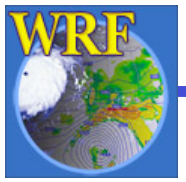
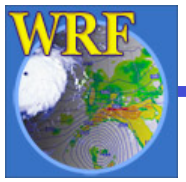
Advanced Features of the WRF Preprocessing System

Michael Duda



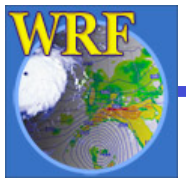
Outline

- The GEOGRID.TBL file
  - What is the GEOGRID.TBL file?
  - Ingesting new static fields
  - Example: Houston urban data
- The METGRID.TBL file
  - What is the METGRID.TBL file?
  - Example: Building a METGRID.TBL entry for a new field
  - Example: Using the METGRID.TBL file for real-time runs



The GEOGRID.TBL File

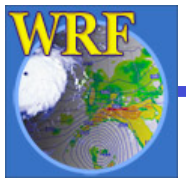
- GEOGRID.TBL is the file that determines which fields are interpolated by geogrid *at runtime*
  - Generally, user will want all of the default fields, so few reasons to edit GEOGRID.TBL
  - When new data sources are involved, or when the default treatment of fields is inadequate, user will want to edit GEOGRID.TBL
  - Each entry in GEOGRID.TBL corresponds to one data source



The GEOGRID.TBL File

- Format of GEOGRID.TBL file is simple text, with specifications of the form <*keyword*>=<*value*>
- Example entry for a 30” landuse data set:

```
==========================================
name=LANDUSEF    # Houston, TX urban data
        priority    = 1
        dest_type   = categorical
        z_dim_name  = land_cat
        interp_option = 30s:nearest_neighbor
        abs_path = 30s:/users/duda/Houston/
==========================================
```



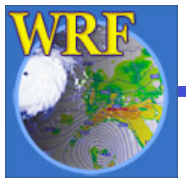
The GEOGRID.TBL File

- The `GEOGRID.TBL` file also allows user to change which interpolation methods are used for each field
- Example:

```
interp_option=sixteen_pt
```

or

```
interp_option=four_pt+average_4pt
```

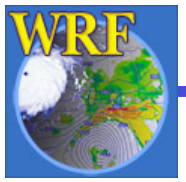


The GEOGRID.TBL File

- Other options in the GEOGRID.TBL include smoothing options and dominant category calculation
- Example:

```
smooth_option=smth-desmth

smooth_passes=2
```

- More complete information on possible options is found in Chapter 3 of the User's Guide

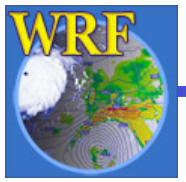


Example: GEOGRID.TBL Entries

```
===============================
name = VEG_CATEGORY
        priority = 1
        dominant_only = VEG_CAT
        dest_type  = categorical
        z_dim_name = veg_cat
        interp_option = default:nearest_neighbor
        abs_path      = default:/data/duda/MODIS/
===============================
name = SOILCTOP
        dominant = SOILCAT
        priority = 1
        dest_type = categorical
        z_dim_name = soil_cat
        interp_option =      2m:sixteen_pt
        interp_option =     10m:sixteen_pt
        rel_path=       2m:soiltype_top_2m/
        rel_path=      10m:soiltype_top_10m/
===============================
```

Entry for the field "VEG_CATEGORY"

Entry for the field "SOILCTOP"



New Fields in GEOGRID.TBL

There are three basic types of new data to be added through the GEOGRID.TBL file:

1) Completely new fields

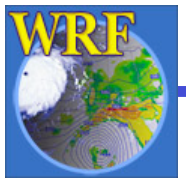
- fields that were previously not processed by geogrid

2) Different resolution data sets for an existing field

- E.g., Adding a 100-meter resolution topography data set

3) Alternative sources for a field that must be used in addition to an existing source

- E.g., A new soil category data set exists, but covers only southern Colorado

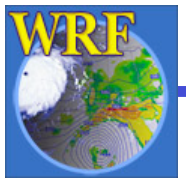


1) Completely new fields

Completely new fields:

For a new field, simply add an entry in GEOGRID.TBL for that field.

```
===================================
name = MY_NEW_FIELD_NAME
   priority = 1
   dest_type  = continuous # continuous or categorical?
   interp_option = four_pt
   abs_path      = /data/duda/mydata/
===================================
```

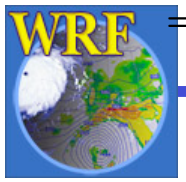


2) Different resolution data set

Different resolution data sets for an existing field :

Specify the path to the new data set and which interpolation methods should be used for the new resolution in the existing entry for that field.

```
==============================
name = HGT_M
     priority = 1
     dest_type = continuous
     smooth_option = smth-desmth
     interp_option =          30s:special(4.0)+four_pt
     interp_option =       my_res:four_pt
     interp_option =     default:four_pt
     rel_path=       30s:topo_30s/
     rel_path=    my_res:new_topo_directory/
     rel_path=  default:topo_2m/
==============================
```



3) Alternative data sources

Alternative sources for a field that must be used in addition to an existing source :

Add a new entry for the field that has the same name as the field's existing entry, but make priority of new entry higher.

```
==================================
name = HGT_M
       priority = 2
       dest_type = continuous
       interp_option = default:four_pt
       rel_path      = default:some_path/
==================================
name = HGT_M
     priority = 1
     dest_type = continuous
     interp_option = default:four_pt
     rel_path      = default:topo_2m/
==================================
```

Preparing new geogrid data sets

To add a new data source, we need to

1) Write the data in the proper binary format

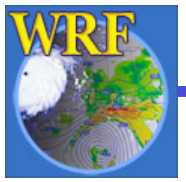
   - See Chapter 3: “Writing Static Data to the Geogrid Binary Format”

   - Can make use of read_geogrid.c and write_geogrid.c

2) Create an “index” metadata file for the data set

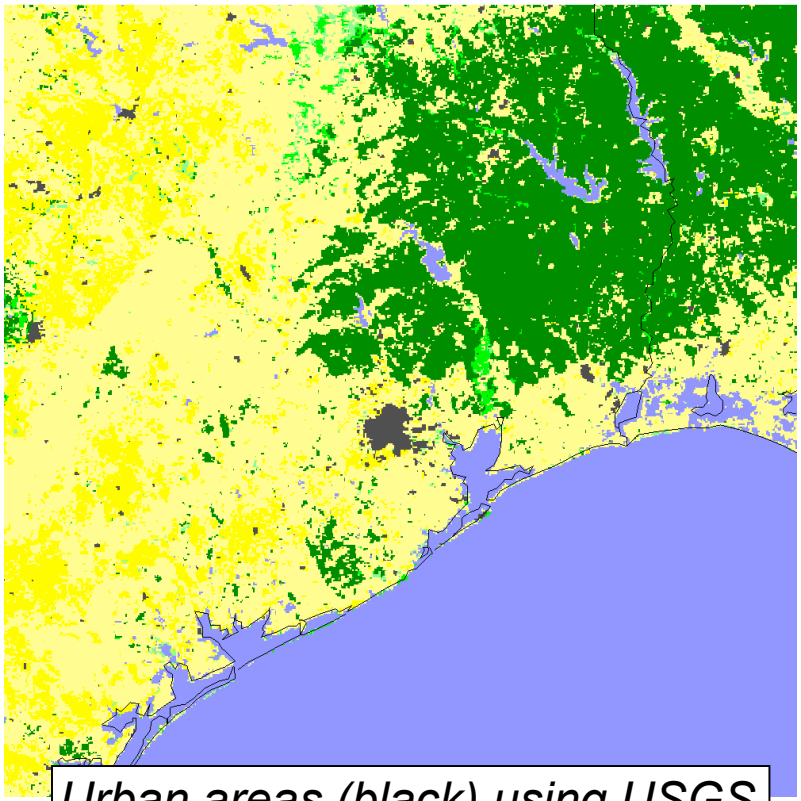
   - This tells geogrid about the projection, coverage, resolution, type, and storage representation of the data set

3) Add entry for the data in the GEOGRID.TBL file

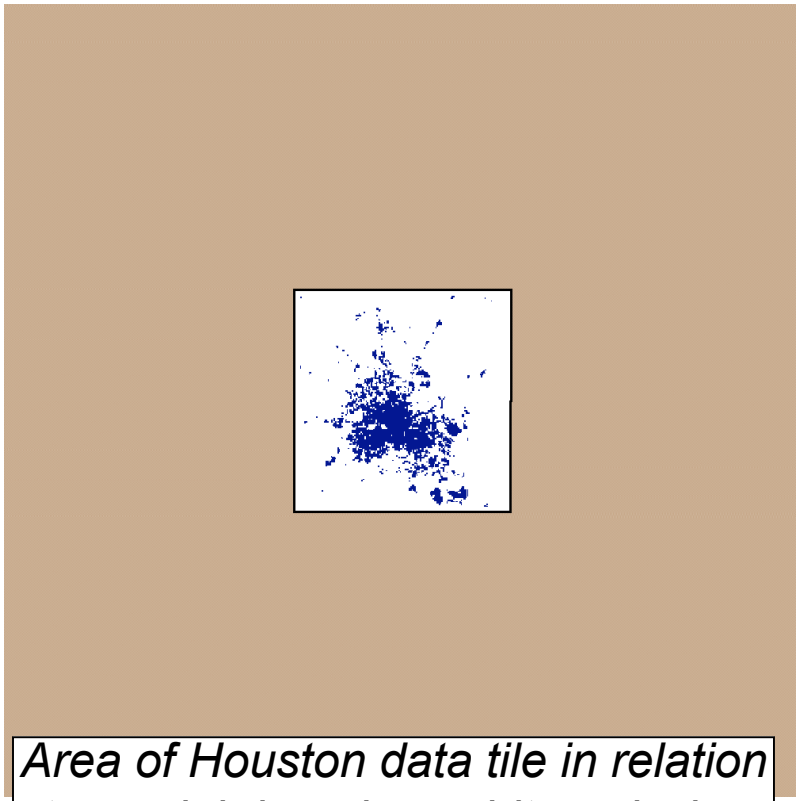


Example: Houston LU Data Set

- Given dataset for new Houston urban land use categories
  - Regular lat/lon projection, 30” resolution; categories 31, 32 & 33



Urban areas (black) using USGS 24-category data set



Area of Houston data tile in relation to model domain – white=missing data and blue=valid data

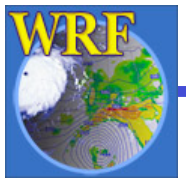
Example: Houston LU Data Set

To make use of the new data, we do the following:

1) Write the data to the binary format used by geogrid

2) Create an index file for the data

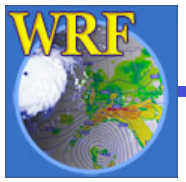
```
type=categorical
category_min=31; category_max=33
projection=regular_ll
dx=0.00833333; dy=0.00833333
known_x=1.0;   known_y=1.0
known_lat=29.3375
known_lon=-95.9958333
wordsize=1
tile_x=157; tile_y=143; tile_z=1
missing_value = 0.
units="category"
description="3-category urban LU"
```



Example: Houston LU Data Set

3) Define an entry for the data in GEOGRID.TBL

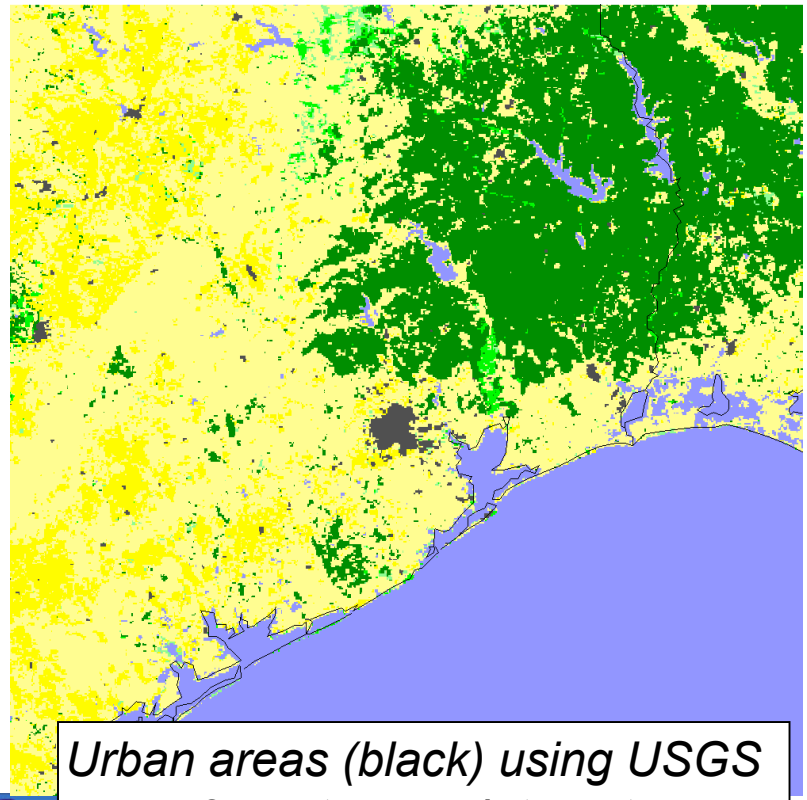
```
==============================
name=LANDUSEF
        priority   = 2
        dest_type  = categorical
        z_dim_name = land_cat
        interp_option = default:nearest_neighbor
        abs_path   = default:/users/duda/Houston/
==============================
```



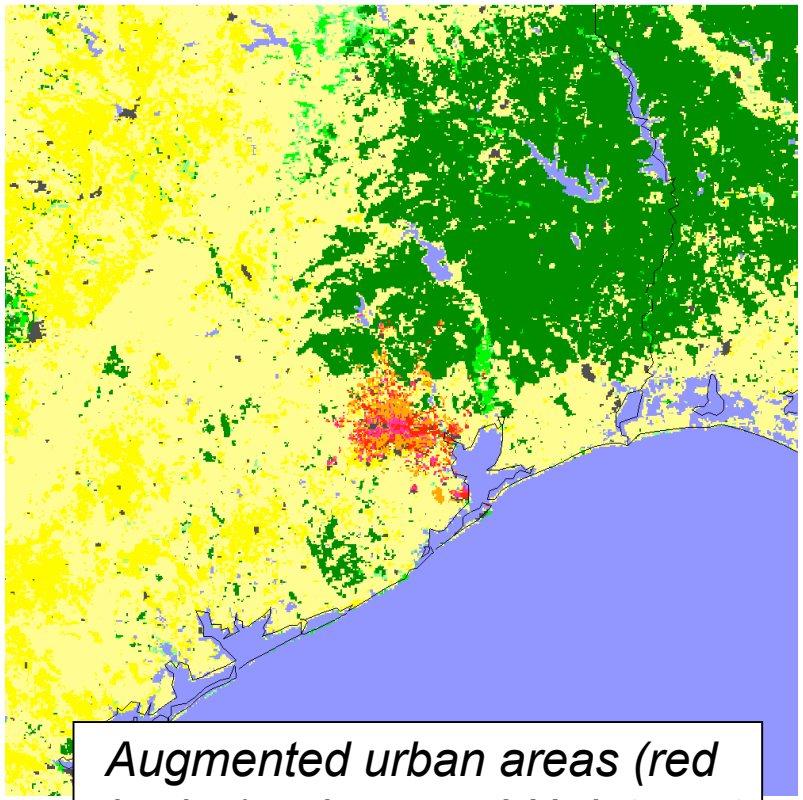
Example: Houston LU Data Set

4) Run geogrid.exe

Any gridpoints covered by Houston data will use it; otherwise default USGS data will be used



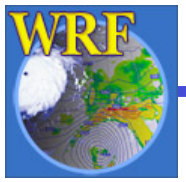
Urban areas (black) using USGS 24-category data set



Augmented urban areas (red shades) using new LU data set

Outline

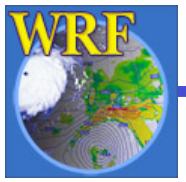
- The GEOGRID.TBL file
  - What is the GEOGRID.TBL file?
  - Ingesting new static fields
  - Example: Houston urban data
- The METGRID.TBL file
  - What is the METGRID.TBL file?
  - Example: Building a METGRID.TBL entry for a new field
  - Example: Using the METGRID.TBL file for real-time runs



The METGRID.TBL File

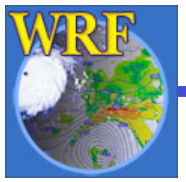
The METGRID.TBL file controls how meteorological fields are interpolated

- Unlike GEOGRID.TBL, METGRID.TBL *does not determine which fields will be processed*, only *how to process them* if they are encountered
- Every field in intermediate files will be interpolated
  - If no entry in METGRID.TBL for a field, a default interpolation scheme (nearest neighbor) will be used
  - It is possible to specify in METGRID.TBL that a field should be discarded



The METGRID.TBL File

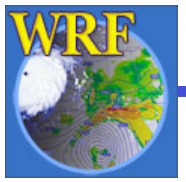
- Suitable entries in METGRID.TBL are provided for common fields
  - *Thus, many users will rarely need to edit METGRID.TBL*
- When necessary, different interpolation methods (and other options) can be set in METGRID.TBL
  - Interpolation options can depend on the source of a field



Metgrid: Program Flexibility

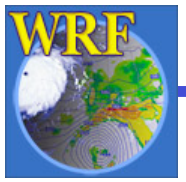
- Example METGRID.TBL entry (for “soil moisture 0-10 cm”)

```
===================================
name=SM000010
interp_option=sixteen_pt+four_pt+average_4pt
masked=water
interp_mask=LANDSEA(0)
fill_missing=1.
flag_in_output=FLAG_SM000010
===================================
```



Building a new METGRID.TBL entry

- Suppose we have a 1000x1000 domain over Houston (dx=500 m)
  - This is the same domain as in the urban land use example
- Meteorological data come from 1-degree GFS
  - *Note that we will be interpolating 1-degree data onto a 500-m grid!*
- We want to create an entry for a new soil moisture field, SM000010



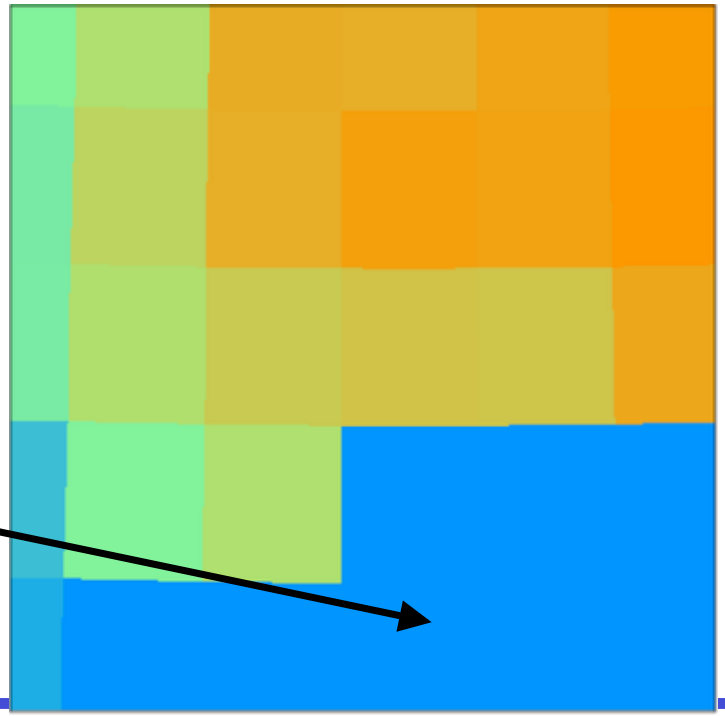
Building a new METGRID.TBL entry

- Initially, we run metgrid.exe and get the message:

```
INFORM: Entry in METGRID.TBL not found for field SM000010.
  Default options will be used for this field!
```

- Resulting field looks like

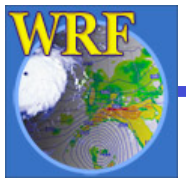
GFS puts -1E30 in water areas
(LANDSEA=0)



Building a new METGRID.TBL entry

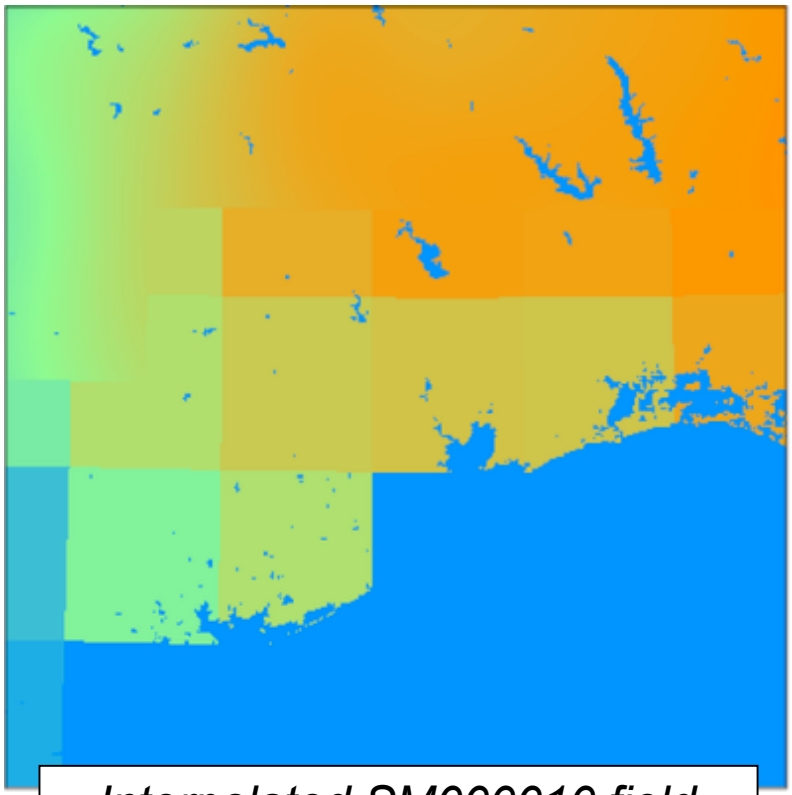
- We add an initial entry in METGRID.TBL for SM000010:

```
==========================

name = SM000010

masked = water

interp_mask = LANDSEA(0)

interp_option = sixteen_pt + nearest_neighbor

fill_missing = 0.

==========================
```

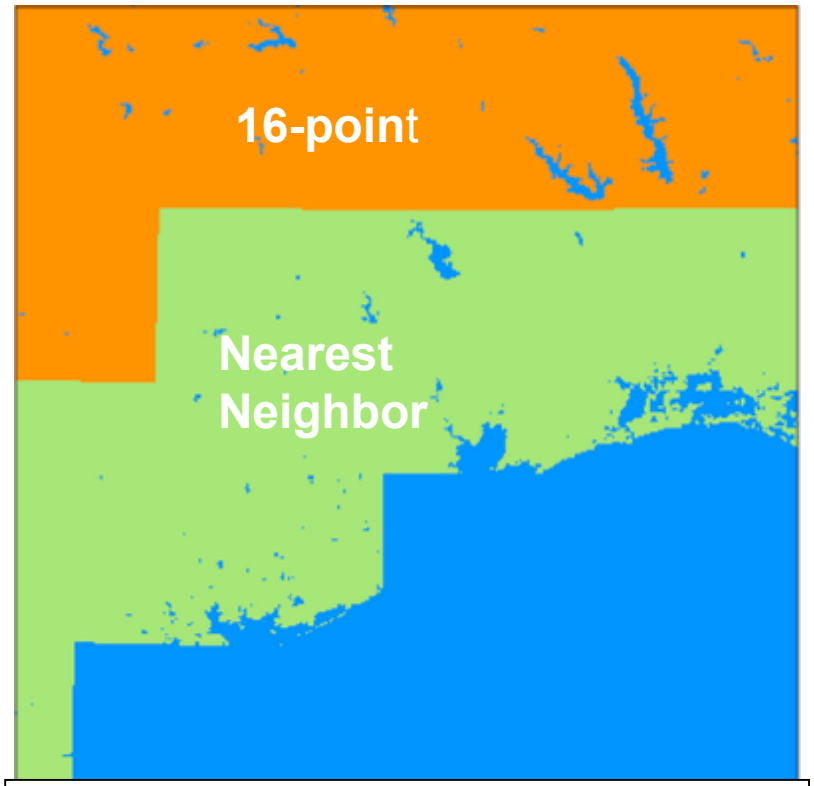


Building a new METGRID.TBL entry

- Running metgrid.exe again, the SM000010 field now looks like



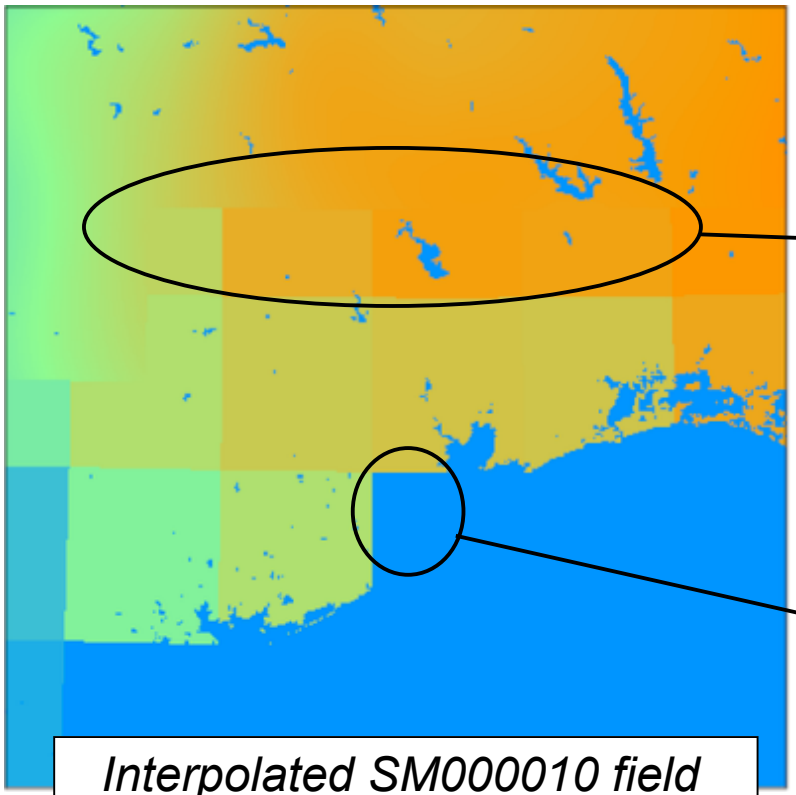
Interpolated SM000010 field (sixteen_pt + nearest_neighbor)



Which interpolator was used at each model grid point

Building a new METGRID.TBL entry

- The interpolated field looks “blocky” near the coastline



Interpolated SM000010 field (sixteen_pt + nearest_neighbor)

Should be sufficient data to use 4-point interpolation in these areas

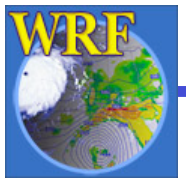
Model grid points here should be adjacent to at least one valid GFS point (though not nearest)

Building a new METGRID.TBL entry

- Update the METGRID.TBL entry for SM000010

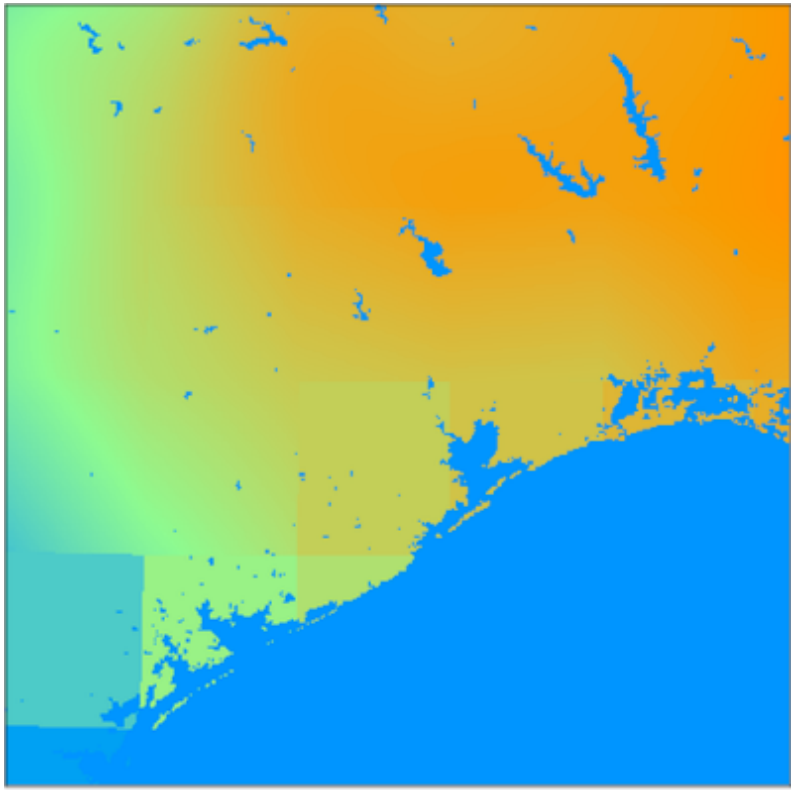
```
===========================

name = SM000010

masked = water

interp_mask = LANDSEA(0)

interp_option = sixteen_pt + four_pt + average_4pt

fill_missing = 0.

===========================
```

- If 16-pt doesn't work, then try 4-pt before reverting to a 4-point average
  - Note that 4-point average will work anywhere nearest_neighbor would (missing/masked values not counted in the average)

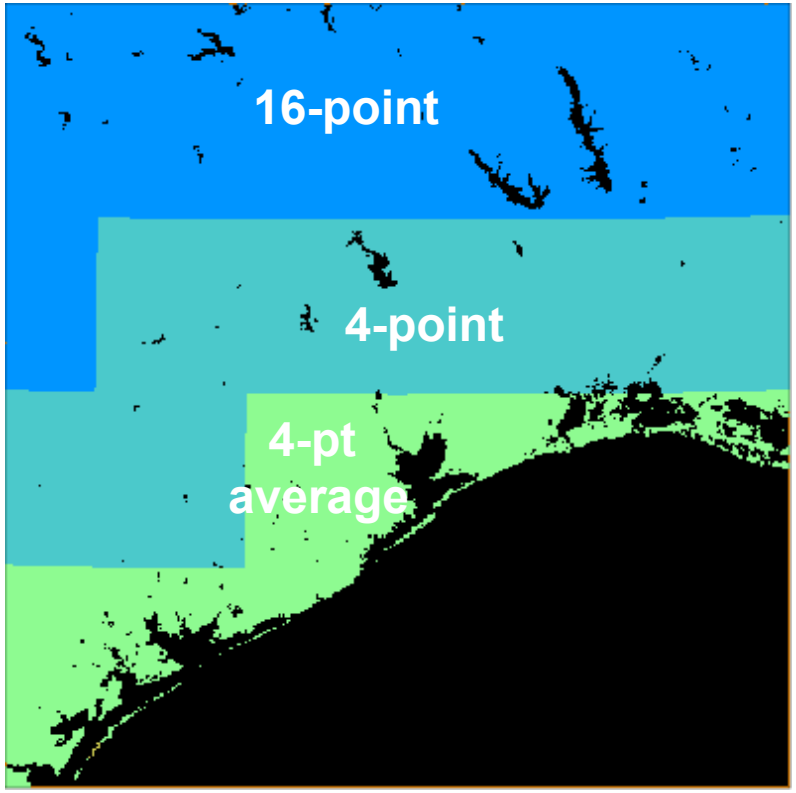


Building a new METGRID.TBL entry

- The resulting field, below-left:



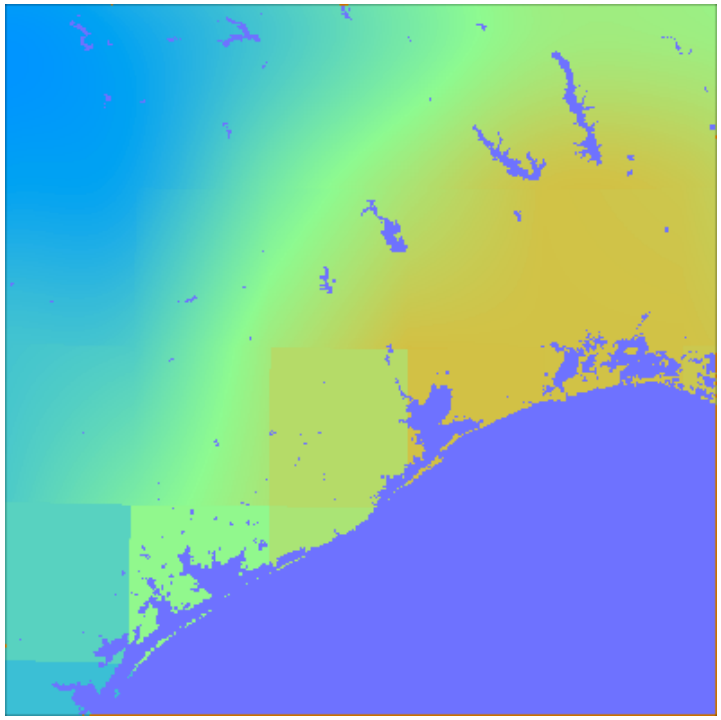
Interpolated SM000010 field (sixteen_pt + four_pt + average_4pt)



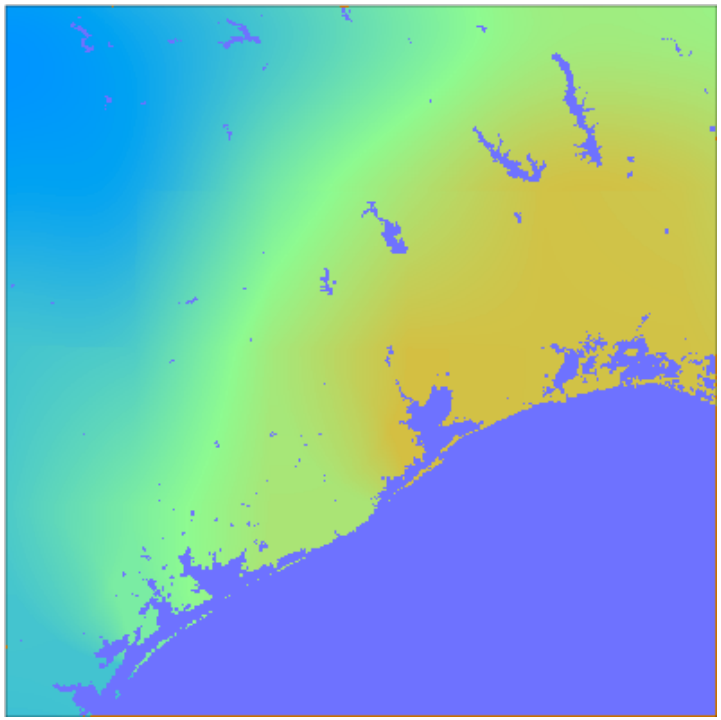
Which interpolator was used at each model grid point

Building a new METGRID.TBL entry

- By using **`wt_average_4pt`** instead of **`average_4pt`**:



sixteen_pt + four_pt + average_4pt

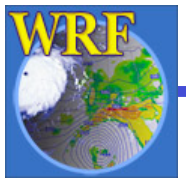


sixteen_pt + four_pt + wt_average_4pt

METGRID.TBL: Real-time System Example

- Suppose we have a real-time system that:
  - Uses GFS for initial and boundary conditions
  - When possible (i.e., if the files are available soon enough) uses *soil moisture* and *soil temperature* fields from AGRMET

- In our system, it may occasionally happen that the AGRMET files are not ready when we want to start our WRF run
  - Because system is real-time, we want to proceed using just the GFS land surface fields!



METGRID.TBL: Real-time System Example

- We already know how to run ungrib on multiple sources of data to get

  GFS:YYYY-MM-DD_HH

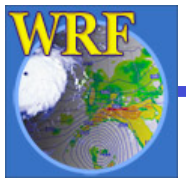
  and

  AGRMET:YYYY-MM-DD_HH

  intermediate files, and specify

  fg_name = 'GFS', 'AGRMET',

  in the `&metgrid` namelist record to use both files



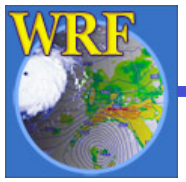
METGRID.TBL: Real-time System Example

Without further changes, what happens if:

1) Only GFS data are available when we run metgrid

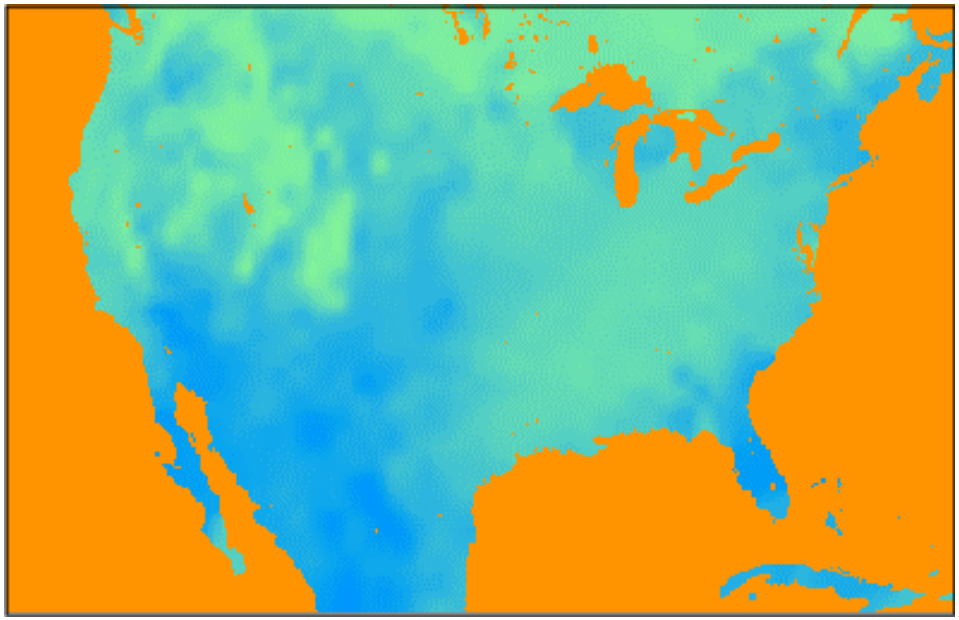
- Metgrid runs and warns that no AGRMET data files were found:

```
Processing 2006-04-01_00
     GFS
     AGRMET
 WARNING: Couldn't open file AGRMET:2006-04-01_00 for
 input.
```



METGRID.TBL: Real-time System Example

And the 0-10 cm soil moisture field (SM000010) looks like:

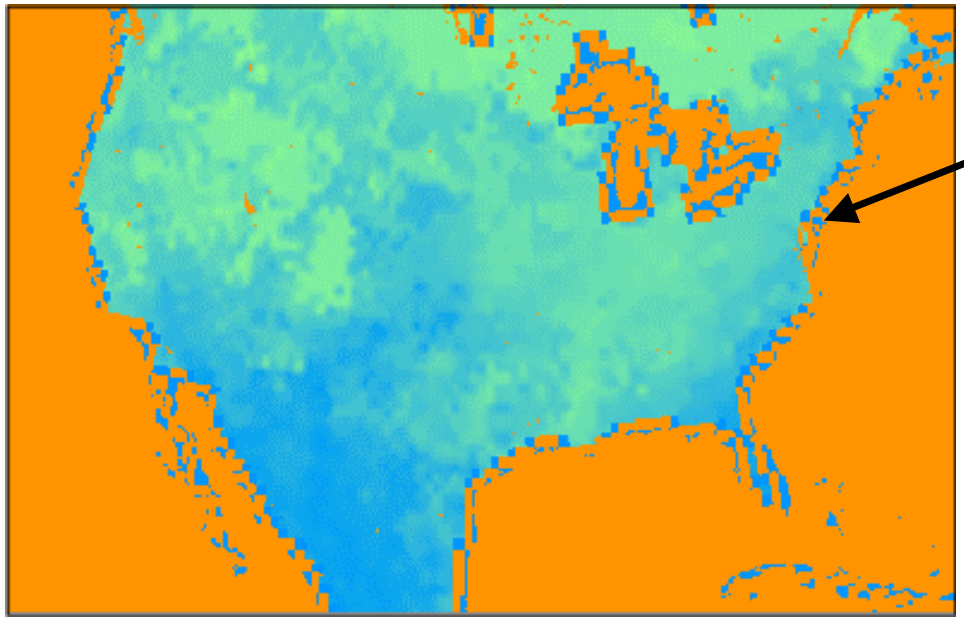


METGRID.TBL: Real-time System Example

However, what happens if:

2) Both GFS and AGRMET files are available when we run metgrid?

Our SM000010 field looks like:



We get unreasonable values with magnitude ~1E30 near land-water boundaries!

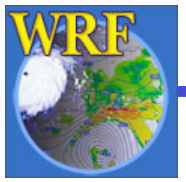
METGRID.TBL: Real-time System Example

What went wrong?

In both Vtable.GFS and Vtable.AGRMET, the land-sea mask field is named LANDSEA

- In METGRID.TBL, our entry for SM000010 says:

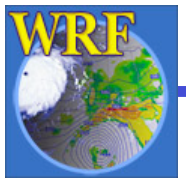
```
==========================================
name=SM000010
interp_option=sixteen_pt+four_pt+wt_average_4pt+search
masked=water
interp_mask=LANDSEA(0)
fill_missing=1.
flag_in_output=FLAG_SM000010
==========================================
```



METGRID.TBL: Real-time System Example

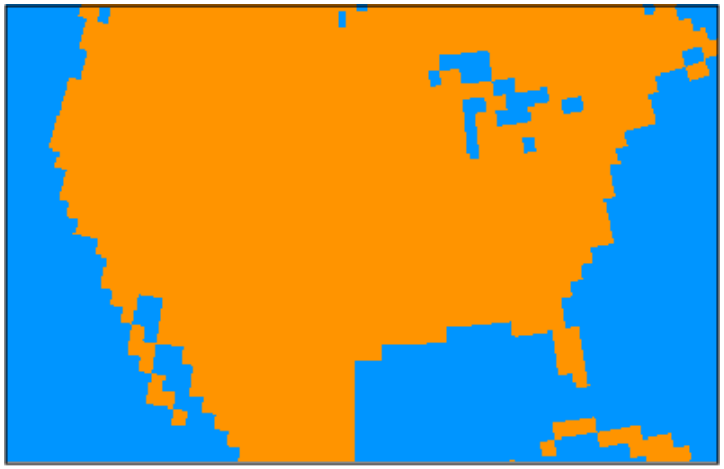
```
========================================
name=SM000010
interp_option=sixteen_pt+four_pt+wt_average_4pt+search
masked=water
interp_mask=LANDSEA(0)
fill_missing=1.
flag_in_output=FLAG_SM000010
========================================
```

After metgrid reads in LANDSEA from GFS file *to use as an interpolation mask*, it ignored the LANDSEA field from AGRMET *for use as a mask*.

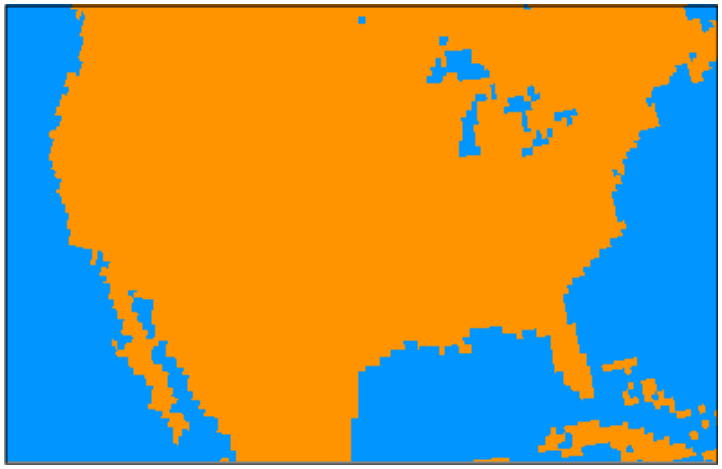


METGRID.TBL: Real-time System Example

When metgrid interpolated SM000010, it used the GFS landmask for a field masked by the AGRMET landmask!



GFS LANDSEA field



AGRMET LANDSEA field

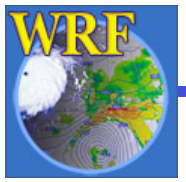
METGRID.TBL: Real-time System Example

Solution:

- Rename LANDSEA to *AGR_LAND* in Vtable.AGRMET
- Rename LANDSEA to *GFS_LAND* in Vtable.GFS
- Create separate entries in METGRID.TBL

  one for GFS SM000010 field

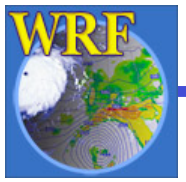
  another for AGRMET SM000010 field



METGRID.TBL: Real-time System Example

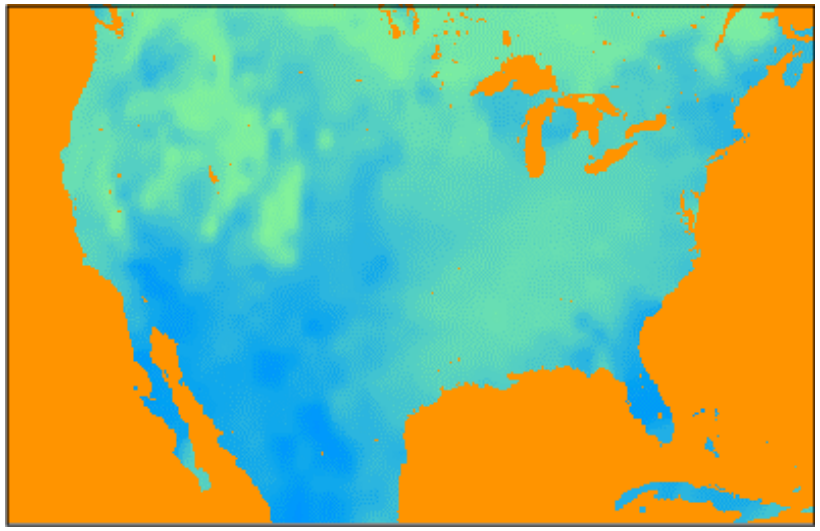
```
===================================
name=SM000010; from_input=GFS
interp_option=sixteen_pt+four_pt+wt_average_4pt+search
masked=water
interp_mask=GFS_LAND(0)
fill_missing=1.
flag_in_output=FLAG_SM000010
===================================
```

```
===================================
name=SM000010; from_input=AGRMET
interp_option=sixteen_pt+four_pt+wt_average_4pt+search
masked=water
interp_mask=AGR_LAND(-1.E30)
fill_missing=1.
flag_in_output=FLAG_SM000010
===================================
```

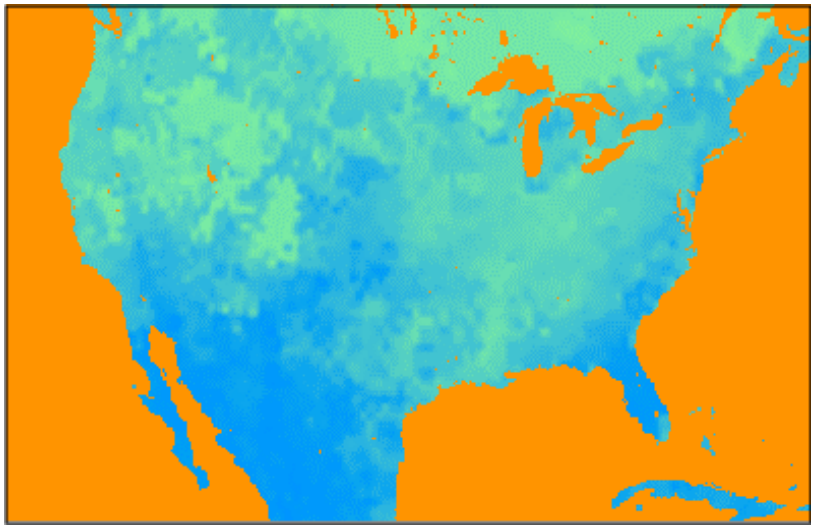


METGRID.TBL: Real-time System Example

With modified Vtables and METGRID.TBL:



The SM000010 field when only GFS files are available



The SM000010 field when both GFS and AGRMET files are available

Questions?