

Off-Grid Systems Modelling with MicroGridsPy

Hands-on 1

GitHub repository for source code:

[SESAM-Polimi/MicroGridsPy-SESAM: MicroGridsPy - SESAM-PoliMi \(github.com\)](https://github.com/SESAM-Polimi/MicroGridsPy-SESAM)

MicroGridsPy is an open-source project, currently under active development, check out the detailed online documentation for usage guidance and updates:

<https://microgridspy-documentation.readthedocs.io/en/latest/>

Learning outcomes

By the end of this exercise, you will learn how to:

- 1) Install and manage Anaconda
- 2) Install the environment for MicroGridsPy
- 3) Install Gurobi Solver (optional: license required)
- 4) Launch the MicroGridsPy User Interface

Install Anaconda

Anaconda's package manager, known as conda, is a powerful tool for managing packages, dependencies, and environments in the Anaconda ecosystem. It is designed to handle various types of packages, including Python and R, and works across multiple platforms (Windows, macOS, and Linux). Here are key features and functions of conda:

- **Package Management:** conda allows users to install, update, and remove packages. It hosts a large repository of scientific packages and ensures that the installed packages are compatible with each other.
- **Environment Management:** One of the most significant features of conda is its ability to create isolated environments. These environments can have different versions of Python and packages, making it easier to manage dependencies for different projects without conflicts.
- **Cross-Platform Compatibility:** conda works on various operating systems, providing a consistent experience across platforms.
- **Large Repository:** Anaconda comes with a vast repository of pre-built packages specifically tailored for scientific computing, data science, and machine learning applications. This saves time and effort in compiling and configuring these complex packages.
- **Open Source and Community-Driven:** conda is open-source and benefits from community contributions, which means a wide range of packages and continuous updates.
- **Ease of Use:** While it offers a command-line interface, conda is designed to be user-friendly, making package and environment management accessible to users regardless of their expertise in command-line tools.

Download the free Conda package manager from the Anaconda distribution website by clicking this link:

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

You can find also a complete list of installation files and old versions here:

<https://repo.anaconda.com/archive/>

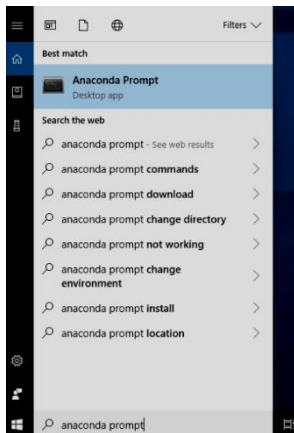
Review the system requirements listed below before installing **Anaconda Distribution**. If you don't want the hundreds of packages included with Anaconda, install **Miniconda**, a mini version of Anaconda that includes just conda, its dependencies, and Python, using this link:

<https://docs.conda.io/projects/miniconda/en/latest/>

Getting started with conda

Conda is a powerful package manager and environment manager that you use with command line commands at the Anaconda Prompt for Windows, or in a terminal window for macOS or Linux.

Windows



From the Start menu, search for and open "**Anaconda Prompt**"

The Anaconda Prompt is a command-line interface tool that comes with Anaconda. It is particularly tailored for managing Python environments and packages within the Anaconda ecosystem and it enables users to execute Python scripts, install new packages via conda (Anaconda's package manager), create isolated environments to avoid dependency conflicts, and update the Anaconda distribution itself. On Windows, all commands below are typed into the Anaconda Prompt window.

MacOS

Open Launchpad, then click the terminal icon. On macOS, all commands below are typed into the terminal window.

Linux

Open a terminal window. On Linux, all commands below are typed into the terminal window.

Managing conda

Verify that conda is installed and running on your system by typing:

```
conda --version
```

Conda displays the number of the version that you have installed. You do not need to navigate to the Anaconda directory.

Note

If you get an error message, make sure you closed and re-opened the terminal window after installing, or do it now. Then verify that you are logged into the same user account that you used to install Anaconda or Miniconda.

Update conda to the current version. Type the following:

```
conda update conda
```

Conda compares versions and then displays what is available to install. If a newer version of conda is available, type y to update:

```
Proceed ([y]/n)? y
```

We recommend that you always keep conda updated to the latest version.

Getting started with Anaconda Navigator

Anaconda Navigator starts by default when Anaconda Distribution is first installed. Anaconda Navigator is a graphical user interface (GUI) tool included with the Anaconda distribution. It serves as an **alternative to the Anaconda Prompt**, offering a more **user-friendly way** to manage the various aspects of the Anaconda environment without needing to use command-line instructions.

Windows

From the Start menu, search for “Anaconda Navigator” and click to open.

MacOS

Open Launchpad, then click the Anaconda-Navigator icon.

Linux

1. Open a terminal window.
2. Open Navigator by using the following command:

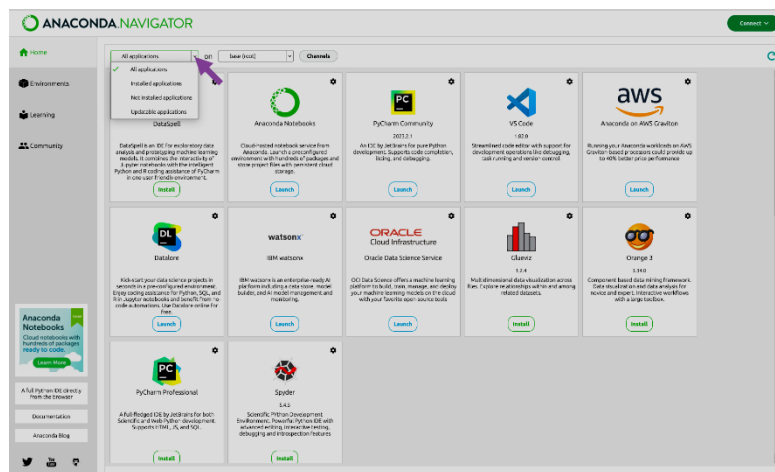
```
anaconda-navigator
```

Managing Navigator

Through the Anaconda Navigator, users can easily manage their Python environments, install and update packages, and launch applications included in the Anaconda distribution, like Jupyter Notebooks, Spyder, RStudio, and others. It's particularly advantageous for those

who prefer a visual interface over command-line operations. The Navigator allows for easy access to different tools and simplifies the process of setting up and maintaining Python environments for various projects. This is especially beneficial for beginners or those who prefer a more intuitive, point-and-click experience in managing their Python development setup.

By default, all application tiles available to launch or install within Navigator are displayed on the Home page. Filter the application tiles with the applications dropdown menu.



Install MicroGridsPy Environment

In conda, an environment is an **isolated space** that allows users to maintain different versions of Python and various packages without interference. Each environment can have its **own specific set of packages and Python versions**, independent of others. This is particularly useful in managing dependencies and avoiding conflicts when working on multiple projects with differing requirements. By using environments, developers and data scientists can ensure consistency and reproducibility of their work across various setups and collaborations.

Create the environment from Anaconda Prompt

When you begin using conda, you already have a default environment named base. To create a modelling environment that already contains everything needed to run MicroGridsPy, it's required to download the environment **YML file** from the following GitHub online repository:

<https://github.com/SESAM-Polimi/MicroGridsPy-SESAM/tree/Environments>

1. Place the YML file (e.g. mgpy_win.yml) in "C:/Users/youruser".

2. Open the Anaconda Prompt
3. Type the following command in the Anaconda Prompt terminal:

```
conda env create -f mgpy_win.yml
```

4. Activate the environment by:

```
conda activate mgpy
```

Note

conda activate only works on conda 4.6 and later versions.

Create the environment from Anaconda Navigator

1. **Launch Anaconda Navigator:** Open Anaconda Navigator on your computer.
2. **Navigate to Environments:** On the left-hand side of the Navigator window, click on the "Environments" tab.
3. **Import Environment:** Look for the "Import" button at the bottom of the environment list and Click on "Import".
4. **Select the YML File:** In the import dialog, you will see a field to choose the YML file. Click on the folder icon next to the text box to browse your computer and select the .yml file you want to use.
5. **Name Your Environment:** Below the file selection, there's a field to name your new environment. Choose a meaningful name for the environment you're creating (e.g. mgpy)
6. **Create Environment:** After selecting the file and naming your environment, click the "Import" button. Anaconda Navigator will start creating the environment using the specifications in the YML file. This process may take some time, depending on the number of packages to be installed and your internet connection speed.
7. **Activation:** Once the environment is created, you can activate it by selecting it from the list in the "Environments" tab.

To use this environment, you can open tools like Jupyter Notebook, Spyder, or a terminal from within the Navigator while the environment is active.

Install Gurobi Solver

Gurobi is a **leading mathematical optimization solver** renowned for its efficiency in solving linear, mixed-integer, and quadratic programming problems. Developed by industry experts, it is widely adopted in both academic and commercial spheres for a variety of applications, ranging from supply chain management to financial planning. Gurobi stands out for its high-performance capabilities, user-friendly interfaces compatible with multiple programming languages, and continuous updates incorporating the latest algorithmic advancements. While it offers **free academic licenses**, its **commercial use is governed by a comprehensive licensing model**, making it an essential tool for researchers and professionals alike in optimizing complex decision-making processes.

The free-to-use solver built-in within the MicroGridsPy environment is named GLPK. GLPK is an open-source solver for Linear Programming (LP) and Mixed Integer Programming (MIP). It's a suitable option for smaller to medium-sized problems and offers a free alternative to commercial solvers. While GLPK is a capable solver for many optimization problems, it may have longer operational times compared to commercial solvers like Gurobi, especially for large or complex problems. The difference can often be substantial, potentially ranging from several times to orders of magnitude faster, depending on the specifics of the problem even if it's important to note that these are general observations, and actual performance will vary with each unique problem. It is advisable to consider this factor when choosing a solver for time-sensitive or large-scale applications.

Obtain a Gurobi License

Before installing Gurobi, you need to obtain a license. Gurobi offers different types of licenses, including academic licenses which are free for academic purposes. Visit the Gurobi website and register for a license and follow their instructions to set up your license:

[The Leader in Decision Intelligence Technology - Gurobi Optimization](#)

Installing Gurobi using Anaconda Prompt

1. **Open the Anaconda Prompt** (or your terminal in Linux/Mac)
2. **Activate the mgpy environment**
3. **Install the Gurobi** package by running:

```
conda install gurobi
```

4. Once Gurobi is installed, you need to **activate your license**. This usually involves running a command provided by Gurobi in your Anaconda Prompt or terminal. If you're using an academic license, you typically run:

```
grbgetkey your-license-key
```

Refer to the Gurobi website for more information about license installation.

Installing Gurobi using Anaconda Navigator

1. **Launch Anaconda Navigator** on your computer
2. In Anaconda Navigator, go to the "**Environments**" tab. Click on "**Channels**" and then on "**Add**". Type gurobi and click on the "**Update channels**" button. This step ensures that the Gurobi package can be found in the Anaconda repository.
3. Click on the "**Home**" tab, then select the MicroGridsPy environment you created from the drop-down menu.
4. In the search bar, type "**Gurobi**". When Gurobi appears in the list, select it and click on "**Apply**" to install.
5. Follow the instructions provided by Gurobi for activating your license. This typically involves running a command in your terminal or Anaconda Prompt.

Launch the MicroGridsPy User Interface

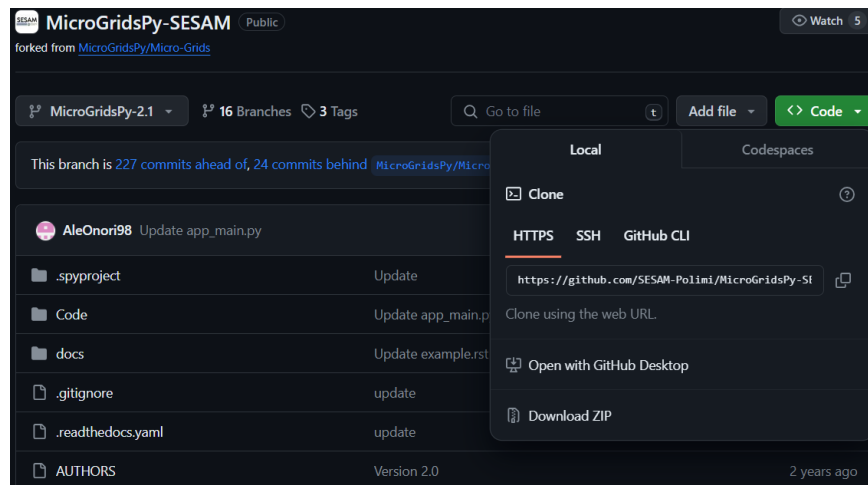
The graphical user interface (GUI) application provides a user-friendly way to define and input data for MicroGridsPy.

Download the MicroGridsPy folder

To actually use MicroGridsPy it's needed first of all to download the folder of the model from GitHub. Open your web browser and go to the **SESAM GitHub repository** at this link:

[SESAM-Polimi/MicroGridsPy-SESAM: MicroGridsPy - SESAM-PoliMi \(github.com\)](https://github.com/SESAM-Polimi/MicroGridsPy-SESAM)

Click the green "**Code**" button on the right side and select "**Download ZIP**" to download the entire folder as a ZIP file. Unzip and place the folder wherever it's easily accessible in your system.



Launch Spyder using Anaconda Prompt

Spyder is an **open-source integrated development environment (IDE)** primarily designed for scientific and data-driven computing in the Python programming language. It provides a user-friendly and interactive environment for tasks such as data analysis, scientific research, machine learning, and numerical computing. Spyder offers features like a code editor, IPython console integration, variable explorer, and a comprehensive set of tools for data visualization and exploration, making it a popular choice among data scientists and researchers for Python-based projects.

1. **Open the Anaconda Prompt**
2. **Activate the mgpy environment**
3. Type the following command to **open the spyder interface**:

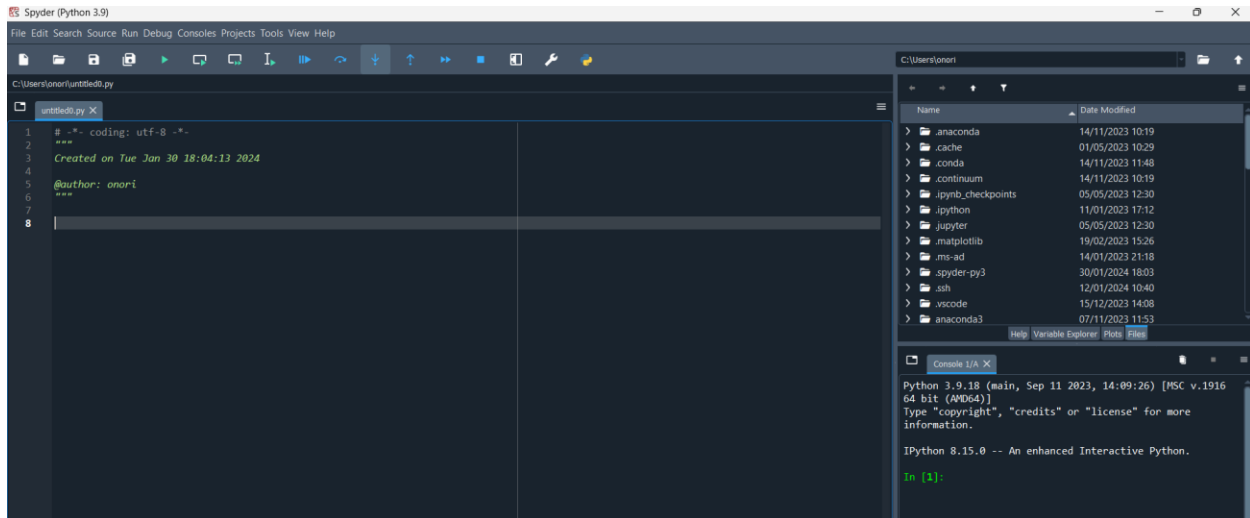
```
conda activate mgpy
spyder
```

Launch Spyder using Anaconda Navigator

1. **Launch Anaconda Navigator:** Start by opening Anaconda Navigator on your computer. You can typically find it in your list of installed applications or use the Anaconda Navigator shortcut if you have one.
2. **Activate the mgpy environment:** Activate the mgpy environment from the "Environments" tab in Anaconda Navigator.
3. **Open Spyder:** In Anaconda Navigator, navigate to the "Home" tab. You will see a list of available applications and tools. Look for "Spyder" in the list. Click on the "Launch" button next to Spyder to open the Spyder IDE.

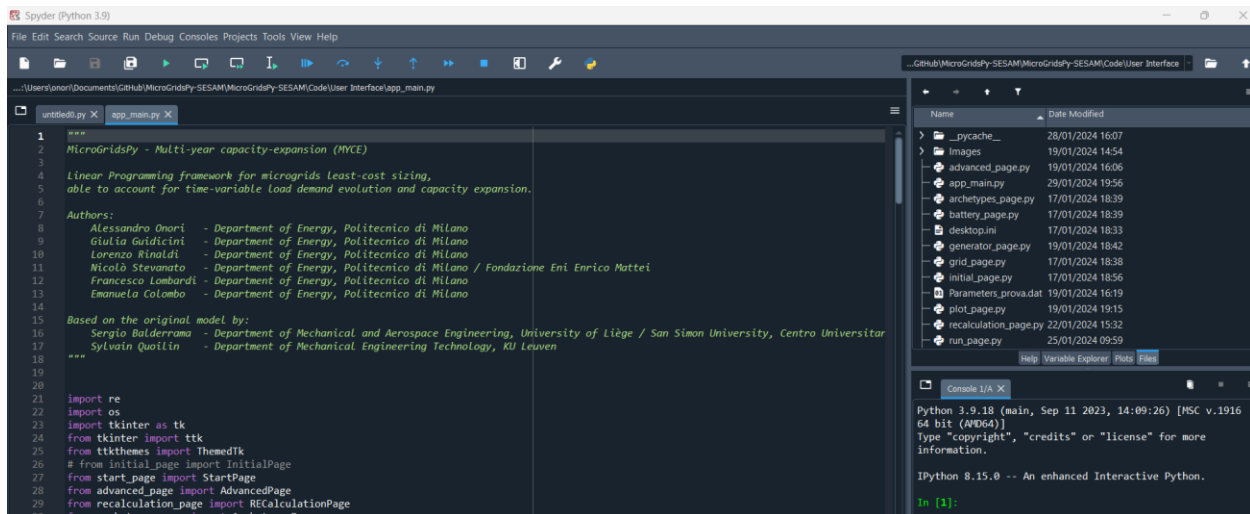
Launch the Interface within Spyder

After launching the Spyder IDE, you should see the following page:



If you have visualization problem, you can always set the default layout from the "View" button.

4. **Locate the MicroGridsPy Working Folder:** In Spyder, go to the "File" menu at the top left corner of the interface. Select "Open..." to open a file or folder.
5. **Navigate to the MicroGridsPy Folder:** Use the file browser to navigate to the location where you have the MicroGridsPy project folder stored on your computer.
6. **Open the MicroGridsPy Folder:** Double-click on the MicroGridsPy project folder to open it within the Spyder interface. You should now see the contents of the project folder displayed in the Spyder File Explorer.
7. **Locate app_main.py:** In the File Explorer panel on the left-hand side of the Spyder interface, navigate to the "Code/User Interface" folder within the MicroGridsPy project folder. Look for the app_main.py file within this folder.
8. **Open app_main.py:** Double-click on the app_main.py file to open it in the Spyder code editor.
9. **Run app_main.py:** With app_main.py open in the code editor, you can run it by pressing F5 or using the "Run" button in Spyder's toolbar. Alternatively, you can right-click in the code editor and select "Run File" from the context menu.

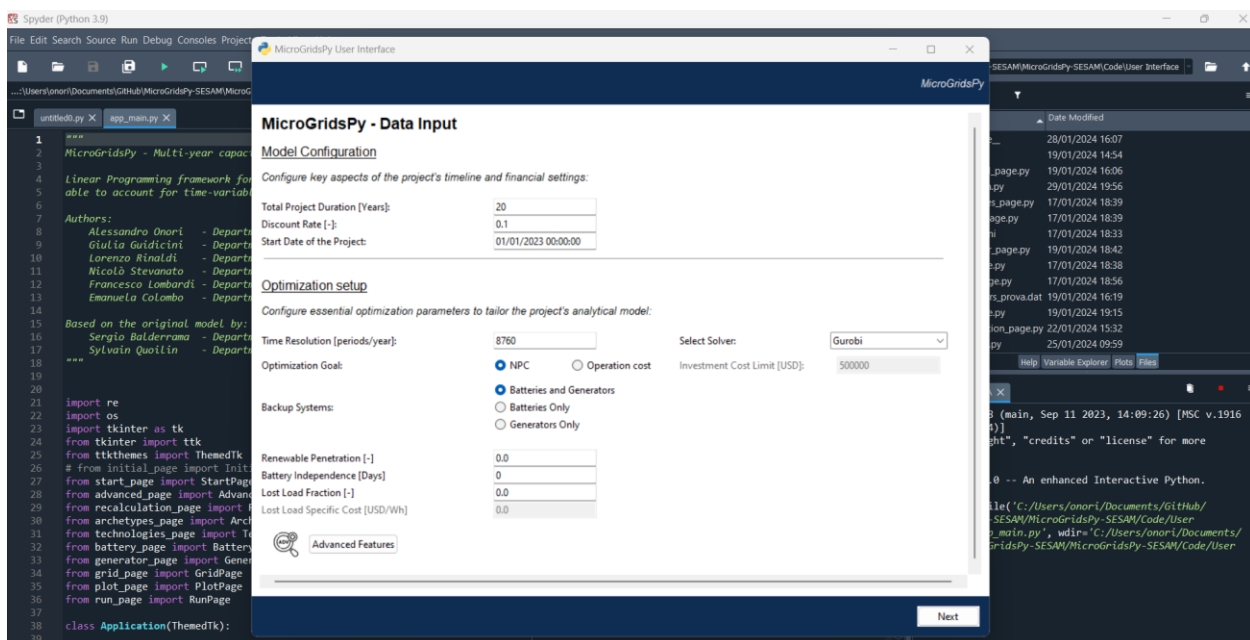


```

1  """
2  MicroGridsPy - Multi-year capacity-expansion (MYCE)
3
4  Linear Programming framework for microgrids least-cost sizing,
5  able to account for time-variable load demand evolution and capacity expansion.
6
7  Authors:
8      Alessandro Onori - Department of Energy, Politecnico di Milano
9      Giulia Guidicini - Department of Energy, Politecnico di Milano
10     Lorenzo Rinaldi - Department of Energy, Politecnico di Milano
11     Nicolò Stevanato - Department of Energy, Politecnico di Milano / Fondazione Eni Enrico Mattei
12     Francesco Lombardi - Department of Energy, Politecnico di Milano
13     Emanuela Colombo - Department of Energy, Politecnico di Milano
14
15 Based on the original model by:
16     Sergio Balderrama - Department of Mechanical and Aerospace Engineering, University of Liège / San Simon University, Centro Universitario
17     Sylvain Quoilin - Department of Mechanical Engineering Technology, KU Leuven
18
19 """
20
21 import re
22 import os
23 import tkinter as tk
24 from tkinter import ttk
25 from ttkthemes import ThemedTk
26 # from initial_page import InitialPage
27 from start_page import StartPage
28 from advanced_page import AdvancedPage
29 from recalculation_page import RECalculationPage
30 from archetypes_page import ArchetypesPage

```

After running app_main.py, the interface of MicroGridsPy should launch within Spyder.



Well done: you can now interact with the application as needed for your specific use case!