Defining a Custom Regional Domain & Grid in the UFS SRW App

Gerard Ketefian

CIRES/NOAA/ESRL/GSL/DTC

September 21, 2021

Grid Generation Methods in the UFS SRW App

- Two regional grid generation methods available (via variable `GRID_GEN_METHOD`):
  - `GRID_GEN_METHOD="GFDLgrid"`
  - `GRID_GEN_METHOD="ESGgrid"`

- `GRID_GEN_METHOD="GFDLgrid"`:
  - Defines regional grid with respect to a parent global cubed-sphere grid.
  - Parent global grid covers the globe with 6 tiles.
  - A 7th "tile" is created as a subregion of tile 6; user specifies subregion limits (i.e. starting and ending indices of tile 7 within tile 6).
  - Tile 7 is used as the native FV3LAM grid; tiles 1 through 6 discarded.
  - Can stretch/compress grid via the Schmidt factor (`GFDLgrid_STRETCH_FAC`).
  - Can refine tile 7 with respect to tile 6 by specifying a refinement ratio (`GFDLgrid_REFINE_RATIO`); e.g. a refinement ratio of 3 means 3 grid cells on tile 7 for every one on tile 6.

- `GRID_GEN_METHOD="ESGgrid"`:
  - ESG = Extended Schmidt Gnomonic
  - Developed by Jim Purser of EMC to create more uniform (in terms of cell size) regional grids than those generated via `"GFDLgrid"` method.
  - Extends the Schmidt stretching approach (and takes advantage of the fact that a regional grid does not need to connect to other "tiles" to form a global grid) to generate very uniform regional grids.
  - Most predefined grids in the UFS SRW App are generated using the `"ESGgrid"` method.

Common Grid Configuration Variables

- `GRID_GEN_METHOD`
  - Grid generation method to use to create custom grid.
  - Valid values: `"GFDLgrid" "ESGgrid"`
  - Default value: `""` (i.e. an empty string)

    This forces the user to set a valid `GRID_GEN_METHOD` in `config.sh` if specifying a custom grid.

- `HALO_BLEND`
  - Number of cells to use for “blending” the external solution (obtained from the LBCs) with the internal solution from the FV3LAM dycore.
  - Blending necessary to smooth out waves generated due to mismatch between the external and internal solutions.
  - Default value: `"10"`
  - Cells at which blending occurs are all **within** the boundary of the native grid; they don’t involve the 4 cells outside the boundary where the LBCs are specified (which is a different halo).

"ESGgrid" Configuration Variables

For the variables below to take effect, ensure that:

1. PREDEF_GRID_NAME is not specified in config.sh (will remain set to its default value of a null string in config_defaults.sh).
2. GRID_GEN_METHOD is set to "ESGgrid" in config.sh.

- ESGgrid_LON_CTR, ESGgrid_LAT_CTR
  - Longitude and latitude (in degrees) of center of ESG grid.

- ESGgrid_DELX, ESGgrid_DELY
  - Nominal dimensions (in meters) of an ESG grid cell along the two horizontal directions x and y.
  - Currently, x and y approximately correspond to west-to-east and south-to-north directions, respectively.

- ESGgrid_NX, ESGgrid_NY
  - Number of grid cells in the x and y directions on the ESG grid.

- ESGgrid_WIDE_HALO_WIDTH
  - Width (in units of number of cells) of the halo around the boundary of the regional grid initially generated by the make_grid task (before "shaving" halo down to only 4 halo cells).
  - Default value: "6"
  - No need to change.

Write-Component Grid

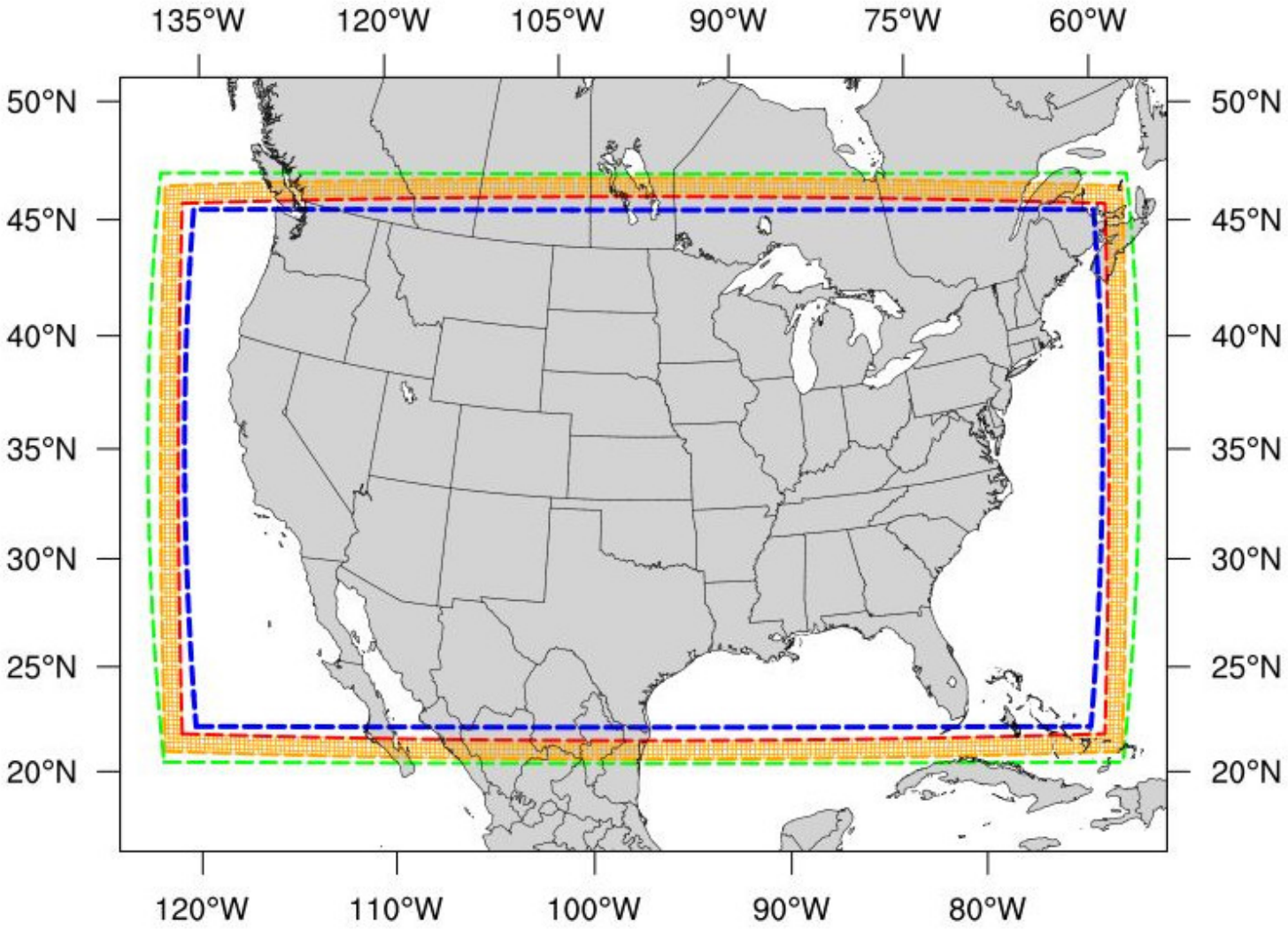
- What is it?
  - The grid to which output fields are interpolated before writing them to file.
  - Interpolation and writing to file performed by the write-component (a component of NEMS (?)).

- Why?
  - UPP (the post-processor called by the `run_post` task) cannot process output on the native grid types (`"GFDLgrid"` and `"ESGgrid"`).
  - The original system used in FV3 to write output to disk (the Flexible Modeling System or FMS) places fields at all output times in a single file ⇒ user cannot access the output (for plotting, etc) before the end of the forecast.

- Write-component offers the following advantages:
  - Interpolates fields to grids that UPP can process.
  - Records fields at each output time in a separate file ⇒ users don't need to wait until end of forecast to view output.
  - Uses a dedicated set of MPI tasks (i.e. separate from those used for the forecast) ⇒ forecast continues while files are being written to disk.

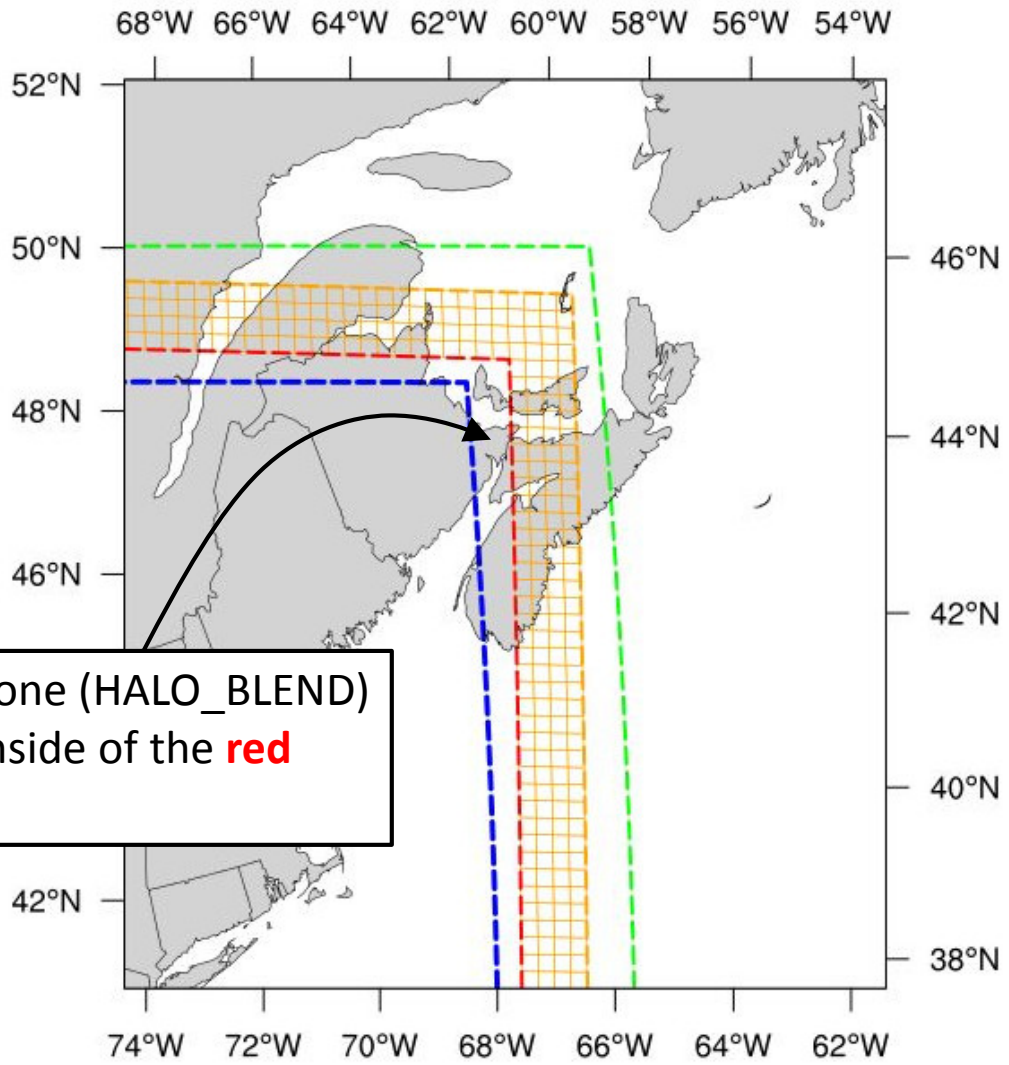
- Currently, three types of write-component grids available in UFS SRW App (specified via `WRTCMP_output_grid`):
  - Ordinary lat-lon grid (`WRTCMP_output_grid="regional_latlon"`)
  - Rotated lat-lon grid (`WRTCMP_output_grid="rotated_latlon"`)
  - Lambert conformal (`WRTCMP_output_grid="lambert_conformal"`)

Write-Component Grid for RRFS_CONUS_25km Grid – Whole Domain



- RED: Boundary of RRFS_CONUS_25km native grid
- BLUE: Boundary of write-component grid corresponding to RRFS_CONUS_25km native grid
- ORANGE: 4-cell-wide halo providing LBCs to RRFS_CONUS_25km native grid
- GREEN: Boundary of external model providing ICs (in this case, HRRR)

Write-Component Grid for RRFS_CONUS_25km Grid – Northeast corner



Blending zone (HALO_BLEND) is on the inside of the **red** boundary.

- RED: Boundary of RRFS_CONUS_25km native grid
- BLUE: Boundary of write-component grid corresponding to RRFS_CONUS_25km native grid
- ORANGE: 4-cell-wide halo providing LBCs to RRFS_CONUS_25km native grid
- GREEN: Boundary of external model providing ICs (in this case, HRRR)

Write-Component Grid Configuration Variables (1 of 2)

- `QUILTING`
  - Flag that specifies whether to use the write-component.
  - Valid values: `"TRUE"` `"FALSE"`
  - Default value: `"TRUE"`
  - If set to `"FALSE"` , the FMS will be used for output (and fields will be on the native grid).

- `WRTCMP_write_groups, WRTCMP_write_tasks_per_group`
  - The number of write groups (i.e. groups of MPI tasks) and the number of MPI tasks per group to use in the write-component.
  - Default values: `"1"` for `WRTCMP_write_groups`, `"20"` for `WRTCMP_write_tasks_per_group`
  - Default values usually sufficient; increase if write-component is getting backed up, e.g. output for a given output forecast hour is waiting for output for the previous output forecast hour to complete (can happen if grid is very large and/or output frequency is very high).

- `WRTCMP_output_grid`
  - The type (coordinate system) of the write-component grid.
  - Valid values: `"rotated_latlon"` `"lambert_conformal"` `"regional_latlon"`
  - Default value: `""` (i.e. empty string)
    This forces the user to set a valid valid `WRTCMP_output_grid` in `config.sh` if specifying a custom grid.

- `WRTCMP_cen_lon`, `WRTCMP_cen_lat`
  - Longitude and latitude (in degrees) of the center of the write-component grid.
  - Can usually set to corresponding values for native grid.

Write-Component Grid Configuration Variables (2 of 2)

- WRTCMP_lon_lwr_left, WRTCMP_lat_lwr_left
  - Longitude and latitude (in degrees) of the center of the lower-left (southwest) cell on the write-component grid. If using "rotated_latlon" coordinate system, this is expressed in terms of the rotated longitude and latitude.
  - Currently, must be set manually; working on automating this (as well as other write-component parameters).

- Additional variables needed if WRTCMP_output_grid has been set to "rotated_latlon":
  - WRTCMP_lon_upr_rght, WRTCMP_lat_upr_rght
    - Longitude and latitude (in degrees) of the center of the upper-left (northeast) cell on the write-component grid (expressed in terms of the rotated longitude and latitude).
  - WRTCMP_dlon, WRTCMP_dlat
    - Size of grid cell (in degrees) of a grid cell on the write-component grid (expressed in terms of the rotated longitude and latitude).

- Additional variables needed if using WRTCMP_output_grid set to "lambert_conformal":
  - WRTCMP_stdlat1, WRTCMP_stdlat2
    - First and second standard latitudes (in degrees) in definition of Lambert conformal projection.
  - WRTCMP_nx, WRTCMP_ny
    - Number of grid points in the x and y coordinates of the Lambert conformal projection.
  - WRTCMP_dx, WRTCMP_dy
    - Grid cell size (in meters) along the x and y directions of the Lambert conformal projection.

Computational Configuration Variables Related to Grid

When specifying a custom grid, will also need to specify the following computational parameters:

- DT_ATMOS
  - The main time step (in seconds) used by the forecast model. Most physics is called on this time step.
- LAYOUT_X, LAYOUT_Y
  - MPI process layout in the forecast model.
- BLOCKSIZE
  - The amount of data that is passed into the cache at a time.

Example of Specifying Custom ESGgrid in `config.sh`

Native ESGgrid parameters

```
GRID_GEN_METHOD="ESGgrid"

ESGgrid_LON_CTR="-97.5"
ESGgrid_LAT_CTR="35.0"

ESGgrid_DELX="3000.0"
ESGgrid_DELY="3000.0"

ESGgrid_NX="840"
ESGgrid_NY="600"

ESGgrid_WIDE_HALO_WIDTH="6"
```

Computational parameters

```
DT_ATMOS="40"

LAYOUT_X="30"
LAYOUT_Y="24"
BLOCKSIZE="35"
```

Write-component parameters

```
QUILTING="TRUE"
if [ "$QUILTING" = "TRUE" ]; then
  WRTCMP_write_groups="1"
  WRTCMP_write_tasks_per_group=$(( 1*LAYOUT_Y ))
  WRTCMP_output_grid="lambert_conformal"
  WRTCMP_cen_lon="${ESGgrid_LON_CTR}"
  WRTCMP_cen_lat="${ESGgrid_LAT_CTR}"
  WRTCMP_stdlat1="${ESGgrid_LAT_CTR}"
  WRTCMP_stdlat2="${ESGgrid_LAT_CTR}"
  WRTCMP_nx="837"
  WRTCMP_ny="595"
  WRTCMP_lon_lwr_left="-109.97410429"
  WRTCMP_lat_lwr_left="26.31459843"
  WRTCMP_dx="${ESGgrid_DELX}"
  WRTCMP_dy="${ESGgrid_DELY}"
fi
```

Questions?