All files described in this document are available in the following Github repository:
https://github.com/ddamicoWX10/DOE_CU_PBL.git
This is a public repository, so it should be clonable for any individual in the group. I can make it a private repository and allow permissions to the group members.

This document is color coded for at least some added convenience as follows:
Hyperlink
[NCL filename]
"$ command line entry"
[python script name]
[directory on Cheyenne or path to your CCPP SCM directory]
[Github directory] -- Note: most of these are in the Github repository above, one is in the CCPP SCM repository.
"Exact_variable_name"
[directory on cumulus]
date placeholder

1. Setting up SCM simulations with any LASSO case.

I have a script available for modifying the LASSO inputs for SCM, but only one date can be used at one time. The script is currently coded to read each date and make a CCPP SCM input file for each of the simulation IDs (SIDs) from the *date* you have provided to the script. This script is called [LASSO_multiple_netcdf.ncl]. The LASSO data was downloaded from https://archive.arm.gov/lassobrowser for this work, but could be simplified in the future on a DOE supercomputing account (from cumulus).

I also have a script that simplifies un-tarring the LASSO tarballs which you have downloaded and zipped, and placed in a directory that you have access to. This script is called [LASSO_tarball.ncl] and will create new directories, move the LASSO data into the correctly dated directory and then un-tar each LASSO file. These un-tarred files will be what you read in [LASSO_multiple_netcdf.ncl]. Lastly, there is a script that will write a namelist (.nml) file for each LASSO SID and a simple python script for running CCPP SCM with "$ ./multi_run_gmtb_scm.py -f ../src/[name_of_generated_python_script.py]". The script that writes the namelist and python files is called [LASSO_namelist.ncl]. Note: Make sure you have moved each namelist to [gmtb-scm/scm/data/processed_case_input/.] and the python file generated to [gmtb-scm/scm/src/.]. Each python file will run each SID for one date.

Scripts for setting up CCPP SCM simulations with LASSO cases:
[LASSO_tarball.ncl]
[LASSO_multiple_netcdf.ncl]
[LASSO_namelist.ncl]
These files are located in the Github directory [SCM-Prep].

2. Scripts for processing SCM, observations, and wrfstat outputs

[LASSO_MLH.ncl]
This is the script that takes the CCPP SCM outputs and makes the MLH plots (using the SCM output variable "atmosphere_boundary_layer_thickness") with LASSO date on the vertical axis and time of day on the horizontal axis. This script is coded to make a plot for one cloud skill score (CSS) and one PBL scheme at a time, so needs to be run multiple times for each PBL scheme and CSS. Color scheme is relatively close to that from the observation plot. Located in the Github directory [MLH-PBLH].

[LASSO_contour_{variable name}.ncl]
This series of plots will make time-height contour plots of a series of variables that are often only available in the CCPP SCM github branch [feature/DOE_PBL_project]. Most of these variables were not used while I worked on the DOE PBL project, but include TKE, $w'q_v'$, $w'\theta'$, w'^2, $w'u'$, $w'v'$, etc. Despite using LASSO in the title, these scripts are for SCM plotting. All are located in the Github directory [SCM-Contour].

[LASSO_input_sfcflux.ncl]
This plot uses the inputs for the SCM to examine the latent and sensible heat fluxes in kg kg^{-1} m s^{-1} and K m s^{-1}, respectively. This ensures that the land surface forcing is the same for all of the SIDs for a given date. Located in the Github directory [SCM-Prep].

[LASSO_MLH_anom.ncl]
This script uses both the "atmosphere_boundary_layer_thickness" and utilizes the 1.5-K theta increase method for calculating the PBLH, and compares the two methods after plotting each method individually. Again, this script only uses one CSS and PBL scheme at one time.

[LASSO_MLH_obsdiff.ncl]
This script compares the observations of MLH from Yufei and Zhien with the "atmosphere_boundary_layer_thickness" output from SCM.

[LASSO_pblcalc_obs.ncl]
This script is one of the latest and uses both the MLH calculations and PBLH observations from Yufei and Zhien and then compares those with the 1.5-theta increase method calculating the PBLH for SCM outputs. This script has three types of plots: 1) compare 15 hours of PBLH obs with the 1.5-K increase method from SCM, 2) compare 15 hours of MLH obs with 1.5-K from SCM, and 3) use MLH obs until 1200 CST and PBLH obs after 1200 CST to compare with 1.5-K SCM.

This script can also be a helpful tool for reading the wrfstat subsets, which will be covered later in this document. So far, this is the only script where I have used any data from the LASSO wrfstat files, where I calculated the 1.5-K theta increase PBL height based on the LASSO output. Each of the last three scripts can be found in the Github directory [MLH-PBLH].

[LASSO_{PBL or CSS}_boxwhisker.ncl]

These scripts make box and whisker plots of the ranges of the variable "atmosphere_boundary_layer_thickness" for each PBL scheme in one CSS [LASSO_CSS_boxwhisker.ncl] and for each CSS in one PBL scheme [LASSO_PBL_boxwhisker.ncl]. Calculates the first and third quartile, along with the median, min, and max. Both are located in the Github directory [SCM-Statistic].

[LASSO_stddev_{PBL or CSS}.ncl]
Calculates and plots the standard deviations in T, q_v, u, v, and PBLH for one CSS and each PBL scheme [LASSO_stddev_PBL.ncl] and the same variables for one PBL scheme and each CSS [LASSO_stddev_CSS.ncl]. Both files are located in the Github directory [SCM-Statistic].

[wrfstat_mkdir.ncl]
This script was written on cumulus, and was necessary because I only had read access to shared drives in the atm118 project directories. I needed to make new directories in my cumulus home directory in order to store the subsetted wrfstat files. The new directories created in my local drive were meant to mimic how Hailey saved the original wrfstat files in [/gpfs/wolf/atm118/proj_shared/NCAR_CU/]. This file is located in the Github directory [WRFSTAT].

[wrfstat_subset.ncl]
This is the more important of the two wrfstat files that I describe. This opens an individual wrfstat_d01_*date*_12:00:00.nc file and reads nearly all of the variables that are either two- or three-dimensional (basically ignoring the four-dimensional "CSV_*" variables) and places them in a new, smaller netCDF file, maintaining all of the metadata from the original wrfstat file. Time is also simplified in this file to "seconds since *date* 12:00:00".

As this script was written on cumulus, the output is, unfortunately, in netCDF3 rather than netCDF4, meaning there is no enhanced compression of the file. Cumulus and netCDF4 did not mesh well after several attempts to write in netCDF4 using various methods (from NCL, I have not quite gotten far enough in Python to write netCDF). Hopefully, this can be remedied and the subsets size can be reduced further. This script is also found in the Github directory [WRFSTAT].