**Aquacrop for python**

*Last updated 15/09/2023 by* [*raphael.payet-burin@alumni.epfl.ch*](mailto:raphael.payet-burin@alumni.epfl.ch)

This tutorial is to use the python-aquacrop library (<https://github.com/aquacropos/aquacrop>) and enables to feed the model from global soil and harvesting date data, as well as climate data from the CLIOPT model (not global/publicly available). This script also provides a framework to manage different scenarios and obtain outputs at a given catchment scale.

The overall organization of the tool is the following:

* *run\_aquacrop.py* is the main script running the aquacrop model for the chosen crops, time period, climate scenarios and providing absolute and relative yields.
* **background\_data** is the folder with datasets enabling to feed the model with climate, soil, and planting date data
* **prepare\_data** is the folder with the scripts*generate\_weather.py, get\_soil\_texture.py,* and *get\_planting\_date.py* that process the background data for the aquacrop model
* **data** is the folder with the processed background data feeding the aquacrop model
* **outputs** is the folder with the outputs from the model (crop yields)

Main limitations of this framework:

* While global soil and planting date data is provided, this framework is not linked to any global climate data, this must be provided separately.
* The framework does not take into account irrigation or other farming practices, however that can be added in the code.
* The framework does not provide a calibration step for crop parameters, for this reason, relative yields are also provided which is more insightful than absolute yields values without a calibration step.
* In general, this framework only explores a limited number of features of the aquacrop model, the github of the model (<https://github.com/aquacropos/aquacrop>), provides a list of examples of what can be done with the model and could be integrated into this framework. An online interactive version of aquacrop also exists here: <https://tinyurl.com/aquaplan>.

# Install

The following packages are necessary:

pip install pandas geopandas rasterstats rioxarray setuptools aquacrop

On windows (because of the aquacrop package – most likely):

**-you need to install msvc compiler, Visual Studio Build Tools (select MSVC – 2015 works).**

-use the development environment, it will use purely python stuff no c++, this is already done in the code by calling os.environ['DEVELOPMENT'] = 'DEVELOPMENT' (no action needed)

# Input parameters

This enables to feed the model with the minimum input parameters, not that climate data is not provided at the global level in this framework.

## Crops

The crop data must be a .csv file in the following form (the order does not matter):

|  |  |  |  |
| --- | --- | --- | --- |
| **crop** | **nculture** | **planting\_date** | **planting\_data** |
| Maize | SumMaize | 15-Nov | Maize |
| Tomato | SumVege | 15-Nov |  |
| SugarCane | Sugarcane | 15-Nov |  |
| PaddyRice | SumRice | 15-Nov | Rice |
| … |  |  |  |

**crop** is the crop name according to the aquacrop nomenclature (predefined crops include: 'Barley', 'Cotton', 'DryBean', 'Maize', 'PaddyRice', 'Potato', 'PotatoLocal', 'Quinoa', 'Sorghum', 'Soybean', 'SugarBeet’, ‘SugarCane', 'Sunflower', 'Tomato', 'Wheat', 'localpaddy', custom crops can also be defined but require to be implemented in this code), **nculture** enables to rename crop names in the output, **planting\_date** is the planting date – which will only be used if no catchment specific planting date is provided (see Planting dates section), **planting\_data** is the corresponding name in the Crop Calendar Dataset (<https://sage.nelson.wisc.edu/data-and-models/datasets/crop-calendar-dataset/>, see Planting dates section) to get catchment specific planting dates, if blank the planting\_date parameter will be used.

## Catchment shapefile

The following climate, soil, and planting date parameters are calculated at a “catchment” scale for which the aquacrop outputs will be provided. Hence a shapefile must be located in the **data** folder. In the *get\_soil\_texture.py* and *get\_planting\_date.py* scripts (see next sections) its name must be indicated as well as the unique ID field of the shapefile (*catchID* in the code). The projection *EPSG:4326* should be used to match the global datasets, this can potentially be adjusted in the code.

## Climate parameters

The climate data must be provided in a .csv file in the following form (the order does not matter):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **scenario** | **ncatch** | **Date** | **Precipitation** | **ReferenceET** | **MaxTemp** | **MinTemp** |
| observed | R\_UpperZambezi | 2020-01-01 | 4.2 | 5.51 | 32.78 | 21.2 |
| observed | R\_Kholombidzo | 2020-01-01 | 10.8 | 3.72 | 30.13 | 19.25 |
| observed | R\_Mupata | 2020-01-01 | 4.61 | 4.88 | 30.68 | 19.44 |
| … |  |  |  |  |  |  |

**scenario** indicates the name of different climate scenarios, **ncatch** is the catchment name/ID which is the spatial resolution matching the catchment shapefile of soil parameters, **Date** is the date in the format year-month-day, **Precipitation** is the daily precipitation (mm), **ReferenceET** is the reference evapotranspiration (mm), **MaxTemp** is the maximum daily temperature (°C), and **MinTemp** is the minimum daily temperature (°C).

Note that the model can be run at the monthly step, but it is not advised.

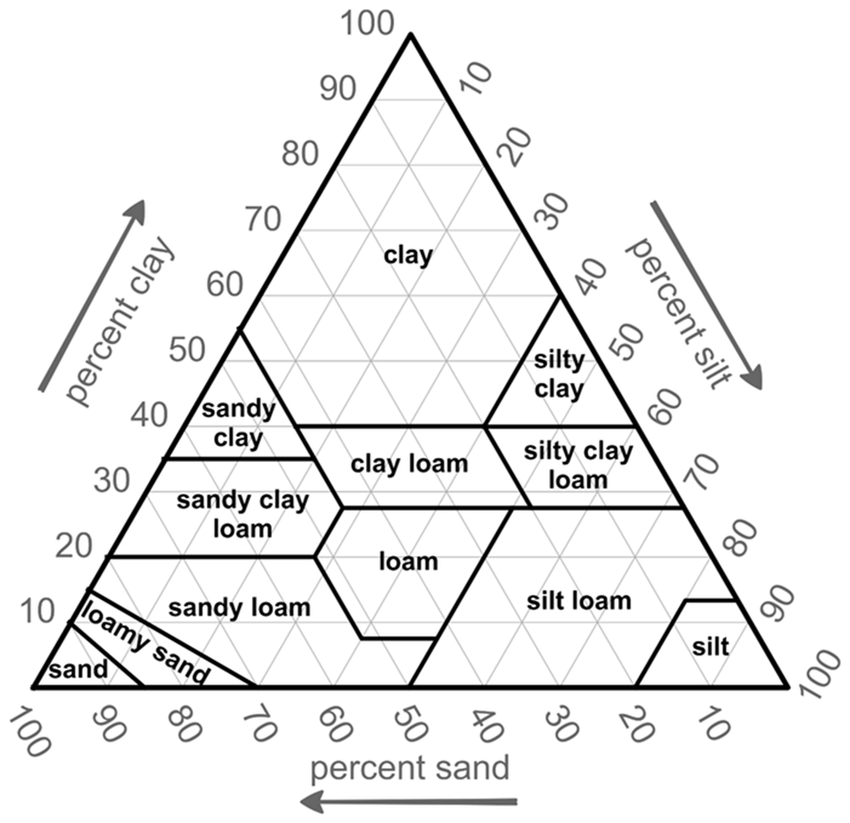
The *generate\_weather.py* script converts CLIOPT derived model results to the aquacrop format. This script is however specific to the Zambezi study case, it is advised to directly provide the data in the format described above.

## Soil parameters

The soil parameters must be a .csv with the following form:

|  |  |
| --- | --- |
| **ncatch** | **aqua\_TEX** |
| R\_Baroste | Sand |
| R\_BatokaGorge | Sand |
| R\_Bubi | LoamySand |
| … |  |

**ncatch** the catchment name/ID which is the spatial resolution, **aqua\_TEX** is the soil texture, which is based on the USDA soil texture classes: *Clay, SiltyClay, Clay, SiltyClayLoam, ClayLoam, Silt, SiltLoam, SandyClay, Loam, SandyClayLoam, SandyLoam, LoamySand, Sand*



The *get\_soil\_texture.py* script generates the soil texture from the Harmonized World Soil Database (<https://www.fao.org/soils-portal/data-hub/soil-maps-and-databases/other-global-soil-maps-and-databases/en/>), the documentation of the database in in <https://www.fao.org/3/aq361e/aq361e.pdf>

Inputs to the script are the Harmonized World Soil Database (composed of a csv file and a shapefile, if you directly downloaded the database you have to generate the shapefile from the .mdb file), and the catchment shapefile representing the resolution at which the model should be run. The script will output the main topsoil type for each catchment in the above format.

One limitation is that specific crops might be located in areas of the catchment with a different topsoil type.

## Planting dates

Planting dates by catchment must be provided in a .csv file with the following format:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ncatch** | Maize | PaddyRice | Soybean | Potato | Cotton | Sorghum | DryBeans |
| R\_UpperZambezi | 10/14 | 11/17 | 1/17 | 9/1 | 12/26 | 12/11 | 10/2 |
| R\_Kholombidzo | 10/16 | 12/7 | 1/17 | 10/16 | 12/26 | 12/17 | 10/2 |
| R\_Mupata | 11/21 | 11/17 | 1717 | 9/1 | 12/8 | 8/23 | 10/2 |
| … |  |  |  |  |  |  |  |

**ncatch** is the catchment name/ID which is the spatial resolution, and Maize, PaddyRice, Soybean … are the different crops represented (following the aquacrop names) for which the planting date in the format month/day is provided. If planting dates are not provided in that csv file, a single planting date for all catchments will be used as provided in the crops input file.

The script *get\_planting\_date.py* enables to generate this table based on the Crop Calendar Dataset (<https://sage.nelson.wisc.edu/data-and-models/datasets/crop-calendar-dataset/>), and the catchment shapefile.

# Run the model

Once all data has been prepared, run the script *run\_aquacrop.py.* Make sure the link to the different datasets are correct, provide the start (startT) and end time (endT) of the crop simulation in the format *f”{year}/month/day”,* the climate data must be at least as long as these.

The model will generate a .csv file *Yields\_aquacrop\_daily.csv* in the **output** folder with the following format:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **scenario** | **ayear** | **acatch** | **acrop** | **Yield (tonne/ha)** | **YieldFactor\_mean** | **YieldFactor\_max** |
| observed | 2020 | R\_Baroste | SumMaize | 14.62109 | 0.999157 | 0.97242 |
| observed | 2020 | R\_Baroste | SumVege | 8.314043 | 1.024209 | 0.880648 |
| observed | 2020 | R\_Baroste | Sugarcane | 0.100237 | 0.954636 | 0.915299 |
| … |  |  |  |  |  |  |

**scenario** corresponds to the climate scenarios provided in the climate data, **ayear** are the years corresponding to the dates of the crop growth (based on the planting date if the crop growth overlaps the calendar year), **acatch** are the catchments as provided in the catchment shapefile, **acrop** the crops provided in the crops file under the nculture parameter, **Yield (tonne/ha)** is the absolute yield calculated by the aquacrop model, however, uncalibrated, this value might give limited insight, **YieldFactor\_mean** is the relative yield compared to the mean yield over the total time period (hence it can be used to multiply an average observed yield to get the weather impacted yield), **YieldFactor\_max** is the relative yield compared to the maximum yield over the time period (the assumption being the maximum yield will be the closest yield under no stress in ideal weather condition, hence this relative factor is the impact of water/temperature constraints).