# GCAM Data Processing System

Page Kyle

March 2014

## Summary

This document provides an overview to the GCAM data processing system developed for the expansion to 32 regions and update to the 2010 calibration year. It is assumed that the user already has downloaded and installed all necessary software, libraries, and data files, and has the system installed to the point that it can be run without crashing. This guide provides an overview of how to do the types of modifications that the system was built to handle easily, and includes a description of each folder, including the locations of other documentation files that may be helpful in learning and customizing the file system.

## Introduction

The GCAM data processing system provides a central location for the generation of nearly all XML inputs files necessary to run GCAM. Exceptions include some input files for MAGICC, the solver configuration file, the main configuration file, and any scenario-specific policy files. Within the data system, all dependencies are tracked, such that a change in one component will force a re-build of all components that are possibly influenced by the change made. The system is designed to flexibly handle any country-to-region mapping list, any model start year from 1971 to 2009, and any model final calibration year from 1972 to 2010. All data required for the generation of input files is processed in all historical years for which we currently have data (1971-2010), so even if not running these periods in the model, users can easily generate graphs of annual historical data that are consistent with the model input and output data. One caveat to this is that the end use data for the buildings and transportation sectors is still built from one “snapshot” year, wherein a single base year (generally around 2005) is used to downscale energy to specific services and in specific regions.

Aside from the present document, there are some additional files in the \_common/documentation that may be helpful in learning how the data system works. Data\_system\_structure.xlsx has the mapping between sector codes in the energy and agricultural system and the names of the sectors. It also has a worksheet indicating the name, location, and purpose of all (1050+) files in the GCAM data system. Most of the descriptions are extracted from the comments of the files themselves. In the future there may be alternative ways of storing and presenting the metadata behind each file.

## Operating the System

Running the “xml” command of the makefile in the root directory will execute all necessary code all the way through to the generation of XML files. The “clean” command will wipe the output of any prior runs of the data system, so the next “xml” after a “clean” will run all code files. A run from a clean workspace currently takes about 5 minutes. Note that the make utility doesn’t always pick up every dependency; particularly after modifications that affect a large number of code files, the system may miss files that should have been re-run. This usually causes either the data system or GCAM to crash. As a general rule, if the changes made will affect a large number of modules (e.g. a change to the country/region mapping), it’s probably best to either clean the workspace prior to re-running, or manually force the system to re-run all code files that are known to be affected by the change(s). One can manually force R code files to be run at the next make by deleting the log files associated with the given code files; the makefile treats the log files as the primary outputs of the code files. The file paths to the log files are as follows:

\*-code/logs/[R code filename].R.log

## Making Modifications to Input Data

In general, each module’s data folder contains assumptions, mappings and data from other sources, in many CSV data tables, and in some R code files that create R data objects. All numerical data in these folders may potentially be changed, but in general, any changes to file names, R data object names, or column names in the CSV tables would need to be complemented with changes to the R processing code. Note that R is case sensitive, so be careful to not change the cases of any names in the code or input data ( .R and .csv).

While the full data system takes about 5 minutes to run, most of this time is in level1, and within level1, most of it is in the first files: L100, L101, and L102. These files read in, downscale, and aggregate the following large databases: all FAOSTAT data, the IEA energy balances, the CDIAC emissions inventories, and the SSP socioeconomic database. Modifications that don’t require these code files to re-run are comparably fast.

### Model Time Periods

Running with different model time periods is simple; just change the model\_base\_years and model\_future\_years in the following file, and save:

modeltime-data/assumptions/A\_modeltime\_data.R

I’d recommend to then delete all logs from level2 processing code; this can be done from a command prompt in the root directory of the GCAM data system as follows:

rm ./\*-code/logs/L2\*.log

Model time periods are currently set with the c() and seq() functions in R. c(), or combine, can take any number of arguments, in this case years, so modifying the data in a c() statement is simple. However seq(), or sequence, generally takes three arguments: from, to, and by. One can use a sequence as an argument to the combine function; for example, to run 5-year timesteps to 2050 and 15-year timesteps thereafter, this would work:

from:

model\_future\_years <- c( seq( 2015, 2100, 5 )

to:

model\_future\_years <- c( seq( 2015, 2050, 5 ), seq( 2065, 2095, 15 ) )

After changing the model years, all level2 processing code will need to be re-run, but no level1 code does, because the level1 files are processed for all historical years and all possible future years; level1 processing code does not depend on the specific model years that are being written to the final XML files. The reason why I recommend deleting the level2 log files is that this change influences a large enough number of code files that the makefile may miss one or two, and if it does, the level2 code files are modular (i.e., level2 code files do not depend on other level2 code files), so any inconsistency here will not produce any warning messages until GCAM crashes.

### Country to Region Mapping

Changing the country-to-region mapping requires a little bit more work, both for the user and the computer. There are six files with region-specific assumptions that will have to be modified:

\_common/mappings/iso\_GCAM\_regID.csv

\_common/mappings/GCAM\_region\_names.csv

aglu-data/assumptions/A\_bio\_frac\_prod\_R.csv

aglu-data/assumptions/A\_biocrops\_R\_AEZ.csv

aglu-data/assumptions/A\_soil\_time\_scale\_R.csv

energy-data/assumptions/A\_regions.csv

Note that there are two separate files for mapping (1) from iso codes to numbered GCAM regions, and (2) from these numbered regions to the region names that will be written to the final XML files. In general, the level1 data is processed for GCAM region numbers, but the regions are not named until level2. The main reason for this differentiation is to allow non-alphabetical region ordering in the intermediate data tables and final XML products. An ancillary benefit of this approach is that one can change the names of the regions, and as long as the change doesn’t affect the region number assignments or the country-to-region mapping, then the level1 code will not need to be re-run.

After modifying the six country-to-region mapping files, all code files will need to be re-run except for the L100.\* files that downscale inventory data to modern-day nations. As these do take a while to re-run, it may be worthwhile to keep those log files intact (rather than cleaning the entire workspace). But given the large number of code files that need to be re-run, it is probably a good idea to clean all other log files prior to running the system again.

### User-Specified Assumptions

Many changes (e.g. efficiencies, costs, logit exponents, shareweights, etc.) will involve changes to the assumptions and mappings files found in the following folders:

\_common/assumptions/

\_common/mappings/

\*-data/assumptions/

\*-data/mappings/

The specific files are detailed in a separate document (\_common/mappings/Data\_System\_Structure.xlsx), but for most of these files, one can simply update the data contained therein, save it, and re-run. For simple changes whose effects are confined to individual sub-modules, this might only involve re-running one R code file and re-building one XML file, which may only take a few seconds. For files that are early in the processing, however, changes may require a complete or near-complete re-build. For instance, in the socioeconomics module, historical population is used in downscaling the agricultural and energy data to all modern-day countries, so changes here may cause every single code file to re-run at the next make. In contrast, in the energy-data/assumptions folder, many of the CSV files pertain only to the sub-module indicated in the prefix of the file names (e.g. A21.\*), where a “sub-module” indicates component of the energy system, such as the buildings or industrial sectors. Modifying these files will typically cause only one or two code files to re-run at the next make.

Note that in the energy-data/assumptions folder, all CSV files can be built from sub-module-specific .xlsm files that have a macro to write out all worksheets as CSV files, saved in excel’s present working directory (generally the directory of the last file opened). While this will over-write more files than were actually changed, it might not increase the number of code files that need to be re-run, as the R code files tend to be structured around the sub-modules.

One can run alternative technology scenarios without changing code by modifying the assumptions in the input CSV files, re-running, and changing the names of the resulting XML files or the folders that contain them afterwards. Changes to the mappings files are also possible, and level0 files may have data that users want to modify or update. Still, many desired changes will require code changes. There is a lot that is possible, and some things will be easier than others. Anyone wanting to make changes is highly encouraged to contact me.

## Data System Structure

The folder structure of the data processing system is specified in the following file:

\_common/mappings/domainmapping.csv.

This section provides a text description of what sorts of files are found in each folder.

There are five modules at present: aglu, emissions, energy, modeltime, and socioeconomics. There is a place-holder for water data and code, though it isn’t clear whether that data should instead be processed within the existing modules. Each module has two folders in the root directory: one for code and one for data. There is a “\_common” folder for shared utilities and data, and an “xml” folder where the final files for model input are written out. The modules are to some extent inter-dependent, though as a practical matter these interdependencies have been minimized.

In general, each module contains data and code in separate directories. Within the data, there are assumptions, mappings, and levels 0, 1, and 2 data, and sometimes rawdata and GIS data. **Rawdata** is data exactly as it was downloaded from the internet, often with a URL provided. Not all data from other sources was downloaded from the internet, and of the data that was, not all of it was saved separately in this folder. Its purpose is really to archive data from sources that is frequently updated (e.g. FAOSTAT), which may facilitate comparisons when we incorporate the data updates. **GIS** data is currently processed by Yuyu Zhou; what is in the \*-data/GIS folders of the GCAM data system is the output of that processing. The input data and code files necessary to generate the GIS data are not currently contained in the GCAM data system, due to the size of the input data files. In any case, this output tends to stay quite stable, as the gridded inventories are unlikely to be updated, and the number of categories written out generally exceed what is needed for our current processing. Note that this data tends to be at the national scale—at present, the GCAM data system does not carry the capability to create sub-national GCAM regions.

The **assumptions** refer to data assumptions that are specific to the given module; they may be in .csv or .R format. Assumptions and mappings shared across modules are found in the “\_common” directory. **Mappings** generally translate from (a) the categories in other data sources to GCAM’s categories; (b) from categories in other sources to intermediate categories that are used in processing the data; or (c) from these intermediate processing categories to the categories in GCAM, or (d) between several intermediate categories. It isn’t always readily apparent how the mapping files are used without looking at the code. **Level0** refers to data from other sources that has been formatted for input to R: that is, in CSV format, with one header row and as many ID columns as need be. Special characters are removed from column names, and data has been formatted to remove apostrophes, quotation marks, accents, and other characters that would be misinterpreted in the read-in to R. **Level1 data** is created by **level1 code**, which takes data from inventories and assumptions and processes it into CSV tables that are used in various stages of generating the parameters read into the model. Level1 data is CSV tables that are notated in the first few rows, which have (a) the code file that created it, (b) the timestamp when it was created, (c) a description of the data in the table, and (4) what the units are. Many modules have L100.\* data and code files for doing country-level downscaling of inventory data that either has multi-country regions, or has countries such as the USSR that contained multiple present-day countries at some point in the historical time series. **Level2 data** is created by **level2 code**, which takes data from assumptions, level0, level1, and mappings, and generates the CSV files that have every parameter read into the model. Level2 code also dynamically generates **xml-batch** files, which script the generation of XML files from CSV files and a common model interface header (\_common/headers/ModelInterface\_headers.txt). The level2 data tables are designed to be read by the model interface; the structure of these tables corresponds to what is indicated in the model interface headers, and also the R file that specifies the column names and order of these level2 data files (\_common/assumptions/level2\_data\_names.R). The format for these tables is typically “molten”—that is, all of the data is found in a single column, with all other columns used for identification purposes. This maximizes the flexibility of the final XML while using a single, static model interface header file. It is a departure from the data processing methods used in GCAM 3.0, where the model years, calibration years, region names, and region ordering were all typically invariant, and as such could be specified in the header file. The tradeoff is that these level2 data CSV files are not formatted to maximize the amount of information that can fit on a computer screen at one time.

All processing code files also generate 2-3 types of log files, written to the **logs** folders: .Rout, .R.log, and .d files. .Rout files are described below, and are only generated when R is run from the makefile. The **.R.log** files contain all of the printlog statements from running a code file; there is one created per code file that is run, and the name of the log file is the same as the name of the code file that created it. They are treated as the primary output of each code file, and if one wants to run selected code files (and associated dependencies) from the makefile, one dependable way is to delete the .R.log files associated with the code files that one wants to run. The **.d** files contain the dependencies associated with each code file. The .d files are similar to the .R.log files in that each code file generates one, and its file name is the same as the log file name. The **.Rout** files are different in that there is only one for each level of code that is run, and what’s in the .Rout file is simply from the last file that was run. Still, these files are quite useful for debugging crashes, because they contain the last line of code read and the warning message from R.

The final XML products are generated in the **xml** directory, under which each module has its own sub-directory.

The following section identifies each individual folder (domain) within the data processing system, along with notes about the files contained therein. Wherever they are indicated as “generic”, refer to the description above.

1. **\_common**: common utilities and data shared by multiple modules
   1. **assumptions**: assumed data that may apply to multiple modules
   2. **documentation:** files that help to illustrate and explain the structure of the systems in GCAM and in the data processing system
   3. **headers:** the .R files contain R functions and utilities that are generic to the whole system and to specific modules. This folder also has the model interface header that scripts the generation of XML input files from level2 data (CSV) files, and a batch XML template file, used as the basis for the generating batch XML files.
   4. **mappings:** contains the central domainmapping.csv file which specifies the name and location of each folder called in the data processing system, and also the country-to-region mapping files.
   5. **ModelInterface**: the ModelInterface is used to build XML files from CSV files.
2. **aglu-data**: data files (both input and output) of the agriculture and land use component of GCAM
   1. **assumptions**: The A\_aglu\_data.R file contains many data assumptions, and is called by every aglu data processing code file. Files that contain the string “\_R” contain region-specific data, and as such need to be updated when changing the country-to-region mapping.
   2. **GIS**: Output of AgLU GIS processing; all necessary files to generate these data files are available in the GCAM 3.0 agriculture and land use data system.
   3. **ISIMIP**: This folder contains the output of GIS data processing of the ISIMIP global gridded crop model intercomparison exercise, for selected scenarios. The file names indicate the scenarios, with the scenario components delimited by “\_” (climate model, crop model, emissions scenario, irrigation, crop, start year, end year).
   4. **level0:** generic
   5. **level1**: generic
   6. **level2**: generic
   7. **mappings**: generic
   8. **rawdata**: generic
3. **aglu-processing-code**: code files for processing agriculture and land use data.
   1. **GIS:** This folder contains the GIS processing code, but note that the files can not be run as the necessary files in the rawdata/GIS folder are not included in the GCAM data system.
   2. **level1:** generic. Note that the LA\* files are not dependent on the energy module, whereas the LB\* files are.
   3. **level2**: generic
   4. **logs**: generic
   5. **xml-batch**: generic
4. **emissions-data**: contains data pertaining to all emissions, including CO2 and various secondary non-CO2 gases and other airborne substances that are modeled in GCAM
   1. **assumptions:** generic, but note that some energy code is dependent on these files
   2. **level0**: generic
   3. **mappings**: generic
   4. **rawdata**: generic
5. **emissions-processing-code**: code for processing the non-CO2 emissions data from other sources into the corresponding parameters in GCAM. Note that CO2-relevant parameters are handled in the energy system.
6. **energy-data**: data for the energy module
   1. **assumptions:** The A\_\*\_data.R files contain sub-module-specific information, in order to allow one to change parameters that are only relevant for certain sectors (sub-modules) without forcing all energy code to be re-run. The prefixes on each file name are identified in:

\_common/documentation/Data\_system\_structure.xlsx file

The A\_regions.csv file contains information specific to each region. Needs to be updated if country-to-region mapping is changed.

All of the files named A\*.csv may be built from macro-enabled excel workbooks, alphabetized at the end of the file list in this folder. Need to be careful when running the macro that it’s writing out to the correct “present working directory,” which generally defaults to the last file that was opened.

* 1. **GIS**: output of GIS scripts not contained in the GCAM data system
  2. **level0:** generic
  3. **level1:** generic
  4. **level2:** generic
  5. **mappings**: generic
  6. **rawdata:** generic

1. **energy-processing-code**
   1. **GIS**: [empty]
   2. **level1**: generic. Note that the LA\* files are not dependent on the aglu module, whereas the LB\* files are.
   3. **level2**: generic
   4. **logs**: generic
   5. **xml-batch**: generic
2. **modeltime-data**: this is only assigned its own module because it does not logically fit within any of the others. It is very small and only contains one file that one might want to change. Note that any changes to this file will require all level2 code files in all modules to be re-run, but no changes to level1 code files in any module.
   1. **assumptions:** the file in this folder (A\_modeltime\_data.R) contains the model timesteps that are going to be used.
   2. **level2**: generic. Note that there is no data from other inventories here, so there is no level1 data processing.
3. **modeltime-processing-code:** code for generating the model timesteps
   1. **level2**: generic
   2. **logs**: generic
   3. **xml-batch**: generic
4. **socioeconomics-data**: All data required to generate the model inputs for population and GDP, along with country-level historical inventories, and a generic parameterization of income elasticities for aggregate buildings, industry, and transportation sectors that is designed to be consistent with the model inputs in GCAM 3.0.
   1. **assumptions**: generic. Some of these assumptions may be called by other modules.
   2. **GIS** [empty]
   3. **level0**. Generic. Note that some of the data tables in the GDP.xlsx file are not currently written to CSV.
   4. **level1:** generic
   5. **level2**: generic. Note that this includes inputs for the SSPs, in order to have a wide range of possible socioeconomic scenarios; nevertheless, the XML files do not modify all sectors of the model according to the SSP narratives.
   6. **mappings:** generic
   7. **rawdata**: generic
5. **socioeconomics-processing-code**
   1. **GIS**: [empty]
   2. **level1**: generic.
   3. **level2**: generic
   4. **logs**: generic
   5. **xml-batch**: generic
6. **xml**
   1. **aglu-xml:** xml output from the aglu module
   2. **energy-xml:** xml output from the energy module
   3. **modeltime-xml:** xml output from the modeltime module
   4. **socioeconomics-xml:** xml output from the socioeconomics module