM	“TRUE”, “YES”, “FALSE”, “NO”
28	DO_SPPT	“TRUE”, “YES”, “FALSE”, “NO”
29	DO_SKEB	“TRUE”, “YES”, “FALSE”, “NO”
30	USE_ZMTNBLCK	“TRUE”, “YES”, “FALSE”, “NO”
31	USE_FVCOM	“TRUE”, “YES”, “FALSE”, “NO”
32	COMPILER	“intel”, “gnu”

Table 2.8: Valid values for the configuration parameters

2.5.4 CCPP Suites Supported in FV3-LAM

References)

- https://ccpp-techdoc.readthedocs.io/en/latest/Overview.html
- https://dtcenter.ucar.edu/GMTB/v4.0/sci_doc/index.html

The Common Community Physics Package (CCPP) is composed of two parts: infrastructure (CCPP-Framework) and library of physical parameterizations (CCPP-Physics). The suites that are supported for FV3-LAM can be found in Table 2.8 (No.6; under ‘CCPP_PHYS_SUITE’). The types of parameterization for each suite are listed in Table 2.9.

Parameterization	CCPP Suite	
	GFS_v15p2	GFS_v16beta
Microphysics	GFDL Cloud Microphysics Scheme	GFDL Cloud Microphysics Scheme
PBL	GFS Hybrid Eddy-Diffusivity Mass-Flux (K-EDMF)	GFS Scale-aware TKE-based Moist Eddy-Diffusion Mass-Flux (TKE-EDMF)
Turbulence	Free Atmospheric Turbulence Scheme	Free Atmospheric Turbulence Scheme
Deep convection	GFS Scale-aware Simplified Arakawa-Schubert (sa-SAS)	GFS Scale-aware Simplified Arakawa-Schubert (sa-SAS)
Shallow convection	GFS SAS-based Mass-Flux (sa-MF)	GFS SAS-based Mass-Flux (sa-MF)

Radiation	Rapid Radiative Transfer Model for GCMs (RRTMG)	Rapid Radiative Transfer Model for GCMs (RRTMG)
Surface layer	GFS Surface Layer Scheme	GFS Surface Layer Scheme
Gravity wave drag	Unified Gravity Wave Drag (uGWD)	Unified Gravity Wave Drag (uGWD)
Land surface	GFS Noah Land Surface Model	GFS Noah Land Surface Model
Ozone	2015 Navy Research Laboratory ozone and stratospheric water vapor schemes (NRL 2015)	2015 Navy Research Laboratory ozone and stratospheric water vapor schemes (NRL 2015)
H_2O	NRL 2015	NRL 2015
Ocean	Near-Surface Sea Temperature (NSST)	Near-Surface Sea Temperature (NSST)
	Operational	Experimental

Table 2.9: CCPP Suites: type of parameterization

2.5.5 Default Configuration: 'config_default.sh'

This file sets the experiment's configuration parameters (global shell variables) to their default values. For many of these parameters, the valid values are defined in '{HOMErrfs}/ush/valid_param_vals.sh' (or Section 2.5.3). Since this file contains the default settings, you do **NOT** need to change it. The user-specified experimental parameters will be replaced by running the user-specified experimental configuration file 'config.[option].sh'.

1. Experimental mode:

Parameter name	Description
RUN_ENVIR	"community": community mode, or "nco": NCO mode

Table 2.10: Configuration - experimental mode

2. Machine and queue parameters:

Parameter name	Description
MACHINE	HPC machine on which the workflow runs
ACCOUNT	Account name to submit jobs to the queue on HPC
SCHED	Job scheduler (e.g. 'slurm')
PARTITION_DEFAULT	Partition to which it will be submitted. If this is not set or is set to an empty string, it will be set to a machine-dependent value. This is not used if SCHED is not set to "slurm".
QUEUE_DEFAULT	Default queue to which workflow tasks are submitted

PARTITION_HPSS	Partition for the tasks that create links to external model files for IC and LBC. If this is not set or is set to an empty string, it will be set to a machine-dependent value. This is not used if SCHED is not set to "slurm".
QUEUE_HPSS	Queue for the external model files such as IC and LBC
PARTITION_FCST	Partition for the task that runs forecasts. If this is not set or set to an empty string, it will be set to a machine-dependent value. This is not used if SCHED is not set to "slurm".
QUEUE_FCST	Queue for a forecast run

Table 2.11: Configuration - machine and queue

3. Cron-associated parameters:

Parameter name	Description
USE_CRON_TO_RELAUNCH	Flag to add a line to the user's cron table to call the experiment launch script every 'CRON_RELAUNCH_ INTVL_MNTS' minutes
CRON_RELAUNCH_INTVL_MNTS	Interval (in minutes) between successive calls of the experiment launch script by a cron job

Table 2.12: Configuration - cron

4. Experiment directories:

Parameter name	Description
EXPT_BASEDIR	Path and name of the base directory where the experiment is created
EXPT_SUBDIR	Name of the subdirectory in the base directory

Table 2.13: Configuration - Experiment directories

5. Parameters only for the NCO mode (when RUN_ENVIR="nco"):

Parameter name	Description
COMINgfs	Path to the parent directory of the GFS input files for IC/LBC
FIXLAM_NCO_BASEDIR	Base directory containing pre-generated grid, orography, and surface climatology files
STMP	Path to the parent directory cycle-dependent model input files, symlinks to cycle-independent input files, and raw forecast output files
NET	Model name (first level of com directory structure)
envir	"test" for initial testing phase, "para" for run in parallel, "prod" in production
RUN	Name of model run (third level of com directory structure)
PTMP	Path to the parent directory of post-processed output files

Table 2.14: Configuration - NCO mode

6. Separator character:

Parameter name	Description
DOT_OR_USCORE	Separator (character) used for the names of the gird, mosaic, and orography fixed files

Table 2.15: Configuration - Separator

7. File names:

Parameter name	Description	Default value
EXPT_CONFIG_FN	Name of the user-specified configuration file	config.sh
RGNL_GRID_NML_FN	Name of the file containing the namelist settings for the ESG (JP) grid	regional_grid.nml
DATA_TABLE_FN	Empty file	data_table
DIAG_TABLE_FN	File name for output fields of the forecast model	diag_table
FIELD_TABLE_FN	File name for tracers of the forecast model	field_table
FV3_NML_BASE_ SUITE_FN	Name of the Fortran namelist file for *FV3*	input.nml.FV3
FV3_NML_YAML_ CONFIG_FN	YAML configuration file containing the forecast model's namelist settings for physics suits	FV3.input.yml
FV3_NML_BASE_ENS _FN	Fortran namelist file containing the forecast model's base ensemble namelist	input.nml.base_ens
MODEL_CONFIG_ FN	Name of the file containing settings and configurations for the NUOPC/ESMF main component	model_configure
NEMS_CONFIG_FN	Name of the NEMS configuration file	nems.configure
FV3_EXEC_FN	Name of the *FV3* executable	NEMS.exe
WFLOW_XML_FN	Name of the rocoto workflow XML file	FV3LAM_wflow .xml
GLOBAL_VAR_DEFNS _FN	Sell script containing the definitions of experiment parameters	var_defns.sh
EXTRN_MDL_ICS_ VAR_DEFNS_FN	Shell script containing parameters of the external model for IC	extrn_mdl_ics _var_defns.sh
EXTRN_MDL_LBCS_ VAR_DEFNS_FN	Shell script containing parameters of the external model for LBC	extrn_mdl_lbcs _var_defns.sh
WFLOW_LAUNCH _SCRIPT_FN	Name of the script to launch the experiment's rocoto workflow	launch_FV3LAM _wflow.sh
WFLOW_LAUNCH _LOG_FN	Name of the log file that contains the output from successive calls to the workflow launch script	log.launch_ FV3LAM_wflow

Table 2.16: Configuration - input files

8. Forecast parameters:

Parameter name	Description
DATE_FIRST_CYCL	Starting date of the first forecast (format: YYYYMMDD)
DATE_LAST_CYCL	Starting date of the last forecast (format: YYYYMMDD)
CYCL_HRS	An array containing the hours of the day at which to launch forecasts. Each element of this array must be a two-digit string less than 24.
FCST_LEN_HRS	Length of each forecast (in integer hours)

Table 2.17: Configuration - forecast parameters

9. Initial and lateral boundary condition generation parameters:

Parameter name	Description	Default
EXTRN_MDL_NAME_ ICS	Name of the external model that provides fields for initial conditions	FV3GFS
EXTRN_MDL_NAME_LBCS	Name of the external model that provides fields for lateral boundary conditions	FV3GFS
LBC_SPEC_INTVL_HRS	Interval of the LBC files (in integer hours)	6
FV3GFS_FILE_FMT_ICS	Format of the model files for IC	nemsio
FV3GFS_FILE_FMT_LBCS	Format of the model files for LBC	nemsio

Table 2.18: Configuration - IC and LBC

10. NOMADS online data associated parameters:

Parameter name	Description	Default
NOMADS	Flag for NOMADS online data	FALSE
NOMADS_file_type	Format of NOMADS data	nemsio

Table 2.19: Configuration - NOMADS

11. User-staged external model directories and files:

Parameter name	Description
USE_USER_STAGED_EXTRN_FILES	Flag for external model files for ICs and LBCs • "TRUE": {BASEDIR}/{CDATE} • "FALSE": {BASEDIR}
EXTRN_MDL_SOURCE_BASEDIR_ICS	Directory containing the external model files for ICs
EXTRN_MDL_FILES_ICS	External model files for ICs
EXTRN_MDL_SOURCE_BASEDIR_LBCS	Directory containing the external model files for LBCs
EXTRN_MDL_FILES_LBCS	External model files for LBCs

Table 2.20: Configuration - user-staged external model

12. CCPP parameters:

Parameter name	Description
CCPP_PHYS_SUITE	Physics suite for CCPP

Table 2.21: Configuration - CCPP parameters

13. Grid-generation method:

Parameter name	Description	Default
GRID_GEN_METHOD	Method of generating a regional grid	ESGgrid

Table 2.22: Configuration - grid generation method

14. Parameters for the GFDL grid refinement:

Parameter name	Description
GFDLgrid_LON_T6_CTR	Longitude of the center of Tile 6 of the parent global domain (in degrees)
GFDLgrid_LAT_T6_CTR	Latitude of the center of Tile 6 of the parent global domain (in degrees)
GFDLgrid_RES	Resolution of the **parent** global cubed-sphere grid (Tile 6)
GFDLgrid_STRETCH_FAC	Schmidt stretching factor: shrink (≥ 1) or expand ($0 \leq [] \leq 1$)
GFDLgrid_REFINE_RATIO	Grid refinement ratio
GFDLgrid_ISTART_OF_RGNL_DOM_ON_T6G	*i*-index on Tile 6 at which the regional grid (Tile 7) starts
GFDLgrid_IEND_OF_RGNL_DOM_ON_T6G	*i*-index on Tile 6 at which the regional grid (Tile 7) ends
GFDLgrid_JSTART_OF_RGNL_DOM_ON_T6G	*j*-index on Tile 6 at which the regional grid (Tile 7) starts
GFDLgrid_JEND_OF_RGNL_DOM_ON_T6G	*j*-index on Tile 6 at which the regional grid (Tile 7) ends
GFDLgrid_USE_GFDLgrid_RES_IN_FILENAMES	Flag for prefix resolution 'C{RES}_'. • "TRUE": {RES}='GFDLgrid_RES' as used in the global forecast model. • "FALSE": 'RES_EQUIV' is calculated and set.

Table 2.23: Configuration - GFDL grid refinement

Notes)

- The regional grid is defined with respect to a 'parent' global cubed-sphere grid. Thus, all the parameters for a global cubed-sphere grid must be specified in order to define

this parent global grid even though the model equations are not integrated on (they are integrated only on the regional grid).

- The above parameters can be obtained by *fv3grid* described in Section 11.1.
- The above parameters for the pre-defined grids are specified in '{HOMErrfs}/ush/set_predef_grid_params.sh'.
- Setting 'GFDLgrid_USE_GFDLgrid_RES_IN_FILENAMES'="FALSE" is a more useful indicator of the grid size because it takes into account the effects of 'GFDLgrid_RES', 'GFDLgrid_STRETCH_FAC', and 'GFDLgrid_REFINE_RATIO' in determining the regional grid's typical grid size, whereas simply setting {RES} to 'GFDLgrid_RES' doesn't take into account the effects of 'GFDLgrid_STRETCH_FAC' and 'GFDLgrid_REFINE_RATIO' on the regional grid-resolution. Nevertheless, some users still prefer to use 'GFDLgrid_RES' in the file names, so we allow for that here by setting this flag to "TRUE".

15. Parameters for the ESG (JP) grid:

Parameter name	Description
ESGgrid_LON_CTR	Longitude of the center of the grid (in degrees)
ESGgrid_LAT_CTR	Latitude of the center of the grid (in degrees)
ESGgrid_DELX	Cell size in the zonal direction of the regional grid (in meters)
ESGgrid_DELY	Cell size in the meridional direction of the regional grid (in meters)
ESGgrid_NX	Total number of cells in the zonal direction on the regional grid
ESGgrid_NY	Total number of cells in the meridional direction on the regional grid
ESGgrid_WIDE_HALO_WIDTH	Width of the halo to add around the regional grid before shaving the halo down to the width(s) expeted by the forecast model (unit: cells)

Table 2.24: Configuration - ESG (JP) grid

Notes)

- In order to generate grid files containing halos that are 3- and 4-cell wide and orography files with halos that are 0- and 3-cell wide, the grid and orography tasks first create files with halos around the regional domain of width 'ESGgrid_WIDE_HALO_WIDTH' cells. These are first stored in files. The files are then read in and shaved down to obtain grid files with 3- and 4-cell-wide halos and orography files with 0- and 3-cell-wide halos. For this reason, we refer to the original halo that then gets shaved down as the 'wide'

halo, i.e. because it is wider than the 0-, 3-, and 4-cell-wide halos that we will eventually end up with. You do **NOT** need to make this a user-specified parameter, but set it in the function of 'set_gridparams_ESGgrid.sh'.

- The above parameters for the pre-defined grids are specified in '{HOMErrfs}/ush/set_predef_grid_params.sh'.

16. Parameters for configurations in 'input.nml' and 'model_configuration':

Parameter name	Description
DT_ATMOS	Time integration step of the main forecast model
LAYOUT_X	The number of MPI tasks (processes) used in the zonal direction on the regional grid
LAYOUT_Y	The number of MPI tasks used in the meridional direction on the regional grid
BLOCKSIZE	Amount of data passed into the cache at a time. Note that if using one of the predefined grids and if BLOCKSIZE is not explicitly set in the user-specified experiment configuration file (EXPT_CONFIG_FN), then the default value of BLOCKSIZE specified here will be overwritten by its default value for that predefined grid.
QUILTING	Flag for write component writing output files
PRINT_ESMF	Flag for extra (debugging) information from ESMF routines, Note that the write component uses ESMF library routines to interpolate from the native forecast model grid to the user-specified output grid.
WRTCMP_write_groups	The number of write groups (groups of MPI tasks) in the write component
WRTCMP_write_tasks_per_group	The number of MPI tasks allocated for each write group
WRTCMP_output_grid	
WRTCMP_cen_lon	
WRTCMP_cen_lat	
WRTCMP_lon_lwr_left	
WRTCMP_lat_lwr_left	
WRTCMP_lon_upr_rght	Parameter for 'rotated_latlon'
WRTCMP_lat_upr_rght	Parameter for 'rotated_latlon'
WRTCMP_dlon	Parameter for 'rotated_latlon'
WRTCMP_dlat	Parameter for 'rotated_latlon'
WRTCMP_stdlat1	Parameter for 'lambert_conformal'
WRTCMP_stdlat2	Parameter for 'lambert_conformal'
WRTCMP_nx	Parameter for 'lambert_conformal'
WRTCMP_ny	Parameter for 'lambert_conformal'
WRTCMP_dx	Parameter for 'lambert_conformal'
WRTCMP_dy	Parameter for 'lambert_conformal'

Table 2.25: Configuration - configuration parameters

Notes)

- The above parameters and additional parameters related to the write component for the pre-defined grids are specified in '{HOMErrfs}/ush/set_predef_grid_params.sh'.
- If LAYOUT_X and/or LAYOUT_Y are not explicitly set in the user-specified configuration file ('config.sh'), then the default values specified here will be overwritten by their default values for that predefined grid.

17. Other configurations:

Parameter name	Description
PREDEF_GRID_NAME	Pre-defined regional grid specified in the script '{HOMErrfs}/ush/set_predef_grid_params.sh'. Setting PREDEF_GRID_NAME provides a convenient method of specifying a commonly used set of grid-dependent parameters.
PREEXISTING_DIR_METHOD	Method for dealing with pre-existing directories: • "delete": The pre-existing directory is deleted and a new directory is created. • "rename": The pre-existing directory is renamed (_oldX) and a new directory is created. • "quit": The pre-existing directory is left unchanged, but the current running script is terminated.
VERBOSE	Flag for printing out more informational message of the experiment generation and workflow task scripts

Table 2.26: Configuration: Others

18. Grid, orography, and/or surface climatology file generation:

Parameter name	Description
RUN_TASK_MAKE_GRID	Flag for the grid file generation task: • "TRUE": The grid generation task runs to create new grid files. • "FALSE": Pre-generated grid files in 'GRID_DIR' are used.
GRID_DIR	Path to the pre-generated grid files in case of 'RUN_TASK_MAKE_ GRID'="FALSE"
RUN_TASK_MAKE_OROG	Flag for the orography generation task
OROG_DIR	Path to the pre-generated orography files in case of 'RUN_TASK_MAKE_OROG' ="FALSE"
RUN_TASK_MAKE_SFC_CLIMO	Flag for the surface climatology generation task
SFC_CLIMO_DIR	Path to the pre-generated surface climatology files in case of 'RUN_TASK_MAKE_ SFC_CLIMO'="FALSE"

Table 2.27: Configuration: grid, orography, and surface climatology

19. Static fields generated by 'MAKE_SFC_CLIMO_TN' task:

Parameter name	Description
SFC_CLIMO_FIELDS	Names of all the surface climatology fields
FIXgsm	System directory where the majority of fixed (time-independent) files
TOPO_DIR	Directory containing the static input files used by 'MAKE_OROG' task
SFC_CLIMO_INPUT_ DIR	Directory containing the surface climatology data
FNGLAC,$\cdots$,FNMSKH	Names of the global data files. These names also appear directly in the input namelist file ('input.nml') of the forecast model.
FIXgsm_FILES_TO_COPY_ TO_FIXam	If not running in 'NCO' mode, this array contains the names of the files to copy from the FIXgsm system directory to the FIXam directory under the experiment directory. • The last element has a dummy value. It will get reset by the workflow generation scripts to the name of the ozone production/loss file to copy from FIXgsm. The name of this file depends on the ozone parameterization and then the CCPP physics suite specified for the experiment. These steps are carried out elsewhere (in one of the workflow generation scripts/functions)
FV3_NML_VARNAME_TO _FIXam_FILES_MAPPING	Some fixed files in the FIXam directory derived from the corresponding workflow variables containing file names
FV3_NML_VARNAME_TO _SFC_CLIMO_FIELD _MAPPING	Surface climatology files (on the native FV3LAM grid) in the FIXsar directory derived from the corresponding surface climatology fields
CYCLEDIR_LINKS_TO _FIXam_FILES_MAPPING	Mapping between the symlinks created in each cycle directory and their target files in FIXam

Table 2.28: Configuration: fixed files

20. Names of the various workflow tasks:

Parameter name	Description	Default
MAKE_GRID_TN	Task name for grid generation	make_grid
MAKE_OROG_TN	Task name for orography generation	make_orog
MAKE_SFC_CLIMO_TN	Task name for generating surface climatology	make_sfc_climo
GET_EXTRN_ICS_TN	Task name for obtaining external data for IC	get_extrn_ics
GET_EXTRN_LBCS_TN	Task name for obtaining external data for LBC	get_extrn_lbcs
MAKE_ICS_TN	Task name for generating IC from the external data	make_ics
MAKE_LBCS_TN	Task name for generating LBC from the external data	make_lbcs
RUN_FCST_TN	Task name for running the forecast model	run_fcst
RUN_POST_TN	Task name for running the post-processing tool	run_post

Table 2.29: Configuration: workflow tasks

21. Number of nodes for each task:

Parameter name	Description
NNODES_MAKE_GRID	Number of nodes for grid generation
NNODES_MAKE_OROG	Number of nodes for orography generation
NNODES_MAKE_SFC_CLIMO	Number of nodes for generating surface climatology
NNODES_GET_EXTRN_ICS	Number of nodes for obtaining external data for IC
NNODES_GET_EXTRN_LBCS	Number of nodes for obtaining external data for LBC
NNODES_MAKE_ICS	Number of nodes for generating IC from the external data
NNODES_MAKE_LBCS	Number of nodes for generating LBC from the external data
NNODES_RUN_FCST	Number of nodes for running the forecast model
NNODES_RUN_POST	Number of nodes for running the post-processing tool

Table 2.30: Configuration: number of nodes

22. Number of MPI processes per node for each task:

Parameter name	Description
PPN_MAKE_GRID	Number of MPI processes for grid generation
PPN_MAKE_OROG	Number of MPI processes for orography generation
PPN_MAKE_SFC_CLIMO	Number of MPI processes for generating surface climatology
PPN_GET_EXTRN_ICS	Number of MPI processes for obtaining external data for IC
PPN_GET_EXTRN_LBCS	Number of MPI processes for obtaining external data for LBC
PPN_MAKE_ICS	Number of MPI processes for generating IC from the data
PPN_MAKE_LBCS	Number of MPI processes for generating LBC from the data
PPN_RUN_FCST	Number of MPI processes for running the forecast model
PPN_RUN_POST	Number of MPI processes for running the post-processing tool

Table 2.31: Configuration: MPI processes

23. Walltime for each task:

Parameter name	Description
WTIME_MAKE_GRID	Walltime for grid generation
WTIME_MAKE_OROG	Walltime for orography generation
WTIME_MAKE_SFC_CLIMO	Walltime for generating surface climatology
WTIME_GET_EXTRN_ICS	Walltime for obtaining external data for IC
WTIME_GET_EXTRN_LBCS	Walltime for obtaining external data for LBC
WTIME_MAKE_ICS	Walltime for generating IC from the external data
WTIME_MAKE_LBCS	Walltime for generating LBC from the external data
WTIME_RUN_FCST	Walltime for running the forecast model
WTIME_RUN_POST	Walltime for running the post-processing tool

Table 2.32: Configuration: walltime

24. Maximum number of attempts for each task:

Parameter name	Description
MAXTRIES_MAKE_GRID	Max. attempts for grid generation
MAXTRIES_MAKE_OROG	Max. attempts for orography generation
MAXTRIES_MAKE_SFC_CLIMO	Max. attempts for generating surface climatology
MAXTRIES_GET_EXTRN_ICS	Max. attempts for obtaining external data for IC
MAXTRIES_GET_EXTRN_LBCS	Max. attempts for obtaining external data for LBC
MAXTRIES_MAKE_ICS	Max. attempts for generating IC from the external data
MAXTRIES_MAKE_LBCS	Max. attempts for generating LBC from the external data
MAXTRIES_RUN_FCST	Max. attempts for running the forecast model
MAXTRIES_RUN_POST	Max. attempts for running the post-processing tool

Table 2.33: Configuration: Maximum number of attempts

25. Parameters for a customized post configuration file:

Parameter name	Description
USE_CUSTOM_POST_CONFIG_FILE	Flag for using a user-provided custom configuration file
CUSTOM_POST_CONFIG_FP	Full path to the custom post flat file (path+file name)

Table 2.34: Configuration: Customized post

26. Parameters for running ensembles:

Parameter name	Description
DO_ENSEMBLE	Flag for running a set of ensemble forecasts
NUM_ENS_MEMBERS	Total number of ensemble members. It should correspond to the number of digits in the directory names by putting leading zeros

Table 2.35: Configuration: Ensembles

27. Stochastic physics options:

Parameter name	Description	Default
DO_SHUM	Logical to tell parent atmospheric model to use SHUM	false
DO_SPPT	Logical to tell parent atmospheric model to use SPPT	false

DO_SKEB	Logical to tell parent atmospheric model to use SKEB	false
SHUM_MAG	Amplitude of random pattern	0.006
SHUM_LSCALE	Decorrelation spatial scales in meters	150000
SHUM_TSCALE	Decorrelation time scales in seconds	21600
SHUM_INT	Interval in seconds to update random pattern	3600
SPPT_MAG	Amplitude of random pattern	1.0
SPPT_LSCALE	Decorrelation spatial scales in meters	150000
SPPT_TSCALE	Decorrelation time scales in seconds	21600
SPPT_INT	Interval in seconds to update random pattern	3600
SKEB_MAG	Amplitude of random pattern	0.5
SKEB_LSCALE	Decorrelation spatial scales in meters	150000
SKEB_TSCALE	Decorrelation time scales in seconds	21600
SKEB_INT	Interval in seconds to update random pattern	3600
SKEB_VDOF	Degree of freedom in vertical for the SKEB random pattern	10
USE_ZMTNBLCK	If TRUE, do not apply perturbations below the dividing streamline that is diagnosed by the gravity wave drag, mountain blocking scheme	false

Table 2.36: Configuration: Stochastic physics

28. SPP stochastic physics:

Parameter name	Description	Default
DO_SPP	Logical to use SPP	false
SPP_VAR_LIST	SPP option (array, without commas or single quotes)	pbl
SPP_MAG_LIST	Variable ‘spp_prt_list’ in ‘input.nml’	0.2
SPP_LSCALE		150000
SPP_TSCALE	Variable ‘spp_tau’ in ‘input.nml’	21600
SPP_SIGTOP1		0.1
SPP_SIGTOP2		0.025
SPP_STDDEV_CUTOFF		1.5

Table 2.37: Configuration: SPP stochastic physics

29. Boundary blending:

Parameter name	Description	Default
HALO_BLEND	Number of rows into the computational domain that should be blended with the LBCs	10

Table 2.38: Configuration: Boundary blending

30. Surface fields from FVCOM:

Parameter name	Description	Default
USE_FVCOM	Flag for updating surface conditions in FV3-LAM with fields generated from the Finite Volume Community Ocean Model (FVCOM)	FALSE
FVCOM_DIR	User defined directory where FVCOM data already interpolated to FV3-LAM grid	
FVCOM_FILE	File name of the FVCOM data	fvcom.nc

Table 2.39: Configuration: Data from FVCOM

31. Compiler:

Parameter name	Description	Default
COMPILER	Type of compiler invoked during the build step	intel

Table 2.40: Configuration: Type of compiler

32. GWD HRRR suite:

Parameter name	Description	Default
GWD_HRRRsuite_ BASEDIR	Temporary base directory where certain fixed orography statistics files for the gravity save drag parameterization in the FV3_HRRR physics suite	-

Table 2.41: Configuration: GWD HRRR suit

2.5.6 User-specific Configuration: 'config.sh'

The parameters specified in the 'config.sh' file will replace the corresponding parameters in the default configuration. You should NOT modify the 'config_default.sh' file BUT add new values of the parameters, which you want to change, to 'config.sh'. All the parameters for configuration are described in detail in Section 2.5.5. There are two sample files in '{HOMErrfs}/ush/': (1) 'config.community.sh' and (2) 'config.nco.sh'. You can copy either of them for your purpose, and change its name to 'config.sh'. Make sure that the values of the configuration parameters should be consistent with them in the '{HOMErrfs}/ush/valid_param_vals' script file.

1. 'community' mode: 'config.community.sh'

Parameter name	Default	New configuration
MACHINE	"BIG_COMPUTER"	See Table 2.44
ACCOUNT	"project_name"	See Table 2.44

EXPT_SUBDIR	" "	"test_community"
RUN_ENVIR	"nco"	"community"
PREEXISTING_DIR_METHOD	"delete"	"rename"
PREDEF_GRID_NAME	" "	"RRFS_CONUS_25km"
GRID_GEN_METHOD	"ESGgrid"	"ESGgrid"
QUILTING	"TRUE"	"TRUE"
CCPP_PHYS_SUITE	"FV3_GSD_V0"	"FV3_GFS_v15p2"
FCST_LEN_HRS	"24"	"48"
LBC_SPEC_INTVL_HRS	"6"	"6"
DATE_FIRST_CYCL	"YYYYMMDD"	"20190615"
DATE_LAST_CYCL	"YYYYMMDD"	"20190615"
CYCL_HRS	("HH1" "HH2")	"00"
EXTRN_MDL_NAME_ICS	"FV3GFS"	"FV3GFS"
EXTRN_MDL_NAME_LBCS	"FV3GFS"	"FV3GFS"
FV3GFS_FILE_FMT_ICS	"nemsio"	"grib2"
FV3GFS_FILE_FMT_LBCS	"nemsio"	"grib2"
USE_USER_STAGED_EXTRN_FILES	"FALSE"	"TRUE"
EXTRN_MDL_SOURCE_BASEDIR_ICS	"/base/dir/···"	See Table 7.1
EXTRN_MDL_FILES_ICS	("ICS_file1" ···)	See Table 7.2
EXTRN_MDL_SOURCE_BASEDIR_LBCS	"/base/dir/···"	See Table 7.1
EXTRN_MDL_FILES_LBCS	("LBCS_file1" ···)	See Table 7.2

Table 2.42: New configuration for the community mode

2. 'nco' mode: 'config.nco.sh'

Parameter name	Default	New configuration
MACHINE	"BIG_COMPUTER"	See Table 2.44
ACCOUNT	"project_name"	See Table 2.44
EXPT_SUBDIR	" "	"test_nco"
RUN_ENVIR	"nco"	"community"
PREEXISTING_DIR_METHOD	"delete"	"rename"
PREDEF_GRID_NAME	" "	"RRFS_CONUS_25km"
QUILTING	"TRUE"	"TRUE"
CCPP_PHYS_SUITE	"FV3_GSD_V0"	"FV3_GFS_v15p2"
FCST_LEN_HRS	"24"	"06"
LBC_SPEC_INTVL_HRS	"6"	"6"
DATE_FIRST_CYCL	"YYYYMMDD"	"20190901"
DATE_LAST_CYCL	"YYYYMMDD"	"20190901"
CYCL_HRS	("HH1" "HH2")	"18"
EXTRN_MDL_NAME_ICS	"FV3GFS"	"FV3GFS"

EXTRN_MDL_NAME_LBCS	"FV3GFS"	"FV3GFS"
RUN	"experiment_name"	"an_experiment"
COMINgfs	"/base/path/···"	See Table 7.1
FIXLAM_NCO_BASEDIR		
STMP	"/base/path/···"	"/scratch2/NCEPDEV/···"
PTMP	"/base/path/···"	"/scratch2/NCEPDEV/···"

Table 2.43: New configuration for the NCO mode

Notes)

- If running in the 'nco' mode, a valid EMC grid must be specified. Make sure that 'EMC_GRID_NAME' is set to a valid value. Its valid values are shown in Table 2.8.
- In the 'nco' mode, 'RUN_TASK_MAKE_GRID', 'RUN_TASK_MAKE_OROG', and 'RUN_TASK_MAKE_SFC_CLIMO' do not need to be explicitly set to "FALSE" in this configuration file because the experiment generation script will do this (along with printing out an informational message).

Machine-specific parameters

Parameter	Machine			
	Hera	WCOSS_dell_p3	WCOSS_cray	Orion
MACHINE	"hera"	"wcoss_dell_p3"	"wcoss_cray"	"orion"
ACCOUNT	"fv3-cam"	"RRFS-T20"	"RRFS-T20"	"fv3-cam"

Table 2.44: Machine-specific queue parameters

Sample configuration scripts

You can find some sample 'config_[option].sh' scripts as follows:

```
git clone https://github.com/chan-hoo/regional_workflow_config.git
cd regional_workflow_config
```

2.5.7 Static (FIX) Files for the 'nco' Mode

The 'nco' mode of the regional workflow intends to run the regional workflow with the pre-generated static (FIX) files including grids, mosaic, orography, and surface climatology which are located in '/expt_dirs/{EXPT_SUBDIR}/fix_lam/'. In this mode, The first three steps in Figure 3.1 are skipped automatically. Therefore, case-specific static (FIX) files should exist in the designated directory ({FIXLAM_NCO_BASEDIR}/{PREDEF_GRID_NAME}).

Machine	Path to the FIX data ({FIXLAM_NCO_BASEDIR})
Hera	/scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/00_DATA/FV3LAM_pregen
WCOSS_dell_p3	/gpfs/dell2/emc/modeling/noscrub/UFS_SRW_App/FV3LAM_pregen
WCOSS_cray	/gpfs/hps3/emc/meso/noscrub/UFS_SRW_App/FV3LAM_pregen
Orion	-

Table 2.45: Path to the parent directory of FIX data for the 'nco' mode

The FIX files necessary for the 'nco' mode are as follows:

- C{res}_grid.tile7.halo[3,4,6].nc
- C{res}_mosaic.halo[3,4,6].nc
- C{res}_oro_data.tile7.halo[0,4].nc
- C{res}.facsf.tile7.halo[0,4].nc
- C{res}.maximum_snow_albedo.tile7.halo[0,4].nc
- C{res}.slope_type.tile7.halo[0,4].nc
- C{res}.snowfree_albedo.tile7.halo[0,4].nc
- C{res}.soil_type.tile7.halo[0,4].nc
- C{res}.substrate_temperature.tile7.halo[0,4].nc
- C{res}.vegetation_greenness.tile7.halo[0,4].nc
- C{res}.vegetation_type.tile7.halo[0,4].nc

2.6 Templates of Input Files

2.6.1 List of Template Files

The template files for a regional workflow are located in '{HOMErrfs}/ush/templates/':

File name	Description
data_table	Cycle-independent file that the forecast model reads in at the start of each forecast. It is an **empty** file. **No** need to change.
diag_table_[CCPP]	File specifying the output fields of the forecast model
field_table_[CCPP]	Cycle-independent file that reads in at the start of each forecast. It specifies the scalars that the forecast model will advect.
FV3.input.yml	YAML configuration file containing the forecast model's namelist settings for various physics suites
FV3LAM_wflow.xml	Rocoto XML file to run the workflow
input.nml.FV3	Namelist file of the weather model.
model_configure	Configurations for the NUOPC/ESMF main component
nems.configure	NEMS (NOAA Environmental Modeling System) configuration file, **no** need to change because *FV3* runs stand-alone in SRW App.
regional_grid.nml	Namelist settings for the code that generates an EGS (JP) grid.
README.xml_templating.md	Instruction of Rocoto XML templating with Jinja

Table 2.46: Template files for a regional workflow

2.6.2 Migratory Route of Input Files in Workflow

Figure 2.4 shows how the case-specific input files in the templates directory, such as 'data_table', 'diag_table', 'field_table', 'nems.configure', and 'input.nml', are migrated to the experiment directory. The value of 'CCPP_PHYS_SUITE' is specified in the configuration file 'config.sh'. The path and name of the template input files are set by the 'setup.sh' script in the '{HOMErrfs}/ush' directory. The 'diag_table' is created by running the 'create_diag_table_files.sh'. The template input files for the specific 'CCPP_PHYS_SUITE', such as 'field_table' and 'nems_configure', are copied to the experiment directory '{EXPTDIR}, and the namelist file of the weather model ('input.nml') is created from the 'input.nml.FV3' and 'FV3.input.yml' files by running the script 'generate_FV3LAM_wflow.sh'. While running the task 'RUN_FCST' in the regional workflow as

shown in Figure 3.1, the ‘data_table’, ‘field_table’, ‘nems.configure’, and ‘input.nml’ files, located in {EXPTDIR} are linked to the case-run directory {RUN_DIR}.

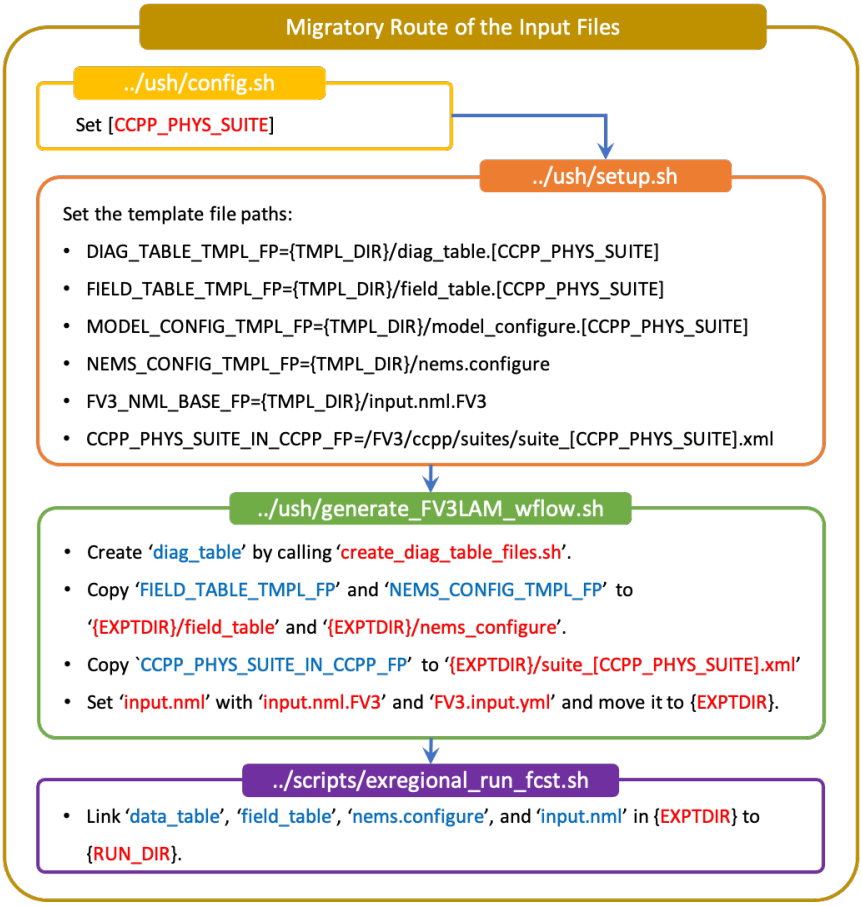

Figure 2.4: Migratory route of the input files

Note) The script ‘{HOMErrfs}/ush/set_namelist.py’ reads ‘input.nml.FV3’ and ‘FV3.input.yml’ in the ‘{HOMErrfs}/ush/templates/’ directory and writes ‘input.nml’. The input parameters are sorted in alphabetical order when they are written in the output ‘input.nml’ file. To make the sort order of the parameters of ‘input.nml’ the same as those in ‘input.nml.FV3’, you should modify the ‘sort’ flag for the command ‘nml.write’ in ‘set_namelist.py’ as follows:

```
307    with open(cla.outfile, 'w') as fn:
308        if cla.type == 'nml':
309 #          nml.write(fn, sort=True)
```

```
310         nml.write(fn, sort=False)
```

2.6.3 data_table

The 'data_table' is empty.

2.6.4 diag_table.[CCPP]

Reference) Link to the UFS weather model: input/output

This file specifies the output fields of the forecast model:

1. Header section:
   - The 'Header' section must reside in the first two lines of the file and contain the title and date of the experiment.
   - The title must be a Fortran character string.
   - The base is the reference time used for the time units, and must be greater than or equal to the model start time.
   - The base date consists of six space-separated integers in the format of 'year month day hour minute second'.
2. Output file entries:

Col.	File entry	Description	Options
1	file_name	Output file name	"grid_spec", "atmos_4xdaily", "atmos_static", "fv3_history", "fv3_history2d"
2	output_freq	Output frequency	-1 (only at the end), 0 (every time step), >0 (designated in column 3)
3	output_freq_units	Units used for output frequency	"years", "months", "days", "hours", "minutes", "seconds"
4	file_format	Format	1 (NetCDF)
5	time_axis_units	Units used to label the time axis in the file	"years", "months", "days", "hours", "minutes", "seconds"
6	time_axis_name	Axis name for the output file time axis	"time"
7	new_file_freq	Frequency for closing the existing file	(optional)

8	new_file_ freq_units	Time units for creating a new file	(optional) “years”, “months”, “days”, “hours”, “minutes”, “seconds”
9	start_time	Time to start the file for the first time	(optional)
10	file_duration	How long file should receive data after start time	(optional)
11	file_duration _units	File duration units	(optional) “years”, “months”, “days”, “hours”, “minutes”, “seconds”

Table 2.47: Output file entries for diagnotics

3. Output format entries:

Column	Description	Options
1	Module name	“dynamics”, “gfs_dyn”, “gfs_phys”, “gfs_sfc”
2	Field name	Field names in modules (Tables 2.49 – 2.52
3	Output name	Field names in output files
4	File name	Output file name in Table 2.47 (column 1)
5	Sampling time	“all” (not used)
6	Data reduction method	.false. (not supported)
7	Bounds of the regional section to capture	“none”
8	Packing	Fortran number ‘KIND’ of the data written: 1 (Double precision), 2 (Float), 4 (Packed 16-bit integer), 8 (Packed 1-bit integer)

Table 2.48: Output file entries for diagnotics

Field lists of the modules in the above table

(a) ‘dynamics’ (‘FV3/atmos_cubed_sphere/tools/fv_diagnostics.F90’):

Field name	Description	Unit
4km_reflectivity	Stoelinga simulated base reflectivity	dBz
aam	Angular momentum	kg m2/s
acl	Column-averaged Cl mixing ratio	kg/kg
acl2	Column-averaged Cl2 mixing ratio	kg/kg
acly	Column-averaged total chlorine mixing ratio	kg/kg
amdt	Angular momentum error	kg m2/s2
area	Cell area	m2

base_reflectivity	Stoelinga simulated base (1km AGL) reflectivity	dBz
bk	Vertical coordinate sigma value	-
cape	Convective available potential energy (surface-based)	J/kg
cat15	Depression < 1000	mb
cat25	Depression < 980	mb
cat35	Depression < 964	mb
cat45	Depression < 944	mb
cin	Convective inhibition (surface-based)	J/kg
ctp	Cloud top pressure	hPa
ctt	Cloud top temperature	K
ctz	Cloud top height	hPa
delp	Pressure thickness	pa
delz	Height thickness	m
divg	Mean divergence	1/s
dp10	10-mb dew point	K
dp50	50-mb dew point	K
dp100	100-mb dew point	K
dp200	200-mb dew point	K
dp250	250-mb dew point	K
dp300	300-mb dew point	K
dp500	500-mb dew point	K
dp700	700-mb dew point	K
dp850	850-mb dew point	K
dp925	925-mb dew point	K
dp1000	1000-mb dew point	K
echo_top	Echo top (<= 18.5 dBz)	m
f15	Cat15 frequency	-
f25	Cat25 frequency	-
f35	Cat35 frequency	-
f45	Cat45 frequency	-
grid_lon	Longitude	Degree E
grid_lat	Latitude	Degree N
grid_lont	Longitude	Degree E
grid_latt	Latitude	Degree N
h_plev	Height	m
hght	Height	m
hyam	Vertical coordinate A value	1E-5 Pa
hybm	Vertical coordinate B value	-
hw	Vertical heat flux	W/m2
intqi	Vertically integrated cloud ice	kg/m2

intql	Vertically integrated cloud water	kg/m2
intqr	Vertically integrated rain	kg/m2
intqg	Vertically integrated graupel	kg/m2
intqs	Vertically integrated snow	kg/m2
intqv	Vertically integrated water vapor	kg/m2
iw	Ice water path	kg/m2
ke	Total KE	m2/s2
lw	Liquid water path	kg/m2
m10C_reflectivity	Reflectivity at -10C level	m
max_reflectivity	Stoelinga simulated maximum composite reflectivity	dBz
mdt	Dt/Dt: fast moist phys	deg/s
mq	Mountain torque	Hadleys/A
o3w	Vertical ozone flux	kg/m2/s
omega	Omega	Pa/s
omg	-mb omega	Pa/s
omg_plev	Omega	Pa/s
oro	Land-water mask	-
pfhy	Hydrostatic pressure	pa
pfnh	Non-hydrostatic pressure	pa
pk	Pressure part of the hybrid coordinate	Pascal
pmask	Masking pressure at lowest level	mb
pmaskv2	Masking pressure at lowest level	mb
ppt	Potential temperature perturbation	K
ppt_ic	Initial potential temperature perturbation	K
ps	Surface pressure	Pa
ps_ic	Initial surface pressure	Pa
pv	Potential vorticity	1/s
q	-mb specific humidity	kg/kg
q_plev	Specific humidity	kg/kg
qdt	Dqv/Dt: fast moist phys	kg/kg/sec
qi_dt_phys	Total ice water tendency from physics	kg/kg/s
ql_dt_phys	Total liquid water tendency from physics	kg/kg/s
qlw	Vertical liquid water flux	kg/m2/s
qvw	Vertical water vapor flux	kg/m2/s
qn	Cloud condensate	kg/m/s2
qn200	200mb condensate	kg/m/s2
qn500	500mb condensate	kg/m/s2
qn850	850mb condensate	kg/m/s2
qp	Precip condensate	kg/m/s2
qv_dt_phys	Water vapor specific humidity tendency from physics	kg/kg/s

rain5km	5-km AGL liquid water	kg/kg
reflectivity	Stoelinga simulated reflectivity	dBz
rh	Relative humidity	%
rh10	10-mb relative humidity	%
rh50	50-mb relative humidity	%
rh100	100-mb relative humidity	%
rh200	200-mb relative humidity	%
rh250	250-mb relative humidity	%
rh300	300-mb relative humidity	%
rh500	500-mb relative humidity	%
rh700	700-mb relative humidity	%
rh850	850-mb relative humidity	%
rh925	925-mb relative humidity	%
rh1000	1000-mb relative humidity	%
rh10_cmip	10-mb relative humidity (CMIP)	%
rh50_cmip	50-mb relative humidity (CMIP)	%
rh100_cmip	100-mb relative humidity (CMIP)	%
rh250_cmip	250-mb relative humidity (CMIP)	%
rh300_cmip	300-mb relative humidity (CMIP)	%
rh500_cmip	500-mb relative humidity (CMIP)	%
rh700_cmip	700-mb relative humidity (CMIP)	%
rh850_cmip	850-mb relative humidity (CMIP)	%
rh925_cmip	925-mb relative humidity (CMIP)	%
rh1000_cmip	1000-mb relative humidity (CMIP)	%
s200	200-mb wind speed	m/s
sgh	Terrain standard deviation	m
sl12	12th L wind speed	m/s
sl13	13th L wind speed	m/s
slp	Sea-level pressure	mb
sphum_ic	Initial surface pressure	Pa
srh01	0–1km storm relative helicity	m/s2
srh03	0–3km storm relative helicity	m/s2
srh25	2–5km storm relative helicity	m/s2
t	-mb temperature	K
t_plev	Temperature	K
T_dt_phys	Temperature tendency from physics	K/s
tb	Lowest layer temperature	K
te	Total energy	J/kg
temp	Temperature	K
theta_e	θ_e	K

tm	Mean 300–500mb temperature	K
tq	Total water path	kg/m2
ts	Skin temperature	K
u	-mb u	m/s
u_plev	Zonal wind	m/s
u_dt_phys	Zonal wind tendency from physics	m/s/s
u100m	100m AGL *u*-wind	m/s
ua_ic	Initial zonal wind	m/s
ucomp	Zonal wind	m/s
uh03	0–3km updraft helicity	m/s2
uh25	2–5km updraft helicity	m/s2
us	Surface *u*-wind	m/s
ustm	*u*-component of storm motion	m/s
uw	Vertical zonal momentum flux	N/m2
v	-mb v	m/s
v_plev	Meridional wind	m/s
v_dt_phys	Meridional wind tendency from physics	m/s/s
v100m	100m AGL *v*-wind	m/s
va_ic	Initial meridional wind	m/s
vcomp	Meridional wind	m/s
vort	Vorticity	1/s
vort200	200-mb vorticity	1/s
vort500	500-mb vorticity	1/s
vort850	850-mb vorticity	1/s
vorts	Surface vorticity	1/s
vs	Surface *v*-wind	m/s
vstm	*v*-component of storm motion	m/s
vw	Vertical meridional momentum flux	N/m2
w	Vertical wind	m/s
w100m	100m AGL *w*-wind	m/s
w200	200-mb *w*-wind	m/s
w500	500-mb *w*-wind	m/s
w700	700-mb *w*-wind	m/s
w850	850-mb *w*-wind	m/s
w5km	5km AGL *w*-wind	m/s
w2500	2.5km AGL *w*-wind	m/s
w1km	1km AGL *w*-wind	m/s
wmaxdn	Column-maximum downdraft	m/s
wmaxup	Column-maximum updraft	m/s
ws	Terrain W	m/s

x850	850-mb vertical comp. of helicity	m/s2
z	-mb height	m
ze	Hybrid z surface	m
zratio	Nonhydro ratio	-
zs	Original mean terrain	m
zsurf	Surface height	m

Table 2.49: Fields in 'dynamics'

(b) 'gfs_dyn' ('FV3/atmos_cubed_sphere/driver/fvGFS/fv_nggps_ diag.F90'):

Field name	Description	Unit
delp	Pressure thickness	pa
delz	Height thickness	m
diss_est	Dissipation estimate	
dq3dt_nophys	Water vapor specific humidity tendency due to non-physics processes	kg/kg/s
dq3dt_o3nophys	Ozone concentration tendency tendency due to non-physics processes	kg/kg/s
dt3dt_nophys	Temperature tendency due to non-physics processes	K/s
du3dt_nophys	*u*-momentum tendency due to non-physics processes	m/s2
dv3dt_nophys	*v*-momentum tendency due to non-physics processes	m/s2
hs	Surface geopotential height	gpm
maxvort01	Maximum hourly 0–1km vertical vorticity	1/s
maxvort02	Maximum hourly 0–2km vertical vorticity	1/s
maxvorthy1	Maximum hourly hybrid level vertical vorticity	1/s
omga	Vertical pressure velocity	pa/s
pfhy	Hydrostatic pressure	pa
pfnh	Non-hydrostatic pressure	pa
ps	Surface air pressure	pa
reflectivity	Stoelinga simulated reflectivity	dBZ
srh01	0-1km storm rel. helicity	m/s2
srh03	0-3km storm rel. helicity	m/s2
temp	Temperature	K
ucomp	Zonal (longitudinal) wind	m/s
uhmax03	Maximum hourly max. 0–3km updraft helicity	m/s2
uhmax25	Maximum hourly max. 2–5km updraft helicity	m/s2
uhmin03	Maximum hourly min. 0–3km updraft helicity	m/s2
uhmin25	Maximum hourly min. 2–5km updraft helicity	m/s2
ustm	*u*-comp. of storm motion	m/s
vcomp	Meridional (latitudinal) wind	m/s
vstm	*v*-comp. of storm motion	m/s
w	Vertical wind velocity	m/s

wmaxdn	Maximum hourly downdraft velocity	m/s
wmaxup	Maximum hourly updraft velocity	m/s
"tracers"	See Table 2.53	

Table 2.50: Fields in 'gfs_dyn'

(c) 'gfs_phys' ('FV3/ccpp/driver/GFS_diagnostics.F90'):

Field name	Description	Unit
ALBDO_ave	Surface albedo	%
AOD_550	Total aerosol optical depth at 550nm	
acond	Aerodynamic conductance	m/s
BC_AOD_550	Soot aerosol optical depth at 550 nm	
ca_deep	CA deep conv	%
ca_micro	CA microphys	%
ca_rad	CA radiation	%
ca_shal	CA shallow conv	%
ca_turb	CA turbulence	%
ca1	Cellular Automata	%
cduvb_ave	Clear sky UV-B downward solar flux	W/m2
cfrzr_ave	Averaged categorical freezing rain	
cicep_ave	Averaged categorical sleet	
cldfra	Instantaneous 3D cloud fraction	
CLDFRA_BL	Subgrid cloud fraction	
cnvw	Subgrid scale convective cloud water	kg/kg
cnvprcp	Surface convective precipitation rate	kg/m2/s
cnvprcp_ave	Averaged surface convective precipitation rate	kg/m2/s
cnvprcpb_ave	Averaged bucket surface convective precipitation rate	kg/m2/s
cpofp	Percent of frozen precipitation	
crain_ave	Averaged categorical rain	
csdlf_ave	Clear sky downward long wave flux	W/m2
csdsf_ave	Clear sky downward short wave flux	W/m2
csnow_ave	Averaged categorical snow	
csulf_ave	Clear sky upward long wave flux	W/m2
csulf_avetoa	Clear sky upward long wave flux: top of atmosphere	
csusf_avetoa	Clear sky upward short solar flux: top of atmosphere	W/m2
csusf_ave	Clear sky upward short solar flux	W/m2
cwork_ave	Cloud work function (valid only with sas)	J/kg
det_sqv	Detrainment water vapor tendency (mynn)	kg kg-1 s-1
det_thl	Detrainment temperature tendency (mynn)	K s-1
DLWRF	Surface downward longwave flux	W/m2

DLWRFI	Instantaneous surface downward longwave flux	W/m2
dlwsfc	Time accumulated downward lw flux at surface	w/m2
dlwsfci	Instantanerous sfc upward lw flux	w/m2
dpt2m	2 meter dew point temperature	K
dq3dt_pbl	Water vapor specific humidity tendency due to PBL	kg/kg/s
dq3dt_deepcnv	Water vapor specific humidity tendency due to deep convection	kg/kg/s
dq3dt_shalcnv	Water vapor specific humidity tendency due to shallow convection	kg/kg/s
dq3dt_mp	Water vapor specific humidity tendency due to microphysics	kg/kg/s
dq3dt_o3column	Ozone concentration tendency due to overhead ozone column	kg/kg/s
dq3dt_o3mix	Ozone concentration tendency tendency due to ozone mixing ratio	kg/kg/s
dq3dt_o3pbl	Ozone mixing ratio tendency due to PBL	kg/kg/s
dq3dt_o3prodloss	Ozone concentration tendency due to production and loss rate	kg/kg/s
dq3dt_o3temp	Ozone concentration tendency due to temperature	kg/kg/s
dq3dt_phys	Water vapor specific humidity tendency due to physics	kg/kg/s
dq3dt_o3phys	Ozone concentration tendency due to physics	kg/kg/s
DSWRF	Averaged surface downward shortwave flux	W/m2
DSWRFI	Instantaneous surface downward shortwave flux	W/m2
DSWRFtoa	Top of atmos. downward shortwave flux	W/m2
dswsfci	Instantaneous sfc downward sw flux	
dt3dt_cnvgwd	Temperature tendency due to convective gravity wave drag	K/s
dt3dt_deepcnv	Temperature tendency due to deep convection	K/s
dt3dt_lw	Temperature tendency due to long wave radiation	K/s
dt3dt_mp	Temperature tendency due to microphysics	K/s
dt3dt_orogwd	Temperature tendency due to orographic gravity wave drag	K/s
dt3dt_pbl	Temperature tendency due to PBL	K/s
dt3dt_pbl_ugwp	T-tendency due to PBL physics	K/s
dt3dt_phys	Temperature tendency due to physics	K/s
dt3dt_rdamp	Temperature tendency due to Rayleigh damping	K/s
dt3dt_shalcnv	Temperature tendency due to shallow convection	K/s
dt3dt_sw	Temperature tendency due to short wave radiation	K/s
DU_AOD_550	Dust aerosol optical depth at 550 nm	
du3dt_cnvgwd	u-momentum tendency due to convective gravity wave drag	m/s2
du3dt_deepcnv	u-momentum tendency due to deep convection	m/s2
du3dt_mtb	axz_oro averaged E-W MTB-tendency	m/s/s
du3dt_ngw	axz_oro averaged E-W GWALL-tendency	m/s/s
du3dt_ogw	axz_oro averaged E-W OROGW-tendency	m/s/s
du3dt_orogwd	u-momentum tendency due to orographic gravity wave drag	m/s2
du3dt_pbl	u-momentum tendency due to PBL	m/s2
du3dt_pbl_ugwp	U-tendency due to PBL physics	m/s/s
du3dt_phys	u-momentum tendency due to physics	m/s2

du3dt_rdamp	*u*-momentum tendency due to convective gravity wave drag	m/s2
du3dt_shalcnv	*u*-momentum tendency due to shallow convection	m/s2
du3dt_tms	axz_oro averaged E-W TMS-tendency	m/s/s
dudt_tot	dudt_tot averaged E-W dycore-tendency	m/s/s
dtdt_tot	dtdt_tot averaged Temp dycore-tendency	m/s/s
dusfc	Surface zonal momentum flux	N/m2
duvb_ave	UV-B downward solar flux	W/m2
dv3dt_cnvgwd	*v*-momentum tendency due to convective gravity wave drag	m/s2
dv3dt_deepcnv	*v*-momentum tendency due to deep convection	m/s2
dv3dt_orogwd	*v*-momentum tendency due to orographic gravity wave drag	m/s2
dv3dt_pbl	*v*-momentum tendency due to PBL	m/s2
dv3dt_pbl_ugwp	V-tendency due to PBL physics	m/s/s
dv3dt_phys	*v*-momentum tendency due to physics	m/s2
dv3dt_rdamp	*v*-momentum tendency due to convective gravity wave drag	m/s2
dv3dt_shalcnv	*v*-momentum tendency due to shallow convection	m/s2
dvsfc	Surface meridional momentum flux	N/m2
edmf_a	Updraft area fraction (from mynn)	
edmf_ent	Updraft entrainment rate (mynn)	m-1
edmf_qc	Mean updraft liquid water (mynn)	kg/kg
edmf_qt	Updraft total water (mynn)	kg/kg
edmf_thl	Mean liquid potential temperature (mynn)	K
edmf_w	Mean updraft vertical velocity (mynn)	m s-1
EL_PBL	Turbulent mixing length	m
evbs_ave	Direct evaporation from bare soil - GFS lsm	W/m2
evcw_ave	Canopy water evaporation - GFS lsm	W/m2
fldcp	Field capacity fraction	
graupel	Graupel fall at this time step	
gflux_ave	Surface ground heat flux	W/m2
gfluxi	Instantaneous surface ground heat flux	W/m2
hgt_hyblev1	Height of hybrid layer 1	m
hpbl	Planetary boundary layer height	
ice	Ice fall at this time step	
lhtfl	Instantaneous surface latent heat net flux	W/m2
lhtfl_ave	Surface latent heat net flux	w/m2
nbdsf_ave	Near IR beam downward solar flux	W/m2
nddsf_ave	Near IR diffuse downward solar flux	W/m2
nifa	Number concentration of ice-friendly aerosols	kg-1
nwfa	Number concentration of water-frriendly aerosols	kg-1
OC_AOD_550	Waso aerosol optical depth at 550 nm	
pevpr	Instantaneous surface potential evaporation	W/m2

pevpr_ave	Averaged potential evaporation rate	W/m2
PRES_avehcb	Pressure high cloud bottom level	%
PRES_avehct	Pressure high cloud top level	%
PRES_avelcb	Pressure low cloud bottom level	%
PRES_avelct	Pressure low cloud top level	%
PRES_avemcb	Pressure middle cloud bottom level	%
PRES_avemct	Pressure middle cloud top level	%
PREScnvclb	Pressure at convective cloud bottom level	pa
PREScnvclt	Pressure at convective cloud top level	pa
QC_BL	Subgrid cloud mixing ratio	
QKE	2×TKE	m2 s-2
psmean	Surface pressure	kPa
psurf	Surface pressure	
pwat	Atmos column precipitable water	kg/m2
rain	Total rain at this time step	
rainc	Convective rain at this time step	
refdmax	Max. hourly 1 km agl reflectivity	dBZ
refdmax263k	Max. hourly -10C reflectivity	dBZ
refl_10cm	Radar reflectivity	dBz
rh02max	Max. hourly 2m RH	%
rh02min	Min. hourly 2m RH	%
sbsno_ave	Sublimation (evaporation from snow) - GFS lsm	W/m2
sfexc	Exchange coefficient	kg/m2/s
shtfl	Instantaneous surface sensible heat net flux	W/m2
shtfl_ave	Surface sensible heat flux	w/m2
shum_wts	Perturbation velocity	m/s
skebu_wts	Perturbation u-velocity	m/s
skebv_wts	Perturbation v-velocity	m/s
snohf	Snow phase-change heat flux - GFS lsm	W/m2
snow	Snow fall at this time step	
snowc_ave	Snow cover - GFS lsm	%
soilm	Total column soil moisture content	kg/m2
spd10max	Hourly maximum wind speed	m/s
spfh_hyblev1	Specific humidity of hybrid layer 1	kg/kg
spfhmax2m	Maximum specific humidity	kg/kg
spfhmin2m	Minimum specific humidity	kg/kg
sppt_wts	Perturbation velocity	m/s
SS_AOD_550	Salt aerosol optical depth at 550 nm	
ssrun_acc	Surface storm water runoff - GFS lsm	kg/m2
SU_AOD_550	Suso aerosol optical depth at 550 nm	

sunsd_acc	Sunshine duration	s
t02max	Max. hourly 2m Temperature	K
t02min	Min. hourly 2m Temperature	K
tau_mtb	ORO-MTB integrated flux from surface	N/m2
tau_ngw	NGW momentum flux at launch level	N/m2
tau_ogw	OGW vertical MF at launch level	N/m2
tau_tofd	ORO-TOFD integrated flux from surface	N/m2
tav_ugwp	T-daily mean for UGWP	K
TCDC_avebndcl	Boundary layer cloud layer total cloud cover	%
TCDC_aveclm	Atmos column total cloud cover	%
TCDC_avehcl	High cloud level total cloud cover	%
TCDC_avelcl	Low cloud level total cloud cover	%
TCDC_avemcl	Mid cloud level total cloud cover	%
TCDCcnvcl	Total cloud cover	
TEMP_avehct	Temperature high cloud top level	K
TEMP_avelct	Temperature low cloud top level	K
TEMP_avemct	Temperature middle cloud top level	K
tmp_hyblev1	Temperature of hybrid level 1	K
tmpmax2m	Max temperature at 2m height	K
tmpmin2m	Min temperature at 2m height	K
totgrp_ave	Surface graupel precipitation rate	kg/m2/s
totgrpb_ave	Bucket surface graupel precipitation rate	kg/m2/s
totice_ave	Surface ice precipitation rate	kg/m2/s
toticeb_ave	Bucket surface ice precipitation rate	kg/m2/s
totprcp_ave	Surface precipitation rate	kg/m2/s
totprcpb_ave	Bucket surface precipitation rate	kg/m2/s
totsnw_ave	Surface snow precipitation rate	kg/m2/s
totsnwb_ave	Bucket surface snow precipitation rate	kg/m2/s
trans_ave	Transpiration - GFS lsm	W/m2
u10m	u-component of 10m wind	m/s
u10max	Hourly maximum magnitude of u-wind	m/s
u10mmax	Maximum magnitude of u-wind	m/s
uav_ugwp	U-daily mean for UGWP	m/s
ugrd_hyblev1	Zonal wind speed of hybrid layer 1	m/s
ULWRF	Surface upward longwave flux	W/m2
ULWRFI	Instantaneous surface upward longwave flux	W/m2
ULWRFtoa	Top of atmos. upward longwave flux	W/m2
ulwsfc	Time accumulated upward lw flux at surface	W/m2
ulwsfci	Instantaneous sfc upward lw flux	W/m2
USWRF	Averaged surfae upward shortwave flux	W/m2

USWRFI	Instantaneous surface upward shortwave flux	W/m2
USWRFtoa	Top of atmos. upward shortwave flux	W/m2
u-gwd_ave	surface zonal gravity wave stress	N/m2
uswsfci	Instantaneous sfc upward sw flux	W/m2
vbdsf_ave	Visible beam downward solar flux	W/m2
vddsf_ave	Visible diffuse downward solar flux	W/m2
v-gwd_ave	surface meridional gravity wave stress	N/m2
v10m	v-component of 10m wind	m/s
v10max	Hourly maximum magnitude of v-wind	m/s
v10mmax	Maximum magnitude of v-wind	m/s
vgrd_hyblev1	Meridional wind speed of hybrid layer 1	m/s
watr_acc	Total water runoff	kg/m2
wet1	Normalized soil wetness	
wilt	Wilting point (volumetric)	
wind10mmax	Maximum wind speed	m/s
zlwb	Height of LWB-level	m
zmtb	Height of dividing streamline	m
zmtnblck	Level of dividing streamline	m/s
zogw	Height of OGW-launch	m

Table 2.51: Fields in 'gfs_phys'

(d) 'gfs_sfc' ('FV3/ccpp/driver/GFS_diagnostics.F90'):

Field name	Description	Unit
alnsf	Mean Near-IR albedo with strong cosz dependency	%
alnwf	Mean Near-IR albedo with weak cosz dependency	%
alvsf	Mean visible albedo with strong cosz dependency	%
alvwf	Mean visible albedo with weak cosz dependency	%
c_0	NSST coefficient 1 for d(tz)/d(ts)	
c_d	NSST coefficient 2 for d(tz)/d(ts)	
canopy	Canopy water (cnwat in gfs data)	
crain	Categorical rain (yes=1, no=0)	
d_conv	NSST thickness of free convective layer	m
dt_cool	Sub-layer cooling thickness	k
f10m	10m wind speed over lowest value	
facsf	Fractional coverage with strong cosz dependency	
facwf	Fractional coverage with weak cosz dependency	
ffhh	Surface exchange coefficient for heat	
ffmm	Surface exchange coeff. for momentum	
fice	Surface ice concentration (ice=1, no ice=0)	

flhc	Surface exchange coefficient for heat	W m-2 K-1
flqc	Surface exchange coefficient for moisture	kg m-2 s-1
hgtsfc	Surface altitude	gpm
hice	Sea ice thickness (iceth in gfs data)	
ifd	NSST index to start dtlm run or not	
maxmf	Maixmum mass-flux in column	m/s
nifa2d	Ice-friendly surface aerosol source	kg-1 s-1
nwfa2d	Water-friendly surface aerosol source	kg-1 s-1
q2m	2 meter specific humidity	kg/kg
qrain	Sensible heat flux due to rainfall	W/m2
uustar	Friction velocity	
shdmax	Maximum areal fractional coverage of annual green vegetation	
shdmin	Minimum areal fractional coverage of annual green vegetation	
slmsksfc	Sea-land-ice mask (0: sea, 1: land, 2: ice)	
slope	Surface slope type	
snoalb	Max. snow albedo over land in fraction	
snowd	Surface snow depth	m
snowfall_acc_ice	Total accumulated frozen precipitation over ice	kg/m2
snowfall_acc_land	Total accumulated frozen precipitation over land	kg/m2
soilt1	Soil column temperature 1	K
soilt2	Soil column temperature 2	K
soilt3	Soil column temperature 3	K
soilt4	Soil column temperature 4	K
soilw1	Total volumetric soil moisture 1	
soilw2	Total volumetric soil moisture 2	
soilw3	Total volumetric soil moisture 3	
soilw4	Total volumetric soil moisture 4	
stype	Soil type	
sub_sqv	Subsidence water tendency (mynn)	kg kg-1 s-1
sub_thl	Subsidence temperature tendency (mynn)	K/s
t2m	2 meter temperature	K
tg3	Deep soil temperature	K
tiice	Internal ice temperature layer	K
tisfc	Surface temperature over ice fraction	K
tprcp	Total precipitation	kg/m2
tref	NSST reference or foundation temperature	K
tsfc	Sea surface temperature	K
vfracsfc	Vegetation frraction	
vtype	Vegetation type	
xlaixy	Leaf area index	

w_0	NSST coefficient 3 for d(tz)/d(ts)	
w_d	NSST coefficient 4 for d(tz)/d(ts)	
weasd	Snow-liquid equivalent	kg/m2
xs	NSST salinity content in DTL	
xt	NSST heat content in diurnal thermocline layer	k·m
xtts	NSST d(xt)/d(ts)	m
xu	NSST u-current content in DTL	m2/s
xv	NSSTv-current content in DTL	m2/s
xz	NSST diurnal thermocline layer thickness	m
xzts	NSST d(xz)/d(ts)	m/k
z_c	NSST sub-layer cooling thickness	m
zm	NSST mixed layer thickness	m
zol	Monin obukhov surface stability parameter	
zorlsfc	Surface roughness length	m

Table 2.52: Fields in 'gfs_sfc'

2.6.5 field_table.[CCPP]

The FMS field and tracer managers manage the tracers in 'field_table'.

Type	Model	Parameter	Long name	Unit	Profile
TRACER	atmos_mod	sphum	Specific humidity	kg/kg	fixed
TRACER	atmos_mod	liq_wat	Cloud-water mixing ratio	kg/kg	fixed
TRACER	atmos_mod	ice_wat	Cloud-ice mixing ratio	kg/kg	fixed
TRACER	atmos_mod	rainwat	Rain-water mixing ratio	kg/kg	fixed
TRACER	atmos_mod	snowwat	Snow-water mixing ratio	kg/kg	fixed
TRACER	atmos_mod	graupel	Graupel mixing ratio	kg/kg	fixed
TRACER	atmos_mod	water_nc	Cloud liquid water number concentration	$/kg$	fixed
TRACER	atmos_mod	ice_nc	Cloud ice water number concentration	$/kg$	fixed
TRACER	atmos_mod	rain_nc	Rain number concentration	$/kg$	fixed
TRACER	atmos_mod	o3mr	Ozone mixing ratio	kg/kg	fixed
TRACER	atmos_mod	cld_amt	Cloud amout	1	fixed
TRACER	atmos_mod	liq_aero	Water-friendly aerosol number concentration	$/kg$	fixed
TRACER	atmos_mod	ice_aero	Ice-friendly aerosol number concentration	$/kg$	fixed
TRACER	atmos_mod	sgs_tke	Subgrid scale turbulent kinetic energy	$m2/s2$	fixed

Table 2.53: Tracers

where the options for 'Profile' are listed as follows:

Method	Method control
fixed	surface_value=A
profile	surface_value=A, top_value=B (exponential profile)

Table 2.54: Options for 'profile_type'

2.6.6 FV3.input.yml

This is an input file for the 'set_namelist.py' script to modify the base template into the case-specific 'input.nml' file. Since the base template is for the 'FV3_GSD_v0' CCPP suite, its 'FV3_GSD_v0' section is empty. For other CCPP suites, the values specified in this file update the corresponding values in the 'input.nml' file (Section 2.6.8). Make sure that you should modify this file for the specific namelist options of your experiment.

2.6.7 FV3LAM_wflow.xml

This is a Jinja-enabled Rocoto XML template. It is filled in using the 'fill_template.py' python script that is called in the 'generate_workflow.sh' script as shown in Figure 2.3. The python script finds all the template parameters and renders the desired parameters. In the template file, several place holders are shown as in Figure 2.55. The instruction script 'README.xml_templating.md' can be found in '{HOMErrfs}/ush/templates/'.

Place holder	Description
{% ⋯ %}	For statements of the structures such as conditionals ('if') and loops ('for')
{{ ⋯ }}	For expressions to fill the parameters
{# ⋯ #}	For comments not includes in the template output

Table 2.55: Jinja templating for 'FV3LAM_wflow.xml': place holders

2.6.8 input.nml.FV3

This is a template file for the namelists of the weather model including *FV3* and CCPP. This template is based on the CCPP option of 'FV3_GSD_v0'. For other CCPP options, their namelists are modified by the '{HOMErrfs}/ush/templates/FV3.input.yml' file that are called by the '{HOMEr-

rfs}/ush/set_namelist.py' script. To make the sort order of the output parameters the same as those in the template, you should modify the 'set_namelist.py' as described in 2.6.2.

References:

- UFS FV3-Dycore documentation (https://noaa-emc.github.io/FV3_Dycore_ufs-v1.0.0/html/index.html)
- CCPP Scientific documentation (https://dtcenter.ucar.edu/GMTB/v4.0/sci_doc/CCPPsuite_nml_desp.html)
- UFS Stochastic-physics documentation (https://stochastic-physics.readthedocs.io/en/ufs-v1.0.0/namelist_options.html)

1. Observed SST and ice mask data set interpolated onto the grid ('amip_interp_nml'):

Parameter	Description	Default
data_set	Name / type of SST data ('amip1', 'reynolds_eof', 'reynolds_oi', 'hurrell', 'daily')	'amip1'
date_out_of_range	Controls the use of climatological monthly mean data when the requested date falls outside the range of the data set • 'fail': will fail if date is not in the data set period • 'initclimo': uses climatological data only if date is prior to data set period • 'climo': uses climatological data anytime	'fail'
interp_oi_sst		false
no_anom_sst		true
use_ncep_ice		false
use_ncep_sst		false

Table 2.56: Input namelist: Observed SST and ice mask data set (FMS)

2. Namelist of atmosphere model ('&atmos_model_nml'):

Parameter	Description	Data type
chksum_debug	Whether to compute checksums for all variables passed into the GFS physics, before and after each physics timestep	Logical
dycore_only	Whether only the dynamical core (and not the GFS physics) is executed when running the model	Logical
fdiag	Array listing the diagnostic output times for the GFS physics	Integer

Table 2.57: Input namelist: atmosphere model

3. Namelist of CIRES Unified Gravity Wave Physics module ('&cires_ugwp_nml'):

Parameter	Description	Default
knob_ugwp_azdir	4D array that defines number of azimuths for propagation of GWs triggered by orography, convection, front-jets, and dynamical imbalance	2,4,4,4
knob_ugwp_doaxyz	Parameter controls application of the momentum deposition for NGW-schemes • 0: the momentum tendencies due to NGWs are calculated, but tendencies do not change the horizontal winds • 1: It changes the horizontal momentum tendencies and horizontal winds	1
knob_ugwp_doheat	Parameter controls application of the heat deposition for NGW-schemes • 0: the temperature tendencies due to NGWs are calculated but tendencies do not change the temperature state • 1: it changes the temperature tendencies and kinetic temperature	1
knob_ugwp_dokdis	Parameter controls application of the eddy diffusion due to instability of NGWs • 0: the eddy diffusion tendencies due to NGWs are calculated but tendencies do not change the model state vector • 1: it computes eddy diffusion coefficient due to instability of NGWs • 2: represents the spectral deterministic solver with background dissipation and spectral saturation • 3: represents the discrete multi-wave solver with the background dissipation, extension of Alexander sand Dunkerton (1999) • 4: represents the spectral solver with background dissipation, extension of Doppler Spread Theory of Hines (1997)	0
knob_ugwp_effac	4D array that controls efficiency of GWs triggered by four types of physics-based sources	1.,1.,1.,1.
knob_ugwp_ndx4lh	Parameter controls the selection of the horizontal wavenumber (wavelength) for NGW schemes	2
knob_ugwp_solver	Parameter controls the selection of UGWP-solvers (wave propagation, dissipation and wave breaking) for NGWs • 1: represents the discrete multi-wave solver with background dissipation and linear wave saturation	1
knob_ugwp_source	4D array for oro, fronts, conv, imbf-owp (1:active, 0:off)	1,0,1,0
knob_ugwp_stoch	4D array that control stochastic selection of GWs triggered by four types of physics-based sources	0,0,0,0
knob_ugwp_version	Parameter selects a version of the UGWP implementation in FV3GFS-127L	0
knob_ugwp_wvspec	4D array defines number of waves in each azimuthal propagation for GWs excited due to the following four sources: (1) sub-grid orography, (2) convective, (3) frontal activity, and (4) number of wave excited by dynamical imbalances	1,32,32,32
launch_level	It defines the interface model level from the surface at which NGWs are launched.	55

Table 2.58: Input namelist: CIRES Unified Gravity Wave Physics module

4. Namelist of diagnostics manager ('&diag_manager_nml'):

Parameter	Description	Data type
prepend_date	Whether the history file has the start date prepended to the file name	Logical

Table 2.59: Input namelist: diagnostics manager

5. Namelist of the external initial condition (‘&external_ic_nml’):

Parameter	Description	Data type
checker_tr	Whether to enable the ‘checkerboard’ tracer test	Logical
filtered_terrain	Whether to use the terrain filtered by the pre-processing tools rather than the raw terrain.	Logical
gfs_dwinds		Logical
levp	Number of levels in the input initial conditions	Integer
nt_checker	Number of tracers to initialize with an idealized ‘checkerboard’ pattern	Integer

Table 2.60: Input namelist: external initial condition

6. Namelist of FMS IO (‘&fms_io_nml’):

Parameter	Description	Default
checksum_required	Compares checksums stored in the attribute of a field against the checksum after reading in the data	true
max_files_r		40
max_files_w		40

Table 2.61: Input namelist: FMS IO

7. Namelist of FMS (‘&fms_nml’):

Parameter	Description	Default
clock_gain	Level of clock granularity used for performance timing sections of the code	‘NONE’
domains_stack_size	Size (in bytes) of memory array reserved for domains	0
print_memory_usage	If true, memory usage statistics will be printed at various points in the code	false

Table 2.62: Input namelist: FMS

8. Namelist of *FV3* Dycore (‘&fv_core_nml’):

Parameter	Description	Default
a_imp	Controls behavior of the non-hydrostatic solver.	0.75

adjust_dry_mass	Whether to adjust the global dry-air mass to the value set by dry_mass. This is only done when using an initial condition from an external data set, interpolated from another resolution.	false
beta	Parameter specifying fraction of time-off-centering for backwards evaluation of the pressure gradient force.	0.0
consv_am	Whether to enable angular momentum fixer.	false
consv_te	Fraction of total energy lost during the adiabatic integration between calls of the physics, to be added back globally as heat; essentially the strength of the energy fixer in the physics.	0.0
d2_bg	Coefficient for background second-order divergence damping	0.0
d2_bg_k1	Strength of second-order diffusion in the top sponge layer	4.0
d2_bg_k2	Strength of second-order diffusion in the second sponge layer from the model top	2.0
d4_bg	Dimensionless coefficient for background higher-order divergence damping	0.16
d_con	Fraction of kinetic energy lost to explicit damping to be converted to heat.	0.0
d_ext	Coefficient for external barotropic mode damping	0.02
dddmp	Dimensionless coefficient for the second-order Smagorinsky-type divergence damping	0.0
delt_max	Maximum allowed magnitude of the dissipative heating rate.	1.0
dnats	Number of tracers which are not to be advected by the dynamical core, but still passed into the dynamical core.	0
do_sat_adj	Same as 'fast_sat_adj=.false.' has fast saturation adjustments	false
do_schmidt	Enable grid stretching and rotation using 'stretch_fac', 'target_lat', and 'target_lon'	false
do_vort_damp	Whether to apply flux damping to the fluxes of vorticity, air mass, and non-hydrostatic vertical velocity	false
dwind_2d	Whether to use a simpler and faster algorithm for interpolating the A-grid wind tendencies computed from the physics to the D-grid.	false
external_eta	If .true., reads the interface coefficients a_k and b_k from either the restart file or from the external initial condition file.	false
external_ic	Initialize the model using the data in the externally specified file in 'res_latlon_dynamics'	false
fill	FIlls in negative tracer values by taking positive tracers from the cells above and below.	false
full_zs_filter	Whether to apply the on-line topography filter during initialization. Only active if 'get_nggps_ic=.true.'	false
fv_debug	Whether to turn on additional diagnostics in fv_dynamics.	false
fv_sg_adj	Timescale at which to remove two-delta-z instability when the local Richardson number is less than 1.	-1
gfs_phil	If true, compute geopotential inside of GFS physics	false
hord_dp	Horizontal advection scheme for mass.	9
hord_mt	Horizontal advection scheme for momentum fluxes.	9
hord_tm	Horizontal advection scheme for potential temperature and layer thickness in non-hydrostatic simulations.	9
hord_tr	Horizontal advection scheme for tracers.	12

hord_vt	Horizontal advection scheme for absolute vorticity and for vertical velocity in non-hydrostatic simulations.	9
hydrostatic	Whether to use the hydrostatic or non-hydrostatic solver.	true
io_layout	Layout of output files on each tile. • 1,1 : Combines all restart and history files on a tile into one file • 0,0 : Every process writes out its own restart and history files	Integers (N_1, N_2)
k_split	Number of vertical remappings per 'dt_atmos' (physical time step)	1
ke_bg	Background KE production over a small step	0.0
kord_mt	Vertical remapping scheme for wind	8
kord_tm	Vertical remapping scheme for temperature	-8
kord_tr	Vertical remapping scheme for tracers	8
kord_wz	Vertical remapping scheme for vertical velocity in non-hydrostatic simulation.	8
make_nh	Whether to re-initialize the nonhydrostatic state, by re-computing dz from hydrostatic balance and setting w to zero.	false
mountain	Takes topography into account when initializing the model. Set this to true to apply the terrain filter (if 'n_zs_filter'=2 or 4) upon startup.	true
n_split	Number of small dynamics (acoustic) time steps between vertical remapping	0
n_sponge	Controls the number of layers at the upper boundary on which the $2\Delta x$ filter is applied	0
n_zs_filter	Number of times to apply a diffusive filter to the topography upon startup if mountain is .true. and the model is not being cold-started.	0
na_init	Number of forward-backward dynamics steps used to initialize adiabatic solver	0
ntiles	Number of tiles on the domain	1
nudge_qv	If .true., the water vapor profile is nudged to an analytic fit to the HALOE climatology.	false
nwat	Number of water species to be included in condensate and water vapor loading	3
p_fac	Safety factor for minimum non-hydrostatic pressures	0.05
phys_hydrostatic	Option to enable hydrostatic application of heating from the physics in a non-hydrostatic simulation.	true
print_freq	Number of hours between print-out of max/min and air/tracer mass diagnostics to standard output (0=no output, -1=every 'dt_atmos')	0
range_warn	Checks whether the values of the prognostic variables are within a reasonable range at the end of a dynamics time step.	false
read_increment	Read in analysis increment and add to restart following are namelist parameters for Stockastic Energy Baskscatter dissipation estimate.	false
regional	Controls whether this is a regional domain	false
res_latlon_dynamics	File name of the input IC file ('INPUT/fv_rst.res.nc')	Character
reset_eta		false
rf_cutoff	Pressure below which no Rayleigh damping is applied if tau>0	30.E2
tau	Time scale for Rayleigh friction applied to horizontal and vertical winds (in days)	0.0
use_hydro_pressure	Whether to compute hydrostatic pressure for input to the physics.	false
vtdm4	Coefficient for background other-variable damping	0.0

warm_start	Whether to start from restart files instead of cold starting	true
z_tracer	Whether to transport sub-cycled tracers layer-by-layer, each with its own computed sub-cycling time step (if q_split=0).	false

Table 2.63: Input namelist: FV3 Dycore

9. Namelist of ('&fv_grid_nml'):

Parameter	Description	Data type
grid_file	If 'grid_type=-1', the name of the 'grid_spec' file to read in 'INPUT/grid_spec.nc' by default	Character

Table 2.64: Namelist of the input grid file

10. Namelist of Physics-related options in CCPP ('&gfs_physics_nml'):

Parameter	Description	Default
bl_mynn_edmf	Flag to activate the mass-flux scheme • 0: deactivate mass-flux scheme • 1: activate dynamic multiplume mass-flux scheme	0
bl_mynn_edmf_mom	Flag to activate the transport of momentum • 0: deactivate momentum transport in mass-flux scheme • 1: activate momentum trnasport in dynamic multiplume mass-flux scheme	1
bl_mynn_tkeadvest	Flag to activate computation of TKE advection	.false.
cal_pre	Flag for calling precipitation type algorithm	.false.
cdmbgwd	Multiplication factors for mountain blocking(1), orographic gravity wave drag (2)	2.0, 0.25, 1.0, 1.0
cnvcld	Flag for convective cloud	.false.
cnvgwd	Flag for convective gravity wave drag scheme dependent on maxval	.false.
cplflx	Flag for cplflx collection	.false.
debug	Flag for debug printout	.false.
do_mynnedmf	Flag to activate MYNN-EDMF scheme	.false.
do_mynnsfclay	Flag to activate MYNN-SFCLAY scheme	.false.
do_shum	Flag for stochastic SHUM option	.false.
do_skeb	Flag for stochastic SKEB option	.false.
do_sppt	Flag for stochastic SPPT option	.false.
dspheat	Flag for using TKE dissipative heating to temperature tendency in hybrid EDMF and TKE-EDMF schemes	.false.
fhcyc	Frequency for surface data cycling in hours (should be **0.0**, otherwise some of the surface fields will be replaced with climatology data at the frequency.)	0.0
fhlwr	Frequency for longwave radiation (in seconds)	3600.0
fhswr	Frequency for shortwave radiation (in seconds)	3600.0

fhzero	hour between clearing of diagnostic buckets	0.0
h2o_phys	Flag for stratosphere h2o scheme	.false.
hybedmf	Flag for calling hybrid EDMF PBL scheme	.false.
iaer	4-digit aerosol flag (DABC for aermdl,volcanic,LW,SW)	111
	• D: tropospheric aerosol model scheme (0 or None: opac-climatology aerosol scheme, 1: gocart climatology aerosol scheme, 2: gocart prognostic aerosol scheme, 5: opac-clim new spectral mapping)	
	• A: stratospheric aerosol (0: use background stratospheric aerosol, 1: include stratospheric volcanic aerosol)	
	• B: tropospheric aerosol in LW radiation (0: no, 1: yes)	
	• C: tropospheric aerosol in SW radiation (0: no, 1: yes)	
ialb	Flag for SW surface albedo control	0
	• 0: Climatology surface albedo scheme for SW	
	• 1: MODIS based land surface albedo for SW	
iau_delthrs	Incremental analysis update (IAU) time interval in hous	6
iau_inc_files		Character
iaufhrs	Forecast hours associated with increment files	-1
icloud_bl	Flag for coupling SGS clouds to radiation	1
	• 0: deactivate coupling subgrid clouds to radiation	
	• 1: activate subgrid cloud coupling to radiation	
ico2	CO_2 data source control flag	0
	• 0: prescribed value (380 ppmv)	
	• 1: yearly global averaged annual mean from obs.	
	• 2: monthly 15 degree horizontal resolution from obs.	
iems	Flag for LW surface emissivity control	0
	• a=0: set surface air/ground t same for LW radiation	
	• a=1: set surface air/ground t different for LW radiation	
	• b=0: fixed surface emissivity=1.0 (black body)	
	• b=1: varying climatology surface emissivity (veg)	
imfdeepcnv	Flag for mass-flux deep convective scheme	1
	• -1: Chikira-Sugiyama deep convection (with cscnv=.T.)	
	• 1: July 2010 version of SAS convective scheme	
	• 2: scale- and aerosol-aware mass-flux deep convective scheme	
	• 3: scale- and aerosol-aware Grell-Freitas scheme (GSD)	
	• 4: new Tiedtke scheme (CAPS)	
imfshalcnv	Flag for mass flux shallow convective scheme	1
	• 1: July 2010 version of mass-flux shallow convective scheme	
	• 2: scale- and aerosol-aware mass-flux shallow convective scheme (2017)	
	• 3: scale- and aerosol-aware Grell-Freitas scheme (GSD)	

	• 4: new Tiedtke scheme (CAPS)	
	• 0: modified Tiedtke's eddy-diffusion shallow convective scheme	
	• -1: no shallow convection	
imp_physics	Choice of microphysics scheme	99
	• 11: GFDL microphysics scheme	
	• 8: thompson microphysics scheme	
	• 10: Morrison-Gettleman microphysics scheme	
isol	Solar constant scheme control flag	0
	• 0: fixed 1366.0 Wm^{-2} (old standard)	
	• 10: fixed 1360.8 Wm^{-2} (new standard)	
	• 1: NOAA ABS-scale TSI table (yearly)	
	• 2: NOAA TIM-scale TSI table (yearly)	
	• 3: CMIP5 TIM-scale TSI table (yearly)	
	• 4: CMIP5 TIM-scale TSI table (monthly)	
isot	Flag for soil type dataset choice	0
	• 0: Zobler soil type (9 categories)	
	• 1: STATSGO soil type (19 categories)	
isubc_lw	Subgrid cloud approx. control flag in LW radiation	0
	• 0: no McICA approximation in LW radiation	
	• 1: McICA with prescribed permutation seeds (test)	
	• 2: McICA with randomly generated permutation seeds	
isubc_sw	Subgrid cloud approx. control flag in SW radiation	0
	• 0: no McICA approximation	
	• 1: McICA with prescribed permutation seeds (test)	
	• 2: McICA with randomly generated permutation seeds	
ivegsrc	Flag for vegetation type dataset choice (0: USGS, 1: IGBP, 2: UMD))	2
ldiag3d	Flag for 3D diagnostic fields	Logical
lheatstrg	Flag for canopy heat storage parameterization	Logical
lradar	Flag for computing radar reflectivity in Thompson MP scheme	.false.
lsm	Flag for land surface model (1: Noah LSM, 2: RUC LSM)	1
lsoil_lsm	Number of soil layers internal to land surface model	-1
ltaerosol	Flag for aerosol climatology in Thompson MP scheme	.false.
lwhtr	Flag for output of longwave heating rate	.true.
ncld	Number of hydrometeors	1
nst_anl	Flag for NSST analysis in gcycle/sfcsub	.false.
nstf_name	NSST related parameters	0,0,1,0,5
	• (1): 0=off, 1= on but uncoupled, 2=on and coupled	
	• (2): 1=spin up on, 0=spin up off	
	• (3): 1=analysis on, 0=analysis off	
	• (4): zsea1 in mm	

	• (5): zsea2 in mm	
oz_phys	Flag for old (2006) ozone physics	.true.
oz_phys_2015	Flag for new (2015) ozone physics	.false.
pdfcld	Flag for PDF clouds	.false.
pre_rad	Flag for testing purpose	.false.
prslrd0	Pressure level above which to apply Rayleigh damping	0.0d0
random_clds	Flag for whether clouds are random	.false.
redrag	Flag for applying reduced drag coefficient for high wind over sea in GFS surface layer scheme	.false.
satmedmf	Flag for calling TKE EDMF PBL scheme	.false.
shal_cnv	Flag for calling shallow convection	.false.
swhtr	Flag for output of shortwave heating rate	.true.
trans_trac	Flag for convective transport of tracers	.false.
ttendlim	Temperature tendency limiter per time step in K/s, set to <0 to deactivate	-999.0
use_ufo		Logical

Table 2.65: Input namelist of CCPP

11. Namelist of FMS interpolator ('&interpolator_nml'):

Parameter	Description	Data type
interp_method	interpolation method	Character

Table 2.66: Input namelist of FMS interpolator

12. Namelist of ('&nam_sfcperts'):

Parameter	Description	Default
iseed_sfc	Random seeds	0
nsfcpert	Number of weights for stochastic surface perturbation	0
pertalb	Magnitude of surface albedo perturbation	-999.0
pertlai	Magnitude of perturbation of leaf area index	-999.0
pertshc	Magnitude of perturbation of soil hydraulic conductivity	-999.0
pertvegf	Magnitude of perturbation of vegetation fraction	-999.0
pertz0	Magnitude of perturbation of momentum roughness length	-999.0
pertzt	Magnitude of perturbation of heat to momentum roughness length ratio	-999.0
sfc_lscale	Length scales	-999.0
sfc_tau	Time scales	-999.0
sppt_land		.false.

Table 2.67: Input namelist of

13. Namelist of stochastic physics ('&nam_stochy'):

Parameter	Description	Default
iseed_shum	Seeds for setting the random number sequence	0
iseed_skeb	Seeds for setting the random number sequence	0
iseed_sppt	Seeds for setting the random number sequence	0
new_lscale		Logical
shum	Amplitude of stochastic boundary layer specific humidity perturbations	-999.
shum_lscale	De-correlation spatial scales in meters	-999.
shum_tau	De-correlation time scales in seconds	-999.
shumint	Interval in seconds to update random pattern	Integer
skeb	Stochastic KE backscatter amplitude	-999.
skeb_lscale	De-correlation spatial scales in meters	-999.
skeb_tau	De-correlation timescales in seconds	-999.
skeb_vdof	Number of degrees of freedom in the vertical for the SKEB random pattern	5
skebint	Interval in seconds to update random pattern	Integer
skebnorm	0: random pattern is stream function, 1: kenorm pattern, 2: vorticity pattern	0
sppt	Amplitude of random patterns	-999.
sppt_logit	Logit transform for SPPT to bounded interval [-1,1]	.false.
sppt_lscale	De-correlation spatial scales in meters	-999.0
sppt_sfclimit	Reduce amplitude of SPPT near surface (lowest 2 levels)	.false.
sppt_tau	De-correlation timescales in seconds	-999.
spptint	Interval in seconds to update random pattern	Integer
use_zmtnblck	Flag for mountain blocking	.false.

Table 2.68: Input namelist of stochastic physics

14. Namelist of FIX files ('&namsfc'):
    List of the static fields that are not specified in the regional workflow:

Parameter	Description	Directory
fabsl		
faisl		
faiss		
fnacna		
fnsnoa		
fntsfa		
fnzorc	igbp	
fsicl		
fsics		

fslpl		
fsmcl(2)		
fsmcl(3)		
fsmcl(4)		
fsnol		
fsnos		
fsotl		
ftsfl		
ftsfs		
fvetl		
fvmnl		
fvmxl		
ldebug		Logical

Table 2.69: Input fields: FIX (not used)

Note) The static fields that will be specified in the namelist ('{EXPTDIR}/input.nml') by the script are listed in Table 2.70. Make sure that the file names of static fields correspond to the files in the 'FIX' directory.

Parameter	File name	Directory
fnabsc	C{res}_maximum_snow_albedo.tileX.nc	fix_lam
fnaisc	CFSR.SEAICE.1982.2012.monthly.clim.grb	fix_am
fnalbc	C{res}_snowfree_albedo.tileX.nc	fix_lam
fnalbc2	C{res}.facsf.tileX.nc	fix_lam
fnglac	global_glacier.2x2.grb	fix_am
fnmskh	seaice_newland.grb	fix_am
fnmxic	global_maxice.2x2.grb	fix_am
fnslpc	C{res}.slope_type.tileX.nc	fix_lam
fnsmcc	globla_soilmgldas.t126.384.190.grb	fix_am
fnsnoc	global_snoclim.1.875.grb	fix_am
fnsotc	C{res}.soil_type.tileX.nc	fix_lam
fntg3c	C{res}.substrate_temperature.tileX.nc	fix_lam
fntsfc	RTGSST.1982.2012.monthly.clim.grb	fix_am
fnvegc	C{res}.vegetation_greenness.tileX.nc	fix_lam
fnvetc	C{res}.vegetation_type.tileX.nc	fix_lam
fnvmnc	C{res}.vegetation_greenness.tileX.nc	fix_lam
fnvmxc	C{res}.vegetation_greenness.tileX.nc	fix_lam

Table 2.70: Input fields: FIX

15. Namelist of ('&surf_map_nml'):

Parameter	Description	Data type
cd2		Integer
cd4		Real
max_slope		Real
n_del2_strong		Integer
n_del2_weak		Integer
n_del4		Integer
peak_fac		Real
zero_ocean	Whether to prevent the smoothing from extending topography out into the ocean	Logical

Table 2.71: Input namelist of

2.6.9 model_configure

This file contains settings and configurations for the NUOPC/ESMF main component, including the simulation start time, the processor layout/configuration, and the I/O selections.

Parameter	Description	Type	Default
total_number	Total number of ensemble member	Integer	1
PE_MEMBER01	Total number of tasks for ensemble number 1	Integer	{PE_MEMBER01}
start_year	Start year of simulation	Integer	{start_year}
start_month	Start month of simulation	Integer	{start_month}
start_day	Start day of simulation	Integer	{start_day}
start_hour	Start hour of simulation	Integer	{start_hour}
start_minute	Start minute of simulation	Integer	0
start_second	Start second of simulation	Integer	0
nhours_fcst	Total forecast length	Integer	{nhours_fcst}
RUN_CONTINUE	Flag for more than one NEMS run	Logical	.false.
ENS_SPS	Flag for the ensemble stochastic coupling	Logical	.false.
dt_atmos	Atmosphere time step in second	Integer	{dt_atmos}
cpl	Flag for coupling with MOM/CICE	Logical	.false.
calendar	Type of calendar year	character	'julian'
memuse_verbose	Flag for printing out memory usage	Logical	.false.
atmos_nthreads	Number of threads for atmosphere	Integer	{atmos_nthreads}
use_hyper_thread		Logical	.false.
ncores_per_node	Number of cores per node	Integer	{ncores_per_node}
debug_affinity		Logical	.true.
restart_interval	Frequency to output restart file	Integer	0

output_1st_tstep_rst		Logical	.false.
print_esmf	Flag for ESMF PET files	Logical	{print_esmf}
quilting	Flag to use the write-component for output (in 'filename_base')	Logical	{quilting}
write_groups	Total number of writing groups (groups of MPI tasks) used in the write-component	Integer	{write_groups}
write_tasks_per_group	Total number of tasks in each writing group	Integer	{write_tasks_per_group}
num_files	Total number of history files	Integer	2
filename_base	Name base for history files	Character	'dyn' 'phy'
output_file	Format of history files ('netcdf')	Character	'netcdf'
write_nemsioflip	Flipping vertical levels to nemsio	Logical	.false.
write_fsyncflag	Checking if a file is synced to disk	Logical	.false.
output_grid	Grid type of output files	Character	
nfhout	Output frequency of history files		1
nfhmax_hf	Forecast length of high-frequency history files		60
nfhout_hf	Output frequency of high-frequency history files		1
nsout	Output frequency of number of time step		-1 (turn off)

Table 2.72: Parameters for *FV3* configuration

Write-components:

There are three options for the write component when 'quilting' ='.true.': (1) lambert_conformal, (2) regional_latlon, and (3) rotated_latlon. The domain-specific parameters for a write-component are set in '{HOMErrfs}/ush/set_predef_grid_params.sh' (see Tables 2.6 and 2.7), and automatically replaced in generating a regional workflow.

Notes) The parameters of 'cen_lon', 'cen_lat', 'lon1', 'lat1', 'lon2', and 'lat2' can be found by 'fv3grid' as shown in Section 11.1.2.

Sub-hourly output frequency:

To make the output frequency less than one hour, you need to modify the last four values in Table 2.72. For example, if you want output files every 30 minute in a run of 'dt_atmos'=300:

Parameter name	New value
nfhout	1
nfhmax_hf	-1

nfhout_hf	1
nsout	6

Table 2.73: Parameters for sub-hourly output

The format of the output files is '{filename_base}fHHH:MM:SS.nc'. Make sure that you should modify the line under 'itag' for post-processing in '{HOMErrfs}/scripts/ exregional_run_post.sh':

```
${post_yyyy}-${post_mm}-${post_dd}_${post_hh}:00:00
```

2.6.10 nems.configure

Various NEMS (NOAA Environmental Modeling System) components and their run-sequence are specified in this file. Since *FV3* runs stand-alone for the regional workflow, this file is shown as follows:

```
EARTH_component_list: ATM
ATM_model:            fv3
runSeq::
    ATM
::
```

2.6.11 regional_grid.nml

Namelist for the ESG grid generator (See Table 5.1):

Parameter	Description	Unit
plon	Longitude of the center of the regional domain	Degree
plat	Latitude of the center of the regional domain	Degree
delx	Grid spacing in the longitudinal direction on the super-grid	Degree
dely	Grid spacing in the latitudinal direction on the super-grid	Degree
lx	Negative of the number of cells in the longitudinal direction on the super-grid from the bottom-left corner to the center of the grid	Cell
ly	Same as 'lx' but in the latitudinal direction	Cell

Table 2.74: Namelist for the ESG grid generator

2.7 HPC Environment Configuration

Parameter name in configuration	Parameter name in scripts	Description
layout_1	LAYOUT_X	The number of tasks along the longitudinal direction (x)
layout_2	LAYOUT_Y	The number of tasks along the latitudinal direction (y)
blocksize	BLOCKSIZE	
-	nx_per_task	= NX/LAYOUT_X = (npx-1)/LAYOUT_X
-	ny_per_task	= NY/LAYOUT_Y = (npy-1)/LAYOUT_Y
PE_MEMBER01	PE_MEMBER01	Total number of MPI tasks for the forecast
atmos_nthreads	-	The number of processes per node for atmosphere
ncores_per_node	NCORES_PER_ NODE	Total number of processes per node in use
write_groups	WRTCMP_write_ groups	Total number of writing groups
write_tasks_ per_group	WRTCMP_write_ tasks_per_group	The number of tasks in each writing group
ppn	PPN_[task]	The number of processes per node
OMP_NUM_ THREADS	-	The number of OpenMP threads (='--cpus-per-task')
		• (1) × (2) + (9) × (10) = (6) • (11) × (12) = (8) • (7) = (12)

Table 2.75: HPC configurations for *FV3*

Chapter 3

Workflow: Forecast

3.1 Structure of Regional Workflow

The user-defined experimental directory '{EXPTDIR}' is created by running 'generate_FV3LAM_wflow.sh' as follows:

```
cd {HOME}/../expt_dirs/{EXPT_SUBDIR}   (={EXPTDIR})
```

where '{EXPT_SUBDIR}' is specified in the 'config.[option].sh' file, and the '{EXPTDIR}' directory contains:

File/directory name	Description
config.sh	User-specified configuration file (Table 2.43)
data_table	Cycle-independent input file (empty)
field_table	Scalar fields in the forecast model (Section 2.6.5)
FV3LAM_wflow.xml	Rocoto XML file to run the workflow
input.nml	Namelist file of *FV3* (Section 2.6.8)
launch_FV3LAM_wflow.sh	Symlink to the shell script of '{HOMErrfs}/ush/launch_FV3LAM_ wflow.sh' that can be used to (re)launch the Rocoto workflow. Each time this script is called, it appends to a log file named 'log.launch_FV3LAM_wflow'.
log.generate_FV3LAM_wflow	Log file of 'generate_FV3LAM_wflow.sh'
nems.configure	NEMS configuration file (Section 2.6.10)
suite_{CCPP}.xml	CCPP suit definition file in case of 'CCPP_PHYS'="TRUE"

var_defns.sh	Shell script defining the experiment parameters. This file is sourced by various other scripts in order to make all the experiment parameters available to these scripts.
YYYYMMDDHH	Cycle directory (empty)
fix_am	Directory containing the global fixed (time-independent) data files. The experiment generation script copies these files from a machine-dependent system directory.
fix_lam	Initially empty, but will contain the regional fixed (time-independent) data files that describe the regional grid, orography, and various surface climatology fields as well as symlinks to pre-generated files.

Table 3.1: Initial status of the user-defined experimental directory

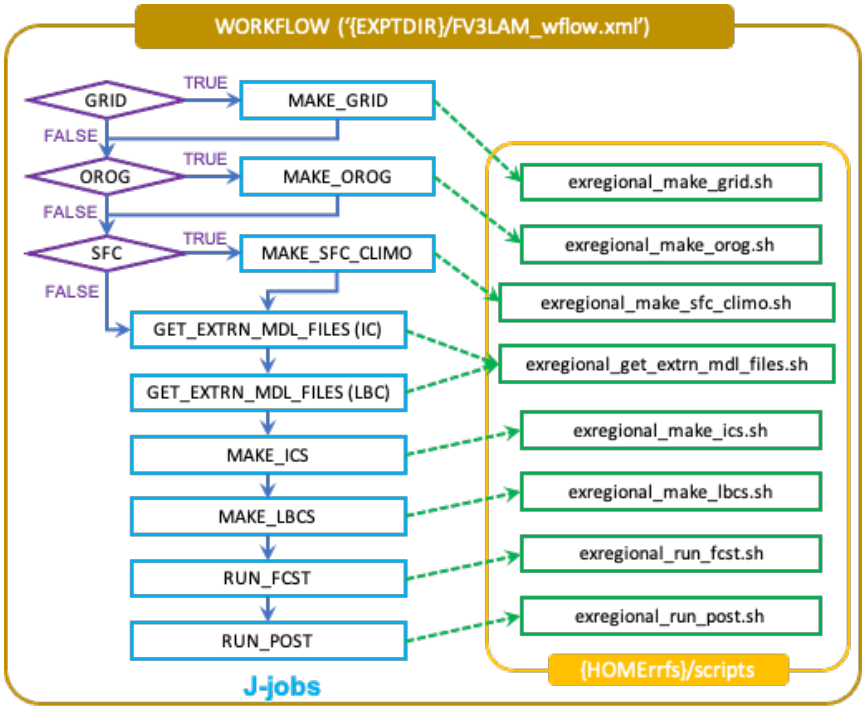

Figure 3.1: Structure of the regional workflow

The flowchart of the regional workflow described in 'FV3LAM_wflow.xml' is shown in Figure 3.1. Each task calls its own J-job script in '{HOMErrfs}/jobs/', and the J-job script calls its main script 'exregional_[task name].sh' located in '{HOMErrfs}/scripts/'. The first three pre-processing tasks;

'MAKE_GRID', 'MAKE_OROG', and 'MAKE_SFC_CLIMO' are optional. If the pre-generated grid, orography, and surface climatology fix files exist, these three task can be skipped by setting the corresponding flags to "FALSE" in the '{HOMErrfs}/ush/config.sh' script before running the 'generate_FV3LAM_wflow.sh' script. As shown in the figure, the 'FV3LAM_wflow.xml' file runs the specific J-job scripts in the prescribed order ('{HOMErrfs}/jobs/JREGIONAL_[task name]') when the 'launch_FV3LAM_wflow.sh' is submitted.

Workflow task	Description	Detail
make_grid	Pre-processing task to generate regional grid files	Section 5.1
make_orog	Pre-processing task to generate regional orography files	Section 5.2
make_sfc_climo	Pre-processing task to generate surface climatology files	Section 5.3
get_extrn_ics	Cycle-specific task to obtain external data for the initial conditions	Section 7.1
get_extrn_lbcs	Cycle-specific task to obtain external data for the lateral boundary conditions	Section 7.1
make_ics	Generate initial conditions from the external data	Section 7.4
make_lbcs	Generate lateral boundary conditions from the external data	Section 7.5
run_fcst	Run the forecast model (UFS weather model)	Section 3.1
run_post	Run the post-processing tool *UPP*	Section 8.1

Table 3.2: Workflow tasks

Table 3.2 describes the tasks in the regional workflow. You can find the detail of the workflow tasks in the specific sections of this document except for the 'run_fcst' task. As shown in Figure 3.2, the 'run_fcst' task plays a role in collecting input files and running the FV3-LAM model as follows:

1. Rename the input fix files to the exact names required by the FV3-LAM model.
2. Link the input files such as 'data_table', 'field_table', 'nems.configure', and 'input.nml' to the experiment directory.
3. Run the FV3-LAM model.

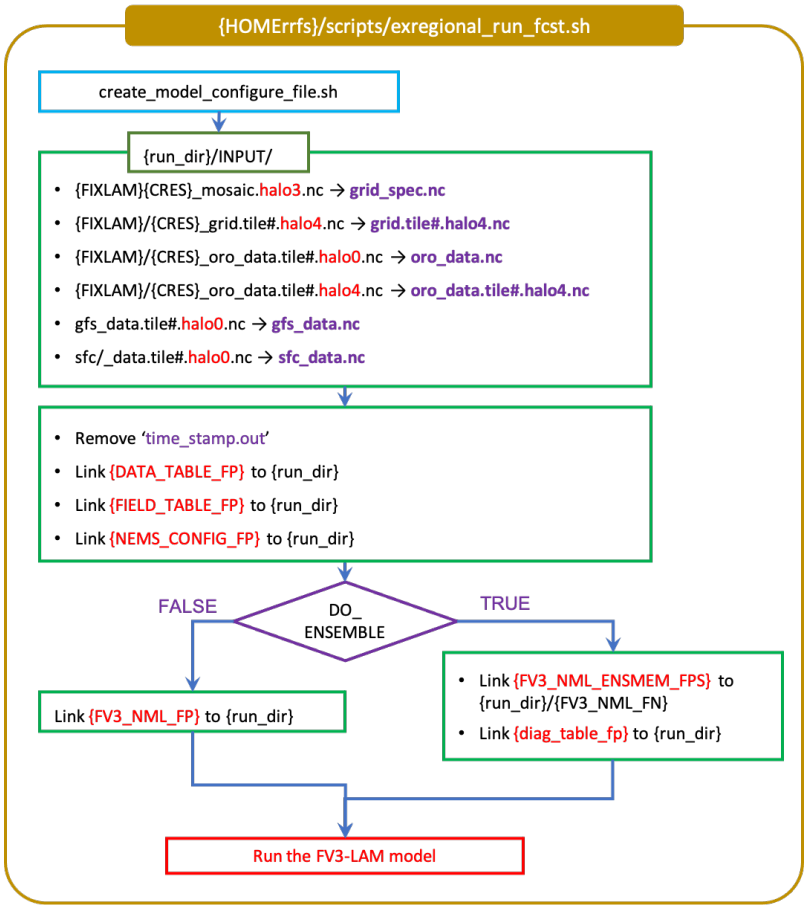

Figure 3.2: Flowchart of 'exregional_run_fcst.sh'

3.2 Modules for Workflow Tasks

The modules that are required for each task of the regional workflow are specified in:

```
vim {HOMErrfs}/modulefiles/tasks/[machine]/[task].local
```

Each file should have a header as follows:

```
#%Module
```

Note) If any specific modules are not necessary for a task, the module file '[task].local' for the task will not have to exist in the directory.

3.3 Launch of Workflow: 'community' Mode

There are two ways to launch the workflow: (1) using the 'launch_FV3LAM_wflow.sh' script, and (2) manually calling the *rocotorun* command as follows:

Note) If the *Python* environment for workflow is not loaded, load it as described in Section 2.5.1.

3.3.1 Launch with the 'launch_FV3LAM_wflow.sh' Script

To launch the 'launch_FV3LAM_wflow.sh' script, simply call it without any arguments.

```
cd {EXPTDIR}
./launch_FV3LAM_wflow.sh
```

This script will create (or append to if it already exists) a log file named 'log.launch_FV3LAM_wflow' in {EXPTDIR}. You can check the contents towards the end of this log file (e.g. the last 30 lines) using the command:

```
tail -n 30 log.launch_FV3LAM_wflow
```

After only one call to 'launch_FV3LAM_wflow.sh', the *tail* command will output something like the following:

```
    CYCLE              TASK                   JOBID          STATE       EXIT STATUS  TRIES  DURATION
==========================================================================================================
202006170000        make_grid  druby://hfe01:33728  SUBMITTING                  -      0       0.0
202006170000        make_orog                    -           -                  -      -         -
202006170000    make_sfc_climo                   -           -                  -      -         -
202006170000     get_extrn_ics  druby://hfe01:33728  SUBMITTING                  -      0       0.0
202006170000     get_extrn_lbcs  druby://hfe01:33728  SUBMITTING                  -      0       0.0
202006170000         make_ics                    -           -                  -      -         -
202006170000        make_lbcs                    -           -                  -      -         -
202006170000         run_fcst                    -           -                  -      -         -
202006170000      run_post_00                    -           -                  -      -         -
202006170000      run_post_01                    -           -                  -      -         -
202006170000      run_post_02                    -           -                  -      -         -
202006170000      run_post_03                    -           -                  -      -         -
202006170000      run_post_04                    -           -                  -      -         -
202006170000      run_post_05                    -           -                  -      -         -
202006170000      run_post_06                    -           -                  -      -         -

Summary of workflow status:
~~~~~~~~~~~~~~~~~~~~~~~~~~

 0 out of 1 cycles completed.
 Workflow status:  IN PROGRESS
```

This shows that the 'make_grid' task (which is cycle-independent and thus run at most once per experiment regardless of the number of cycles specified in 'config.sh') as well as the 'get_extrn_ics'

and 'get_extrn_lbcs' tasks (which are cycle-dependent and thus must be run for each cycle; here, they are being run for the first cycle starting on 202006170000) are being submitted to the batch system. The overall status of the workflow is "IN PROGRESS".

Check if any tasks have error messages:

```
cd {EXPTDIR}/log/
grep -n -i error *.log
```

- **Note)** You can check the status of the run directly as follows:

  ```
  watch -n 5 squeue -u [UserID]
  ```

  This command prints out the status of queues by [UserID] every 5 seconds.

In order to launch more tasks in the workflow, we must now call the launch script again. We can combine the call to this script with the *tail* command to see output from its log file into one line as follows:

```
cd {EXPTDIR}
./launch_FV3LAM_wflow.sh ; tail -n 30 log.launch_FV3LAM_wflow
```

The output from this is something like:

```
     CYCLE              TASK                   JOBID          STATE       EXIT STATUS  TRIES  DURATION
==========================================================================================================
202006170000         make_grid                 8794245      SUCCEEDED                 -      1       4.0
202006170000         make_orog  druby://hfe01:33728      SUBMITTING                 -      0       0.0
202006170000    make_sfc_climo                       -              -                 -      -         -
202006170000      get_extrn_ics                 8794246      SUCCEEDED                 -      1      47.0
202006170000     get_extrn_lbcs                 8794246      SUCCEEDED                 -      1      61.0
202006170000         make_ics                       -              -                 -      -         -
202006170000        make_lbcs                       -              -                 -      -         -
202006170000         run_fcst                       -              -                 -      -         -
202006170000      run_post_00                       -              -                 -      -         -
202006170000      run_post_01                       -              -                 -      -         -
202006170000      run_post_02                       -              -                 -      -         -
202006170000      run_post_03                       -              -                 -      -         -
202006170000      run_post_04                       -              -                 -      -         -
202006170000      run_post_05                       -              -                 -      -         -
202006170000      run_post_06                       -              -                 -      -         -

Summary of workflow status:
~~~~~~~~~~~~~~~~~~~~~~~~~~

 0 out of 1 cycles completed.
 Workflow status:  IN PROGRESS
```

This shows that the ‘make_grid’ task as well as ‘get_extrn_ics’ and ‘get_extrn_lbcs’ tasks for the 202006170000 cycle have completed successfully and that the ‘make_orog’ task (which is cycle-independent and thus run at most once per experiment regardless of the number of cycles specified in ‘config.sh’) is being submitted to the batch system.

Once the pre-processing tasks complete without any errors, you should find the files representing completion of the tasks in ‘{EXPTDIR}/log/’ as follows:

Task	File of completion
make_grid	make_grid_task_complete.txt
make_orog	make_orog_task_complete.txt
make_sfc_climo	make_sfc_climo_task_complete.txt

Table 3.3: File names representing completion of the pre-processing tasks

Note) You can call the launch script as often as you like, i.e. you do not have to wait until the currently running tasks are finished. If they are still running, the script will simply append to the log file a new table (which is generated using *Rocoto*’s *rocotostat* command) listing the state of any running tasks as “RUNNING”.

Once the ‘make_grid’, ‘make_orog’, and ‘make_sfc_climo’ tasks and the ‘get_extrn_ics’ and ‘get_extrn_lbcs’ tasks for the 202006170000 cycle have completed successfully, the new files and sub-directories are as follows:

No.	Directory/file name	Description
1	2020061700	This is the directory created when the first cycle-specific workflow tasks are run, which are ‘get_extrn_ics’ and ‘get_extrn_lbcs’ (they are launched simultaneously for each cycle in the experiment). We refer to this as a “cycle directory”. Cycle directories are created to contain cycle-specific files for each cycle that the experiment runs. If ‘DATE_FIRST_CYCL’ and ‘DATE_LAST_CYCL’ were different, and/or ‘CYCL_HRS’ contained more than one element in the ‘config.sh’ file, then more than one cycle directory would be created under the experiment directory.
2	grid	This is a directory generated by the ‘make_grid’ task. It contains the grid files for the experiment.

3	log	This is a directory in which the log files generated by the overall workflow as well as its various tasks are stored. Look in these files to trace why a task may have failed.
4	orog	This is a directory generated by the 'make_orog' task. It contains the orography files for the experiment.
5	sfc_climo	This is a directory generated by the 'make_sfc_climo' task. It contains the surface climatology files for the experiment.
6	FV3LAM_wflow.db FV3LAM_wlfow_lock.db	These are database files that are generated with *rocoto* is called (by the launch script) to launch the workflow. There is usually no need for the user to modify these files. If you relaunch the workflow from scratch, **delete** these files and then call the launch script (multiple times, as usual).
7	log.launch_FV3LAM_wflow	This is the log file to which the launch script ('launch_FV3LAM_wflow.sh') appends its output each time it is called. Take a look at the last 30–50 lines of this file to check the status of the workflow.

Table 3.4: New directories and files created by *Rocoto*

Notes) To avoid having to manually call the launch script, the experiment generation script allows users to automatically place the call to this script in the user's crontab and have *cron* call it with a specified frequency.

If everything goes smoothly, you will eventually get the following workflow status table as follows:

CYCLE	TASK	JOBID	STATE	EXIT STATUS	TRIES	DURATION
202006170000	make_grid	8854765	SUCCEEDED	0	1	6.0
202006170000	make_orog	8854809	SUCCEEDED	0	1	27.0
202006170000	make_sfc_climo	8854849	SUCCEEDED	0	1	36.0
202006170000	get_extrn_ics	8854763	SUCCEEDED	0	1	54.0
202006170000	get_extrn_lbcs	8854764	SUCCEEDED	0	1	61.0
202006170000	make_ics	8854914	SUCCEEDED	0	1	119.0
202006170000	make_lbcs	8854913	SUCCEEDED	0	1	98.0
202006170000	run_fcst	8854992	SUCCEEDED	0	1	655.0
202006170000	run_post_00	8855459	SUCCEEDED	0	1	6.0
202006170000	run_post_01	8855460	SUCCEEDED	0	1	6.0
202006170000	run_post_02	8855461	SUCCEEDED	0	1	6.0
202006170000	run_post_03	8855462	SUCCEEDED	0	1	6.0
202006170000	run_post_04	8855463	SUCCEEDED	0	1	6.0
202006170000	run_post_05	8855464	SUCCEEDED	0	1	6.0
202006170000	run_post_06	8855465	SUCCEEDED	0	1	6.0

If all the tasks are completed successfully, the workflow status in this log file will be set to "SUCCESS". If one or more tasks are not completed successfully for some reason, the workflow

status will be set to "FAILURE".

```
The end-to-end run of the workflow for the forecast experiment specified
by expt_name has completed with the following workflow status (wflow_-
status):
  expt_name = "test_community_hrrr25"
  wflow_status = "SUCCESS"
```

3.3.2 Launch Manually by Calling the *rocotorun* Command

Load the *Rocoto* module if it is not loaded:

- Hera:

```
module load rocoto
```

- WCOSS_DELL_P3:

```
module load lsf/10.1
module use /gpfs/dell3/usrx/local/dev/emc_rocoto/modulefiles/
module load ruby/2.5.1 rocoto/1.2.4
```

- WCOSS_CRAY:

```
module load xt-lsfhpc/9.1.3
module use -a /usrx/local/emc_rocoto/modulefiles
module load rocoto/1.2.4
```

Launch the workflow as follows:

```
cd {EXPTDIR}
rocotorun -w FV3LAM_wflow.xml -d FV3LAM_wflow.db -v 10
```

where '{EXPTDIR}' is the user-defined experimental directory.

To see the status of the workflow, issue a *rocotostat* command as follows:

```
cd {EXPTDIR}
rocotostat -w FV3LAM_wflow.xml -d FV3LAM_wflow.db -v 10
```

Wait a few seconds and issue a second set of *rocotorun* and *rocotostat* commands as follows:

```
rocotorun -w FV3LAM_wflow.xml -d FV3LAM_wflow.db -v 10
rocotostat -w FV3LAM_wflow.xml -d FV3LAM_wflow.db -v 10
```

Note) Issuing a 2nd (3rd, etc) *rocotostat* command without an intervening *rocotorun* command will not result in an updated workflow status table. It is the *rocotorun* command that updates the workflow database file (in this case 'FV3LAM_wflow.db', located in {EXPTDIR}). The *rocotostat* command just reads the database file and prints the table to the screen. Thus, to see an updated table, you must always first issue a *rocotorun* command and then follow it up with a *rocotostat* command.

3.3.3 Turning On/off the Cycle-independent Workflow Tasks

The first three tasks of the workflow ('make_grid', 'make_orog', and 'make_sfc_climo') generate fixed files that are not cycle dependent. Thus, they do not need to be (and are not) run for each cycle. The following three experiment parameters determine whether or not each of these tasks is run at all in the workflow:

1. RUN_TASK_MAKE_GRID
2. RUN_TASK_MAKE_OROG
3. RUN_TASK_MAKE_SFC_CLIMO

By default, these are set to "TRUE" in '{HOMErrfs}/ush/config_defaults.sh' as follows:

```
RUN_TASK_MAKE_GRID="TRUE"
RUN_TASK_MAKE_OROG="TRUE"
RUN_TASK_MAKE_SFC_CLIMO="TRUE"
```

In this case, these tasks will run **only once** at the beginning of the workflow before any cycle-dependent tasks are launched. They will generate grid, filtered orography, and surface climatology files that are needed by the subsequent cycle-dependent tasks.

If you're running a set of cycles for which you've already generated these files (e.g. from a previous end-to-end run of the workflow), then you can skip these three tasks by setting the above flags to "FALSE" in your 'config.sh' file, i.e.

```
RUN_TASK_MAKE_GRID="FALSE"
RUN_TASK_MAKE_OROG="FALSE"
```

```
RUN_TASK_MAKE_SFC_CLIMO="FALSE"
```

In this case, you also need to specify the directories in which the workflow should look for your pre-generated grid, orography, and surface climatology files. These directories are specified in the following three experiment parameters:

1. GRID_DIR
2. OROG_DIR
3. SFC_CLIMO_DIR

Thus, to skip the 'make_grid', 'make_orog', and 'make_sfc_climo' tasks, you would set the above three 'RUN_TASK_MAKE_[task]' flags to "FALSE" and specify the paths to your pre-generated files in your '{HOMErrfs}/ush/config.sh' file as follows:

```
RUN_TASK_MAKE_GRID="FALSE"
GRID_DIR="/path/to/pre-generated/grid/files"
RUN_TASK_MAKE_OROG="FALSE"
OROG_DIR="/path/to/pre-generated/orog/files"
RUN_TASK_MAKE_SFC_CLIMO="FALSE"
SFC_CLIMO_DIR="/path/to/pre-generated/surface/climo/files"
```

Note) If 'RUN_TASK_MAKE_GRID' is set to "TRUE", the value of 'GRID_DIR' which is set in either 'config_defaults.sh' or in 'config.sh' is ignored. The same is true for 'RUN_TASK_MAKE_OROG' and 'OROG_DIR' and for 'RUN_TASK_MAKE_SFC_CLIMO' and 'SFC_CLIMO_DIR'.

You can choose to skip any one or any two of these three tasks instead of all three. For example, you can skip the 'make_grid' and 'make_orog' tasks but run the 'make_sfc_climo' task by:

1. Set 'RUN_TASK_MAKE_GRID'="FALSE" and 'RUN_TASK_MAKE_OROG'="FALSE" in your 'config.sh'.
2. Set 'GRID_DIR' and 'OROG_DIR' in your 'config.sh' to the locations in which grid and orography files for the grid you want to run on can be found.
3. Set 'RUN_TASK_MAKE_SFC_CLIMO'="TRUE" in your 'config.sh' (and not setting 'SFC_CLIMO_DIR' in 'config.sh' since its value will not be used).

3.3.4 How to Restart a 'DEAD' Task

When something goes wrong, a workflow task would end up in the 'DEAD' state as follows:

```
rocotostat -w FV3LAM_wflow.xml -d FV3LAM_wflow.db -v 10
       CYCLE             TASK         JOBID     STATE     EXIT STATUS  TRIES DURATION
==========================================================================================
201905200000        make_grid       9443237   QUEUED               -      0        0.0
201905200000        make_orog             -        -               -      -          -
201905200000   make_sfc_climo             -        -               -      -          -
201905200000    get_extrn_ics       9443293     DEAD             256      3        5.0
```

Once the issue has been resolved, the failed task can re-run using the 'rocotorewind' command (See A.1):

```
rocotorewind -w FV3LAM_wflow.xml -d FV3LAM_wflow.db -v 10 -c 201905200000 -t
    get_extrn_ics
```

where

Flag	Description
-w	Path to workflow XML file ('FV3LAM_wflow.xml')
-d	Path to workflow database file ('FV3LAM_wflow.db')
-v	Run Rocoto in verbose mode [LEVEL]
-c	List of cycles (YYYYMMDDHHMM) 'c1,c2,c3' or 'c1:c2' or all
-t	List of tasks 'a,b,c'

Table 3.5: Flags for the 'rocotorewind' command

3.3.5 How to Re-run Forecast and Post Tasks Only

There are two approaches to run the forecast and post tasks again after modification of the source code: (1) Using *rocotorewind*, and (2) Modifying 'FV3LAM_wflow.xml'.

1. Use the *rocotorewind* command to undo the tasks as described in Section 3.3.4.

   (a) Load the *rocoto* module as described in Section 3.3.2.

   (b) Remove the result files generated by the previous forecast and post tasks:

```
cd {EXPTDIR}/{CYCLE}
rm -f atmos_4xdaily.nc atmos_static.nc
rm -f dynf* logf* phyf*
rm -rf postprd
```

(c) Run *rocotorewind* for the forecast and post tasks:

```
rocotorewind -w FV3LAM_wflow.xml -d FV3LAM_wflow.db -v 10 -c
    201906150000 -t 'run_fcst,run_post_f001,run_post_f002,run_post_f003,
    run_post_f004,run_post_f005,run_post_f006'
```

(d) Run the forecast and post tasks again:

```
./launch_FV3LAM_wflow.sh
```

2. Modify the workflow XML file 'FV3LAM_wflow.xml'.

(a) Remove the result files generated by the previous forecast and post tasks:

```
cd {EXPTDIR}/{CYCLE}
rm -f atmos_4xdaily.nc atmos_static.nc
rm -f dynf* logf* phyf*
rm -rf postprd
```

(b) Remove the workflow data files:

```
cd ..
rm -f FV3LAM_*.db
```

(c) Modify the workflow XML file 'FV3LAM_wflow.xml':

- Remove other tasks listed below (between lines 101–319 approx.):
  - MAKE_GRID_TN ("make_grid")
  - MAKE_OROG_TN ("make_orog")
  - MAKE_SFC_CLIMO_TN ("make_sfc_climo")
  - GET_EXTRN_ICS_TN ("get_extrn_ics")
  - GET_EXTRN_LBCS_TN ("get_extrn_lbcs")
  - MAKE_ICS_TN ("make_ics")

  - MAKE_LBCS_TN (“make_lbcs”)

- Remove the dependency of the ‘&RUN_FCST_TN’ task:

```
<dependency>
 <and>
  <taskdep task="&MAKE_ICS_TN;"/>
  <taskdep task="&MAKE_LBCS_TN;"/>
 </and>
</dependency>
```

Finally, the task part of ‘FV3LAM_wflow.xml’ should be as follows:

```
<workflow realtime="F" scheduler="&SCHED;" cyclethrottle="20">

 <cycledef group="at_start">00 00 15 06 2019 *</cycledef>
 <cycledef group="forecast">201906150000 201906150000 24:00:00</cycledef>

 <log>
  <cyclestr>&LOGDIR;/FV3LAM_wflow.log</cyclestr>
 </log>

<!--
************************************************************************
************************************************************************
-->
 <task name="&RUN_FCST_TN;" maxtries="1">

  &RSRV_FCST;
  <command>&LOAD_MODULES_RUN_TASK_FP; "&RUN_FCST_TN;" "&JOBSDIR;/JREGIONAL_RUN_FCST"</command>

  <cores>12</cores>
  <native>--cpus-per-task 4 --exclusive</native>

  <walltime>01:00:00</walltime>
  <jobname>&RUN_FCST_TN;</jobname>
  <join><cyclestr>&LOGDIR;/&RUN_FCST_TN;_@Y@m@d@H.log</cyclestr></join>

  <envar><name>GLOBAL_VAR_DEFNS_FP</name><value>&GLOBAL_VAR_DEFNS_FP;</value></envar>
  <envar><name>PDY</name><value><cyclestr>@Y@m@d</cyclestr></value></envar>
  <envar><name>CDATE</name><value><cyclestr>@Y@m@d@H</cyclestr></value></envar>
  <envar><name>CYCLE_DIR</name><value><cyclestr>&CYCLE_BASEDIR;/@Y@m@d@H</cyclestr></value></envar>
  <envar><name>SLASH_ENSMEM_SUBDIR</name><value><cyclestr></cyclestr></value></envar>
  <envar><name>ENSMEM_INDX</name><value><cyclestr>##</cyclestr></value></envar>

 </task>
<!--
************************************************************************
************************************************************************
-->
 <metatask name="&RUN_POST_TN;">

  <var name="fhr">  000 001 002 003 004 005 006 007 008 009 010 011 012 013 014 015 016 017 018 019 020 021 022 023 024 025 026 027 028 029 030
       031 032 033 034 035 036 037 038 039 040 041 042 043 044 045 046 047 048 </var>

  <task name="&RUN_POST_TN;_f#fhr#" maxtries="1">

   &RSRV_DEFAULT;
   <command>&LOAD_MODULES_RUN_TASK_FP; "&RUN_POST_TN;" "&JOBSDIR;/JREGIONAL_RUN_POST"</command>
   <nodes>2:ppn=24</nodes>
   <walltime>00:15:00</walltime>
   <nodesize>&NCORES_PER_NODE;</nodesize>
   <jobname>&RUN_POST_TN;_f#fhr#</jobname>
   <join><cyclestr>&LOGDIR;/&RUN_POST_TN;_f#fhr#_@Y@m@d@H.log</cyclestr></join>
```

```
  <envar><name>GLOBAL_VAR_DEFNS_FP</name><value>&GLOBAL_VAR_DEFNS_FP;</value></envar>
  <envar><name>PDY</name><value><cyclestr>@Y@m@d</cyclestr></value></envar>
  <envar><name>CDATE</name><value><cyclestr>@Y@m@d@H</cyclestr></value></envar>
  <envar><name>CYCLE_DIR</name><value><cyclestr>&CYCLE_BASEDIR;/@Y@m@d@H</cyclestr></value></envar>
  <envar><name>SLASH_ENSMEM_SUBDIR</name><value><cyclestr></cyclestr></value></envar>
  <envar><name>cyc</name><value><cyclestr>@H</cyclestr></value></envar>
  <envar><name>fhr</name><value>#fhr#</value></envar>

  <dependency>
   <or>
    <taskdep task="&RUN_FCST_TN;"/>
    <and>
     <datadep age="05:00"><cyclestr>&CYCLE_BASEDIR;/@Y@m@d@H/dynf#fhr#.nc</cyclestr></datadep>
     <datadep age="05:00"><cyclestr>&CYCLE_BASEDIR;/@Y@m@d@H/phyf#fhr#.nc</cyclestr></datadep>
    </and>
   </or>
  </dependency>

 </task>

</metatask>
```

(d) Run the forecast and post tasks again:

```
./launch_FV3LAM_wflow.sh
```

3.4 Launch of Workflow: 'nco' Mode

3.4.1 Mosaic Halo Files

Different from the previous EMC's regional workflow, the unified regional workflow in the short range weather application requires 'mosaic_halo3.nc' and 'mosaic_halo4.nc' files as well as 'mosaic_halo6.nc'. Previously, there was just one mosaic file 'C{res}_mosaic.nc' that points to the pre-shave **halo6** file. A mosaic file is used by both the preprocessing codes and by *FV3*. Since halo6 has more layers than halo3, things will not go out of bounds but it might cause some potential issues. Therefore, it is important for each grid file to have its own mosaic file corresponding to the number of halos.

1. Load the modules used for build the executable 'make_solo_mosaic':

   - On Hera:

     ```
     source {HOMErrfs}/../env/build_hera_intel.env
     ```

   - On WCOSS_dell_p3:

```
source {HOMErrfs}/../env/build_wcoss_dell_p3_intel.env
```

- On WCOSS_cray:

```
source {HOMErrfs}/../env/build_wcoss_cray_intel.env
```

- On Orion:

```
source {HOMErrfs}/../env/build_orion_intel.env
```

2. Run the executable 'make_solo_mosaic' for each mosaic file:

```
cd {FIXLAM_NCO_BASEDIR}/{PREDEF_GRID_NAME}/
{BIN}/make_solo_mosaic --num_tile 1 --dir "{FIXLAM_NCO_BASEDIR}/{PREDEF_␣
    GRID_NAME}" --tile_file "C{res}_grid.tile7.halo6.nc" --mosaic "C{res}
    _mosaic.halo6"
{BIN}/make_solo_mosaic --num_tile 1 --dir "{FIXLAM_NCO_BASEDIR}/{PREDEF_␣
    GRID_NAME}" --tile_file "C{res}_grid.tile7.halo3.nc" --mosaic "C{res}
    _mosaic.halo3"
{BIN}/make_solo_mosaic --num_tile 1 --dir "{FIXLAM_NCO_BASEDIR}/{PREDEF_␣
    GRID_NAME}" --tile_file "C{res}_grid.tile7.halo4.nc" --mosaic "C{res}
    _mosaic.halo4"
```

where '{BIN}' is the directory where the executable 'make_solo_mosaic' is located (typically ='{HOME}/bin/').

3.4.2 Launch with the 'launch_FV3LAM_wflow.sh' Script

Launch the 'launch_FV3LAM_wflow.sh' script:

```
cd {EXPTDIR}
./launch_FV3LAM_wflow.sh
```

This script will create (or append to if it already exists) a log file named 'log.launch_FV3LAM_wflow' in {EXPTDIR}. You can check the contents towards the end of this log file (e.g. the last 30 lines) using the command:

```
tail -n 30 log.launch_FV3LAM_wflow
```

After only one call to 'launch_FV3LAM_wflow.sh', the *tail* command will output something like the following:

```
     CYCLE              TASK                   JOBID          STATE       EXIT STATUS  TRIES  DURATION
==========================================================================================================
202006180000      get_extrn_ics  druby://hfe01:33424     SUBMITTING                 -      0       0.0
202006180000    get_extrn_lbcs  druby://hfe01:33424     SUBMITTING                 -      0       0.0
202006180000       make_ics                     -              -                   -      -         -
202006180000       make_lbcs                    -              -                   -      -         -
202006180000        run_fcst                    -              -                   -      -         -
202006180000    run_post_00                     -              -                   -      -         -
202006180000    run_post_01                     -              -                   -      -         -
202006180000    run_post_02                     -              -                   -      -         -
202006180000    run_post_03                     -              -                   -      -         -
202006180000    run_post_04                     -              -                   -      -         -
202006180000    run_post_05                     -              -                   -      -         -
202006180000    run_post_06                     -              -                   -      -         -

Summary of workflow status:
~~~~~~~~~~~~~~~~~~~~~~~~~~

 0 out of 1 cycles completed.
 Workflow status:  IN PROGRESS
```

In order to launch more tasks in the workflow, we must now call the launch script again. We can combine the call to this script with the *tail* command to see output from its log file into one line as follows:

```
cd {EXPTDIR}
./launch_FV3LAM_wflow ; tail -n 30 log.launch_FV3LAM_wflow
```

The output from this is something like:

```
     CYCLE              TASK                   JOBID          STATE       EXIT STATUS  TRIES  DURATION
==========================================================================================================
201909011800      get_extrn_ics                8887541      SUCCEEDED                 0      1       4.0
201909011800    get_extrn_lbcs                 8887542      SUCCEEDED                 0      1       4.0
201909011800       make_ics    druby://hfe11:33475     SUBMITTING                 -      0       0.0
201909011800       make_lbcs   druby://hfe11:33475     SUBMITTING                 -      0       0.0
201909011800        run_fcst                    -              -                   -      -         -
201909011800    run_post_00                     -              -                   -      -         -
201909011800    run_post_01                     -              -                   -      -         -
201909011800    run_post_02                     -              -                   -      -         -
201909011800    run_post_03                     -              -                   -      -         -
201909011800    run_post_04                     -              -                   -      -         -
201909011800    run_post_05                     -              -                   -      -         -
201909011800    run_post_06                     -              -                   -      -         -
```

If everything goes smoothly, you will eventually get the following workflow status table as follows:

```
     CYCLE              TASK                   JOBID          STATE       EXIT STATUS  TRIES  DURATION
==========================================================================================================
201909011800      get_extrn_ics                8887541      SUCCEEDED                 0      1       2.0
201909011800    get_extrn_lbcs                 8887542      SUCCEEDED                 0      1       2.0
201909011800       make_ics                    8887551      SUCCEEDED                 0      1     160.0
```

201909011800	make_lbcs	8887552	SUCCEEDED	0	1	148.0
201909011800	run_fcst	8887753	SUCCEEDED	0	1	365.0
201909011800	run_post_00	8888085	SUCCEEDED	0	1	7.0
201909011800	run_post_01	8888086	SUCCEEDED	0	1	7.0
201909011800	run_post_02	8888087	SUCCEEDED	0	1	8.0
201909011800	run_post_03	8888106	SUCCEEDED	0	1	6.0
201909011800	run_post_04	8888107	SUCCEEDED	0	1	7.0
201909011800	run_post_05	8888108	SUCCEEDED	0	1	8.0
201909011800	run_post_06	8888109	SUCCEEDED	0	1	6.0

If all the tasks completed successfully, the workflow status in this log file will be set to "SUCCESS". If for some reason one or more tasks did not complete successfully, the workflow status will be set to "FAILURE".

```
The end-to-end run of the workflow for the forecast experiment specified
by expt_name has completed with the following workflow status (wflow_-
status):
  expt_name = "test_nco_conus96"
  wflow_status = "SUCCESS"
```

3.5 Automatic Resubmission with Cron

3.5.1 Syntax of Crontab

Cron runs jobs at designated times and intervals periodically. Cron is controlled by a 'crontab' file. Each line in a crontab file represents a separate job, and it has five time components followed by a shell command to execute as follows:

```
A B C D E <command to execute>
```

where 'A' is minute (0–59), 'B' is hour (0–23), 'C' is day of the month (1–31), 'D' is month (1–12), and 'E' is day of the week (0–6 which mean Sunday–Saturday).

Notes)

- '*': Run for every time.
- '*/n' : Run for every n-th interval of time.
- Multiple specific time intervals can be specified with commas (',').
- '-': Range of values.

- Comments begin with a comment mark '#'.

For example, the below would run 'my_script.sh' every 10 minutes of every first, second, and third hour between the 10th and 15th of every month:

```
*/10 1,2,3 10-15 * * /scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/my_script.sh
```

3.5.2 Setting up Command Line on Hera / WCOSS Cray

There are two ways to set up a command line in the cron: 1) using a command line, and 2) putting a command line in a file.

1. Using a command line:

   Open a crontab editor (vim):

   ```
   crontab -e
   ```

   Put a command line:

   ```
   */10 1,2,3 * * * /scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/my_script.sh
   ```

   Check the crontab:

   ```
   crontab -l
   ```

   You can remove the crontab as follows:

   ```
   crontab -r
   ```

2. Putting a command in a file:

   Open a file:

   ```
   vim my_crontab.txt
   ```

   Put a command line:

   ```
   */10 1,2,3 * * * /scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/my_script.sh
   ```

   Put it in the cron:

   ```
   crontab my_crontab.txt
   ```

3.5.3 Setting up Command Line on WCOSS Dell

The 'crontab' command, described in Section 3.5.2, does not work on the WCOSS Dell. Instead, a directory named 'cron' exists in the home directory as '/u/Chan-Hoo.Jeon/cron'. In this directory, you can find a 'mycrontab' file where command lines are added like the second approach of Section 3.5.2. The difference is that you do not have to type 'crontab' here.

Open a file:

```
cd /u/Chan-Hoo.Jeon/cron
vim mycrontab
```

Put a command line:

```
*/10 1,2,3 * * * /gpfs/dell2/emc/modeling/noscrub/Chan-Hoo.Jeon/test/my_script.sh
```

Note) it usually takes about 5 minutes to put your script in the cron, which is different from other machines where it is immediately in.

3.5.4 Sample Command for Launch Script

Below is a sample command to launch the script described in Sections 3.3.1 and 3.4.2:

```
*/05 * * * * cd /scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/expt_dirs/
    grid_RRFS_CONUS_13km_ics_FV3GFS_lbcs_FV3GFS_suite_GFS_v15p2 && ./
    launch_FV3LAM_wflow.sh >> ./log.launch_FV3LAM_wflow 2>&1
```

3.6 Output

3.6.1 Type of Output Files

The output files that contain the physical variables are converted into other formats such as NetCDF and GRIB2 in the regional workflow as shown in Figure 3.3. The file names for the physical variables, 'fv3_history' and 'fv3_history2d', are specified in the input file 'diag_table'. When 'quilting' is set to '.true.' for the write components in the input file 'model_configure', the two files of 'fv3_history' and 'fv3_history2d' are converted into the two NetCDF files named 'dynfXXX.nc' and 'phyfXXX.nc'. The bases of the file names are specified in the input file 'model_configure'. These NetCDF files are converted into the two GRIB2 files named 'BGDAWP' and 'BGRD3D'.

Finally, the 'BGDAWP' and 'BGRD3D' files are linked to '[domain].tXXz.bgdawpXX.tmXX' and '[domain].bgrd3dXX.tmXX', respectively.

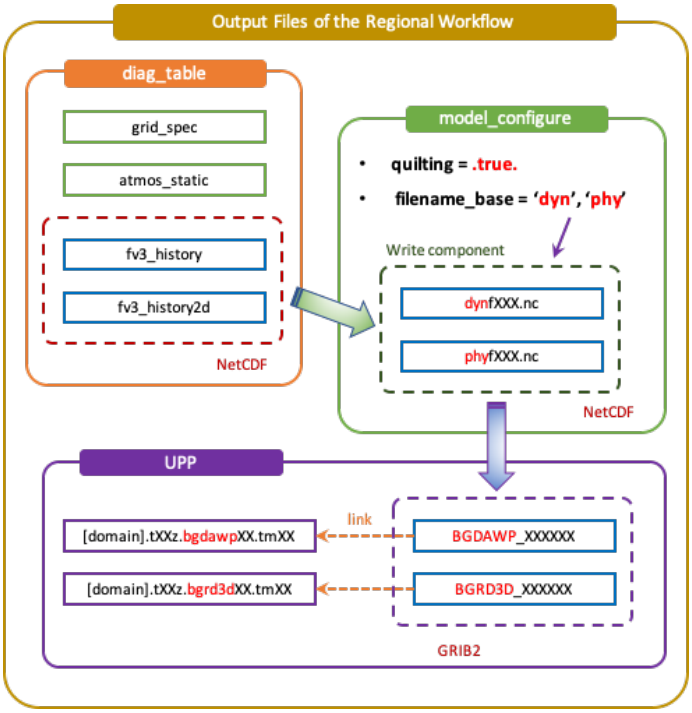

Figure 3.3: Output files of the regional workflow

The output files 'gird_spec.nc' and 'atmos_static.nc' contains the fields as follows:

1. 'grid_spec.nc':

Field name	Description
grid_lon	Longitude of the bottom-left corner (index; top-right on the map) of each grid cell
grid_lat	Latitude of the bottom-left corner (index; top-right on the map) of each grid cell
grid_lont	Longitude of the center of each grid cell
grid_latt	Latitude of the center of each grid cell
area	Cell area

Table 3.6: Fields in 'grid_spec.nc'

2. 'atmos_static.nc':

Field name	Description
pk	Pressure at the interface 'k' (hybrid coordinate)
bk	Vertical coordinate sigma value
hyam	Hybrid coefficient A at the vertical levels
hybm	Hybrid coefficient B at the vertical levels
zsurf	Surface height (orography)

Table 3.7: Fields in 'atmos_static.nc'

3.6.2 Output Files: 'community' Mode

1. NetCDF files:

```
cd {EXPTDIR}/YYYYMMDDHH
```

   - dynfXXX.nc (See Section 11.2.15 for plotting)
   - phyfXXX.nc (See Section 11.2.16 for plotting)

2. GRIB2 files:

```
cd {EXPTDIR}/YYYYMMDDHH/postprd
```

   - {domain}.t{cyc}z.bgdawp{fhour}.tmXX (or) BGDAWP_XXXXXXXXXfhourXX (See Section 11.2.18 for plotting)
   - {domain}.t{cyc}z.bgrd3d{fhour}.tmXX (or) BGRD3D_XXXXXXXXXfhourXX (See Section 11.2.17 for plotting)

3.6.3 Output Files: 'nco' Mode

1. NetCDF files:

```
cd {STMP}/stmp/tmpnwprd/{RUN}/YYYYMMDDHH
```

   - dynfXXX.nc (See Section 11.2.15 for plotting)

- phyfXXX.nc (See Section 11.2.16 for plotting)

2. GRIB2 files:

```
cd {PTMP}/com/rrfs/para/{RUN}.YYYYMMDD/HH
```

- {domain}.t{cyc}z.bgdawp{fhour}.tmXX (or) BGDAWP_XXXXXXXXXfhourXX (See Section 11.2.18 for plotting)
- {domain}.t{cyc}z.bgrd3d{fhour}.tmXX (or) BGRD3D_XXXXXXXXXfhourXX (See Section 11.2.17 for plotting)

Chapter 4

Workflow End-to-End (WE2E) Tests

4.1 Components of WE2E Tests

The Workflow End-to-End (WE2E) test is a tool to check if the workflow accomplishes the designated tasks using the pre-defined configuration files and input data. A set of WE2E tests can be found in '{HOMErrfs}/tests' as shown in Table 4.1. The list of the available tests is in 'baselines_list.txt', and their corresponding configuration files are located in the 'baseline_configs' directory. You can create your own list from 'baselines_list.txt' .

File/Directory name	Description
baseline_configs	Set of the test configuration files ('config.[test_name].sh').
baselines_list.txt	List of the available tests
run_experiments.sh	Script to run the tests

Table 4.1: Files and directory for the WE2E tests.

4.2 Tests Available on Specific Machines

Since it is computationally too expensive to run all the tests listed in 'baseline_list.xtx', this document only supports the specific tests shown in Table 4.2 that are available on Hera and WCOSS.

No.	Test name	Description
1	grid_RRFS_CONUS_25km_ics_FV3GFS_ lbcs_FV3GFS_suite_GFS_v15p2	Mode: community, domain: RRFS_CONUS, res: 25km, IC: FV3GFS, LBC: FV3GFS, CCPP: FV3_GFS_v15p2

2	grid_RRFS_CONUS_3km_ics_FV3GFS_ lbcs_FV3GFS_suite_GFS_v15p2	Mode: community, domain: RRFS_CONUS, res: 3km, IC: FV3GFS, LBC: FV3GFS, CCPP: FV3_GFS_v15p2
3	grid_RRFS_CONUS_3km_ics_HRRR_ lbcs_RAP_suite_GFS_v15p2	Mode: community, domain: RRFS_CONUS, res: 3km, IC: HRRR, LBC: RAP, CCPP: FV3_GFS_v15p2
4	grid_RRFS_SUBCONUS_3km_HRRR_ RAP_suite_GFS_v15p2	Mode: community, domain: RRFS_SUBCONUS, res: 3km, IC: HRRR, LBC: RAP, CCPP: FV3_GFS_v15p2
5	nco_grid_RRFS_CONUS_3km_ics_ FV3GFS_lbcs_FV3GFS_suite_GFS_v15p2	Mode: nco, domain: RRFS_CONUS, res: 3km, IC: FV3GFS, LBC: FV3GFS, CCPP: FV3_GFS_v15p2
6	pregen_grid_orog_sfc_climo	Mode: community, domain: RRFS_CONUS, res: 25km, IC: FV3GFS, LBC: FV3GFS, CCPP: FV3_GFS_v15p2

Table 4.2: Tests available on the specific machines.

4.3 Data for WE2E Tests

The input data for the WE2E tests are specified in the 'run_experiments.sh' script as follows:

1. Pre-generated files:

Machine	Path to the data ('pregen_basedir')
Hera	/scratch2/BMC/det/FV3LAM_pregen
WCOSS_dell_p3	/gpfs/dell2/emc/modeling/noscrub/UFS_SRW_App/FV3LAM_pregen
WCOSS_cray	/gpfs/hps3/emc/meso/noscrub/UFS_SRW_App/FV3LAM_pregen

Table 4.3: Path to the data for the WE2E tests: pre-generated files

- Grid files:

```
GRID_DIR="${pregen_basedir}/${PREDEF_GRID_NAME}"
```

  where the '${PREDEF_GRID_NAME}' is set in '{HOMErrfs}/ush/config.sh'.

- Orography files:

```
OROG_DIR="${pregen_basedir}/${PREDEF_GRID_NAME}"
```

- Surface climatology files:

```
SFC_CLIMO_DIR="${pregen_basedir}/${PREDEF_GRID_NAME}"
```

2. 'COMINgfs' for the 'nco' mode:

Machine	Path to the data ('COMINgfs')
Hera	/scratch1/NCEPDEV/hwrf/noscrub/hafs-input/COMGFS
WCOSS_dell_p3	/gpfs/dell2/emc/modeling/noscrub/UFS_SRW_App/COMGFS
WCOSS_cray	/gpfs/hps3/emc/meso/noscrub/UFS_SRW_App/COMGFS

Table 4.4: Path to the data for the WE2E tests: 'COMINgfs' for the 'nco' mode

3. User-staged external model files:

Machine	Path to the data ('extrn_mdl_source_basedir')
Hera	/scratch2/BMC/det/Gerard.Ketefian/UFS_CAM/staged_extrn_mdl_files
WCOSS_dell_p3	/gpfs/dell2/emc/modeling/noscrub/UFS_SRW_App/extrn_mdl_files
WCOSS_cray	/gpfs/hps3/emc/meso/noscrub/UFS_SRW_App/extrn_mdl_files

Table 4.5: Path to the data for the WE2E tests: user-staged external model files

4.4 Running WE2E Tests

The WE2E test script 'run_experiments.sh' assumes that all of the executables have been built in advance. If you want to run the specific tests on the list 'baselines_list.txt', you will need to make a separate list file. For each test, the script will generate an experiment directory and launch its workflow. By default, each workflow will be re-submitted via a cron job until all the tasks completes successfully. Make sure that the necessary *Python* modules should be loaded before the tests are run as shown in Section 1.1 (Step 7).

1. Create or modify the list file 'expts_fie' (for example, 'my_expts.txt') in the 'regional_workflow /tests' directory. The available tests can be found in Table 4.2.

```
cd {HOME}/regional_workflow/tests
vim my_expts.txt
```

2. Load *Python* environment for the workflow:

```
source ../../env/wflow_[machine].env
```

where the '[machine]' is 'hera', 'wcoss_dell_p3', or 'wcoss_cray'.

3. Run the tests:

```
./run_experiments.sh expts_file="my_expts.txt" machine=hera account="fv3-cam
   " use_cron_to_relaunch=TURE cron_relaunch_intvl_mnts=05 >& we2e.log &
```

where

Argument name	Description	Default
expts_file	Name of the file containing the list of experiments to run	None
machine	Machine name	None
account	HPC account	None
use_cron_to_relaunch	Set to TRUE to use crontab to relaunch the workflow	FALSE
cron_relaunch_intvl_mnts	Interval (in minutes) set in crontab to relaunch the workflow. Used only if 'use_cron_to_ relaunch' is TRUE	03

Table 4.6: Command line arguments of the WE2E tests.

Notes)

- You can create a crontab as described in Section 3.5.
- You can check any errors in generating WE2E tests with 'grep -n -i error we2e.log'.

4. If you set 'use_cron_to_relaunch' to 'TRUE', check the crobtab or the 'mycrontab' file:

```
vim /u/Chan-Hoo.Jeon/cron/mycrontab (on WCOSS Dell)
crontab -l  (on Hera / WCOSS Cray / Orion)
```

5. Check the runs in the experimental directories:

```
cd ../../../expt_dirs/[experiment dir]
```

Notes)

- The results of the WE2E tests can be found as described in 3.6
- The status of an individual WE2E test can be found in the 'log.launch_FV3LAM_wflow' file located in its experiment directory.

Chapter 5

UFS_UTILS: Grid and Static Fields with Workflow

5.1 ESG Grid and GFDL Grid Refinement

5.1.1 Adding a New Pre-defined Grid to Workflow

1. Add the name of a new grid to the list defined by 'valid_vals_PREDEF_GRID_NAME':

```
cd {HOMErrfs}/ush/
vim valid_param_vals.sh
```

2. Add the new grid to the list of grids in the case statement of '{PREDEF_GRID_NAME}':

```
cd {HOMErrfs}/ush/
vim set_predef_grid_params.sh
```

Note) There is an if-statement that checks whether it is a 'GFDLgrid' type or 'ESGgrid' type of grid. If you provide the parameters only for the 'ESGgrid' case, you should put in a 'print_err_msg_exit' call for the 'GFDLgrid'.

5.1.2 Flowchart of the Grid-generation Job in Workflow

The flowchart of the grid-generation job in the regional workflow is illustrated in Figure 5.1. As a result, the grid and mosaic files with three different halos of 3, 4, and 6 are generated in '{EXPTDIR}/grid/'. The list of executables used for the grid-generation job is as follows:

1. ‘make_hgrid’
2. ‘regional_esg_grid’
3. ‘global_equiv_resol’
4. ‘shave’

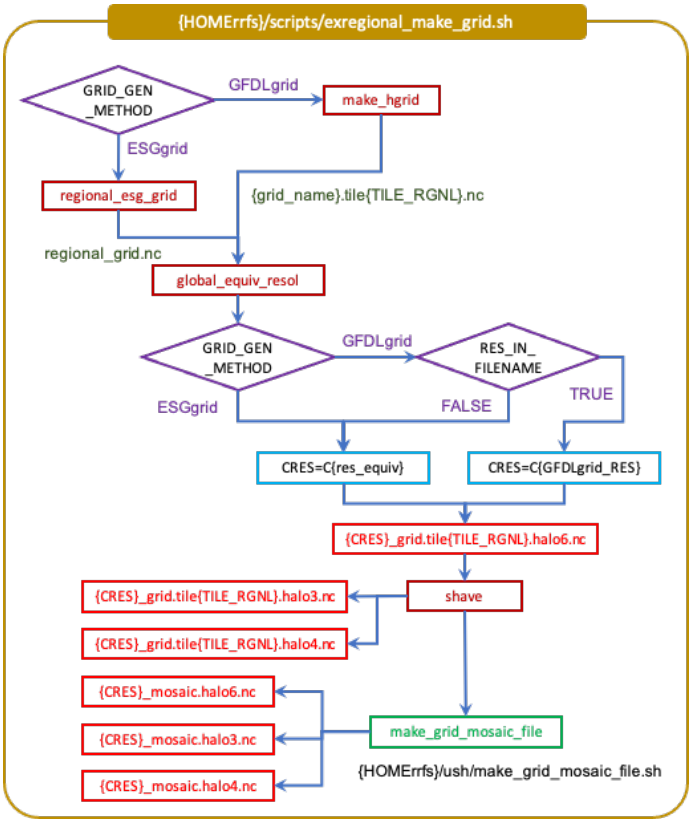

Figure 5.1: Flowchart of ‘exregional_make_grid.sh’

5.1.3 Global Equivalent Resolution

In the above flowchart, the executable ‘global_equiv_resol’ calculates an equivalent global uniform cubed-sphere resolution (unit: number of cells) for the pre-defined regional grid as shown in Eq.(5.1). This is the ‘C{res}’ that a global uniform (i.e. stretch factor of 1) cubed-sphere grid would need to

have in order to have the same nominal cell size as the average cell size of the regional grid. This value is required for the topography filtering to work correctly.

$$\text{RES\_equiv} = \text{int}\left(\frac{2.0\times\pi\times R_E}{4.0\times\text{avg\_cell\_size}}\right) \tag{5.1}$$

where R_E is the mean radius of the earth ($\approx 6371km$). Therefore, the 'RES_equiv' physically means the number of the average cells between the equator and the pole along the earth's surface. For example, the global equivalent resolution of a regional grid whose average-cell-size is $2.981km$ is C3357.

5.1.4 Parameters for an ESG grid

The default values of the parameters related to an ESG grid are set in the 'config_default.sh' file as shown in Table 2.24. In addition, the grid-specific parameters such as those in Table 2.25, and 'write-component' parameters for a new regional grid can be specified in the 'set_predef_grid_params.sh'.

1. Namelist of the ESG grid generator 'regional_grid' ('regional_grid.nml' in '{HOMErrfs}/ush/templates/'):

Name	Description
plon	Longitude of the center of the regional domain (in degrees)
plat	Latitude of the center of the regional domain (in degrees)
delx	Grid spacing in the longitudinal direction on the *FV3* super-grid (**in degrees**) $\rightarrow$ delx = $\Delta_x/(2\times6371)\times(180/\pi)$ where (Δ_x : km)
dely	Grid spacing in the latitudinal direction on the *FV3* super-grid (**in degrees**) $\rightarrow$ dely = $\Delta_y/(2\times6371)\times(180/\pi)$ where (Δ_y : km)
lx	Negative of the number of cells in the longitudinal (x) direction on the super-grid from the lower-left corner to the grid center $\rightarrow$ lx= $-(\text{nx}+2\times N_{\text{halo}})$
ly	Negative of the number of cells in the longitudinal (y) direction on the super-grid from the lower-left corner to the grid center $\rightarrow$ ly= $-(\text{ny}+2\times N_{\text{halo}})$
panzi	Parameter for the starting indexes (1, 1) (0.0=SW, 180.0=NE)

Table 5.1: Namelist of the the ESG (JP) grid generator.

Notes)

- Since the super-grid has twice the resolution of the actual grid, the grid spacing in the x direction on the actual grid will be twice this resolution.
- The physical grid spacing Δ_x on the actual grid is related to 'delx' by $\Delta_x = 2 \times \text{delx} \times (P_E/360)$ where P_E is the Earth's circumference, and the factor 2 appears due to 'delx' being the grid angle in the x direction of the super-grid. Therefore, delx = $\Delta_x/(2 \times R_E) \times (360/2\pi)$ where R_E is the radius of the Earth ($\approx 6371km$). For example, 'delx=0.0135' for the $3km$ actual grid spacing.
- These parameters can be found by *fv3grid* as shown in Section 11.1.3.
- The source code can be found in '~/UFS_UTILS/sorc/grid_tolls.fd/regional_esg_grid.fd'.

2. Parameters in 'set_predef_grid_params.sh' and 'set_gridparams_ESGgrid.sh':

No.	Name in 'set_predef_ grid_params.sh'	Name in 'set_ gridparams_ESGgrid.sh'	Name in Table 5.1
1	ESGgrid_LON_CTR	lon_ctr	plon
2	ESGgrid_LAT_CTR	lat_ctr	plat
3	ESGgrid_NX	nx	nx
4	ESGgrid_NY	ny	ny
5	ESGgrid_DELX (in **meters**)	delx (in **meters**)	Δ_x
6	ESGgrid_DELY (in **meters**)	dely (in **meters**)	Δ_y
7	ESGgrid_WIDE_HALO_WIDTH	halo_width	N_{halo}
8	ESGgrid_ALPHA_PARAM	-	a
9	ESGgrid_KAPPA_PARAM	-	k
10	-	stretch_factor	-
11	-	del_angle_x_sg	delx
12	-	del_angle_y_sg	dely
13	-	neg_nx_of_dom_with_ wide_halo	lx
14	-	neg_ny_of_dom_with_ wide_halo	ly

Table 5.2: Grid parameters for an ESG (JP) grid.

Notes)

- Make sure that 'ESGgrid_DELX' and 'ESGgrid_DELY' are in meters because the 'radius_Earth' (R_E in Table 5.1) is set in meters in 'constants.sh'.
- The variable names of 'delx' and 'dely' means totally different parameters between 'set_gridparams_ESGgrid.sh' and the namelist of 'regional_grid'. The grid spacing parameters 'delx' and 'dely' in degrees are calculated in 'set_gridparams_ESGgrid.sh' based on the input parameters 'delx' and 'dely' in meters of 'set_predef_grid_params.sh'.
- If any 'ESGgrid_XXX' parameters are not specified in 'set_predef_grid_params.sh', the default values in 'config_defaults.sh' or the pre-defined values in 'config.sh' will be used.

5.1.5 Parameters for a GFDL grid

The default values of the parameters related to a GFDL grid are set in the 'config_default.sh' file as shown in Table 2.23. In addition, the grid-specific parameters such as those in Table 2.25, and 'write-component' parameters can be specified in the 'set_predef_grid_params.sh'.

1. Namelist of the GFDL grid generator 'make_hgrid' (shown in '{HOMErrfs}/scripts/exregional_make_grid.sh'):

Parameter name	Description
grid_type	Type of topography (='gnomonic_ed': equal distance gnomonic cubic grid)
nlon	Number of model grid points (super-grid) for each zonal regions of varying resolution (2×{res})
grid_name	Name of output grid (=C{res}_grid)
do_schmidt	Set to do Schmidt transformation to create stretched grid
stretch_factor	Stretching factor for the grid
target_lon	Center longitude of the regional domain (tile 7)
target_lat	Center latitude of the regional domain (tile 7)
nest_grid	Option for a regional grid
parent_tile	Parent tile number of a regional grid (=6)
refine_ratio	Grid refinement ratio for a regional grid

istart_nest	Starting *i*-directional index of a regional grid on the parent tile super-grid (Fortran index)
jstart_nest	Ending *i*-directional index of a regional grid on the parent tile super-grid (Fortran index)
iend_nest	Starting *j*-directional index of a regional grid on the parent tile super-grid (Fortran index)
jend_nest	Ending *j*-directional index of a regional grid on the parent tile super-grid (Fortran index)
halo	Lateral boundary halo size (only for a regional grid)
great_circle_ algorithm	When specified, the 'greate_circle_algorithm' will be used to compute grid cell areas

Table 5.3: Namelist of the the GFDL grid generator.

2. Parameters in 'set_predef_grid_params.sh' and 'set_gridparams_GFDLgrid.sh':

No.	Name in 'set_predef_ grid_params.sh'	Name in 'set_gridparams_ GFDLgrid.sh'	Name in Table 5.3
1	GFDLgrid_LON_T6_CTR	lon_of_t7_ctr	target_lon
2	GFDLgrid_LAT_T6_CTR	lat_of_t7_ctr	target_lat
3	GFDLgrid_STRETCH_FAC	stretch_factor	stretch_factor
4	GFDLgrid_RES	res_of_t6g	-
5	GFDLgrid_REFINE_RATIO	refine_ratio_t6g_to_t7g	refine_ratio
6	GFDLgrid_ISTART_OF_ RGNL_DOM_ON_T6G	istart_of_t7_on_t6g	-
7	GFDLgrid_IEND_OF_RGNL_ DOM_ON_T6G	iend_of_t7_on_t6g	-
8	GFDLgrid_JSTART_OF_ RGNL_DOM_ON_T6G	jstart_of_t7_on_t6g	-
9	GFDLgrid_JEND_OF_RGNL_ DOM_ON_T6G	jend_of_t7_on_t6g	-
10	GFDLgrid_USE_GFDLgrid_ RES_IN_FILENAMES	jend_of_t7_on_t6g	-
11	-	nx_of_t7_on_t7g	-
12	-	ny_of_t7_on_t7g	-

13	-	halo_width_on_t7g	-
14	-	istart_of_t7_with_halo_on_t6sg	istart_nest
15	-	iend_of_t7_with_halo_on_t6sg	iend_nest
16	-	jstart_of_t7_with_halo_on_t6sg	jstart_nest
17	-	jend_of_t7_with_halo_on_t6sg	jend_nest

Table 5.4: Grid parameters for a GFDL grid.

Notes)

- The index numbers of 5–9 in Table 5.4 are **NOT** those on the super-grid but they are the numbers on a general grid such as orography. Therefore, if you use 'fv3grid' to set up the size of a new regional domain, you should divide the numbers on 'fv3grid' by 2.
- In the scripts for the GFDL grid-generation, 'GFDLgrid_RES' is used to set the number of cells along the longitudinal/latitudinal direction on a general grid (not super-grid). For example, 'GFDLgrid_RES' = 768 in the 'C768' grid system. The number of cells along the longitudinal (latitudianl) direction on the super-grid is 2×768.

The GFDL gird generation executable 'make_hgrid' requires the index limits such as starting and ending indices of the regional grid on the super-grid of its **parent** tile (Tile 6). The index limits on the super-grid of the parent tile are represented as

```
istart_of_t7_on_t6sg = 2 × istart_of_t7_on_t6g -1
iend_of_t7_on_t6sg = 2 × iend_of_t7_on_t6g
jstart_of_t7_on_t6sg = 2 × jstart_of_t7_on_t6g -1
jend_of_t7_on_t6sg = 2 × jend_of_t7_on_t6g
```

where 't6sg' denotes the super-grid of Tile 6, and 't6g' means the normal grid of Tile 6.

To obtain a regional grid with a halo, the index limits must include the halo number. The starting indices on the super-grid of Tile 6 with wide halo must be odd while the ending indices must be even. We subtract 1 from the starting indices if they are even (which ensures that there will be at least 'halo_width_on_t7g' halo cells along the left and bottom boundaries), and we add 1 to the ending indices if they are odd for the right and top boundaries:

```
istart_of_t7_with_halo_on_t6sg = istart_of_t7_on_t6g − halo_width_on_t6sg (−1, if even)
```

iend_of_t7_with_halo_on_t6sg = iend_of_t7_on_t6g + halo_width_on_t6sg (+1, **if** odd)
jstart_of_t7_with_halo_on_t6sg = jstart_of_t7_on_t6g − halo_width_on_t6sg (−1, **if** even)
jend_of_t7_with_halo_on_t6sg = jend_of_t7_on_t6g + halo_width_on_t6sg (+1, **if** odd)

where

halo_width_on_t6sg={2×({$NH4 = 4$} + 1)+ refine_ratio_t6g_to_t7g −1}/refine_ratio_t6g_to_t7g

To calculate the number of cells along the longitudinal (x) and latitudinal (y) directions in the regional domain:

nx_of_t7_on_t7g={(iend_of_t7_on_t6sg−istart_of_t7_on_t6sg+1)/2}×refine_ratio_t6g_to_t7g
ny_of_t7_on_t7g={(jend_of_t7_on_t6sg−jstart_of_t7_on_t6sg+1)/2}×refine_ratio_t6g_to_t7g

Note) There are two different ways to create a regional grid as shown in Table 5.5. One is to set the indices of 11 to 14 in Table 5.3 to be the same as the exact regional domain and set 'halo' to 3 as shown in Sections 6.2 and **??** ('without workflow'). The other is to include the adjusted halo number ('halo_width_on_t6sg') in the indices of 11 to 14 and set 'halo' to 1 as described in this section ('with workflow'). However, in either case, the three different domains with halo0, halo3, and halo4 will be created by the executable 'shave.x' later.

Namelist parameter	Case 1 (w/o workflow)	Case 2 (w/ workflow)
istart_nest	istart_of_t7_on_t6g	istart_of_t7_with_halo_on_t6sg
iend_nest	iend_of_t7_on_t6g	iend_of_t7_with_halo_on_t6sg
jstart_nest	jstart_of_t7_on_t6g	jstart_of_t7_with_halo_on_t6sg
jend_nest	jend_of_t7_on_t6g	jend_of_t7_with_halo_on_t6sg
halo	3	1

Table 5.5: Two different approaches to create a GFDL grid.

5.1.6 Running a Workflow for a New Grid with an Example

1. Get the parameters for a new grid from *fv3grid* as describe in Section 11.1:

   - ESG grid parameters:

Note) To match the parameters of *fv3grid* with those of the script, make sure that lx × −2 = nx and ly × −2 = ny.

Parameter	Value	Name in the script
plon	-75.0	ESGgrid_LON_CTR
plat	28.5	ESGgrid_LAT_CTR
delx	0.0135	(=3000 for 'ESGgrid_DELX')
dely	0.0135	(=3000 for 'ESGgrid_DELX')
lx	-1500	-(ESGgrid_NX)
ly	-1200	-(ESGgrid_NY)
nx	3000	2×ESGgrid_NX
ny	2400	2×ESGgrid_NY

Table 5.6: Parameters for ESG grid

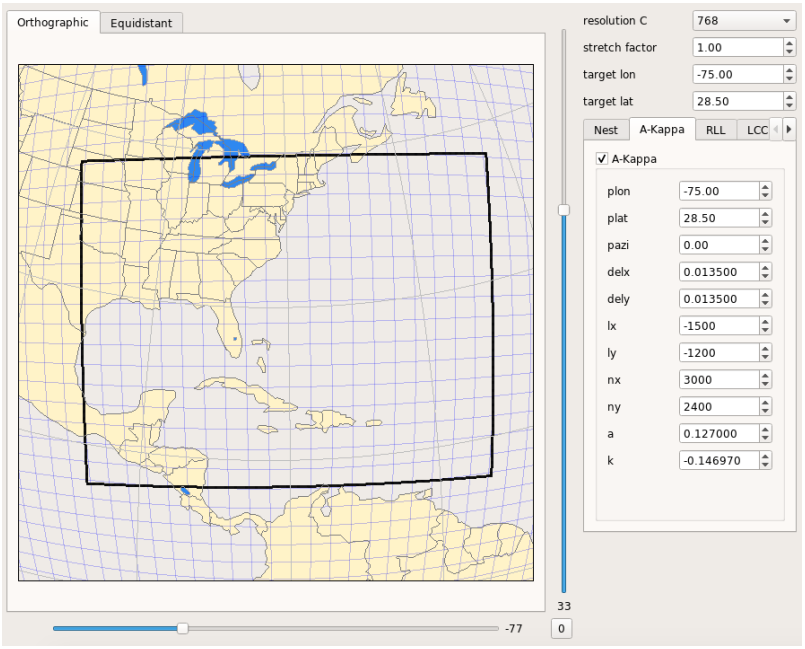

Figure 5.2: ESG grid parameters for the East Coast domain by *fv3grid*

- GFDL grid parameters:

- Since we supposed that the center of the regional domain is located at the center of the parent tile (Tile 6), we recommend to make the domain symmetry by setting 'I end'=2×'resolution C'-('I start'-1) and 'J end'=2×'resolution C'-('J start'-1).
- You can use 'num_margin_cells_T6_[left/right/bottom/top]' to make a symmetry domain along the centerline of the parent tile 6.
- The starting and ending indices (7–10 in Table 5.7) should be adjusted in 'set_gridparams_GFDLgrid.sh' to satisfy the HPC environment configuration shown in Section 2.7.

Parameter	Value	Name in the script
resolution C	768	GFDLgrid_RES
stretch factor	1.50	GFDLgrid_STRETCH_FAC
target lon	-75.0	GFDLgrid_LON_T6_CTR
target lat	28.5	GFDLgrid_LAT_T6_CTR
parent	6	-
refine ratio	3	GFDLgrid_REFINE_RATIO
I start	261	≈ 2×GFDLgrid_ISTART_OF_RGNL_DOM_ON_T6G
J start	361	≈ 2×GFDLgrid_JSTART_OF_RGNL_DOM_ON_T6G
I end	1280	≈ 2×GFDLgrid_IEND_OF_RGNL_DOM_ON_T6G
J end	1160	≈ 2×GFDLgrid_JEND_OF_RGNL_DOM_ON_T6G

Table 5.7: Parameters for GFDL grid

- Rotated Lat-Lon (RLL) parameters (for 'QUILTING'='TRUE'):

No.	Parameter	Value	Name in the script
1	tlm0d	-75.0	WRTCMP_cen_lon
2	tph0d	28.5	WRTCMP_cen_lat
3	wbd	-20.0	WRTCMP_lon_lwr_left
4	sbd	-15.5	WRTCMP_lat_lwr_left

Table 5.8: Parameters for the write component of the GFDL grid

2. Add the name of the new grid to the list defined by 'valid_vals_PREDEF_GRID_NAME':

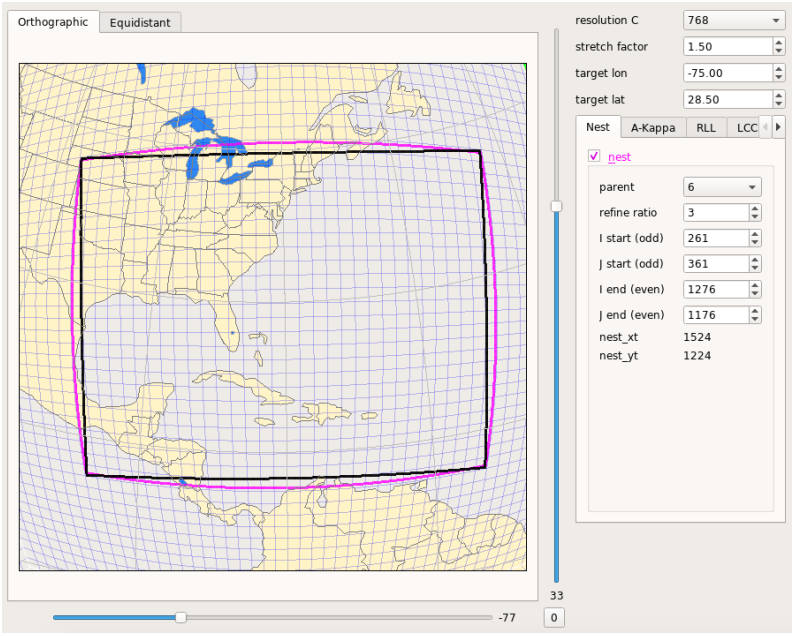

Figure 5.3: GFDL grid parameters for the East Coast domain by *fv3grid*

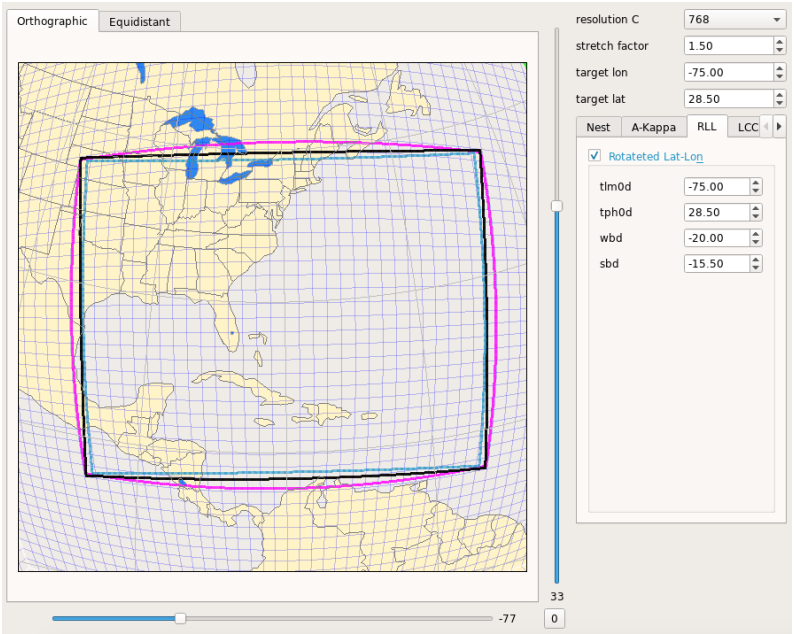

Figure 5.4: Write-component parameters for the East Coast domain by *fv3grid*

```
cd {HOMErrfs}/ush/
vim valid_param_vals.sh
(add) ‘‘JCH_ECOAST_3km’’ \
```

3. Add the new grid to the list of grids in the case statement of ‘{PREDEF_GRID_NAME}’:

```
cd {HOMErrfs}/ush/
vim set_predef_grid_params.sh
```

Add below:

```
#----------------------------------------
# Test grid for East Coast by Chan-Hoo Jeon
#----------------------------------------
#
"JCH_ECOAST_3km")
 if [ "${GRID_GEN_METHOD}" = "GFDLgrid" ]; then
   GFDLgrid_LON_T6_CTR=-75.0
   GFDLgrid_LAT_T6_CTR=28.5
   GFDLgrid_STRETCH_FAC=1.5
   GFDLgrid_RES="768"
   GFDLgrid_REFINE_RATIO=3

   num_margin_cells_T6_left=120
   GFDLgrid_ISTART_OF_RGNL_DOM_ON_T6G=$(( num_margin_cells_T6_left + 1 ))
   num_margin_cells_T6_right=120
   GFDLgrid_IEND_OF_RGNL_DOM_ON_T6G=$(( GFDLgrid_RES - num_margin_cells_T6_right ))
   num_margin_cells_T6_bottom=180
   GFDLgrid_JSTART_OF_RGNL_DOM_ON_T6G=$(( num_margin_cells_T6_bottom + 1 ))
   num_margin_cells_T6_top=180
   GFDLgrid_JEND_OF_RGNL_DOM_ON_T6G=$(( GFDLgrid_RES - num_margin_cells_T6_top ))
   GFDLgrid_USE_GFDLgrid_RES_IN_FILENAMES="TRUE"

   DT_ATMOS="18"
   LAYOUT_X="24"
   LAYOUT_Y="24"
   BLOCKSIZE=18

   if [ "$QUILTING" = "TRUE" ]; then
     WRTCMP_write_groups="1"
     WRTCMP_write_tasks_per_group=$(( 1*LAYOUT_Y ))
     WRTCMP_output_grid="rotated_latlon"
     WRTCMP_cen_lon="-75.0"
     WRTCMP_cen_lat="28.5"
     WRTCMP_lon_lwr_left="-20.0"
     WRTCMP_lat_lwr_left="-15.5"
     WRTCMP_lon_upr_rght="20.0"
     WRTCMP_lat_upr_rght="15.5"
     WRTCMP_dlon="0.02"
     WRTCMP_dlat="0.02"
   fi
 elif [ "${GRID_GEN_METHOD}" = "ESGgrid" ]; then
   ESGgrid_LON_CTR=-75.0
   ESGgrid_LAT_CTR=28.5
   ESGgrid_DELX="3000.0"
   ESGgrid_DELY="3000.0"
   ESGgrid_NX=1500
   ESGgrid_NY=1200
```

```
  ESGgrid_WIDE_HALO_WIDTH=6

  DT_ATMOS="18"
  LAYOUT_X="30"
  LAYOUT_Y="40"
  BLOCKSIZE=30

  if [ "$QUILTING" = "TRUE" ]; then
    WRTCMP_write_groups="1"
    WRTCMP_write_tasks_per_group=$(( 1*LAYOUT_Y ))
    WRTCMP_output_grid="rotated_latlon"
    WRTCMP_cen_lon="-75.0"
    WRTCMP_cen_lat="28.5"
    WRTCMP_lon_lwr_left="-20.0"
    WRTCMP_lat_lwr_left="-15.5"
    WRTCMP_lon_upr_rght="20.0"
    WRTCMP_lat_upr_rght="15.5"
    WRTCMP_dlon="0.02"
    WRTCMP_dlat="0.02"
  fi
 fi
 ;;
#
```

4. Edit 'config.community.sh'.

```
cd {HOMErrfs}/ush/
vim config.community.sh
```

- ESG (JP) grid:

```
EXPT_SUBDIR="jch_ecoast_esg"
RUN_ENVIR="community"
PREDEF_GRID_NAME="JCH_ECOAST_3km"
GRID_GEN_METHOD="ESGgrid"
RUN_TASK_MAKE_GRID="TRUE"
RUN_TASK_MAKE_OROG="TRUE"
RUN_TASK_MAKE_SFC_CLIMO="TRUE"
```

Note) Unless the pre-generated surface climatology files exist, 'RUN_TASK_MAKE_SFC_CLIMO' should be set to "TRUE".

- GFDL grid:

```
EXPT_SUBDIR="jch_ecoast_gfdl"
RUN_ENVIR="community"
PREDEF_GRID_NAME="JCH_ECOAST_3km"
GRID_GEN_METHOD="GFDLgrid"
RUN_TASK_MAKE_GRID="TRUE"
```

```
RUN_TASK_MAKE_OROG="TRUE"
RUN_TASK_MAKE_SFC_CLIMO="TRUE"
```

5. Load the machine-specific modules on HPC:

   - On Hera:

```
module use -a /contrib/miniconda3/modulefiles
module load miniconda3
conda activate regional_workflow
```

6. Generate a workflow:

```
cd {HOMErrfs}/ush/
cp config.community.sh config.sh
./generate_FV3LAM_wflow.sh
```

7. Run the workflow:

```
cd {EXPTDIR}
./launch_FV3LAM_wflow.sh
tail -n 30 log.launch_FV3LAM_wflow
```

8. The grid and mosaic files should be found in:

```
cd {EXPTDIR}/grid/
```

   Note) Once the orography files are obtained from Section 5.2, the grid and orography files can be plotted by the supporting tool shown in Section 11.2.6.

5.2 Orography

Once the grid-generation task described in Section 5.1 is complete, the orography files can be obtained by running the regional workflow again:

```
cd {EXPTDIR}
./launch_FV3LAM_wflow.sh
tail -n 30 log.launch_FV3LAM_wflow
```

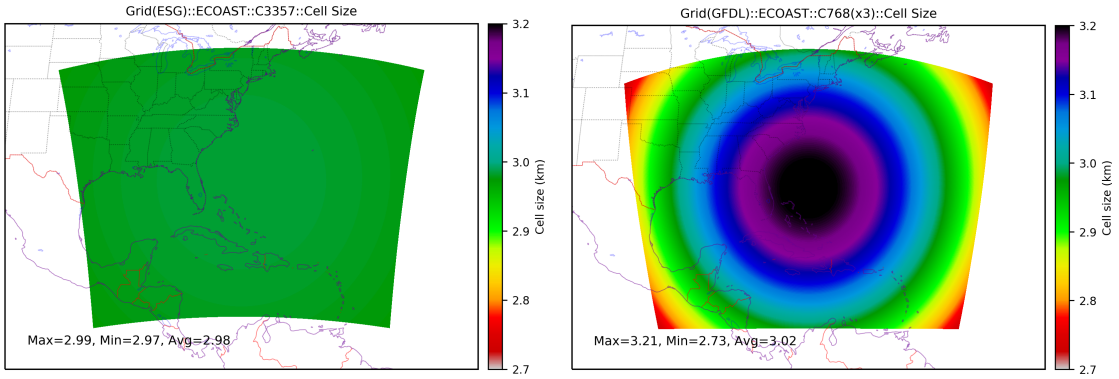

Figure 5.5: Comparison of cell sizes between ESG and GFDL

The flowchart of orography generation is shown in Figure 5.6. The list of executables used for the orography-generation job is as follows:

1. orog
2. filter_topo
3. shave

The input files required for this job are as follows:

1. C{res}_grid.tile7.halo6.nc
2. C{res}_mosaic.halo6.nc
3. Topography files (‘{FIXgsm}/../fix_orog/’):
   (a) thirty.second.antarctic.new.bin
   (b) landcover30.fixed
   (c) gmted2010.30sec.int

The orography files should be found in:

```
cd {EXPTDIR}/orog/
```

Note) Once the orography files are obtained from Section 5.2, the grid and orography files can be plotted by the supporting tool shown in Section 11.2.6.

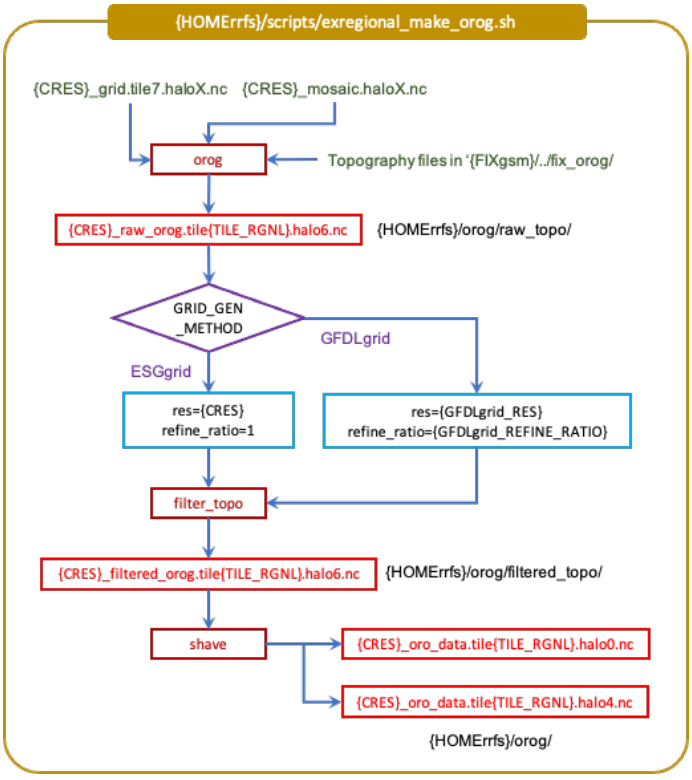

Figure 5.6: Flowchart of 'exregional_make_orog.sh'

5.3 Regional Static (Fix) Fields: Surface Climatology

Once the grid- and orography-generation tasks described in Sections 5.1 and 5.2 are complete, the regional surface climatology files can be obtained by running the regional workflow again:

```
cd {EXPTDIR}
./launch_FV3LAM_wflow.sh
tail -n 30 log.launch_FV3LAM_wflow
```

Namelist of the executable 'sfc_climo_gen':

Parameter name	Description	File name / default option
input_facsf_file	Path/name of fractional strong/weak zenith angle albedo data	facsf.1.0.nc
input_substrate_temperature_file	Path/name of soil substrate temperature data	substrte_temperature.2.6x1.5.nc
input_maximum_snow_albedo_file	Path/name of maximum snow albedo data	maximum_snow_albedo.0.05.nc

input_snowfree_albedo_file	Path/name of snow-free albedo data	snowfree_albedo.4comp.0.05.nc
input_slope_type_file	Path/name of global slope type data	slope_type.1.0.nc
input_soil_type_file	Path/name of soil type data	soil_type.statsgo.0.05.nc
input_vegetation_type_file	Path/name of vegetation type data	vegetation_type.igbp. 0.05.nc
input_vegetation_greenness_file	Path/name of monthly vegetation greenness data	vegetation_greenness.0.144.nc
mosaic_file_mdl	Path/name of the model mosaic file	{FIXlam}/C{res}_mosaic.halo4 .nc
orog_dir_mdl	Directory containing the model orography files	{FIXlam}
orog_files_mdl	List of model orography files	C{res}_oro_data.tile6.halo4.nc
halo	Number of rows/cols of the lateral boundary halo	4
maximum_snow_albedo_method	Interpolation method for this field	bilinear
snowfree_albedo_method	Interpolation method for this field	bilinear
vegetation_greenness_method	Interpolation method for this field	bilinear

Table 5.9: Namelist of 'sfc_climo_gen'.

The regional surface climatology files should be found in:

```
cd {EXPTDIR}/sfc_climo/
```

Note) The surface climatology files can be plotted by the supporting tool shown in Section 11.2.7.

5.4 Path to Machine-specific FIX Files

The paths to the machine-specific FIX files are specified in '{HOMErrfs}/ush/setup.sh' as follows:

- Hera:

Name	Path
FIXgsm	/scratch1/NCEPDEV/global/glopara/fix/fix_am
TOPO_DIR	/scratch1/NCEPDEV/global/glopara/fix/fix_orog
SFC_CLIMO_INPUT_DIR	/scratch1/NCEPDEV/global/glopara/fix/fix_sfc_climo

Table 5.10: Path to the FIX directories: Hera

- WCOSS_dell_p3:

Name	Path
FIXgsm	/gpfs/dell2/emc/modeling/noscrub/emc.glopara/git/fv3gfs/fix/fix_am
TOPO_DIR	/gpfs/dell2/emc/modeling/noscrub/emc.glopara/git/fv3gfs/fix/fix_orog
SFC_CLIMO_INPUT_DIR	/gpfs/dell2/emc/modeling/noscrub/emc.glopara/git/fv3gfs/fix/fix_sfc_climo

Table 5.11: Path to the FIX directories: WCOSS_DELL_P3

- WCOSS_cray:

Name	Path
FIXgsm	/gpfs/hps3/emc/global/noscrub/emc.glopara/git/fv3gfs/fix/fix_am
TOPO_DIR	/gpfs/hps3/emc/global/noscrub/emc.glopara/git/fv3gfs/fix/fix_orog
SFC_CLIMO_INPUT_DIR	/gpfs/hps3/emc/global/noscrub/emc.glopara/git/fv3gfs/fix/fix_sfc_climo

Table 5.12: Path to the FIX directories: WCOSS_CRAY

- Orion:

No.	Name	Path
1	FIXgsm	/work/noaa/global/glopara/fix/fix_am
2	TOPO_DIR	/work/noaa/global/glopara/fix/fix_orog
3	SFC_CLIMO_INPUT _DIR	/work/noaa/global/glopara/fix/fix_sfc_climo

Table 5.13: Path to the FIX directories: Orion

Chapter 6

UFS_UTILS: GFDL Grid and Static Fields without Workflow

6.1 Structure of Pre-processing

The structure of pro-processing for gird, mosaic, orography, and static surface climatology files is shown in Figure 6.1. This process can be achieved by running the script 'driver_grid.[machine].sh' located in the directory of '{TOP_dir}/UFS_UTILS/driver_scripts/'. This script first sets not only grid specifications such as resolution, grid type, stretch factor, refinement ratio, and indexes of the regional domain on the global coordinates, but also paths to the home and working directories. Then it submits another script ('fv3gfs_driver_grid.sh') running the sub-scripts for grid, orography, topography filtering, halo files, and static surface climatology fields one by one.

```
cd {TOP_dir}/UFS_UTILS/ush/
vim fv3gfs_driver_grid.sh
```

where

Name	Description
exec_dir	Directory for executables
topo	Name and path of the directory where the input files for orography are located ('$home_dir/fix/fix_orog')
ntiles	Number of tiles (=1)
tile	Tile number (=7)
name	(temporary) Name of the parent directory for grid and orography files

grid_dir	Path to the directory for grid and mosaic files (‘/$name/grid’)
orog_dir	Path to the directory for orography files (‘/$name/orog’)
filter_dir	Path to the directory for topography filtering (same as ‘orog_dir’)

Table 6.1: Parameters of the script for creating grid and static fields.

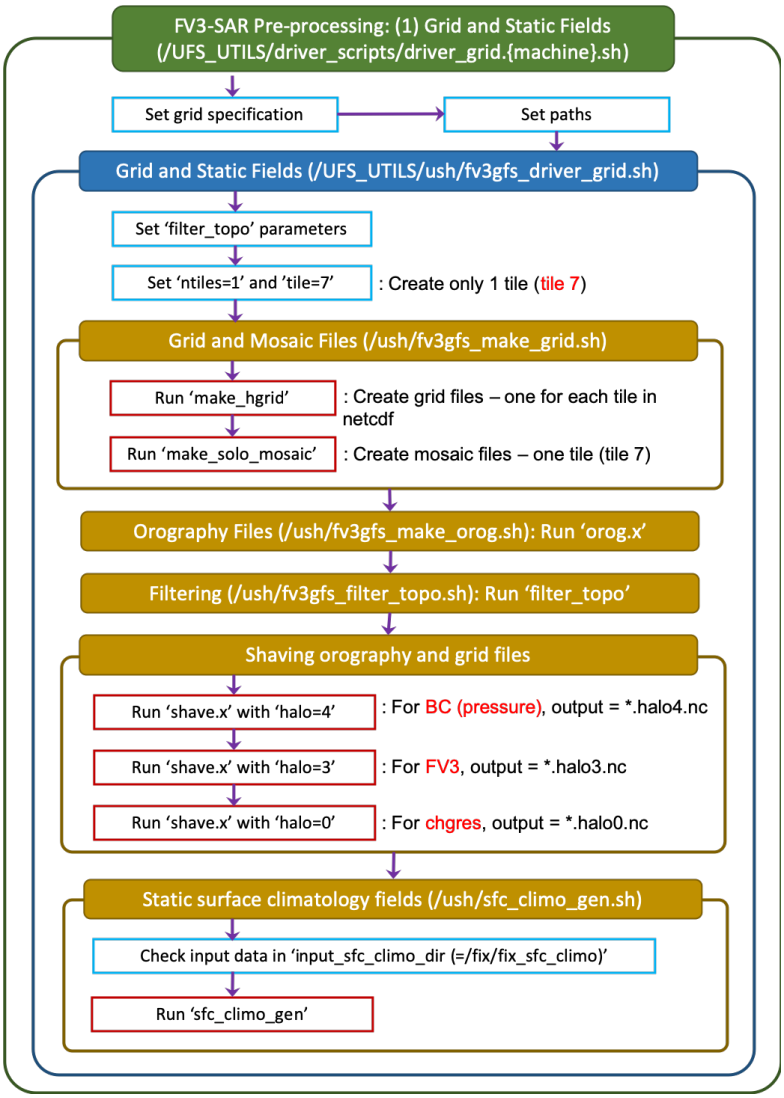

Figure 6.1: Structure of pre-processing for grid and static fields

6.2 Grid Generation

A high-resolution regional grid can be created from a low-resolution global grid by grid refinement. Refinement is a multiplicative factor named 'refine_ratio' in the scripts. Make sure that **the refined grid and its orography files keep using the resolution of the original low-resolution global grid on their file names** to avoid inaccurate filtering. For example, if a regional domain was created from a C768 domain (13km) with 'refine_ratio=3', its grid and orography files should be named 'C768_grid.tile7.haloX.nc' and 'C768_oro_data.tile7.haloX.nc', respectively, even though their resolution (3km) is three times as fine as their original domain.

Grid and mosaic files contain the longitudes and latitudes of a regional *FV3* super-grid, and other records that describe the model grid.

```
cd {TOP_dir}/UFS_UTILS/ush/
vim fv3gfs_make_grid.sh
```

1. Grid resolution of the original (global) grid:

$$\text{physical resolution (cell size}, km) = \frac{360^\circ \times 111 km}{4.0 \times \text{res}} \tag{6.1}$$

Table 6.2: Grid resolution of the original (global) grid:

Resolution ('res')	C96	C192	C384	C768	C1152	C3072
Cell size (km)	104	52	26	13	8.7	3.25

2. Grid file by 'make_hgrid':

   - Creates the grid files that contain geo-referencing records for the model grid.
   - Includes global uniform and GFDL stand-alone regional grids.
   - The grids are gnomonic that all great circles are straight lines.

   (a) Namelist of 'make_hgrid': Table 5.3

   (b) Additional parameters for the grid-generation script:

Name	Description
outdir	Working directory
exec_dir	Path to the directory where the executable is located
excutable	Grid generation program (‘make_hgrid’)
grid_dir	Directory where a grid file is created ($TMPDIR/grid))

Table 6.3: Additional parameters for grid generation.

(c) Output data:

Name	Description	Unit
x	Geographic longitude	Degrees
y	Geographic latitude	Degrees
dx	Grid edge ‘x’ distance	Meters
dy	Grid edge ‘y’ distance	Meters
area	Grid cell area	m^2
angle_dx	Grid vertex ‘x’ angle with respect to geographic east	Degrees
angle_dy	Grid vertex ‘y’ angle with respect to geographic north	Degrees

Table 6.4: Output of the ‘make_hgrid’ grid generation.

Notes)

- The source code can be found in ‘~/UFS_UTILS/sorc/fre-nctools.fd/tools/make_hgrid’.
- The variables of the namelist for the specific regional domain can be easily found by *fv3grid* as described in Section 11.1.

3. Mosaic file by ‘make_solo_mosaic’:

(a) Parameters in the script:

Name	Description
num_tiles	Number of tiles (=1)
dir	Working directory
mosaic	Output file name (=C{res}_mosaic)

tile_file	Input grid file name (=C{res}_grid.tile7.nc)

Table 6.5: Parameters for mosaic generation.

(b) Input data: The tiled grid files created by 'make_hgrid' or 'regional_esg_grid'.

(c) Output data: The output NetCDF mosaic file contains

Name	Description	Data type
mosaic	Name of mosaic	Character
gridlocation	Directory containing the grid files	Character
gridfiles	Names of each grid file	Character array
gridtiles	List of each tile number	Character array
contacts	List of tile contact regions (global grids only)	Character array
contact_index	List of contact regions as specified by i,j index (global grids only)	Character array

Table 6.6: Parameters in the output mosaic file.

Notes)

- The number of tiles is 1 for a regional grid (tile number=7), 6 for a global grid (tile number=1–6), and 7 for a nest case (tile number: 1–6 for a parent, 7 for a child).
- You can find an example of how to run 'make_solo_mosaic' independently in Section 3.4.1.
- The source code can be found in '~/UFS_UTILS/sorc/fre-nctools.fd/tools/make_solo_mosaic'.

6.3 Orography Generation

An orography file contains land mask, terrain, and gravity wave drag fields.

```
cd {TOP_dir}/UFS_UTILS/ush/
vim fv3gfs_make_orog.sh
```

where

Name	Description
$nargv	Grid type (=7 for a cubed-sphere grid, specified in 'fv3gfs_driver_grid.sh')
excutable	Program for creating orography files (='orog.x')
indir	Directory for input files (same as '$topo' in 'fv3gfs_dirver_grid.sh')

Table 6.7: Parameters of the script for orography.

- Input files:

Input file	Description	Fort
C{res}_grid.tile7.nc	Grid file	
thirty.second.antarctic.new.bin	30-arc-second Radarsat Antarctic Mapping Project (RAMP) Antarctic terrain data	fort.15
landcover30.fixed	Global 30-arc-second University of Maryland land cover data	fort.10
gmted2010.30sec.int	Global 30-arc-second USGS GMTED2010 orography data	fort.235

Table 6.8: Input files and fort numbers for orography.

- Output file: 'oro.C{res}.tile7.nc'

Name	Description	Unit
geolon	Longitudes	Degrees East
geolat	Latitudes	Degrees North
slmsk	Land-sea mask (0:nonland, 1:land)	None
land_frac	Land fraction	Fraction
orog_raw	Orography	Meters
orog_filt	Orography	Meters
stddev	Standard deviation of orography	Meters
convexity	Orographic convexity	None
oa[1–4]	Orographic asymmetry (W/S/SW/NW)	None
ol[1–4]	Orographic length scale (W/S/SW/NW)	None
Theta	Angle of mountain range with respect to East	Degrees

gamma	Anisotropy	None
sigma	Slope of orography	None
elvmax	Maximum height above mean	Meters

Table 6.9: Variables in the orography file.

Note) The source code of 'orog' can be found in '~/UFS_UTILS/sorc/orog.fd'.

6.4 Filtering Topography

```
cd {TOP_dir}/UFS_UTILS/ush/
vim fv3gfs_filter_topo.sh
```

where

Name	Description
stretch	Grid stretch factor
refine_ratio	Grid refinement ratio
excutable	Program for filtering topography (='filter_topo')
mosaic_grid	Mosaic file
topo_file	Orography file

Table 6.10: Parameters of the script for orography.

- Parameters:

Parameter	Grid resolution					
	C96	C192	C384	C768	C1152	C3072
cd4	0.12	0.15	0.15	0.15	0.15	0.15
peak_fac	1.1	1.05	1.0	1.0	1.0	1.0
max_slope	0.12	0.12	0.12	0.12	0.16	0.30
n_del2_weak	8	12	12	16	20	24

Table 6.11: Parameters for filtering topography.

- Input files:

Input file	Description
C{res}_grid.tile7.nc	Grid file
C{res}_mosaic.nc	Mosaic file
oro.C{res}.tile7.nc	Orography file

Table 6.12: Input files for filtering topography.

- Output file: 'oro.C{res}.tile7.nc'

6.5 Halo Files for Boundary, *FV3*, and *chgres_cube*

Three types of orography and grid files (halo0, halo3, and halo4) are created by the executable 'shave' for initial and boundary fields. The 'grid' and 'orography' files for regional grids are first created with rows and columns extending beyond the halo. This is required for the topography filtering code to work correctly in the halo region. After filtering, the shave program removes these extra points from the files.

```
cd {TOP_dir}/UFS_UTILS/ush/
vim fv3gfs_driver_grid.sh
```

where

Name	Description
npts_cgx	Number of grid points (*i*-index, latitudinal)
npts_cgy	Number of grid points (*j*-index, longitudinal)
halo	Lateral boundary halo size (=3)
halop1	halo+1 (=4)

Table 6.13: Parameters for halo files.

Steps:

1. Create a domain with 4 boundary rows/columns for pressure (halo4).
    - C{res}_oro_data.tile7.halo4.nc

- C{res}_grid.tile7.halo4.nc

2. Create a domain with 3 boundary rows/columns for lateral boundary conditions of *FV3* (halo3).

   - C{res}_oro_data.tile7.halo3.nc
   - C{res}_grid.tile7.halo3.nc

3. Create a domain with 0 boundary row/column for *chgres* (halo0).

   - C{res}_oro_data.tile7.halo0.nc
   - C{res}_grid.tile7.halo0.nc

6.6 Regional Static (Fix) Fields: Surface Climatology

Time-independent surface climatology fields such as soil and vegetation types for a regional domain are created by the executable 'sfc_climo_gen':

```
cd {TOP_dir}/UFS_UTILS/ush/
vim sfc_climo_gen.sh
```

where

Name	Description
input_sfc_climo_dir	Directory for input files (=$home_dir/fix/fix_sfc_climo)
FIX_FV3	$out_dir
SAVE_DIR	$sout_dir/fix_sfc
mosaic_file_mdl	$mosaic_file
orog_dir_mdl	$FIX_FV3
orog_files_mdl	Orography file (='C{res}_oro_data.tile7.nc')
halo	Number of row/cols of the lateral boundary halo
maximum_snow_albedo_method	Interpolation method (default="bilinear")
snowfree_albedo_method	Interpolation method (default="bilinear")
vegetation_greenness_method	Interpolation method (default="bilinear")

Table 6.14: Parameters for static surface fields.

- The namelist of 'sfc_climo_gen' is described in Table 5.9.
- Input files:

Input file	Description
facsf.1.0.nc	Global 1-degree fractional coverage strong/weak zenith angle albedo
maximum_snow_albedo.0.05.nc	Global -0.05-degree maximum snow albedo
substrate_temperature.2.6x1.5.nc	Global 2.6×1.5-degree soil substrate temperature
snowfree_albedo.4comp.0.05.nc	Global 0.05-degree four component monthly snow-free albedo
slope_type.1.0.nc	Global 1.0-degree categorical slope type
soil_type.statsgo.0.05.nc	Global 0.05-degree categorical STATSGO soil type
vegetation_type.igbp.0.05.nc	Global 0.05-degree categorical IGBP vegetation type
vegetation_greenness.0.144.nc	Global 0.144-degree monthly vegetation greenness in percent
Model mosaic file	
Model orography files	

Table 6.15: Input files for creating regional surface climatology.

- Output files:

Output file	Description
C{res}.facsf.1.0.nc	Fractional coverage strong/weak zenith angle albedo
C{res}.substrate_temperature.2.6x1.5.nc	Maximum snow albedo
C{res}.maximum_snow_albedo.0.05.nc	Soil substrate temperature
C{res}.snowfree_albedo.4comp.0.05.nc	Snow free albedo
C{res}.slope_type.1.0.nc	Slope type
C{res}.soil_type.statsgo.0.05.nc	Soil type
C{res}.vegetation_type.igbp.0.05.nc	Vetetation type
C{res}.vegetation_greenness.0.144.nc	Vegetation greenness

Table 6.16: Output files for filtering topography.

Notes)

- The source code of 'sfc_climo_gen' can be found in '~/UFS_UTILS/sorc/sfc_climo_gen.fd'.
- The 'sfc_climo_gen' program only runs with an MPI task count that is a multiple of six. This is a requirement of the ESMP libraries. Large grids may require tasks spread across multiple nodes.

Chapter 7

UFS_UTILS: Initial and Lateral Boundary Fields

7.1 External Model Data for IC/LBC in Workflow

Path to the directory ('EXTRN_MDL_SOURCE_BASEDIR_ICS/LBCS') and file name ('EXTRN_MDL_FILES_ICS/LBCS') for the external model data for initial conditions (ICS) and lateral boundary conditions (LBCS) can be specified in '{HOMErrfs}/ush/config.sh' as shown in Tables 7.1 and 7.2. Each machine provides only data from the last few days in the directory shown in Table 7.1. You should check if the data that you need exist in the directory. If you want to run the regional workflow for other dates, these paths should be modified to point to the data set.

- Path to the directory of external data:

Model	Machine	External data directory
GFS	Hera	/scratch1/NCEPDEV/rstprod/com/gfs/prod/gfs.{YYYYMMDD}/{HH}
	WCOSS	/gpfs/dell1/nco/ops/com/gfs/prod/gfs.{YYYYMMDD}/{HH}
	Orion	-
RAP	Hera	/scratch1/NCEPDEV/rstprod/com/rap/prod/rap.{YYYYMMDD}
	WCOSS	/gpfs/hps/nco/ops/com/rap/prod/rap.{YYYYMMDD}
	Orion	-
HRRR	Hera	/scratch1/NCEPDEV/rstprod/com/hrrr/prod/hrrr.{YYYYMMDD}/conus
	WCOSS	/gpfs/hps/nco/ops/com/hrrr/prod/hrrr.{YYYYMMDD}/conus

	Orion	-
NAM	Hera	/scratch1/NCEPDEV/rstprod/com/nam/prod/nam.{YYYYMMDD}
	WCOSS	/gpfs/dell1/nco/ops/com/nam/prod/nam.{YYYYMMDD}
	Orion	-

Table 7.1: External model data for IC and LBC

- File name:

Model	Type	Format	File name
GFS	IC ('ANL')	nemsio	gfs.t{HH}z.{'atm'/'sfc'}anl.nemsio
		grib2	gfs.t{HH}z.pgrb2.0p25.f000
	LBC ('FCST')	nemsio	gfs.t{HH}z.atmf{fcst_hhh}.nemsio
		grib2	gfs.t{HH}z.pgrb2.0p25.f{fcst_hhh}
RAP	IC ('ANL')	-	{YY}{DDD}{HHH}{mn}{fcst_hh}{fcst_mn}
	LBC ('FCST')	-	{YY}{DDD}{HHH}{mn}{fcst_hh}{fcst_mn}
HRRR	IC ('ANL')	-	{YY}{DDD}{HHH}{mn}{fcst_hh}{fcst_mn}
	LBC ('FCST')	-	{YY}{DDD}{HHH}{mn}{fcst_hh}{fcst_mn}
NAM	IC ('ANL')	-	nam.t{HH}z.btrdsfi{HH}
	LBC ('FCST')	-	nam.t{HH}z.bgrdsf

Table 7.2: File name of the external model data

7.2 Global Time-dependent Data

7.2.1 GFS Data: GRIB2

1. 0.25° data (last 10days only): 'gfs.t{HH}z.pgrb2.0p25.f{XXX}'

```
https://nomads.ncep.noaa.gov/pub/data/nccf/com/gfs/prod/gfs.{YYYYMMDD}/{HH}/
```

Download on HPC:

```
cd [directory where data will be downloaded]
wget -c https://nomads.ncep.noaa.gov/pub/data/nccf/com/gfs/prod/gfs.{
    YYYYMMDD}/{HH}/[file name]
```

2. 0.5° data: 'gfs_4_{YYYYMMDD}_00{HH}_{XXX}.grb2'

```
https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/global-
    forcast-system-gfs/
# Select 'Data Access Links' at 'GFS Forecasts/004-Domain'
```

Download from 'HTTPS' on HPC:

```
cd [directory where data will be downloaded]
wget -c https://nomads.ncdc.noaa.gov/data/gfs4/{YYYYMM}/{YYYYMMDD}/[file
    name]
```

3. 1.0° data: 'gfs_3_{YYYYMMDD}_00{HH}_{XXX}.grb2':

   - Access GFS Dataset (https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/global-forcast-system-gfs/) and select 'Data Access Links' at 'GFS Forecasts/003-Domain'
   - Download from 'HTTPS' on HPC:

```
cd [directory where data will be downloaded]
wget -c https://nomads.ncdc.noaa.gov/data/gfs-avn-hi/{YYYYMM}/{YYYYMMDD
    }/[file name]
```

7.2.2 GFS Data: NEMSIO

1. T1534 gaussian (last 10 days only)

```
https://nomads.ncep.noaa.gov/pub/data/nccf/com/gfs/prod/gfs.{YYYYMMDD}/{HH}/
```

   - Atmospheric fields: 'gfs.t{HH}z.atmanl.nemsio'
   - Surface fields: 'gfs.t{HH}z.sfcanl.nemsio'

Download on HPC:

```
cd [directory where data will be downloaded]
wget -c https://nomads.ncep.noaa.gov/pub/data/nccf/com/gfs/prod/gfs.{
    YYYYMMDD}/{HH}/[file name]
```

7.3 *chgres_cube*

The program *chgres_cube* creates initial condition (IC) files to 'cold-start' the forecast model. It converts atmospheric, surface, and nst data. The configuration of *chgres_cube* for a Regional Domain. The format of the input data can be Global Forecast System (GFS) gridded binary version 2 (GRIB2), NOAA Environmental Modeling System Input/Output (NEMSIO), or Network Common Data Form (NetCDF).

- References

  1. /UFS_UTILS/sorc/chgres_cube.fd/program_setup.f90
  2. /UFS_UTILS/ush/chgres_cube.sh
  3. https://github.com/NOAA-EMC/UFS_UTILS/blob/release/public-v1/docs/source/chgres_cube.rst

No.	Parameter name	Description	Usage (data type)
1	fix_dir_target_grid	Directory (path and name) containing pre-computed fixed data on the target grid (e.g. soil type)	GRIB2 / NEMSIO / NetCDF
2	mosaic_file_target_grid	Mosaic file (path and name) for the target grid	GRIB2 / NEMSIO / NetCDF
3	orog_dir_target_grid	Directory (path and name) containing orography files for the input grid.	GRIB2 / NEMSIO / NetCDF
4	orog_files_target_grid	Orography files (name) for the target grid	GRIB2 / NEMSIO / NetCDF
5	vcoord_files_target_grid	Vertical coordinate definition file (path and name)	GRIB2 / NEMSIO / NetCDF
6	data_dir_input_grid	Directory (path and name) containing atm or sfc files on the input grid	GRIB2 / NEMSIO / NetCDF
7	grib2_file_input_grid	File name of GRIB2 input data	GRIB2
8	varmap_file	Directory (path and name) containing the variable mapping (VARMAP) table	GRIB2
9	atm_files_input_grid	File name of input atmospheric data Not used for INPUT_TYPE="restart"	NEMSIO / NetCDF
10	sfc_files_input_grid	File name of input surface / Near Sea Surface Temperature (NSST) data	NEMSIO / NetCDF
11	input_type	Input data type: • 'restart' - Tiled warm restart files (netcdf) • 'history' - Tiled history files (netcdf) • 'gaussian_nemsio' - Gaussian NEMSIO files • 'gaussian_netcdf' - Gaussian NetCDF files • 'grib2' - FV3GFS GRIB2 files • 'gfs_gaussian_nemsio' - Spectral GFS NEMSIO files	GRIB2 / NEMSIO / NetCDF

		• 'gfs_sigio' - Spectral GFS sigio/sfcio files	
12	cycle_mon	Cycle month	GRIB2 / NEMSIO / NetCDF
13	cycle_day	Cycle day	GRIB2 / NEMSIO / NetCDF
14	cycle_hour	Cycle hour	GRIB2 / NEMSIO / NetCDF
15	convert_atm	Convert atmospheric data when 'true'.	GRIB2 / NEMSIO / NetCDF
16	convert_sfc	Convert surface data when 'true'.	GRIB2 / NEMSIO / NetCDF
17	convert_nst	Convert nst data when 'true'.	NEMSIO / NetCDF
18	tracers_input	Names of atmospheric tracers in the input file. In most cases, they are different from those of tracers	NEMSIO / NetCDF
19	tracers	Names of atmospheric tracers to be processed. These names will be used to identify the tracer records in the output files	NEMSIO / NetCDF
20	regional	For regional target grids: • '0' : uniform, stretch, or nest (for global domain) • '1' : Creating initial and lateral boundary files from atmospheric/surface data • '2' : Creating lateral boundary file only	GRIB2 /NEMSIO / NetCDF
21	halo_bndy	Number of row/columns of lateral halo where pure lateral boundary conditions are applied (regional target grids only)	GRIB2 / NEMSIO / NetCDF
22	halo_blend	Number of row/columns of blending halo where model tendencies and lateral boundary tendencies are applied. (regional target grids only)	GRIB2 / NEMSIO / NetCDF

Table 7.3: Namelist of *chgres_cube* for a regional domain.

Notes)

- Some caveats in initializing with GRIB2 data (from Reference 3)

  1. GRIB2 data do not contain the fields required for the NSST scheme.
  2. Relatively low-resolution data, compared to the NCEP operation GFS, may result in inaccurate initialization especially for the surface fields.

Grid resolution	Recommendation
C96	0.25, 0.5, or 1.0 degree
C192	0.25, or 0.5 degree
C384	0.25 degree
C768	0.25 degree (may not work well)

Table 7.4: Recommended resolution of GRIB2 data

3. Sea/lake ice thickness and column temperatures are not available. Nominal values of $1.5m$ and $265K$ are used.
4. Soil moisture is evaluated using bilinear interpolation.
5. Ozone is not available at all isobaric levels. Missing levels are set to a nominal value defined in the variable mapping (VARMAP) file.
6. It is not guaranteed to use old GFS data (older than GFS v14).

7.4 Initial Conditions: Cold Start

Creates initial conditions (ICs and BC at the initial time) for a regional domain using global GFS data:

```
cd {HOMEfv3}/scripts/
vim exregional_make_ic.sh
```

where {HOMEfv3} is the home directory of the regional_workflow package.

Set the links to use the grid, orography, and static field files with **FOUR (4)** halos:

```
ln −sf {FIXsar}/C{res}_grid.tile7.halo4.nc {FIXsar}/C{res}_grid.tile7.nc
ln −sf {FIXsar}/C{res}_oro_data.tile7.halo4.nc {FIXsar}/C{res}_oro_data.tile7.nc
ln −sf {FIXsar}/C{res}.[variables].tile7.halo4.nc {FIXsar}/C{res}.[variables].tile7.nc
```

- List of the static fields:

Original name	Transferred name
vegetation_greenness.tile7.halo4.nc	C{res}.vegetation_greenness.tile7.nc
soil_type.tile7.halo4.nc	C{res}.soil_type.tile7.nc
slope_type.tile7.halo4.nc	C{res}.slope_type.tile7.nc
substrate_temperature.tile7.halo4.nc	C{res}.substrate_temperature.tile7.nc
facsf.tile7.halo4.nc	C{res}.facsf.nc
maximum_snow_albedo.tile7.halo4.nc	C{res}.maximum_snow_albedo.tile7.nc
snowfree_albedo.tile7.halo4.nc	C{res}.snowfree_albedo.tile7.nc
vegetation_type.tile7.halo4.nc	C{res}.vegetation_type.tile7.nc

Table 7.5: Static fields transferred to the INPUT directory.

- Other input files:

  1. C{res}_mosaic.nc (mosaic file)
  2. global_hyblev.l{LEVS}.txt (vertical coordinate definition file)
  3. GFSphys_var_map.txt (only for GRIB2)

Create namelist and run *chgres_cube*:

- Namelist for *chgres_cube* (GRIB2):

Parameter	Input
mosaic_file_target_grid	"{FIXsar}/C{res}_mosaic.nc"
fix_dir_target_grid	"{FIXsar}"
orog_dir_target_grid	"{FIXsar}"
orog_files_target_grid	"C{res}_oro_data.tile7.halo4.nc"
vcoord_file_target_grid	"{FIXam}/global_hyblev.l{LEVS}.txt"
data_dir_input_grid	"gfs.{ymd}/{hour}"
grib2_file_input_grid	"gfs.t{cyc}z.pgrb2.0p25.f{hour_name}"
varmap_file	"{UFS_UTILS}/parm/varmap_tables/GFSphys_var_map.txt"
input_type	"grib2"
cycle_mon	{month}
cycle_day	{day}
cycle_hour	{cyc}
convert_atm	**.true.**
convert_sfc	**.false.**
convert_nst	**.false.**
regional	**1**
halo_bndy	4
halo_blend	10

Table 7.6: Namelist for *chgres_cube* with GRIB2 data.

- Namelist for *chgres_cube* (NEMSIO):

Parameter name	Input
mosaic_file_target_grid	"{FIXsar}/C{res}_mosaic.nc"
fix_dir_target_grid	"{FIXsar}"
orog_dir_target_grid	"{FIXsar}"
orog_files_target_grid	"C{res}_oro_data.tile7.halo4.nc"
vcoord_file_target_grid	"{FIXam}/global_hyblev.l{LEVS}.txt"
data_dir_input_grid	"{ANLDIR}"
atm_files_input_grid	"gfs.t{cyc}z.atmanl.nemsio"
sfc_files_input_grid	"gfs.t{cyc}z.sfcanl.nemsio"
cycle_mon	{month}
cycle_day	{day}
cycle_hour	{cyc}
convert_atm	**.false. (.true.)**
convert_sfc	**.true.**
convert_nst	**.true.**
input_type	"gaussian_nemsio"
tracers	"sphum", "liq_wat", "o3mr", "ice_wat", "rainwat", "snowwat", "graupel"
tracers_input	"spfh", "clwmr", "o3mr", "icmr", "rwmr", "snmr", "grle"
regional	**1**
halo_bndy	4
halo_blend	10

Table 7.7: Namelist for *chgres_cube* with NEMSIO data.

Notes)

- The list of 'tracers' and 'tracers_input' can be found in Table 2.53.
- 'regional=1' because both initial and boundary conditions are evaluated at the initial time.

Rename output files and move them to the INPUT directory for *FV3*:

```
mv gfs_ctrl.nc {INPdir}/.
```

```
mv gfs.bndy.nc {INPdir}/gfs_bndy.tile7.000.nc
mv out.atm.tile1.nc {INPdir}/gfs_data.tile7.nc
mv out.sfc.tile1.nc {INPdir}/sfc_data.tile7.nc
```

- Output files:

  1. gfs_ctrl.nc
  2. gfs.bndy.nc (boundary condition at the initial time)
  3. out.atm.tile1.nc (initial conditions of time-dependent fields inside the regional domain)
  4. out.sfc.tile1.nc (initial conditions of static fields inside the regional domain)

7.5 Lateral Boundary Conditions

Creates lateral boundary conditions every 3 hours after initialization for a regional domain using global GFS data:

```
cd {HOMEfv3}/scripts/
vim exregional_make_bc.sh
```

where {HOMEfv3} is the home directory of the regional_workflow package.

Set the links to use the grid, and orography files with **FOUR (4)** halos:

```
# Grid
ln -sf {FIXsar}/C{res}_grid.tile7.halo4.nc {FIXsar}/C{res}_grid.tile7.nc
# Orography
ln -sf {FIXsar}/C{res}_oro_data.tile7.halo4.nc {FIXsar}/C{res}_oro_data.tile7.nc
```

- Other input files:

  1. C{res}_mosaic.nc (mosaic file)
  2. global_hyblev.l{LEVS}.txt (vertical coordinate definition file)

Create namelist and run *chgres_cube*:

- Namelist for *chgres_cube* (GRIB2):

Parameter name	Input
mosaic_file_target_grid	"{FIXsar}/C{res}_mosaic.nc"
fix_dir_target_grid	"{FIXsar}"
orog_dir_target_grid	"{FIXsar}"
orog_files_target_grid	"C{res}_oro_data.tile7.halo4.nc"
vcoord_file_target_grid	"{FIXam}/global_hyblev.l{LEVS}.txt"
data_dir_input_grid	"gfs.{ymd}/{hour}"
grib2_file_input_grid	"gfs.t{cyc}z.pgrb2.0p25.f{hour_name}"
varmap_file	"{UFS_UTILS}/parm/varmap_tables/GFSphys_var_map.txt"
input_type	"grib2"
cycle_mon	{month}
cycle_day	{day}
cycle_hour	{cyc}
convert_atm	**.true.**
convert_sfc	**.false.**
convert_nst	**.false.**
regional	**2**
halo_bndy	4
halo_blend	10

Table 7.8: Namelist for *chgres_cube* with GRIB2 data.

- Namelist for *chgres_cube* (NEMSIO):

Variable	Input
mosaic_file_target_grid	"{FIXsar}/C{res}_mosaic.nc"
fix_dir_target_grid	"{FIXsar}"
orog_dir_target_grid	"{FIXsar}"
orog_files_target_grid	"C{res}_oro_data.tile7.halo4.nc"
vcoord_file_target_grid	"{FIXam}/global_hyblev.l{LEVS}.txt"
data_dir_input_grid	"{INIDIR}"
atm_files_input_grid	"gfs.t{cyc}z.atmf{hour_name}.nemsio"

sfc_files_input_grid	“gfs.t{cyc}z.sfcanl.nemsio”
cycle_mon	{month}
cycle_day	{day}
cycle_hour	{cyc}
convert_atm	**.true.**
convert_sfc	**.false.**
convert_nst	**.false.**
input_type	“gaussian_nemsio”
tracers	“sphum”, “liq_wat”, “o3mr”, “ice_wat”, “rainwat”, “snowwat”, “graupel”
tracers_input	“spfh”, “clwmr”, “o3mr”, “icmr”, “rwmr”, “snmr”, “grle”
regional	**2**
halo_bndy	4
halo_blend	10

Table 7.9: Namelist for *chgres_cube* with NEMSIO data.

Note) ‘regional=2’, ‘convert_sfc=.false.’, and ‘convert_nst=.false.’ for boundary conditions.

Rename output files and move them to the INPUT directory for *FV3*:

```
mv gfs.bndy.nc {INPdir}/gfs\_bndy.tile7.{hour_name}.nc
```

- Output files:

  1. gfs.bndy.nc (boundary condition at the certain hour)

Chapter 8

Unified Post Processor (UPP)

8.1 Converting into GRIB2

Reference) https://upp.readthedocs.io/en/ufs-v1.0.0/InputsOutputs.html

- On Hera:

```
cd /home/Chan-hoo.Jeon/fv3sar/regional_scripts
vim post.regional_{domain}_{res}.hera
# Check parameters
sh post.regional_{domain}_{res}.hera
```

- On Orion:

```
cd /home/chjeon/regional_scripts
vim post.regional_{domain}_{res}.orion
# Check parameters
sh post.regional_{domain}_{res}.orion
```

where

Parameter name	Description
fhr	Hour of the input data (2 digits, 'XX')
domain	Domain name
RUN	Case name
INPUT_DATA	Path to the directory where output files are created

PARMfv3	Path to the directory where UPP input files are located

Table 8.1: Parameters in the script of UPP.

1. Input files

   (a) 'itag' namelist: created in the script

   - Path and name of the *FV3* **pressure-level** output file ({DIR}/dynf0XXX.nc)
   - Format of the *FV3* output (netcdf, binarynemsio)
   - Format of *UPP* output (grib2)
   - Forecast valid time in *FV3* format (not the model start time but the forecast time desired to be post-processed; YYYY-MM-DD_HH:00:00)
   - Name of the dynamic core
   - Path and name of the *FV3* **surface** output file ({DIR}/phyf0XXX.nc)
   - Name of configuration file ('postxconfig-NT.txt')

   (b) GRIB2 control file: 'postxconfig-NT.txt'

   - GRIB2 output table ('dtcenter.org/sites/default/files/community-code/upp-grib2-table_0.pdf')
     i. BGRD3DXX.tmXX

Field description	Type of level	Short name	nlvl
Temperature on model surface	hybrid	t	npz
Temperature on pressure surface	isobaricInhPa	t	4
Temperature in boundary layer	pressureFromGroundLayer	t	6
Dew point temperature in boundary layer	pressureFromGroundLayer	dpt	6
Specific humidity on model surface	hybrid	q	npz
Specific humidity on pressure surface	isobaricInhPa	q	4
Specific humidity in boundary layer	pressureFromGroundLayer	q	6
Cloud mixing ratio on model surface	hybrid	clwmr	npz
Rain mixing ratio on model surface	hybrid	rwmr	npz
Snow mixing ratio on model surface	hybrid	snmr	npz
Turbulent kinetic energy on model surface	hybrid	tke	npz

U component of wind on model surface	hybrid	u	npz
U component of wind on pressure surface	isobaricInhPa	u	4
U wind at flight levels	heightAboveSea	u	10
U wind in boundary layer	pressureFromGroundLayer	u	6
V component of wind on model surface	hybrid	v	npz
V component of wind on pressure surface	isobaricInhPa	v	4
V wind at flight levels	heightAboveSea	v	10
V wind in boundary layer	pressureFromGroundLayer	v	6

Table 8.2: Grib2 fields of 'BGRD3D' produced by Unipost (selected).

ii. BGDAWPXX.tmXX

Field description	Type of level	Short name	nlvl
Temperature on pressure surface	isobaricInhPa	t	46
Temperature at specific height	heightAboveGround	t	14
Temperature in boundary layer	pressureFromGroundLayer	t	6
Relative humidity on pressure surface	isobaricInhPa	rh	46
Dew point temperature on pressure surface	isobaricInhPa	dpt	46
Specific humidity on pressure surface	isobaricInhPa	q	46
Specific humidity at specific height	heightAboveGround	q	14
Specific humidity in boundary layer	pressureFromGroundLayer	q	6
U component of wind on pressure surface	isobaricInhPa	u	46
U component of wind at specific height	heightAboveGround	u	14
U component of wind in boundary layer	pressureFromGroundLayer	u	6
U component of storm motion	heightAboveGroundLayer	ustm	1
V component of wind on pressure surface	isobaricInhPa	v	46
V component of wind at specific height	heightAboveGround	v	14
V component of wind in boundary layer	pressureFromGroundLayer	v	6
V component of storm motion	heightAboveGroundLayer	vstm	1
Vertical velocity in boundary layer	pressureFromGroundLayer	w	6
Absolute vorticity in boundary layer	isobaricInhPa	absv	6

Turbulent kinetic energy on pressure surface	isobaricInhPa	tke	46
Cloud mixing ratio on pressure surface	isobaricInhPa	clwmr	46
Ice water mixing ratio	isobaricInhPa	icmr	46
Rain mixing ratio on pressure surface	isobaricInhPa	rwmr	46
Snow mixing ratio on pressure surface	isobaricInhPa	snmr	46
2M temperature	heightAboveGround	2t	1
2M dew point temperature	heightAboveGround	2d	1
2M relative humidity	heightAboveGround	2r	1
10M *u* component wind	heightAboveGround	10u	1
10M *v* component wind	heightAboveGround	10v	1
10M wind speed	heightAboveGround	10si	1
Total cloud cover	"unknown"	tcc	1
Total precipitation	surface	tp	1
Storm surface runoff	surface	ssrun	1
Sea surface temperature	surface	sst	1
Derived radar reflectivity	hybrid	refd	2
Maximum/Composite radar reflectivity	"unknown"	refc	1
Derived radar reflectivity backscatter from rain	"unknown"	refzr	1
Derived radar reflectivity backscatter from ice	"unknown"	refzi	1
Derived radar reflectivity	isothermal	refd	1

Table 8.3: Grib2 fields 'BGDAWP' produced by Unipost (selected).

In the above tables, 'nlvl' is the total number of layers, and 'npz' is the number of vertical layers in a case.

(c) Look-up table: 'eta_micro_lookup.dat'

(d) Coefficient files for satellite

2. Regridding

If necessary, the *UPP* output can be interpolated onto a different grid with *wgrib2*.

(a) General Format:

- Latitude-longitude grid

```
-new_grid latlon lon0:nlon:dlon lat0:nlat:dlat outfile
```

where

Name	Description	Format
lon0	Longitude of the first grid point	In degree
nlon	Total number of grid points in the longitudinal direction	Integer
dlon	Longitudinal distance between two adjacent grid points	Float
lat0	Latitude of the first grid poin	In degree
nlat	Total number of grid points in the latitudinal direction	Integer
dlat	Latitudinal distance between two adjacent grid points	Float

Table 8.4: Parameters for a longitude-latitude grid.

- Lambert conic grid

```
-new_grid lambert:lov:latin1:latin2 lon0:nx:dx lat0:ny:dy outfile
```

where

Name	Description	Format
lov	Longitude where *y* axis is parallel to meridian	In degrees
latin1	First latitude from pole which cuts the secant cone	In degrees
latin2	Second latitude from pole which cuts the secant cone	In degrees
lon0	Longitude of the first grid point	In degrees
lat0	Latitude of the first grid point	In degrees
nx	Total number of grid points along *x*	Integer
ny	Total number of grid points along *y*	Integer
dx	Longitudinal distance between two adjacent grid points	In meters
dy	Latitudinal distance between two adjacent grid points	In meters

Table 8.5: Parameters for Lambert conic grid.

- Polar stereographic grid

```
-new_grid nps(or sps):lov:lad lon0:nx:dx lat0:ny:dy outfile
```

where

Name	Description	Format
nps / sps	North / South polar stereographic	
lov	Longitude where y axis is parallel to meridian	In degrees
lad	Latitude where dx and dy are specified	60(nps) / -60(sps)
lon0	Longitude of the first grid point	In degrees
lat0	Latitude of the first grid point	In degrees
nx	Total number of grid points along x	Integer
ny	Total number of grid points along y	Integer
dx	Longitudinal distance at 'lad'	In meters
dy	Latitudinal distance at 'lad'	In meters

Table 8.6: Parameters for Polar stereographic grid.

(b) Winds:

The wind directions are defined in advance for interpolation.

```
-new_grid_winds grid(or earth)
```

where

Option	Description
grid	u-wind goes from grid (i, j) to $(i+1, j)$
earth	u- and v-winds go eastward and northward, respectively

Table 8.7: Option for wind directions.

(c) Interpolation:

The default interpolation is bilinear, but it can be set by '-new_grid_interpolation'.

```
-new_grid_interpolation neighbor
```

3. Output files

Output file	Transferred name
BGDAWP{fhr}.{tmmark}	{RUN}.tz.{domain}.natprs.f{fhr}.{tmmark}.grib2
BGRD3D{fhr}.{tmmark}	{RUN}.tz.{domain}.natlev.f{fhr}.{tmmark}.grib2

Table 8.8: Output files of UPP.

Chapter 9

UFS Weather Model

9.1 Structure of the UFS Weather Model

The Unified Forecast System (UFS) Weather Model is a prognostic model that is used for Short- and Medium-range research and operational forecasts by the National Oceanic and Atmospheric Administration (NOAA). Figure 9.1 illustrates the hierarchical repository structure of the UFS weather model, and the authoritative repositories of the major components are shown in Table 9.1. The Flexible Modeling System (FMS) is a software infrastructure used for functions such as parallelization. The Common-Community Physics Package (CCPP) is a library of physical parameterizations and the framework to use it with the model. The NOAA Environmental Modeling System (NEMS) model driver is used to create the main program.

Component	Authoritative repository
UFS weather model	https://github.com/ufs-community/ufs-weather-model
FMS	https://github.com/NOAA-GFDL/FMS
FV3 (fv3atm)	https://github.com/NOAA-EMC/fv3atm
atmos_cubed_sphere	https://github.com/NOAA-EMC/GFDL_atmos_cubed_sphere
ccpp-framework	https://github.com/NCAR/ccpp-framework
ccpp-physics	https://github.com/NCAR/ccpp-physics
NEMS	https://github.com/NOAA-EMC/NEMS
stochastic_physics	https://github.com/noaa-psd/stochastic_physics

Table 9.1: Authoritative repositories of the components of the UFS weather model

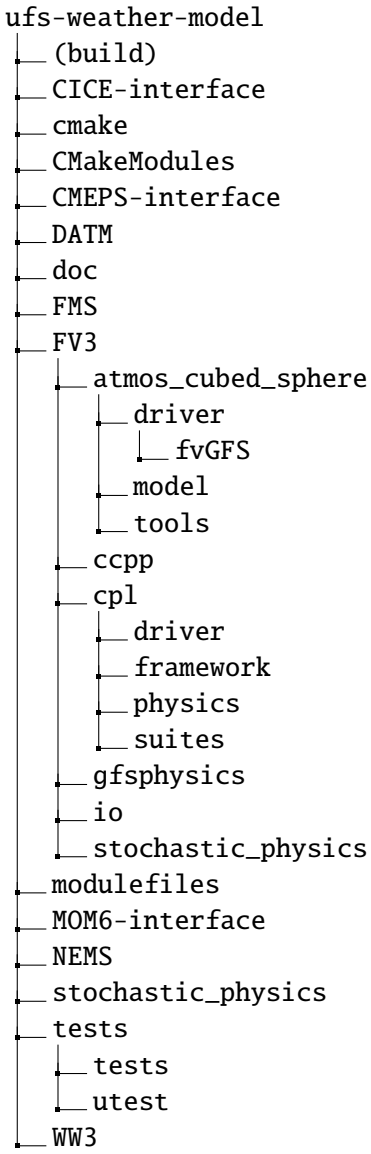

Figure 9.1: Structure of the UFS weather model

Since FV3-LAM is based on the stand-alone FV3 and is not coupled with other components such as MOM6, CICE, and WW3, this section focuses on the FV3 dycore and CCPP suites.

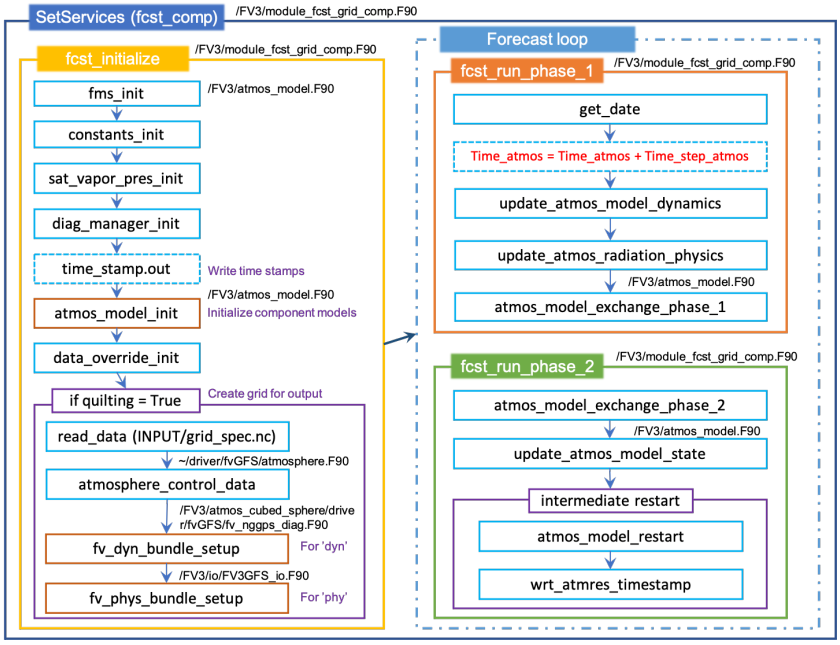

Figure 9.2: Structure of the UFS weather model: SetService (fcst_comp)

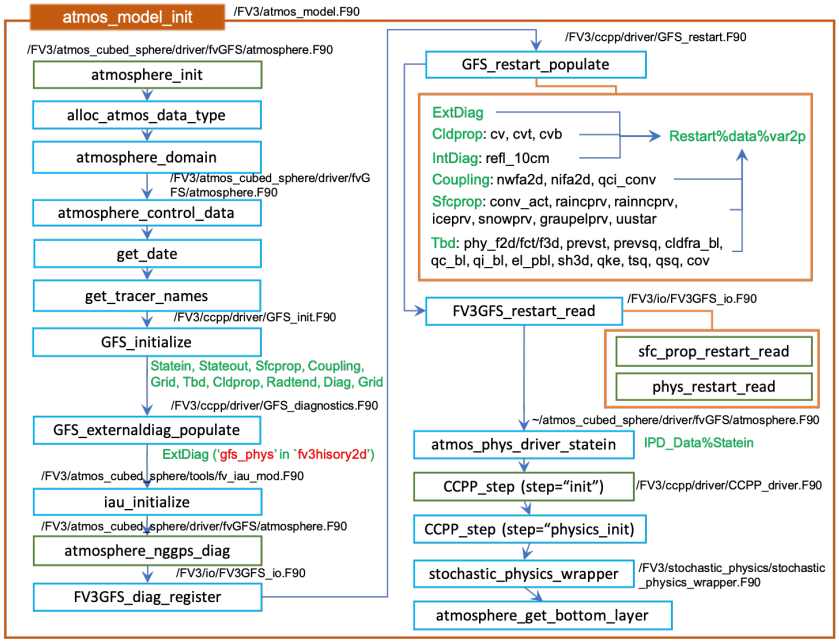

Figure 9.3: Structure of the UFS weather model: atmos_model_init

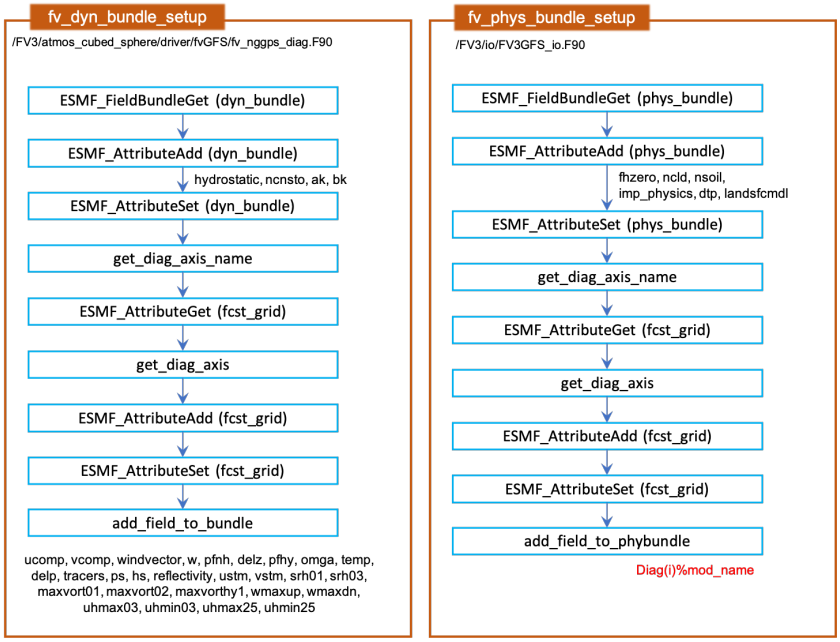

Figure 9.4: Structure of the UFS weather model: dyn/phys_bundle_setup

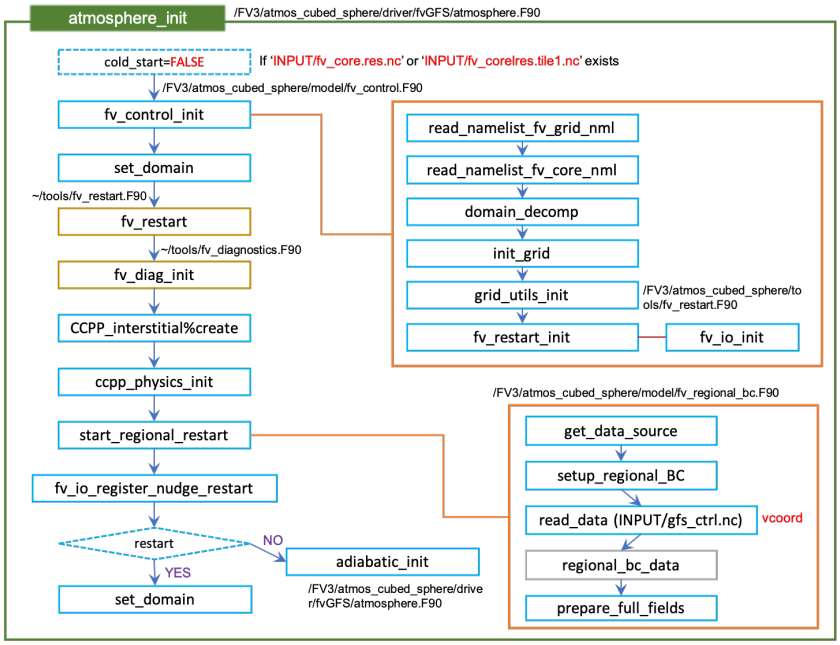

Figure 9.5: Structure of the UFS weather model: atmosphere_init

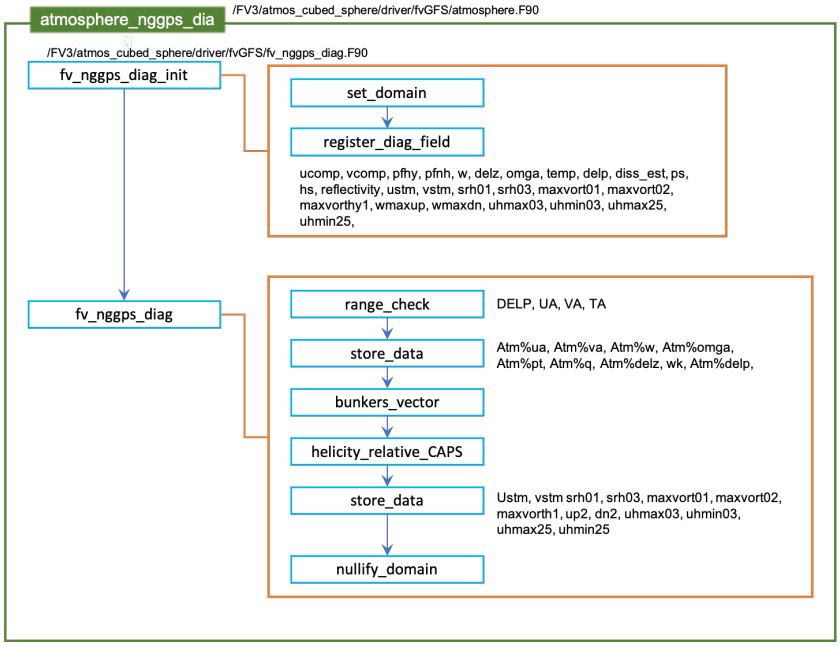

Figure 9.6: Structure of the UFS weather model: atmosphere_nggps_diag

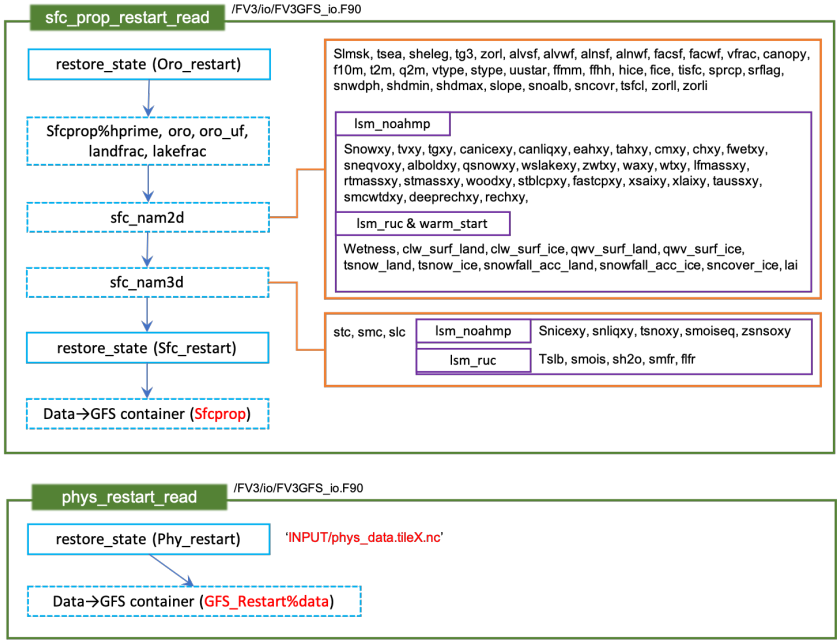

Figure 9.7: Structure of the UFS weather model: restart_read

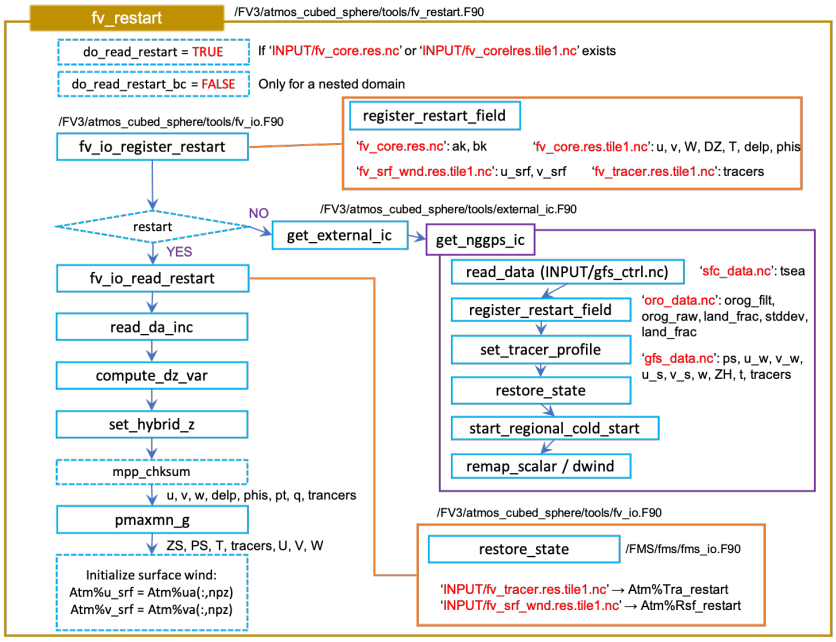

Figure 9.8: Structure of the UFS weather model: fv_restart

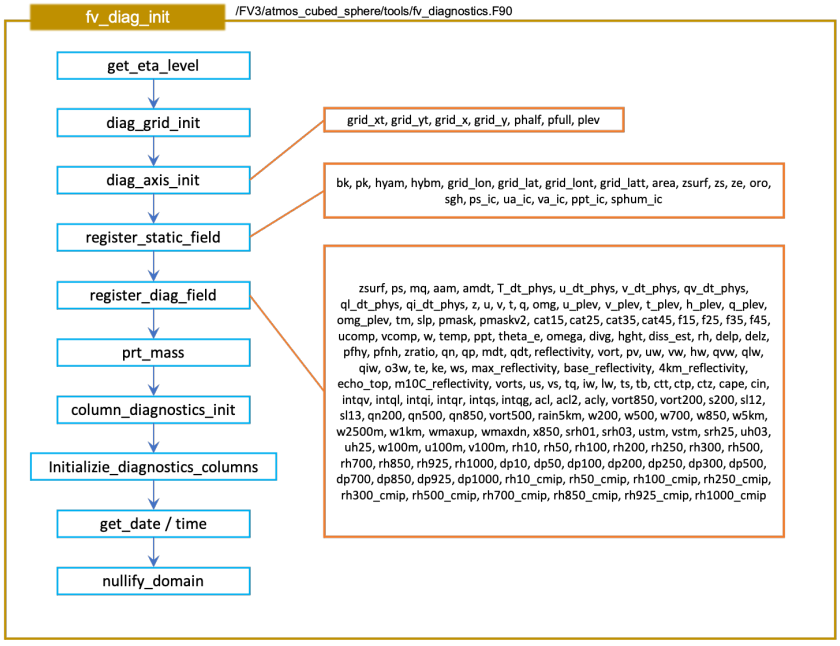

Figure 9.9: Structure of the UFS weather model: fv_diag_init

9.2 Compiling the UFS Weather Model

9.2.1 Building in the {UFS_HOME} Directoy

Reference) http://ufs-weather-model.readthedocs.io/ (Chapter 3.3)

Clone the 'develop' branch of the UFS weather model:

```
git clone --recursive https://github.com/ufs-community/ufs-weather-model
```

The cloned directory ('~/ufs-weather-model') is referred to as '{UFS_HOME}'.

Note) You can follow the steps described in Sections 10.1.10 and 10.1.5 for submitting a PR in the future.

Load the required modules and set the environment variables:

```
cd ~/ufs-weather-model
vim COMPILE.sh

#!/bin/bash
set -xue
export SRC="/scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/ufs-weather-model"
export CCPP_SUITES="FV3_GFS_v15_thompson_mynn"
module use ${SRC}/modulefiles/hera.intel
module load fv3
cd ${SRC}
sh build.sh
```

Check the configuration:

```
vim CMakeLists.txt (lines 21-35)
```

where the default values are

Flag name	Description	Default
32BIT	Single precision arithmetic in dycore	OFF
AVX2	AVX2 instruction set	ON
SIMDMULTIARCH	Multi-target SIMD instruction sets	OFF
CCPP	CCPP	ON
DEBUG	DEBUG mode	OFF

DEBUG_LINKMPI	Linkmpi option when DEBUG mode is on	ON
INLINE_POST	Inline post	OFF
MULTI_GASES	MULTI_GASES	OFF
OPENMP	OpenMP threading	ON
PARALLEL_NETCDF	Parallel NetCDF	OFF
QUAD_PRECISION	Quad precision for certain grid metric terms in dycore	ON
REPRO	REPRO mode	OFF
WW3	WW3	OFF
S2S	S2S	OFF
DATM	Data atmosphere	OFF

Table 9.2: Configuration flags and their default value

Run the script:

```
sh COMPILE.sh
```

Check if the executable 'ufs_model' is created in the 'build' directory:

```
cd ~/ufs-weather-model/build/
```

9.2.2 Building in the '{UFS_HOME}/tests' Directory

1. Using *CMAKE* (Release branch)

   Compile it with *CMAKE*:

```
cd ~/ufs-weather-model/tests/
(rm -rf build_fv3_1)
sh compile_cmake.sh ${PATHTR} ${BUILD_TARGET} ${MAKE_OPT} ${BUILD_NAME} ${
    clean_before} ${clean_after} >& log.txt
```

   where

Name	Description
PATHTR	Path to the source code
BUILD_TARGET	Machine specific name
MAKE_OPT	Compile option for 32 bit ("32BIT=Y")

BUILD_NAME	Sub-name of the executable ('fv3_(?).exe')
clean_before	Flag for cleaning the entire code before compiling (YES or NO)
clean_after	Flag for cleaning the entire code after the compilation is done (YES or NO)

Table 9.3: Variables for compiling.

- On Hera:

```
cd ~/ufs-weather-model/tests/
sh compile_cmake.sh {HOME}/ufs-weather-model hera.intel "32BIT=Y␣STATIC=
    Y␣CCPP=Y␣SUITES=FV3_GFS_v15_thompson_mynn" 32bit YES NO >& log.txt
# check if the executable 'fv3_32bit.exe' exists.
```

Notes) Make sure that the CCPP flag ("CCPP=Y") should be placed ahead of the SUITE flag.

2. Using 'compile.sh' (Develop branch)

   Load the required modules and set the environment variables:

```
cd ~/ufs-weather-model/tests
module purge
sh compile.sh hera.intel "32BIT=Y␣SUITES=FV3_GFS_v15_thompson_mynn" 32bit
    YES NO >& log.txt
# check if the executable 'fv3_32bit.exe' exists.
```

9.2.3 Building the DEBUG mode

For the purpose of debugging, the UFS weather model can be compiled with the debug mode. You need to put the flag for debugging ("DEBUG=Y") into the list of compile options:

```
cd ~/ufs-weather-model/tests
module purge
sh compile.sh hera.intel "DEBUG=Y␣32BIT=Y␣SUITES=FV3_GFS_v15_thompson_mynn" 32bit
     YES NO >& log.txt
```

9.3 Regression Test

9.3.1 Running Regression Test

Reference) Link to UFS-weather-model regression-test wiki

Check the HPC account ('ACCNR') that you can use for the specific HPC machine:

```
cd ~/ufs-weather-model/tests/
vim rt.sh
# For example, ACCNR=fv3-cam for hera (line 214)
```

Run the script with necessary flags:

```
cd ~/ufs-weather-model/tests/
./rt.sh -e (or -r) &> log_file
```

where

Flag	Description
-c	Create new baseline results
-e	Use *ecFlow* workflow manager
-h	Display this help
-k	Keep run directory
-l	Runs test specified in <file> (-l <file>)
-m	Compare against new baseline results
-n	Run single test <name> (-n <name>)
-r	Use *Rocoto* workflow manager

Table 9.4: Flags for the regression test

Notes)

- You do NOT need to compile the UFS weather model separately.
- If you run a test with the flag '-k', you will find the run directories in the machine-specific path ('PTMP') specified in 'ufs-weather-model/tests/ rt.sh'.

9.3.2 Building a Specific Test Case from Regression Test

Reference) Link to UFS-weather-model FAQ

Specific test cases of the UFS Weather Model can be built by the regression test ('rt.sh'). The advantages of this approach are as follows:

- A workflow, pre- and post-processing steps are not necessary.
- The input data required for any specific machines are already set up.
- Working copies that be used for code-development are generated in the run directory.
- A batch submission script (named 'job_card') is generated in the working copy.

The steps are as follows:

1. Create a custom configuration file from the built-in configuration file:

```
cd ~/ufs-weather-model/tests/
cp rt.conf my_rt.conf
vim my_rt.conf
```

2. Edit the configuration file for a specific test as needed. Two lines are essential for any custom cases: 'COMPILE' and 'RUN'.

   (a) COMPILE:

   COMPILE | (configuration flag described **in** Table 9.2) | (machine & compiler) | (−) | (−)

   (b) RUN:

   RUN | (parameter file **in** 'ufs_weather_model/tests/tests) | (−) | (−) | (−)

   Below is an example for a regional test case with four specific run directories using the 'FV3_GFS_15_thompson_mynn' CCPP suite and the flag '32BIT=Y':

```
COMPILE | SUITES=FV3_GFS_v15_thompson_mynn 32BIT=Y  |      | fv3 |
RUN     | fv3_ccpp_regional_control                  |      | fv3 |
RUN     | fv3_ccpp_regional_restart                  |      | fv3 | fv3_ccpp_regional_control
RUN     | fv3_ccpp_regional_quilt                    |      | fv3 |
RUN     | fv3_ccpp_regional_quilt_netcdf_parallel    |      | fv3 |
```

3. Modify the machine-specific parameters in the 'rt.sh' script:

```
vim rt.sh

# specify the following parameters:
  ACCNR=(account name)
  dprefix=(output directory)
```

4. Run the 'rt.sh' script:

```
./rt.sh -k -l my_rt.conf >& my_rt.out &
```

5. Modify the input files as needed, and re-run the test case by submitting the 'job_card' file:

```
cd {expt_dir}
sbatch job_card
(or bsub job_card)
```

Note) You can find another example for a restart run in Section 9.6.1.

9.4 Unit Test

Reference) Link to the UFS-weather-model unit-test wiki

The unit test is a suite of tests and its purpose is to satisfy the operational implementation requirement. Compared to the regression test, the unit test focuses on the specific CCPP suite and inspects various reproducibility aspects such as threads, MPI processes, domain decomposition, and restart. Developers ensure their implementations pass the unit test before requesting a Pull Request (PR).

Note) The unit test for FV3-LAM was **NOT** developed yet as of 11/10/20.

9.4.1 Components of the Unit Test

1. Thread reproducibility (THR):
   - Variable: Number of threads
   - Number of threads=2, tasks per node=2

2. MPI process reproducibility (MPI):
   - Variable: Number of MPI tasks
   - JNPES=2, WRITE_GROUP=2, WRTTASK_PER_GROUP=12
3. Domain decomposition reproducibility (DCP):
   - Variable: Tile layout of FV3
   - Swap 'INPES' and 'JNPES'
4. Restart reproducibility (RST):
   - Use restart files in the middle of the run (SHOUR+FHMAX/2) and compare results at the end of the run (SHOUR+FHMAX).
5. 64/32 (BIT)
6. Debug (DBG)

9.4.2 Running Unit Test

1. Pick a test case under '~/ufs-weather-model/tests/tests' that is relevant to the code change.
2. Modify the test case or add a new test case.
3. Run the test case.

```
cd ~/ufs-weather-model/tests
./utest -n <test-name>
```

where

Flag	Description
-n	Run all unit tests for <test-name> (-n <test-name>)
-c	Runs a specific unit test case (-c <test-case>)
-h	Display usage of shell script
-k	Keep the run directory
-e	Use *ecFlow* workflow manager
-x	Skip compilation

-z	Skip running (compilation only)
-b	Test bit reproducibility
-d	Test debug reproducibility

Table 9.5: Flags for the unit test

For examples, in case of <test-name>='fv3_ccpp_control':

(a) To run all tests:

```
./utest -n fv3_ccpp_control
```

(b) To run THR test only:

```
./utest -n fv3_ccpp_control -c thr
```

(c) To run MPI and DCP tests only:

```
./utest -n fv3_ccpp_control -c mpi,dcp
```

4. Check and commit the unit test report/logfile to the developer's feature branch.

9.4.3 Input and Output

1. Input

   - 'utest.bld'

   ```
   cd ~/ufs-weather-model/tests
   vim utest.bld
   ```

     - Specify the required build options in the format of '<test-name> | <build_opt>'.
     - <build_opt> can be 'WW3=Y', 'CCPP=Y', etc.
     - <test-name>, <INPUT_NML>, <FV3_RUN>: all similar to those in the regression test

2. Output

   - File location: '~/ufs-weather-model/tests/'

- Unit test logfile: 'UnitTests_[machine]_[compiler].log' (commit)
- Compile logfile: 'Compile_ut_[machine]_[compiler].log'
- Directory: 'log_ut_[machine].[compiler]

Note) Do not commit the 'Compile_ut_[machine]_[compiler].log' and 'log_ut_[machine].[compiler] files.

9.5 Initial and Lateral Boundary Fields

9.5.1 Source Codes

Almost all modification for regional IC/LBC can be found at

```
cd ~/ufs-weather-model/FV3/atmos_cubed_sphere/model/
vim fv_regional_bc.F90
```

Sometimes it happens at

```
cd ~/ufs-weather-model/FV3/atmos_cubed_sphere/tools/
vim external_ic.F90
```

9.5.2 Regional Boundary Update

Figure 9.10 illustrates the halo and blending zones of the longitudinal (u) velocity field updated by the 'regional_boundary_update' call. In general, the scalar variables such as 'delp' and 'pt', and the vertical velocity 'w' have three halo rows in both i and j directions along the boundary. However, the velocity fields such as 'u', 'v', 'ut', and 'vt' have one more layer. The variables of 'u' and 'vt' have four halo rows in the j direction while the variables of 'v' and 'ut' have four halo rows in the i direction.

9.6 Restart Option: Warm Start

9.6.1 Regression Test for a Regional Restart

Clone the latest 'develop' branch of the UFS weather model:

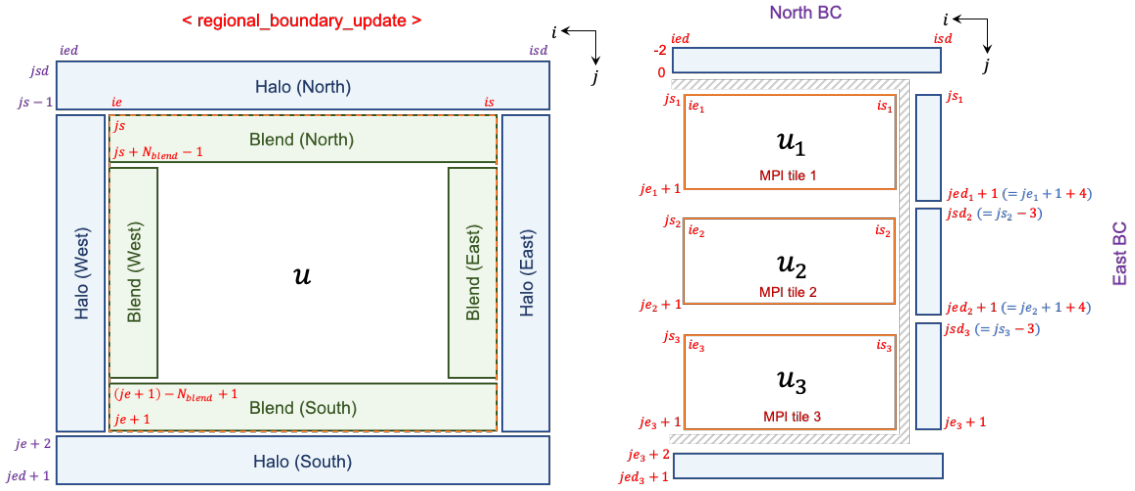

Figure 9.10: Regional boundary update: u field

```
git clone https://github.com/ufs-community/ufs-weather-model
cd ufs-weather-model
git submodule update --init --recursive
```

Check the HPC account ('ACCNR') that you can use for the specific HPC machine:

```
cd ~/ufs-weather-model/tests/
vim rt.sh
# For example, ACCNR=fv3-cam for hera (line 214)
```

Open the configuration file of the regression test:

```
vim rt.conf
```

Edit only for two tests of control and restart as follows (remove all the rest):

```
COMPILE | SUITES=FV3_GFS_v15_thompson_mynn 32BIT=Y   |   | fv3 |
RUN     | fv3_ccpp_regional_control                  |   | fv3 |
RUN     | fv3_ccpp_regional_restart                  |   | fv3 | fv3_ccpp_regional_control
```

Run the regression test

```
./rt.sh -ek
```

The three directories should be created as below. All output NetCDF files in control (2) and restart (3) should be bitwise identical.

```
cd /scratch1/NCEPDEV/stmp2/Chan-hoo.Jeon/FV3_RT/rt_#     (on Hera)
```

1. compile_1
2. fv3_ccpp_regional_control_prod
3. fv3_ccpp_regional_restart_prod

Note) You can check the machine-specific path ('PTMP') in 'ufs-weather-model/ tests/rt.sh'.

9.6.2 Checklist for a Restart Run

1. The parameters listed in Table 9.6 should be modified:

Parameter	Cold start	Warm start (restart)
external_ic	.true.	.false
mountain	.false.	.true.
na_init	1	0
nggps_ic	.true.	.false.
warm_start	.false.	.true.

Table 9.6: Parameters to be modified for a restart run

2. Copy the input files for restart, which are presented in the 'RESTART' directory, into the 'INPUT' directory.

```
cp {base_dir}/RESTART/* {EXPTDIR}/INPUT/
```

The list of the additional input files is as follows:

File name	Description / variables
coupler.res	Model start time and current model time
fv_core.res.nc	ak, bk
fv_core.res.tile1.nc	u, v, W, DZ, T, delp, phis
fv_srf_wnd.res.tile1.nc	u_srf, v_srf

fv_tracer.res.tile1.nc	sphum, liq_wat, rainwat, ice_wat, snowwat, graupel, o3mr, sgs_tke, cld_amt
phy_data.nc	cv, cvt, cvb, phy_f2d_01, totprcp_ave, cnvprcp_ave, phy_f3d_01, phy_f3d_02
sfc_data.nc	Surface climatology data

Table 9.7: Additional input files for a restart run

Note) Variables in the initial condition file 'gfs_data.nc' which is the input file of the base case: ps, w, zh, t, delp, sphum, liq_wat, o3mr, ice_wat, rainwat, snowwat, graupel, u_w, v_w, u_s, v_s

9.7 Near-boundary Blending

The blending works by not only generating data for the boundary region but by inserting additional 'boundary rows' into the BC files that overlap the outermost rows of the integration domain. The *chgres* code in 'UFS_UTILS' can add those extra rows in the boundary files. When running the forecast set the 'input.nml' variable 'nrows_blend' to the number of rows you want to blend within the outer rows of the integration domain.

9.7.1 Blending in a Forecast Run

1. Check that *FV3*, which is installed in your system, supports the 'blending' option:

```
cd ~/ufs-weather-model/FV3/atmos_cubed_sphere/model/
grep -n -i blend fv_control.F90 fv_arrays.F90
```

2. Create IC/LBC fields including blending layers (Section 7).

   - Run 'chgres_cube' with 'halo_blend':

```
regional = 1 (or 2)
halo_bndy =4
halo_blend = 10
```

3. Turn on the blending option in the configuration 'input.nml' (Section **??**).

```
(under '&fv_core_nml')
    regional_bcs_from_gsi = .false.   (for DA blending)
    write_restart_with_bcs = .false.   (for DA blending)
    nrows_blend = 10
```

Note) Make sure that 'halo_blend' is equal to 'nrows_blend'.

9.7.2 Blending in a DA Run

1. Two variables in 'input.nml' need to be used: (1) write_restart_with_bcs (.true. / .false.), and (2) regional_bcs_from_gsi (.true. / .false.).

   - 'write_restart_with_bcs':

     Typically the model writes out restart files that contain only the integration domain in which case set this flag 'false'. However, in order to include the boundaries in the assimilation, those boundary rows should be included in the restart files so set the first flag to 'true'. This will make restart files write the boundary rows in addition to writing out normal core and tracers.

   - 'regional_bcs_from_gsi':

     The second flag says to treat all the BC files in the usual way when it is 'false'. The 'usual' means that the variables are remapped in the vertical from the external forecast's layer distribution to the forecast's, and the winds are rotated from geographic lat/lon to the forecast grids' orientation. When the assimilation writes out restart files, though the fields put into the new post-GSI, BC files are already on the proper levels and are oriented properly so the model is told to **NOT** remap or rotate by setting this flag to 'true'.

2. To create and write the new larger restart files, which include the blending boundary layers, for the GSI, the code needs original restart files to get all the specifications and field names in the '/INPUT' subdirectory for NetCDF:

   (a) Copy any normal-sized 'fv_core.res.tile1.nc' and 'fv_tracer.res.tile1.nc' into the '/INPUT' subdirectory before starting the forecast, basically as templates. Make sure that all three dimensions in those template restart files are the values you want to use.

   (b) Change their names from 'tile1' to 'temp'. Otherwise, FMS will get confused.

(c) The forecast's normal and enlarged restart files will show up in the '/RESTART' subdirectory of the working directory as usual. The enlarged restart file names will end with '.tile1_new.nc'.

3. After your run writes out the two enlarged restart files for the data assimilation, you also need 'sfc_data.nc' and 'grid_spec.nc' files that have been similarly enlarged. To get them, run the following serial code:

```
prep_for_regional_DA.F90
```

- Input files:

Original name	Transferred name	Description
sfc_data.tile7.nc	sfc_data_orig.nc	Output of IC
grid_spec.nc	grid_spec_orig.nc	Output of a forecast run
grid.tile7.halo3.nc	grid.tile7.halo3.nc	Output of 'grid_driver' (pre-processing)

Table 9.8: Input files for enlarged restart files

- Output files:

Original name	Transferred name
sfc_data_new.nc	sfc_data.nc
grid_spec_new.nc	grid_spec.nc

Table 9.9: Output files for enlarged restart files

Note) This job assumes that the original smaller files have been renamed 'sfc_data_orig.nc' and 'grid_spec_orig.nc' and it then generates the larger files called 'sfc_data_new.nc' and 'grid_spec_new.nc'.

4. The data assimilation runs on the enlarged core and tracers restart files and the enlarged 'sfc_data.nc' and 'grid_spec.nc' files. When it is finished, the updated data must be put back into the files the model will read in the forecast.

```
move_DA_update_data.F90
```

- Input files:

File name	Description
fv_core.res.tile1_new.nc	Enlarged core restart file
fv_tracer.res.tile1_new.nc	Enlarged tracers restart file
gfs_bndy.tile7.HHH.nc	Original BC file valid at the time (HHH) of the restart (no DA changes)

Table 9.10: Input files for updated BC files

- Run:

  Currently the move program takes 'HHH' as a command line argument. i.e. if the BC file is 'gfs_bndy.tile7.000.nc' then to execute the program:

```
./a.out 000
```

  When the code executes it will copy all the interior data from the updated enlarged restart files back into the original-sized restart files that were written prior to running the data assimilation (both normal and enlarged restart files are written when 'write_restart_with_bcs = .true.'). And a new BC file is generated called 'gfs_bndy.tile7.HHH_gsi.nc' that is taken from the boundary region of the updated enlarged restart files and it has a different structure from the original BC files.

- Output files:

File name	Description
gfs_bndy.tile7.HHH_gsi.nc	

Table 9.11: Output files for updated BC files

5. When you start up the next free forecast following the DA update set:

```
regional_bcs_from_gsi = .true.
```

So the model will use 'gfs_bndy.tile7.HHH_gsi.nc' as the starting BC file (with no vertical remap or wind rotation) but then will read the following BC file valid at the end of your boundary update interval with its original name (no '_gsi' in it) and will do the remap and wind rotation as usual.

9.7.3 Modification of the source code

The code can be cloned from GitHUB:

```
git clone https://github.com/TomBlack-NOAA/GFDL_atmos_cubed_sphere.git -b tracers
```

The files under 'atmos_cubed_sphere' that are changed for the blending and for including the regional boundary in the data assimilation are shown in Table 9.11. Here are two versions of two files needed to switch between the NAM's 60-layer vertical structure and the 65 layers.

1. For 65 layers:

```
cp external_ic.F90_65lyrs external_ic.F90
cp fv_eta.F90_65lyrs fv_eta.F90
```

2. For NAM's 60 layers:

```
cp external_ic.F90_NAM_lyrs external_ic.F90
cp fv_eta.F90_NAM_lyrs fv_eta.F90
```

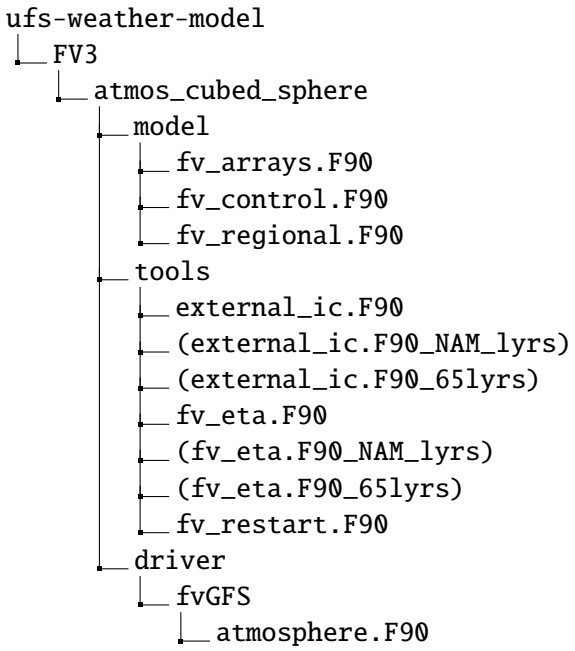

Figure 9.11: Structure of directory

Chapter 10

Code Management on GitHub

10.1 Modification of the UFS Weather Model on GitHub

10.1.1 Overall Steps of Modification and Pull Request

UFS applications have hierarchical structure managed through git submodules. If you are making changes in UFS sub-components, you should follow these steps:

1. Check the hierarchical structure of the UFS weather model.
2. Create your own forks from the repositories that you are going to make changes to.
3. Configure Git remotes for a sub-component.
4. Update the branch in the local and fork repositories.
5. Create a 'feature/fix_[xxx]' branch in the cloned sub-component.
6. Make changes to the feature branch.
7. Commit (save) the changes to the local feature branch.
8. Push the local feature branch to your remote 'origin' branch (sub-component).
9. Configure Git remotes for the UFS weather model.
10. Create a feature branch in the cloned 'ufs-weather-model' repository.
11. Make and commit changes if necessary.

12. Update the sub-component in the '.gitmodules' file and sync the submodules.
13. Push the local feature branch (ufs-weather-model)
14. Create pull requests (PR).
15. Update and modify the branch based on reviewers' comments.
16. Clean up the branch once your pull request is merged.

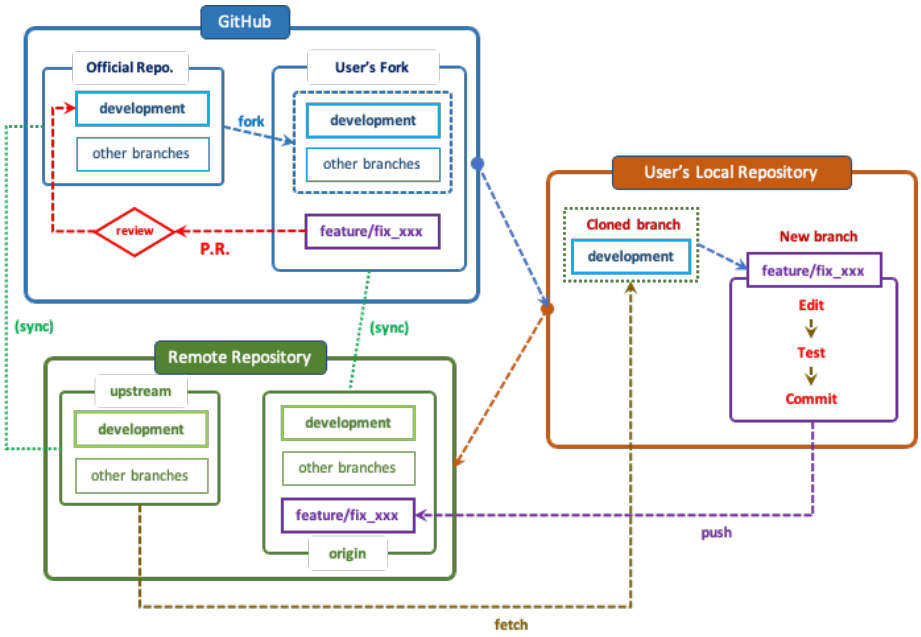

Figure 10.1: Flowchart for Pull Request (PR)

10.1.2 Structure of the UFS Weather Model and Its Sub-components

Sub-component	Repository	Branch
ufs-weather-model	ufs-community/ufs-weather-model	develop
CMakeModules	NOAA-EMC/CMakeModules	develop
DATM	NOAA-EMC/NEMSdatm	develop
FMS	NOAA-GFDL/FMS	master

FV3	NOAA-EMC/fv3atm	develop
atmos_cubed_sphere	NOAA-EMC/GFDL_atmos_cubed_sphere	dev/emc
CCPP framework	NCAR/ccpp-framework	master
CCPP physics	NCAR/ccpp-physics	master
NEMS	NOAA-EMC/NEMS	develop
WW3	NOAA-EMC/WW3	develop
stochastic_physics	noaa-psd/stochastic_physics	master

Table 10.1: UFS weather model: GitHub repository and branch

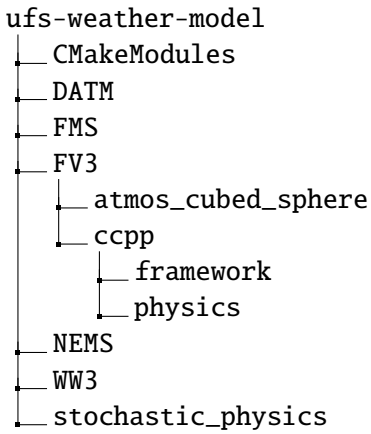

Figure 10.2: UFS weather model and its sub-components

10.1.3 Forking the UFS Weather Model and Sub-components

If you create forks in sub-component, make sure that you should create forks for all the repositories in the path from the sub-component up to the UFS weather model. For example, if you need to make changes in 'FV3', you should create forks from the EMC's 'fv3atm' repository (FV3) as well as the ufs-community's 'ufs-weather-model' repository.

In a web browser, go to the GitHub page shown in Table 10.1.

```
https://github.com/ufs-community/ufs-weather-model
https://github.com/NOAA-EMC/fv3atm
```

Press the 'Fork' button on the top right. You can see the repositories in your GitHub account. The URL of your fork will be

```
https://github.com/{GitHubID}/ufs-weather-model
https://github.com/{GitHubID}/fv3atm
```

10.1.4 Cloning the UFS Weather Model

Clone the UFS weather model to the '{HOMEdir}' directory (local repository) and check out the 'develop' branch:

```
cd {HOMEdir}
git clone --recursive https://github.com/ufs-community/ufs-weather-model
```

where '{HOMEdir}' is the parent directory where the 'develop' branch will be cloned.

Note) You do not need to clone the sub-components separately.

10.1.5 Configuring Git Remotes for the Cloned Sub-component

Move to the bottom level of the sub-components for your changes (For example, FV3):

```
cd {HOMEdir}/ufs-weather-model/FV3
```

When you clone your fork, git will create a remote (i.e. a pointer to a repository location) named 'origin' that points to the authoritative repository. To check the list of remotes that are defined in your clone, issue the command:

```
git remote -v
```

- Even though you cloned the UFS weather model, its sub-component 'FV3' is linked to 'NOAA-EMC/fv3atm'. Therefore, the output should be as follows:

  ```
  origin https://github.com/NOAA-EMC/fv3atm (fetch)
  origin https://github.com/NOAA-EMC/fv3atm (push)
  ```

Rename the 'origin' remote repository to the 'upstream' remote repository:

```
git remote rename origin upstream
```

Create a new 'origin' remote repository that points to your fork 'GitHubID/fv3atm' repository:

```
git remote add origin https://github.com/{GitHubID}/fv3atm
git remote update
git remote -v
```

- The output should be as follows:

```
origin   https://github.com/{GitHubID}/fv3atm (fetch)
origin   https://github.com/{GitHubID}/fv3atm (push)
upstream   https://github.com/NOAA-EMC/fv3atm (fetch)
upstream   https://github.com/NOAA-EMC/fv3atm (push)
```

10.1.6 Updating the Branch in the Cloned Sub-component

You should update the 'develop' branch in your clone (local repository) as well as in your fork in order to have the latest changes in the 'develop' branch of the authoritative 'NOAA-EMC/fv3atm' repository.

- When you create new feature branches to address new issues, you start off with the latest version of 'develop' in the authoritative repository in those branches.
- Updating the 'develop' branch in your clone entails merging the latest version of the 'develop' branch of the 'NOAA-EMC/fv3atm' into the 'develop' branch of your fork.

Fetch the contents of the 'upstream' repository (i.e. the 'NOAA-EMC/fv3atm' repository) to your local disk:

```
cd {HOMEdir}/ufs-weather-model/FV3
git remote -v (You should see upstream remote on the list)
```

This will fetch all the branches from the 'NOAA-EMC/fv3atm' repository and put them on your local disk. However it will not automatically check any of them out.

- You can see which branch you are currently on (branch with an asterisk):

```
git branch -vva
```

  There should be a list of remote branches which include the one you want to merge into your local 'develop' branch.

Merge the 'remote/upstream/develop' branch into your local 'develop' branch:

```
git checkout develop
git pull upstream develop (or 'git fetch upstream' then 'git merge upstream/
    develop')
```

Your local 'develop' branch will now be an exact copy of the latest version of the 'develop' branch of the 'NOAA-EMC/regional_workflow' repository.

- It is recommended not to make your own changes to your local 'develop' branch.
- Any changes you make to address issues should be made in a separate feature branch.
- Keeping your local 'develop' branch unchanged (except for occasionally updating it with the latest from the authoritative branch) allows you
  1. to merge the latest changes from other developers into your feature branch.
  2. to quickly create a new and up-to-date feature branch whenever you need to address a new issue.

Update the 'develop' branch in your fork. You can do this by pushing your updated local 'develop' branch to your fork.

```
git push origin develop
(GitHub Username)
(GitHub Password)
```

10.1.7 Creating a New Feature Branch of the Sub-component

- It is suggested that users create their own feature branch in their fork.
- Code changes will be committed (pushed) to users' feature branch in their own fork testing.
- Users' develop/feature branch in their fork should be synced with authoritative (official) repository periodically.
- When development work is done, users will sync their feature branch with the latest develop (or master) branch in the authoritative repository, run the regression test and make a pull request (PR) to the authoritative repository.

- It is suggested to delete the feature branch in users' personal fork when the code changes are merged into the authoritative repository.

Make sure that you are on the local 'develop' branch:

```
cd {HOMEdir}/ufs-weather-model/FV3
git checkout upstream/develop
```

Note) The name of the main develop branch may differ in repositories.

Create a new feature branch named 'fix_[xxx]':

```
git checkout -b feature/fix_[xxx]
```

You will now be on this new branch with the command:

```
git branch -vva
```

At this point, the 'feature/fix_[xxx]' is only a local branch, i.e. it only exists in your clone. It does not exist in your fork (nor in the authoritative repository).

Push the local 'feature/fix_[xxx]' branch to your fork:

```
git push origin feature/fix_[xxx]
(GitHub Username)
(GitHub Password)
```

Now you can see the new branch 'feature/fix_something' in your fork in the 'branches' tab on 'https://github.com/{GitHubID}/fv3atm'.

10.1.8 Making Changes to the Feature Branch of the Sub-component

Make sure that your are on the 'feature/fix_something' branch:

```
cd {HOMEdir}/ufs-weather-model/FV3
git checkout feature/fix_[xxx]
```

Merge your local 'develop' branch into your local 'feature/fix_[xxx]' branch:

```
git merge develop
```

This command will report merge conflicts, in which case you will have to resolve them manually. If you have set up a visual mergetool in your git configuration file, you can use the 'git mergetool' command to visually resolve the conflict.

Edit/add/delete files as necessary in your local 'feature/fix_[xxx]' branch in order to address issues.

10.1.9 Committing and Pushing Your Changes

Commit (save) the changes you made to your local 'feature/fix_something' branch.

```
git add file1 file2 file3 ...
# For example, if '{sub_dir}/file1' is modified, type 'git add {sub_dir}/file1.
git commit
```

This will open up a text editor window in which you can enter a commit message. The commit will occur once you close the editor window.

Push your local 'feature/fix_[xxx]' branch to the remote 'origin' repository:

```
git push origin feature/fix_[xxx]
(GitHub Username)
(GitHub Password)
```

Your fork will now have a version of the 'feature/fix_[xxx]' branch with all of your commits.

10.1.10 Configuring the UFS Weather Model (Parent Repo.)

Check the list of remotes that are defined in your 'ufs-weather-model' repository:

```
cd {HOMEdir}/ufs-weather-model
git remote -v
```

- The output should be as follows:

```
origin https://github.com/ufs-community/ufs-weather-model (fetch)
origin https://github.com/ufs-community/ufs-weather-model (push)
```

Rename the 'origin' remote repository to the 'upstream' remote repository:

```
git remote rename origin upstream
```

Create a new 'origin' remote repository that points to your fork 'GitHubID/ufs-weather-model' repository:

```
git remote add origin https://github.com/{GitHubID}/ufs-weather-model
git remote update
git remote -v
```

- The output should be as follows:

```
origin   https://github.com/{GitHubID}/ufs-weather-model (fetch)
origin   https://github.com/{GitHubID}/ufs-weather-model (push)
upstream   https://github.com/ufs-community/ufs-weather-model (fetch)
upstream   https://github.com/ufs-community/ufs-weather-model (push)
```

10.1.11 Updating the Branch of the UFS Weather Model

Fetch the 'upstream' repository:

```
cd {HOMEdir}/ufs-weather-model
git remote -v (You should see upstream remote on the list)
```

Merge the 'remote/upstream/develop' branch into your local 'develop' branch:

```
git checkout develop
git pull upstream develop (or 'git fetch upstream' then 'git merge upstream/
    develop')
```

Update the 'develop' branch in your fork

```
git push origin develop
(GitHub Username)
(GitHub Password)
```

10.1.12 Updating the '.gitmodules' File

Since the UFS weather model has hierarchical structure through git submodules, you should update the '.gitmodules' file in the **parent** directory of your changes. For example, if you made changes in 'FV3', you should update the '.gitmodules' file in the 'ufs-weather-model' directory.

```
cd {HOMEdir}/ufs-weather-model   (parent directory of FV3)
vim .gitmodules
```

Change the 'url' and 'branch' under "FV3" to your feature branch in your fork where you made changes in the previous sections:

```
[submodule "FV3"]
    path = FV3
    url = https://github.com/{GitHubID}/fv3atm
    branch = feature/fix_[xxx]
```

Check the status of the repository. You should see '.gitmodules' and 'FV3' modified.

```
git status
```

Sync the sub-components with the updated '.gitmodules' file.

```
git submodule sync
git submodule
```

10.1.13 Create a New Feature Branch of the UFS Weather Model

Create a new feature branch:

```
cd {HOMEdir}/ufs-weather-model
git checkout -b feature/fix_[xxx]
```

Edit/add/delete files as necessary in your local 'feature/fix_[xxx]' branch in order to address issues.

After making your changes, you will likely need to run one or more **tests** (e.g. workflow end-to-end tests, regression tests) to ensure that your modifications do not break the code under various configurations (See Section 9.3 for the regression test).

10.1.14 Committing the '.gitmodules' File and Submodules

Commit (save) the changes you made to your local 'feature/fix_[xxx]' branch.

```
git add .gitmodules FV3
git commit
(write a commit message and close the commit window)
```

This will open up a text editor window in which you can enter a commit message. The commit will occur once you close the editor window.

Note) If you do not write a commit message, your modification will not be committed to your feature branch.

Push your local 'feature/fix_[xxx]' branch to your fork:

```
git push origin feature/fix_[xxx]
(GitHub Username)
(GitHub Password)
```

Your fork will now have a version of the 'feature/fix_[xxx]' branch with all of your commits.

10.1.15 Creating Pull Requests (PR)

Create pull requests to get the changes you made in your 'feature/fix_[xxx]' branch into the 'develop' branch of the authoritative repositories. Make sure that you should **open an issue** corresponding to the PR. When opening an issue in the category of 'Bug', you should provide description as mush detail as possible and list all the steps to reproduce the case.

- FV3:

  1. On your web browser, go to your GitHub fork (https://github.com/{GitHubID}/fv3atm) and switch to the 'feature/fix_[xxx]' branch.
  2. Create a pull request ('compare & pull request' button):

Name	Target / example
Base repository	NOAA-EMC/fv3atm
Base branch	develop
Head repository	{GitHubID}/fv3atm
Head branch	feature/fix_[xxx]

Table 10.2: Set-up for pull request: FV3.

  3. Fill in the message for the PR with information on what the PR is for and what tests you have run, etc.

4. Submit the Pull Request (PR).

- UFS weather model:

1. On your web browser, go to your GitHub fork (https://github.com/{GitHubID}/ufs-weather-model) and switch to the 'feature/fix_[xxx]' branch.
2. Create a pull request ('compare & pull request' button):

Name	Target / example
Base repository	ufs-community/ufs-weather-model
Base branch	develop
Head repository	{GitHubID}/ufs-weather-model
Head branch	feature/fix_[xxx]

Table 10.3: Set-up for pull request: UFS weather model.

3. Fill in the message for the PR with information on what the PR is for and what tests you have run, etc. **Note)** Put the PR number of sub-components as dependencies.
4. Submit the Pull Request (PR).

10.1.16 Modifying the Feature Branch

If you have comments from reviewers and need to modify the feature brach,

Go to the feature branch in your local repository:

```
cd {HOMEdir}/ufs-weather-model/FV3
```

Check the remotes. The 'origin' should be your personal fork.

```
git remote -v
```

Make sure that you are in the feature branch:

```
git branch -a
git checkout feature/fix_[xxx]
```

Update the feature branch with the latest upstream/develop branch:

```
git fetch upstream
git merge upstream/develop
("merge␣to␣the␣latest␣develop␣branch")
```

Edit the source code as required.

Add the files:

```
git add file1 {sub_dir}/file2
```

Commit

```
git commit
("fix␣something")
```

Note) If there is no additional change in the previous steps 'Edit/Add', you don't need to commit this again.

Push to the remote 'origin' repository:

```
git push origin feature/fix_[xxx]
```

Do the same for the UFS weather model:

```
cd {HOMEdir}/ufs-weather-model/
git remote -v
git branch
git checkout feature/fix_[xxx]
git add FV3
git submodule sync
git submodule
git fetch upstream
git merge upstream/develop
(Edit)
git commit
("Update␣submodule␣pointer␣for␣FV3")
git push origin feature/fix_[xxx]
```

Run the Regression test on the Tier 1 HPC machines:

```
cd {HOMEdir}
git clone --recursive -b feature/fix_[xxx] https://github.com/{GitHubID}/ufs-
    weather-model
cd ufs-weather-model/tests
vim rt.sh
(Check Account name and paths)
./rt.sh -fe (or -fr) &> log_file
```

Collect all the log files on the machine used for development using 'scp'.

Add the log file, commit, and push:

```
cd {HOMEdir}/ufs-weather-model
git add tests/RegressionTests_[machine]_[compiler].log
git commit
("Regression␣test␣passed␣on␣Hera,␣Orion,␣and␣Wcoss")
git push origin feature/fix_[xxx]
```

Check the change on GitHub.

10.1.17 Pointing Submodule to the Develop Branch

Once your PR for the submodule, FV3, is merged, you should point the '.gitmodule' file to the develop branch of FV3:

Switch the feature branch of FV3 to the develop branch:

```
cd {HOMEdir}/ufs-weather-model/FV3
git checkout develop
```

Update the develop branch:

```
git pull
```

Update the '.gitmodule' file:

```
cd ..
vim .gitmodule
```

1. Change the 'url' and 'branch' under "FV3" to your feature branch in your fork where you made changes in the previous sections:

```
[submodule "FV3"]
    path = FV3
    url = https://github.com/NOAA-EMC/fv3atm
    branch = develop
```

2. Sync the sub-components with the updated '.gitmodules' file.

```
git submodule sync
git submodule
```

Commit the changes you made to your local 'feature/fix_[xxx]' branch.

```
git add .gitmodules FV3
git commit
(write a commit message and close the commit window)
```

Push your local 'feature/fix_[xxx]' branch to your fork:

```
git push origin feature/fix_[xxx]
(GitHub Username)
(GitHub Password)
```

10.1.18 Deleting the Feature Branch

Once your PR is accepted and merged, to reduce clutter you should delete both your local feature branch and its copy in your fork.

- To delete your local branch:

```
git branch -d feature/fix_[xxx]
```

- To delete the 'feature/fix_[xxx]' branch on your fork, you can use one of two methods:

    1. Delete from within your clone on your local machine.

```
git push origin -d feature/fix_[xxx]
```

2. Delete using your web browser:
   (a) Go to the URL of your fork (https://github.com/ {GitHubID}/fv3atm).
   (b) Click on the '# branches' tab on the head bar (NOT the 'Branch: *Name*').
   (c) Find your feature branch in the 'Your branches' block, and click on the red trash can icon on the same line.

10.1.19 Deleting the Forked Repository

1. Open the repository and navigate to settings (top right):

```
github.com/{GITHUBUSER}/regional_workflow/settings
```

2. Scroll to the bottom of the 'Setting' page.
3. Click on the 'Delete this repository' in Danger Zone on the bottom.
4. Type '{GitHubID}/[fork_name]' to confirm.
5. Click on the red bar on the bottom.
6. Put your GitHub password.

10.2 Modification of Regional Workflow in SRW App on GitHub

10.2.1 Modification of UFS SRW App

1. Fork the UFS SRW App:

   In a web browser, go to the GitHub page:

```
https://github.com/ufs-community/ufs-srweather-app
```

   Press the 'Fork' button on the top right. You can see the repositories in your GitHub account. The URL of your fork will be

```
https://github.com/chan-hoo/ufs-srweather-app
```

2. Clone the UFS SRW App to the '{HOMEdir}' directory (local repository):

```
cd {HOMEdir}
git clone -b master https://github.com/ufs-community/ufs-srweather-app
```

3. Configure Git Remotes for SRW App:

   When you clone your fork, git will create a remote named 'origin' that points to the authoritative repository. Check the list of remotes that are defined in your clone:

```
cd ufs-srweather-app
git remote -v
```

   - The output should be as follows:

```
origin https://github.com/ufs-community/ufs-srweather-app (fetch)
origin https://github.com/ufs-community/ufs-srweather-app (push)
```

   Rename the 'origin' remote repository to the 'upstream' remote repository:

```
git remote rename origin upstream
```

   Create a new 'origin' remote repository that points to your fork:

```
git remote add origin https://github.com/chan-hoo/ufs-srweather-app
git remote update
git remote -v
```

   - The output should be as follows:

```
origin   https://github.com/chan-hoo/ufs-srweather-app (fetch)
origin   https://github.com/chan-hoo/ufs-srweather-app (push)
upstream   https://github.com/ufs-community/ufs-srweather-app (fetch)
upstream   https://github.com/ufs-community/ufs-srweather-app (push)
```

4. Update the fork of SRW App:

   Update your local 'master' branch with the latest upstream repository:

```
git checkout master
git pull upstream master
```

Update the 'master' branch in your fork

```
git push origin master
```

5. If a 'master_wcoss' branch does not exist, create a new branch named 'master_wcoss':

```
cd {HOMEdir}/ufs-srweather-app
git checkout -b master_wcoss
```

Push the 'master_wcoss' branch into your fork:

```
git push origin master_wcoss
```

6. Check out the external components including the regional workflow:

```
./manage_externals/checkout_externals
```

Note) This step intends to clone the 'develop' branch (or the specific hash) of the official 'NOAA-EMC/regional_workflow' repository, specified in 'Externals.cfg', as the 'upstream' remote repository.

7. Modify the 'Externals.cfg' file:

Open the configuration file 'Externals.cfg':

```
cd {HOMEdir}/ufs-srweather-app
vim Externals.cfg
```

Modify the path of the repository and its branch name:

```
repo_url = https://github.com/chan-hoo/ufs-srweather-app
branch = master_wcoss
```

8. Commit the change in SRW App:

Commit (save) the changes you made to your local 'master_wcoss' branch.

```
cd {HOMEdir}/ufs-srweather-app
git add Externals.cfg
git commit
(write a commit message and close the commit window)
```

Push your local 'master_wcoss' branch to your fork:

```
git push origin master_wcoss
```

Your fork will now have a version of the 'master_wcoss' branch with all of your commits.

10.2.2 Modification of Regional Workflow

1. Fork the regional workflow:

   In a web browser, go to the GitHub page:

   ```
   https://github.com/NOAA-EMC/regional_workflow
   ```

   Press the 'Fork' button on the top right. You can see the repositories in your GitHub account. The URL of your fork will be

   ```
   https://github.com/chan-hoo/regional_workflow
   ```

2. Configure Git Remotes for the regional workflow:

   When you clone your fork, git will create a remote named 'origin' that points to the authoritative repository. Check the list of remotes that are defined in your clone:

   ```
   cd {HOMEdir}/ufs-srweather-app/regional_workflow
   git remote -v
   ```

   - The output should be as follows:

     ```
     origin https://github.com/NOAA-EMC/regional_workflow (fetch)
     origin https://github.com/NOAA-EMC/regional_workflow (push)
     ```

   Rename the 'origin' remote repository to the 'upstream' remote repository:

   ```
   git remote rename origin upstream
   ```

   Create a new 'origin' remote repository that points to your fork 'GitHubID/regional_workflow' repository:

   ```
   git remote add origin https://github.com/chan-hoo/regional_workflow
   git remote update
   git remote -v
   ```

- The output should be as follows:

```
origin   https://github.com/chan-hoo/regional_workflow (fetch)
origin   https://github.com/chan-hoo/regional_workflow (push)
upstream   https://github.com/NOAA-EMC/regional_workflow (fetch)
upstream   https://github.com/NOAA-EMC/regional_workflow (push)
```

3. If a 'devlop_wcoss' branch does not exist, create a new branch named 'develop_wcoss':

```
git checkout -b develop_wcoss
```

Push the 'develop_wcoss' branch into your fork:

```
git push origin develop_wcoss
```

4. **Edit/Add/Delete** files in 'regional_workflow'.

5. Commit (save) the changes you made to your local 'develop_wcoss' branch:

```
cd {HOMEdir}/ufs-srweather-app/regional_workflow
git add file1 {sub_dir}/file2
git commit
(write a commit message and close the commit window)
```

Push your local 'develop_wcoss' branch to your fork:

```
git push origin develop_wcoss
```

Your fork will now have a version of the 'develop_wcoss' branch with all of your commits.

10.2.3 Cloning Your Fork's Branches to Other Machines

1. Clone the UFS SRW App to the '{HOMEdir}' directory (local repository):

```
cd {HOMEdir}
git clone -b master https://github.com/ufs-community/ufs-srweather-app
```

When you clone your fork, git will create a remote named 'origin' that points to the authoritative repository. Check the list of remotes that are defined in your clone:

```
cd ufs-srweather-app
git remote -v
```

- The output should be as follows:

```
origin https://github.com/ufs-community/ufs-srweather-app (fetch)
origin https://github.com/ufs-community/ufs-srweather-app (push)
```

2. Check out the external components including the regional workflow:

```
./manage_externals/checkout_externals
```

Note) This step intends to clone the 'develop' branch (or the specific hash) of the official 'NOAA-EMC/regional_workflow' repository, specified in 'Externals.cfg', as the 'upstream' remote repository.

3. Configure Git Remotes for SRW App:

Rename the 'origin' remote repository to the 'upstream' remote repository:

```
git remote rename origin upstream
```

Create a new 'origin' remote repository that points to your fork:

```
git remote add origin https://github.com/chan-hoo/ufs-srweather-app
git remote update
git remote -v
```

- The output should be as follows:

```
origin   https://github.com/chan-hoo/ufs-srweather-app (fetch)
origin   https://github.com/chan-hoo/ufs-srweather-app (push)
upstream   https://github.com/ufs-community/ufs-srweather-app (fetch)
upstream   https://github.com/ufs-community/ufs-srweather-app (push)
```

4. Switch to the 'master_wcoss' Branch of SRW App:

```
git branch -a
git checkout master_wcoss
```

5. Configure Git Remotes for the regional workflow:

   When you clone your fork, git will create a remote named 'origin' that points to the authoritative repository. Check the list of remotes that are defined in your clone:

```
cd {HOMEdir}/ufs-srweather-app/regional_workflow
git remote -v
```

   - The output should be as follows:

```
origin https://github.com/NOAA-EMC/regional_workflow (fetch)
origin https://github.com/NOAA-EMC/regional_workflow (push)
```

   Create a new 'origin' remote repository that points to the 'chan-hoo/regional_workflow' repository:

```
git remote rename origin upstream
git remote add origin https://github.com/chan-hoo/regional_workflow
git remote update
git remote -v
```

   - The output should be as follows:

```
origin   https://github.com/chan-hoo/regional_workflow (fetch)
origin   https://github.com/chan-hoo/regional_workflow (push)
upstream   https://github.com/NOAA-EMC/regional_workflow (fetch)
upstream   https://github.com/NOAA-EMC/regional_workflow (push)
```

6. Switch to the 'develop_wcoss' Branch of SRW App:

```
git branch -a
git checkout develop_wcoss
```

7. **Edit/Add/Delete** files in 'regional_workflow'.

8. Commit (save) the changes you made to your local 'develop_wcoss' branch:

```
cd {HOMEdir}/ufs-srweather-app/regional_workflow
git add file1 {sub_dir}/file2
git commit
(write a commit message and close the commit window)
```

Push your local 'develop_wcoss' branch to your fork:

```
git push origin develop_wcoss
```

Your fork will now have a version of the 'develop_wcoss' branch with all of your commits.

9. **Edit/Add/Delete** files in 'ufs-srweather-app'.

10. Commit (save) the changes you made to your local 'master_wcoss' branch:

```
cd {HOMEdir}/ufs-srweather-app
git add file1 {sub_dir}/file2
git commit
(write a commit message and close the commit window)
```

Push your local 'master_wcoss' branch to your fork:

```
git push origin master_wcoss
```

Your fork will now have a version of the 'master_wcoss' branch with all of your commits.

10.2.4 Updating Cloned SRW and Workflow with the Latest Versions

1. Check the remote repositories and current branch:

```
cd {HOMEdir}/ufs-srweather-app
git remote -v
git remote update
git branch -a
```

2. Update the 'master' branch of your local repository and fork with the remote 'upstream' repository:

```
git checkout master
git pull
git push origin master
```

3. Merge the local ‘master_wcoss’ branch with the latest ‘upstream/master’ branch:

```
git checkout master_wcoss
git pull upstream master
```

4. Update the local ‘master_wcoss’ branch with the remote ‘origin’ repository (your fork):

```
git pull origin master_wcoss
```

5. Update the ‘master_wcoss’ branch in your fork:

```
git push origin master_wcoss
```

6. Check the remote repositories and current branch of the regional workflow:

```
cd {HOMEdir}/ufs-srweather-app/regional_workflow
git remote -v
git remote update
git branch -a
```

7. Update the ‘develop’ branch of your local repository and fork with the remote ‘upstream’ repository:

```
git checkout develop
git pull
(or 'git fetch upstream' and then 'git merge <hash #>')
git push origin develop
```

8. Merge the local ‘develop_wcoss’ branch with the latest ‘upstream/develop’ branch:

```
git checkout develop_wcoss
git pull upstream develop
```

9. Update the local ‘develop_wcoss’ branch with the remote ‘origin’ repository (your fork):

```
git pull origin develop_wcoss
```

10. Update the 'develop_wcoss' branch in your fork:

```
git push origin develop_wcoss
```

10.2.5 Cloning Another Remote Branch to the Local Repository

You can clone another remote branch to your existing local repository. For example, you have the local SRW repository where the 'master' branch is only cloned, and you want to add the 'release/public-v1' branch to the local SRW repository:

1. Check the branches cloned in your local repository:

```
cd {HOMEdir}/ufs-srweather-app
git remote -v
git remote update
git branch -a
```

2. If the 'release/public-v1' branch does not exist, you can clone it with the flag '-t' from the remote repository:

```
git checkout -t remotes/upstream/release/public-v1
git branch -a
```

3. Update the 'release/public-v1' branch in you fork:

```
git push origin release/public-v1
```

Chapter 11

Supporting Tools

11.1 Grid Coordinates: *fv3grid*

The coordinates of global (parent) and regional (child, nest) grids can be depicted by *fv3grid*. This tool is useful to check the location of a regional domain on the global domain as well as on the rotated domain.

11.1.1 Regional Domain

Parameters for creating a new regional domain can be found as:

1. Pop up a display window of *fv3grid*:

   - On Hera

   ```
   /scratch2/NCEPDEV/fv3-cam/Dusan.Jovic/dbrowse/fv3grid
   ```

   Note) To pop up a display window on the terminal, 'ssh' should accompany with '-Y' as

   ```
   'ssh -Y Chan-hoo.Jeon@hera-rsa.rdhpcs.noaa.gov'.
   ```

   Note) If your local machine is Mac, you should install 'XQuartz' in advance to pop up a display window.

2. Set the parameters for Tile 6 of the global grid:

Name	Description
resolution C	Grid resolution
stretch factor	Stretching factor for the global grid
target lon	Center longitude of Tile 6 (in degrees)
target lat	Center latitude of Tile 6 (in degrees)

Table 11.1: Parameters for a global grid.

Note) It is recommended that the 'target lon' and 'target lat' have the same values with the center of the regional domain for convenience.

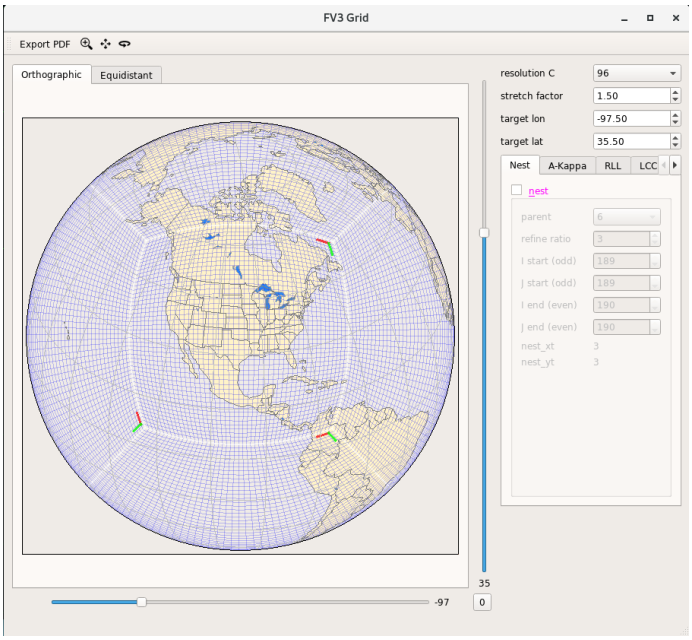

Figure 11.1: Change of resolution and stretch factor

3. Check the box of 'nest' for a regional grid, and set the parameters:

Name	Description
parent	Number of the parent tile where the regional grid is located (=6)
refine ratio	Grid refinement ratio
I start (odd)	Lowest *i*-index of the parent-tile super-grid corresponding to the regional domain

J start (odd)	Lowest *j*-index of the parent-tile super-grid corresponding to the regional domain
I end (even)	Highest *i*-index of the parent-tile super-grid corresponding to the regional domain
J end (even)	Highest *j*-index of the parent-tile super-grid corresponding to the regional domain

Table 11.2: Parameters for a regional grid.

Notes)

- To make the domain symmetry along the centerline, 'I end'=2×'resolution C'-('I start'-1), and 'J end'=2×'resolution C'-('J start'-1).
- The 'resolution C' means that the size of the parent tile 6 is {2×'resolution C'}×{2×'resolution C'} for the *fv3* super-grid or {'resolution C'}×{'resolution C'} for a general grid.

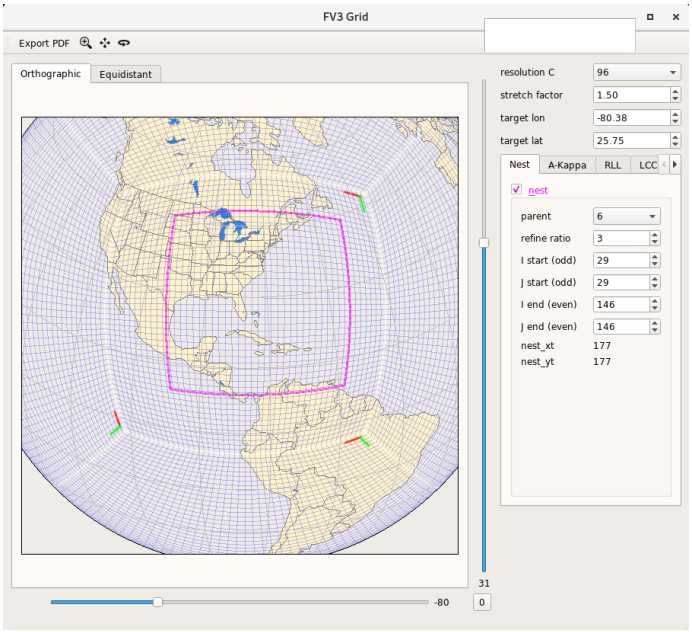

Figure 11.2: New regional region of interest

11.1.2 Rotated Domain for 'output_grid'

Parameters for plotting extent in 'model_configure' can be found as:

1. Pop up a display window of *fv3grid*:

   - On Hera:

   ```
   /scratch2/NCEPDEV/fv3-cam/Dusan.Jovic/dbrowse/fv3grid
   ```

2. Set the parameters for Tile 6 in the global and regional domains as shown in Section 11.1.1:

3. Click the 'RLL' tab on the right of the window, and check the box of 'Rotated Lat Lon':

4. Adjust the parameters to put the blue box inside the regional domain in purple:

Name	Description
tlm0d	Longitude of the center of the rotated domain (in degrees)
tph0d	Latitude of the center of the rotated domain (in degrees)
wbd	Longitudinal distance from the center to the bottom-left corner of the rotated domain (in degrees)
sbd	Latitudinal distance from the center to the bottom-left corner of the rotated domain (in degrees)

Table 11.3: Parameters for a RLL grid.

Note) 'tlm0d' and 'tph0d' do not need to be the same as 'target_lon' and 'target_lat', respectively. However, it is recommended to make them the same each other.

Note) The top-right corner of the rotated domain (box in blue) is automatically set as the same distance from the center as 'wbd' and 'sbd'.

11.1.3 A–Kappa (α–κ) Domain for an ESG (JP) Grid

1. Pop up a display window of *fv3grid*:

   - On Hera:

   ```
   /scratch2/NCEPDEV/fv3-cam/Dusan.Jovic/dbrowse/fv3grid
   ```

2. Click the 'nest' check box in the 'Nest' tap, and set up the GFDL domain for comparison if necessary:

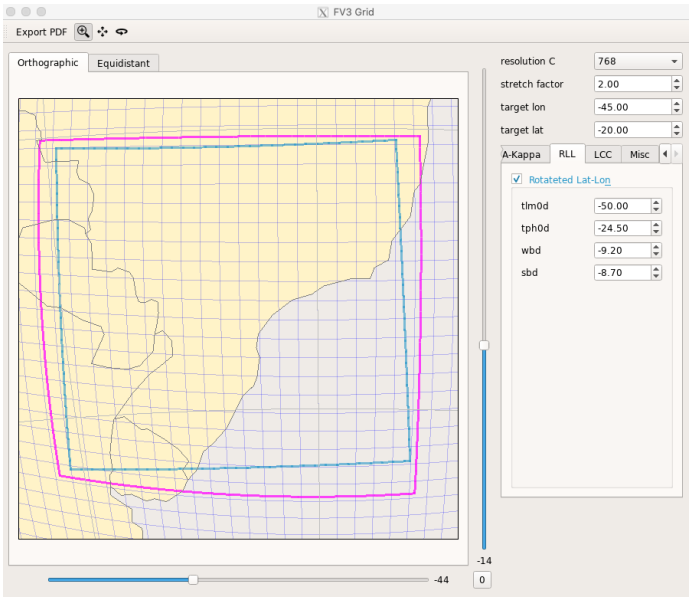

Figure 11.3: Parameters on the rotated grid

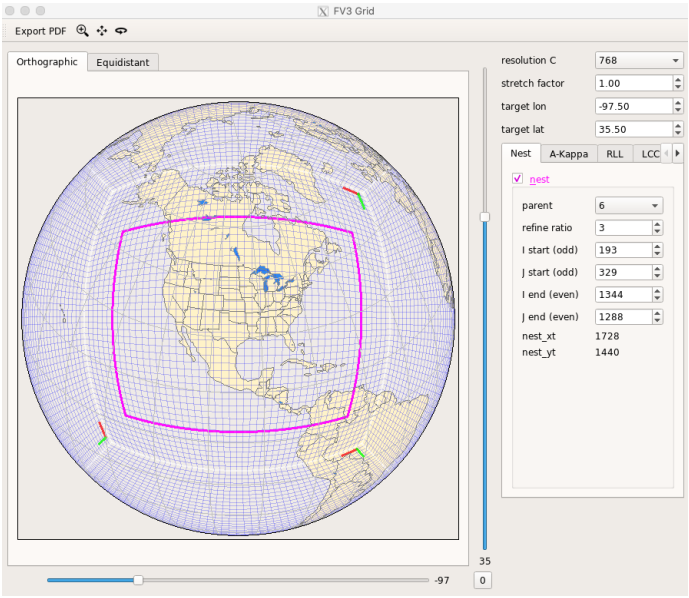

Figure 11.4: Parameters of a GFDL grid for camparison

3. Click the ‘A-Kappa’ check box in the ‘A-Kappa’ tap, and adjust the grid parameters for an ESG (JP) grid:

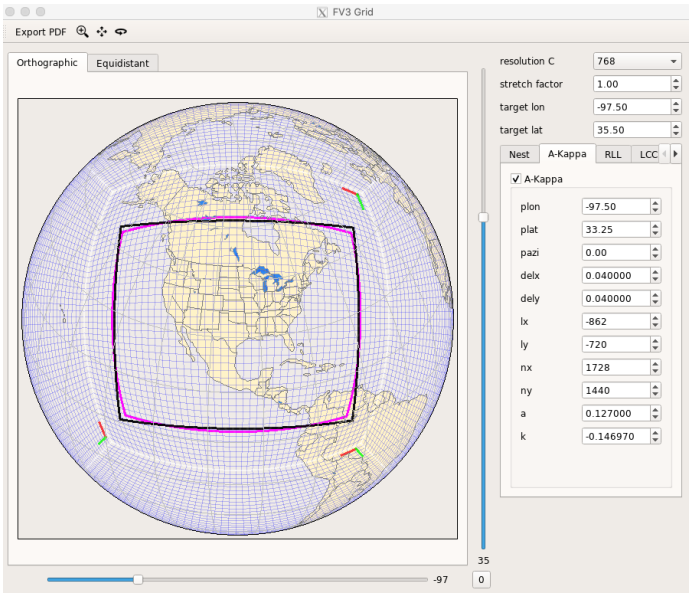

Figure 11.5: Parameters for an ESG (JP) grid

11.2 Plotting with *Python*

11.2.1 'Natural Earth' Data for Background

Download shape files:

```
www.naturalearthdata.com/downloads/
```

The default scale (resolution) of background attributes built in *python* is 1:110m. This is good enough for the global domain. However, the medium scale (1:50m) or large scale (1:10m) data are recommended for a regional domain as shown in Figure 11.6.

- For 'Cultural':

```
www.naturalearthdata.com/downloads/50m-cultural-vectors/
# Click 'Download all 50m cultural thememes'
```

- For 'Physical':

```
www.naturalearthdata.com/downloads/50m-physical-vectors/
# Click 'download all 50m physical themes'
```

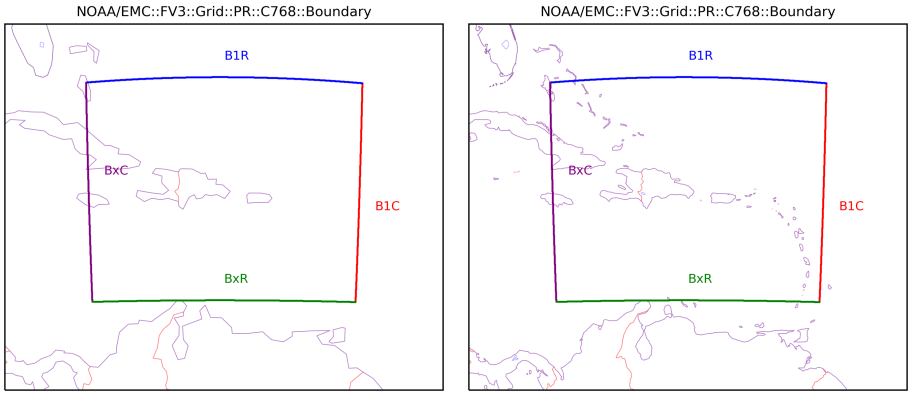

Figure 11.6: Background scales: 1/110m vs. 1/50m

Put the files into the specific structures of directories as below:

```
$data_dir/shapefiles/natural_earth/physical (or cultural)
```

Or, the files can be found as follows:

- On Hera:

```
/scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/tools/NaturalEarth/
```

- On Orion:

```
/home/chjeon/tools/NaturalEarth/
```

Set the path in the script:

- On Hera:

```
cartopy.config['data_dir']='/scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/tools/
    NaturalEarth'
```

- On Orion:

```
cartopy.config['data_dir']='/home/chjeon/tools/NaturalEarth'
```

11.2.2 Cartopy Background Image

Cartopy provides the 'background_img()' method to add background images in a convenient way. You can specify custom background images.

1. Download background images such as Natural Earth data and put them in the specific directory.

2. Set up the environment variable for the path of the directory that contains the background images:

```
{
os.environ["CARTOPY_USER_BACKGROUNDS"]="/scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/tools/NaturalEarth/
    raster_files"
}
```

3. The arguments of 'name' and 'resolution' for the 'background_img()' are specified in the configuration file 'images.json' that should be located in the above directory. An example of the 'images.json' file is as follows:

```
{
  "NE": {
  "__comment__": "Natural Earth raster file",
  "__source__": "www.naturalearthdata.com/downloads",
  "__projection__": "PlateCarree",
  "high": "NE1_50M_SR_W.tif"}
}
```

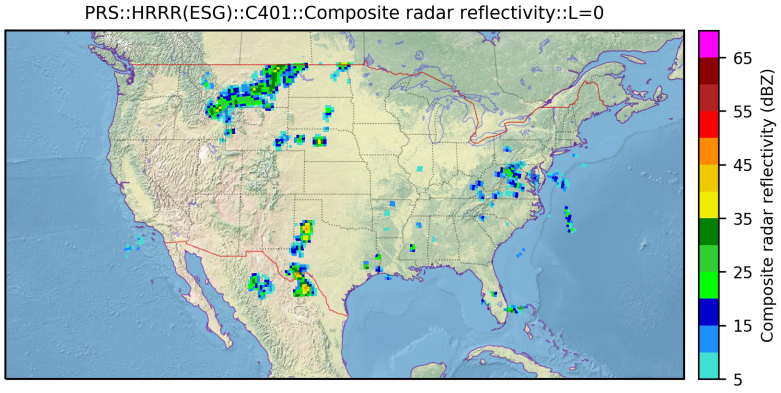

Figure 11.7: Background image: Natural Earth data

4. Call 'background_img()' in the script as follows:

```
ax.background_img(name='NE', resolution='high')
```

Note) A low-resolution background image can be easily called by 'stock_img()' in a script.

11.2.3 Modules

If a HPC, such as Hera, provides a module of 'Anaconda/Miniconda', you can load modules for running python scripts as follows:

- On Hera:

```
module load intel
module use -a /contrib/anaconda/modulefiles
module load anaconda/latest
```

Note) If 'Miniconda (or Anaconda)' is installed in the home directory, you do not need to load the above modules for running post-processing python scripts.

To import 'pygrib' on WCOSS, you should modules as follows:

- WCOSS Dell:

```
module load ips/19.0.5.281
module load impi/19.0.5
module load grib_util/1.1.1
module load python/3.6.3
module use -a /u/Benjamin.Blake/modulefiles
module load python3/test
export GRIB_DEFINITION_PATH=/usrx/local/nceplibs/dev/lib/grib_api/share/
    grib_api/definitions
```

11.2.4 Installation of Miniconda (Anaconda)

If a HPC, such as Orion, does not provide a module of 'Anaconda' for python packages, you need to install either 'Miniconda' or 'Anaconda' in your home/work directory.

1. Download the installer from the Miniconda repository:

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

2. Verify the installer hashes:

```
sha256sum Miniconda3-latest-Linux-x86_64.sh
# Compare the hash code to that on the website ('https://conda.io/projects/
    conda/en/latest/user-guide/install/linux.html')
```

3. Install Miniconda:

```
bash Miniconda3-latest-Linux-x86_64.sh
(enter)
(space bar)
(yes)
(enter) or specify the location to be installed ({$CONDA_DIR}).
(enter)
```

4. Update the path

```
vim ~/.bashrc
# insert
export PATH={$CONDA_DIR}/miniconda3/bin:$PATH
# close the '.bashrc' file
source ~/.bashrc
```

5. Test the installation

```
conda list
```

6. Install the packages required for running the post-processing python scripts:

```
conda install cartopy
conda install xarray
conda install netcdf4
conda install dask
conda install -c conda-forge pygrib
```

11.2.5 Python Scripts on GitHub

The python scripts described in this section can be cloned from GitHub as:

```
git clone https://github.com/chan-hoo/FV3LAM_plot_python.git
```

The list of the scripts in the above repository is shown in Table 11.4. The scripts are based on Python 3.7, and Python libraries of 'matplotlib', 'xarray', and 'cartopy' are applied.

Script name	Description
modules_hera	Module list for Anaconda on Hera
modules_wcoss_dell	Module list for pygrib on the WCOSS Dell
plot_fv3lam_grid_oro.py	Grid and orography files
plot_fv3lam_gridonly.py	Grid file alone
plot_fv3lam_static.py	Regional static field files
plot_fv3lam_fixam.py	Global static field files
plot_fv3lam_icbc.py	Time-dependent IC/LBC fields
plot_fv3lam_icsfc.py	Initial surface climatology fields
plot_fv3lam_co2his.py	Historical CO_2 data
plot_fv3lam_gridspec.py	Output 'grid_spec.nc' file
plot_fv3lam_atmstat.py	Output 'atmos_static.nc' file
plot_fv3lam_his2d_bndr.py	Boundary regions of output 'fv3_history2d.nc' file
plot_fv3lam_dynf.py	Output 'dynffXXX.nc' file
plot_fv3lam_phyf.py	Output 'phyfXXX.nc' file
plot_fv3lam_bgrd3d.py	Output 'BGRD3D' GRIB2 file
plot_fv3lam_bgdawp.py	Output 'BGDAWP' GRIB2 file
plot_fv3lam_comp2f.py	Comparison of two NetCDF (GRIB2) files
plot_fv3lam_mrms.py	MRMS radar data for reflectivity
plot_fv3lam_ani_comref.py	Animation of hourly comparison of GRIB2 and radar data
plot_fv3lam_grb2his.py	Time-history plot of max. total precipitation/ reflectivity from BGDAWP GRIB2 files

Table 11.4: Python scripts for plotting input and output files.

11.2.6 Grid and Orography

The python script 'plot_fv3lam_grid_oro.py' creates figures for grid and orography in a regional domain.

Note) If you want to plot the grid file without the orography file, you can use another script 'plot_ fv3lam_grid.py'.

1. Input files
   - gird.tile7.halo4.nc (grid)
   - oro_data.tile7.halo4.nc (orography)
2. Output files
   - fv3lam_grid.png (grid)
   - fv3lam_grid_bndr.png (boundary of domain)
   - fv3lam_grid_dxy.png (cell size)
   - fv3lam_grid_crnr_RXCX.png (grid at the four corners)
   - fv3lam_orog_{variables}.png (orography variables)
3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_grid_oro.py
# Run the script
python3 plot_fv3sar_grid_oro.py
```

Name	Description
machine	HPC machine for plotting
dnm_data	Path to the directory where the grid file is located
dnm_orog	Path to the directory where the orography file is located
fnm_in_grid	File name of the input grid file
fnm_in_orog	File name of the input orography file
n_skip	Number of grid points in lows/column skipped for plotting
out_fig_dir	Path to the directory where output files are created

out_grd_title	Basic title in the output grid files
out_grd_fname	Basic file name of the out grid files
out_orog_title_base	Basic title in the output orography files
out_orog_fname_base	Basic file name of the out orography files
orog_vars	Variable names plotted in the orography file
cmap_range_grd	Colormap range option (‘symmetry’, ‘round’, ‘fixed’)
back_res	Resolution of background natural earth data (‘50m’ or ‘110m’)

Table 11.5: Namelist in the script for plotting gird and orography.

4. Output:
   For example, the following files for the CONUS (C96) domain are created in the output directory:

   (a) Grid points for every ‘n_skip’ rows/columns (‘fv3_grid_CONUS_C96.png’)

   - For a clear view, grid points are shown at every ‘n_skip’-th row and column from ($i = 2, j = 2$) for the super-grid of *fv3* in red, and ($i = 1, j = 1$) for the typical grid such as the grid of orography in green. This is because the super-grid includes not only a grid point at the center of a cell but also grid points at the corner of a cell.
   - Default values of ‘n_skip’: 10 for ‘CONUS’, 50 for ‘AK’, 20 for ‘PR’, and 50 for others.

   (b) Grid points along the boundary (‘fv3_gird_CONUS_C96_bndr.png’)

   - This figure only shows the grid points along the four boundaries.
     i. B1C: Along the first column ($j = 1$) in red
     ii. B1R: Along the first row ($i = 1$) in blue
     iii. BxC: Along the last column ($j = x$) in purple
     iv. BxR: Along the last row ($i = x$) in green

   (c) Grid cell sizes (‘fv3_grid_CONUS_C96_dxy.png’)

   (d) Grid points at the four corners (‘fv3_grid_CONUS_C96_crnr_[X].png’)

   - Each corner has a separate plot: R1C1, R1Cx, RxC1, and RxCx.
     i. R1C1: Around the corner ($i = 1, j = 1$), intersection between B1R and B1C
     ii. R1Cx: Around the corner ($i = 1, j = x$), intersection between B1R and BxC

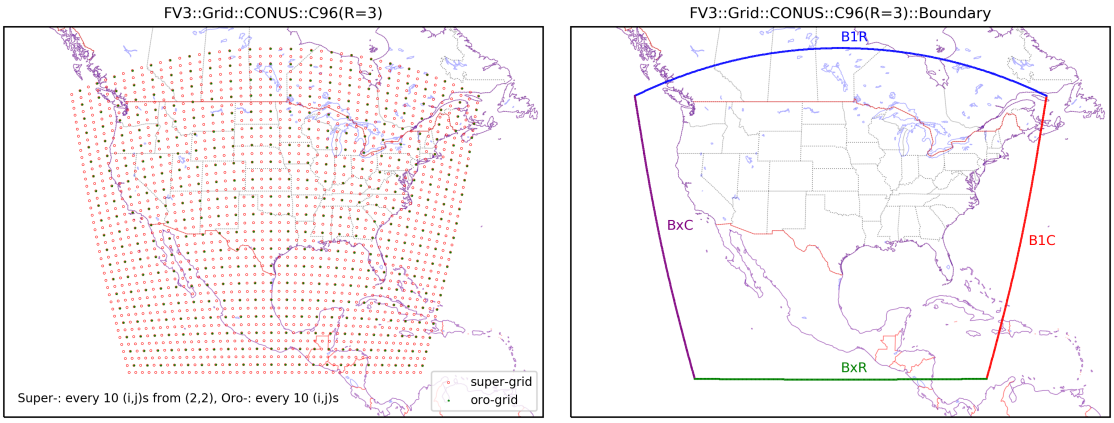

Figure 11.8: Grid points at every 10 rows and columns, and along boundaries

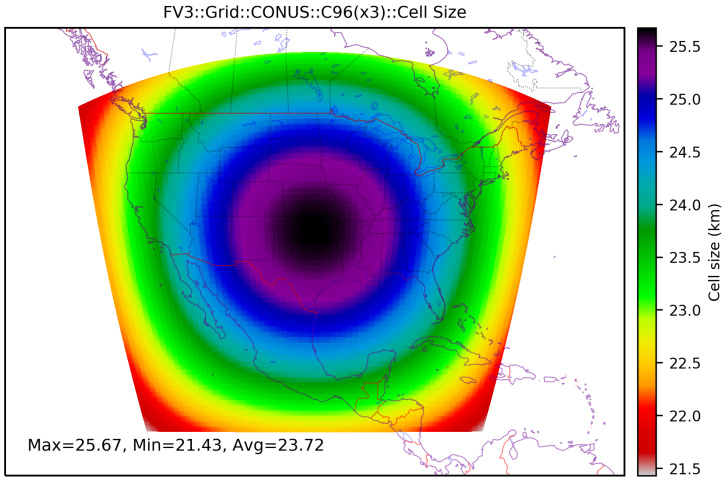

Figure 11.9: Cell size

iii. RxC1: Around the corner ($i = x, j = 1$), intersection between BxR and B1C

iv. RxCx: Around the corner ($i = x, j = x$), intersection between BxR and BxC

- Each figure shows 30 rows/columns by the default. It can be adjusted by modifying the value of 'N_crn' in the script.
- They plot both the *fv3* super-grid and the typical grid ('oro-grid' on the figures).

(e) Orography (raw and filtered)

(f) Sea-land mask (sea:0, land:1), and land fraction

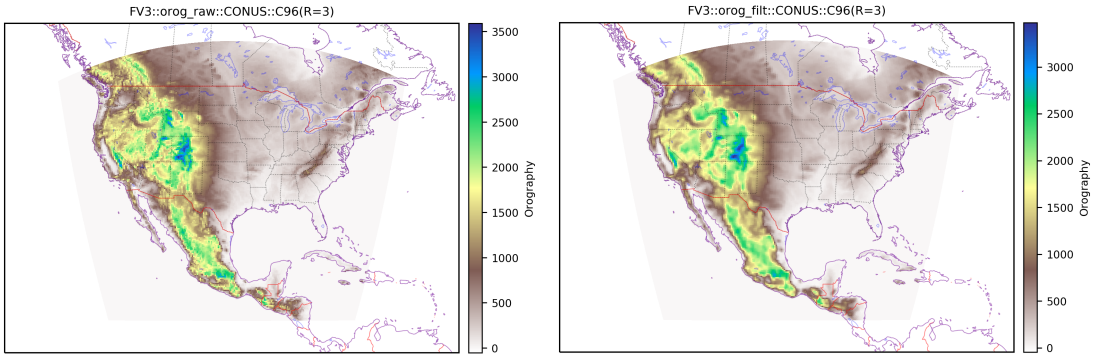

Figure 11.10: Orography

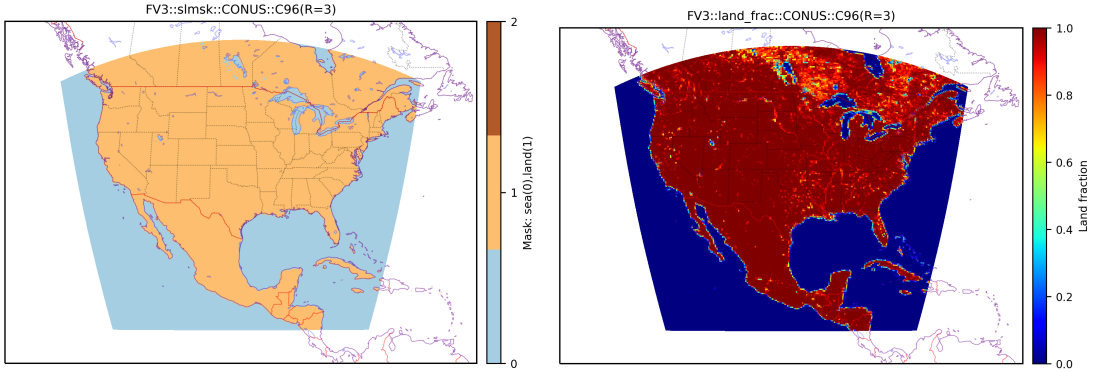

Figure 11.11: Sea-land mask, and land fraction

Note) The variables in the input orography file ('C{res}_oro_data.tile7.halo4.nc') are listed in Table 6.9.

11.2.7 Static Fields: Regional ('fix_lam')

The python script 'plot_fv3lam_static.py' creates figures for static fields in a regional domain as well as the original (global) data for comparison. The required files can be found in the namelist as described in Table 2.70.

1. Input files

   - C{res}.{variable}.tile7.halo[0 or 4].nc
   - C{res}_oro_data.tile7.halo[0 or 4].nc (orography file for coordinates)

- Original global static field files

Note) The halo number ('0' or '4') of the orography file must correspond to that of the variable file.

2. Output files

   - fv3lam_static_{domain}_C{res}_{variable}.png

3. Run on HPC

   - On Hera:

```
# Check Input and Output paths and names
vim plot_fv3lam_static.py
# Interactive mode on HPC
salloc --x11=first -q debug -t 0:30:00 --nodes=1 -A fv3-cam
# Run the script
python3 plot_fv3lam_static.py
```

   - On Orion:

```
# Check Input and Output paths and names
vim plot_fv3lam_static.py
# Interactive mode on HPC
salloc --ntasks=1 --time=00:30:00
# Run the script
python3 plot_fv3lam_static.py
```

Name	Description
machine	HPC machine for plotting
prt_data	Path to the original (global) data files
out_fig_dir	Path to the directory where output files are created
dnm_data	Path to the directory where the input NetCDF files are located
res	Resolution (C+resolution)
fnm_in_base	Basic form of the input NetCDF files
fnm_in_orog	File name of the orography file

vars_static	Specific field name of the input files
out_title_base	Basic form of the title in the figures
out_fname_base	Basic form of the output file name for the fields
out_title_base_g	Basic form of the title for the original global data
back_res	Resolution of the background plot

Table 11.6: Namelist in the script for plotting static fields.

Notes)

- Since the NetCDF files for the static fields do not contain the coordinates, the longitudes and latitudes of the grid points are imported from the orography file.
- In case of time-limit on the system such as an interactive job on Hera (limit=30min), split 'vars_static' into two: 'snowfree_albedo' and others. Moreover, the variables in 'snowfree_albedo' should be split into two in the function of 'static_plot': 'visible' and 'near_IR'.

4. Output: maximum snow albedo, and substrate temperature

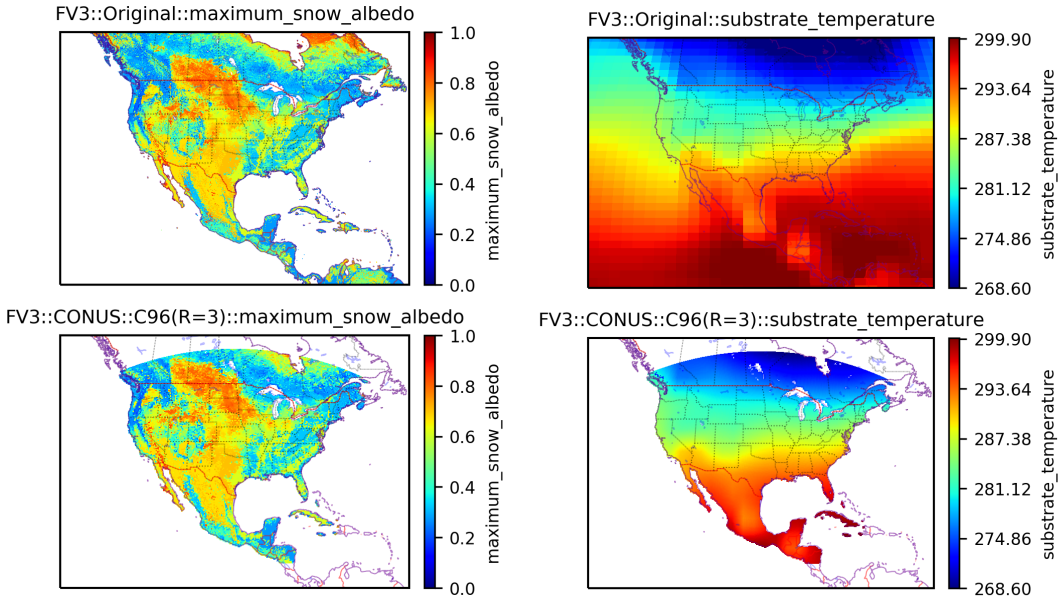

Figure 11.12: Maximum snow albedo, and Substrate temperature

11.2.8 Static Fields: Global ('fix_am')

The python script 'plot_fv3lam_fixam.py' creates figures for global static fields (input files, format: GRB) in the directory of 'fix_am'. The required files can be found in the namelist as described in Table 2.70.

1. Input files
   - global_glacier.2x2.grb
   - global_maxice.2x2.grb
   - global_snoclim.1.875.grb
   - global_soilmgldas.t1534.3072.1536.grb
   - CFSR.SEAICE.1982.2012.monthly.clim.grb
   - RTGSST.1982.2012.monthly.clim.grb
2. Output files
   - fv3lam_fixam_{variable}.png
   - fv3lam_fixam_{variable}_m{month}.png
   - fv3lam_fixam_{variable}_m{month}_L{level}.png
3. Run on HPC
   - On Hera:

```
# Check Input and Output paths and names
vim plot_fv3lam_fixam.py
# Submit interactive job for a long run-time (<30min)
salloc -q debug -t 0:30:00 --nodes=1 -A fv3-cam
# Run python script
python3 plot_fv3lam_fixam.py
```

   - On Orion:

```
# Check Input and Output paths and names
vim plot_fv3lam_fixam.py
# Interactive mode on HPC
salloc --ntasks=1 --time=00:30:00
```

```
# Run the script
python3 plot_fv3lam_fixam.py
```

Name	Description
machine	HPC machine for plotting
out_fig_dir	Path to the directory where output files are created
dnm_data	Path to the directory where the input NetCDF files are located
vars_fixam	Specific field name of the input files
out_title_base	Basic form of the title in the figures
out_fname_base	Basic form of the output file name for the fields
back_res	Resolution of the background plot

Table 11.7: Namelist in the script for plotting global static fields.

4. Output: sea Surface Temperature, and soilmgldas (March, L10)

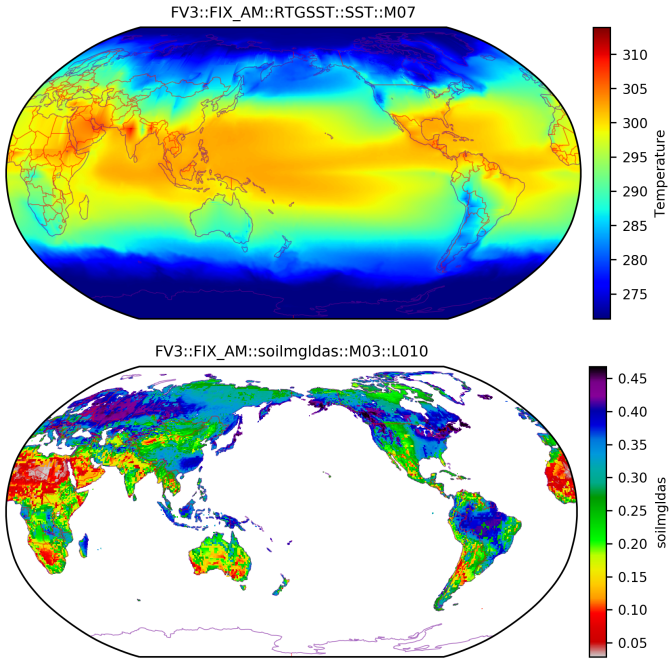

Figure 11.13: Sea surface temperature, and soilmgldas

11.2.9 Time-dependent IC/LBC Fields

The python script 'plot_fv3lam_icbc.py' creates figures for time-dependent initial (IC) and lateral boundary (LBC) fields on the halo and blending layers along the boundary of a regional domain as well as the original data of the fields for comparison. Since 'halo=4', '0 ≤n_blend≤ 10', and the number of grid points along the axis is more than 190, the IC/LBC fields in the files are too narrow and long to plot in the regular coordinate system. Therefore, the target areas in this script are four corners of the domain. Since *FV3* uses the Arakawa C and D grid systems, velocity components are evaluated at the centers of top, bottom, left, and right grid faces. Therefore, some of top, bottom, left, and right boundaries have one more layer (row/column) than other fields depending on which component is evaluated. The axes of top, bottom, left, and right are contracted to the middle of the axes to secure reasonable spaces for comparison between cells near the corners. However, this approach makes us look some discontinuous points. To verify the boundary fields, this script creates another file for the original GFS data. Since this GFS file contains the initial condition at t=000, it should be compared to the data of 'gfs_bndy.tile7.000.nc'. The figure shows two different coordinate systems with the same data: (1) general equi-distant spacing, and (2) contracted spacing to the middle, which is the same as the boundary plot.

1. Input files
   - gfs_bndy.tile7.[XXX].nc (for boundary plot)
   - gfs_data.tile7.nc (for initial GFS data plot in the main domain)
2. Output files
   - fv3lam_icbc_{Variable}_L{layer number}_t{time}_B{n_blend}.png (boundary)
   - fv3lam_icbc_{Variable}_L{layer number}_gfs.png (initial GFS data)
3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_icbc.py
# Run the script
python3 plot_fv3lam_icbc.py
```

Name	Description
machine	HPC machine for plotting
out_fig_dir	Path to the directory where output files are created
dnm_data	Path to the directory where the input NetCDF files are located
fnm_in_bndr_base	Basic form of the input NetCDF files
bndr_step	Time steps in the file names
vars_bc	Variables for plotting
ilvl	Specific number of the vertical level (layer number) for 3-D variables
out_title_base	Basic form of the title in the figure
out_fname_base	Basic form of the output file name
n_halo	Number of additional boundary arrays (halo), typically halo=4
n_blend	Number of blending layers

Table 11.8: Namelist in the script for plotting IC/LBC fields.

Notes)

- The namelist of the tracers can be found in Table 2.53.
- In the figures, the dashed lines in white represent the boundary of the regional domain.
- Total number of layers = n_halo + n_blend.

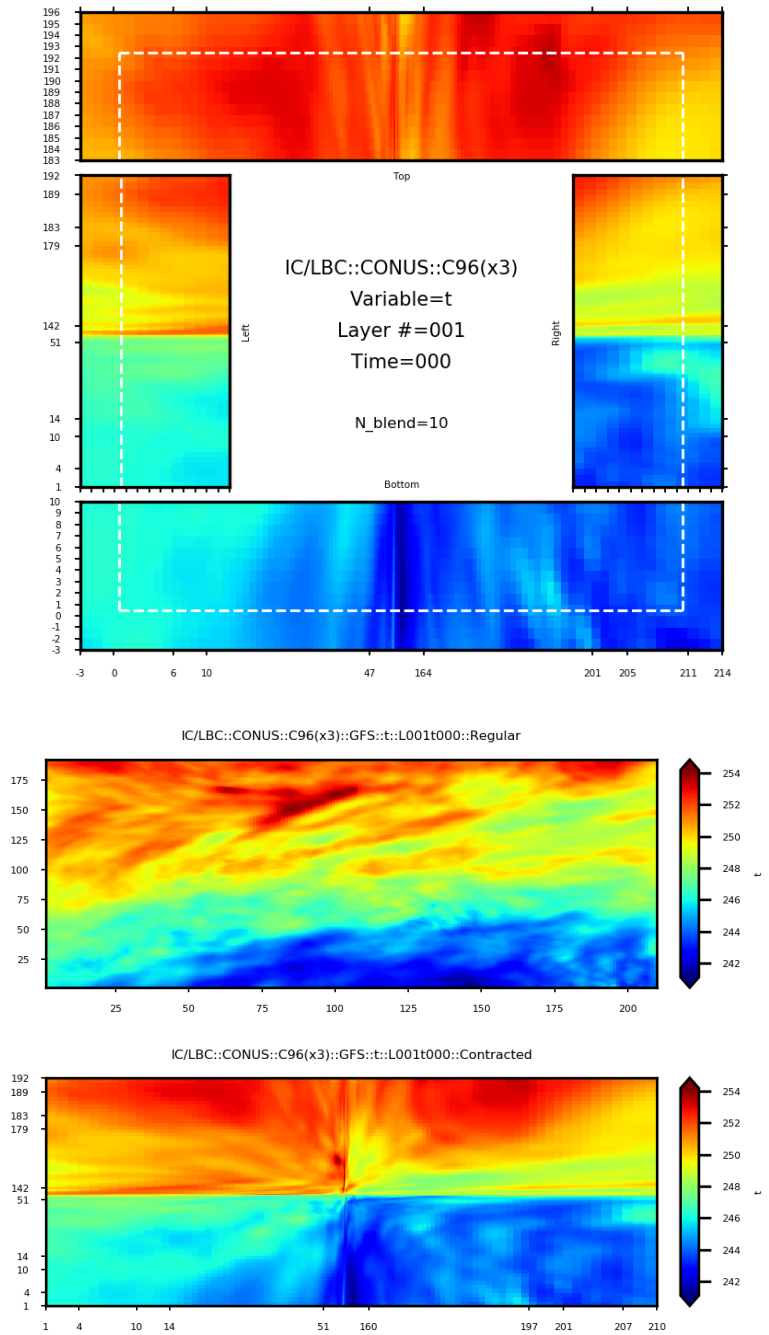

Figure 11.14: Temperature (t) along the boundary at the first layer (initial boundary condition, time=000, layer number=1) and GFS data inside the domain

11.2.10 Initial Surface Climatology Fields

This script creates a figure to illustrate surface climatology fields in the input 'fix' files.

1. Input files
   - sfc_data.nc
   - oro_data.nc
2. Output files
   - fv3lam_isfc_{domain}_C{res}_{Variable}.png
3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_isfc.py
# Run the script
python3 plot_fv3sar_isfc.py
```

Name	Description
machine	HPC machine for plotting
out_fig_dir	Path to the directory where output files are created
dnm_data	Path to the directory where the input NetCDF files are located
oro_path	Path to the orography file
fnm_input	Input NetCDF file
fnm_in_orog	Input orography file
vars_isfc	plotting variables in the data file (Table 2.52)
out_title_base	Basic form of the title in the figure
out_fname_base	Basic form of the output file name
back_res	Background resolution

Table 11.9: Namelist for plotting initial surface climatology.

4. Output:

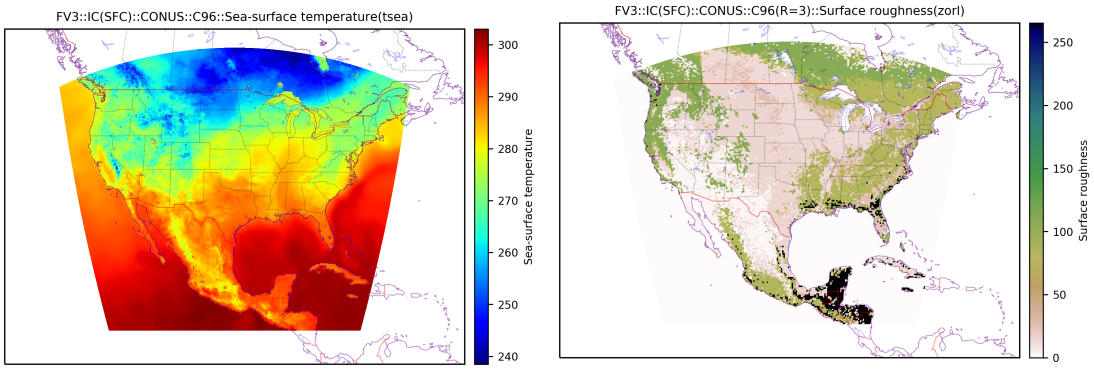

Figure 11.15: Sea-surface temperature, and roughness

11.2.11 Historical CO_2 Data

This script creates figures to illustrate monthly CO_2 data in the input 'co2historicaldata_20XX.txt' file.

1. Input files
   - co2historicaldata_20XX.txt
2. Output files
   - fv3lam_co2his_m{month}.png
3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_co2his.py
# Run the script
python3 plot_fv3lam_co2his.py
```

Name	Description
machine	HPC machine for plotting
out_fig_dir	Path to the directory where output files are created
dnm_data	Path to the directory where the input file is located
fnm_in	Input text file
out_title_base	Base of the figure title

out_fname_base	Base of the output file name
back_res	Background resolution

Table 11.10: Namelist in the script for plotting 'grid_spec'.

4. Output:

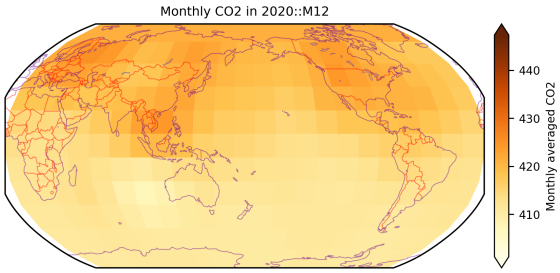

Figure 11.16: Monthly CO_2 input data

11.2.12 Output: 'grid_spec'

This script creates a figure to illustrate some grid points in the output 'gird_spec.nc' file.

1. Input files
   - grid_spec.nc
   - grid.tile7.halo4.nc
   - oro_data.tile7.halo4.nc
2. Output files
   - fv3lam_out_grdspec_{domain}_C{res}.png
3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_gridspec.py
# Run the script
python3 plot_fv3lam_gridspec.py
```

Name	Description
machine	HPC machine for plotting
out_fig_dir	Path to the directory where output files are created
dnm_out	Path to the directory where the input NetCDF files are located
fnm_input	Input NetCDF file
n_gpt	Number of grid points in plot
dnm_ref	Path to the directory where the grid and orography files are located
fnm_ref_grd	Grid file
fnm_ref_oro	Orography file
out_grd_title	Title of the figure
out_grd_fname	Output file name
back_res	Background resolution

Table 11.11: Namelist in the script for plotting 'grid_spec'.

4. Output:

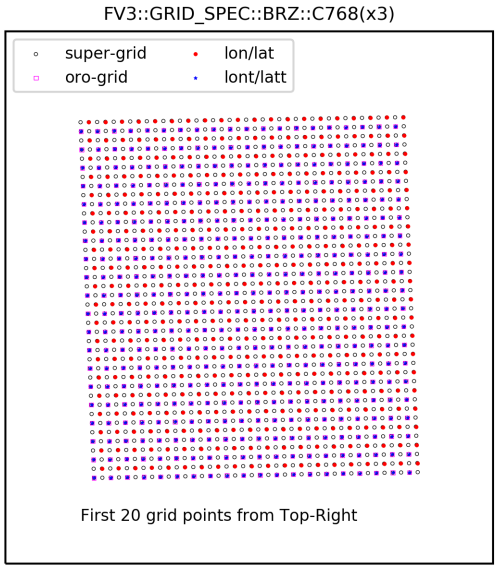

Figure 11.17: Output: grid_spec

11.2.13 Output: 'atmos_static'

1. Input files

   - atmos_static.nc
   - oro_data.nc

2. Output files

   - fv3lam_out_atmfix_[variable].png

3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_atmfix.py
# Run the script
python3 plot_fv3lam_atmfix.py
```

Name	Description
machine	HPC machine for plotting
data_dir	Path to the Natural Earth data for background
out_fig_dir	Path to the directory where output files are created
mfdf_kwargs	mfdataset argument
dnm_out	Path to the directory where the input NetCDF files are located
fnm_input	Input NetCDF file
dnm_ref	Path to the directory where the grid and orography files are located
fnm_ref_oro	Orography file
vars_atm	Variables in the input file, which can be found in Table 3.6
out_title_base	Title of the figure
out_fname_base	Output file name
back_res	Background resolution

Table 11.12: Namelist in the script for plotting 'atmos_static'.

4. Output:

   (a) One-dimensional data:

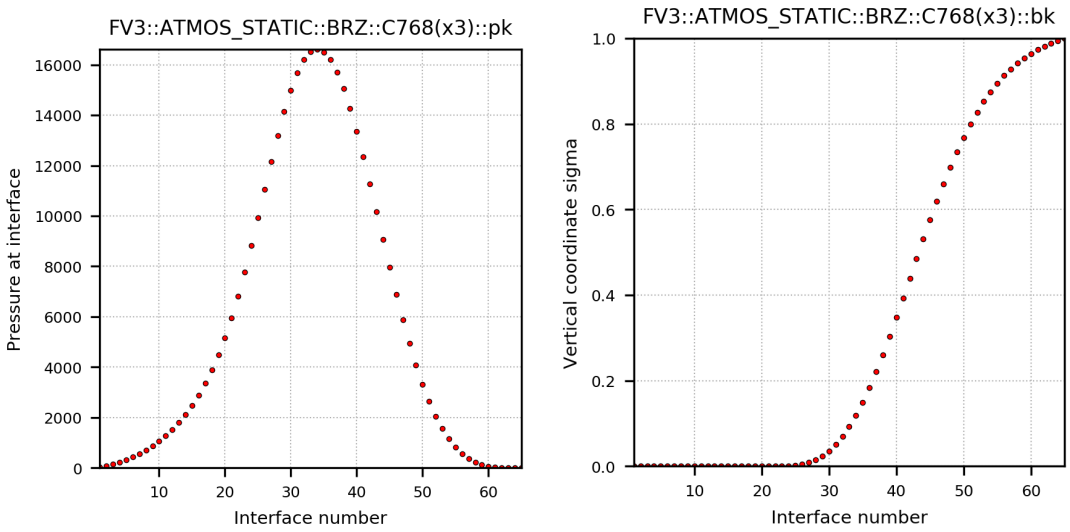

Figure 11.18: Output: atmos_static (‘pk’ and ‘bk’)

(b) Two-dimensional data:

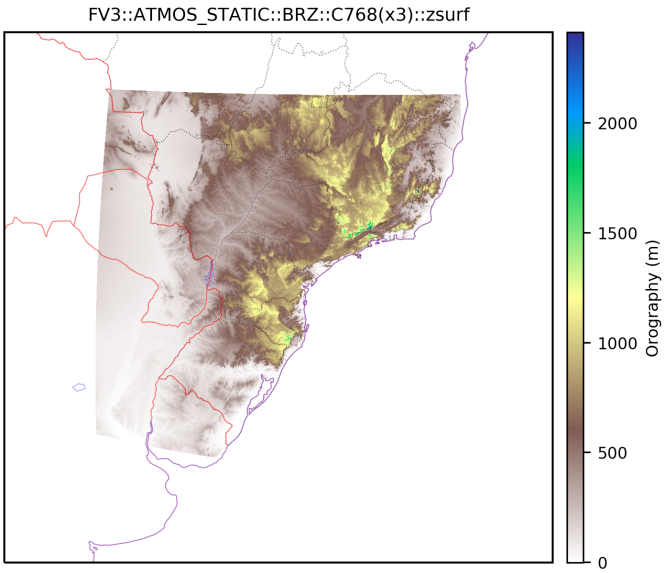

Figure 11.19: Output: atmos_static (‘zsurf’)

11.2.14 Output: 'fv3_history2d' - Boundary

This script creates figures to illustrate boundary regions of the output 'fv3_history2d.nc' file as well as the entire domain.

1. Input files
   - fv3_history2d.nc
2. Output files
   - fv3lam_out_his2d_[variable]_domain.png
   - fv3lam_out_his2d_[variable]_bndr.png
3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_his2d_bndr.py
# Run the script
python3 plot_fv3lam_his2d_bndr.py
```

Name	Description
machine	HPC machine for plotting
dnm_in	Path to the directory where the input NetCDF files are located
fnm_input	Input file name
vars_his	Variable of input file to be plotted
prt_tlvl	Time level of plotting
len_bndr	Length of boundary in plotting (unit: grid points)
dtick_s	Tick interval for short sides of boundary regions
dtick_l	Tick interval for long sides of boundary regions
out_fig_dir	Path to the directory where output files are created
out_title_base	Title of the figure
out_fname_base	Output file name

Table 11.13: Namelist in the script for plotting 'fv3_his2d_bndr'.

4. Output:

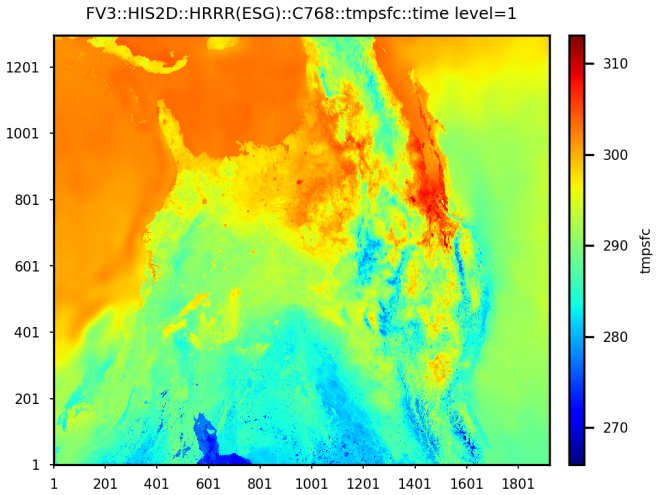

Figure 11.20: Output: fv3_history2d_domain

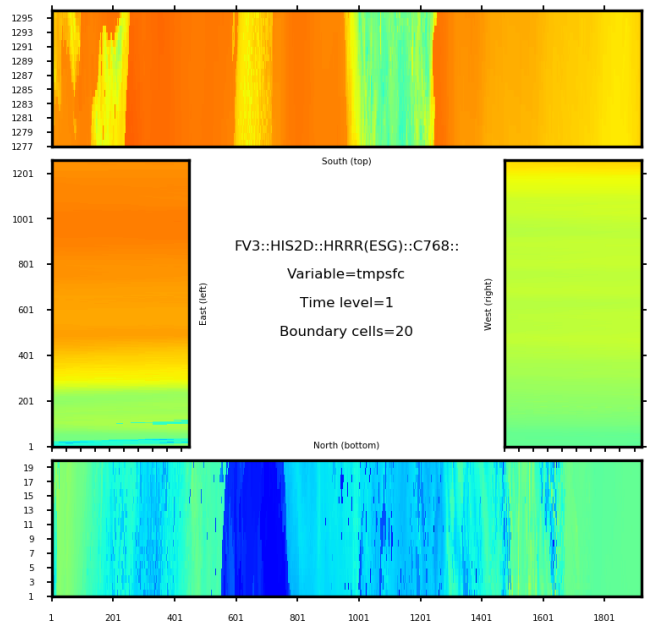

Figure 11.21: Output: fv3_history2d_bndr

11.2.15 Output: 'dynf'

1. Input files

   - dynfXXX.nc

2. Output files

   - fv3_out_dyn_[variable].png

3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_dynf.py
# Run the script
python3 plot_fv3lam_dynf.py
```

Name	Description
machine	HPC machine for plotting
data_dir	Path to the Natural Earth data for background
out_fig_dir	Path to the directory where output files are created
mfdf_kwargs	mfdataset argument
dnm_out	Path to the directory where the input NetCDF files are located
fnm_hr	Input file number
vars_dyn	Variables in the input file, which can be found in Section 2.6.4
lvl	Vertical layer number (only for 3-dimensional)
out_title_base	Title of the figure
out_fname_base	Output file name
back_res	Background resolution

Table 11.14: Namelist in the script for plotting 'dynfXXX'.

Note) Since 'dynf' and 'phyf' are the results of the write component option ('quilting=.true.'), their coordinates (longitude and latitude) would be different from those of input grid and orography, and output 'grid_spec.nc' files.

4. Output:

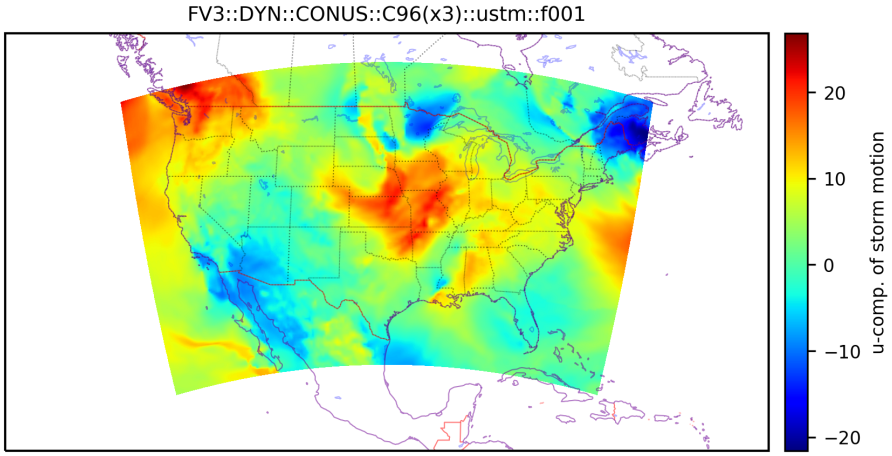

Figure 11.22: Output: dynfXXX

11.2.16 Output: 'phyf'

1. Input files
   - phyfXXX.nc
2. Output files
   - fv3lam_out_phy_[variable].png
3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_phyf.py
# Run the script
python3 plot_fv3lam_phyf.py
```

Name	Description
machine	HPC machine for plotting
out_fig_dir	Path to the directory where output files are created
dnm_out	Path to the directory where the input NetCDF files are located
fnm_hr	Input file number
vars_phy	Variables in the input file, which can be found in Section 2.6.4

out_title_base	Title of the figure
out_fname_base	Output file name
back_res	Background resolution

Table 11.15: Namelist in the script for plotting 'phyfXXX'.

4. Output:

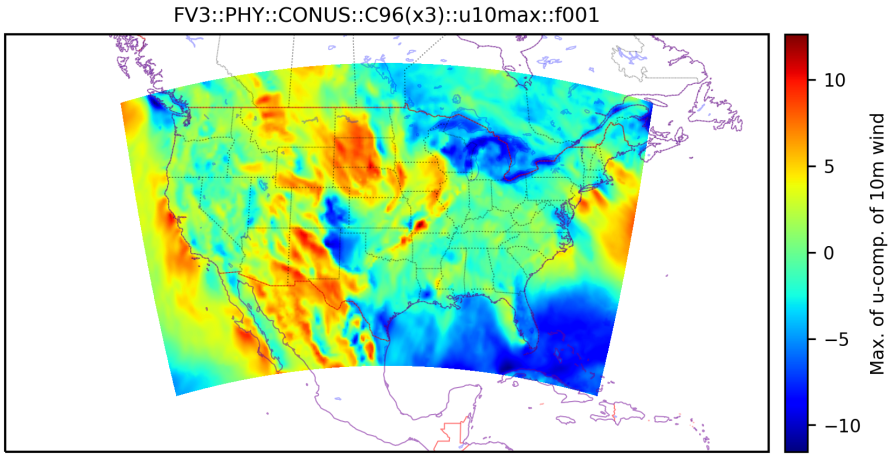

Figure 11.23: Output: phyfXXX

11.2.17 Output: 'BGRD3D (natlev.grib2)'

1. Input files
   - rrfs.tXXz.bgrd3dfXXX.tmXX.grib2 (*.natlev.grib2)
2. Output files
   - fv3lam_out_bgrd3d_fXXX_[variable].png
3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_bgrd3d.py
# Run the script
python3 plot_fv3lam_bgrd3d.py
```

Name	Description
machine	HPC machine for plotting
out_fig_dir	Path to the directory where output files are created
dnm_data	Path to the directory where the input GRIB2 file is located
fnm_hr	Input file hour number (‘fXXX’)
fnm_in	Input file name
vars_grb2	Variables in the input file (Table 8.2)
nlvl	Total number of vertical layers in the model
ilvl	Layer number to be plotted
out_title_base	Title of the figure
out_fname_base	Output file name
back_res	Background resolution (‘10m’, ‘50m’, ‘100m’)
back_img	Background image (‘on’, ‘off’)

Table 11.16: Namelist in the script for plotting ‘bgrd3d.grib2’.

4. Output:

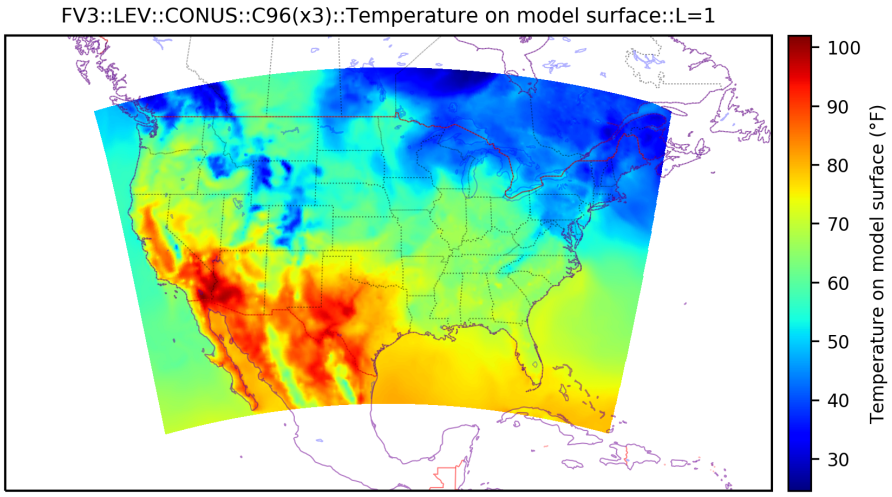

Figure 11.24: Output: BGRD3D (grib2)

11.2.18 Output: 'BGDAWP (natprs.grib2)'

1. Input files

   - rrfs.tXXz.bgdawpfXXX.tmXX.grib2 (*.natprs.grib2)

2. Output files

   - fv3lam_out_bgdawp_fXXX_[variable].png

3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_bgdawp.py
# Run the script
python3 plot_fv3lam_bgdawp.py
```

Name	Description
machine	HPC machine for plotting
data_dir	Path to the Natural Earth data for background
out_fig_dir	Path to the directory where output files are created
dnm_out	Path to the directory where the input GRIB2 file is located
fnm_hr	Input file hour number ('fXXX')
fnm_in	Input file name
vars_grb2	Variables in the input file (Table 8.3)
ilvl	Layer number to be plotted
out_title_base	Title of the figure
out_fname_base	Output file name
back_res	Background resolution ('10m', '50m', '100m')
back_img	Background image ('on', 'off')

Table 11.17: Namelist in the script for plotting 'bgdawp.grib2'.

4. Output:

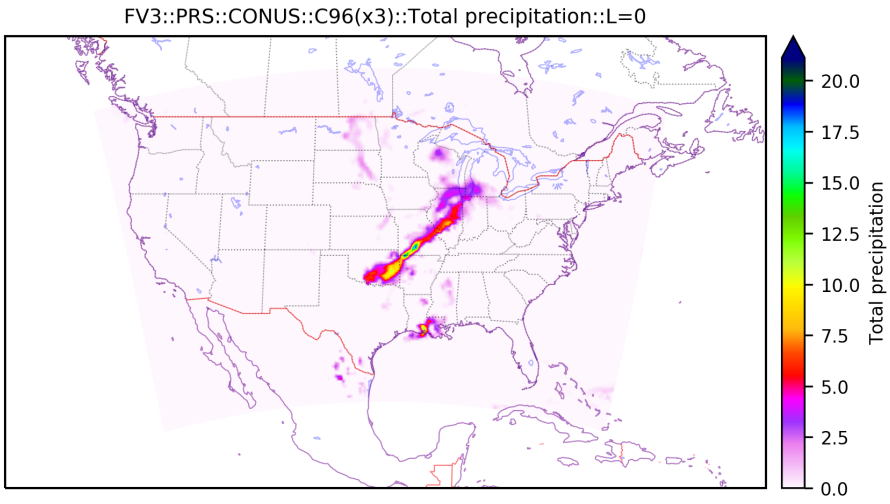

Figure 11.25: Output: BGRAWP (grib2)

11.2.19 Output: Comparison of Two NetCDF (GRIB2) Files

This script creates a figure to illustrate a comparison between two NetCDF or GRIB2 files on the same grid.

1. Input files
   - Two NetCDF or GRIB2 files on the same grid.
2. Output files
   - fv3lam_comp_[variable].png
3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_comp2f.py
# Run the script
python3 plot_fv3lam_comp2f.py
```

Name	Description

machine	HPC machine for plotting
out_fig_dir	Path to the directory where output files are created
dnm_in1	Path to the directory where the first input file is located
dnm_in2	Path to the directory where the second input file is located
fnm_in1	Name of the first Input file
fnm_in2	Name of the second Input file
vars_phy	Variables in the input file
grb_name	Name of the variable in GRIB2 (only for GRIB2)
grb_typlvl	Type of level of the variable in GRIB2 (only for GRIB2)
ilvl	Layer number to be plotted (only for 3-dimensional)
cmp_lbl	Additional text for comparison in title and label
out_title_base	Title of the figure
out_fname_base	Output file name
back_res	Background resolution

Table 11.18: Namelist in the script for plotting comparison.

4. Output:

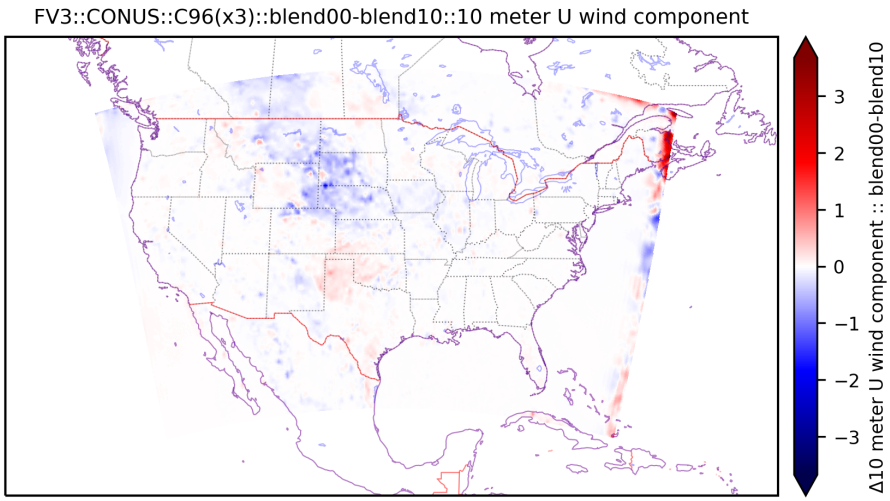

Figure 11.26: Output: comparison

11.2.20 MRMS: Composite Reflectivity Radar Data

1. Input files

   - QCComposite_00.50_[date]_[time]_new.grib2 (Composite reflectivity)
   - ref3D_[date]_[time]_new.grib2 (26 layer reflectivity)

   Notes)

   - The MRMS radar data can be obtained from HPSS as shown in Section B.2.
   - The original GRIB2 file should be converted to complex packing by *wgrib2* to avoid an API error using *Python*.

2. Output files

   - fv3lam_mrms_comRefl_[date]_[time].png (Composite reflectivity)
   - fv3lam_mrms_xzRefl_[date]_[time].png (Cross-sectional reflectivity)
   - fv3lam_mrms_3dRefl_[date]_[time].png (3-D reflectivity)

3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_mrms.py
# Run the script
python3 plot_fv3lam_mrms.py
```

Name	Description
machine	HPC machine for plotting (‘hera’)
out_fig_dir	Path to the directory where output files are created
path_NE	Path to the Natural Earth data
s_date	Date of the input data (YYYYMMDD)
s_time	Time of the input data (HHmm)
dnm_data	Path to the directory where the input GRIB2 file is located
fnm_in_com	Input file name of composite reflectivity
fnm_in_3d	Input file name of 3-D reflectivity
plt_com	Flag for plotting composite reflectivity (‘yes’ or ‘no’)

plt_crx	Flag for plotting cross-sectional reflectivity (‘yes’ or ‘no’)
crx_lon / crx_lat	Longitude/latitude of the target point located at the center of the cross sections (in degrees)
crx_dist	Distance from the target point for the limits of the cross sections (in degrees)
plt_3d	Flag for plotting 3-dimensional reflectivity (‘yes’ or ‘no’)
out_title_com	Title of the output figure of the composite reflectivity
out_fname_com	Output file name of the composite reflectivity
out_fname_crx	Output file name of the cross-sectional reflectivity
out_fname_3d	Output file name of the 3-dimensional reflectivity
back_res	Background resolution (‘10m’, ‘50m’, ‘100m’)
back_img	Background image (‘on’, ‘off’)

Table 11.19: Namelist in the script for plotting ‘mrms_rada_reflectivity.grib2’.

4. Output:

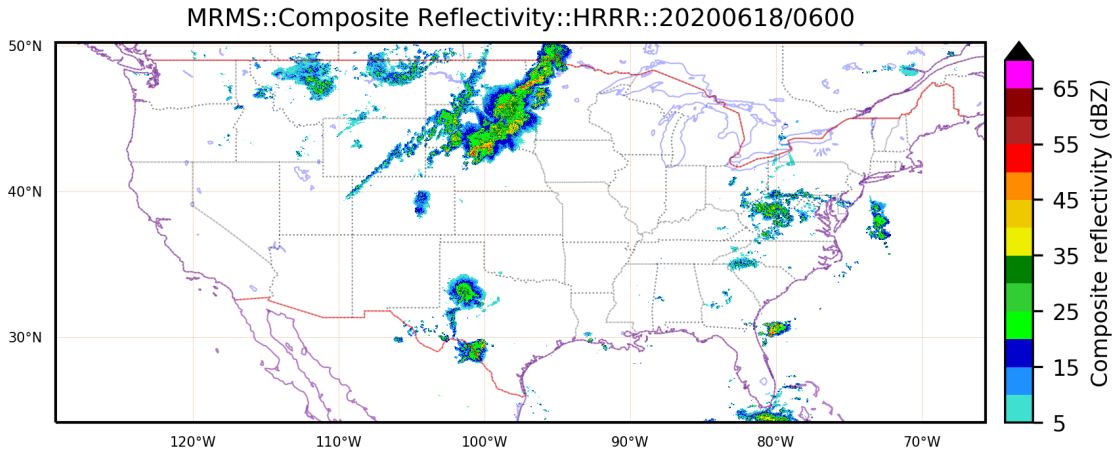

Figure 11.27: MRMS radar: composite reflectivity

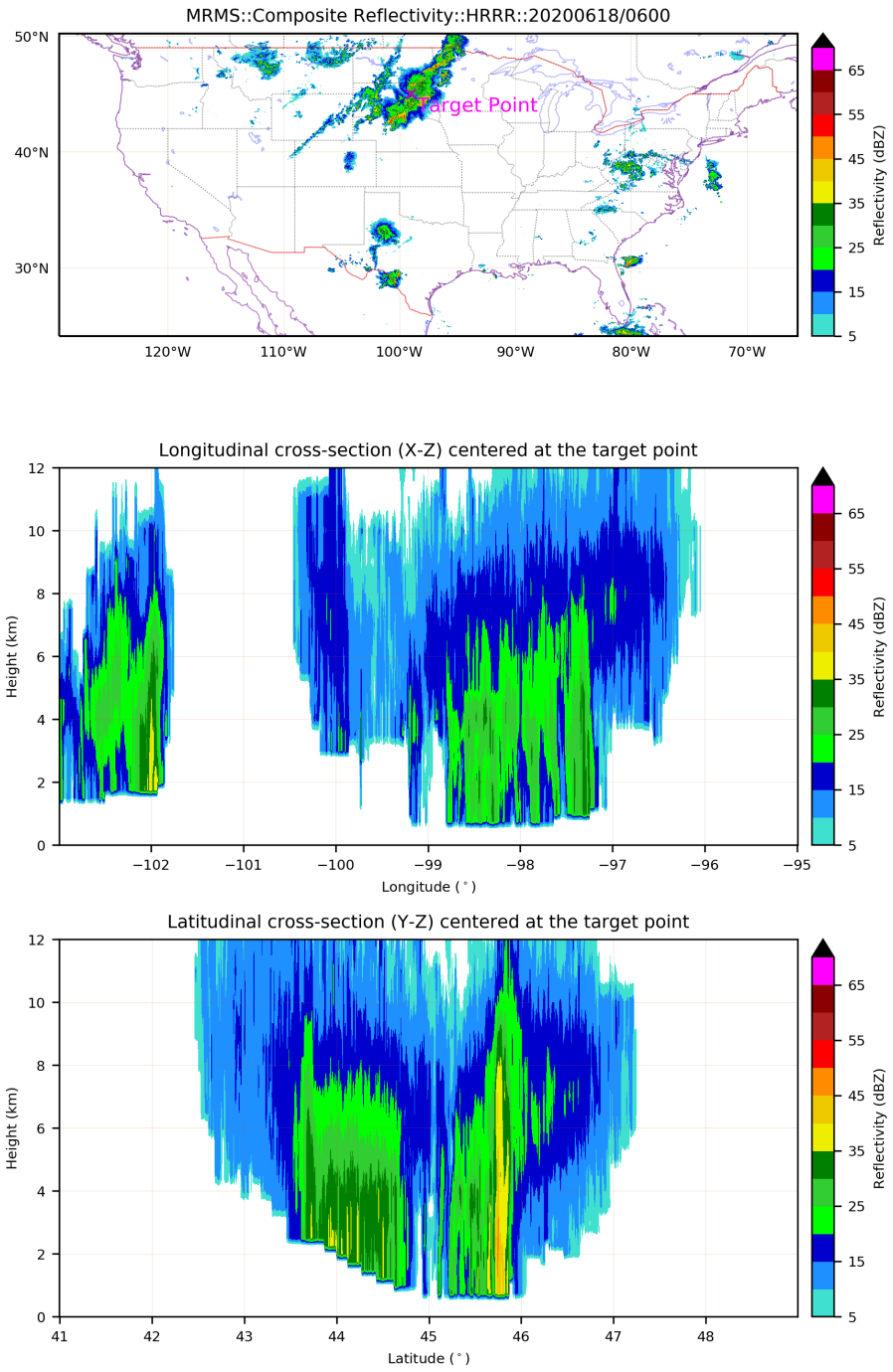

Figure 11.28: MRMS radar: cross-sectional reflectivity

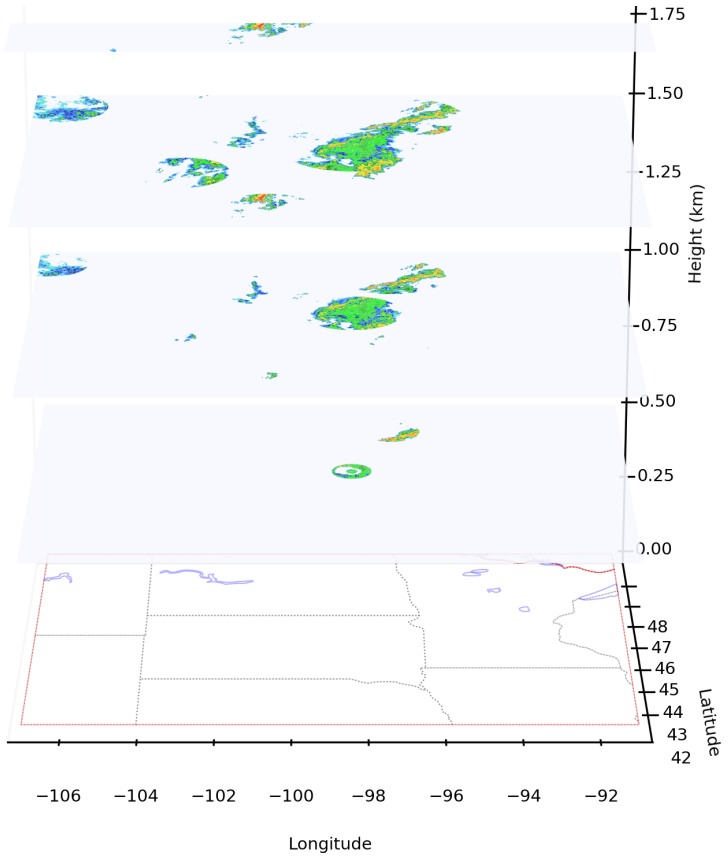

Figure 11.29: MRMS radar: 3-dimensional reflectivity

11.2.21 Animation: Hourly Comparison of Composite Reflectivity

1. Input files

   - QCComposite_00.50_[date]_[time]_new.grib2 (radar data files)
   - rrfs.tXXz.bgdawpf0XX.tmXX.grib2 (FV3-LAM result files)

   Notes)

   - The MRMS radar data can be obtained from HPSS as shown in Section B.2.
   - The original GRIB2 file should be converted to complex packing by *wgrib2* to avoid an API error using *Python*.

2. Output files

   - fv3lam_out_ani_refl_comp_[date].gif

3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_ani_comref.py
# Run the script
python3 plot_fv3lam_ani_comref.py
```

Name	Description
machine	HPC machine for plotting (‘hera’)
out_fig_dir	Path to the directory where output files are created
path_NE	Path to the Natural Earth data
s_date	Date in the data directory name (YYYYMMDD)
plot_hrs	Plotting hours for animation
s_time	Time of the input data (HHmm)
dnm_data_mdl	Path to the directory where the output GRIB2 files are located
dnm_data_dat	Path to the directory where the radar data are located
svar	Output field name
out_title_sub1	Sub-title of the sub-plot 1 for output data
out_title_sub2	Sub-title of the sub-plot 2 for radar data
back_res	Background resolution (‘10m’, ‘50m’, ‘100m’)
back_img	Background image (‘on’, ‘off’)

Table 11.20: Namelist in the script for animation comparing output data.

4. Output:

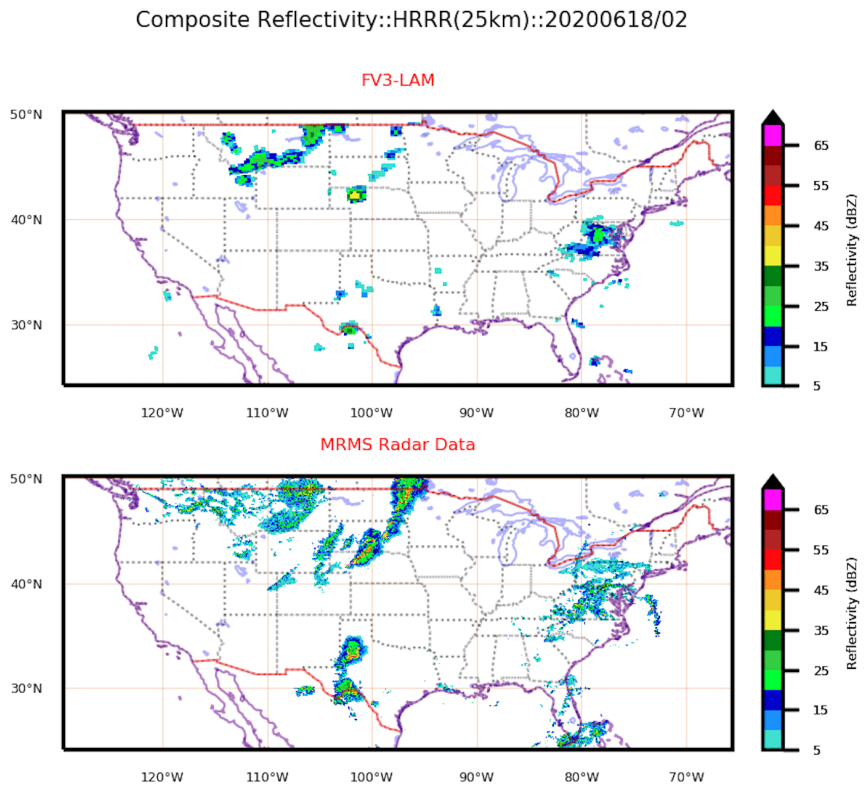

Figure 11.30: Animation: hourly comparison with the MRMS rada data

11.2.22 Time-history of Maximum Temperature

This script is designed to confirm that the simulation results have been exported to the GRIB2 files successfully and the results on several machines are reasonably close to each other by plotting the maximum values of temperatures at 2 and 100 meters from the multiple result GRIB2 files.

1. Input files
   - rrfs.tXXz.bgdawpf0XX.tmXX.grib2 (FV3-LAM result files)
2. Output files
   - fv3lam_out_grb2his_[machine].gif
3. Run on HPC

```
# Check Input and Output paths and names
vim plot_fv3lam_grb2his.py
# Run the script
python3 plot_fv3lam_grb2his.py
```

Name	Description
machine	HPC machine for plotting (‘hera’, ‘wcoss_dell’, ‘wcoss_cray’, ‘orion’)
out_fig_dir	Path to the directory where output files are created
dnm_data	Path to the directory where the output GRIB2 files are located
fnm_hr_s	Input file hour range: start
fnm_hr_e	Input file hour range: end
fnm_hr_i	Input file hour range: interval
out_title	Title of the output figure
out_fname	File name of the output figure

Table 11.21: Namelist in the script for time-history of output grib2 files.

4. Output:

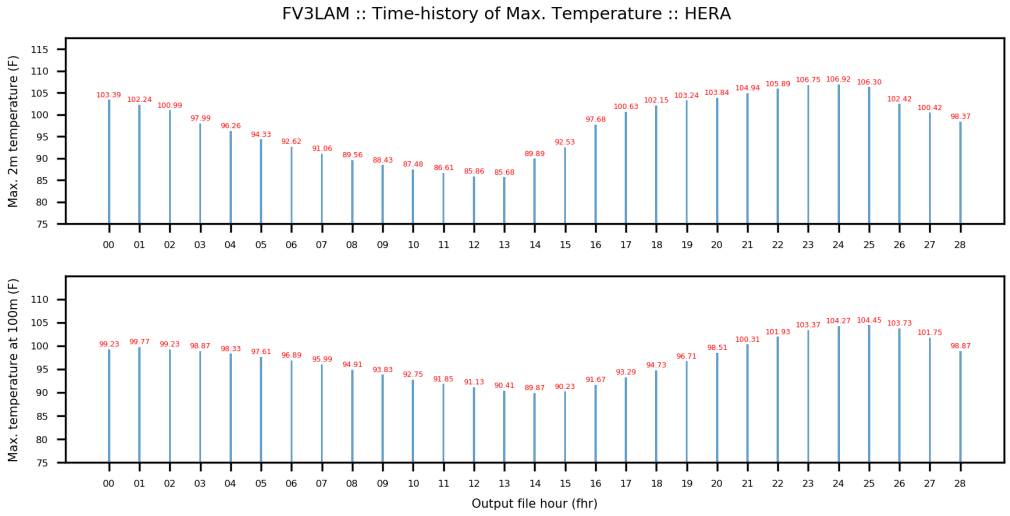

Figure 11.31: Time-history of output GRIB2 files

Appendix A

Workflow Manager: *Rocoto*

Ref.) https://github.com/christopherwharrop/rocoto/wiki/documentation

A.1 How the *Rocoto* Engine Works

Rocoto:

- is a 'Ruby' program that interfaces to the underlying batch system.
- is designed to run weather and climate workflows (not general purpose).
- does all the book keeping necessary to submit tasks when dependencies are satisfied and tracks the progress of the workflow.
- automatically resubmit failed tasks up to a maximum number of attempts, and can recover from system outages.
- has no control over when jobs start to run.
- runs one instance of the workflow for a set of user-defined 'circles'. A 'circle' usually corresponds to a model analysis or initialization time.
- users must define their workflows with the XML language.

1. Executing workflows:
    - The 'rocotorun' command is used to run a workflow:

```
rocotorun -w /path/to/workflow/xml/file -d /path/to/workflow/database/
    file
```

where

- The '-w' flag specifies the name of the workflow definition file. This must be an XML file.
- The '-d' flag specifies the name of the database file that is to be used to store the state of the workflow. The database file is a binary file created and used only by *Rocoto* and need not exit prior to the first time the command is run.

- It is very important to understand that the running process is iterative. Each time the above command is executed, *Rocoto* performs the following actions:
  (a) Read the last known state of the workflow from the database file specified by the '-d' flag.
  (b) Query the batch system to acquire the current state of the workflow.
  (c) Take actions based on new state information (resubmit crushed jobs or submit next jobs)
  (d) Save the current state of the workflow to the file specified by the '-d' flag
  (e) Quit

2. Checking the status of workflows

- 'rocotostat': query the status of a set of cycles and tasks.

```
rocotostat -w /path/to/workflow/xml/file -d /path/to/workflow/database/
    file [-c YYYYMMDDHHMM, [YYYYMMDDHHMM, ...]] [-t taskname, [taskname,
     ...]] [-s] [-T]
```

  - The '-c' option allows you to select specific cycles to query.
  - The '-t' option allows you to select specify tasks.
  - The '-T' option sorts the output by taskname rather than by cycle.
  - The '-s' option will provide a summary report of the status of the cycles themselves rather than information about the tasks.

- 'rocotocheck': query detailed information about a specific task for a specific cycle.

```
rocotockeck -w /path/to/workflow/xml/file -d /path/to/workflow/database/
    file -c YYYYMMDDHHMM -t taskname
```

3. Forcing tasks to run

   - 'rocotoboot': force a task to be submitted, regardless of whether or not dependencies are satisfied or throttling violations would occur.

```
rocotoboot -w /path/to/workflow/xml/file -d /path/to/workflow/database/
    file -c YYYYMMDDHHMM -t taskname
```

4. Rerunning several tasks in the workflow

   - 'rocotoewind': undo the effect of having run some jobs, and mark those jobs as not having run.

```
rocotoewind -w /path/to/workflow/xml/file -d /path/to/workflow/database/
    file -c YYYYMMDDHHMM -t taskname
```

A.2 *Rocoto* XML Language

1. XML Header

   Every XML file must have this text at the top:

```
<?xml version="1.0"?>
<!DOCTYPE workflow
[
]>
```

2. ENTITY

   An 'ENTITY' represents a value that is used in lots of places in the XML.

```
<!ENTITY CYCLE_THROTTLE "4">
<!ENTITY GTYPE "@[GTYPE:-regional]">
```

   The 'ENTITY' is referenced by the systex, '*&{ENTITY_NAME}*;'.

```
<!ENTITY HOME_DIR = "chan-hoo.jeon/fv3sar/">
<!ENTITY LOG = "$HOME_DIR;/log">
```

3. Tag: <workflow>

   This is the main tag for defining workflows. Everything except for the header and ENTITY definitions must be contained within the <workflow> tags.

```
<workflow realtime="F" scheduler="moabtorque" cyclelifespan="0:01:00:00"
    cyclethrottle="5" taskthrottle="5">

# Everything else goes in here

</workflow>
```

   (a) **realtime** : define whether or not the workflow is to be run in realtime ("T" or "True"), or in retrospective mode ("F" or "False"). The difference between realtime and retrospective mode is in how cycles are activated. In realtime mode, cycles are activated based on the current time of day. In retrospective mode, there is no time dependency that must be dhered to. Thus cycles will be activated immediately, in chronological order.

   (b) **scheduler** : must be set to one of "sge", "lsf" (for WCOSS), "torque", "moabtorque" (for Jet), or "moab" (for Gaea). This attribute tells *Rocoto* which batch system to use when managing the workflow. It is recommended that the keep time be set to a minimum of 24 hours.

   (c) **cyclelifespan** : specifies how long a cycle can be active before it expires in the format of "DD:HH:MM:SS". Wall-clock-time requests are automatically capped such that they will not exceed the time when a cycle expires.

   (d) **cyclethrottle** : limits how many cycles may be active at one time. Cycles are active if they have not expired and not all tasks for the cycle have completed successfully.

   (e) **corethrottle** : limits the total number of cores that may be consumed by jobs submitted to the batch system. Any job that is submitted to the batch system, whether it is running or queued, counts against this total.

   (f) **taskthrottle** : limits how many tasks may be active at one time. A task is active if a job has been submitted for it and that job has not finished. Any job that is submitted to the batch system, whether it is running or queued, counts against this total.

4. Tag: <log>

   This defines the path and name of *Rocoto* log files. In general, it is best to define the name of the log to be dynamically dependent on the cycle being processed.

```
<log><cyclestr>&LOG;/workflow_@Y@m@d@H.log</cyclestr></log>
```

5. Tag: <cyclestr>

   It is often necessary to refer to the various components of the current cycle time when defining various aspects of the workflow. The <cyclestr> tag uses flags to represent the various time components of the current cycle. These flags will be replaced with the appropriate date/time component of the current cycle being processed. They can be used in any combination to represent any date/time string desired inside a <cyclestr> ... </cyclestr> block.

   - **offset**: represents offset before or after the current cycle. The format of the offset attribute is 'dd:hh:mm:ss'. Leading field whose values are 0 do not need to be specified.

```
<cyclestr offset="1:00:00">@Y@m@d@H.log</cyclestr>
```

6. Tag: <cycledef>

   The <cycledef> tag defines the set of cycles the workflow is to be run on. *Rocoto* uses the union of all sets of cycles specified by the <cycledef> tags to create the overall cycle pool. If there is overlap between definitions, it will not cause cycles to be run twice. There are two methods to specify cycles. You can mix and match the two different ways to specify cycles as needed.

   (a) **The start-stop-step method**: Specify a start cycle, an end cycle, and an increment. The format of the start and stop cycles is 'yyyymmddhhmm'. The format of the increment is 'dd:hh:mm:ss'.

```
<cycledef>200607060900 201304230900 06:00:00</cycledef>
```

   (b) **The crontab-like method** : Uses a crontab-like format to represent groups of cycles. Six fields such as minute, hour, day, month, year, and weekday must be defined. Each field can be a single value, a range of values, a comma separated list of values, etc.

Table A.1: <cyclestr> tags:

Flag	Time component
@a	Abbreviated weekday name (e.g. “Sun”)
@A	Full weekday name (e.g. “Sunday”)
@b	Abbreviated month name (e.g. “Jan”)
@B	Full month name (e.g. “January”)
@c	Preferred local date and time representation (e.g. “Thu Jul 5 11:27:59 2012”)
@d	Day of month (01–31)
@H	Hour of the day, 24 hour clock (00–24)
@I	Hour of the day, 12 hour clock (01–12)
@j	Day of the year (001–365)
@m	Month of the year (01–12)
@M	Minute of the hour (00–59)
@p	Meridian indicator (“AM” or “PM”)
@P	Meridian indicator (“am” or “pm”)
@s	Number of seconds since January 1, 1970 00:00:00 UTC
@S	Second of the minute (00–59)
@U	Week number of the current year, starting with the first Sunday as the first day
@W	Week number of the current year, starting with the first Monday as the first day
@w	Day of week (Sunday is 0, 0–6)
@x	Preferred representation for the date alone (“07/05/12”)
@X	Preferred representation for the time alone (“11:34:58”)
@y	Year without century (00–99)
@Y	Year with century
@Z	Time zone name

- **group**: Defines distinct sets of cycles because some tasks should only be run for certain subsets of cycles. Multiple <cycledef> tags may be assigned to the same group, but a <cycledef> tag may not be assigned to more than one group.

```
<cycledef group="15min">* /15 * * * 2006-2013 *></cycledef>
```

where an asterisk (*) denotes a shorthand for all values of the field. The above example defines a set of cycles consisting of every 15 minutes of every hour for every day and

every month of the years 2006–2013.

7. Tag: <task>

   It defines the computations that you want to run. Every task must have a unique name defined.

   (a) **cycledefs** : (optional). If set, its value must be a comma separated list of <cycledef> tag group names. If it is not set, the task will be run for every cycle in the general pool of cycles.

   (b) **maxtries** : (optional). When *Rocoto* detects that a task has failed, it will attempt to resubmit it. The maxtries attribute limits the number of times a task can be retried.

   (c) **throttle** : (version 1.1 and higher, optional). If set, its value must be a positive integer. The throttle limits the number of instances of the task which may be queued or running at any one time. There is one instance of the task per cycle. Therefore, the task throttle limits the number of cycles for which the task may be active at any one time.

```
<task name="wrf" cycledefs="3hourly,␣6hrlyJanFeb" maxtries="3">

</task>
```

8. Tag: <command>

   Every <task> tag must contain a <command> tag. This is the command that carries out the task's work and is the command that *Rocoto* will submit to the batch system for execution.

```
<command><cyclestr>&SCRIPTS;/wrf.ksh -c @Y@m@d@H</cyclestr></command>
```

9. Tag: <account>

   Every <task> tag should contain an <account> tag. It is optional to allow for situations where a default may be used. This defines the batch system account/project.

```
<account>&ACCOUNT</account>
```

10. Tag: <queue>

    Every <task> tag should contain a <queue> tag. This defines the batch system queue that *Rocoto* will submit the task to for execution.

```
<queue>&QUEUE_PE</queue>
```

11. Tag: <cores>

    Every <task> tag must contain either one <cores> tag or one <nodes> tag. The <cores> tag defines the number of cores that *Rocoto* will request when submitting the task for execution.

```
<cores>1</cores>
```

12. Tag: <nodes> (version 1.1 and higher)

    The <nodes> tag defines the task geometry (a list of node counts and cores per node). The <nodes> tag should only be used if the task has a requirement to use less than all of the available cores on at least one of its nodes. The <nodes> tag is primarily intended to support OpenMP/MPI hybrid codes, and MPMD codes.

```
<nodes>1:ppn=1+10:ppn12</nodes>
```

    This example requests 1 core on the first node and 12 cores on each of the next 10 nodes.

13. Tag: <walltime>

    Every <task> tag must contain a <walltime> tag. This defines the amount of wall-clock time. The requested walltime is automatically reduced so as not to exceed the amount of time remaining before the task expires.

```
<walltime>00:00:10</walltime>
```

14. Tag: <memory>

    The <memory> tag is usually only needed for serial tasks. This defines the amount of memory.

```
<memory>512M</memory>
```

15. Tag: <jobname>

    Every <task> tag should contain a <jobname> tag because it helps in visual tracking of jobs in queue status outputs. It is optional to allow for situations where a default may be used.

```
<jobname>test</jobname>
```

16. Tags: <stdout>, <stderr>, and <join>

    Every <task> tag should contain either both <stdout> and <stderr> tags or a <join> tag because it helps in tracking and finding the output of the jobs. These tags define the location of the stdout and stderr output of the job that executes the task. The <join> tag is used when you want both stdout and stderr to go to the same place. **DO NOT** set the value of <stdout> and <stderr> to the **same** thing. Instead, use **<join>** to do that.

```
<join>/home/test/test.log</join>

or

<stdout>/home/test/test.out</stdout>
<stderr>/home/test/test.err</stderr>
```

17. Tag: <envar>

    The <envar> tag is used inside the <task> tags to define environment variables. It consists of (name,value) pairs. All <envar> tags must contain a <name> tag. However, <value> tags can be absent in cases where a variable needs to be set, but does not need to be assigned a value.

```
<envar>
   <name>ANALYSIS_TIME</name>
   <value><cyclestr>@Y@m@d@H@M</cyclestr></value>
</envar>
```

18. Tag: <dependency>

    The <dependency> tags are located inside <task> tags, and used to describe the inter-dependencies of the tasks. Dependencies are defined as boolean expressions. There are three types of dependencies: (1) task dependency, (2) data dependency, and (3) time dependency. They can be combined in any combination in a boolean expression to form a task's overall dependencies. There must be exactly **one** tag of the following tags inside the <dependency> tags:

```
<dependency>
        <taskdep>
        <datadep>
        <timedep>
```

```
        w/ boolean operator tag (<and>, <or>, <not>, <nand>, <nor>, <xor>, <
    some>)
</dependency>
```

(a) Tag: <taskdep>

The task attribute of the <taskdep> tag must be set to the name of the task which must complete in order for the dependency to be satisfied. The task attribute must refer to a task that has already been defined in the XML document.

- **cycle_offset** : A task may have a dependency on a task from a different cycle. The 'cycle_offset' attribute allows users to specify an offset from the current cycle to define inter cycle dependencies. The offset can be positive or negative.
- **state** : The 'state' attribute is optional. It allows you to specify whether you want the dependency to be satisfied when a task has completed successfully, or when a task has failed and exhausted retries. It may be set to either "Succeeded" (default) or "Dead".

```
  <taskdep state="Dead" task="wrfpost_f006" cycle_offset="-6:00:00"/>
```

(b) Tag: <datadep>

Data dependencies are defined inside <dependency> tags. File dependencies are satisfied when the file in question exists, has not been modified for at least the amount of time specified by 'age', and is at least as large as the size specified by 'minsize'.

- **age** : (optional). It contains the time (dd:hh:mm:ss) that the file must not be modified before the file is considered to be available. The default is zero.
- **minsize** : (optional). It contains the minimum size for the file before it is considered to be available. The default value is zero. B(or b): byte (default), K(or k): kilobyte, M (or m): megabyte, G (or g): gigabyte.

```
  <datadep age="00:02:00" minsize="1024b" /datadep>
```

(c) Tag: <timedep>

The value between the start and end of the <timedep> tag is a time in 'yyyymmddhhmmss' format. Time dependencies are satisfied with the wall clock time is equal to or greater than the time specified. All times are calculated in GMT.

```
<timedep><cyclestr offset="&DEADLINE;">@Y@m@d@H@M@S</cyclestr> </
 timedep>
```

(d) The boolean operator tags

There are several boolean operators that may be used to compose boolean expressions of dependencies. The operators and the operands (<taskdep>, <datadep>, <timedep>) can be combined without limit.

Table A.2: Boolean operator tags:

Flag	Time component
<and>	Satisfied if all enclosed dependencies are satisfied
<or>	Satisfied if at least one of the enclosed dependencies is satisfied
<not>	Satisfied if the enclosed dependency is not satisfied
<nand>	Satisfied if at least one of the enclosed dependencies is not satisfied
<nor>	Satisfied if none of the enclosed dependencies are satisfied
<xor>	Satisfied if exactly one of the enclosed dependencies is satisfied
<some>	Satisfied if the fraction of enclosed dependencies that are satisfied exceeds some threshold ($0 \leq$ threshold ≤ 1)

Appendix B

Data Transfer from/to/between NOAA HPCs

B.1 Trusted Data Transfer Node: Between NOAA HPCs

- DTN is only accessible from some hosts **within .noaa.gov**.
- The DTN supports ssh-based authentication transfer methods, which currently include scp, rsync, and bbcp (Jet and Hera only).
  - Hera: dtn-hera.fairmont.rdhpcs.noaa.gov
  - Niagara: dtn-niagara.fairmont.rdhpcs.noaa.gov
  - Jet: dtn-jet.rdhpcs.fairmont.noaa.gov
- Default authentication uses your RSA token.
- Username is case sensitive in the scp command.
- Only the high-performance filesystems (**scratch**) are available, **NOT** your **/home** filesystem.

B.1.1 On Hera

1. Hera to Jet:

```
cd [path to the parent directory of the file/directory on Hera]
scp [file / -r directory] Chan-hoo.Jeon@dtn-jet.rdhpcs.noaa.gov:/mnt/lfs1/
    projects/jetmgmt/Chan-hoo.Jeon/
```

B.1.2 On WCOSS

1. WCOSS to Hera:
   SCP transfer to Hera can be accomplished via the MARS/VENUS access nodes:

```
cd [path to the parent directory on WCOSS]
scp [file / -r directory] Chan-hoo.Jeon@dtn-hera.fairmont.rdhpcs.noaa.gov:/
    scratch2/NCEPDEV/stmp1/Chan-hoo.Jeon/
```

2. Hera to WCOSS:

```
cd [path to the parent directory on WCOSS]
scp (-r: only for directory) Chan-hoo.Jeon@dtn-hera.fairmont.rdhpcs.noaa.gov
    :/scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/(file or directory) .
```

B.1.3 On Orion

1. Hera to Orion:

```
cd [path to the parent directory on Orion]
scp (-r : only for directory) Chan-hoo.Jeon@dtn-hera.fairmont.rdhpcs.noaa.
    gov:/scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/[file or directory] .
```

2. Orion to Hera:

```
cd [path to the parent directory on Orion]
scp [file / -r directory] Chan-hoo.Jeon@dtn-hera.fairmont.rdhpcs.noaa.gov:/
    scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/
```

B.2 Data Transfer from HPSS

The centralized, long-term data archive system at NESCC is based on the High Performance Storage System (HPSS).
On HPSS:

- MRMS (Multi-Radar/Multi Sensor) radar data (reflectivity):

```
/BMC/fdr/Permanent/{YYYY}/{MM}/{DD}/data/radar/mrms/{YYYYMMDDHH}00.zip
```

– Each zip file contains 180 zip files for every minute for the next three hours and three gauge zip files. For example, the '202006180000.zip' file contains '202006180000.zip' to '202006180259.zip'. The last two digits denote minutes.

```
202006180000.zip
├── 202006180000.zip
│   ├── MRMS_MergedReflectivityQCComposite_00.50_[*].grib2
│   ├── MRMS_MergedReflectivityQC_{AA.BB}_[*].grib2
│   └── MRMS_Reflectivity_{XX}C_00.50_[*].grib2
├── 202006180001.zip
├──     ⋮
├── 202006180258.zip
├── 202006180259.zip
├── 202006180000.Gauge.zip
├── 202006180100.Gauge.zip
└── 202006180200.Gauge.zip
```

Figure B.1: Structure of an MRMS-data zip file

– Data format: 'MRMS_MergedReflectivityQC_{AA.BB}_{YYYYMMDD}-{HHmmSS}.grib2' where {AA.BB} denotes the height of the field in kilometers.

Note) To avoid the 'GRIB_API ERROR:' with *Python*, the original GRIB2 files should be converted to new GRIB2 files using complex packing by *wgrib2*.

- 3 hourly CCPA (Climatology-Calibrated Precipitation Analysis) data (total precipitation):

```
/NCEPPROD/hpssprod/runhistory/rh{YYYY}/{YYYYMM}/{YYYYMMDD}/
   com_ccpa_prod_ccpa.{YYYYMMDD}.tar
```

Note) There is a bug in the hourly CCPA data. The time stamp is wrong for the '00Z' directory. I typically use the 3 hourly data and set my fhzero to 3 hours.

B.2.1 On Hera

1. Load the HPSS module:

```
module load hpss
```

2. Search for files on HPSS:

```
hsi
cd /BMC/fdr/Permanent/2020/06/18/data/radar/
cd /NCEPPROD/hpssprod/runhistory/rh2020/202006/20200618/
# Check if the files you want exist
exit
```

3. Pull data from HPSS:

   - MRMS radar data:

```
cd /scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/00_DATA/MRMS
mkdir mrms_2020061806
cd mrms_2020061806
hsi get /BMC/fdr/Permanent/2020/06/18/data/radar/mrms/202006180600.zip
mv 202006180600.zip 2020061806.zip
unzip -l  2020061806.zip. (list of files)
unzip 2020061806.zip 202006180600.zip
unzip 202006180600.zip
```

   (a) Sample script (‘/home/Chan-hoo.Jeon/tools/plot_refl/00_make_reflc_files.ksh’):

```
#!/bin/bash

module purge
module load hpss
module list

set -x

cycle=2019070100

case="mrms_${cycle}"

path="/scratch2/NCEPDEV/fv3-cam/$LOGNAME/00_DATA/MRMS/${case}"

if [ ! -d ${path} ];then
  mkdir -p ${path}
fi
cd ${path}

yr=`echo $cycle | cut -c1-4`
mo=`echo $cycle | cut -c5-6`
dy=`echo $cycle | cut -c7-8`
hr=`echo $cycle | cut -c9-10`

echo

hrlist=("03" "06")
#hrlist=("06")
for i in "${hrlist[@]}"
```

```
do
  fname0="${yr}${mo}${dy}${i}"
  fname1="${yr}${mo}${dy}${i}00"

  hsi get /BMC/fdr/Permanent/${yr}/${mo}/${dy}/data/radar/mrms/${fname1}.zip

  sleep 10

  mv ${fname1}.zip ${fname0}.zip

  unzip ${fname0}.zip ${fname1}.zip
  unzip ${fname1}.zip -d ${fname1}
  cd ${fname1}
  cat *MRMS_MergedReflectivityQC_00.50_* *MRMS_MergedReflectivityQC_00.75_* *MRMS_MergedReflectivityQC_01.00_* *
      MRMS_MergedReflectivityQC_01.25_* *MRMS_MergedReflectivityQC_01.50_* *MRMS_MergedReflectivityQC_01.75_* *
      MRMS_MergedReflectivityQC_02.00_* *MRMS_MergedReflectivityQC_02.25_* *MRMS_MergedReflectivityQC_02.50_* *
      MRMS_MergedReflectivityQC_02.75_* *MRMS_MergedReflectivityQC_03.00_* *MRMS_MergedReflectivityQC_03.50_* *
      MRMS_MergedReflectivityQC_04.00_* *MRMS_MergedReflectivityQC_04.50_* *MRMS_MergedReflectivityQC_05.00_* *
      MRMS_MergedReflectivityQC_05.50_* *MRMS_MergedReflectivityQC_06.00_* *MRMS_MergedReflectivityQC_06.50_* *
      MRMS_MergedReflectivityQC_07.00_* *MRMS_MergedReflectivityQC_07.50_* *MRMS_MergedReflectivityQC_08.00_* *
      MRMS_MergedReflectivityQC_08.50_* *MRMS_MergedReflectivityQC_09.00_* *MRMS_MergedReflectivityQC_10.00_* *
      MRMS_MergedReflectivityQC_11.00_* *MRMS_MergedReflectivityQC_12.00_* > ../ref3D_${fname1}.grib2

  cat *MRMS_MergedReflectivityQCComposite_00.50* *MRMS_Reflectivity_-15C* *MRMS_Reflectivity_-10C* *MRMS_Reflectivity_-5C*
      *MRMS_Reflectivity_0C* > ../ref2D_${fname1}.grib2

  cp *MRMS_MergedReflectivityQCComposite_00.50* ../QCComposite_00.50_${fname1}.grib2
  cp *MRMS_SeamlessHSR_* ../seamlessHSR_${fname1}.grib2
  cp *MRMS_PrecipRate_00.00* ../prate_${fname1}.grib2

  cd ..
  rm -rf ${fname1}
  rm -f ${fname1}.zip
done

exit
```

(b) Submit the sample script ('/home/Chan-hoo.Jeon/tools/plot_refl/01_submit_job'):

```
#!/bin/bash --login

#SBATCH -A fv3-cam
#SBATCH --job-name=heratransfer_1
#SBATCH --partition=service
#SBATCH --time=00:20:00
##SBATCH --nodes=1
#SBATCH -n 1

srun 00_make_reflc_files.ksh
```

To run the script

```
cd /home/Chan-hoo.Jeon/tools/plot_refl/
sbatch 01_submit_job
```

- CCPA data:

```
cd /scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/00_DATA/MRMS
mkdir ccpa_20200618
```

```
cd ccpa_20200618
htar -xvf /NCEPPROD/hpssprod/runhistory/rh2020/202006/20200618/
    com_ccpa_prod_ccpa.20200618.tar
```

4. Convert to new GRIB2 file using complex packing:

```
module load intel
module load wgrib2
wgrib2 input.grib2 -set_grib_type c3(or complex3) -grib_out output.grib2
```

Sample script:

```
#!/bin/bash

module purge
module load intel
module load wgrib2
module list

cycle=2019070100

case="mrms_${cycle}"
path="/scratch2/NCEPDEV/fv3-cam/$LOGNAME/00_DATA/MRMS/${case}"
path_sub="$path/new_grib2"

if [ ! -d ${path} ];then
  mkdir -p ${path}
fi
if [ ! -d ${path_sub} ];then
  mkdir -p ${path_sub}
fi

yr=`echo $cycle | cut -c1-4`
mo=`echo $cycle | cut -c5-6`
dy=`echo $cycle | cut -c7-8`
hr=`echo $cycle | cut -c9-10`

hrlist=("03" "06")
#hrlist=("06")
for i in "${hrlist[@]}"
do
  fname0="${yr}${mo}${dy}${i}00"

  fn1="QCComposite_00.50_${fname0}"
  fn2="prate_${fname0}"
  fn3="seamlessHSR_${fname0}"
  fn4="ref2D_${fname0}"
  fn5="ref3D_${fname0}"

  fnlist=("$fn1" "$fn2" "$fn3" "$fn4" "$fn5")

  for j in "${fnlist[@]}"
  do
     wgrib2 "${path}/${j}.grib2" -set_grib_type c3 -grib_out "${path_sub}/${j}_new.grib2"
  done
done
```

B.3 External Data Transfer from Hera to Local Desktop

1. Transfer data to the 'eDTN' (on Window #1, Hera)

```
cd [path to the files]
tar -cvzf [zip file name: fv3.tar.gz] [file names: fv3_isfc_*]
scp [zip file name] Chan-hoo.Jeon@edtn.fairmont.rdhpcs.noaa.gov:
```

2. Transfer data from 'eDTN' to a local desktop (on Window #2, local)

```
cd [path to the directory where the file will be downloaded on the local
    desktop]
scp Chan-hoo.Jeon@edtn.fairmont.rdhpcs.noaa.gov:[file name] .
tar -xvzf [zip file name: fv3.tar.gz]
```

B.4 SSH Port Tunnel from Linux-like Systems: Between Desktop and HPC

1. Find local port number
   - Close all current sessions opens on the HPC system.
   - Open a window terminal on your Desktop.
   - Find your unique local port number by logging onto your specified HPC system (Hera/-Jet). It can be found on the terminal windows as

     ```
     Local port 2027 forwarded to remoted host.
     Remote port YYYYY forwarded to local host.
     ```

   - Close all sessions once it's been recorded.
2. Open two terminal windows.
3. On Window #1, enter the following (e.g. local port number=2027):

```
ssh -X -L2027:localhost:2027 Chan-hoo.Jeon@hera-rsa.rdhpcs.noaa.gov
```

4. On Window #2, After the first session has been opened with the port forwarding, then any further connections (login via ssh, copy via scp) will work as expected (**Password: PIN+Token code**).

- To transfer a file (files) to HPC

```
scp -P 2027 /export/emc-lw-chjeon/cjeon/NOAA/file_name.png Chan-hoo.
    Jeon@localhost:/scratch2/NCEPDEV/fv3-cam/Chan-hoo.Jeon/tools/
    fv3sar_pre_plot/Fig/
```

or

```
rsync <put rsync options here> -e 'ssh -l Chan-hoo.Jeon -p 2027' /local/
    path/to/files Chan-hoo.Jeon@localhost:/path/to/files/on/HPC
```

- To transfer a file from HPC

```
scp -P 2027 Chan-hoo.Jeon@localhost:/scratch2/NCEPDEV/fv3-cam/Chan-hoo.
    Jeon/tools/fv3sar_pre_plot/Fig/file_name.png /export/emc-lw-chjeon/
    cjeon/NOAA/Fig/
```

or

```
rsync <put rsync options here> -e 'ssh -l Chan-hoo.Jeon -p 2027' Chan-
    hoo.Jeon@localhost:/path/to/files/on/HPC /local/path/to/files
```

Appendix C

HPC Job Commands

C.1 Hera

C.1.1 Slurm

Command	Description
sbatch <runscript>	Submit a job
scancel <jobID>	Cancel the job from the queue
squeue -u <userID>	Display a listing of the current jobs in the queue
scontrol show job <jobID>	Display detailed information about jobs
squeue -- start	Display a point-in-time estimate of when jobs may start

Table C.1: Slurm commands on Hera

C.1.2 Other Useful Commands

Command	Description
saccount_params	Display information regarding project for a user
shpcrpt -c hera -u <userID>	Display current MTD compute information for all projects for a user
module list	Show loaded modules
module avail	Show all available modules
module load <package>	load the package

module unload <package>	unload the package
module swap <A> <B>	Swap package A for package B
module purge	Unload all loaded modules
module spider	Show all possible modules

Table C.2: Other commands on Hera

C.2 WCOSS

If not loaded, the 'LSF' module should be loaded on WCOSS as follows:

```
module load lsf/10.1
```

C.2.1 LSF

Command	Description
bsub <runscript>	Submit a job
bkill <jobID>	Terminate the job from the queue
bjobs -u <userID>	View all jobs for a specific user
bqueues -u <userID>	View the list of queues available

Table C.3: LSF commands on WCOSS

C.3 Orion

C.3.1 Slurm

Command	Description
sbatch <runscript>	Submit a job
scancel <jobID>	Cancel the job from the queue
squeue -u <userID>	Display a variety of information
squeue -p batch –start	Report the start time of pending jobs
showuserjobs	Display the current node and batch job status broken down into userids

sacct	Display accounting information from the Slurm database
sjstat	Display short summary of running jobs and scheduling pool data
sview	Graphical user interface to get and update state information

Table C.4: Slurm commands on Orion

C.3.2 Other Useful Commands

Command	Description
quota -s	Display user's disk usage
saccount_params	Display information regarding project for a user ('module load contrib noaatools' must be run in advance.)
module list	Show loaded modules
module avail	Show all available modules
module load <package>	load the package
module unload <package>	unload the package
module swap <A> <B>	Swap package A for package B
module purge	Unload all loaded modules
module spider	Show all possible modules

Table C.5: Other commands on Orion

Appendix D

Reference Documentations

Documentation	Web location
UFS SRW users' guide	https://ufs-srweather-app.readthedocs.io/en/latest/
UFS MRW users' guide	https://ufs-mrweather-app.readthedocs.io/en/ufs-v1.0.0
UFS weather model	https://ufs-weather-model.readthedocs.io/en/ufs-v1.0.0
FV3	https://noaa-emc.github.io/FV3_Dycore_ufs-v1.0.0/html/index.html
chgres_cube	https://ufs-utils.readthedocs.io/en/ufs-v1.0.0
CCPP Technical	https://ccpp-techdoc.readthedocs.io/en/v4.0/
CCPP Scientific	https://dtcenter.org/GMTB/v4.0/sci_doc/
UPP	https://upp.readthedocs.io/en/ufs-v1.0.0
Stochastic	https://stochastic-physics.readthedocs.io/en/ufs-v1.0.0
NCEPLIBS	https://github.com/NOAA-EMC/NCEPLIBS/wiki
NCEPLIBS-external	https://github.com/NOAA-EMC/NCEPLIBS-external/wiki
NEMS	https://noaa-emc.github.io/NEMS_doc_ufs-v1.0.0/index.html
ESMF	http://www.earthsystemmodeling.org/esmf_releases/public/ESMF_8_0_0/ESMF_refdoc/
NUOPC layer	http://www.earthsystemmodeling.org/esmf_releases/public/ESMF_8_0_0/NUOPC_refdoc/

Table D.1: Web location of the reference documentations