CAlifornia natural and working LANDs carbon and greenhouse gas model (CALAND)

Version 3

Authors

Alan Di Vittorio
Maegen Simmonds
Lawrence Berkeley National Laboratory

Citing CALAND

Please cite the appropriate version using the corresponding Digital Object Identifier (DOI) and our forthcoming peer-reviewed manuscripts, the citations for which will be made available in the most current readme file on CALAND's Github repository as they are published. If the papers are not yet available at the time you wish to cite CALAND, please use the following citation:

Di Vittorio, A., and M. Simmonds (2019) California Natural and Working Lands Carbon and Greenhouse Gas Model (CALAND), Version 3, Technical Documentation.

Repository updates

- July 2020:
  - Updated Copyright and License. See below for details.
- October 2019:
  - Added two new function files to estimate county-level management effects:
  - write_scaled_raw_scenarios.r: scales county-level raw scenarios to the region level for input to write_caland_inputs()
  - write_scaled_outputs.r: scales the CALAND outputs back to the county level
  - These functions are used together to effectively apply CALAND at the county level in place of the region levels, so that county-level effects of county-level management can be estimated
  - Caveats of county-level scaling

    - Increase in estimation uncertainty that is difficult to quantify
    - The scaled outputs approximate what CALAND would return for a single county if the region boundaries were counties instead of the current nine regions
    - County-level management has limitations compared to regular CALAND operation:
        - No land use/cover change (except what may be prescribed in a restoration scenario)
        - No land protection management (because there is no urban growth)
        - Always full Forest regeneration
        - Annual practices must be in a separate scenario from restoration practices
        - No fire with restoration practices (fire can be used with annual practices)
        - Land categories with zero county area cannot be restored at the county-levels (except for Delta Fresh_marsh)
        - Some county restoration targets may not be possible to limitations in scaled source area
    - Alternatively, one could simply simulate county-level targets without scaling and obtain the region-level effects of county-level management

Version History

- CALAND version 3.0.0: 25 June 2019; DOI: 10.5281/zenodo.3256727
- CALAND version 2.0.0: 23 October 2017; not an official release
- CALAND version 1.0.0: 25 January 2017; not an official release

License and Copyright

CALAND is licensed as open source software under a modified BSD license.

Funding

The California Natural Resources Agency

Table of Contents

- Instructions for your first test runs with CALAND

Overview

CALAND is a system of algorithms (.r files) and data developed to quantify the impacts of various suites of California State-supported land use and land management strategies on landscape carbon and greenhouse gas emissions (CO_2, CH_4, and optional black carbon) relative to a baseline scenario for the Draft California 2030 Natural and Working Lands Climate Change Implementation Plan (2019). CALAND consists of the model itself (CALAND.r), a data pre-processing algorithm (write_caland_inputs.r) for generating model input files, and three post-processing algorithms (plot_caland.r, plot_scen_types.r, plot_uncertainty.r) for diagnosing, visualizing, and summarizing model ouputs.

Installation

Get the CALAND files and tools needed to use them by following the instructions below for downloading R, RStudio, and CALAND. CALAND can be run using R, RStudio, or command line. Note that using RStudio will require downloading both R and RStudio. Here are instructions for downloading R, RStudio, and CALAND.

Downloading R

Mac Users

1. Go to www.r-project.org.
2. Click the "download R" link in the middle of the page under "Getting Started."
3. Select a CRAN location (a mirror site) nearest you and click the corresponding link.
4. Click on the "Download R for (Mac) OS X" link at the top of the page.
5. Click on the file containing the latest version of R under "Latest Release".
6. Save the .pkg file, double-click it to open, and follow the installation instructions.
7. Now that R is installed, you can download and install RStudio.

Windows Users

1. Go to www.r-project.org.
2. Click the "download R" link in the middle of the page under "Getting Started."
3. Select a CRAN location (a mirror site) nearest you and click the corresponding link.
4. Click on the "Download R for Windows" link at the top of the page

5. Click on the "install R for the first time" link at the top of the page.
6. Click "Download R for Windows" and save the executable file (.exe) somewhere on your computer. Run the .exe file and follow the installation instructions.
7. Now that R is installed, you can download and install RStudio.

Downloading RStudio

1. Go to www.rstudio.com and click on "Download RStudio".
2. Click on "Download" under RStudio Desktop.
3. Click on the version recommended for your system under Installers, and save the .dmg file (Mac User) or .exe (Windows User) on your computer, double-click it to open, and then drag and drop it to your applications folder.
4. Now RStudio is installed.

Three options for downloading CALAND:

(1) Downloading CALAND releases from the Zenodo digital archive

1. Go to Zenodo.org.
2. Search for CALAND.
3. Find the CAlifornia natural and working LANDs carbon and greenhouse gas model (CALAND) in the search results and click on it (e.g., selecting CALAND v3.0.0 should direct to http://zenodo.org/record/3256727).
4. Download the .zip file for the desired version of CALAND (e.g., caland-v3.0.0.zip). Now CALAND is on your local computer.
5. Unzip the file and and save the resulting caland folder where you want the directory to be located.

(2) Downloading CALAND releases from California Natural Resources Agency (CNRA) Open Data Portal

1. Go to the CNRA Natural and Working Lands Open Data Portal.
2. Search for CALAND.
3. Find the desired CALAND version in the search results and click on it. For example, CALAND version 3 should go to https://data.cnra.ca.gov/dataset/caland-version-3.
4. Download the .zip file for the desired version of CALAND (e.g., CALAND_v3.0.0.zip). Now CALAND is on your local computer.
5. Unzip the file and and save the resulting caland folder where you want it to be located.

(3) Downloading the full CALAND repository from github

1. Sign up for a free github account.

2. Go to the CALAND github repository.
3. Under repository name, click Clone or download.
4. In the Clone with HTTPs section, click the clipboard icon to copy the clone URL for the repository.
5. Open Terminal (MAC and Linux), or Git Bash (Windows).
6. Change the current working directory to the location where you want the cloned directory to be located.
7. Type `git clone` and then paste the URL you copied in Step 4.
8. Press Enter. Your local clone will be created.

Ancillary Data not Included with CALAND

Downloading the initial 2010 land category raster data:

1. Go to the CNRA Natural and Working Lands Open Data Portal.
2. Search for CALAND.
3. Find the desired CALAND version in the search results and click on it. For example, CALAND version 3 should go to https://data.cnra.ca.gov/dataset/caland-version-3.
4. Download the .zip file for the desired land category raster data (e.g., calandv3_2010_land_category_raster_data.zip). Now it is on your local computer.
5. Unzip the file and and save the raster data folder where you want it to be located.

Ancillary Data Included with CALAND

The CALAND release includes some ancillary data that can be used for visualizing outputs. There are four files included in caland-3.0.0/ancillary/:

- A .csv file containing a table defining the initial 2010 land categories and their areas in square meters
- A .zip file containing a shapefile of the CALAND ownership boundaries
- A .zip file containing a shapefile of the CALAND region boundaries
- A .zip file containing a shapefile of the CALAND region-ownership boundaries

Usage

The five R files (.r) located in the caland-3.0.0/ directory include (1) write_caland_inputs.r, (2) CALAND.r, (3) plot_caland.r, (4) plot_scen_types.r, and (5) plot_uncertainty.r, the first three of which are designed to be run in sequential order. However, for your first test runs you can skip write_caland_inputs.r, as there are example input files in caland-3.0.0/inputs directory that were created for the Draft California 2030 Natural and Working Lands Climate Change

Implementation Plan. Each scenario that you want to model must be run individually using the functions in CALAND.r before proceeding to using plot_caland.r to compare individual scenarios and compute *changes* in carbon dynamics (increase or decrease) due to the effects of one scenario compared to another. Using plot_caland.r is essential for obtaining valid results; the outputs from CALAND.r are carbon emissions for individual scenarios, which are not reliable estimates of the carbon budget due to high uncertainty in the input data. Thus, CALAND is not intended to provide insight into whether the landscape is a net source or sink of carbon under a given scenario, but it is intended to quantify the change in carbon dynamics of a given scenario compared to a reference baseline scenario. The plot_caland.r file will produce graphics (.pdf) and data tables (.csv) for both individual scenario values and differences between scenarios. The individual scenario outputs are for diagnosing issues and understanding the model; while the differences between scenarios are the valid model results and the main purpose of CALAND. After using plot_caland.r, you can proceed to creating more detailed graphs using plot_scen_types.r or plot_uncertainty.r.

Each R file contains scripts (commands) that comprise the essential functions of CALAND. To run any of them you must make sure that the working directory is caland-3.0.0/ by typing the following:

```
setwd("<your_path>/caland-3.0.0/")
```

Here you will learn about the main functions in each of the R files and the settings (arguments) you must choose when running each of them. Note that some of the arguments are vectors, or a collection of two or more elements. To assign multiple elements to a vector we use `c()` which concatenates (i.e., links) elements into a vector. For example, to assign "Coastal_marsh", "Fresh_marsh", and "Cultivated" to the landtype argument `lt`, you would type the following:

```
lt = c("Coastal_marsh", "Fresh_marsh", "Cultivated")
```

R files and their functions

(1) write_caland_inputs.r

Overview of `write_caland_inputs()`

The `write_caland_inputs()` function is defined in the write_caland_inputs.r file. Its purpose is to read the raw data files found in the caland-3.0.0/raw_data directory, and organize them into the detailed input files needed to run the `CALAND()` model (i.e., one carbon input file (.xls) and an individual scenario input file (.xls) for each defined scenario). Each of the input files that `write_caland_inputs()` creates are Excel workbooks, which are comprised of individual worksheets, one for each data table. The raw data files define the management scenarios;

initial 2010 areas and carbon densities; annual area changes; ecosystem carbon fluxes; management parameters; parameters for land conversion to cultivated or developed lands; wildfire parameters; wildfire areas; mortality fractions; and climate change scalars. There is a suite of settings (arguments) with various options that you choose when running `write_caland_inputs()`, such as climate (historical, RCP 4.5, or RCP 8.5) and whether you are controlling for wildire and land use land cover change (LULCC) for sensitivity testing of individual practices.

The `CALAND()` model operates on 940 land categories, but some of the raw data are not land category-specific. Thus, one of the main purposes of `write_caland_inputs()` is to disaggregate all non-land-category-specific raw data into individual land categories, and to save these expanded data tables in the carbon and scenario `CALAND()` input files. On the other hand, some of the raw data are land category-specific, including the initial 2010 areas and carbon densities, forest vegetation carbon fluxes, forest management carbon parameters, and some management areas.

CAUTION: Creating new raw data files and using this function to create new input files for `CALAND()` is a complicated task. All of the raw data have to be carefully assembled to ensure it is processed as intended. Therefore, creating new input files is not recommended unless you have a good understanding of the required data and how the model works.

Raw data input files to `write_caland_inputs()`

The inputs to `write_caland_inputs()` are referred to as raw data, as they are generally not region- and ownership-specific and must be disaggregated to all the 940 individual land categories simulated by `CALAND()`. The raw data include both .xls files and .csv files, which are in caland-3.0.0/raw_data/. The raw data used to generate the CALAND results reported in the Draft California 2030 Natural and Working Lands Implementation Plan (2019) are in parentheses next to each input file below.

- Carbon parameter raw data file (e.g., lc_params.xls)
    - The carbon parameter raw data file is an Excel workbook comprised of 8 individual worksheets with the following names: vegc_uptake, soilc_accum, conversion2ag_urban, forest_manage, dev_manage, grass_manage, ag_manage, and wildfire. They represent vegetation carbon fluxes (historic climate); soil carbon fluxes (historic climate); land conversion C parameters; management C parameters for forest, developed lands, rangelands, and cultivated lands; and carbon parameters for low, med, and high wildfire severity, respectively.
    - Column headers are on row 12
    - Only non-zero values are included (missing values are zero)
    - More rows can be added that define new management practices as appropriate

- Scenarios raw data file (e.g., nwl_scenarios_v6_ac.xls)

    - The scenarios raw data file is an Excel file comprised of at least one scenario of management practices and corresponding areas or fractions (one scenario per worksheet). Note the practice determines whether an area or fraction is used; it is not arbitrary.
    - 10 columns each: Region, Land_Type, Ownership, Management, start_year, end_year, start_area, end_area, start_frac, end_frac.
    - Either area (start_area and end_area) or fractional (start_frac, end_frac) values are used depending on the practice; the other two columns are NA. If area is required, the units can be in ha or ac, which must be specified by the `units_scenario` argument.
    - Region and Ownership can be "All" or a specific name
    - Each record defines management for a specific time period; outside of the defined time periods the management values are zero
    - Each scenario worksheet will result in the creation of a single input scenario file (.xls), which will be comprised of additional worksheets of data tables derived from the other raw data files described below.
    - The worksheet name for each scenario in the scenarios raw data file will determine the scenario input file name that is created.
    - Notes on restoration practices of the following land types:
        - Fresh marsh: comes out of only private and state land in the Delta, so make sure that these ownerships are designated.
        - Coastal marsh comes out of only private and state land in the coastal regions, so make sure these are available, from cultivated
        - Meadow restoration is only in the sierra cascades, and in private, state, and usfs nonwilderness, from shrub, grass, savanna, and woodland
        - Woodland restoration comes from cultivated and grassland
        - Afforestation is forest area expansion and comes from shrubland and grassland
        - Reforestation is forest area expansion that comes from shrubland only to match non-regeneration of forest

- Climate raw data file (e.g., climate_c_scalars_iesm_rcp85.csv)

    - The climate raw data file is a .csv file of annual climate scalars (fraction) for vegetation and soil carbon fluxes.
    - The .csv file consits of four ID columns: Region, Land_Type, Ownership, and Component (Soil or Vegetation); and one column for each year, starting in year 2010.
    - These fractions will directly scale the vegetation and soil C flux values under a historic climate in `CALAND()`
    - There must be a valid climate file, even if historic climate is designated (`CLIMATE = HIST`), in which case all climate scalars will be changed to 1.

- Wildfire raw data file (e.g., fire_area_canESM2_85_bau_2001_2100.csv)
    - The wildfire raw data file is a .csv file of annual wildfire areas (units = ha).
    - The .csv file consists of two ID columns (Region and Ownership), and a fire area column for each year, starting in year 2001. All possible Region-Ownership combinations are included (81 total), even if they don't exist.
    - The wildfire areas from 2001 to 2015 are averaged by `write_caland_inputs()` to compute the initial 2010 wildfire areas written to the scenario input file. If historic climate is designated, (`CLIMATE = HIST`), the average 2001 to 2015 burn areas in the RCP 8.5 wildfire raw data are used for initial 2010 burn areas and throughout the entire simulation (same value, no trend).
    - Wildfire severity fractions of the annual wildfire areas are hard-wired in `write_caland_inputs()`; and are not dependent on the selected climate. The high, medium, and low severity fractions are uniform across the State of California, starting with high = 0.26, medium = 0.29, and low = 0.45. The high severity fraction increases annually at a historical rate (i.e., 0.0027) and the low and medium fractions increase proportionally in order to account for the total wildfire area each year.
    - Non-regeneration areas are parameterized as a function of the high severity fraction and a threshold distance from burn edge; however this is handled in `CALAND()` with the `NR_Dist` argument.

- Mortality raw data file (e.g., mortality_annual_july_2018.csv)
    - The mortality raw data file is a .csv file of mortality fractions (units = fraction), representing the annual mortality rate as a fraction of live biomass carbon (above-ground main canopy and roots).
    - The .csv file consits of three ID columns (Region, Land_Type, and Ownership) and one data column (Mortality_frac).
        - Land type must be specific but region and/or ownership can be "All".
        - Valid land types include any woody land type that have C accumulation in vegetation (i.e., Forest, Shrubland, Savanna, Woodland, and Developed). However, if rows do not exist for all valid land types, write_caland_inputs() will assign it a default annual mortality fraction of 0.01.
    - Unique treatment of Forest mortality: A doubled forest mortality trend is computed by `write_caland_inputs()` for 10 years (2015-2024) to emulate the recent and expected forest mortality due to insects and drought. This is the default setting, but can be changed in the arguments.
    - Unique treatment of Developed mortality: Mortality in Developed lands is processed differently from others in `CALAND()` because the urban system is highly managed, and allows for more control of what happens to the dead biomass; the mortality is transferred to the above-ground harvest for dead removal management.

- New area GIS raw data file (e.g., CALAND_Area_Changes_2010_to_2101.csv)

    - The new area GIS raw data file is a .csv file of annual area changes (sq m) for each land category, starting in year 2010.
    - The .csv file consists of one ID column (Landcat) and one annual area change column for each year.
    - With the exception of Seagrass, Fresh marsh is the only land type that does not initially exist in this GIS raw data files. The Fresh marsh land categories are added by `write_caland_inputs()` with initial areas of 0, which only increases by restoration.

- Original area GIS raw data files

    - CALAND v1 and v2 used two remote-sensing based area GIS files (i.e., area_lab_sp9_own9_2001lt15_sqm_stats.csv and area_lab_sp9_own9_2010lt15_sqm_stats.csv), but these files were not used in the Draft California 2030 Natural and Working Lands Climate Change Implementation Plan.
    - The original area GIS files include two .csv files of areas (sq m) for each land category in 2001 and 2010, respectively.
    - The .csv files consist of six ID columns (no headers, but rows 1 through 6 are Region integer code, Region, Land_type integer code, Land_type, Ownership integer code, and Ownership, respectively), and one area change column.
    - The land category (Land_cat) integer codes will be calculated and then matched with the integer codes in the carbon GIS raw data files.
    - The year has to be in the name of the file; currently `write_caland_inputs()` assumes that the years are 2001 and 2010.

- Carbon GIS raw data files (e.g., gss_soc_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_agc_se_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_agc_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_bgc_se_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_bgc_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_ddc_se_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_ddc_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_dsc_se_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_dsc_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_ltc_se_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_ltc_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_usc_se_tpha_sp9_own9_2010lt15_stats.csv,
  lfc_usc_tpha_sp9_own9_2010lt15_stats.csv)

    - The carbon GIS raw data files are .csv files (13 total) of univariate statistics of the initial C densities (units = Mg C per ha) of the seven carbon pools (i.e., soil, above-ground biomass, below-ground biomass (roots), down dead biomass, standing dead biomass, litter biomass, and understory biomass). These files were created by

processing GIS raster data with a 30 m2 resolution. Thus, the statisitics are based on carbon density data associated with 30m2 cells within each 2010 land category boundary.
    - Each .csv file consists of 14 columns in the following order: zone (i.e., land category code), label (no data), valid cell count (i.e., count of non-missing cells), null cell count (count of missing cells), min, max, range, mean, mean of absolute values (mean_of_abs), stddev (standard deviation), variance, coefficient of variation (coeff_var), sum, absolute sum (sum_abs).
    - Only the min, max, mean, and standard deviation are used.
    - Vegetation C densities are from the California Air Resources Board inventory, one value per carbon pool for each land category.
    - Soil organic C densities are from gSSURGO; any zero values in the original data set were assumed not to be valid, and were therefore omitted in calculating the land category averages.
    - In the soil C density raster file (i.e., the file from which the soil C raw data file is summarizing) there were 17601 ha (mostly rivers) with zeros, and 12236544 ha (mostly desert) with missing data. The missing values will be converted to zero by `write_caland_inputs()`. However, all of these zero values are excluded from calculating the averages.

- Controller for Wildfire and LULCC raw data file

    - The wildfire and LULCC raw data file is a .csv file designed for sensitivity testing of individual practices or processes in `CALAND()`; it turns on/off wildfire and/or LULCC for each scenario with the purpose of isolating an effect.
    - The .csv file consists of three columns, in order: Scenario, Wildfire, LULCC
        - Scenario column: matches the worksheet names in the scenarios raw data file; these scenario names MUST MATCH between these two files
        - Wildfire column: 1 = wildfire is on; 0 = wildfire is off
        - LULCC column: 1 = LULCC is on; 0 = LULCC is off

Arguments in `write_caland_inputs()`

1. `c_file`: Assign a name to the carbon input file that `write_caland_inputs()` will create for `CALAND()` (needs to be .xls); default is `c_file = "carbon_input_nwl.xls"`.
2. `inputs_dir`: The directory within "./inputs" to put the generated input files; default is `c_file = ""` so they go into "./inputs".
3. `parameter_file`: Carbon parameter raw data file (needs to be .xls file); default is `parameter_file = "lc_params.xls"`.
4. `scenarios_file`: Scenarios raw data file with region- and/or ownership-generic management scenarios that will be expanded upon in the scenario files created for `CALAND()` (needs to be .xls file with one scenario per worksheet); default is `scenarios_file`

`= "nwl_scenarios_v6_ac.xls"`.

5. `units_scenario`: Specify units for input areas in scenarios_file (must be `"ac"` or `"ha"`); default is `units_scenario = "ac"`.
6. `climate_c_file`: Climate raw data file containing either RCP 4.5 or RCP 8.5 climate scalars for vegetation and soil carbon fluxes (needs to be .csv); the RCP must match the RCP associated with the wildfire raw data file. If historic climate is desired (`CLIMATE = "HIST"`), assign either RCP 4.5 or RCP 8.5 climate raw data file, as there needs to be a valid file from which `write_caland_inputs()` can convert all values to 1 (i.e., no future climate effect). Default is `climate_c_file = "climate_c_scalars_iesm_rcp85.csv"`.
7. `fire_area_file`: The wildfire raw data file containing annual burned area by region-ownership (needs to be .csv); the RCP must match the climate raw data file; needs to be RCP 8.5 for historic climate (`CLIMATE = "HIST"`). Default is `fire_area_file = "fire_area_canESM2_85_bau_2001_2100.csv"`.
8. `mortality_file`: Mortality raw data file containing mortality rates as annual fraction of above-ground biomass; includes values for woody land types that have C accumulation in vegetation; any missing valid land types will be assigned a default annual mortality fraction of 0.01. Default is `mortality_file = "mortality_annual_july_2018.csv"`.
9. `area_gis_files_new`: New area GIS raw data file of GIS statistics of annual area change by land category (sq m) (needs to be a .csv); default is `area_gis_files_new = "CALAND_Area_Changes_2010_to_2101.csv"`.
10. `area_gis_files_orig`: Original area GIS raw data files of GIS statistics of areas by land category (sq m) that are compared to compute annual area changes (concatenate two .csv file names, one for each year); default is `area_gis_files_orig = c("area_lab_sp9_own9_2001lt15_sqm_stats.csv", "area_lab_sp9_own9_2010lt15_sqm_stats.csv")`.
11. `land_change_method`: Assigning `"Landcover"` will use the original method of remote sensing landcover change from 2001 to 2010 (i.e., files assigned to `area_gis_files_orig`), or assigning `"Landuse_Avg_Annual"` will use the average LUCAS-modeled annual area changes from 2010 to 2050 based on projected change in cultivated and developed areas (i.e., file assigned to `area_gis_files_new`); default is `land_change_method = "Landuse_Avg_Annual"`.
12. `carbon_gis_files`: Carbon GIS raw data files of GIS statistics of carbon density by carbon pool and land category (Mg C per ha) (concatentate 13 .csv file names); default is `carbon_gis_files = c("gss_soc_tpha_sp9_own9_2010lt15_stats.csv", "lfc_agc_se_tpha_sp9_own9_2010lt15_stats.csv", "lfc_agc_tpha_sp9_own9_2010lt15_stats.csv", "lfc_bgc_se_tpha_sp9_own9_2010lt15_stats.csv", "lfc_bgc_tpha_sp9_own9_2010lt15_stats.csv", "lfc_ddc_se_tpha_sp9_own9_2010lt15_stats.csv", "lfc_ddc_tpha_sp9_own9_2010lt15_stats.csv", "lfc_dsc_se_tpha_sp9_own9_2010lt15_stats.csv", "lfc_dsc_tpha_sp9_own9_2010lt15_stats.csv",`

`"lfc_ltc_se_tpha_sp9_own9_2010lt15_stats.csv"`, `"lfc_ltc_tpha_sp9_own9_2010lt15_stats.csv"`, `"lfc_usc_se_tpha_sp9_own9_2010lt15_stats.csv"`, `"lfc_usc_tpha_sp9_own9_2010lt15_stats.csv")`.

13. `scen_tag`: (optional) A scenario file name tag that will be appended to the scenario names in the worksheets in the scenarios raw data file (`scenarios_file`); default is `scen_tag = "default"`.
14. `start_year`: The initial year of the simulation, which must match the year of the carbon GIS raw data files. Additionally, the new area file should include this year, and one of the original area GIS files needs to be for this year. Default is `start_year = 2010`.
15. `end_year`: this is the final year output desired from the `CALAND()` simulation (matches the `CALAND()` end_year argument); default is `end_year = 2101`.
16. `CLIMATE`: projected climate (RCP 4.5 or RCP 8.5) or historical climate (`"PROJ"` or `"HIST"`, respectively); affects wildfire areas and vegetation and soil carbon flux values; the projected climate scenario is determined by the fire and climate input files; `"HIST"` uses RCP 8.5 wildfire average areas from 2001 to 2015 for historical average burn area, and sets all climate scalars to 1. Deafult is `CLIMATE = "HIST"`.
17. `forest_mort_fact`: Assign a number that will be used to adjust the forest mortality during the period specified by `forest_mort_adj_first` and `forest_mort_adj_last`; default is `forest_mort_fact = 2`.
18. `forest_mort_adj_first`: the first year to adjust forest mortality; default is `forest_mort_adj_first = 2015`.
19. `forest_mort_adj_last`: the last year to adjust forest mortality; default is `forest_mort_adj_last = 2024`.
20. `control_wildfire_lulcc_file`: .csv file with flags (1 or 0) for each scenario in `scenarios_file` to control wildfire and LULCC. This is for sensitivity testing of individual practices or processes in `CALAND()`. Default is `control_wildfire_lulcc_file = "individual_proposed_sims_control_lulcc_wildfire_aug2018.csv"`. Note this will not be used unless `control_wildfire_lulcc = TRUE`.
21. `control_wildfire_lulcc`: if TRUE, read `control_wildfire_lulcc_file` to control wildfire and LULCC; default is `control_wildfire_lulcc = FALSE`.

Outputs from `write_caland_inputs()`

Output files are written to caland-3.0.0/inputs/ (unless a sub-directory is specified differently from the default `inputs_dir = ""`). There is one carbon output file and one scenario output file for each scenario in the `scenarios_file`:

- Carbon .xls file: `c_file`
  - Carbon densities are in Mg C per ha (t C per ha)
  - Factors (or scalars) are fractions
- Scenario .xls file(s): `<scenario_name>_scen_tag_<climate>.xls`

- One for each scenario in the `scenarios_file`
- `<scenario_name>` will be named based on the worksheet names in the `scenarios_file`
- `<climate>` will be named based on the climate associated with the `climate_c_file` unless historical climate is chosen (`CLIMATE = "HIST"`), in which case "HIST" will be used.
- areas are in ha

Example of `write_caland_inputs()` arguments used to generate the `CALAND()` input files for the Draft California 2030 Natural and Working Lands Climate Change Implementation Plan:

```
write_caland_inputs(c_file = "carbon_input_nwl.xls", inputs_dir = "", parameter_file = "lc_params.xls", scenarios_file = "nwl_scenarios_v6_ac.xls", units_scenario = "ac", climate_c_file = "climate_c_scalars_iesm_rcp85.csv", fire_area_file = "fire_area_canESM2_85_bau_2001_2100.csv", mortality_file = "mortality_annual_july_2018.csv", area_gis_files_new = "CALAND_Area_Changes_2010_to_2051.csv", area_gis_files_orig = c("area_lab_sp9_own9_2001lt15_sqm_stats.csv", "area_lab_sp9_own9_2010lt15_sqm_stats.csv"), land_change_method = "Landuse_Avg_Annual", carbon_gis_files = c("gss_soc_tpha_sp9_own9_2010lt15_stats.csv", "lfc_agc_se_tpha_sp9_own9_2010lt15_stats.csv", "lfc_agc_tpha_sp9_own9_2010lt15_stats.csv", "lfc_bgc_se_tpha_sp9_own9_2010lt15_stats.csv", "lfc_bgc_tpha_sp9_own9_2010lt15_stats.csv", "lfc_ddc_se_tpha_sp9_own9_2010lt15_stats.csv", "lfc_ddc_tpha_sp9_own9_2010lt15_stats.csv", "lfc_dsc_se_tpha_sp9_own9_2010lt15_stats.csv", "lfc_dsc_tpha_sp9_own9_2010lt15_stats.csv", "lfc_ltc_se_tpha_sp9_own9_2010lt15_stats.csv", "lfc_ltc_tpha_sp9_own9_2010lt15_stats.csv", "lfc_usc_se_tpha_sp9_own9_2010lt15_stats.csv", "lfc_usc_tpha_sp9_own9_2010lt15_stats.csv"), scen_tag = "default", start_year = 2010, end_year = 2101, CLIMATE = "PROJ", forest_mort_fact = 2, forest_mort_adj_first = 2015, forest_mort_adj_last = 2024, control_wildfire_lulcc_file = "individual_proposed_sims_control_lulcc_wildfire_aug2018.csv", control_wildfire_lulcc = FALSE)
```

(2) CALAND.r

Overview of `CALAND()`

The `CALAND()` function is the carbon and greenhouse gas accounting model, which is defined in the CALAND.r file. It uses the input files generated by `write_caland_inputs()`. A single scenario file is simulated each time `CALAND()` is run, producing a single main output .xls file that summarizes outputs for 214 variables including annual and cumulative metrics, each in an individual worksheet. There is a suite of settings (arguments) with various options that you choose when running `CALAND()`, such as which carbon values to use from the carbon inputs file (i.e., mean, mean+sd, mean-sd, min, or max) and which level of forest non-regeneration to assume following high severity wildfire.

Model structure & order of operations:

- This model follows basic density (stock) and flow guidelines similar to IPCC Tier 3 protocols by integrating observed historical carbon flows (fluxes) and initial carbon densities with wildfire, climate effects, and land cover change derived from external models.
- Resolution is at the land category level (region-landtype-ownership combination), which only has a spatial boundary for the intial simulation year. Beyond that, processes within each land category are implemented on non-spatially explicit *areas* within static region-ownership boundaries. In other words, there are no grid-cell level computations. This highlights how `CALAND()` is not a stand-level model.
- Carbon calculations occur in `start_year` up to `end_year - 1`.
- `end_year` denotes the final area after the changes in `end_year - 1`.
- Initial carbon density input values are the univariate statistics of the total pixel population within each land category (i.e., land type-region-ownership combination).
- Carbon accumulation (flux) input values are univariate statistics of literature values for a given land type (coarse resolution) or land category (fine resolution), depending on the availability of data.
- The initial land category area data are used for carbon operations in the first year.
- Annual ecosystem carbon fluxes are calculated for each land category using inputs for historic carbon fluxes and mortality rates, scaled up to the annual input area data with adjustments made for (in order of operation) climate, management, and wildfire.
- Land conversion carbon fluxes for each land category are calculated after ecosystem carbon fluxes within the same year.
- Each subsequent step uses the updated carbon values from the previous step.
- The new carbon density and area for each land category are assigned to the beginning of the next year.
- All wood products are lumped together and labeled as "wood".
- Wood product emissions are based on landfill decay of discarded products.
- Bioenergy emissions are based on combustion of biomass feedstock by current California bioenergy plants.

Input files to `CALAND()`:

The input .xls files for `CALAND()` are in caland-3.0.0/inputs/ (unless a sub-directory `indir` is specified differently from the default of no subdirectory (`indir = ""`) (see Arguments section below). The following Excel input files have matching number of header rows (rows preceding the first row of data):

- Carbon input file: The initial carbon state (carbon densities) of the seven carbon pools, and all the carbon flow parameters (fluxes and scalars) are in a .xls file in the inputs/ directory. This file is created by `write_caland_inputs`, and it does not change unless the

raw land carbon input file lc_params.xls is modified.

    - **carbon_input_nwl.xls** Default filename, containing the initial carbon densities (mean, min, max, SD) in seven carbon pools and the historical soil and vegetation carbon fluxes (mean, min, max, SD) for each land category, the carbon adjustment parameters for wildfire, conversion of any land type to Cultivated or Developed lands, and Forest, Developed, and Rangeland (Grassland, Savanna, Woodland) management, and the soil carbon fluxes (mean, min, max, SD) in Cultivated lands under the 'soil conservation' practice.

- Scenario input file: The scenario that will be simulated.

    - `<scenario_name>.xls` Contains the initial areas of each land type per region-ownership combination (i.e., land category), annual net area changes per land category; annual wildfire area per region-ownership combination; annual mortality per land category; annual managed area per land category; and climate change scalars for vegetation and soil carbon fluxes per land category.

    - Example scenario input files: There are five scenario input xls files in the caland-3.0.0/inputs/ directory that were created using `write_caland_inputs()` for the Draft California 2030 Natural and Working Lands Climate Change Implementation Plan (2019). These scenarios incorporate RCP8.5 climate effects on carbon exchange and wildfire area. 'Default' in the filename means that the scenarios include doubled forest mortality from 2015 to 2024 to emulate recent and ongoing die-off due to insects and drought.

        - Historical Baseline Scenario: The NWL_Historical_v6_default_RCP85.xls file is the reference scenario used to compare the changes in management that are prescribed in the following alternative scenarios. It does not include any California State-funded management or increases in the forest fraction of urban lands. Note that this scenario was intended to run in `CALAND()` with the setting for maximum non-regeneration (i.e., forest conversion to shrubland) in forest areas burned by high-severity wildfire.
        - Alternative A Scenario: The NWL_Alt_A_v6_default_RCP85.xls file adds desired, *low levels* of California State-funded management to the Historical Baseline Scenario. Note that agricultural management is limited in scope, and only represents California Natural Resources Agency-funded areas for soil conservation in cultivated lands. Note that this scenario was intended to run in `CALAND()` with the setting for full regeneration post-wildfire to represent maximum reforestation of forest areas that would not otherwise recover fully following high-severity wildfire.
        - Alternative B Scenario: The NWL_Alt_B_v6_default_RCP85.xls file adds desired,

high levels of California State-funded management to the Historical Baseline Scenario. Note that agricultural management is limited in scope, and only represents California Natural Resources Agency-funded areas for soil conservation in cultivated lands. Note that this scenario was intended to run in `CALAND()` with the setting for full regeneration post-wildfire to represent maximum reforestation of forest areas that would not otherwise recover fully following high-severity wildfire.

- Alternative A Scenario without Avoided Conversion: The NWL_Alt_A_v6_NoAC_default_RCP85.xls file mimics the *low levels* of state-funded management in Alternative A, but *without avoided conversion*.
- Alternative B Scenario without Avoided Conversion: The NWL_Alt_B_v6_NoAC_default_RCP85.xls file mimics the *high levels* of state-funded management in Alternative B, but *without avoided conversion*.

Arguments in `CALAND()`:

`CALAND()` has 15 arguments that control which input files and values are used, and how the model will operate for each run. A single scenario is simulated at a time. At a minimum, you must specify the scenario filename each time you run `CALAND()` and the other arguments will automatically be assigned default values explained here:

1. `scen_file_arg`: Assigns the scenario .xls file; assumed to be in caland-3.0.0/inptus/`indir`. This is the only argument that does not have a default value. Thus, it is required that you assign it. All other arguments will be assigned default values unless you change them.
2. `c_file_arg`: Assigns the carbon parameter input file; assumed to be in caland-3.0.0/inputs/`outdir`. If nothing is specified, default is: `c_file_arg = "carbon_input_nwl.xls"`.
3. `indir`: Assigns the directory in caland-3.0.0/inputs/ that contains `scen_file` and `c_file`; do not include "/" character at the end. If nothing is specified, the default is: `indir = ""` meaning that it is located in caland-3.0.0/inputs/.
4. `outdir`: Assigns the directory in caland-3.0.0/outputs/ to save `CALAND()` output files; do not include "/" character at the end. If nothing is specified, default is blank: `outdir = ""` meaning that output files will be saved in caland-3.0.0/outputs/.
5. `start_year`: Simulation begins at the beginning of this year. If nothing is specified, default is: `start_year = 2010`.
6. `end_year`: Simulation ends at the beginning of this year (so the simulation goes through the end of end_year - 1). If nothing is specified, default is: `end_year = 2101`.
7. `value_col_dens`: Selects which carbon density values to use; 5 = min, 6 = max, 7 = mean, 8 = std dev. If nothing is specified, default is the mean: `value_col_dens = 7`.
8. `value_col_accum`: Integer code identifying which soil and vegetation carbon flux values to use (i.e., 5 = min, 6 = max, 7 = mean, 8 = std dev). If nothing is specified, default is the mean: `value_col_accum = 7`.

9. `value_col_soilcon`: Integer code identifying which soil carbon flux values to use for the cultivated soil conservation practice (i.e., 6 = min, 7 = max, 8 = mean, 9 = std dev). If nothing is specified, default is the mean: `value_col_soilcon = 8`.
10. `ADD_dens`: For use with `value_col_dens = 8`; assign `ADD_dens = TRUE` to add the std dev to the mean carbon density values, or assign `ADD_dens = FALSE` to subtract the std dev from the mean carbon density values. If nothing is specified, default is addition: `ADD_dens = TRUE`.
11. `ADD_accum`: For use with `value_col_accum = 8`; assign `ADD_accum = TRUE` to add the std dev to the mean carbon flux values, or assign `ADD_accum = FALSE` to subtract the std dev from the mean carbon flux values. If nothing is specified, default is addition: `ADD_accum = TRUE`.
12. `ADD_soilcon`: For use with `value_col_soilcon = 9`; assign `ADD_soilcon = TRUE` to add the std dev to the mean carbon flux value of the cultivated soil conservation practice, or assign `ADD_soilcon = FALSE` to subtract the std dev from the mean carbon flux of the cultivated soil conservation practice. If nothing is specified, default is addition: `ADD_soilcon = TRUE`.
13. `NR_Dist`: For adjusting the amount of non-regenerating forest after high severity wildfire, use -1 for full regeneration (i.e., no non-rengeration) or 120 for maximum non-regeneration, which is the threshold distance (m) to the edge of a burn patch, beyond which the forest will not regenerate and will convert to shrubland. The default is maximum non-regeneration: `NR_Dist = 120`. A shorter distance increases non-regenerated area, and a longer distance decreases non-regenerated area.
14. `WRITE_OUT_FILE`: Chooses whether to save the output file; `WRITE_OUT_FILE = TRUE` saves the output file, and `WRITE_OUT_FILE = FALSE` does not save the output file. If nothing is specified, default is to save: `WRITE_OUT_FILE = TRUE`.
15. `blackC`: Chooses how the global warming potential (GWP) of black carbon is computed; `blackC = TRUE` assigns a GWP of 900, and `blackC = FALSE` assigns a GWP of 1 (equivalent to CO_2). If nothing is specified, default is to treat black C the same as CO_2: `blackC = FALSE`, which is recommended as our current understanding is that black C does not behave like the main greenhouse gases.

Outputs from `CALAND()`:

Output files are written to caland-3.0.0/outputs/ (unless a sub-directory is specified differently from the default `outdir = ""`). There are two outputs files:

- Main output .xls file: `<scenario_name>_output_<tags>.xls`
    - Sign (+/-) of output carbon values: carbon emissions versus land carbon uptake depends on the sign and the varibale name:
        - Carbon variables in which a negative value indicates carbon emissions and positive value indicates land carbon uptake:

        - variable names containing "den": a positive value indicates land carbon uptake.
            - variable names containing "Gain_C_stock" but not "Atmos": a positive value indicates land carbon uptake.
            - all other stock variables except those containing "Loss_C_stock" or "Atmos"

    - Carbon variables in which a positive value indicates carbon emissions and negative value indicates land carbon uptake:

        - variable names containing "CumCO2", "CumCH4eq", CumBCeq", "AnnCO2", "AnnCH4eq", AnnBCeq"

        - variable names containing "Loss_C_stock"

        - variable names containing "Atmos"

        - Precision: to the integer for ha, Mg C, and Mg C/ha

- Filename description: "_output_" is appended to the input scenario name, followed by a series of tags that denote (i) which type of input value was used (mean, mean+/-sd, min, or max) for carbon density, historical carbon fluxes, and the 'soil conservation' soil carbon flux on Cultivated lands; (ii) how black carbon was accounted for; and (iii) the level of forest non-regeneration following high-intensity wildfire.

    - Key for "" labeling of filename:
    - D = carbon density
    - A = historical carbon accumulation (flux)
    - S = soil carbon accumulation under 'soil conservation' management
    - No variable specified means that the same value type was used for the three variables
    - sd = standard deviation
    - mean, max, min are self explanatory
    - `+` = add (applied to sd only)
    - `-` = subtract (applied to sd only)
    - BC1 = black carbon treated like CO_2
    - BC900 = black carbon segregated as a greenhouse gas with a global warming potential of 900
    - `NR<number>` = non-regeneration threshold distance [m] to edge of a high-severity burn patch, above which forest will not regenerate (i.e., converts to shrubland)

- Log file output of soil carbon depletion:

`<scenario_name>_output_<tags>_land_cats_depleted_of_soil_c_&_sum_neg_cleared-<sum>.csv`

- Filename description: Same as above plus an aggregate sum of soil carbon depletion across all land categories at the end of the filename.

(3) plot_caland.r

Overview of `plot_caland()`

The `plot_caland()` function, defined in the plot_caland.r file, compares outputs from the `CALAND()` model. It is designed to compare the `CALAND()` output .xls file for the baseline scenario to any number of alternative scenarios (at least one required), and will output a suite of individual data tables (.csv files) and corresponding graphics (.pdf files) that summarize various variables at your desired level of spatial aggregation of region, land type, and ownership combination(s). The aggregation level can be zoomed out to the entire State of California (summing across all regions, land types, and ownerships) or as granular as a specific region, land type, and ownership combination (i.e., land category), such as North Coast, Forest on Private lands.

Sensitivity tests of individual practices

`plot_caland()` has the ability to generate per area outputs for sensitivity tests of individual practices by designating `INDIVIDUAL = TRUE` in `plot_caland()`. These diagnostics are designed to estimate effects of a practice *in isolation*. Thus, these outputs are only valid when the following conditions are met:

- Comparison of a static run (i.e., baseline with only ecosystem exchange enabled, no LULCC, no management practices, no wildfire, except when evaluating effects on fire emissions) with a run with a single practice enabled. One exception is for evaluating avoided conversion effects, in which case LULCC has to be on in the baseline so that urban growth rate can be reduced in the scenario.
- For any runs with restoration/reforestation/afforestation/LULCC, the land type must be "All_land" in order to capture the dynamics of LULCC and wildfire; any other land type will not give valid results.

These effects change over time because most practices have effects beyond the year(s) of implementation (e.g. rangeland compost application, forest harvest, forest fuel reduction practices). One consideration is that restoration activities have secondary effects on land use/cover change related to forcing conversion of other land types and the proportion of land types burned by wildfire.

Example `CALAND()` input scenario files designed for sensitivity testing can be found in /caland-

3.0.0/inputs/ind_sims_27aug2018.

Input files to `plot_caland()`

The input .xls files for `plot_caland()` are the main .xls output files from `CALAND()`, located in caland-3.0.0/outputs/ (unless the sub-directory `data_dir` is specified differently from the default of no subdirectory (`data_dir = ""`) (see Arguments section below). Two or more of these files are required, one of which will serve as the reference baseline scenario and the other as an alternative scenario that tests the impacts of change(s) to the baseline. Additional alternative scenarios can be compared to the baseline in a single run as well.

Arguments in `plot_caland()`:

1. `scen_fnames` is a vector of scenario output file names written by `CALAND()`.
    - The first scenario file in `scen_fnames` must be the name of the baseline scenario to which the other scenarios will be compared. For example: `scen_fnames = c("Baseline.xls", "A.xls", "B.xls", "C.xls", "D.xls")`.
        - The number of scenario files in `scen_fnames` must correspond directly to the labels in `scen_snames` (see argument #2).
2. `scen_snames` vector of abbreviated scenario labels which will be printed in the plot legends (8 characters maximum for each label).
3. `data_dir` path to the directory containing the `CALAND()` output files; do not include the "/" character at the end; default is `data_dir = "./outputs"`.
4. `reg` vector of individual regions to plot, including the entire State of California `All_region`; can be any number of available regions. Default is all of them:
   `reg = c("All_region", "Central_Coast", "Central_Valley", "Delta", "Deserts", "Eastside", "Klamath", "North_Coast", "Sierra_Cascades", "South_Coast", "Ocean")`.
5. `lt` vector of land types to plot; can be any number of available types. Default is all of them:
   `lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass")`.
6. `own` array of ownerships to plot; can be any number of available types. Default is the aggregation of all ownerships (All_own): `own = "All_own"`. However any number of them can be specified:
   `own = c("All_own", "BLM", "DoD", "Easement", "Local_gov", "NPS", "Other_fed", "Private", "State_gov", "USFS_nonwild")`.
7. `figdir` folder within data_dir to save the graphs (.pdf) and corresponding data tables (.csv); do not include the "/" character at the end. Default is: `figdir = "figures"`.
8. `INDIVIDUAL` Indicates whether a sensitivity test is being performed on scenarios that isolate the effects of individual practices. This test is only valid if the scenarios were configured for this purpose. The default is not to compute these outputs: `INDIVIDUAL =`

`FALSE`.

9. `units_ha` TRUE = units of `plot_caland()` outputs will be in "ha" (same as `CALAND()` outputs), FALSE = units of `plot_caland()` outputs will be converted to "ac". Default is `units_ha = "FALSE"`.
10. `blackC` TRUE = black GWP equal to 900, FALSE = black GWP equal to 1. Default is `blackC = FALSE`. Note: this needs to match the `CALAND()` `blackC` argument that was used.
11. `blackC_plot` TRUE = plot BC, CO2, and CH4, FALSE = plot only CO2 and CH4 (BC added to CO2 which is only valid if black C is FALSE). Default is `blackC_plot = FALSE`. Note: this needs to match the `CALAND()` `blackC` argument that was used.
12. `last_year` the last year to plot; this should be the final run year + 1 because cumulative and area outputs reflect previous years e.g., 2050 is the final run year, but everything is defined and output to 2051, so last_year = 2051. Default is `last_year = 2051`.

Example of `plot_caland()` arguments used to compute the 2010 through 2050 effects of Alternative A scenario relative to the Baseline scenario in the Draft California 2030 Natural and Working Lands Climate Change Implementation Plan:

```
plot_caland(scen_fnames = c("NWL_Historical_v6_default_RCP85_output_mean_BC1_NR120.xls", "NWL_Alt_A_v6_default_RCP85_output_mean_BC1.xls"), scen_lnames = c("Baseline", "Alt_A"), scen_snames = c("BASE", "AltA"), data_dir = "./outputs", reg = c("All_region", "Central_Coast", "Central_Valley", "Delta", "Deserts", "Eastside", "Klamath", "North_Coast", "Sierra_Cascades", "South_Coast", "Ocean"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own = c("All_own"), figdir = "figures", INDIVIDUAL = FALSE, units_ha=FALSE, blackC = FALSE, blackC_plot = FALSE, last_year = 2051)
```

Optimizing processing time

It will take several hours to run `plot_caland()` for five scenarios in all individual land type and region combinations (including `"All_land"` and `"All_region"`) with all ownerships aggregated (`"All_own"`) in R or RStudio. Assuming you have at least four cores on your computer, the processing time can be reduced by about half by splitting up `plot_caland()` into separate instances and running them in command line (not in R or RStudio). Specifically, you can split up each scenario comparison (five scenarios = four comparisons; one baseline compared to four alternative scenarios) into four instances each. Then split up the regions into the four instances for each scenario comparison (11 regions total, including `"Ocean"`, and all regions (`"All_region"`). You will run each separate instance simultaneously via command line. Thus, to optimize processing time for 5 scenarios including the baseline, you will create 16 individual instances in command line (4 scenario comparisons x 4 region groupings). Here are 16 individual instances you could run in command line if you had `caland()` .xls output files for five scenarios (e.g., "Baseline.xls", "A.xls", "B.xls", "C.xls", "D.xls"):

Baseline x A

```
plot_caland(scen_fnames = c("Baseline.xls", "A.xls"), scen_snames = c("Baseline","A"), reg = c("All_region", "Ocean"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "A.xls"), scen_snames = c("Baseline","A"), reg = c("Central_Coast", "Central_Valley", "Delta"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "A.xls"), scen_snames = c("Baseline","A"), reg = c("Deserts", "Eastside", "Klamath"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "A.xls"), scen_snames = c("Baseline","A"), reg = c("North_Coast", "Sierra_Cascades", "South_Coast"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

Baseline x B

```
plot_caland(scen_fnames = c("Baseline.xls", "B.xls"), scen_snames = c("Baseline","B"), reg = c("All_region", "Ocean"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "B.xls"), scen_snames = c("Baseline","B"), reg = c("Central_Coast", "Central_Valley", "Delta"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "B.xls"), scen_snames = c("Baseline","B"), reg = c("Deserts", "Eastside", "Klamath"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "B.xls"), scen_snames = c("Baseline","B"), reg = c("North_Coast", "Sierra_Cascades", "South_Coast"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest",
```

```
"Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"),
own=c("All_own"))
```

Baseline x C

```
plot_caland(scen_fnames = c("Baseline.xls", "C.xls"), scen_snames = c("Baseline","C"), reg =
c("All_region", "Ocean"), data_dir = "", figdir = "", lt = c("All_land", "Water", "Ice",
"Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest",
"Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"),
own=c("All_own"), INDIVIDUAL = FALSE))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "C.xls"), scen_snames = c("Baseline","C"), reg =
c("Central_Coast", "Central_Valley", "Delta"), lt = c("All_land", "Water", "Ice", "Barren",
"Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow",
"Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "C.xls"), scen_snames = c("Baseline","C"), reg =
c("Deserts", "Eastside", "Klamath"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse",
"Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow",
"Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "C.xls"), scen_snames = c("Baseline","C"), reg =
c("North_Coast", "Sierra_Cascades", "South_Coast"), lt = c("All_land", "Water", "Ice",
"Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest",
"Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"),
own=c("All_own"))
```

Baseline x D

```
plot_caland(scen_fnames = c("Baseline.xls", "D.xls"), scen_snames = c("Baseline","D"), reg =
c("All_region", "Ocean"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert",
"Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh",
"Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "D.xls"), scen_snames = c("Baseline","D"), reg =
c("Central_Coast", "Central_Valley", "Delta"), lt = c("All_land", "Water", "Ice", "Barren",
"Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow",
"Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "D.xls"), scen_snames = c("Baseline","D"), reg =
c("Deserts", "Eastside", "Klamath"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse",
"Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow",
"Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

```
plot_caland(scen_fnames = c("Baseline.xls", "D.xls"), scen_snames = c("Baseline","D"), reg = c("North_Coast", "Sierra_Cascades", "South_Coast"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

Outputs from `plot_caland()`

Output files consist of a suite of graphs (.pdf) and corresponding data tables (.csv), which are written to caland-3.0.0/`data_dir`/`figdir`/ within each region, land type, and ownership combination directory, where `data_dir` and `figdir` are arguments to `plot_caland()`. Naming of the .pdf and .csv filenames is automatic and determined from the `caland()` .xls outout filenames that are assigned to `scen_fnames`. Specifically, the text preceding "_output_….xls" is used as the .pdf and .csv filenames.

Notes on outputs

- Seagrass carbon is represented in totality by the soil carbon pool, which means that all other pools, except total organic carbon, will have zero values.

(4) plot_scen_types.r

Overview of `plot_scen_types()`

The `plot_scen_types()` function, defined in the plot_scen_types.r file, is designed to use the .csv outputs from `plot_caland()` to plot individual land types for a designated variable and all available scenarios in the .csv file. You also specify which land types, region (or all regions aggregated), and ownership (or all ownerships aggregated) to plot for each scenario.

Input files to `plot_scen_types()`

The .csv input files, created by `plot_caland()`, are assumed to be in caland-3.0.0/`data_dir`/`figdir`, in the appropriate land type and ownership directories. The `plot_scen_types()` output files will go directly into caland-3.0.0/`data_dir`/`figdir`/`reg`/`own`, which should be the same as used in `plot_caland()`.

Arguments in `plot_scen_types()`

1. `varname`: name of variable to plot. See the outputs from `plot_caland()`; the name is between the ownership and "_output.csv" in these file names; do not include the surrounding "_" characters.
2. `ylabel`: label for y-axis corresponding to the units of `varname`, and whether they are

changes from baseline (i.e., `varname` ending in "_diff") or absolute values. Note that this function does not convert units so the output units of your desired plotting variable must be correctly matched with the .csv file.

3. `data_dir`: The path to the directory containing the `CALAND()` output files, which also contains `figdir`; do not include the "/" character at the end; default is `data_dir = "./outputs"`.
4. `file_tag`: Additional tag to file name to note what regions, landtypes, and/or ownerships are included; default is `file_tag = ""` (nothing added).
5. `reg`: Vector of region names to plot; default is: `reg = c("All_region", "Central_Coast", "Central_Valley", "Delta", "Deserts", "Eastside", "Klamath", "North_Coast", "Sierra_Cascades", "South_Coast")`.
6. `lt`: Vector of land types to plot; can be any number of available land types; default is: `lt = c("Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all")`.
7. `own`: Vector of ownerships to plot; can be any number of available ownerships; default is: `own = c("All_own")`
8. `figdir`: The directory within `data_dir` containing the .csv data to plot, and where to save the figures that `plot_scen_types()` creates; do not include the "/" character at the end.

Notes on `plot_scen_types()`:

- Plotting the Ocean region does not provide any comparison with land types because only Seagrass exists in the Ocean.
- Seagrass has only a subset of the `plot_caland()` output files, so if including All_region make sure that the desired `varname` is available for Seagrass.
- Seagrass is not in the default land type list.

Example of `plot_scen_types()` arguments that will create a single figure comparing the total organic C stock in land types over time, aggregated across all regions and ownerships, for each scenario (Alternative A and Baseline):

```
plot_scen_types(varname = "All_orgC_stock", ylabel = "MMTC", data_dir = "./outputs", file_tag = "", reg = c("Central_Coast", "Central_Valley", "Delta", "Deserts", "Eastside", "Klamath", "North_Coast", "Sierra_Cascades", "South_Coast", "All_region"), lt = c(Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all"), own = c("All_own"), figdir = "figures")
```

Outputs from `plot_scen_types()`

Output files consist of a suite of graphs (.pdf) and corresponding data tables (.csv), which are

written to caland-3.0.0/`data_dir`/`figdir`/ within each region directory, where `data_dir` and `figdir` are arguments to `plot_scen_types()`. Naming of the .pdf and .csv filename is automatic and determined from the region, ownership, scenario name in the input .csv file, the `varname` argument, and an optional `file_tag` argument.

(5) plot_uncertainty.r

Overview of `plot_uncertainty()`

The `plot_uncertainty()` function, defined in the plot_uncertainty.r file, is designed to plot shaded uncertainty bounds around average values for a single variable over time for each scenario using .csv outputs from `plot_caland()`. This requires having run `CALAND()` and `plot_caland()` three times each for each desired scenario; once for the mean inputs, and the other two times for lower and upper uncertainty bounds. For example, using the following combination of inputs to `CALAND()` will generate minimum and maximum emissions:

- Low emissions: low initial carbon density (i.e., mean-SD) and high carbon fluxes (i.e., mean+SD)
- High emissions: high initial carbon density (i.e., mean+SD) and low carbon fluxes (i.e., mean-SD)

Input files to `plot_uncertainty()`

Inputs to `plot_uncertainty()` are .csv output files from `plot_caland()`. Within each .csv file, there can be any number of scenarios. Each group of scenarios must have a corresponding .csv file for the mean, low, and high emissions. It is possible to plot up to three groupings (a, b, c) with `plot_uncertainty()`. Three scenario groups amounts to a total of nine .csv input files to `plot_uncertainty()`: three input .csv files (mean, low, and high emissions) for each group. All .csv files have matching formats. For a single group ("group a"), they are assumed to be in caland-3.0.0/outputs/mean, caland-3.0.0/outputs/low, caland-3.0.0/outputs/high, respectively, unless `figdir` is specified differently than the default: `figdir = c("mean","low","high")`.

Arguments in `plot_uncertainty()`

1. `start_year`: year to start plotting; default is `start_year = 2010`.
2. `end_year`: year to end plotting; default is `end_year = 2051`.
3. `varname`: name of a single variable to plot (see the .csv output filenames from `plot_caland()`); the variable name is between the ownership and "_output.csv" in the filename. However, do not include the surrounding "_" characters.
4. `ylabel`: label for y-axis corresponding to the units of `varname`, and whether they are

changes from baseline (i.e., `varname` ending in "diff") or absolute values. Note that this function does not convert units so the output units of your desired plotting variable must be correctly matched with the .csv file.
    - Here are some examples:
        - `ylabel = "Change from Baseline (MMT CO2eq)"`
        - `ylabel = "Change from Baseline (MMT CO2eq)"`
        - `ylabel = "MMT CO2eq"`
        - `ylabel = "Change from Baseline (MMT C)"`
        - `ylabel = "MMT C"`
        - `ylabel = "Change from Baseline (Mg C/ha)"`
        - `ylabel = "Mg C/ha"`
        - `ylabel = "Change from Baseline (Mg C/ac)"`
        - `ylabel = "Mg C/ac"`
5. `file_tag`: tag to add to end of the new file names created by `plot_uncertainty()` (e.g., to note what time period is plotted); default is `file_tag = ""` (nothing added).
6. `data_dir_a`: the path to the directory containing the three folders of `plot_caland()` outputs (mean, low, and high emissions) for scenario group a; do not include the "/" character at the end; default is `data_dir_a = "./outputs"`.
7. `figdirs_a`: a vector of three folder names within `data_dir_a` containing the .csv data files to plot. The folder names must be assigned in order of mean, low, and high; do not include the "/" character at the end of each folder name. The default is `figdirs_a = c("mean", "low", "high")`; thus, the .csv files for the mean and lower and upper uncertainty bounds are assumed to be in `data_dir_a`/mean, `data_dir_a`/low, and `data_dir_a`/high, respectively, and in the appropriate region, land type, and ownership directories. The figures will be written to the folder representing the mean.
8. `scenarios_a`: a vector of one or more scenario names for group 'a' that are listed in the Scenario column in the .csv file containing `varname` for the mean.
9. `scen_labs_a`: a vector of one or more scenario labels; must match the same number of elements in `scenarios_a`, and correspond directly to the order of elements in `scenarios_a`.
10. `data_dir_b`: the path to the directory containing the three folders of `plot_caland()` outputs (mean, low, and high emissions) for scenario group b; do not include the "/" character at the end; default is `data_dir_b = NA` (i.e., no group b), which will plot only scenario group a.
11. `figdirs_b`: a vector of three folder names within `data_dir_b` containing the .csv data files to plot. The folder names must be assigned in order of mean, low, and high; do not include the "/" character at the end of each folder name. The default is `figdirs_b = NA` (i.e., no group b). However if you choose to plot a group b, the location of the .csv files for the mean and lower and upper uncertainty bounds should follow the same logic as group a. The figures will be written to the folder representing the mean.
12. `scenarios_b`: a vector of one or more scenario names for group 'b' that are listed in the Scenario column in the .csv file containing `varname` for the mean.
13. `scen_labs_b`: a vector of one or more scenario labels; must match the same number of

elements in `scenarios_b`, and correspond directly to the order of elements in `scenarios_b`.

14. `data_dir_c`: the path to the directory containing the three folders of `plot_caland()` outputs (mean, low, and high emissions) for scenario group c; do not include the "/" character at the end; default is `data_dir_c = NA` (i.e., no group c), which will only plot scenario group a, or group a and b.
15. `figdirs_c`: a vector of three folder names within `data_dir_c` containing the .csv data files to plot. The folder names must be assigned in order of mean, low, and high; do not include the "/" character at the end of each folder name. The default is `figdirs_c = NA` (i.e., no group c). However if you choose to plot a group c, the location of the .csv files for the mean and lower and upper uncertainty bounds should follow the same logic as group a. The figures will be written to the folder representing the mean.
16. `scenarios_c`: a vector of one or more scenario names for group 'c' that are listed in the Scenario column in the .csv file containing `varname` for the mean.
17. `scen_labs_c`: a vector of one or more scenario labels; must match the same number of elements in `scenarios_c`, and correspond directly to the order of elements in `scenarios_c`.
18. `reg`: a vector of one or more region names to plot; default is `reg = "All_region"` (i.e., aggregated across all regions), but can be any number of available regions: "All_region", "Central_Coast", "Central_Valley", "Delta", "Deserts", "Eastside", "Klamath", "North_Coast", "Sierra_Cascades", "South_Coast".
19. `lt`: a vector of one or more land types to plot; default is `lt = "All_land"` (i.e., aggregated across all land types), but can be any number of available land types
20. `own`: a vector of one or more ownerships to plot; default is `own = "All_own"` (i.e., aggregated across all ownerships), but can be any number of available ownerships.

Notes on `plot_uncertainty()`:

- Plotting the ocean region does not provide any comparison with land types because only seagrass exists in the ocean.
- Seagrass has only a subset of all the .csv files output from `plot_caland()`; thus, if including Seagrass make sure that the desired `varname` is available.

Outputs from `plot_uncertainty()`

Output plots (.pdf) and corresponding data (.csv) will be saved to the mean folder for scenario group a.

Instructions for your first test runs with CALAND

Once you have completed installing R or R and RStudio, and downloading CALAND, you will have a local copy of all the R scripts, four example input scenario files, a carbon input file, all the raw files for creating new input files on your local computer, and the tools to use them. Now you can use the carbon and scenario input files that were already created with

`write_caland_inputs()` to do a test run with `CALAND()` and two of the plotting functions: `plot_caland()` and `plot_scen_types()`.

Load the `CALAND()`, `plot_caland()`, and `plot_scen_types()` functions in R

1. Open the following R files containing the functions (located in your caland-3.0.0/ directory): CALAND.r, plot_caland.r, and plot_scen_types.r.

2. In R or RStudio, find the console in the bottom-left corner. This is where you will interact with R and run the the functions.

3. Set the working directory to caland-3.0.0/ by typing the following into the console: `setwd("<your_path>/caland-3.0.0/")`

4. Load the functions in each file by clicking anywhere in your script window, and then choose Edit→Run all (in R), or click the Source button in the top-right corner of the editor window (in RStudio). Alternatively, you can highlight the entire code, and then press Ctrl+R (in R) or Ctrl+Enter (in RStudio).

Run the functions in R

4. Read the `CALAND.r` section.

5. Run `CALAND()` twice using two example sets of arguments:

    - Run the historical baseline scenario using RCP8.5 climate change effects on wildfire and ecosystem carbon fluxes, mean initial carbon density, mean carbon accumulation inputs, maximum non-regeneration, and with black carbon counted as CO_2. Type (or copy and paste) the following into the R console:
    `CALAND(scen_file="NWL_Historical_v6_default_RCP85.xls", c_file_arg = "carbon_input_nwl.xls", indir = "", outdir = "", start_year = 2010, end_year = 2101, value_col_dens = 7, ADD_dens = TRUE, value_col_accum = 7, ADD_accum = TRUE, value_col_soilcon=8, ADD_soilcon = TRUE, NR_Dist = 120, WRITE_OUT_FILE = TRUE, blackC = FALSE)`

    - Next, run the Alternative A scenario using the same arguments except for the scenario filename `scen_file`. Type (or copy and paste) the following into the R console: `CALAND(scen_file="NWL_Alt_A_v6_default_RCP85.xls", c_file_arg = "carbon_input_nwl.xls", indir = "", outdir = "", start_year = 2010, end_year = 2101, value_col_dens = 7, ADD_dens = TRUE, value_col_accum = 7, ADD_accum = TRUE, value_col_soilcon=8, ADD_soilcon = TRUE, NR_Dist = 120, WRITE_OUT_FILE = TRUE, blackC = FALSE)`

6. Read the `plot_caland()` and `plot_scen_types()` sections above and make some plots by running the following examples in order. Note that `CALAND()` must be finished running before starting `plot_caland()`, and `plot_caland()` must be finished running before starting `plot_scen_types()`, as each function uses output files from the previous function as input files to the next function.

- `plot_caland()` example. The following will compare `CALAND()` outputs for Alternative Scenario A to the Baseline Scenario at the Statewide level (`All_region`, `All_land`, and `All_own`), as well as each land type aggregated across all regions (`All_region`) and all ownerships (`All_own`):

```
plot_caland(scen_fnames = c("NWL_Historical_v6_default_RCP85_output_mean_BC1_NR120.xls", "NWL_Alt_A_v6_default_RCP85_output_mean_BC1.xls"), scen_snames = c("Baseline","A"), reg = c("All_region"), lt = c("All_land", "Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all", "Seagrass"), own=c("All_own"))
```

- `plot_scen_types()` example. The following will plot the change in total organic C stock of all the individual land types in Scenario A relative to the Baseline Scenario, aggregated across all regions and ownerships:

```
plot_scen_types(varname = "All_orgC_stock_diff", ylabel = "Change from Baseline (MMTC)", data_dir = "./outputs", file_tag = "", reg = c("All_region"), lt = c(Water", "Ice", "Barren", "Sparse", "Desert", "Shrubland", "Grassland", "Savanna", "Woodland", "Forest", "Meadow", "Coastal_marsh", "Fresh_marsh", "Cultivated", "Developed_all"), own = c("All_own"), figdir = "figures")
```