



LOCOMOTION

Low-carbon society:

An enhanced modelling tool for the transition to sustainability

The pywiliam model

Simulating WILIAM using a Python model

LOCOMOTION project



This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 821105.



Universidad de Valladolid



UNIVERSITÀ DI PISA



AUSTRIAN ENERGY AGENCY



BASQUE CENTRE
FOR CLIMATE CHANGE
Klima Aldaketa Ikergai
Sustainability, that's it!



Centre of Economic Research
Analysis and Research



KAPE
CRES



FCIências
ASSOCIAÇÃO PARA A
INVESTIGAÇÃO E
DESENVOLVIMENTO
DE CIÊNCIAS



UNITED NATIONS
UNIVERSITY
UNU-EHS
Institute for Environment
and Human Security



EEB
European
Environmental
Bureau



CREAF



CARTIF



6.1 Contents

1. Main features of pywiliam
2. Installing Anaconda Python
3. Opening Anaconda prompt and moving to the pywiliam directory
4. Creating a virtual environment using conda
5. Running a simulation
6. Creating and simulating a new scenario
7. Additional configurations
8. Plotting the results
9. Additional tools (exporting netCDF to csv)

6.2 Main features of pywiliam

WILIAM model v1.1 translated to **Python** using the PySD library
(<https://github.com/SDXorg/pysd>)







Fully open source (MIT)



Multiplatform (GNU/Linux, Mac, Windows, etc)

PySD: System Dynamics Modeling in Python

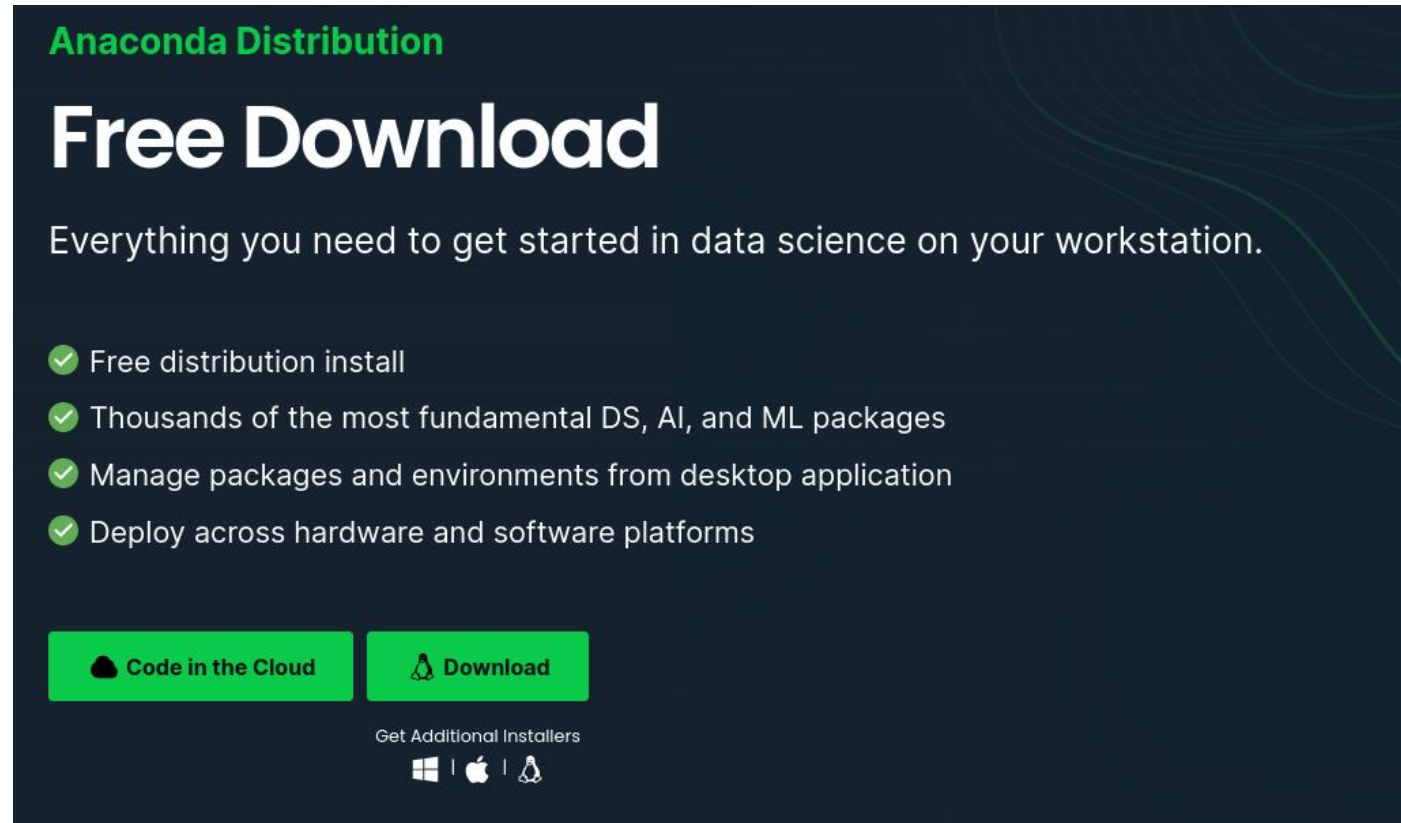
Eneko Martin-Martinez ^{1*}, Roger Samsó ^{1*¶}, James Houghton ², and Jordi Solé ^{1,3}

¹ CREA, Centre de Recerca Ecològica i Aplicacions Forestals, E08193 Bellaterra (Cerdanyola del Vallès), Catalonia, Spain ² Computational Social Science Lab, University of Pennsylvania, Philadelphia PA, 19104, United States of America ³ Departament de Dinàmica de la Terra i l'Oceà, Universitat de Barcelona (UB) E08007, Catalonia, Spain ¶ Corresponding author * These authors contributed equally.

DOI: [10.21105/joss.04329](https://doi.org/10.21105/joss.04329)

6.3 Installing Anaconda Python

<https://www.anaconda.com/download>





Anaconda Distribution




Free Download

Everything you need to get started in data science on your workstation.

- ✓ Free distribution install
- ✓ Thousands of the most fundamental DS, AI, and ML packages
- ✓ Manage packages and environments from desktop application
- ✓ Deploy across hardware and software platforms

 **Code in the Cloud**  **Download**

Get Additional Installers

 |  | 

6.4 Opening Anaconda prompt and moving to the pywiliam directory

```
Anaconda Powershell Prompt x + v
(base) PS C:\Users\ivaram> D:
(base) PS D:\> cd DATA_3
(base) PS D:\DATA_3> cd pywiliam-main
(base) PS D:\DATA_3\pywiliam-main> conda config --add channels conda-forge
(base) PS D:\DATA_3\pywiliam-main> conda create --name wiliam python=3.11 --file requirements.txt
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 4.8.3
  latest version: 23.10.0

Please update conda by running

  $ conda update -n base conda

WARNING conda.gateways.disk.delete:unlink_or_rename_to_trash(140): Co
s\boost-cpp-1.78.0-h9f4b32c_1.tar.bz2. Please remove this file manua
WARNING conda.gateways.disk.delete:unlink_or_rename_to_trash(140): Co
s\boost-cpp-1.78.0-h9f4b32c_1\Library\bin\boost_log_setup.dll. Pleas

to free file handles)

## Package Plan ##

environment location: C:\Users\ivaram\.conda\envs\wiliam

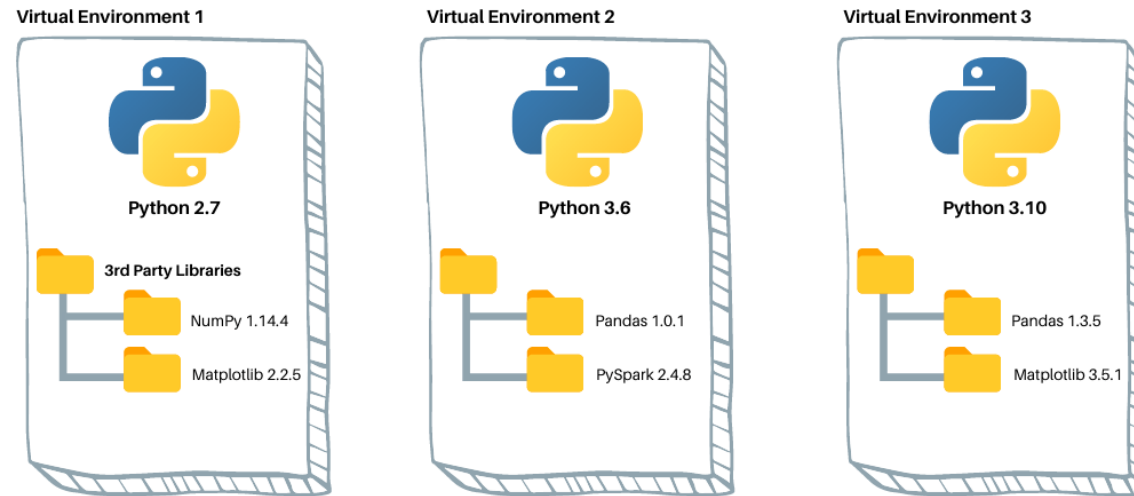
added / updated specs:
- dacite
- dask
- matplotlib
- netcdf4
- pysd[version='>=3.10']
- python=3.11
- tk

The following packages will be downloaded:
```

package	build		
aws-c-auth-0.7.5	hf425296_2	96 KB	conda-forge
aws-c-cal-0.6.9	h6f45060_0	54 KB	conda-forge
aws-c-common-0.9.5	hcfcfb64_0	214 KB	conda-forge
aws-c-compression-0.2.17	h6f45060_5	22 KB	conda-forge
aws-c-event-stream-0.3.2	h65f5141_5	54 KB	conda-forge
aws-c-http-0.7.13	h4076b5e_9	176 KB	conda-forge
aws-c-io-0.13.35	he032580_7	155 KB	conda-forge
aws-c-mqtt-0.9.8	he63eff1_1	154 KB	conda-forge
aws-c-s3-0.3.22	h40b9648_1	84 KB	conda-forge
aws-c-sdkutils-0.1.12	h6f45060_4	51 KB	conda-forge

6.5 Creating a virtual environment with conda

What are Python virtual environments and why we need them?



dataquest.io

6.5 Creating a virtual environment with conda

Add conda-forge channel:

```
conda config --add channels conda-forge
```

To create a virtual environment using conda, run:

```
conda create --name wiliam python=3.11 --file requirements.txt
```

The previous command created a virtual environment named `wiliam`. To activate it, run:

```
conda activate wiliam
```

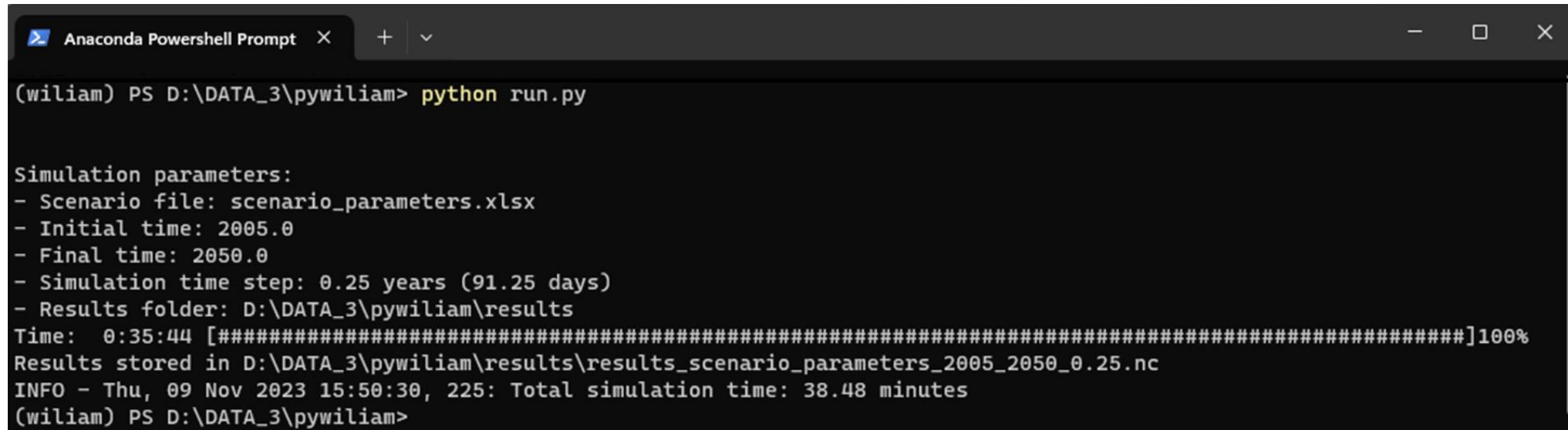
To run the wiliam model, you need to be inside the virtual environment, but if you want to exit it, run:

```
conda deactivate
```

6.6 Running a simulation

A wiliam simulation using default parameters can be run by simply typing:

```
python run.py
```

A screenshot of an Anaconda Powershell Prompt window. The title bar shows 'Anaconda Powershell Prompt' with standard window controls. The command prompt shows the user typing 'python run.py' at the prompt '(wiliam) PS D:\DATA_3\pywiliam>'. The output displays simulation parameters: Scenario file: scenario_parameters.xlsx, Initial time: 2005.0, Final time: 2050.0, Simulation time step: 0.25 years (91.25 days), and Results folder: D:\DATA_3\pywiliam\results. It also shows a progress bar at 100%, the results file path, and the total simulation time of 38.48 minutes.

```
(wiliam) PS D:\DATA_3\pywiliam> python run.py

Simulation parameters:
- Scenario file: scenario_parameters.xlsx
- Initial time: 2005.0
- Final time: 2050.0
- Simulation time step: 0.25 years (91.25 days)
- Results folder: D:\DATA_3\pywiliam\results
Time: 0:35:44 [#####]100%
Results stored in D:\DATA_3\pywiliam\results\results_scenario_parameters_2005_2050_0.25.nc
INFO - Thu, 09 Nov 2023 15:50:30, 225: Total simulation time: 38.48 minutes
(wiliam) PS D:\DATA_3\pywiliam>
```

Additionally, if the user only wants to play with scenario parameters, a good initialisation speedup may be achieved by loading the model parameters from a binary file, as follows:

```
python run.py -e wiliam/model_parameters/model_parameters.nc
```


6.7 Creating and simulating a new scenario

To create a new scenario, duplicate the scenario_parameters.xlsx file and give it a descriptive name (eu_green_deal.xlsx).

To run a simulation with the newly created scenario, run:

```
python run.py -x eu_green_deal
```

	A	B	C	D	E
1			LAND AND WATER		
2	LIST OF LAND AND WATER POLICIES AND HYPOTHESIS				
3					
4	AFFORESTATION_SP				
5	POLICY SCENARIO PARAMETERS	SWITCH_AFFORESTATION_SP	YEAR_INITIAL_FOREST_PLANTATIONS_SP	YEAR_FINAL_AFFORESTATION_SP	OBJECTIVE_AFFOR
6	REGIONS_IJ_UNIT	DMNL	YEAR	YEAR	DMNL
7	EU27	0	2015	2030	1
8	UK	0	2015	2030	1
9	CHINA	0	2015	2030	1
10	EASOC	0	2015	2030	1
11	INDIA	0	2015	2030	1
12	LATAM	0	2015	2030	1
13	RUSSIA	0	2015	2030	1
14	USMCA	0	2015	2030	1
15	ROW	0	2015	2030	1
16					
17	FOREST_PLANTATIONS_INCREASE_SP	POLICY_SWITCH_FOREST_PLANTATIONS_SP	YEAR_INITIAL_FOREST_PLANTATIONS	YEAR_FINAL_FOREST_PLANTATION	OBJECTIVE_FOREST_PLANTATIONS
18	POLICY SCENARIO PARAMETERS	SWITCH_FOREST_PLANTATIONS_SP	YEAR_INITIAL_FOREST_PLANTATIONS_SP	YEAR_FINAL_FOREST_PLANTATIONS_SP	OBJECTIVE_FOREST
19	REGIONS_IJ_UNIT	DMNL	YEAR	YEAR	DMNL
20	EU27	0	2025	2050	1
21	UK	0	2025	2050	1
22	CHINA	0	2025	2050	1
23	EASOC	0	2025	2050	1
24	INDIA	0	2025	2050	1
25	LATAM	0	2025	2050	1
26	RUSSIA	0	2025	2050	1
27	USMCA	0	2025	2050	1
28	ROW	0	2025	2050	1
29					
30	LAND_PROTECTION_BY_POLICY_SP	POLICY_SWITCH_PRIMARY_FOREST_PROTECTION_SP	POLICY_YEAR_INITIAL_PRIMARY_FOREST_PROTECTION_SP	POLICY_YEAR_FINAL_PRIMARY_FOREST_PROTECTION_SP	POLICY_OBJECTIVE_PRIMARY
31	POLICY SCENARIO PARAMETERS	SWITCH_POLICIES_LAND_PROTECTION_SP	YEAR_INITIAL_LAND_PROTECTION_SP	YEAR_FINAL_LAND_PROTECTION_SP	OBJECTIVE_LAND_P
32	REGIONS_IJ_UNIT	DMNL	YEAR	YEAR	DMNL
33	EU27	0	2015	2030	1
34	UK	0	2015	2030	1
35	CHINA	0	2015	2030	1
36	EASOC	0	2015	2030	1
37	INDIA	0	2015	2030	1
38	LATAM	0	2015	2030	1
39	RUSSIA	0	2015	2030	1
40	USMCA	0	2015	2030	1
41	ROW	0	2015	2030	1

6.8 Additional configurations

```
python run.py -h
```

```
usage: usage: pywilliam [-h] [-v] [-n FILE] [-e FILE] [-p] [-c SHEET] [-x FILE] [-b] [-s] [-F VALUE] [-T VALUE] [-S VALUE]
                        [--missing-values {warning,raise,ignore,keep}]
                        [variable=new_value ...] [variable:initial_value ...]

WILLIAM model

positional arguments:
  variable=new_value    redefine the value of variable with new value.variable must be a model component, new_value can be
                        a float or a a list of two list
  variable:initial_value
                        redefine the initial value of variable.variable must be a model stateful element, initial_value
                        must be a float

options:
  -h, --help            show this help message and exit
  -v, --version          show program's version number and exit
  -n FILE, --fname FILE
                        name of the results file, default is results_{scenario sheet}_{initial time}_{final time}_{time
                        step}.csv
  -e FILE, --externals FILE
                        path to the netCDF file where the external objects are stored
  -p, --plot            opens the plot gui after simulation
  -c SHEET, --scen SHEET
                        scenario file path
  -x FILE, --export FILE
                        export stateful objects states to a pickle at the end of the simulation
  -b, --headless        headless mode (only CLI, no GUI)
  -s, --silent          silent mode. No user input will be required during execution. Usefulwhen running batch simulations

model arguments:
  Modify model control variables.

  -F VALUE, --final-time VALUE
                        modify final year of the simulation, default is 2050.0
  -T VALUE, --time-step VALUE
                        modify time step (in years) of the simulation, default is 0.25
  -S VALUE, --saveper VALUE
                        modify time step (in years) of the output, default is 1.0 year

warning and errors arguments:
  Modify warning and errors management.

  --missing-values {warning,raise,ignore,keep}
                        exception with missing values, 'warning' (default) shows a warning message and interpolates the
                        values, 'raise' raises an error, 'ignore' interpolates the values without showing anything, 'keep'
                        keeps the missing values
```

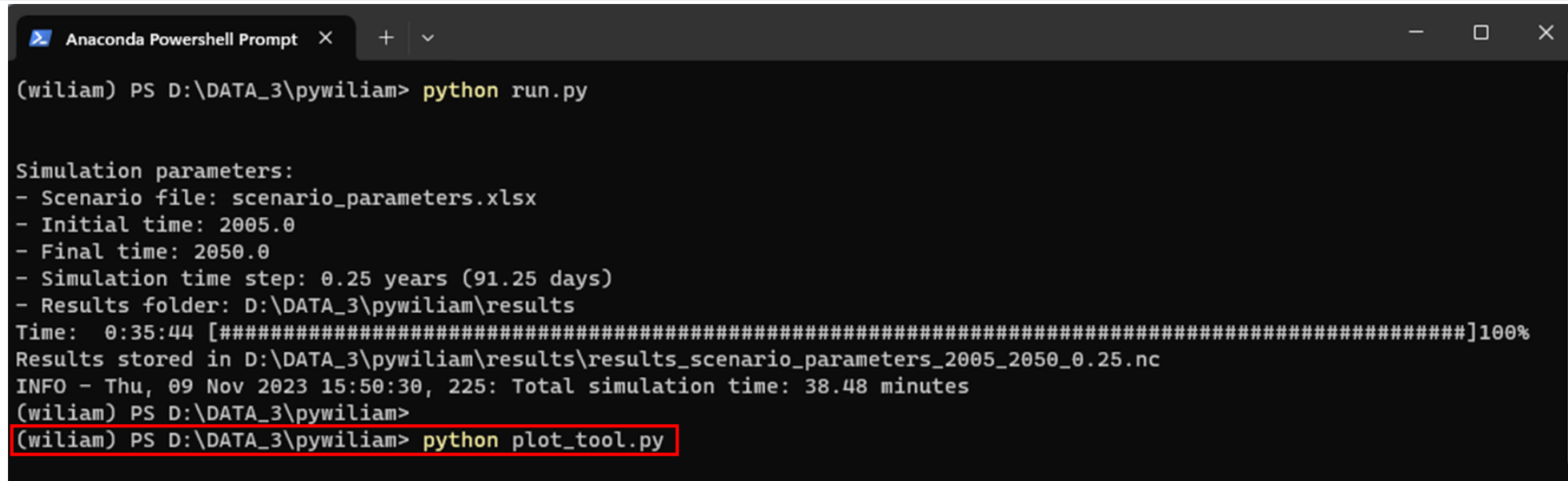
6.9 Plotting the results

If the `-p` argument is passed to the CLI, a graphical user interface will load automatically after the simulation ends, which lets the users plot the results:

```
python run.py -h
```

Alternatively, the plot tool may be launch in standalone mode, by running:

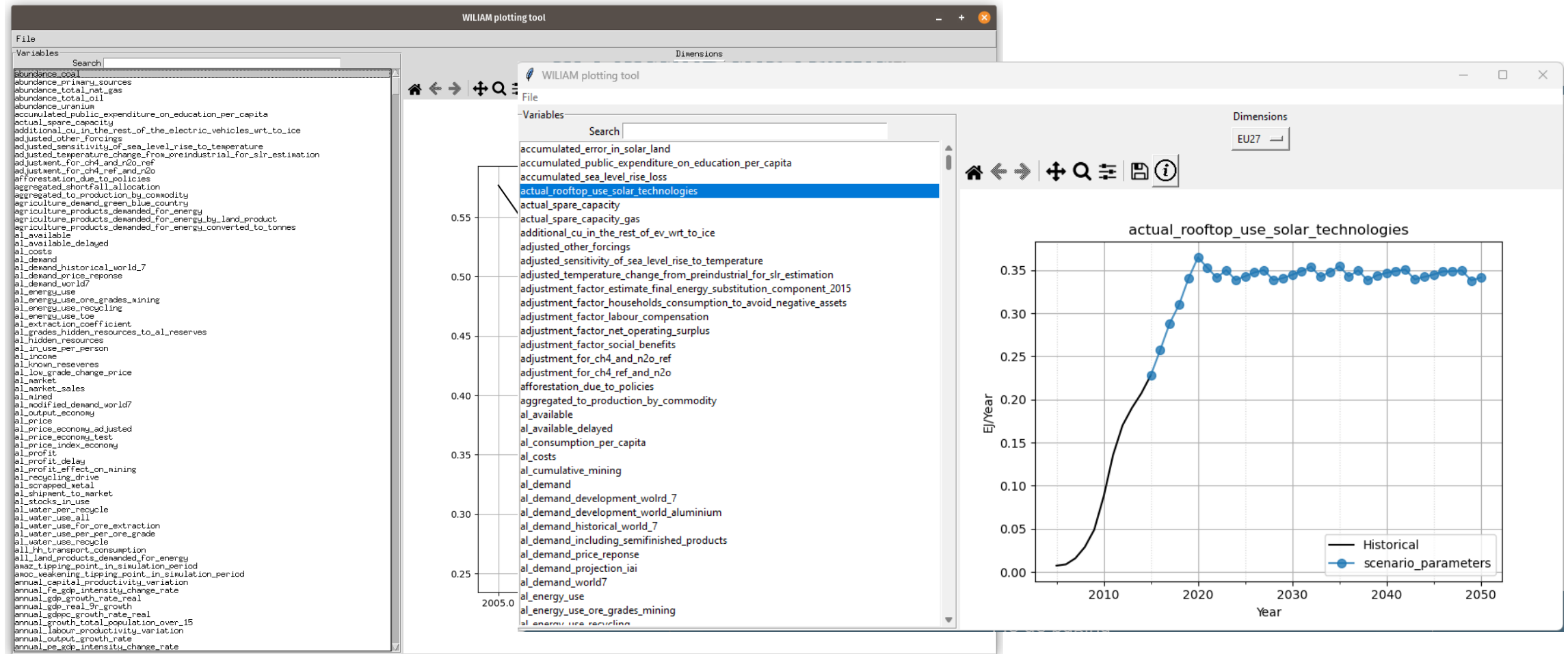
```
python plot_tool.py
```



```
Anaconda Powershell Prompt
(wiliam) PS D:\DATA_3\pywiliam> python run.py

Simulation parameters:
- Scenario file: scenario_parameters.xlsx
- Initial time: 2005.0
- Final time: 2050.0
- Simulation time step: 0.25 years (91.25 days)
- Results folder: D:\DATA_3\pywiliam\results
Time: 0:35:44 [#####]100%
Results stored in D:\DATA_3\pywiliam\results\results_scenario_parameters_2005_2050_0.25.nc
INFO - Thu, 09 Nov 2023 15:50:30, 225: Total simulation time: 38.48 minutes
(wiliam) PS D:\DATA_3\pywiliam>
(wiliam) PS D:\DATA_3\pywiliam> python plot_tool.py
```

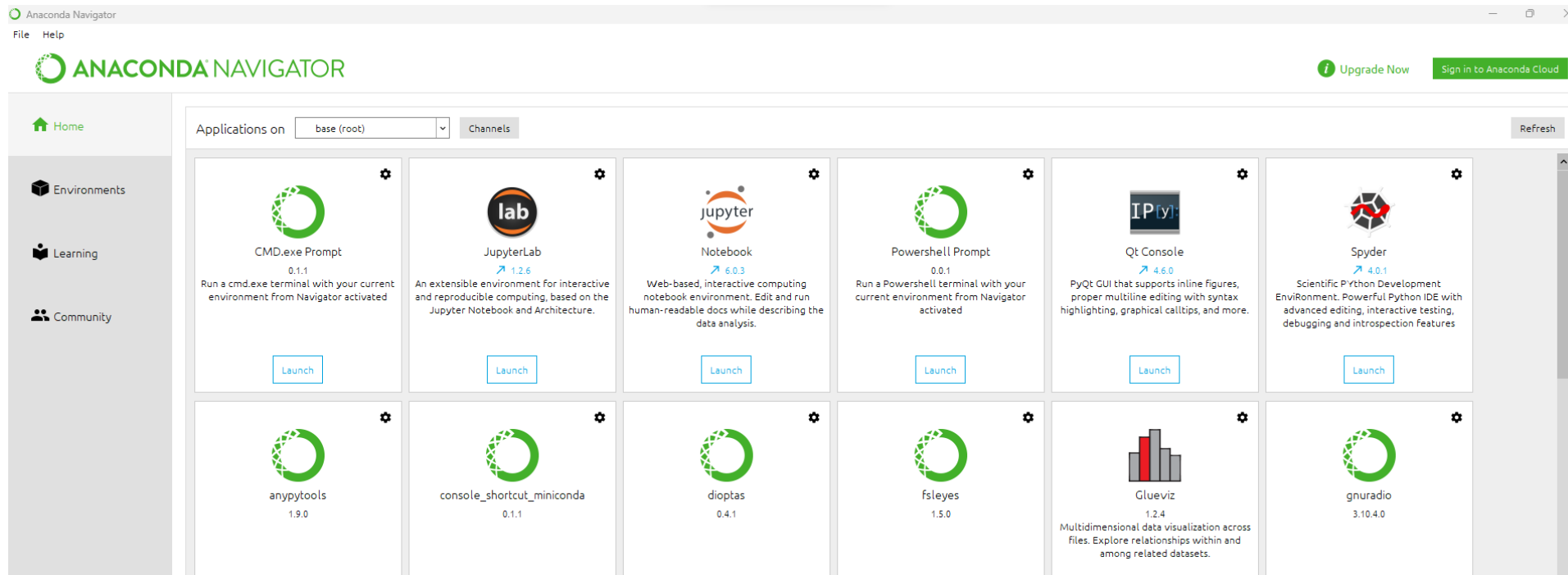
6.9 Plotting the results



6.10 Additional tools (exporting from netCDF to csv)

Use Jupyter Notebook

- Launch Anaconda
- Create a New Environment
- Install libraries and open the environment



6.10 Additional tools (exporting from netCDF to csv)



Managing and exporting datasets from .nc files

By following this tutorial, you should be able to open a nc file, visualize the parameters and variables stored on it and export some of them to a tab or csv file.

```
In [ ]: import pysd
        from pysd.tools.ncfiles import NCFFile
        import pandas as pd
        import warnings

        pd.set_option('display.max_row', None)
        warnings.filterwarnings('ignore')
```

WARNING: To use this script, you must be using PySD 3.8 or newer, and the netCDF4 and dask Python libraries must be installed in your environment.

```
In [ ]: pysd.__version__
```

```
In [ ]: # folder where the nc file is located
        # no need to change this one, unless you moved the nc file to a different folder
        results_folder = "../results/"
```

Configure exports paths

```
In [ ]: # Put here the name of the nc file you want to export data from
        simulation_results = "results_scenario_parameters_2005_2010_0.25.nc"
```

The exported data will be stored in the results folder in a tab file with the same name (i.e. results_scenario_parameters_2005_2010_0.25_export.tab)

6.10 Additional tools (exporting from netCDF to csv)

Managing and exporting datasets from .nc files

By following this tutorial, you should be able to open a nc file, visualize the parameters and variables stored on it and export some of them to a tab or csv file.

```
In [ ]: import pysd
        from pysd.tools.ncfiles import NCFile
        import pandas as pd
        import warnings

        pd.set_option('display.max_row', None)
        warnings.filterwarnings('ignore')
```

WARNING: To use this script, you must be using PySD 3.8 or newer, and the netCDF4 and dask Python libraries must be installed in your environment.

```
In [ ]: pysd.__version__
```

```
In [ ]: # folder where the nc file is located
        # no need to change this one, unless you moved the nc file to a different folder
        results_folder = "../results/"
```

Configure exports paths

```
In [ ]: # Put here the name of the nc file you want to export data from
        simulation_results = "results_scenario_parameters_2005_2010_0.25.nc"
```

The exported data will be stored in the results folder in a tab file with the same name (i.e. results_scenario_parameters_2005_2010_0.25_export.tab)

Define variables to export

Define here which model parameters or variables you wish to export.

You can choose any model variable (including inputs)

```
In [ ]: # Modify this list
        variables_to_export = ["total_water_demand_region", "total_mineral_supply"]
```

