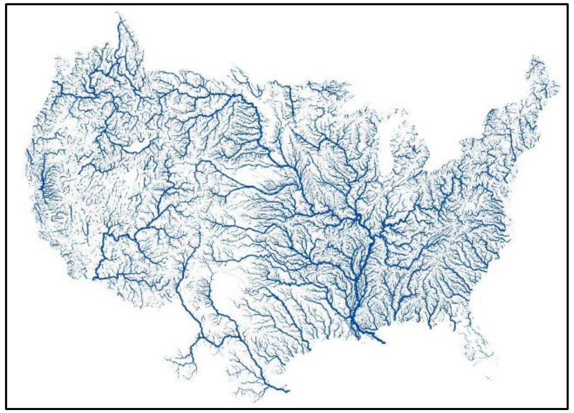
WRF-Hydro GIS Pre-processing Tool Overview



K. Sampson

National Center for Atmospheric Research

Outline

- WRF Pre-processing
- WRF-Hydro ArcGIS Pre-processing tools
- Basic GIS terrain pre-processing for WRF-Hydro
- Demonstration: Generating WRF-Hydro Routing Grids

WRF-Hydro ArcGIS Pre-Processing Toolkit

WRF-Hydro GIS Tools

- Pre-processing tools, written in Python, using ArcGIS python API (`arcpy`)
- Variety of WRF-Hydro configuration options supported
- Fast, efficient method for producing the 'routing stack' necessary to run WRF-Hydro
- Consistent processing methodology between domains, regions, datasets
- Provides WRF-Hydro with a complete set of hydrologically processed routing grids and spatial metadada
- Removes the heavy GIS burden from modelers

https://ral.ucar.edu/projects/wrf_hydro/pre-processing-tools

GitHub https://github.com/NCAR/wrf_hydro_arcgis_preprocessor

- GEOGRID_STANDALONE.pyt
  - Processing
    - Process GEOGRID File
  - Utilities
    - Add Lake Parameters
    - Add reach-based routing
    - Build GWBUCKPARM Table
    - Build Spatial Metadata File
    - Create Domain Boundary Shapefile
    - Examine Outputs of GIS Preprocessor
    - Export ESRI projection file (PRJ) from GEOGRID file
    - Export grid from GEOGRID file
    - Generate Latitude and Longitude Rasters

WRF-Hydro & ArcGIS

- Desktop GIS Application Suite
- Site-licenses available at most US academic institutions
- Ecosystem of compatible hydrology tools
  - Spatial Analyst
  - ArcHydro
  - TauDEM
- Extensible using Python API (arcpy)
- Handles everything from projections, to analysis, to mapmaking in one library
- netCDF4-Python included as of 10.3

Requirements

- ArcGIS for Desktop
  - Version 10.3.1 or higher
    - has been minimally tested with ArcGIS 10.4, 10.5, and 10.6.
    - Any license level (Basic, Standard, or Advanced)
    - Spatial Analyst extension required
  - Any Python version installed with ArcGIS Desktop
    - ArcGIS 10.3: Python 2.7.8, NumPy 1.7.1
  - Minimal functionality in ArcGIS Pro / Python 3.x
    - Not fully tested

Python Toolboxes

- Python script wrapped to act as an ArcGIS Toolbox

Toolbox
Script (.pyt)

Function
Script (.py)

- PYT file is the toolbox script containing multiple toolsets
  - Functions called from separate script (`wrf_hydro_functions.py`)
- Parameter handling and validation

<u>Advantages</u>

Easy to modify
Portable
Many tools organized

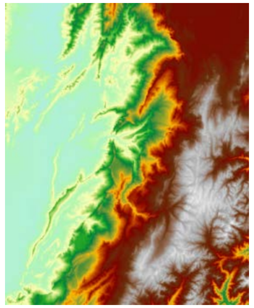
Python Toolboxes (.pyt)

- Toolboxes wrapped in Python script…

```
# --- Toolbox Classes --- #
class Toolbox(object):
    def __init__(self):
        """Define the toolbox (the name of the toolbox is the name of the
        .pyt file)."""
        self.label = "WRFHydro_GIS_Pre-Processor"
        self.alias = ""
        self.description = "This is a standalone ArcGIS geoprocessing toolbox for WRF-Hydro."

        # List of tool classes associated with this toolbox
        self.tools = [ProcessGeogridFile,
                      ExportGrid,
                      ExamineOutputs,
                      ExportPRJ,
                      GenerateLatLon,
                      SpatialMetadataFile,
                      DomainShapefile,
                      Reach_Based_Routing_Addition,
                      Lake_Parameter_Addition]

class ProcessGeogridFile(object):
    def __init__(self):
        """Define the tool (tool name is the name of the class)."""
        self.label = "Process GEOGRID File"
        self.description = "This tool takes an input WRF GEOGRID file in NetCDF format" + \
                           " and uses the HGT_M grid and an input high-resolution elevation grid" + \
                           "to produce a high-resolution hydrologically processed output."
        self.canRunInBackground = True                                        #self.canRunInBackground = False
        self.category = "Processing"

    def getParameterInfo(self):
        """Define parameter definitions"""

        in_nc = arcpy.Parameter(
            displayName="Input GEOGRID File",
            name="in_nc",
            datatype="File",
            parameterType="Required",
            direction="Input")

        in_csv = arcpy.Parameter(
            displayName="Forecast Points (CSV)",
            name="in_csv",
            datatype="File",
```

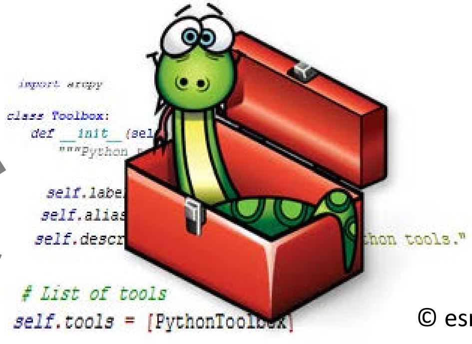
Preprocessor

Inputs



netCDF

Other parameters



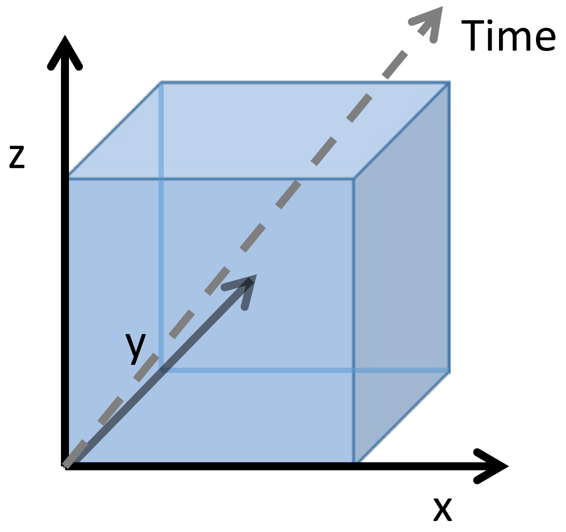
© esri

Outputs

- FRNG_1km_250m_TestData_Full
  - Fulldom_hires.nc
  - GEOGRID_LDASOUT_Spatial_Metadata.nc
  - gw_basns_geogrid.txt
  - gw_basns_geogrid.prj
  - LAKEPARM.nc
  - lakes.shp
  - Route_Link.nc
  - streams.shp

NetCDF File Format

- network Common Data Form
  - ".nc" extension
- Self describing
  - Includes information about the data coordinate system
- Machine independent
  - Usable in many operating systems
- Used extensively in Atmospheric Science
- Multidimensional
  - x,y,z,t



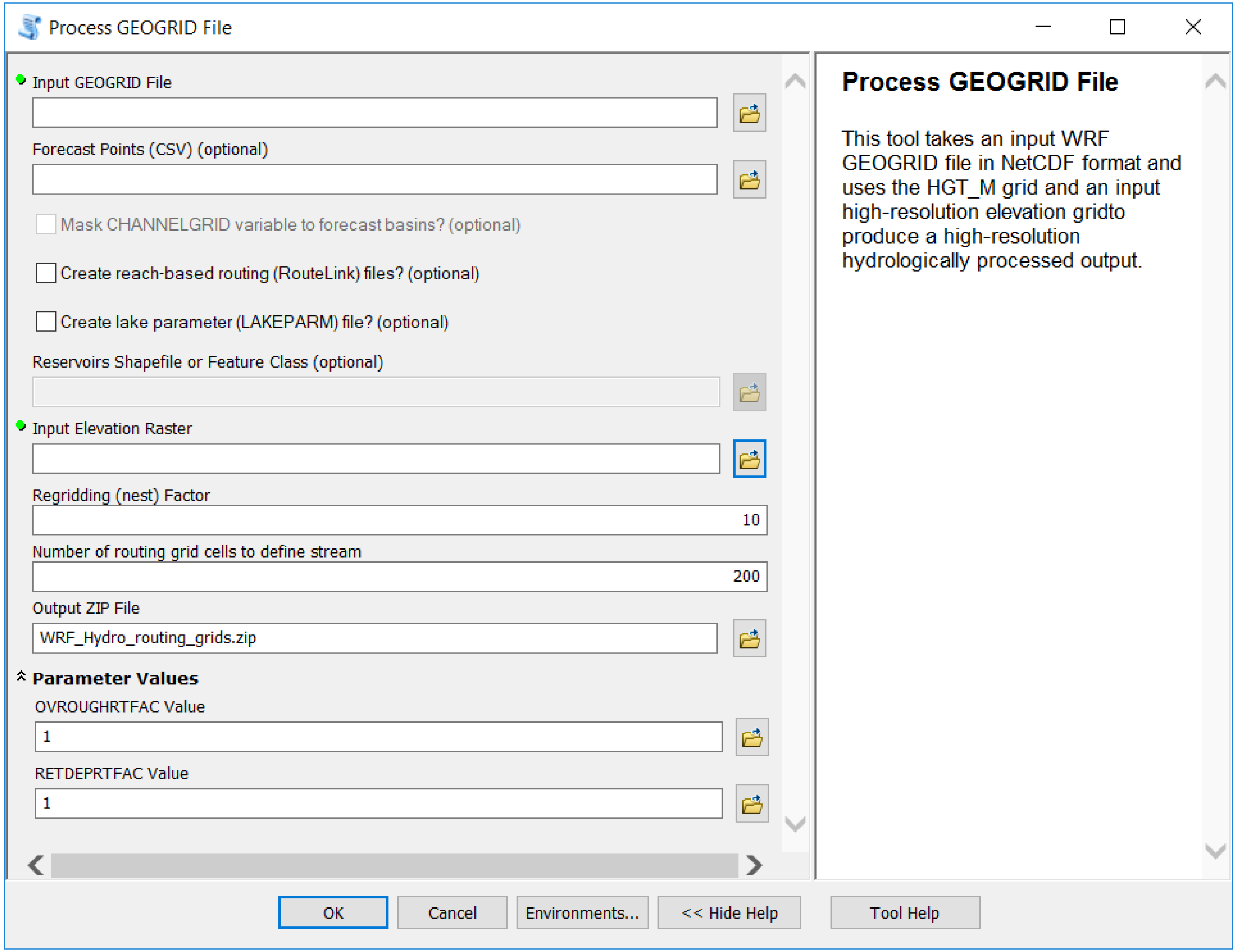
Fulldom File (Routing Grids)

- netCDF WRF-Hydro input file
- Full high-resolution domain file (Fulldom_hires.nc)
- Stores all routing grids as 2-dimensional variables
- Stores CF-compliant spatial metadata
  - grid_mapping
  - Projection information
  - Coordinate System variable
  - ArcGIS-compliant projection information
- Easy to import into GIS Applications (ArcGIS, QGIS)
- Ingested directly by WRF-Hydro

A Note on CF Metadata

- Climate and Forecast Conventions for netCDF data
  - Like a standard
  - Current version 1.7
  - http://cfconventions.org/latest.html
- CF conventions for just about any type of data
  - Gridded
  - Point
  - Profile
  - timeSeries
- CF-compliant netCDF files make them much easier to use in client applications
  - Panoply, ArcGIS, QGIS

Process Geogrid File



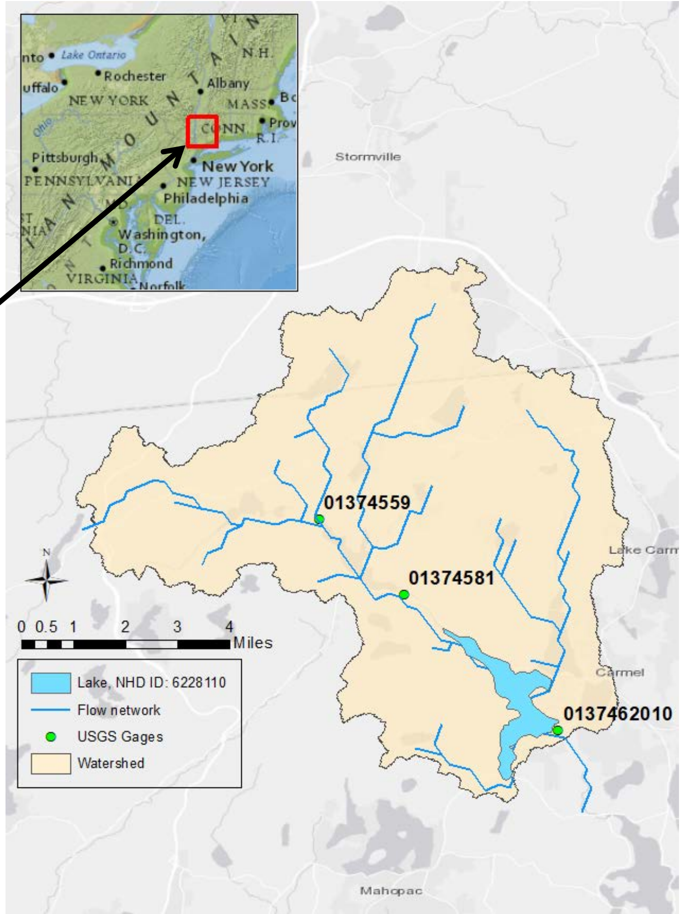
Inputs

- Required:
  - WRF GEOGRID file (.nc)
  - High-resolution Elevation
    - Elevation file (Esri GRID, GeoTIFF, etc.)
    - Mosaic Dataset
- Parameters
  - Regridding Factor – nesting relationship of routing:land grids
  - Minimum basin size (in routing grid cells)
  - OVROUGHRTFAC – constant
  - RETDEPRTFAC – constant
  - LKSATFAC – constant
- Optional:
  - Station Locations (.csv)
  - Lake Polygons (polygon feature class or .shp)

Model Domain

- Area of interest
- Defines model domain
  - Size
  - Location
  - Horizontal resolution
- Defined by GEOGRID file
  - Example:

geo_em.d03.nc



Input: WRF Geogrid File

The purpose of the Geogrid file is to define the simulation domain and interpolate various static geographical datasets to the model grid.

- GEOGRID is used in the WRF-Hydro GIS Pre-processor to define the domain's coordinate reference system, extent, resolution, and certain 2D variables:
  - HGT_M (elevation)
  - LU_INDEX (landuse)
- Currently supported GEOGRID coordinate systems
  - MAP_PROJ = 1 (Lambert Conformal Conic)
  - MAP_PROJ = 3 (Mercator)
  - MAP_PROJ = 6 (Cylindrical Equidistant but NOT w/ rotated pole)
  - MAP_PROJ = 2 (Polar Stereographic)

GEOGRID: Projected Coordinate System

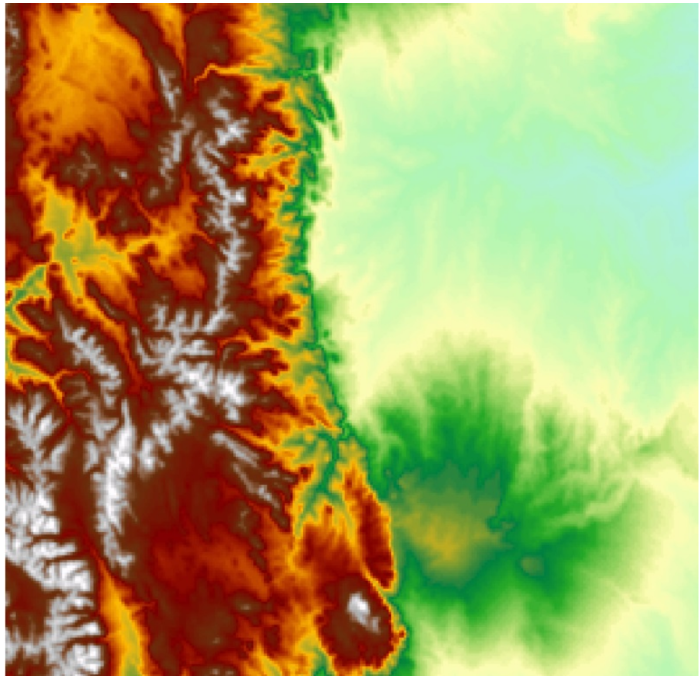
Front_Range_geo_em.d02.nc

```
:TITLE = "OUTPUT FROM GEOGRID V3.5.1";
:SIMULATION_START_DATE = "0000-00-00_00:00:00";
:WEST-EAST_GRID_DIMENSION = 50; // int
:SOUTH-NORTH_GRID_DIMENSION = 36; // int
:BOTTOM-TOP_GRID_DIMENSION = 0; // int
:WEST-EAST_PATCH_START_UNSTAG = 1; // int
:WEST-EAST_PATCH_END_UNSTAG = 49; // int
:WEST-EAST_PATCH_START_STAG = 1; // int
:WEST-EAST_PATCH_END_STAG = 50; // int
:SOUTH-NORTH_PATCH_START_UNSTAG = 1; // int
:SOUTH-NORTH_PATCH_END_UNSTAG = 35; // int
:SOUTH-NORTH_PATCH_START_STAG = 1; // int
:SOUTH-NORTH_PATCH_END_STAG = 36; // int
:GRIDTYPE = "C";
:DX = 1000.0f; // float
:DY = 1000.0f; // float
:DYN_OPT = 2; // int
:CEN_LAT = 39.940014f; // float
:CEN_LON = -105.42999f; // float
:TRUELAT1 = 30.0f; // float
:TRUELAT2 = 50.0f; // float
:MOAD_CEN_LAT = 39.940014f; // float
:STAND_LON = -105.0f; // float
:POLE_LAT = 90.0f; // float
:POLE_LON = 0.0f; // float
:corner_lats = 39.783337f, 40.093864f, 40.095993f, 3
:corner_lons = -105.714264f, -105.71753f, -105.14442
:MAP_PROJ = 1; // int
:MMINLU = "USGS";
:NUM_LAND_CAT = 24; // int
:ISWATER = 16; // int
:ISLAKE = -1; // int
:ISICE = 24; // int
:ISURBAN = 1; // int
:ISOILWATER = 14; // int
:grid_id = 1; // int
:parent_id = 1; // int
:i_parent_start = 1; // int
:j_parent_start = 1; // int
:i_parent_end = 50; // int
:j_parent_end = 36; // int
:parent_grid_ratio = 1; // int
:sr_x = 1; // int
:sr_y = 1; // int
:FLAG_MF_XY = 1; // int
}
```

WKT

```
"PROJCS['Lambert_Conformal_Conic',GEOGCS['GCS_Sphere',DATUM['D_Sphere',SPHEROID['Sphere',6370000.0,0.0]],PRIMEM['Greenwich',0.0],UNIT['Degree',0.0174532925199433]],PROJECTION['Lambert_Conformal_Conic'],PARAMETER['false_easting',0.0],PARAMETER['false_northing',0.0],PARAMETER['central_meridian',-105.0],PARAMETER['standard_parallel_1',30.0],PARAMETER['standard_parallel_2',50.0],PARAMETER['latitude_of_origin',39.9400138855],UNIT['Meter',1.0]];-36695400 -29251300 10000;-100000 10000;-100000 10000;0.001;0.001;0.001;IsHighPrecision"
```

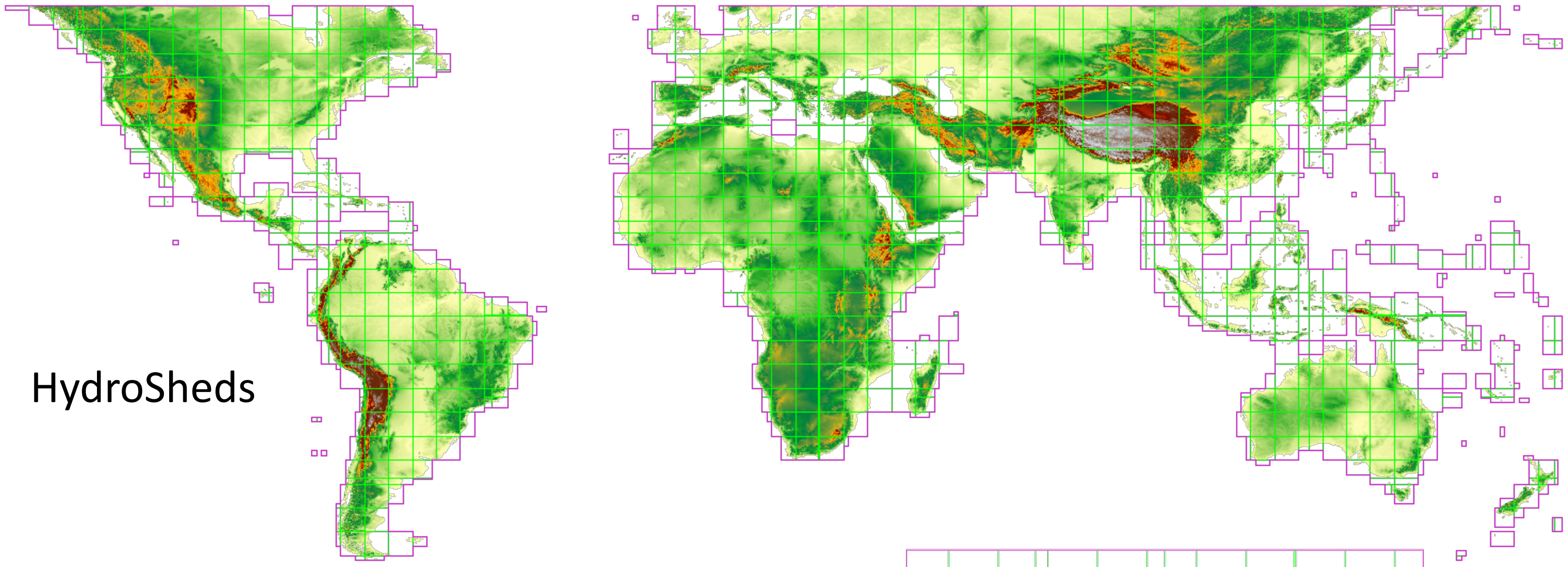
geo_em.d01.boulder_creek_1km.prj

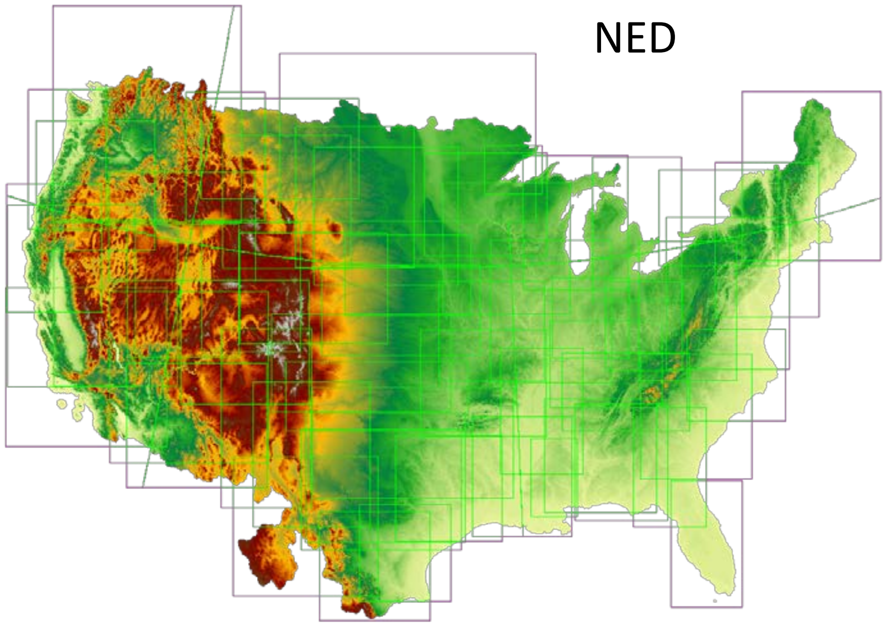


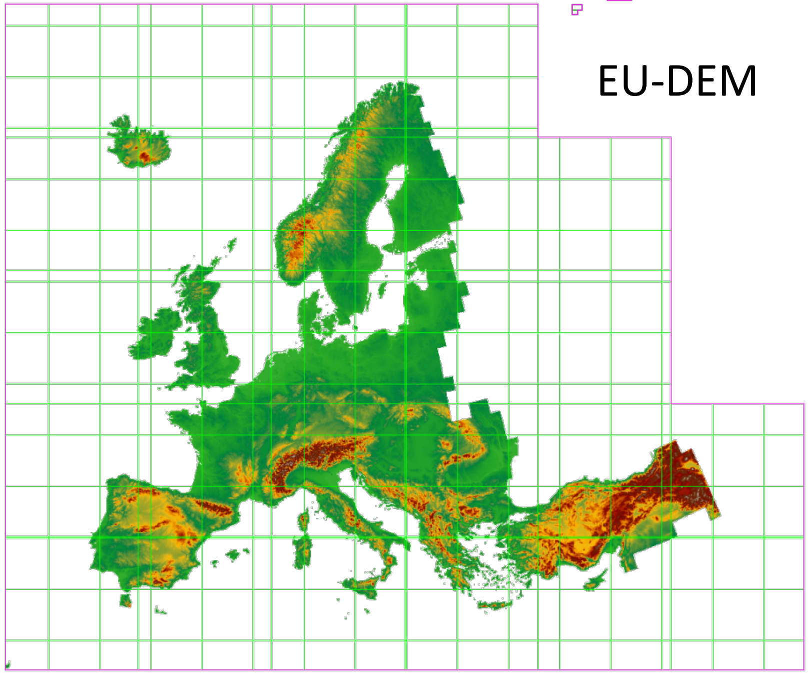
Input Elevation Raster

- Must be an ArcGIS-readable raster format
- Must contain valid coordinate reference system
- Must cover entire extent (and more) of your GEOGRID domain
- Elevation units must be converted to meters (m)
- Should by hydrologically corrected
  - Not necessary but helps with channel placement, hydro enforcement, etc.

Input Elevation Mosaics

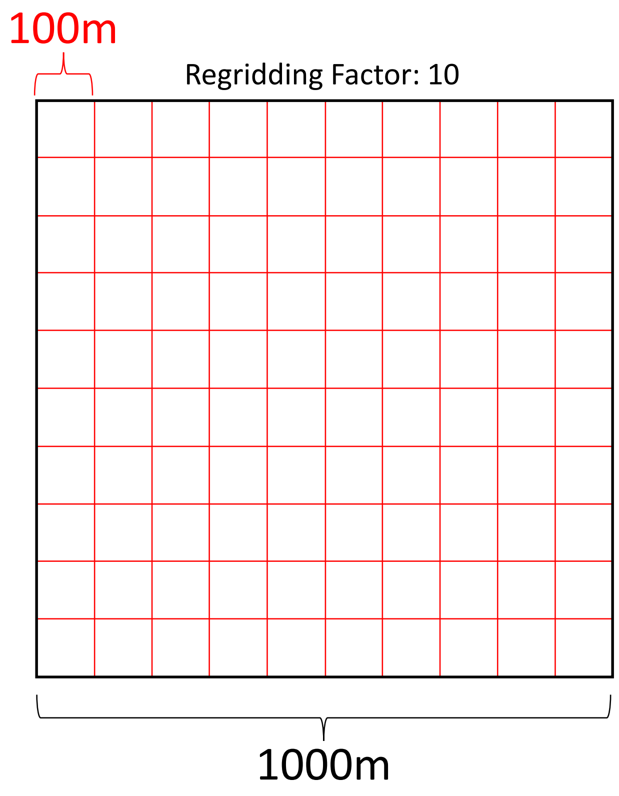


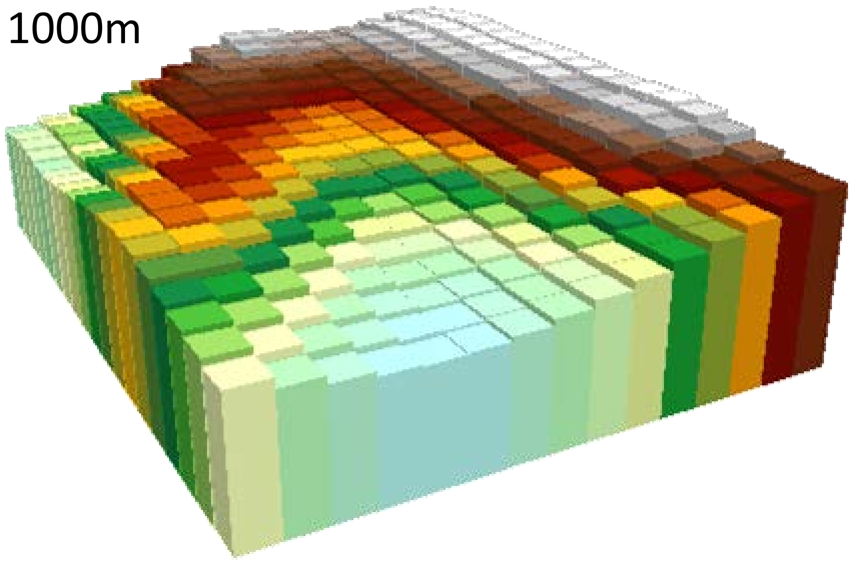


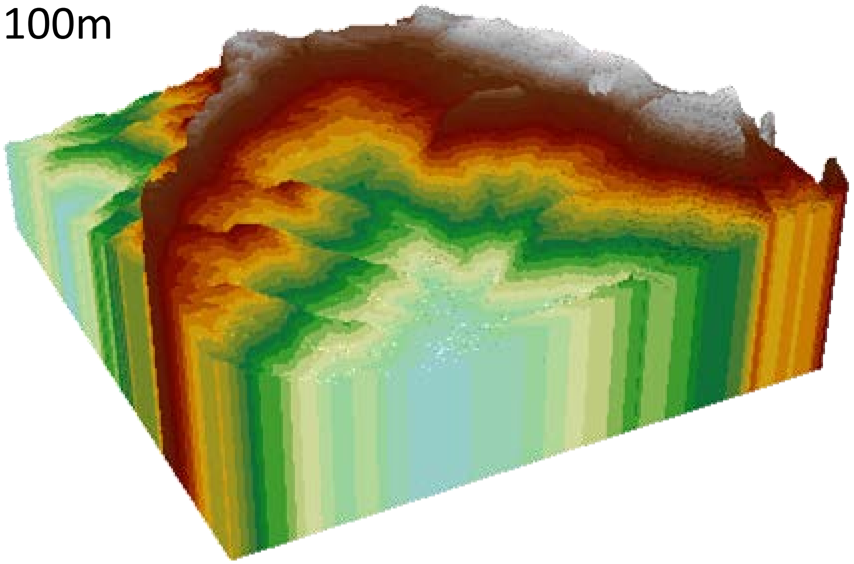


Input Regriddng Factor

$$\frac{\text{GEOGRID Resolution}}{\text{Regridding Factor}} = \text{Routing Resolution}$$

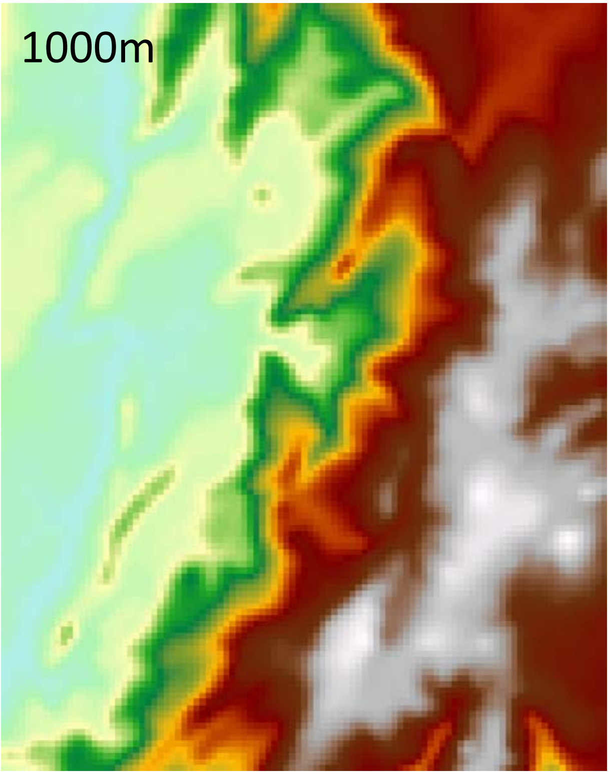




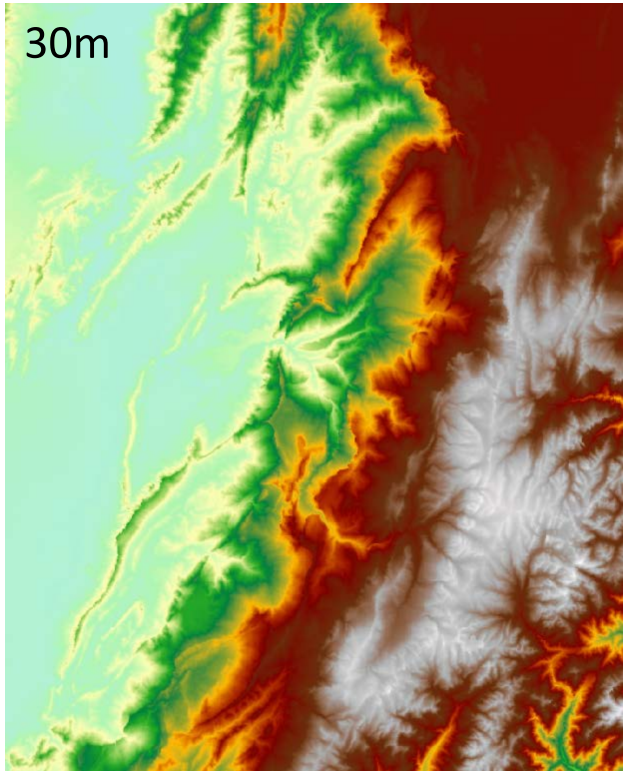


Raster Resolution for Terrain Processing

Model Resolution

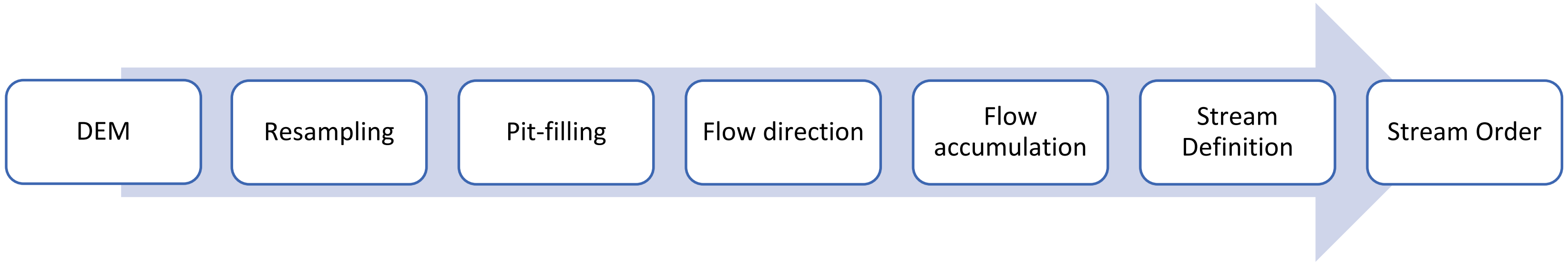


High Resolution



Terrain Pre-processing Workflow

- Resample high-resolution DEM and land use
- Void-fill the resampled DEM
- D8 Flow Direction
- Flow Accumulation
- Derive CHANNELGRID from flow accumulation raster using threshold of minimum basin size
- Derive Strahler stream order from CHANNELGRID



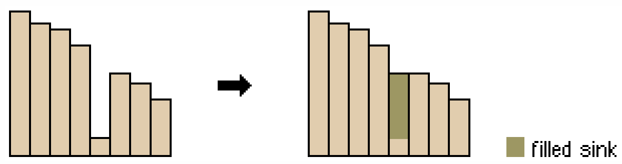
Basic workflow for terrain pre-processing of WRF-Hydro routing grids.

Reproject & Resample Hydro DEM

- Project input DEM to model projection and domain
- Resample to routing grid resolution
  - BILINEAR resampling uses a distance-weighted average of the 4 nearest cell centers.
- Re-projection and resampling can ‘break’ the input HydroDEM.
  - Causing artificial ‘pits’.
  - Filling in ‘burned in’ areas.
- Even though we start with a HydroDEM, we ‘break’ it, then re-condition it.

Process: Pit Filling

Spatial Analyst “Fill“ Tool



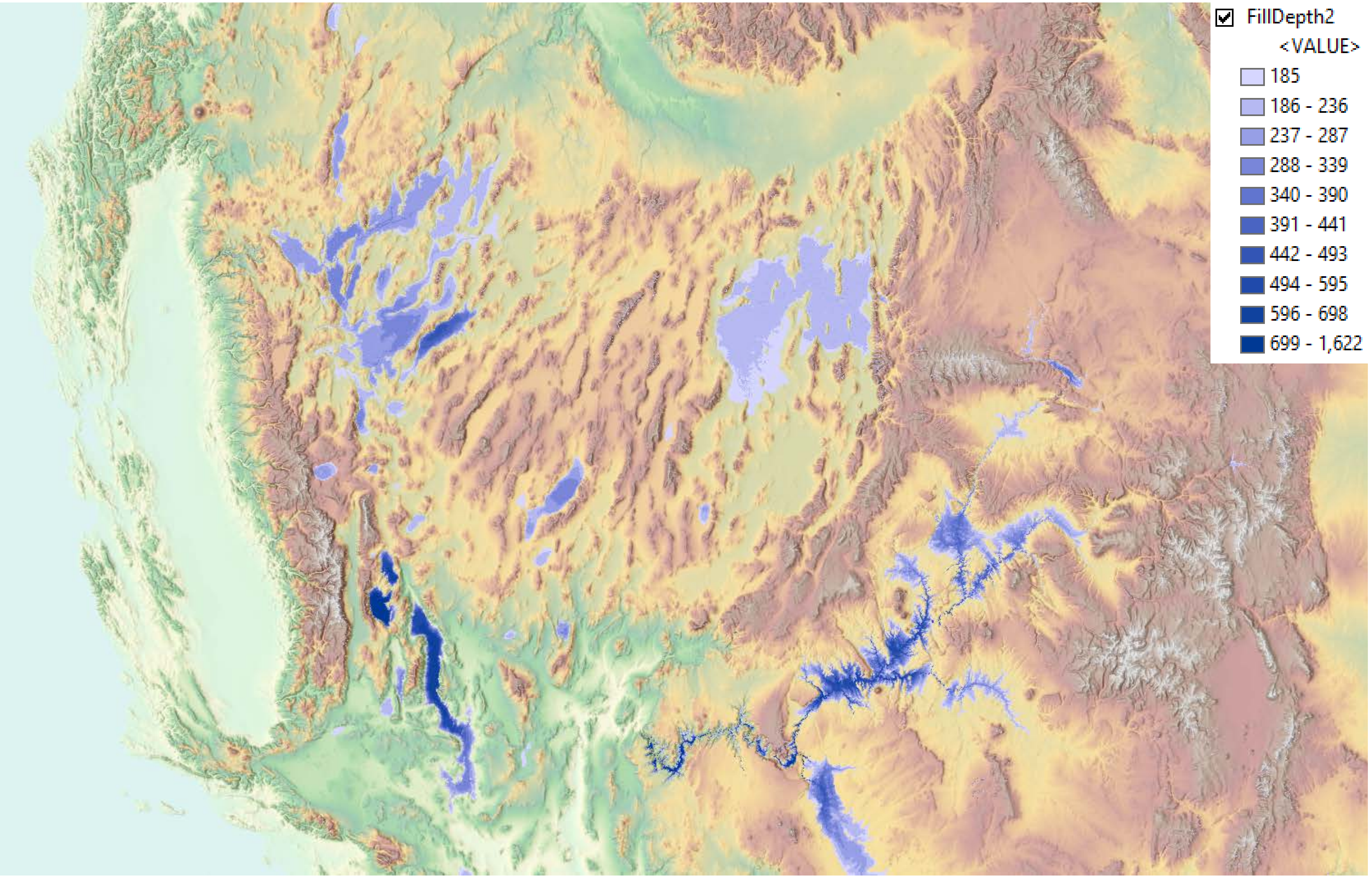
Profile view of a sink before and after running Fill

- Fill depressions so that water can roll downhill only. This also creates a smoother Dem than you might find in nature.
- This simple hydro-enforcement method can resolve most flow issues in a DEM.
- Optional z-limit (global variable) to limit fill depth.

Methods: Tarboton et al (1991)

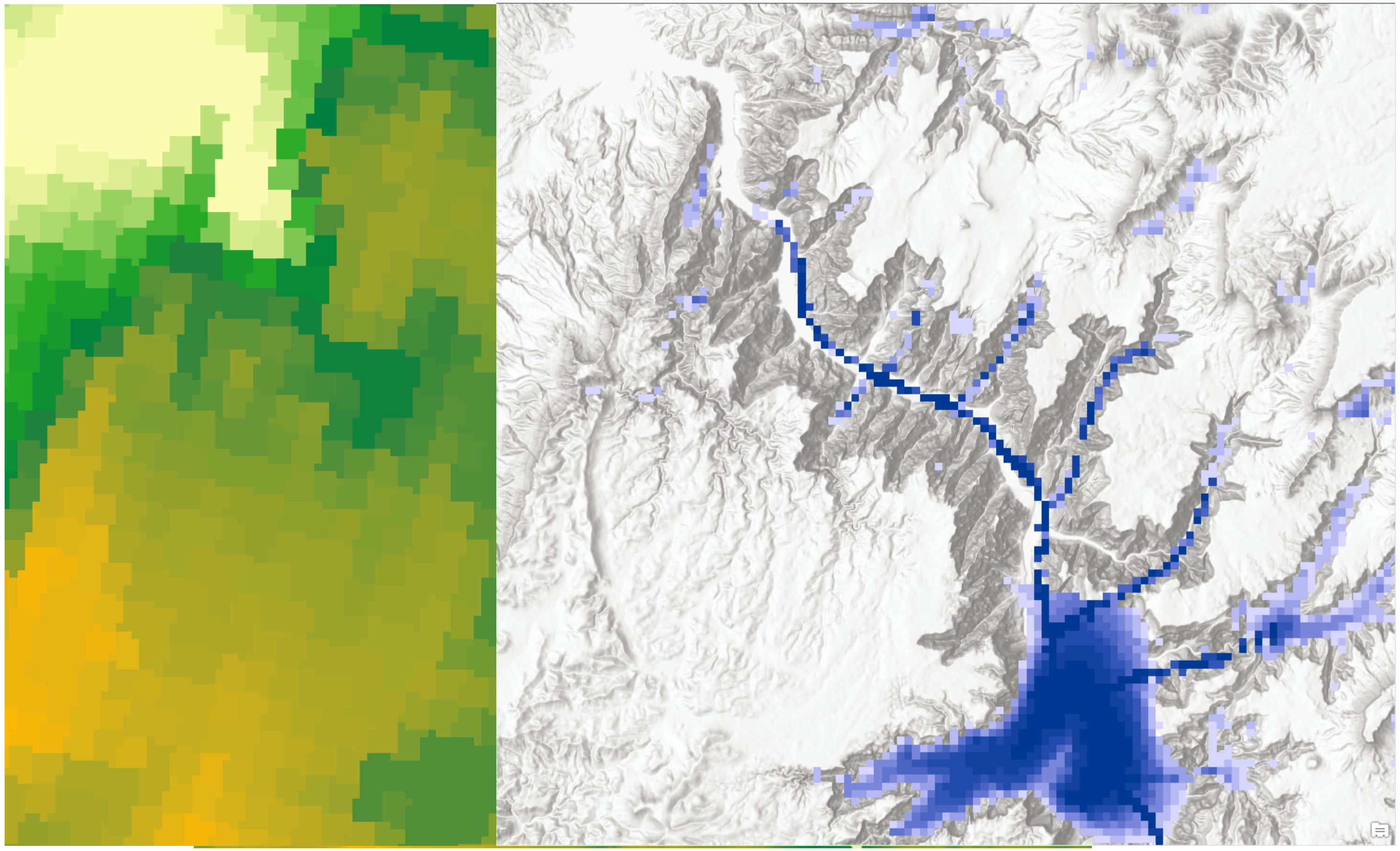
Pit Filling Issues

- …some pits are natural, some are not.



Resampling can break hydrologic connectivity

- Coarsening a HydroDEM can break hydrologic connectivity.
- Try not to get too much coarser than input HydroDEM, or perform extensive hydro-enforcement on you input DEM first.
- Here, a canyon outlet is filled, causing entire valley to fill during pit-filling process.

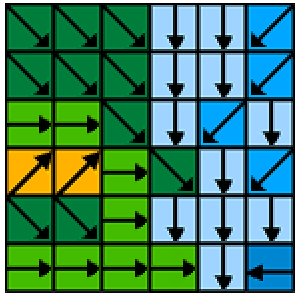


Flow Direction & Flow Accumulation

- D8 Flow Direction

32	64	128
16		1
8	4	2

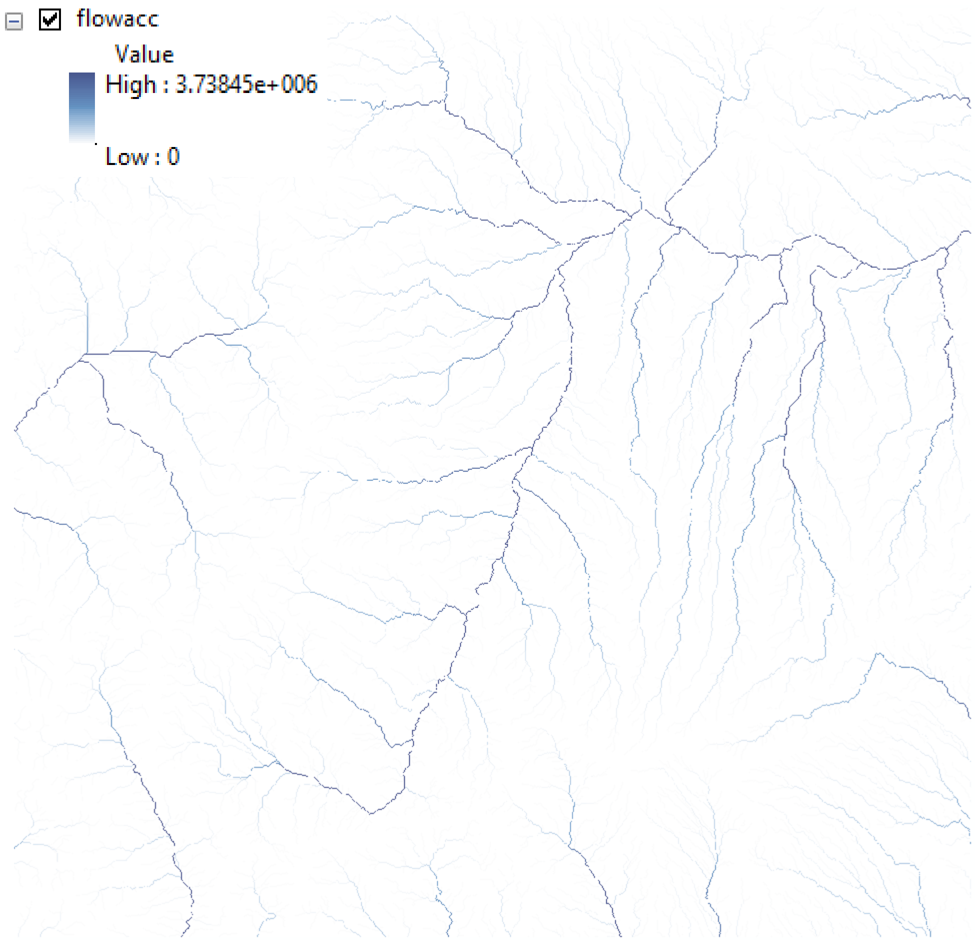
- Flow Accumulation



Flow direction

0	0	0	0	0	0
0	1	1	2	2	0
0	3	7	5	4	0
0	0	0	20	0	1
0	0	0	1	24	0
0	2	4	7	35	1

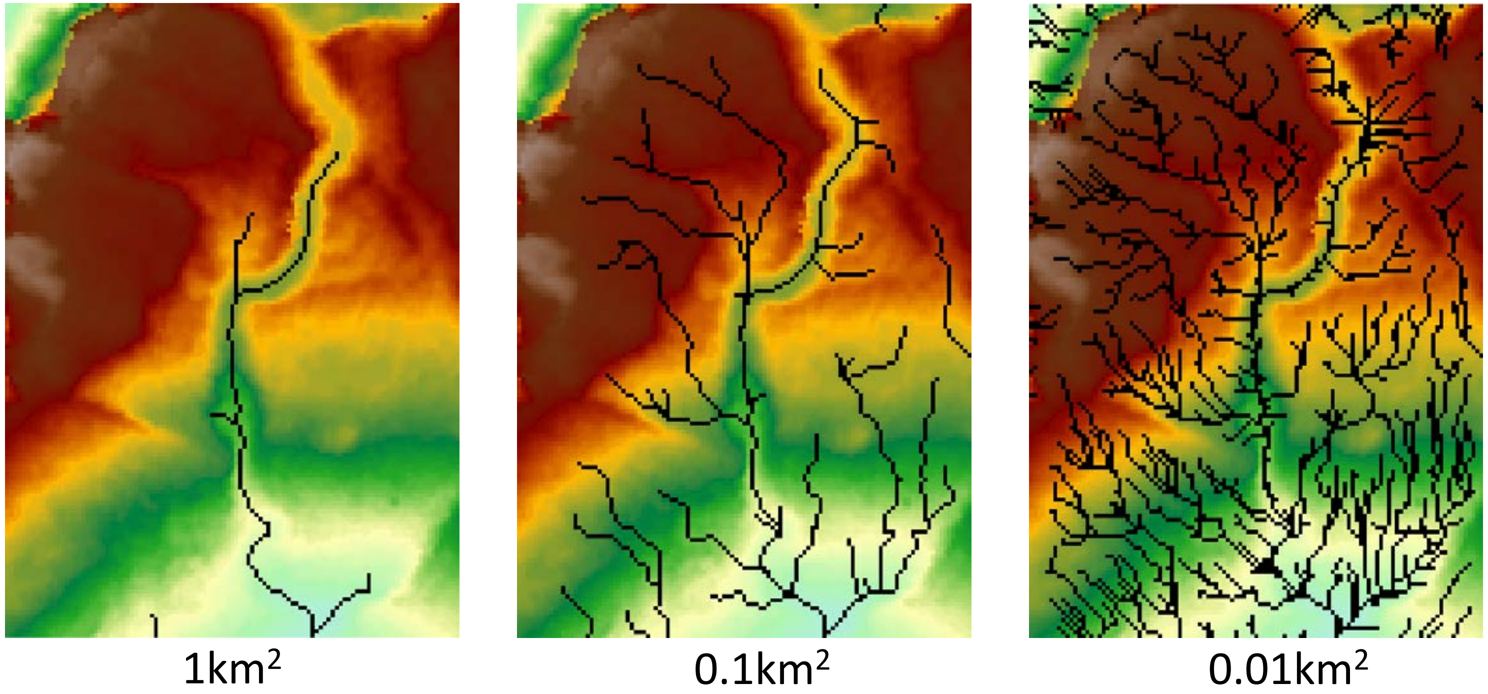
Flow accumulation



Methods: Jenson and Domingue (1988)

Stream Definition

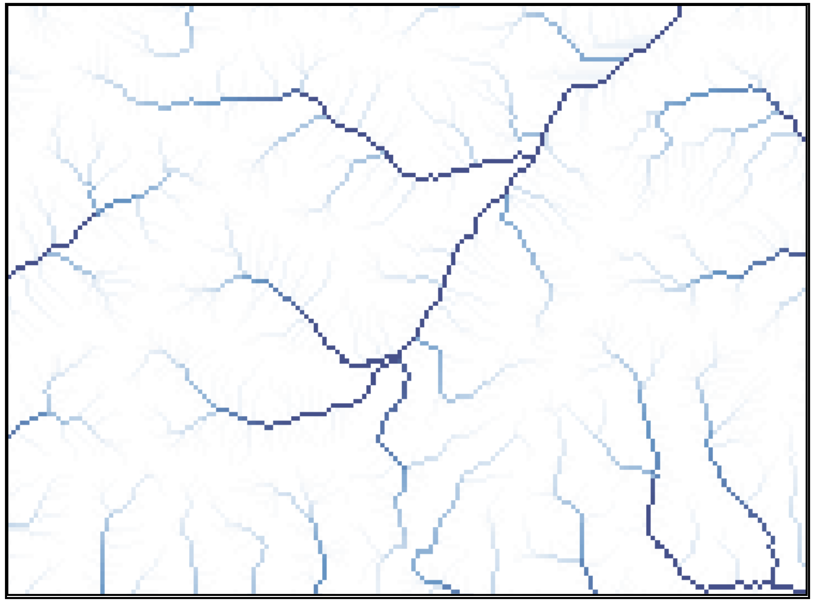
- Input Parameter: Number of pixels to define stream
  - Yields a minimum ‘basin’ size
  - Given in pixels (unitless), on the routing grid
  - Affects density of generated channel network



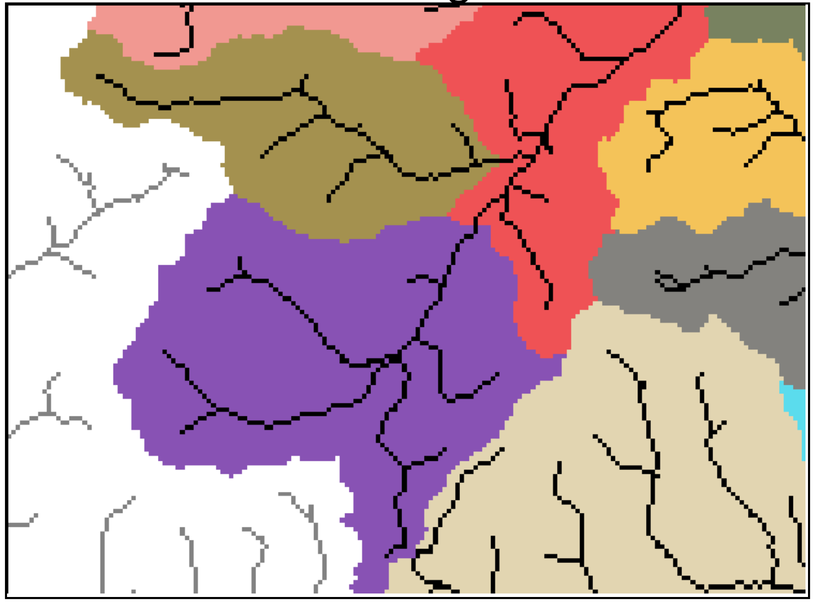
An analytic method for determining an appropriate threshold value for stream network delineation is presented in Tarboton et al. (1991)

Stream Definition

Flow Accumulation



Channelgrid



- Use flow accumulation threshold to define channels
- Option: use gaged basins as mask to assign CHANNELGRID values
- If reach-based routing is selected, **Stream to Feature** used to create vector geometry of streams
  - `streams.shp` shapefile written to output directory

Forecast Points

Input: Sta

FID_	LON	LAT	STATION	Name	HOST	ELEV	DRAIN_AREA_SQMI	DRAIN_AREA_SQKM
1	-103.79889	40.26861	S_PLATTE_at_FT_MORGAN	6759500	USGS	4260	14627	37883.93
2	-108.26556	39.23917	COLO_at_CAMEO	9095500	USGS	4813	8050	20849.5
3	-104.39861	38.24806	ARKANSAS_nr_AVONDALE	7109500	USGS	4509	6327	16386.93
4	-105.88002	37.481392	RIO_GRANDE_nr_ALAMOSA	8223000	USGS	-9999	0	0

```
FID,LON,LAT,STATION,Name
15,-105.92833,40.08139,Fraser_at_Granby,9033300
18,-105.9,40.12083,COLO_nr_GRANBY,9019500
20,-106.3333,39.8803,Blue_R_blw_Grn_Mtn,9057500
```

- Create in Excel, Numbers, Word, etc.
- Direct output of attribute table from shapefile or feature class
- “LON”, “LAT”, “FID” required
- If present, basins will be delineated using the points provided
  - `frxst_basns` output variable will be created
  - `frxst_pts` & `basin_msk` variables will be populated
- If masked to basins, CHANNELGRID will have values -1, 0, -9999

Process: Basin Delineation

- Snap points to streams
- 'Walk' down channel network a specified distance
  - Default = 3 pixel widths (global variable)
- Delineate basin using **Watershed** Spatial Analyst tool

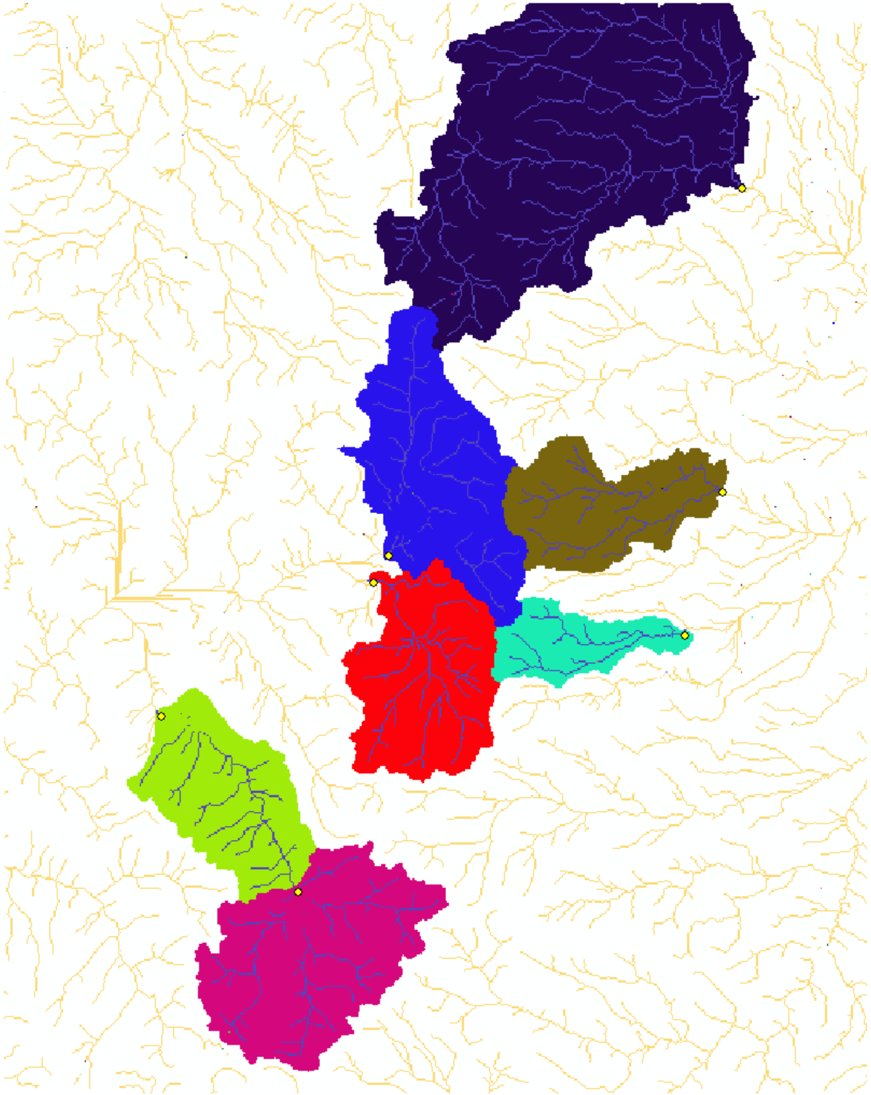
- Writes output file to:

  `frxst_pts`

  `basn_msk`

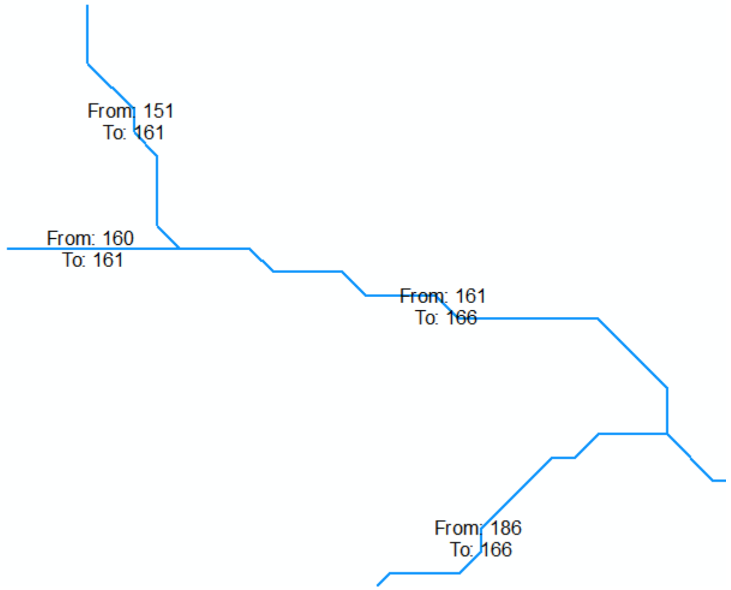
Basin Masking

- Option to ‘turn off’ channel networks outside gaged basins.
- If gages are provided and the option to mask CHANNELGRID to basins is selected.
    - Channel pixels inside gaged basins = 0
    - Channels outside = -1



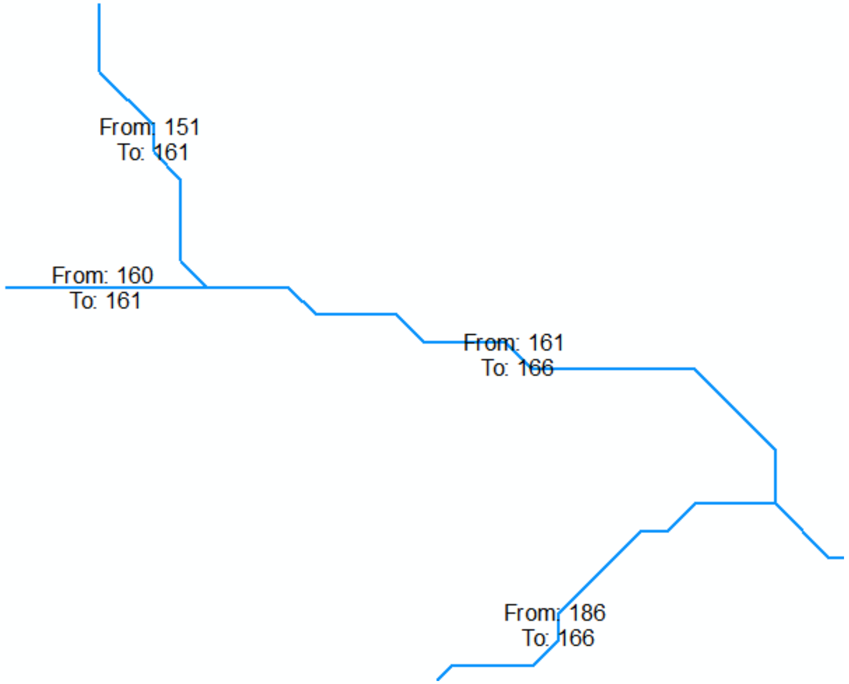
Reach-Based Routing Background

- A vector-based approach to routing flow
- Channel network is comprised of ‘links’ instead of pixels
- Derive the channel network automatically from the
- Muskingum-Cunge parameters applied to reaches
- With network topology defined, flow can be routed down reaches
- Computational efficiency vs. gridded methods



Process: Reach-Based Routing

- CHANNELGRID raster is converted to a line vector (`streams.shp`)
- Decomposes line geometry to nodes, and gathers elevation, Latitude, and Longitude at each node
- LINKID grid in Fulldom file is created and populated with link ID values
- Constructs a .nc file with necessary parameters for reach-based routing:
  - Length, Slope, Order, Drop, X/Y, etc.
  - Writes output file to `Route_Link.nc`



Reach-Routing Table

- CF-netCDF file containing reach-routing parameters
- Mix of derived attributes and default values

Parameter	Description
link	Link ID
from	From Link ID
to	To Link ID
lon	longitude of the start node
lat	latitude of the start node
alt	Elevation in meters from DEM at start node
order	Stream order (Strahler)
Qi	Initial flow in link (CMS)
MusK	Muskingum routing time (s)
MusX	Muskingum weighting coefficient
Length	Stream length (m)
n	Manning's roughness
So	Slope (meters/meter)
ChSlp	Channel side slope
BtmWdth	Bottom width of channel
Kchan	Channel conductivity (mm/hr)
x	X-coordinate in projected coordinate system
y	Y-coordinate in projected coordinate system

Route_Link.nc

- netCDF file to store link information
  - 1-Dimension (linkDim)
- CF-netCDF ‘timeSeries’ convention

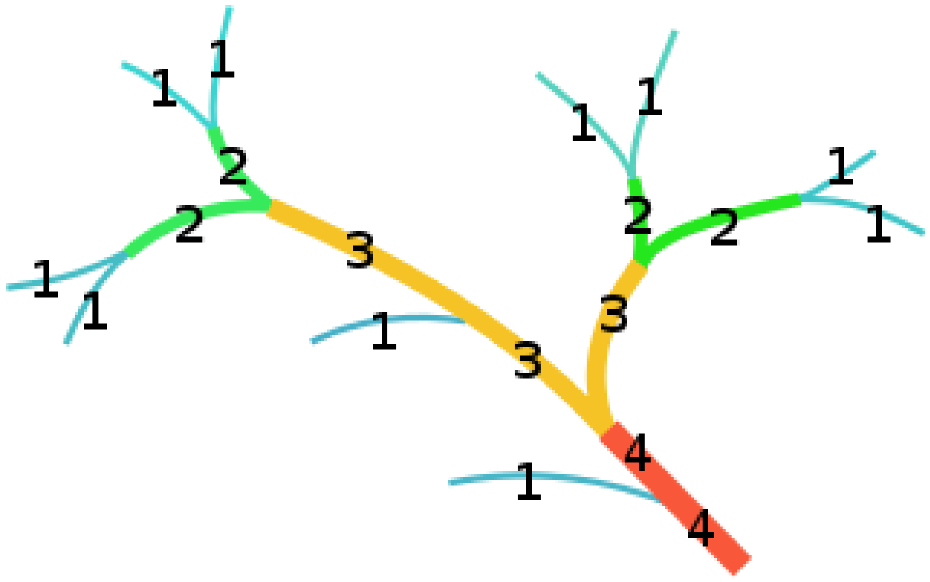
<u>Defaults</u>

Qi:	0 cms
MusK:	3600 s
MusX:	0.2
n:	0.035
ChSlp:	0.05
BtmWdth:	5 m
Kc:	0 mm/hr

```
netcdf Route_Link {
dimensions:
        linkDim = 5726 ;
variables:
        int link(linkDim) ;
                link:long_name = "Link ID" ;
                link:cf_role = "timeseries_id" ;
        int from(linkDim) ;
                from:long_name = "From Link ID" ;
                from:coordinates = "lat lon" ;
        int to(linkDim) ;
                to:long_name = "To Link ID" ;
                to:coordinates = "lat lon" ;
        float lon(linkDim) ;
                lon:long_name = "longitude of the start node" ;
                lon:units = "degrees_east" ;
                lon:standard_name = "longitude" ;
        float lat(linkDim) ;
                lat:long_name = "latitude of the start node" ;
                lat:units = "degrees_north" ;
                lat:standard_name = "latitude" ;
        float alt(linkDim) ;
                alt:long_name = "Elevation in meters at start node" ;
                alt:standard_name = "height" ;
                alt:units = "m" ;
                alt:positive = "up" ;
                alt:axis = "Z" ;
        int order(linkDim) ;
                order:long_name = "Stream order (Strahler)" ;
                order:coordinates = "lat lon" ;
        float Qi(linkDim) ;
                Qi:long_name = "Initial flow in link (CMS)" ;
                Qi:coordinates = "lat lon" ;
        float MusK(linkDim) ;
                MusK:long_name = "Muskingum routing time (s)" ;
                MusK:coordinates = "lat lon" ;
        float MusX(linkDim) ;
                MusX:long_name = "Muskingum weighting coefficient" ;
                MusX:coordinates = "lat lon" ;
        float Length(linkDim) ;
                Length:long_name = "Stream length (m)" ;
                Length:coordinates = "lat lon" ;
        float n(linkDim) ;
                n:long_name = "Manning\'s roughness" ;
                n:coordinates = "lat lon" ;
        float So(linkDim) ;
                So:long_name = "Slope (%; drop/length)" ;
                So:coordinates = "lat lon" ;
        float ChSlp(linkDim) ;
                ChSlp:long_name = "Channel side slope (%; drop/length)" ;
                ChSlp:coordinates = "lat lon" ;
        float BtmWdth(linkDim) ;
                BtmWdth:long_name = "Bottom width of channel" ;
                BtmWdth:coordinates = "lat lon" ;
        float time ;
                time:standard_name = "time" ;
                time:long_name = "time of measurement" ;
                time:units = "days since 2000-01-01 00:00:00" ;

// global attributes:
                :featureType = "timeSeries" ;
                :history = "Created Mon May 02 14:43:17 2016" ;
}
```

Process: Stream Order

- **Stream Order** Spatial Analyst tool
  - Strahler stream order
  - Writes output file to STREAMORDER variable



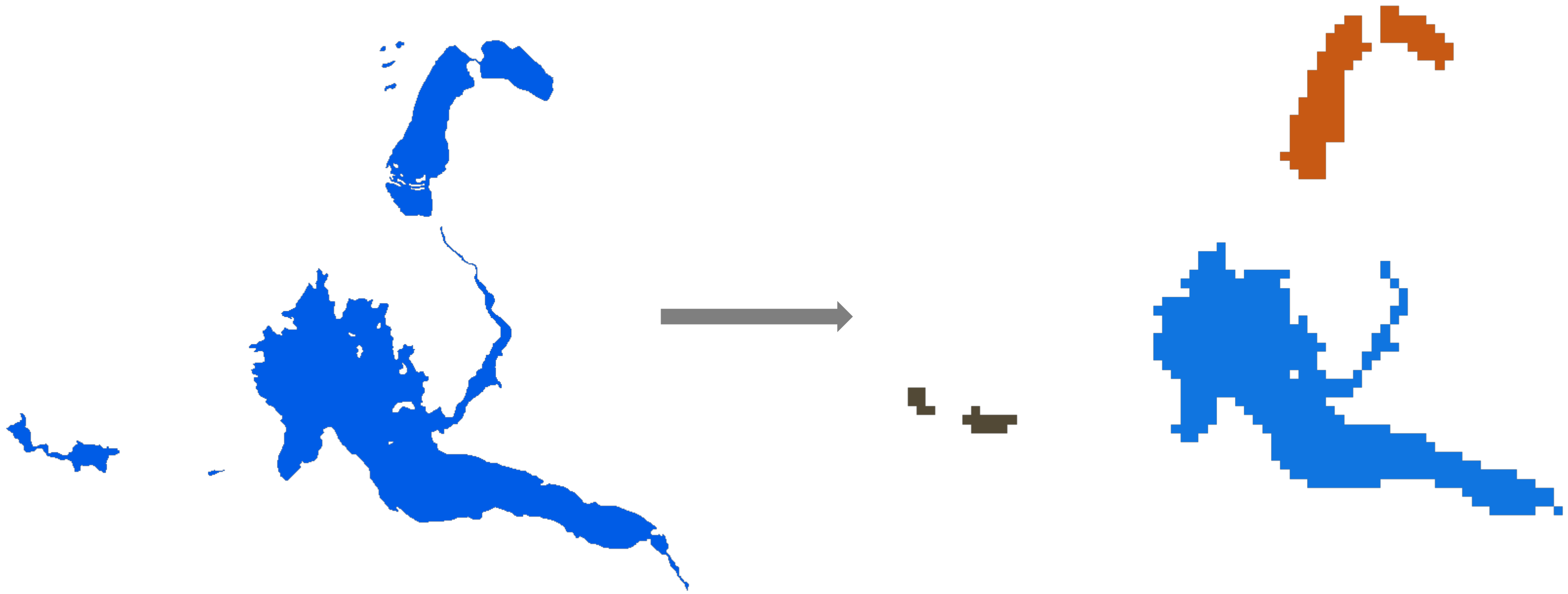
Process: Reservoir Routing

- If the option is selected, a polygon shapefile or feature class is required as input.
- Populates LAKEGRID variable
- Assigns lake ID values to pixels where lakes drain into channel
- Constructs a LAKEPARM.nc file with necessary variables for reservoir routing:
  - Lake area, max elevation, min elevation, base elevation, orifice elevation

```
netcdf LAKEPARM {
dimensions:
        nlakes = 82 ;
variables:
        int lake_id(nlakes) ;
                lake_id:long_name = "Lake ID" ;
                lake_id:cf_role = "timeseries_id" ;
        double LkArea(nlakes) ;
                LkArea:long_name = "Gridded lake area (sq. km)" ;
                LkArea:coordinates = "lat lon" ;
        double LkMxH(nlakes) ;
                LkMxH:long_name = "Maximum lake elevation (m ASL)" ;
                LkMxH:coordinates = "lat lon" ;
        double WeirC(nlakes) ;
                WeirC:long_name = "Weir coefficient" ;
                WeirC:coordinates = "lat lon" ;
        double WeirL(nlakes) ;
                WeirL:long_name = "Weir length (m)" ;
                WeirL:coordinates = "lat lon" ;
        double OrificeC(nlakes) ;
                OrificeC:long_name = "Orifice coefficient" ;
                OrificeC:coordinates = "lat lon" ;
        double OrificeA(nlakes) ;
                OrificeA:long_name = "Orifice cross-sectional area (sq. m)" ;
                OrificeA:coordinates = "lat lon" ;
        double OrificeE(nlakes) ;
                OrificeE:long_name = "Orifice elevation (m ASL)" ;
                OrificeE:coordinates = "lat lon" ;
        float lat(nlakes) ;
                lat:long_name = "latitude of the lake centroid" ;
                lat:units = "degrees_north" ;
                lat:standard_name = "latitude" ;
        float lon(nlakes) ;
                lon:long_name = "longitude of the lake centroid" ;
                lon:units = "degrees_east" ;
                lon:standard_name = "longitude" ;
        float alt(nlakes) ;
                alt:long_name = "vertical distance above mean sea level in (m ASL)" ;
                alt:standard_name = "height" ;
                alt:units = "m" ;
                alt:positive = "up" ;
                alt:axis = "Z" ;
```

LAKEGRID/LAKEPARM.nc

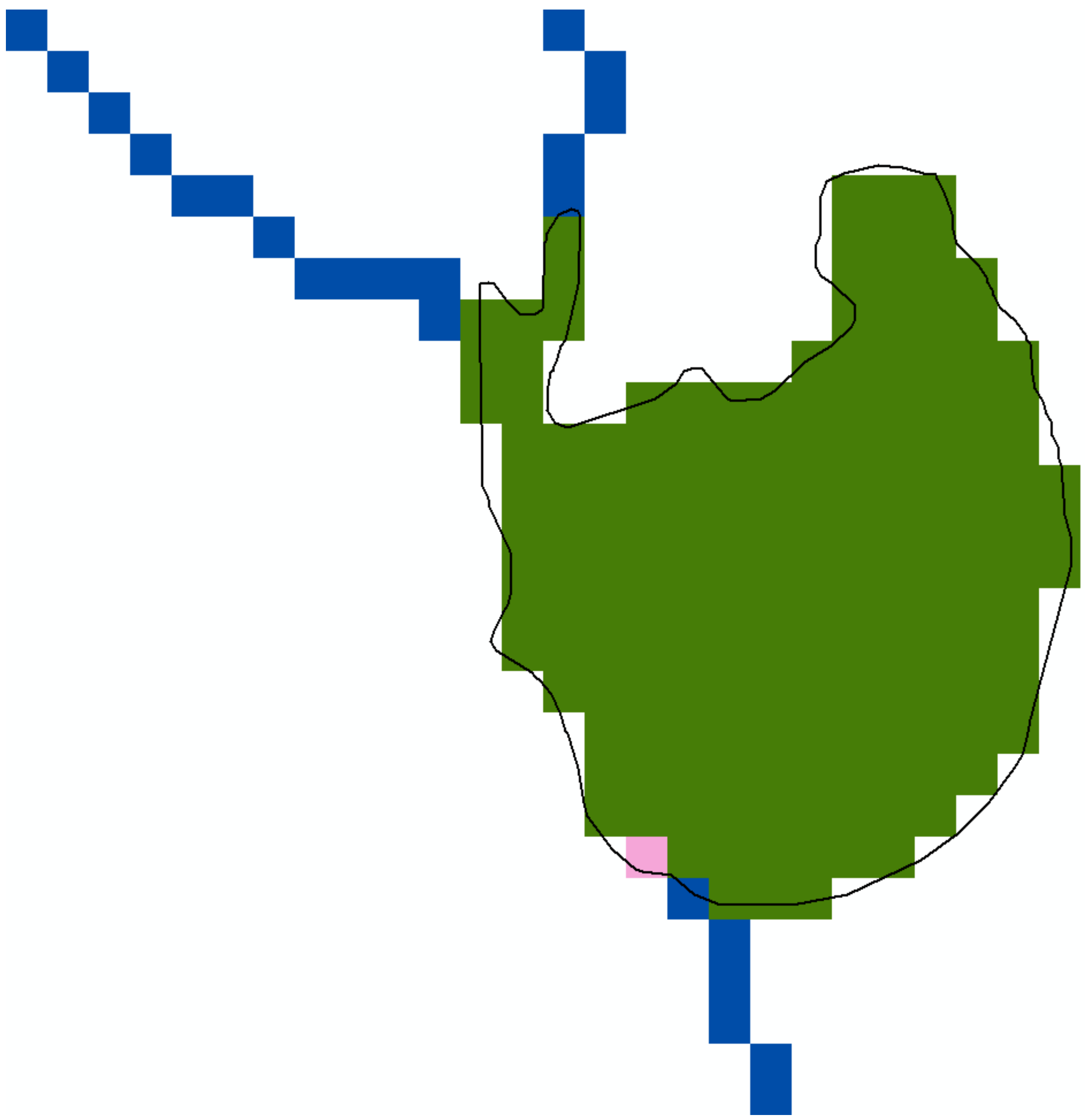
- Input: Reservoirs shapefile or feature class (polygon)
- Polygons are resolved on the model grid if they are large enough
- Lake ID is renumbered to 1-n



Original polygons

Lakes on the routing grid

Reservoirs & Channels



Lakes/Reservoir Routing in the NWM

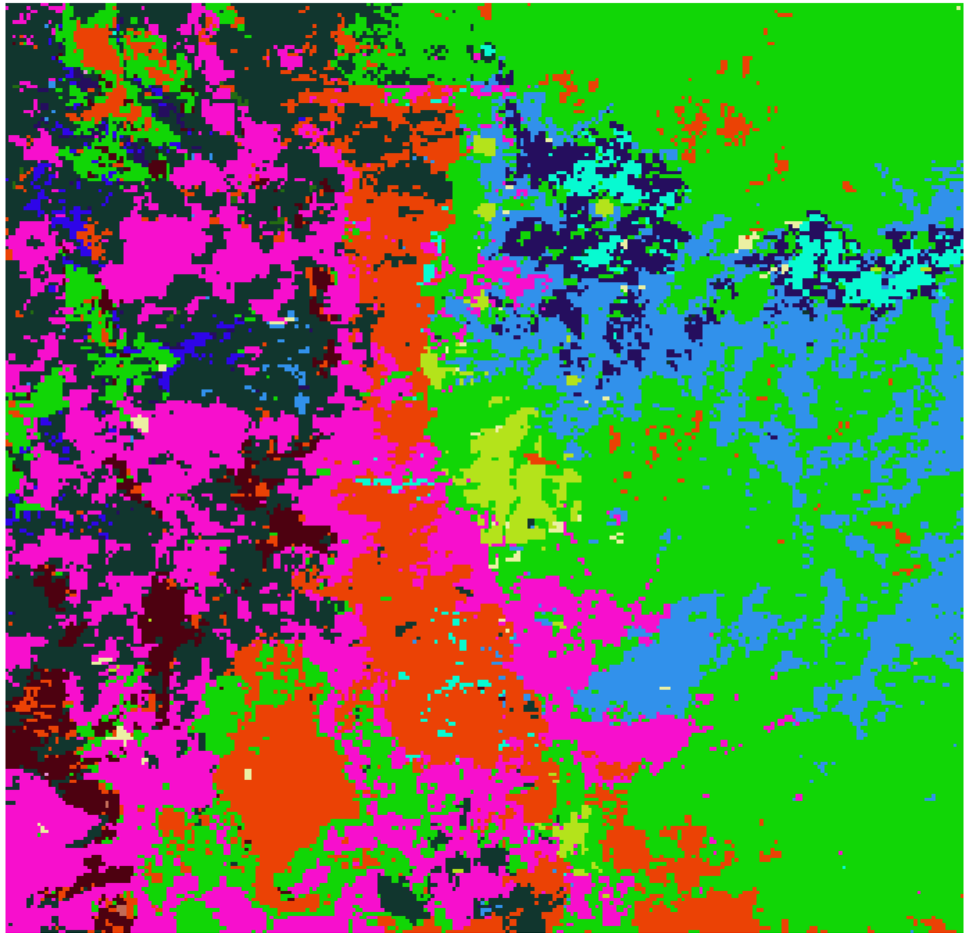
- Lake parameters are stored in the LAKEPARM.nc lake routing table
- Lakes are defined on the routing grid, parameters in the table

Lake Routing Table

Parameter	Description
LkArea	Gridded lake area (sq. km)
LkMxH	Maximum lake elevation (m ASL)
OrificeA	Orifice cross-sectional area (sq. m)
OrificeC	Orifice coefficient
OrificeE	Orifice elevation (m ASL)
WeirC	Weir coefficient
WeirH	Weir Height (m ASL)
WeirL	Weir length (m)
Ifd	Initial fractional water depth (% full)
lake_id	Lake ID
lat	latitude of the lake centroid
lon	longitude of the lake centroid

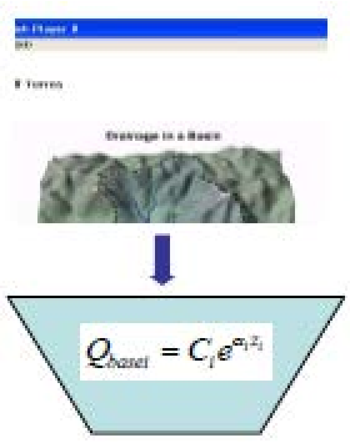
Other Grids

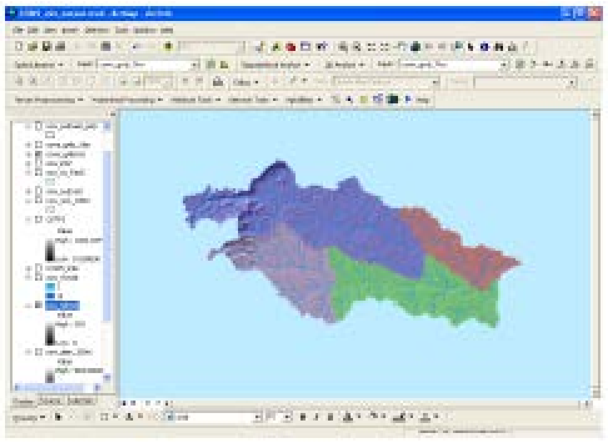
- Landuse
  - GEOGRID LU_INDEX resampled (nearest neighbor) to routing grid
- OVROUGHRTFAC
  - Constant 1.0 (float32)
- RETDEPRTFAC
  - Constant 1.0 (float32)
- LKSATFAC
  - Constant 1000.0 (float32)



Groundwater Buckets

- Conceptualized baseflow
- Spatially aggregated drainage from soil profile stored in ‘buckets’ representative of an aquifer
- GWBUCKPARM.nc bucket parameter file
- Buckets resolved on the coarse grid, written to a 2D netCDF file
  GWBASINS.nc





Basin	Coeff.	Expon.	Zmax	Zinit
1	1.0000	3.000	150.00	10.0000
2	1.0000	3.000	250.00	40.0000
3	1.0000	3.000	150.00	30.0000
4	1.0000	3.000	100.00	20.0000
5	1.0000	3.000	100.00	50.0000

From WRF-Hydro User Guide, Figure 3.7

Groundwater Representation

- Groundwater Bucket Parameters
  - Built using default groundwater bucket parameters combined with LINKID-based local contributing basins.
  - Other methods available to produce groundwater basins from

    Forecast Points

    Polygon Shapefile

Windows PowerShell

```
PS C:\Data\Projects\Gochis\NWM_v1_2\GW_BUCKPARM> ncdump -h .\GWBUCKPARM_NWM_v1_2_20170320_sorted.nc
netcdf GWBUCKPARM_NWM_v1_2_20170320_sorted {
dimensions:
        BasinDim = 2677135 ;
variables:
        int Basin(BasinDim) ;
                Basin:long_name = "Basin monotonic ID (1...n)" ;
        float Coeff(BasinDim) ;
                Coeff:long_name = "Coefficient" ;
        float Expon(BasinDim) ;
                Expon:long_name = "Exponent" ;
        float Zmax(BasinDim) ;
                Zmax:long_name = "Zmax" ;
        float Zinit(BasinDim) ;
                Zinit:long_name = "Zinit" ;
        float Area_sqkm(BasinDim) ;
                Area_sqkm:long_name = "Basin area in square kilometers" ;
        int ComID(BasinDim) ;
                ComID:long_name = "NHDCatchment FEATUREID (NHDFlowline ComID)" ;

// global attributes:
                :featureType = "point" ;
                :history = "Created Mon Mar 20 16:35:22 2017" ;
}
```

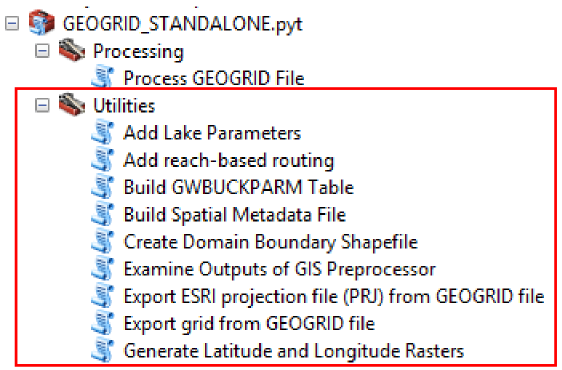
GIS Pre-processor Outputs

- Set of netCDF, shapefile, ASCII & log files
  - 2-6 netCDF files
  - 0-1 ASCII Raster (.txt)
  - 0-2 Shapefiles
  - 1 .log file

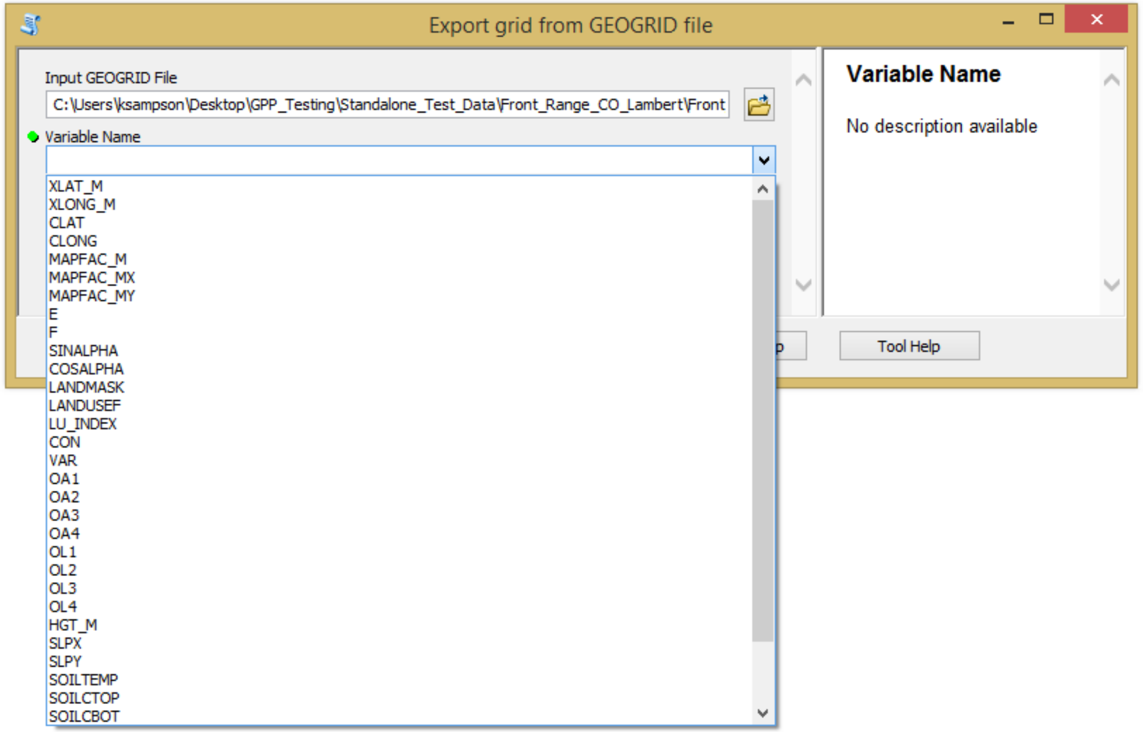
- FRNG_1km_250m_Basic
  - Fulldom_hires.nc
  - GEOGRID_LDASOUT_Spatial_Metadata.nc

- FRNG_1km_250m_TestData_Full
  - Fulldom_hires.nc
  - GEOGRID_LDASOUT_Spatial_Metadata.nc
  - gw_basns_geogrid.txt
  - gw_basns_geogrid.prj
  - LAKEPARM.nc
  - lakes.shp
  - Route_Link.nc
  - streams.shp

Other Utilities

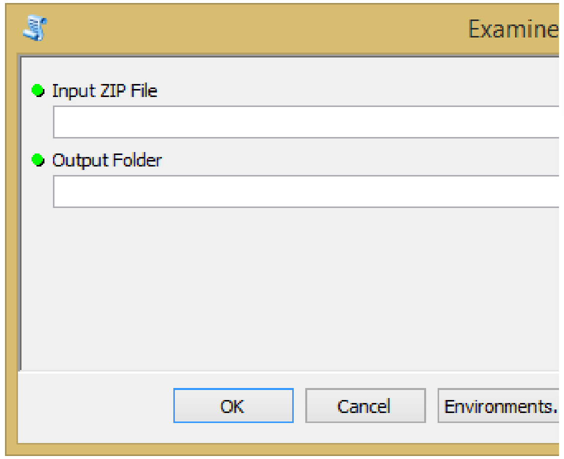


- Examine Outputs of GIS Preprocessor
  - Extracts .zip output file to individual rasters for viewing in Desktop GIS applications.



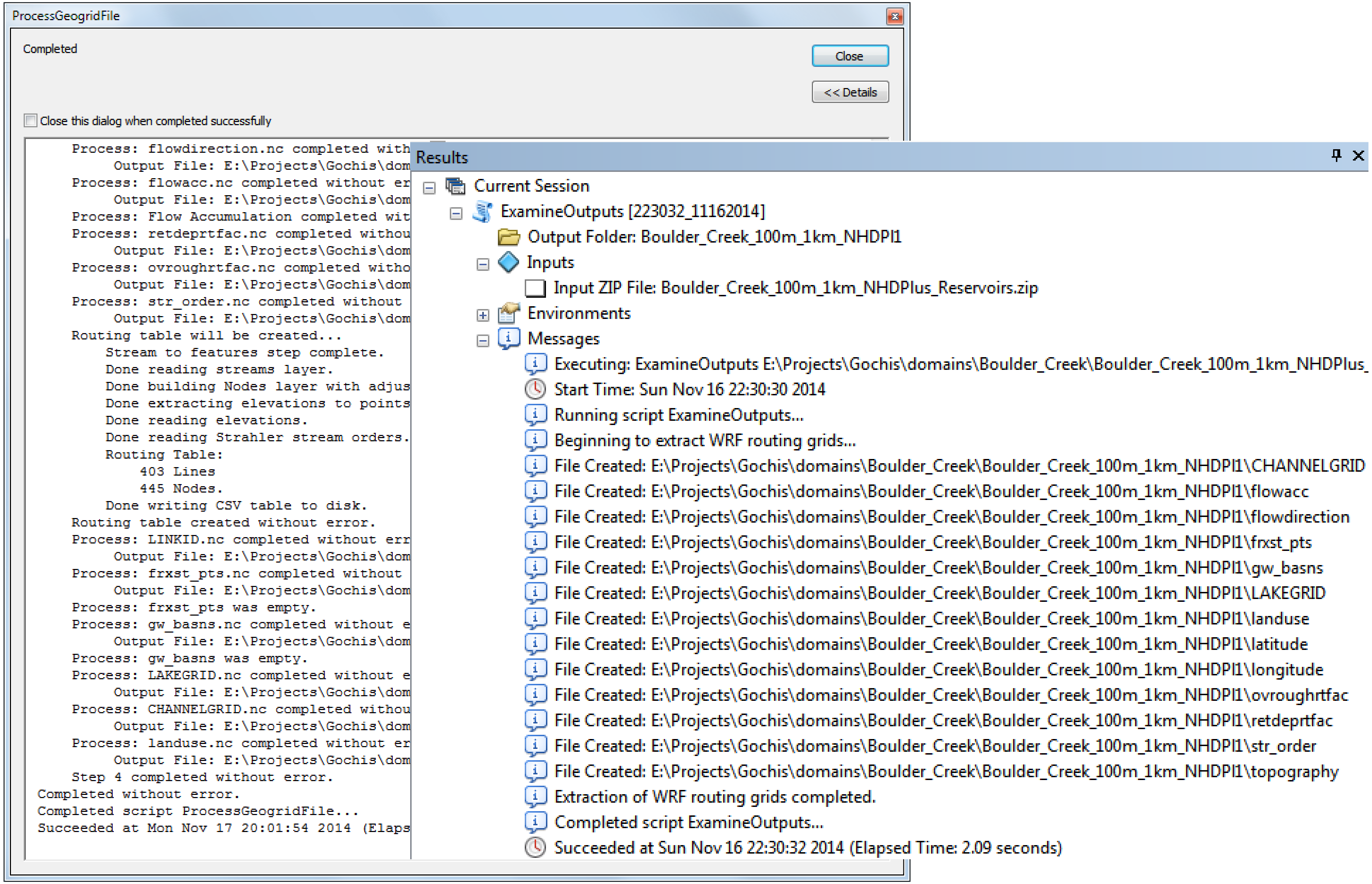
file

ster



  - Creates a polygon shapefile defining the domain boundary from a GEOGRID file
- Build Groundwater Inputs
  - Creates groundwater input files in 3 ways

Tool Messages



Documentation & Test Data

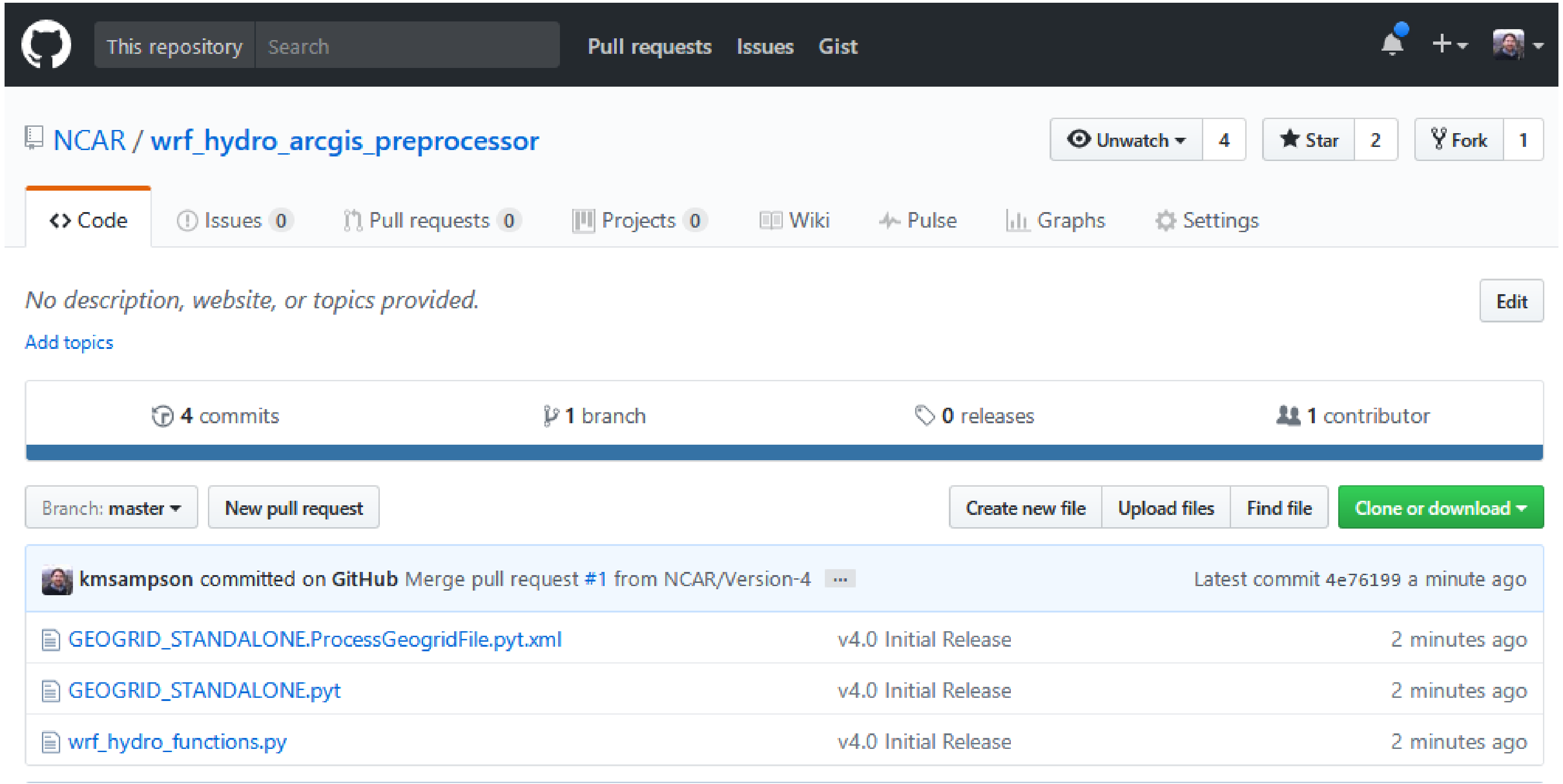
- Detailed documentation
  - 40+ page PDF
  - Describes tool capabilities, requirements, parameters, and GIS methods used in the tool chain.
- Small GEOGRID domains for testing tool functionality
  - Front Range (Lambert Conformal Conic)
  - India (Mercator)
- Expected Output provided for comparison
- Required Elevation files (.tif) provided
- Optional stream gages & lakes provided (Front Range)

https://ral.ucar.edu/projects/wrf_hydro/pre-processing-tools

Bottlenecks/Constraints

- Project high-resolution dataset for large areas
  - Can be avoided by pre-projecting/resampling high res data before running the GIS pre-processor
- Flow Accumulation – slowest part of the process
- Not multi-threaded
  - Process runs on one core
  - Process chain not well suited to parallelization
- Windows only GIS platform

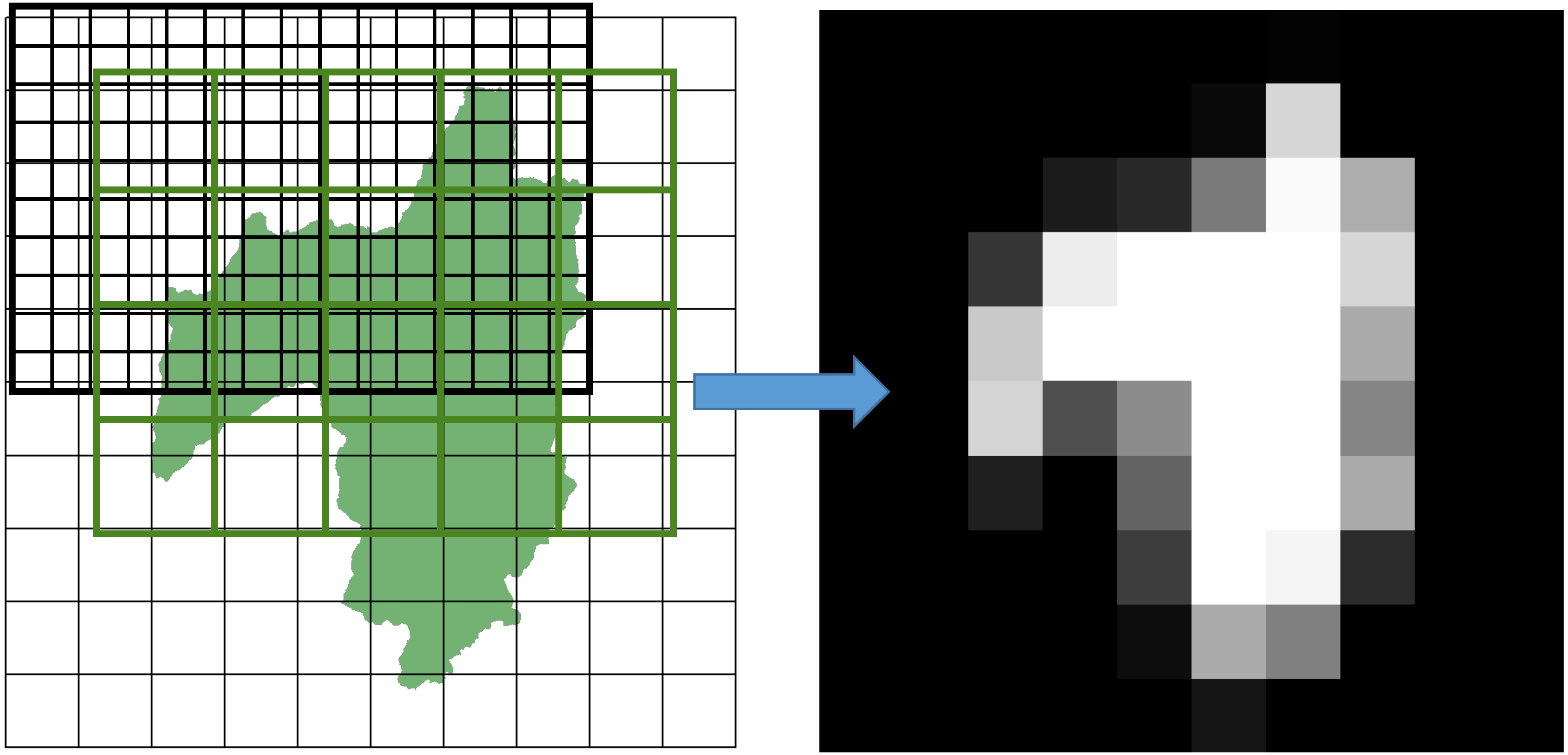
- https://github.com/NCAR/wrf_hydro_arcgis_preprocessor.git



Catchments & Grid-to-Basin Mapping

Grid-to-Basin Mapping

fraction of each grid cell in each sub-basin

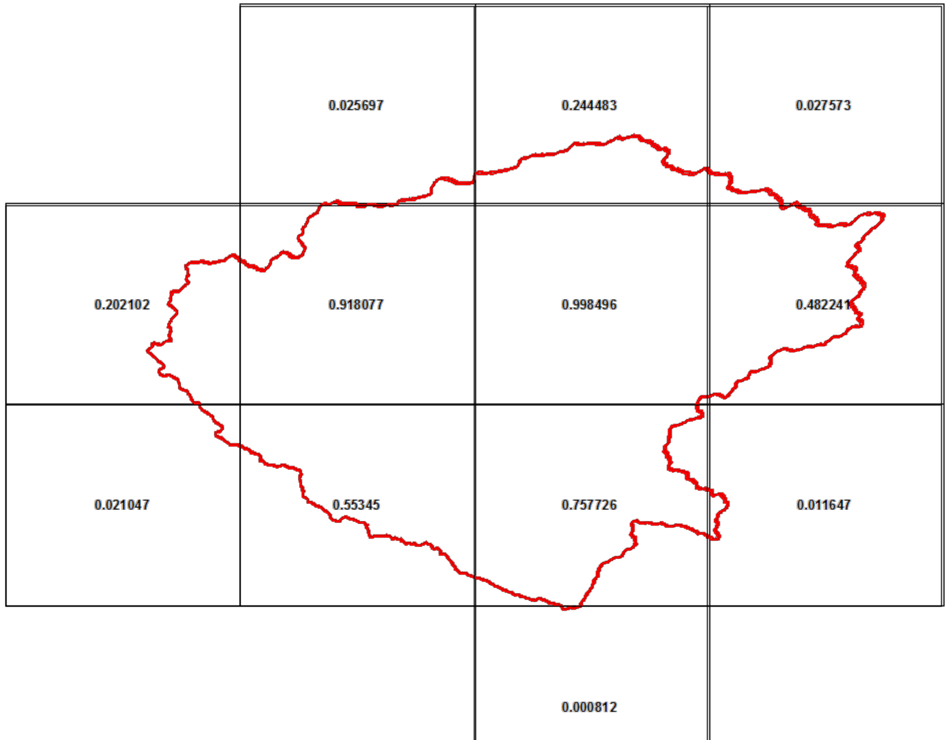


maps gridded meteorological fields to sub-basins within the model domain

Grid-to-Basin Mapping

- Correspondence between basins and grid must be established
- Spatial weights allow conservative remapping of variables between grids and catchments
- This method enables the NHD reach-routing scheme by moving overland and subsurface flow into the reach associated with each catchment
- Custom, open-source, parallel Python tools for generating the mapping.

Spatial Weight



Intersections:
1km: 21.5M
250m: 172M

Custom Correspondence netCDF

Custom Correspondence netCDF

```
netcdf spatialweights_1km_v1_2_all_basins_20170320 {
dimensions:
        polyid = 2677135 ;
        data = 21508911 ;
variables:
        int polyid(polyid) ;
                polyid:long_name = "ID of polygon" ;
        int overlaps(polyid) ;
                overlaps:long_name = "Number of intersecting polygons" ;
        double weight(data) ;
                weight:long_name = "fraction of polygon(polyid) intersected by polygon identified by poly2" ;
        double regridweight(data) ;
                regridweight:long_name = "fraction of intersecting polyid(overlapper) intersected by polygon(polyid)" ;
        int IDmask(data) ;
                IDmask:long_name = "Polygon ID (polyid) associated with each record" ;
        int i_index(data) ;
                i_index:long_name = "Index in the x dimension of the raster grid (starting with 1,1 in LL corner)" ;
        int j_index(data) ;
                j_index:long_name = "Index in the y dimension of the raster grid (starting with 1,1 in LL corner)" ;

// global attributes:
                :history = "Created Wed Dec 23 10:17:36 2015" ;
                :processing_notes = "Correspondence between 2,647,250 NHDPlus Catchment basins (full resolution) as well
as O-CONUS basins and the CONUS 1km IOC grid." ;
}
```

GIS Pre-processing Demonstration

Thank you

Questions: wrfhydro@ucar.edu