# 1. How to run WOFOST using the PCSE package in PyCharm?

The aim of this section is show how to set up an environment to run WOFOST and run a simple example simulations. It is based on an example designed by Allard de Wit ([pcse\_notebooks/01 Getting Started with PCSE.ipynb at master · ajwdewit/pcse\_notebooks](https://github.com/ajwdewit/pcse_notebooks/blob/master/01%20Getting%20Started%20with%20PCSE.ipynb)).

## 1.1 How to install PyCharm?

The WOFOST crop growth model runs within a modeling framework called PCSE (Python Crop Simulation Environment). It is programmed in the Python programming language and, therefore, an environment to run Python is required to run the WOFOST crop growth model. For this purpose, we will use PyCharm. PyCharm is a IDE (Integrated Development Environment) from which Python code can be developed and run. It can be downloaded and installed using the following steps:

1. Browse to the download page of PyCharm: <https://www.jetbrains.com/pycharm/download/?section=windows>.
2. Running WOFOST through PCSE does not require the Professional Edition which requires a fee. Instead, we will use the free Community Edition. Browse halfway the webpage and click on the Download button for the Community Edition. Alternatively, use the following download link: <https://www.jetbrains.com/pycharm/download/download-thanks.html?platform=windows&code=PCC>. This will download an executable to download the PyCharm Community Edition. At the time that I downloaded it, this package is called “pycharm-community-2025.1.exe”, but this name may differ depending on the time that the package was downloaded.
3. Open the downloaded executable and follow the installation instructions.

## 1.2 How to set up a project in PyCharm to run WOFOST using the PCSE package?

The aim of this section is to explain how to set up a simple example WOFOST simulation in PyCharm.

1) Open PyCharm Community edition. This will open the following screen:

2) Click on "New Project"

3) The Location text box contains the location of the newly created project. The default location is C:\Users\<username>\PyCharmProjects\PythonProject. Change "PythonProject" in FirstWofostProject such that the project location becomes "C:\Users\<username>\PyCharmProjects\FirstWofostProject PythonProject"

4) Make sure that "Project venv" is selected as Intepreter type. If not, click on the "Project venv" button to change this.

5) Click on the "Create" button to create a new Python project.

6) The PCSE package needs to be installed in a virtual environment to run WOFOST before PCSE can be used. Click on the terminal button (or press Alt + F12) to open a terminal. This button is encircled in the figure below.

A screenshot of a computer

AI-generated content may be incorrect.

The last line of the newly opened terminal should be:

(.venv) PS C:\Users\{user\_name}\PycharmProjects\FirstWofostProject.

7. Type "pip install pcse" in the terminal and press Enter. This will install the PCSE package and its dependencies in your virtual environment.

8. In the example simulation, two other packages will be used as well. These are the Pandas package and the Matplotlib package. The Pandas package is a Python library for data analysis and manipulation. The Matplotlib package is a Python library for plotting. Type "pip install pandas" in the terminal to install Pandas and its dependencies and press Enter. Next, type "pip install matplotlib" in the terminal to install Matplotlib and press Enter.

## 1.3 How to run a simple Wofost simulation?

1) Right click on the FirstWofostProject folder in the Project tool window.

2) Choose New.

3) Choose Python file

4) Write "main" in the Name text box and press Enter. This will add a file called "main.py" to the FirstWofostProject project.

5. Double click on the main.py file to open it. Copy the following text in main.py:

import pcse.start\_wofost as start\_wofost

import matplotlib.pyplot as plt

import pandas as pd

wofostPP = start\_wofost(mode="pp")

wofostPP.run\_till\_terminate()

output = wofostPP.get\_output()

dfPP = pd.DataFrame(output).set\_index("day")

fig, axs = plt.subplots(nrows = 1, ncols = 2, figsize=(16,8))

dfPP["LAI"].plot(ax=axs[0], label="LAI", color='k')

dfPP["TAGP"].plot(ax=axs[1], label="Total biomass")

dfPP["TWSO"].plot(ax=axs[1], label="Yield")

axs[0].set\_title("Leaf Area Index")

axs[1].set\_title("Crop biomass")

r = axs[1].legend()

fig.autofmt\_xdate()

plt.show()

print(dfPP)

6) The code above runs an example Wofost simulation for winter wheat in Southern Spain grown under potential growth conditions. It generates time plots for the LAI, the total aboveground dry matter and the grain dry weight:

A screenshot of a graph

AI-generated content may be incorrect.

7). Close the figure with the time plots and watch the console. The output of the three last lines of the script (i.e. print(dfPP.head().to\_string(), print("..."), and print(dfPP.tail().to\_string())) shows the first and last lines of the dataframe that was generated by the code and used to generate this figure.

DVS LAI TAGP TWSO TWLV TWST TWRT TRA RD SM WWLOW

day

2000-01-01 0.000000 0.144690 105.000000 0.0 68.250000 36.750000 105.000000 0.005768 10.0 0.3175 None

2000-01-02 0.008867 0.154184 111.889710 0.0 72.728312 39.161399 111.889710 0.003312 11.2 0.3175 None

2000-01-03 0.017157 0.164025 119.031178 0.0 77.370266 41.660912 119.031178 0.002589 12.4 0.3175 None

2000-01-04 0.024293 0.173908 126.203520 0.0 82.032288 44.171232 126.203520 0.006309 13.6 0.3175 None

2000-01-05 0.032706 0.185033 134.276367 0.0 87.279639 46.996729 134.276367 0.003616 14.8 0.3175 None

...

DVS LAI TAGP TWSO TWLV TWST TWRT TRA RD SM WWLOW

day

2000-05-27 1.934169 0.0 18091.006102 8729.399813 3126.21567 6235.390619 1613.465879 0.0 60.0 0.3175 None

2000-05-28 1.953874 0.0 18091.006102 8729.399813 3126.21567 6235.390619 1613.465879 0.0 60.0 0.3175 None

2000-05-29 1.974056 0.0 18091.006102 8729.399813 3126.21567 6235.390619 1613.465879 0.0 60.0 0.3175 None

2000-05-30 1.995758 0.0 18091.006102 8729.399813 3126.21567 6235.390619 1613.465879 0.0 60.0 0.3175 None

2000-05-31 2.000000 0.0 18091.006102 8729.399813 3126.21567 6235.390619 1613.465879 0.0 60.0 0.3175 None

It can be seen that simulated values were stored for the following variables:

- DVS: Development stage (-)

- LAI: Leaf area index (ha leaf ha-1 ground)

- TAGP: Total aboveground dry matter production (kg dry matter ha-1 ground)

- TWSO: Total dry weight grain (kg dry matter ha-1 ground)

- TWLV: Total leaf dry weight grain (kg dry matter ha-1 ground)

- TWLV: Total stem dry weight grain (kg dry matter ha-1 ground)

- TWRT: Total root stem dry weight grain (kg dry matter ha-1 ground)

- TRA: daily transpiration rate (cm water d-1)

- SM: Soil moisture content (cm water)

8) Close PyCharm.

## 1.4 Clone repository from Github

The code from the previous simulation is also available at a GitHub repository (https://github.com/herman-berghuijs/wofost-tutorial). GitHub is a platform that allows programmers to store, manage, and share their code. The source code of Wofost is also available as a GitHub repository (https://github.com/ajwdewit/pcse). This subsection will demonstrate how to use GitHub by downloading the source code from the previous example from GitHub, set it up in PyCharm and run it.

1) Open PyCharm

2) Click on the "Clone Repository" button

3) Copy the URL of the repository of the tutorial (https://github.com/herman-berghuijs/wofost-tutorial) and paste it in the URL textbox.

4) Click Enter.

5) This will download a copy of the repository ("clone") in a newly created project wofost-tutorial (C:\<user\_name>\PyCharmProjects\wofost-tutorial).

6) The project wofost-tutorial includes a file requirements.txt that contains the names of all the additional packages that have to be installed. In order to install all of them, first a virtual environment needs to be created. To do so, click on the hamburger button (≡) and choose Settings -> Project: wofost-tutorial -> Python Interpreter -> Add interpreter -> Add local interpreter. After the virtual environment has been created, click OK to close the window. Click OK again to leave the settings window.

7) Open a terminal, and type "pip install -r requirements.txt" and click Enter.

8). Double click on the file 01\_FirstWofostTutorial.py to open the code. The code is identical to the one in the previous example. Run the code by clicking on the run button.

# 2. How to run WOFOST using the PCSE package in PyCharm and custom input files?

Follow the instructions of section 1.4 to clone the repository of the WOFOST tutorial, if you have not done this yet. The example simulation from the first section used predefined input data (i.e. the crop was winter wheat, grown somewhere in Northern-Spain, sown at January 1 2020). The aim of this section is to show how to run WOFOST simulations with custom input data. For this purpose, the repository contains a second script 02\_RunWofostCustomInput.py. It runs a simulation for the growth of winter wheat in Wageningen under potential growth conditions.

The repository also contains a folder "input data", which contains input files that can be used to run WOFOST. WOFOST needs four different types of input data to run:

- Agromanagement data: contains data related to crop management and system parameters (start of simulation, sowing/emergence date, harvest date, etc.).

- Crop data: contains crop and variety parameters.

- Soil data: contains data related to soil type and soil physical properties.

- Site data: contains site parameters that are not directly related to the soil properties.

- Weather data: contains daily observed weather data.

1) Open the script 02\_RunWofostCustomInput.py and study the code.

2) Open the input files in the folders .../input\_data and .../input\_data/crops and study its contents.

3) Run the code.

4) The code generates a time plot in the folder .../output\_data. This figure should look like:

A graph of two people

AI-generated content may be incorrect.

5) Close PyCharm.

# 3. How to run PCSE WOFOST through a Jupyter Notebook?

A Jupyter Notebook is a document that blends text, code, and code output. Allard de Wit created several Jupyter notebooks that explain on how to use PCSE and WOFOST (https://github.com/ajwdewit/pcse\_notebooks). The aim of this tutorial is to clone the pcse\_notebooks repository, install the Jupyter notebook packages, and run the notebooks.

1) Open PyCharm.

2) Click the "Clone repository" button

3) Copy the URL of the PCSE repository (https://github.com/ajwdewit/pcse\_notebooks.git) and paste it in the URL textbox.

4) Press the "Clone" button

5) You will be asked whether you want to create a virtual environment that installs the packages in requirements.txt. Chose C:\Users\<user\_name>\AppData\Local\Programs\Python\Python3<version\_no>\python.exe as an interpreter and click OK. Wait until all packages are installed in the virtual environment.

6) Open a terminal, enter "pip install jupyter" and press Enter.

7) After the jupyter packages has been installed, type "ipython kernel install --user --name=venv" in the terminal and press Enter. This will open Jupyter in your browser in the Project directory:

A screenshot of a computer

AI-generated content may be incorrect.

7) The project directory contains 15 Jupyter notebooks (i.e. files with the \*.ipynb extension) that contain tutorials for basic operations in PCSE and WOFOST and more advanced topics. Double click on the file 01\_Getting Started with PCSE.ipynb to open the first tutorial:

A screenshot of a computer

AI-generated content may be incorrect.

8) Click on "Python 3 (ipkernel)", open the drop down menu, and click on "venv" under "Start Python kernel". This ensures that the code in the Notebook will be run using the packages that were installed in the virtual environment.

9) Choose "Run all cells" under the menu item "Run" in the menu bar to run the entire Notebook.

# 4. How to run PCSE and WOFOST directly from the source code

In the previous sections, the PCSE package was installed in a virtual environment. Next, the scripts imported the PCSE package from the virtual environment and it was used in the WOFOSTsimulations. Rather than installing the PCSE package, it is also possible to run PCSE directly from the source code. This can be convenient if one, for example, wants to make changes the PCSE source code and tests how these changes affect the simulated result. The aim of this section is to show how PCSE can be run directly from the source code.

1) Open PyCharm.

2) Chose "Clone repository"

3) Copy the URL https://github.com/herman-berghuijs/wofost-tutorial-runsource in the URL textbox. This repository is almost identical to the repository of section 2 with one exception. The requirement.txt button does not include the PCSE library. Instead, it contains the names of the libraries that PCSE requires to run. Click on the "Clone" button.

4) You will be asked whether you want to create a virtual environment that installs the packages in requirements.txt. Chose C:\Users\<user\_name>\AppData\Local\Programs\Python\Python3<version\_no>\python.exe as an interpreter and click OK. Wait until all packages are installed in the virtual environment.

5) Double click on 01\_FirstTutorial.py and click the Run button. This will result in the following error message in the console:

Traceback (most recent call last):

File "C:\Users\herma\PycharmProjects\wofost-tutorial-runsource\01\_FirstWofostTutorial.py", line 1, in <module>

import pcse.start\_wofost as start\_wofost

ModuleNotFoundError: No module named 'pcse'

The reason that 01\_FirstTutorial .py cannot run is that the PCSE package and there is also no PCSE module in the repository.

6) Try to run 02\_RunWofostCustomInput.py. This will result in a similar error message:

Traceback (most recent call last):

File "C:\Users\herma\PycharmProjects\wofost-tutorial-runsource\02\_RunWofostCustomInput.py", line 2, in <module>

from pcse.base import ParameterProvider

ModuleNotFoundError: No module named 'pcse'

7) We will now add the PCSE package as a module to the project. Click on the hamburger button (≡) and chose "Project from version control"

8) Paste the URL of the PCSE source code (https://github.com/ajwdewit/pcse) in the URL textbox and click Enter. This will clone the PCSE source code in the directory C:\Users\{user\_name}\PycharmProjects\pcse

9. After the PCSE source code has been cloned, a window pops up that ask you how to open the project. Click on the "Attach" button.