THE LAND SURFACE DATA TOOLKIT (LDT): TUTORIAL AND TEST CASES

The "preprocessor" to LIS-7.x

Aug 31, 2015

The Land Data Toolkit (LDT)

- A new preprocessing toolkit for LIS's model parameters
  - Processes and groups parameters needed for each land surface model (LSM)
  - Multiple processing options
- Observation-based data assimilation options
  - CDF-matching, etc.
- Generate custom-made restart files for LSMs
  - e.g., ensemble-based restart files

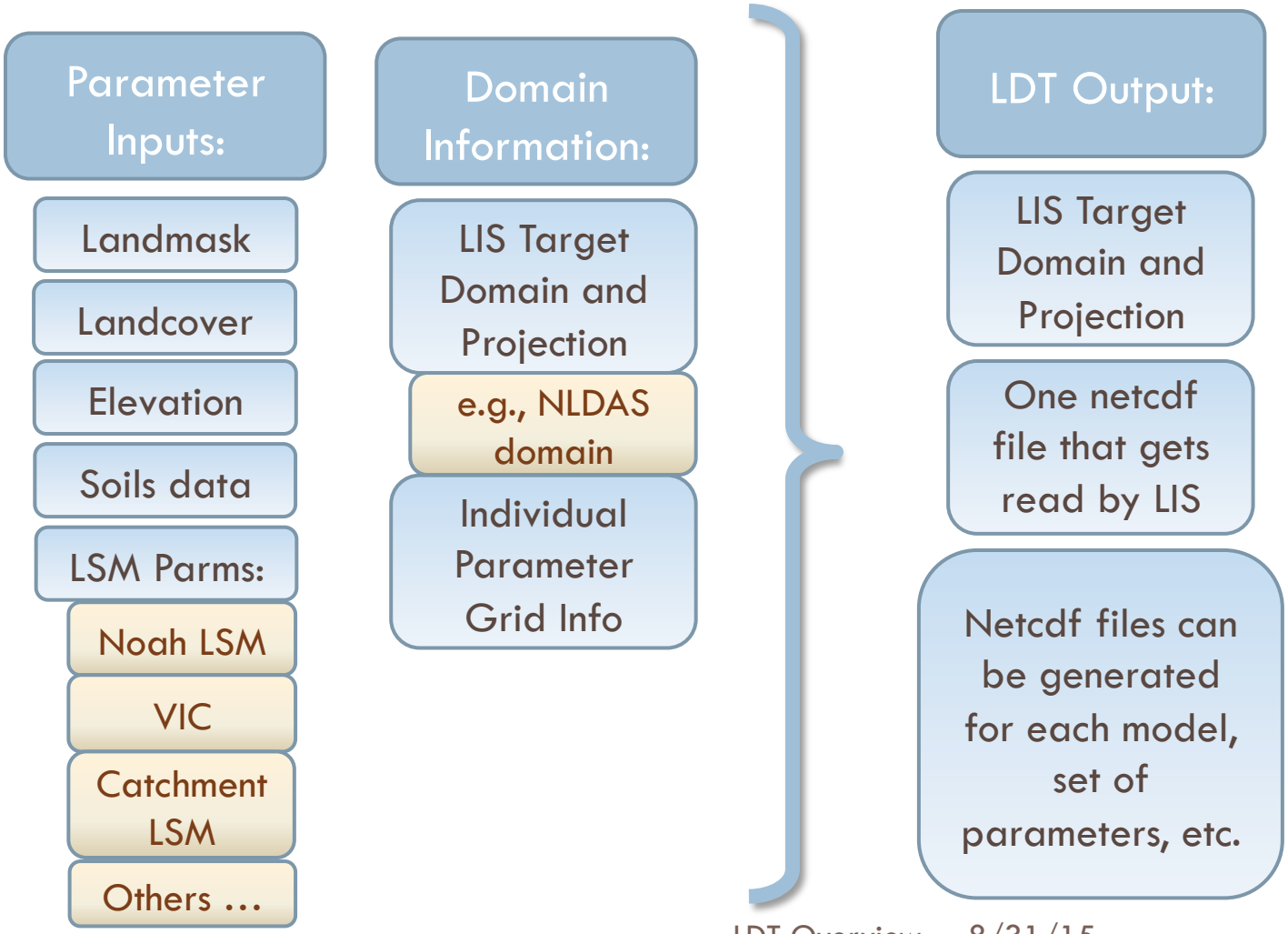
LDT Parameter Preprocessing

- Offers subsetting, reprojecting, aggregating/ downscaling and new tiling options for parameter files onto the targeted LIS-7 grid and domain;
- Group parameter files together in to a common Netcdf 4 file, which includes header and grid information for the data layers;
- This grouped netcdf file can then be easily loaded and read by LIS-7 at run-time;
- Building into LDT also LSM run-time checks (e.g., correct parameters selected) and parameter statistics (e.g., land/ water mask gridcell effects)

LDT Data Input “Philosophy”

- With the advent of LDT, we are introducing a "new" philosophy that data and model parameter files should be read in from their “native” grids and file formats and be written to a common descriptive data format, like NetCDF.
- With that, LDT is being developed to read in the "native" or raw original data files as how they are provided by a data center, government agency, university group, etc.
- In the past, the LIS team processed several different parameters onto a common binary format standard, which were provided to the community, and typically referred to as “LIS-data”.

The Land Data Toolkit (LDT): Parameter Processing



LDT Data Assimilation Observation Processing

For DA preprocessing, 3 options are supported:

- Generate cumulative density functions (CDFs) ➔ for CDF scaling of observations within LIS;
- Generate mean and standard deviations for normal-deviate based scaling within LIS;
- Anomaly correction to observations (for GRACE-DA primarily).

LDT Restart File Generation

(featuring ensemble restart generation)

- For ensemble restarts, two current options:
  - *Upscaling*: going from one to many ensemble members, and
  - *Downscaling*: going from many members to one.
- Future restart generation options may include:
  - Climatological restart generation,
  - Spatial aggregation/downscaling of existing restart files

Current LDT Capabilities

- Able to read in the original LIS-team produced parameter and observation files (aka, "LIS-data") with binary format of:
  - Direct-access, 4-byte real, big-endian binary …
  - Lat-lon gridded files with resolutions starting at 0.01° (~1KM), which can be aggregated to other coarser resolutions and projections;
- Moving towards reading in all "Native" or raw parameter files, obtained from the original institutional sources.
- Checks performed on parameter file processing;
  - e.g., Ensure sand+silt+clay fractions == 1
  - e.g., Parameter-landmask land/water agreement.

Other LDT features (as originally in LIS)

- Handle different LIS projections:
  - Lat-lon, Lambert, Polar stereographic, etc.
  - And *interpolation* routines as available in LIS;
- Working toward reading in all original LSM parameter files (of LSMs available in LIS-7);
- Writes run-time diagnostic information to a log file, like in LIS;
- Works with multiple member ensembles;
- Run multiple domain nests

SOME CURRENT FEATURES

Main LDT Configuration File Options

LDT Main Run Modes

LDT Option	Option	Description
LDT Running Mode:	*LSM parameter processing*	Processes parameters required for a given model run within LIS and outputs to a common netcdf file
	DA preprocessing	Prepares certain bias correction statistics and observation processing for data assimilation (DA) runs in LIS
	Ensemble restart processing	Generates ensemble restart files, e.g., netcdf-4 file with header info
	Metforce processing	Can process meteorological forcing files, like in LIS-7
	Metforce temporal downscaling	Temporally downscale coarser time-step forcings with finer scale ones; works mostly with precip. for now
Processed LSM parameter filename:	*e.g., lis_input.d01.nc*	Generated LSM parameter file in netcdf-4 format, required for running LIS-7.

LIS Output Grid Options

LDT Option	Option	Description
Map projection of the LIS domain:	*latlon, lambert, polar, hrap, gaussian, mercator*	Different output projection options on which to write the parameters for a LIS experiment.
Lat/Lon Rectangular Grid Dimensions (Extents)		
Run domain lower left lat/lon:	*e.g., 40.125; -94.875*	Enter the lowest-left corner extent latitude and longitude values for your LIS target grid
Run domain upper right lat/lon:	*e.g., 49.875; -75.125*	Enter the upper-right corner extent latitude and longitude values for your LIS target grid
Run domain resolution (dx/dy)	*e.g., 0.25; 0.25*	Enter the x- and y-direction gridcell resolution for your LIS target grid

The *LDT.config* file

```
LDT running mode:            "LSM parameter processing"  # LDT type of run-mode (top-level option)

Processed LSM parameter filename:  ./lis_input.d01.nc   # Final output file read by LIS-7

LIS number of nests:                  1                 # Total number of nests run by LIS
Number of surface model types:        1                 # Total number of desired surface model types
Surface model types:                 "LSM"              # Surface models:  LSM | Openwater
Land surface model:                  "Noah.3.3"         # Enter LSM(s) of choice
Lake model:                          "none"             # Enter Lake model(s) of choice
Water fraction cutoff value:          0.5               # Fraction at which gridcell is designated as 'water'

Number of met forcing sources:        0                 # Enter number of forcing types
Met forcing sources:                 "none"             # Enter 'none' if no forcing selected
Met spatial transform methods:      bilinear            # bilinear | budget-bilinear | neighbor | average
Topographic correction method (met forcing):  "none"    # none | lapse-rate

LDT diagnostic file:                  ldtlog            # Log-based diagnostic output file
Undefined value:                    -9999.0             # Universal undefined value
Metrics output directory:            OUTPUT             # If metrics or stats are written out
Number of ensembles per tile:         1                 # The number of ensemble members per tile

# Processor layout (currently not available)
Number of processors along x:      1
Number of processors along y:      1

# LIS domain:  (See LDT User's Guide for other projection information)
Map projection of the LIS domain:    latlon
Run domain lower left lat:           25.0625
Run domain lower left lon:         -124.9375
Run domain upper right lat:          52.9375
Run domain upper right lon:         -67.0625
Run domain resolution (dx):          0.125
Run domain resolution (dy):          0.125
```

Land cover and mask options

LDT Option	Option	Description
Land cover classification:	*UMD, IGBPNCEP, USGS, and others*	Enter the land cover/land use classification scheme, which varies with different sources
Create or readin landmask:	*create*	Creates a landmask from the landcover file
	readin	Reads in a landmask file
Landcover map projection:	*latlon, and others (depending on the parameter read in)*	Projection/grid type for the input land cover parameter file.
Landcover spatial transform:	*e.g., tile, mode, nearest neighbor*	Mode for transforming the input parameter projection/grid to the LIS output (run-time) grid
Landcover [grid-domain inputs]:	*For example,* *Landcover lower left lat: -59.995* *Landcover lower left lon: -179.995*	Similar to how LIS run domain extents were specified

Current LDT Processing of Landcover and Masks

Two Major User-Given Options

1. Read in "LIS-Data" Landcover and Landmask (original LIS approach)

1. Read in *landcover and mask* and subset to desired LIS target output grid (current approach in LIS-6)
2. Apply mask to subsequent routines and parameters

2. "Native" Landcover and "Create" Landmask

1. Read in *landcover* (e.g., @ 0.0833° res.) and aggregate or interpolate to LIS target grid
2. Create new land/water mask from new mapped-to-LIS-grid land cover field
3. Apply mask to subsequent routines and parameters

The *LDT.config* file (con't.)

Original LIS-team produced data entries:

```
# Landcover/Mask Parameter Inputs
Landcover classification:     "IGBPNCEP"          # Enter land cover classification type
Landcover file:       ../noah_lisparms_testcase/UMD/1KM/landcover_IGBP_NCEP.1gd4r  # Landcover map path
Landcover spatial transform:    tile              # none | mode | neighbor | tile
Landcover fill option:          none              # none | neighbor (Not needed if creating landmask)
Landcover map projection:       latlon
# For "LIS" based files, must enter grid info:
Landcover lower left lat:      -59.995            # For "LIS" based files, must enter grid info
Landcover lower left lon:     -179.995
Landcover upper right lat:      89.995
Landcover upper right lon:     179.995
Landcover resolution (dx):       0.01
Landcover resolution (dy):       0.01
```

"Native" or raw data entries:

```
# Landcover/Mask Parameter Inputs
Landcover classification:       "IGBPNCEP"          # Enter land cover classification type
Landcover file:              ./input/igbp.bin       # Landcover map path
Landcover spatial transform:      tile              # none | mode | neighbor | tile
## Note: The fill entries here will 'fill' a missing landcover value with a neighboring one or assigned value.
Landcover fill option:          neighbor            # none | neighbor (Not needed if creating landmask)
Landcover fill radius:           3.                 # Number of pixels to search for neighbor
Landcover fill value:            5.                 # Static value to fill where missing
Landcover map projection:       latlon              # Landcover map projection
```

* Options for ensuring agreement between landmask and landcover (and other parameters).

The *LDT.config* file (con't.)

Read in Landmask entries:

```
# Create landmask field from readin landcover map or read in separate landmask file
Create or readin landmask:      "readin"              # create | readin
Landmask file:           ./input/landmask_UMD.1gd4r  # Land mask file (if needed to be read-in)
Landmask spatial transform:       mode                # none | mode | neighbor
## Note: The 'mode' entry here will determine the dominant mask value for each gridcell. ##
Landmask map projection:          latlon              # Landmask map projection
Landmask lower left lat:         -59.995              # Extents of the 1KM UMD landmask
Landmask lower left lon:        -179.995
Landmask upper right lat:         89.995
Landmask upper right lon:        179.995
Landmask resolution (dx):          0.01
Landmask resolution (dy):          0.01
```

"Creating" Landmask from Landcover entry:

```
# Create landmask field from readin landcover map or read in separate landmask file
Create or readin landmask:    "create"           # create | readin
Landmask file:                  none             # Land mask file (if needed to be read-in)
Landmask spatial transform:     none             # none | mode | neighbor
Landmask map projection:       latlon
```

Spatial Transform Options

LDT Option	Option	Description
[Parameter] spatial transform*:	**none**	No scaling or transformation needed.
	average	Average finer resolution grid to coarser scale one (*upscale*).
	mode	Select dominant (mode) of finer scale gridcells to coarser one (*upscale*).
	tile	Apply accounting system for estimating coarser sub-grid "tiles" for output grid.
	neighbor	Apply nearest neighbor method to input grid (*upscale/downscale*).
	bilinear	Apply bilinear interpolation method to input grid (*downscale*).
	budget-bilinear	Apply conservative "budget" bilinear interpolation method to input grid (*downscale*).
	**** NOTE: Not all options available yet for all parameters.***	

Fields that can be tiled (or binned) …

LDT Option	Option	Description
Land cover	*Number of vegetation tiles*	Derives fractions for vegetation type-tiles (summing to 100%)
Lake surface	*Number of lakes (as relates to FLake model at this time)*	Lake fractions can be estimated from lake depth or lake type
Soil fractions	*Fraction of sand, silt and clay*	Soil fraction bins (or tiles) along with average fractions per bin
Soil texture	*Soil texture tiles*	Derives fractions for soil texture types
Elevation	*Elevation tiles (aka, "bands")*	Average elevations associated with ranked tiles or "bands"
Slope, Aspect	*Tile fractions and associated average slope or average aspect*	Average slope or aspect associated with ranked tiles

Major Update:

The ***param_attribs.txt*** file is

NO LONGER USED!

LDT Data Assimilation Obs Inputs

- **Inputs:**
  - Enter DA observation type (several options)
  - Specify the number of bins to estimate the CDF
  - Different temporal and spatial map (mask) options for CDF stats
  - Observation count in estimation

```
LDT running mode:                       "DA preprocessing"
DA observation source:                  "RT SMOPS soil moisture"
Apply anomaly correction to obs:        0
Compute CDF:                            1
Number of bins to use in the CDF:       100
Name of CDF file:                       'rtsmops_cdf'
Temporal averaging interval:            "1da"
Apply external mask:                    0
External mask directory:                none
Observation count threshold:            400 #20 % of the record

RT SMOPS soil moisture observation directory:  ../RT_SMOPS/
RT SMOPS soil moisture use ASCAT data:         1
```

Example NetCDF Output Entries

(e.g., *lis_input.d01.nc*)

```
netcdf original_lis_input.d01 {
dimensions:
        east_west = 464 ;
        north_south = 224 ;
        month = 12 ;
        time = 1 ;
        sfctypes = 20 ;
        soiltypes = 16 ;
variables:
        float time(time) ;
        float LANDMASK(north_south, east_west) ;
                LANDMASK:standard_name = "MODIS-IGBP land mask" ;
                LANDMASK:units = "-" ;
                LANDMASK:scale_factor = 1.f ;
                LANDMASK:add_offset = 0.f ;
                LANDMASK:missing_value = -9999.f ;
                LANDMASK:vmin = 0.f ;
                LANDMASK:vmax = 0.f ;
                LANDMASK:num_bins = 1 ;
        float SURFACETYPE(sfctypes, north_south, east_west) ;
                SURFACETYPE:standard_name = "Surface type" ;
                SURFACETYPE:units = "-" ;
                SURFACETYPE:scale_factor = 1.f ;
                SURFACETYPE:add_offset = 0.f ;
                SURFACETYPE:missing_value = -9999.f ;
                SURFACETYPE:vmin = 0.f ;
                SURFACETYPE:vmax = 0.f ;
                SURFACETYPE:num_bins = 20 ;
        float LANDCOVER(sfctypes, north_south, east_west) ;
                LANDCOVER:standard_name = "MODIS IGBP (Native) land cover" ;
                LANDCOVER:units = "-" ;
                LANDCOVER:scale_factor = 1.f ;
                LANDCOVER:add_offset = 0.f ;
                LANDCOVER:missing_value = -9999.f ;
                LANDCOVER:vmin = 0.f ;
                LANDCOVER:vmax = 0.f ;
                LANDCOVER:num_bins = 20 ;
```

```
float TEXTURE(soiltypes, north_south, east_west) ;
        TEXTURE:standard_name = "STATSGO soil texture" ;
        TEXTURE:units = "-" ;
        TEXTURE:scale_factor = 1.f ;
        TEXTURE:add_offset = 0.f ;
        TEXTURE:missing_value = -9999.f ;
        TEXTURE:vmin = 1.356316e-19f ;
        TEXTURE:vmax = 1.356316e-19f ;
        TEXTURE:num_bins = 16 ;
float SLOPETYPE(north_south, east_west) ;
        SLOPETYPE:standard_name = "NCEP (Native) slope type" ;
        SLOPETYPE:units = "-" ;
        SLOPETYPE:scale_factor = 1.f ;
        SLOPETYPE:add_offset = 0.f ;
        SLOPETYPE:missing_value = -9999.f ;
        SLOPETYPE:vmin = 0.f ;
        SLOPETYPE:vmax = 0.f ;
        SLOPETYPE:num_bins = 1 ;
float GREENNESS(month, north_south, east_west) ;
        GREENNESS:standard_name = "NCEP (Native) monthly greenness climatology" ;
        GREENNESS:units = "-" ;
        GREENNESS:scale_factor = 1.f ;
        GREENNESS:add_offset = 0.f ;
        GREENNESS:missing_value = -9999.f ;
        GREENNESS:vmin = 0.f ;
        GREENNESS:vmax = 0.f ;
        GREENNESS:num_bins = 1 ;
float SHDMIN(north_south, east_west) ;
        SHDMIN:standard_name = "NCEP (Native) greenness min" ;
        SHDMIN:units = "-" ;
        SHDMIN:scale_factor = 1.f ;
        SHDMIN:add_offset = 0.f ;
        SHDMIN:missing_value = -9999.f ;
        SHDMIN:vmin = 0.f ;
        SHDMIN:vmax = 0.f ;
        SHDMIN:num_bins = 1 ;
float SHDMAX(north_south, east_west) ;
        SHDMAX:standard_name = "NCEP (Native) greenness max" ;
        SHDMAX:units = "-" ;
        SHDMAX:scale_factor = 1.f ;
        SHDMAX:add_offset = 0.f ;
        SHDMAX:missing_value = -9999.f ;
        SHDMAX:vmin = 0.f ;
        SHDMAX:vmax = 0.f ;
        SHDMAX:num_bins = 1 ;
float ALBEDO(month, north_south, east_west) ;
        ALBEDO:standard_name = "NCEP (Native) monthly albedo climatology" ;
        ALBEDO:units = "-" ;
        ALBEDO:scale_factor = 1.f ;
        ALBEDO:add_offset = 0.f ;
        ALBEDO:missing_value = -9999.f ;
        ALBEDO:vmin = 0.f ;
        ALBEDO:vmax = 0.f ;
        ALBEDO:num_bins = 1 ;
```

LDT TEST CASES AND EXAMPLES

Highlighting new and unique features of LDT and LIS-7 inputs

Focus of LDT Test Cases

Learning the many capabilities and features LDT …

- To help demonstrate some of LDT's current parameter and input capabilities, several LDT test cases have been developed and provided to the users.
- The test cases prepared for the user showcase main and unique features that either already exist or are new to the LIS-software suite.

LDT Webpage

A new LDT webpage!

- With LDT going "public", a new webpage for LDT now exists:

http://lis.gsfc.nasa.gov/LDT

- These LDT testcases can be found at the LDT website;
- Also, where and how to obtain the "LIS" data and the "Native" parameters is further described.

How the LDT Test Cases are Organized

- Each test case as provided on the LDT webpage has an input script (the "wget" scripts) and output generated files (where available);
- The required input ldt.config, param_attribs.txt and README files are provided in the LDT source code within a directory called: *testcases*
- The README file provided in each test case provides additional instructions on how to run through the test case;
- The input data scripts point to and "wget" the directories or files that are needed to a designated target directory*.

* Note: Each test case *ldt.config* input file is already setup to match the input *wget* scripts internal target directory.

How the LDT Test Cases are Organized

- Parameter Processing:
  - **Native** *vs.* **LIS** data parameter processing examples
  - *Other features*: Mask-parameter consistency options, lapse-rate correction to temperature fields, etc.
- Meteorological Parameter Processing:
  - Forcing terrain height and observed elevation field processing (for lapse-rate adjustment in LIS-7)
- Data Assimilation (DA) Input Processing
- Ensemble restart file generation

Obtaining the Input Data

- **The "LIS" data sets**
  - The **LIS Data Portal**: A web data space that LIS data inputs can be staged for users
  - Requires user to have an account (username, password)
  - Scripts available to access data on the portal
- **"Native" data sets**
  - "wget" scripts provided to users to obtain files from their original sources
  - Sometimes may be provided via the LIS Data Portal, if original "native" data not found on "public" sites.

How to Access Data on Portal via the 'wget' Scripts

- **First:** Do you have a LIS Data Portal account? If not, go to: http://lis.gsfc.nasa.gov/LDT/LDT_dataportal.php

  and a request an account.

- **Second:** Once you have an LDT Data Portal account, you can set up a ***.wgetrc*** file with your LIS Data Portal *username* and *password* doing the following steps:

  1) Go to your home directory (e.g., /home/username)

  2) open a (new) file named: .wgetrc

  3) in that .wgetrc file, enter the following fields:

  --http-user=[LIS DATA PORTAL USERNAME]

  --http-password=[LIS DATA PORTAL PASSWORD]

  and replace [...] entries with your actual username and password. If you have an NCCS account (for like Discover), you will use your NCCS username and password.

  4) Make sure to set permissions for your ***.wgetrc*** file so only **you** can view/ modify it.

Summary

- The Land surface Data Toolkit (LDT) is a new preprocessing toolkit for LIS-7's model parameters and DA inputs.
- LDT offers several features:
  - Multiple parameter processing options;
  - Observation-based DA options (e.g., CDF-matching);
  - Generates ensemble-based restart files
- LDT supports a variety of options, like parameter tiling, and parameter data types, like irrigation maps and lake model data

Future Work

- Add generic capability to bias correct forcing variables (e.g., precipitation).
- The ability to process OPTUE outputs for use in a subsequent LIS run.
- Implement observational correction strategies used (Cressman, OI) into LDT -- for updating snow (and possibly other) data sources.
- Add a layer of machine learning tools (ANN/Bayesian classifier) that will enable the blending of different observational sources (e.g., reprocess LPRM against in-situ data).

Future Work

- Apply HYMAP ("Hydrological Mapping") parameter processing for hydrological modeling applications.
- Implement original LSM parameter preprocessing code (e.g., for CLSM, VIC, etc.).
- Improve computational I/O (e.g., parallel netcdf and other options).
- Replace the spatial interpolation code with ESMF (would be major change also needed by LIS and LVT).