pyDSSAT Documentation

Release 1.0

Xiaogang HE, Liqing PENG, He SUN

January 17, 2015

CONTENTS

Your can download a hardcopy of the pyDSSAT `documentation`.

CHAPTER

ONE

ABOUT PYDSSAT

1.1 Python Interface for Crop Model DSSAT

The python library pyDSSAT is a set of wrapper scripts which initialize, run and post process the Decision support system for agrotechnology transfer model (DSSAT). It is developed for the primary purpose of agriculture research based on DSSAT crop model, and for research in different fields such as climate, meteorology, ecology, and hydrology involving agriculture.

In general, this python library is not only convenient for beginners to use rather than execute original Fortran program on Linux, but also valuable for researchers to utilize crop model for different purposes. In particular, pyDSSAT wrap the fortran code into a callable class using f2py. It includes many enhancements over the terminal command line execution, including the ability of controlling input files, analyzing output files with Matplotlib capabilities, and providing a GUI toolkit for efficient interactive work. Many useful tools, such as output file convertor, time series analysis, are newly implemented in the library. More tools will be implemented continuously.

pyDSSAT is developed as part of the final project for the course APC524 (Software Engineering for Scientific Computing) at Princeton University.

CHAPTER

TWO

DOWNLOAD AND INSTALL

2.1 Before you install and use pyDSSAT

We assume:

- You have a valid license of DSSAT.

  Note: If you don't, check it out! (What's DSSAT?) To use DSSAT, you need to request the DSSAT software for Windows platform from DSSAT webpage.

- You're one of the DSSAT developers and have access to the source code of CSM.
- You have good knowledge of DSSAT and CSM in general.
- You know how to run CSM from the command line.
- You have GNU Fortran Compiler (gfortran) or a valid license of Intel Fortran Compiler (ifort) for Linux/Unix.

  Note: If you don't have the source code, you need to contact DSSAT CSM ver 4.5 Open Source project for it. After you have the source code, here is a tutorial about using DSSAT on Linux/Unix. (How can I compile and run DSSAT on Linux/Unix?)

2.2 Set Up DSSAT Environment

2.2.1 Copy Windows DSSAT45 Directory

Prepare your executing directory **EXEPATH** where you have all the input data files there (weather, soil, cultivar, ecotype, species, standard input, etc). We strongly recommend you copy the "DSSAT45" folder from Windows software under your account, e.g., ~/DSSAT45, which means **EXEPATH** =~.

2.2.2 Change File Path

No matter you run DSSAT from Linux or pyDSSAT, you need to set up the DSSAT environment. Use vi you can easily modify the path in the file DSSATPRO.v45:

```
cd ~/DSSAT45
vi DSSATPRO.v45
```

Change the windows directory format to linux directory format. Type the following command:

```
:%s!\!/!g
:%s!C:!EXEPATH!g
```

Note that **EXEPATH** is the directory where you put the "DSSAT45".

2.3 Install pyDSSAT Library

2.3.1 Git Clone

Enter the EXEPATH/DSSAT45 and install our pyDSSAT Library there.

To install pyDSSAT library, please make a clone of the git repository from pyDSSAT, by executing this command in your terminal, which includes the source code of our pyDSSAT python libraries. You should find a new directory "pyDSSAT under the folder EXEPATH/DSSAT45/" right now:

```
cd ~/DSSAT45
git clone https://github.com/XiaogangHe/pyDSSAT
```

2.3.2 Copy and Modify DSSAT Source Code

Enter the repository you just clone, and copy the DSSAT fortran source code (**SRCPATH** is the directory where you store your DSSAT source code) to the directory "src":

```
cd pyDSSAT
cp SRCPATH/*.FOR ./src
cp SRCPATH/*.for ./src
```

1. Change source code extensions from .FOR and .for to .f90:

   ```
   cp ./Tools/rename.sh ./src
   bash rename.sh
   ```

2. Change the following lines in ModuleDefs.f90 to appear as follows. Note 6 blanks before 1st character:

   ```
   !      CHARACTER(LEN=5), PARAMETER :: OPSYS = 'WINDO'   !DOS, Windows
         CHARACTER(LEN=5), PARAMETER :: OPSYS = 'LINUX'   !Linux, UNIX
   ```

3. In the UTILS.f90 source, there were two commented out definitions of the type of the function SYSTEM that need to be uncommented. The first was line 832:

   ```
   INTEGER ERR, I, LUN, LUNTMP, SYS  !, SYSTEM
   ```

   Change it to:

   ```
   INTEGER ERR, I, LUN, LUNTMP, SYS, SYSTEM
   ```

   The second one is line 914:

   ```
   INTEGER i, COUNT, LUNLST, LUNTMP, SYS !, SYSTEM
   ```

   Change it to:

   ```
   INTEGER i, COUNT, LUNLST, LUNTMP, SYS, SYSTEM
   ```

4. In CRSIMDEF.f90, change the following lines to read:

   ```
   !      CHARACTER(LEN=1),PARAMETER::SLASH = '\' !DOS, Windows
         CHARACTER(LEN=1),PARAMETER::SLASH = '/' !Linux, Unix
   ```

In case you didn't know, here is the tutorial about using DSSAT on Linux/Unix. (How can I compile and run DSSAT on Linux/Unix?)

2.3.3 Build Python Library

Compile DSSAT and generate the python library using f2py:

```
cd PyScripts
python ./make.py
```

2.4 Dependencies for install

2.4.1 Must Have

- Git
- Python: 2.5 or later
- NumPy: 1.2 or later: Numpy is an array library for Python numpy
- Matplotlib: python plotting library.
- NetCDF: NetCDF 4 library.

2.4.2 Strong Recommendations

- IPython: Interactive Python environment.

CHAPTER THREE

GETTING STARTED

This page will show you how easy it is to take control of our pyDSSAT model. There are two options to use pyDSSAT either through terminal or GUI as shown below.

3.1 Use pyDSSAT in the terminal

You can run DSSAT model using the following steps:

1. Import DSSAT_LIBRARY:

   ```
   import DSSAT_LIBRARY as DL
   ```

2. Specify crop type, weather station information, simulation start year, simulation end year, plant month, plant date and the running mode:

   ```
   DL.File(crop='Maize',soil='Sand',weather='DTCM',st_yr=1948,ed_yr=2010,plant_month=6,plant_date=1
   ```

3. Define the model and conduct the simulation:

   ```
   model=DL.Model('S','run.v45','../DSCSM045.CTR')
   model.run()
   ```

In case you want to do some post-process:

1. Load simulation output and its corresponding description file for DSSAT variables:

   ```
   out=DL.postProcess('../','POST-VAR-INFO.CDE')
   ```

2. Draw figures for different variables, e.g., yield, precipitation and temperature:

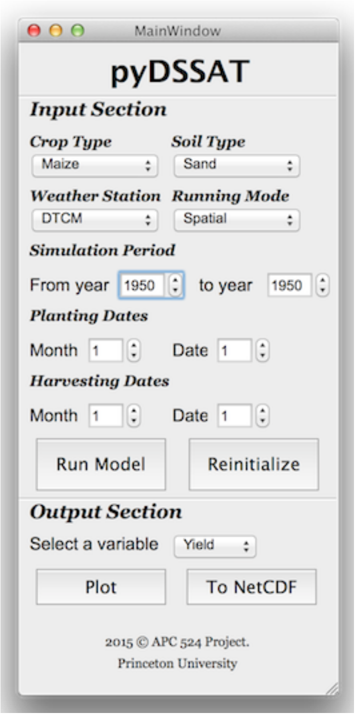
   ```
   out.drawTimeSeries('./Summary.OUT','HWAM')
   out.drawTimeSeries('./Summary.OUT','PRCM')
   out.drawTimeSeries('./Summary.OUT','TMAXA')
   ```

3.2 Use pyDSSAT GUI

You can also use our designed python GUI for pyDSSAT. To do this, you need to:

```
cd PyScripts
python ./DSSAT_GUI.py
```

Then you will be able to see the GUI like this:



MainWindow

pyDSSAT

Input Section

Crop Type: Maize

Soil Type: Sand

Weather Station: DTCM

Running Mode: Spatial

Simulation Period

From year 1950 to year 1950

Planting Dates

Month 1 Date 1

Harvesting Dates

Month 1 Date 1

Output Section

Select a variable: Yield

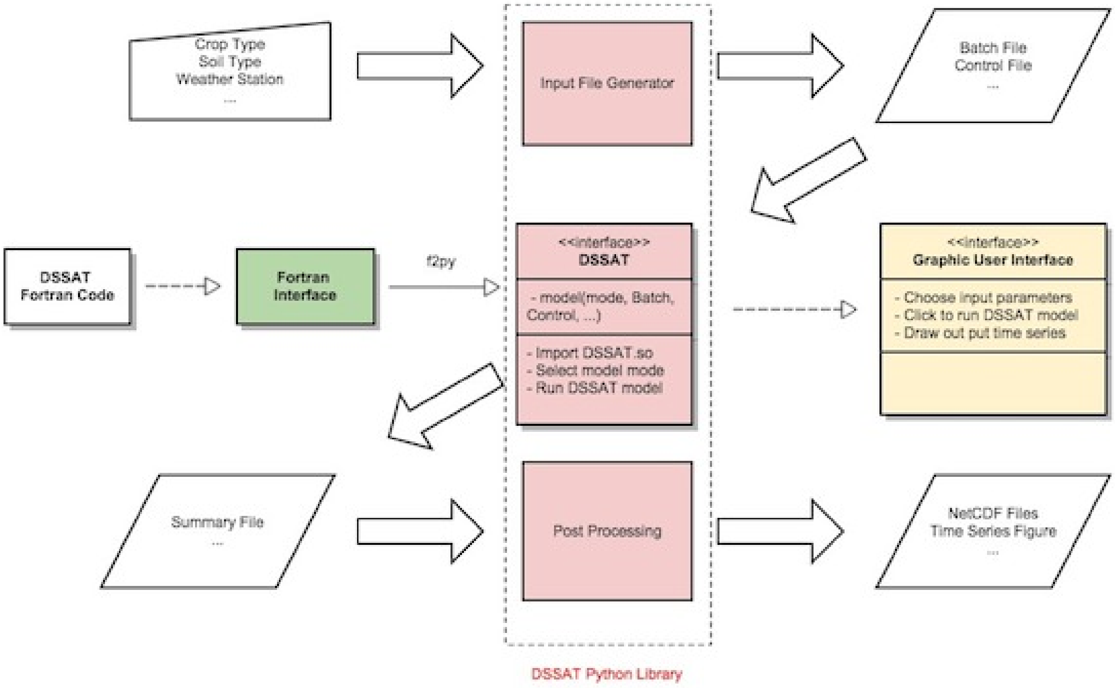
2015 © APC 524 Project.
Princeton University

CHAPTER

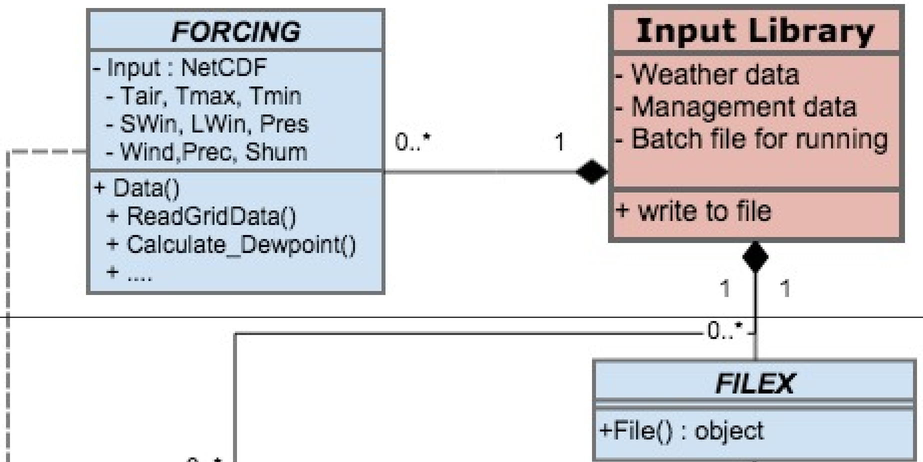
FOUR

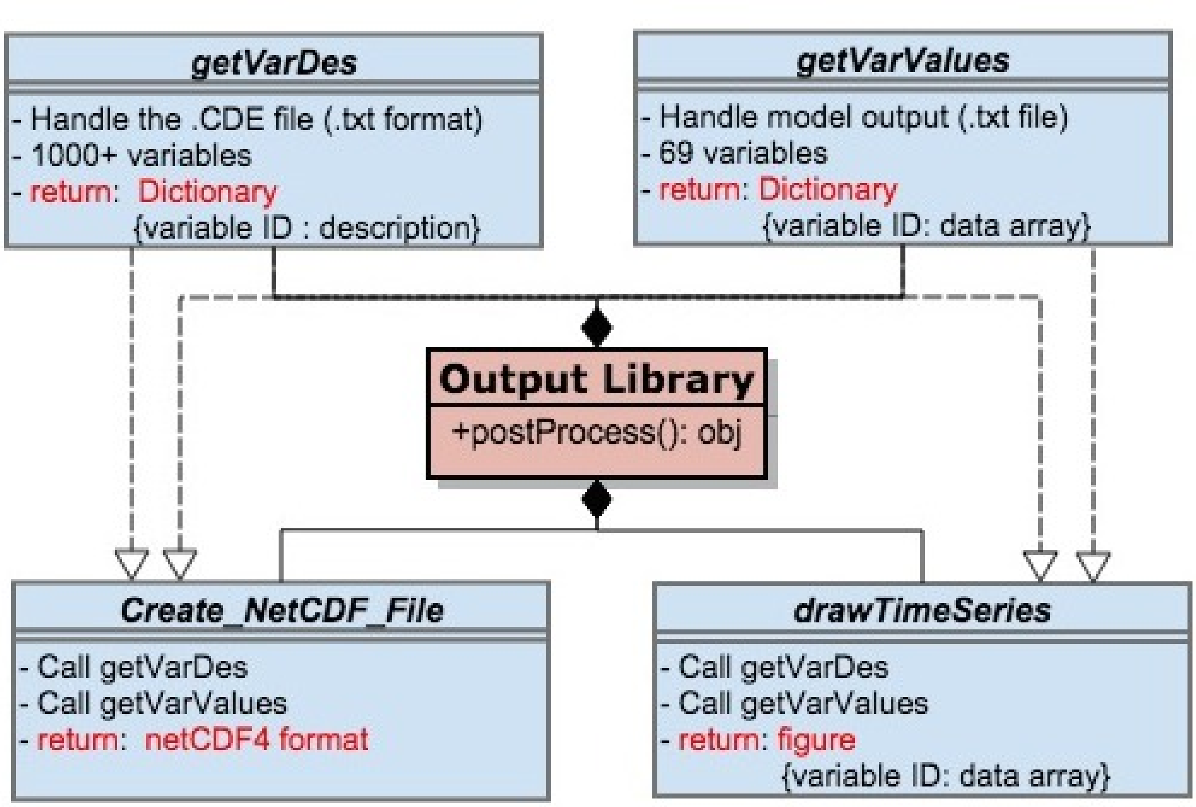
API

4.1 Import f2py generated library



4.2 Prepare the input files




getVarDes
- Handle the .CDE file (.txt format)
- 1000+ variables
- return: Dictionary
{variable ID : description}
getVarValues
- Handle model output (.txt file)
- 69 variables
- return: Dictionary
{variable ID: data array}
Output Library
+postProcess(): obj
Create_NetCDF_File
- Call getVarDes
- Call getVarValues
- return: netCDF4 format
drawTimeSeries
- Call getVarDes
- Call getVarValues
- return: figure
{variable ID: data array}

CHAPTER FIVE

BUG REPORTS

Find a bug?

If you find a bug in pyDSSAT, please send an email to hexg@princeton.edu.

Thanks!

CHAPTER

SIX

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*