

PyPSA Demystified

Core concepts, best practices and new features

Priyesh Gosai

Director – Energy Solutions | PAS Automation

26 November 2025

pypsa-introduction

Background

- From South Africa, now based in Edinburgh, Scotland.
- Mechanical engineer with 15 years in the energy sector.
- Turbine plant engineer at Eskom.
- Researcher manager, lecturer at the University of Cape Town.
- Independent consultant developing numerical simulation tools
 - Thermal fluid modelling.
 - Energy systems modelling.
 - Dispatch optimisation for power markets.
 - Co-optimisation of power systems with large shares of hydropower.
 - Deliver training courses to introduce working professionals into the area of energy system modelling.
 - Develop and implement custom workflows using opensource tools.

Founder of:

**Innovate
for Impact**

Director at:

PAS
The Human Reliability Company

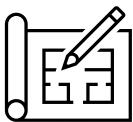
Clients:



energynautics
solutions for sustainable development



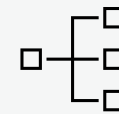
Objectives of this training



Walk participants through the tools, environment, and dependencies required to model energy systems with PyPSA.



Explain how a PyPSA model is structured and how its core components operate.



Showcase the latest improvements and newly added capabilities in PyPSA.



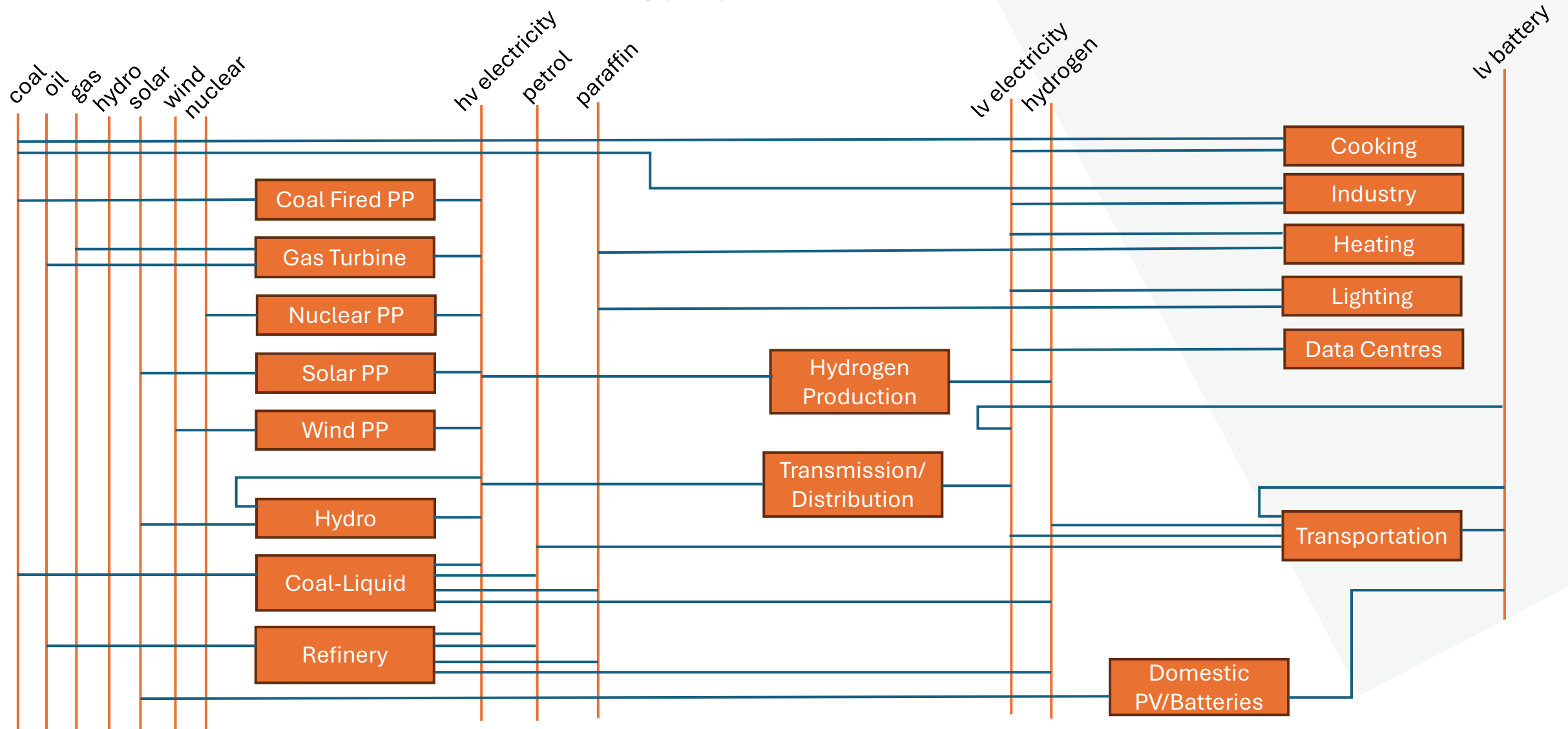
Allow attendees to explore and manipulate a working PyPSA network themselves.



Guide users through building, executing, and interpreting a basic PyPSA model from scratch.



Evolution of South African energy system



What is the optimal location and capacity for new renewable energy installations considering geographic and grid constraints?

How sensitive is the energy system to changes in technology costs, policy environments, and global market conditions?

How can pumped storage and other energy storage technologies complement intermittent renewable generation?

What policy interventions could accelerate the transition to a more sustainable energy system?

What is the optimal mix of renewable energy sources (wind, solar, offshore wind, etc.) to ensure reliable electricity supply?

What keeps energy leaders up at night?

What is the optimal pathway to reduce carbon emissions while ensuring energy security?

How can regional energy integration improve overall system resilience?

How can battery storage and electric vehicles be strategically integrated to provide grid flexibility?

What are the potential benefits and challenges of developing a hydrogen economy?

What fossil fuel capacity is required to maintain national energy security under increasing renewable penetration?

What kinds of questions related to the energy sector keep you up at night?



Use the chat feature to post your questions.



PyPSA to co-optimize expansion planning and operation

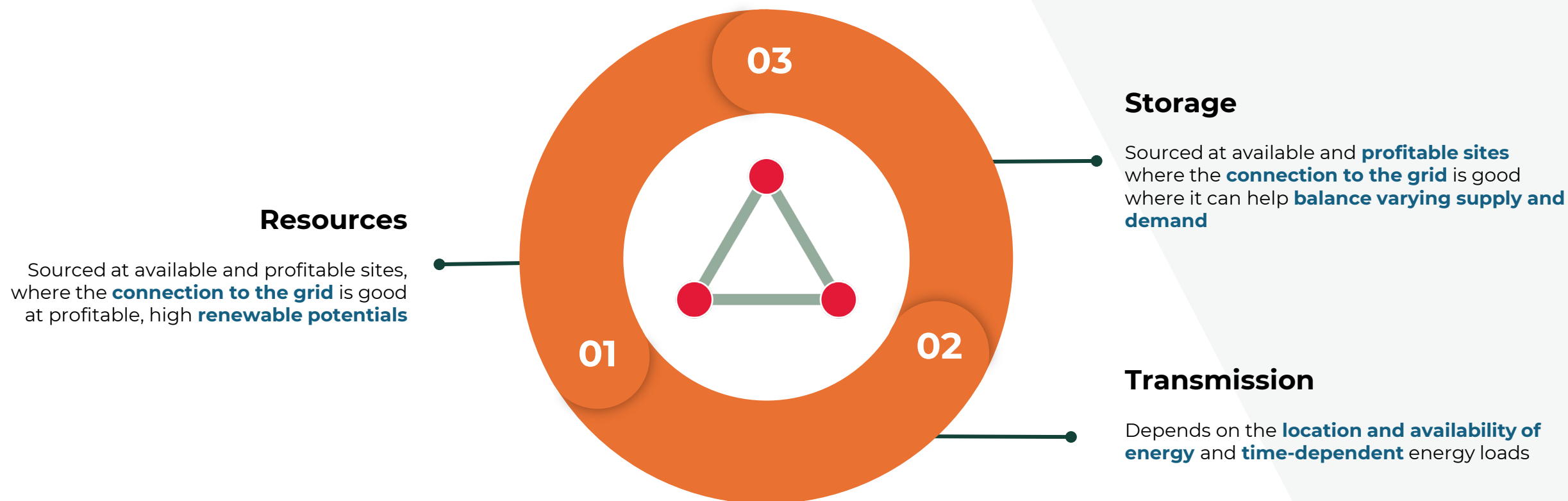
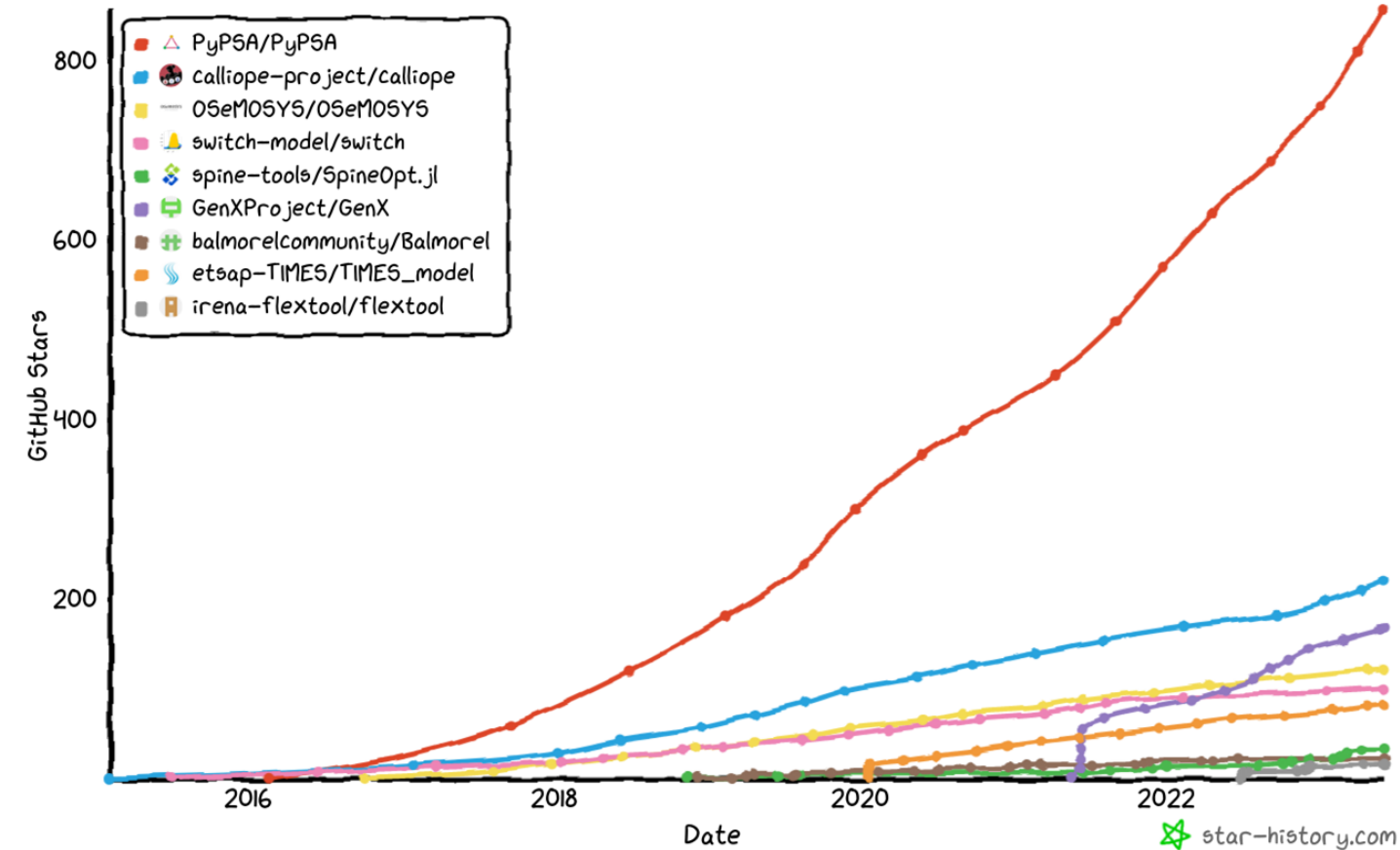


Image credit: Martha Frystaki (OET)



Adoption of PyPSA



Major advantages of PyPSA:

- **co-optimization & sector coupling**
- **industry support**
- **global model coverage**
- excellent research **contributions**
- detailed **documentation**
- **fully open source**: data & model

Packages needed



High-level programming language

Widely used in data science and modelling

Easy to learn, with rich libraries



Platform for version control using Git

Enables collaboration and code sharing

Hosts projects, issues, and documentation



Python distribution with data science tools

Includes Conda for environment management

Useful for reproducible, isolated setups



IDE Options



Runs Python code in the browser — no setup needed

Free access to GPUs/TPUs for basic workloads

Easy to share notebooks via link

Limited local file access and custom environment control



Visual Studio Code

Powerful local code editor with rich extensions

Full control over Python environments (e.g., via Conda/venv)

Great for large projects and debugging

Requires local setup and more resources

And many more ...



Python Toolboxes



Data wrangling toolkit



Numerical array operations



Static data visualization



Workflow automation engine

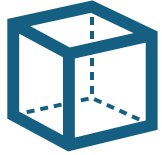
Libraries maintained
in the PyPSA
ecosystem



atlite



Network Object



Network framework containing integrated tools for optimisation, power flow modelling, data management, statistical analysis, and visual network representation.

Input/output



Handles CSV, netCDF, Excel, and HDF5 file formats, enabling network creation within Python scripts and seamless file operations across local and cloud-based storage systems.

API



Simple yet powerful API interface with clear, user-friendly commands that maximise flexibility while maintaining ease of use and accessibility.

Components



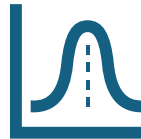
Energy system components feature structured attributes with specified data types, default values, and detailed descriptions.

Mapping and Plotting



Interactive and static plotting capabilities with full customisation options. Users can generate various charts and maps for any network metric.

Statistics Library



Summarises and visualises key data and results through postprocessing, streamlining data analysis and presentation for user accessibility.

Solving



The solver handles Economic Dispatch, optimal power flow variants, Capacity Expansion Planning, Stochastic Optimization, and alternative scenario generation.

Options



Full customisation capabilities allow users to configure all aspects of the model, ensuring the framework can be tailored to any specific application or need.



PyPSA Model Structure

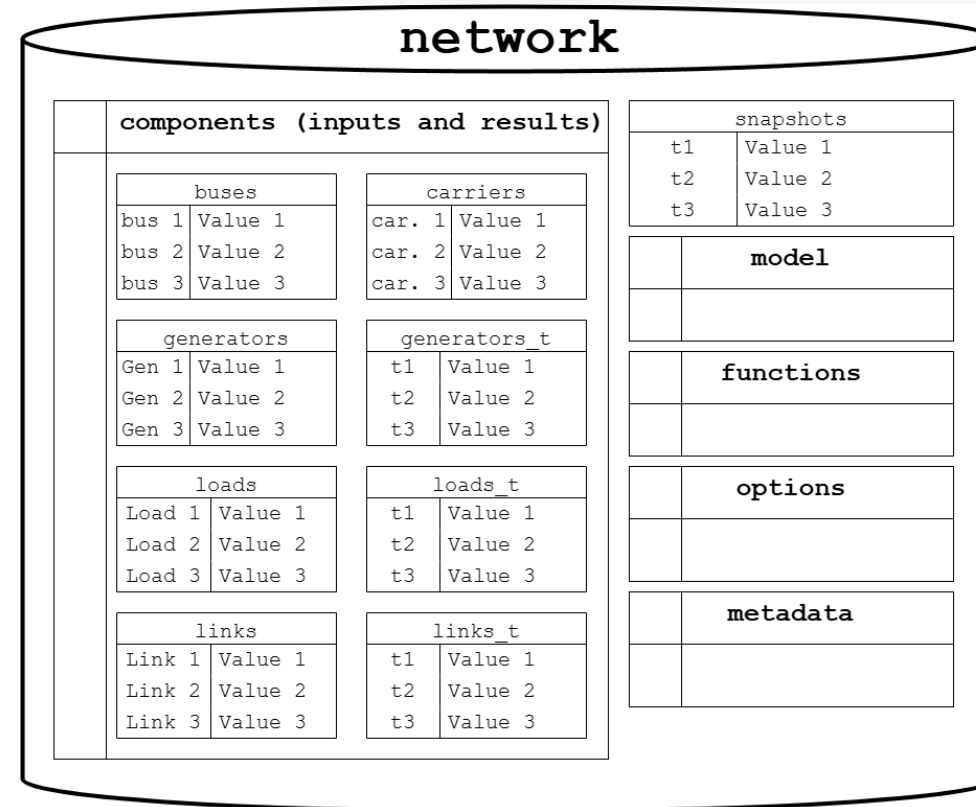
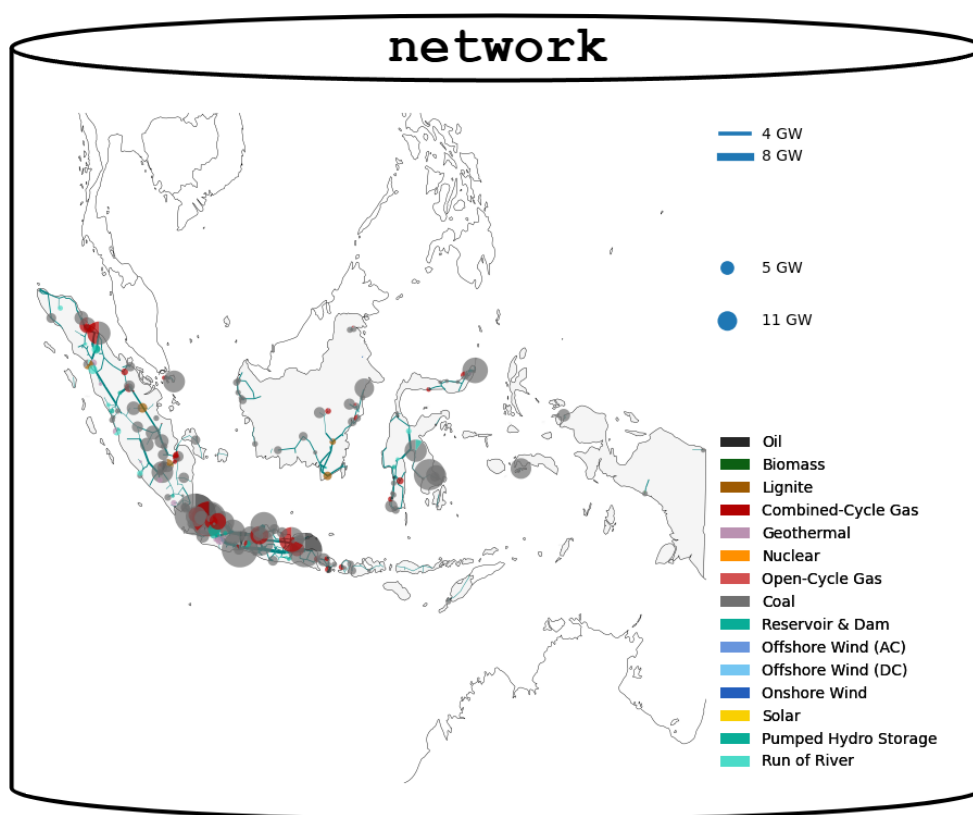
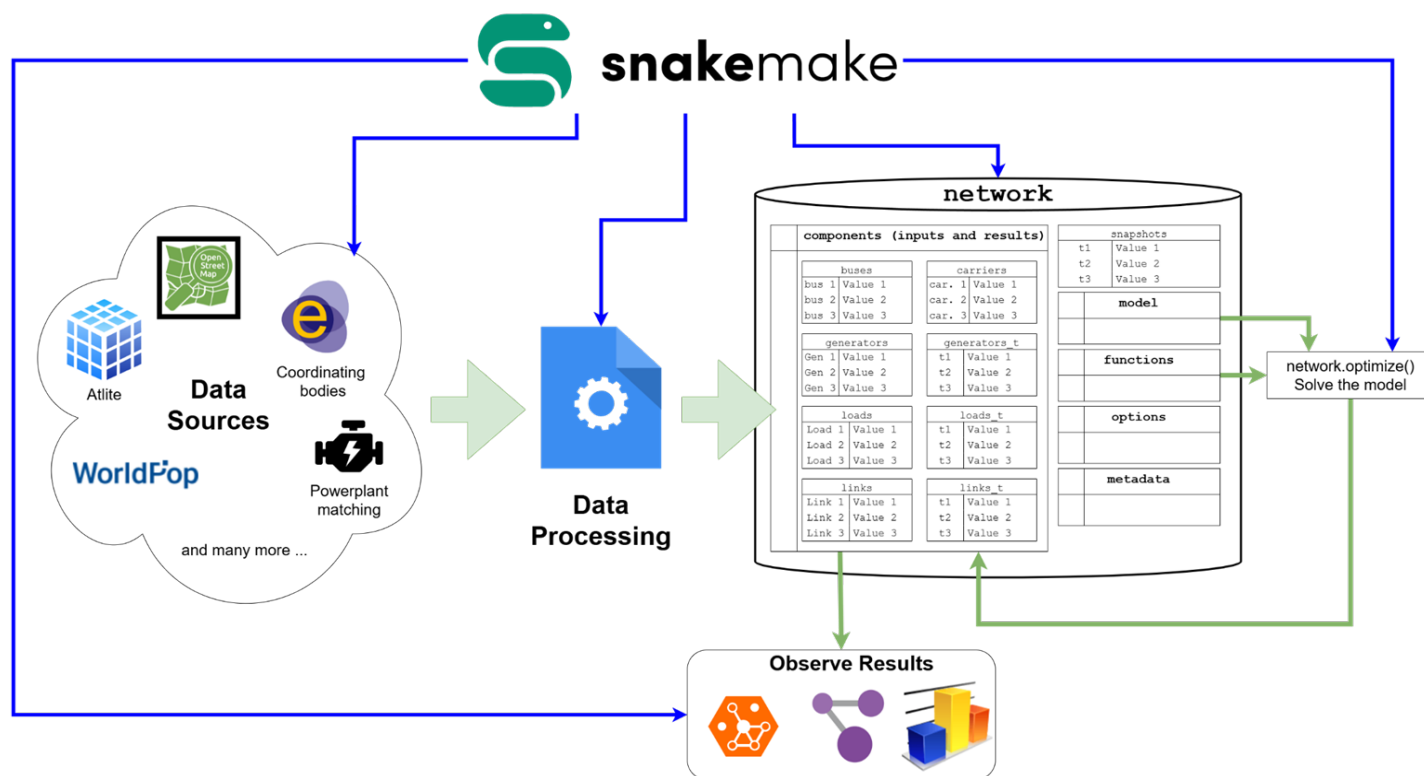


Image credit: Arizeo Salac (Univ. of Pisa)



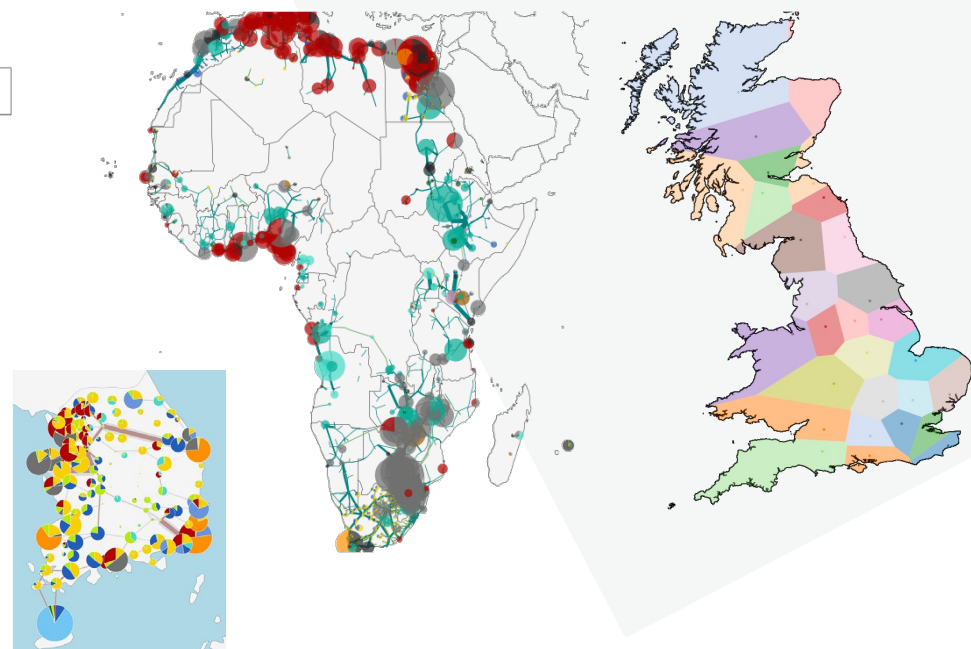
Typical PyPSA Workflow



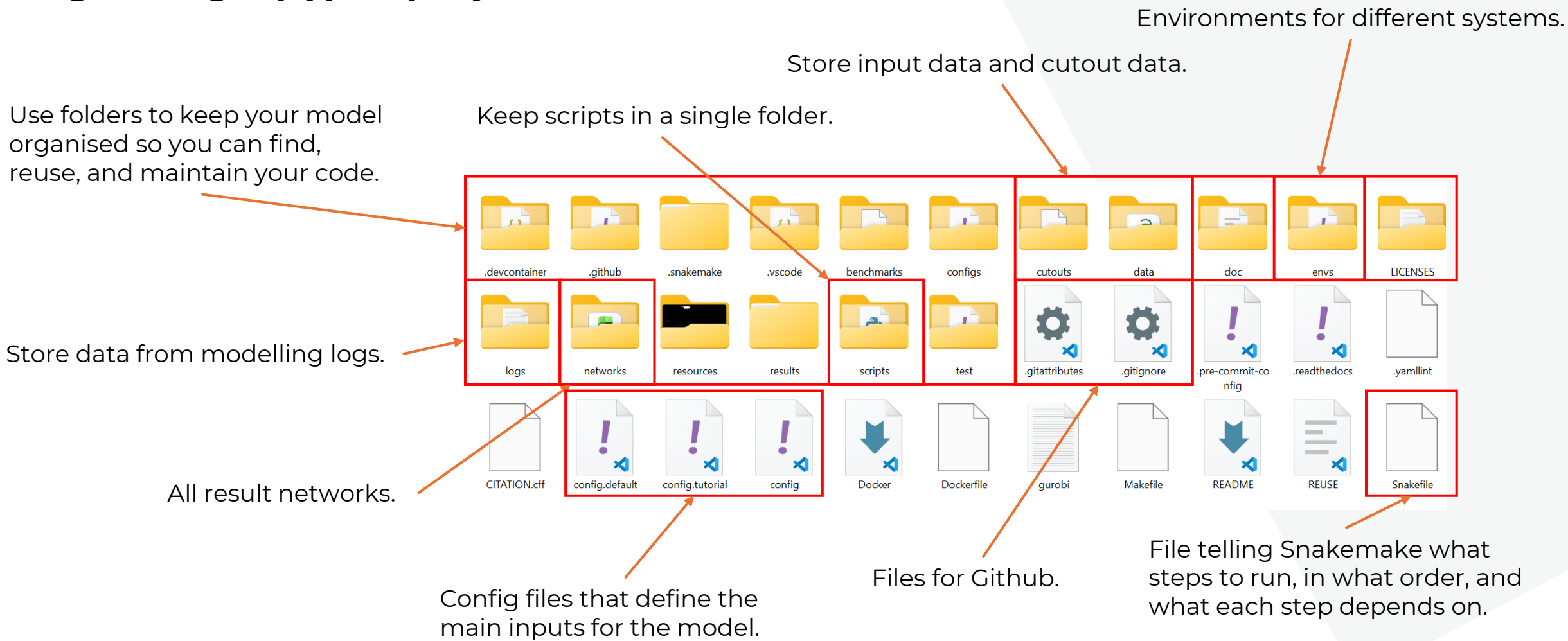
Green arrows - Process flow
 Blue arrows - Coordination

The modelling workflow:

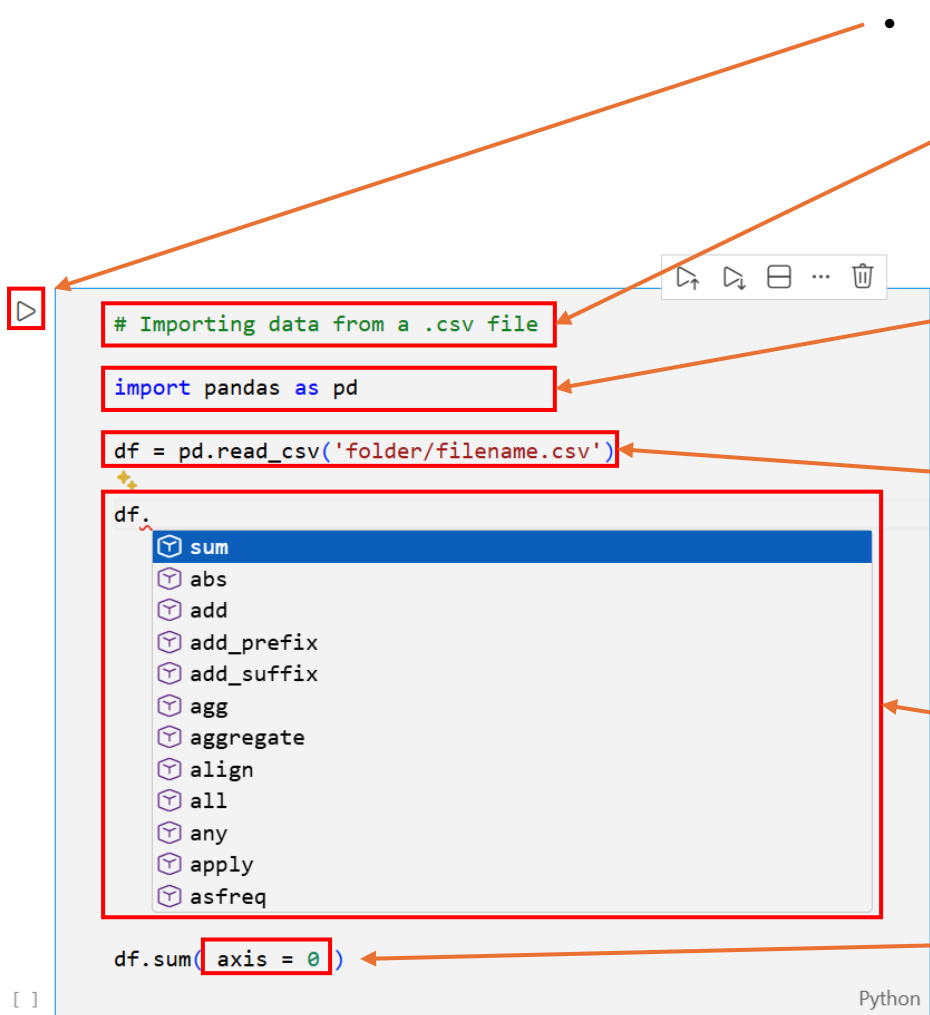
- **Data collection and import** into the model.
- **Constraint formulation and application using Linopy.**
- **Solve** the model using HiGHS or other solvers.
- **Review and analyse** the results.
- Snakemake **orchestrates** the processes



Organising a pypsa-project



Components of a Python Script



The screenshot shows a Jupyter notebook cell with the following code:

```
# Importing data from a .csv file
import pandas as pd
df = pd.read_csv('folder/filename.csv')
df.
df.sum(axis = 0)
```

Annotations with orange arrows point to specific parts of the code:

- A red box around the first line of code, with an arrow pointing to it from the first bullet point.
- A red box around the second line of code, with an arrow pointing to it from the second bullet point.
- A red box around the third line of code, with an arrow pointing to it from the third bullet point.
- A red box around the fourth line of code, with an arrow pointing to it from the fourth bullet point.
- A red box around the fifth line of code, with an arrow pointing to it from the fifth bullet point.

- Clicking on the triangle will execute the cell in Python.
- Comments start with # and help explain code; Python ignores everything after the # on that line.
- Pandas is a library designed for handling and analyzing large datasets efficiently.
- Import brings in external packages; use "as" to abbreviate the package name for easier reference
- Assign data to a variable named "df," short for dataframe.
- "pd" prefix tells Python to use the function from the Pandas library.
- The read_csv function opens a CSV file and converts it into a dataframe you can work with.
- After creating df, you access its methods by typing df. (dot) followed by the method name.
- sum() adds up values in the dataframe; axis=0 specifies it sums down each column.

Accessing the PyPSA API

Old component API Expected deprecation in V2.0.0

```
import pypsa

network = pypsa.Network('basic_network.xlsx')
```

```
network.generators
```

```
network.generators_t.p_max_pu
```

```
network.optimize()
```

```
network.generators_t.p
```

Import a pypsa network and create an object in the variable network.

Enable the new components api

Access all the static generator data as a dataframe.

Access all the dynamic generator data as a dataframe.

Optimize the model.

Access results data as a dataframe.

Python

New component API >V 1.0.0

```
import pypsa

pypsa.options.api.new_components_api = True

network = pypsa.Network('basic_network.xlsx')
```

```
network.generators.static
```

```
network.generators.dynamic.p_max_pu
```

```
network.optimize()
```

```
network.generators.dynamic.p
```

Python

Dataframe handling

Old component API Expected deprecation in V2.0.0

```
import pypsa

network = pypsa.Network('basic_network.xlsx')

network.generators
network.generators_t.p_max_pu

network.optimize()

network.generators_t.p
```

Python

All data is stored as pandas
DataFrames.

Apply the full range of DataFrame
operations—like `sum()`, `resample()`,
and `loc[]`

It is recommended that users
familiarise themselves with Pandas.



New component API >V 1.0.0

```
import pypsa
pypsa.options.api.new_components_api = True

network = pypsa.Network('basic_network.xlsx')

network.generators.static
network.generators.dynamic.p_max_pu

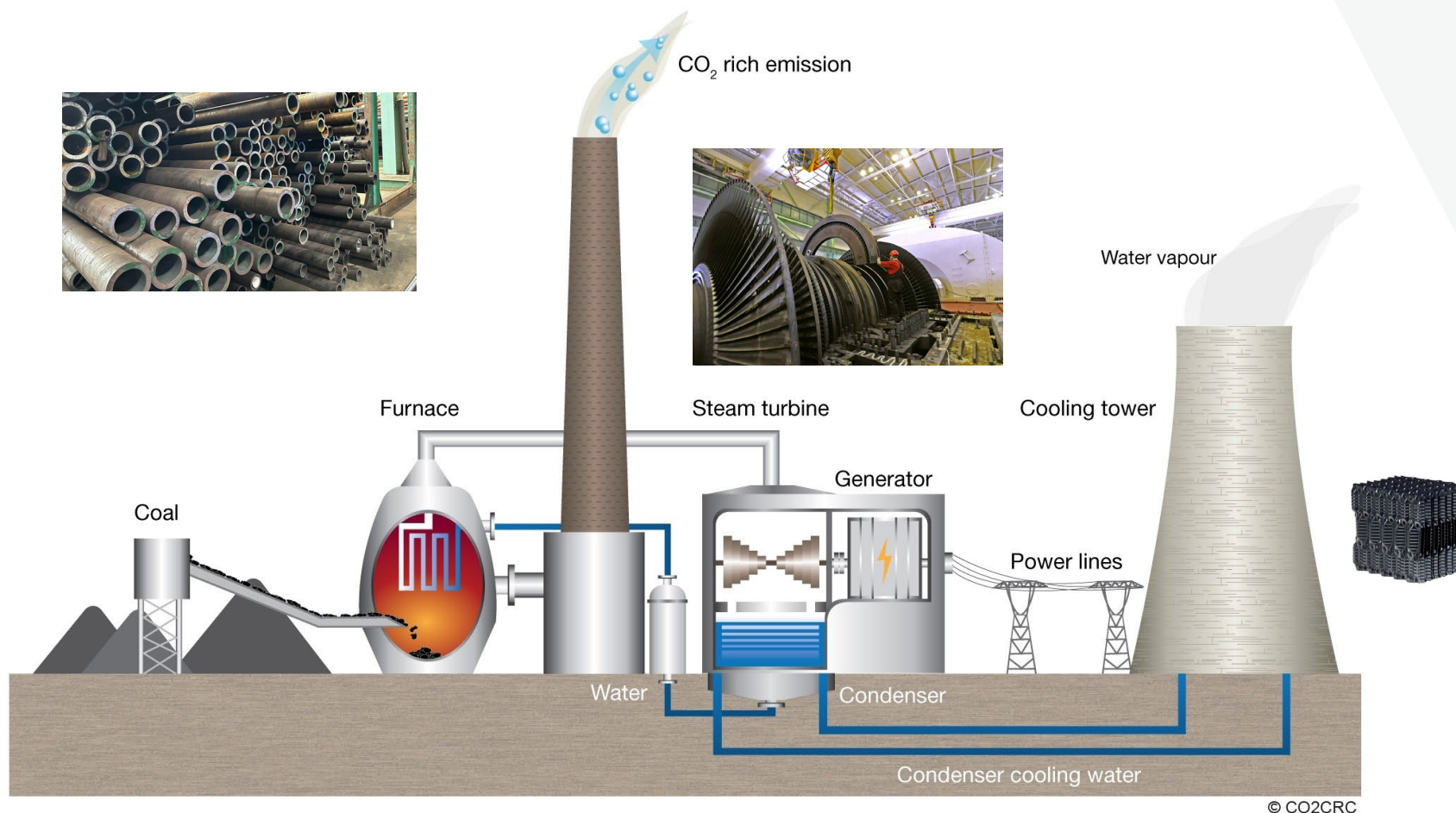
network.optimize()

network.generators.dynamic.p
```

Python



Formulation of constraints

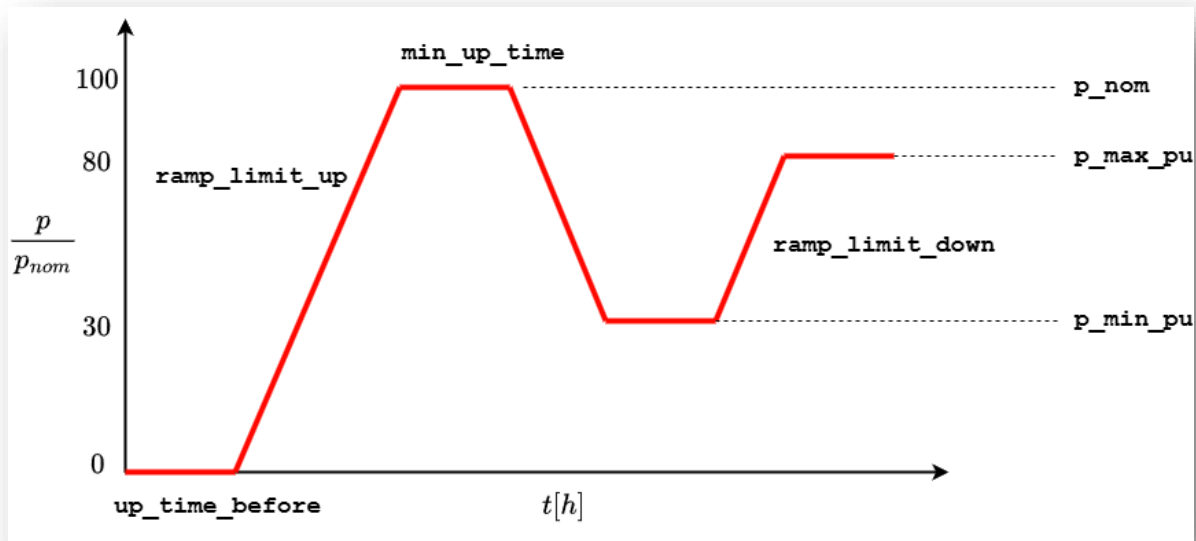


Considerations

- CAPEX
- Fuel and operational cost
- Long startup times
- Slow ramp rates
- Limits to maximum output

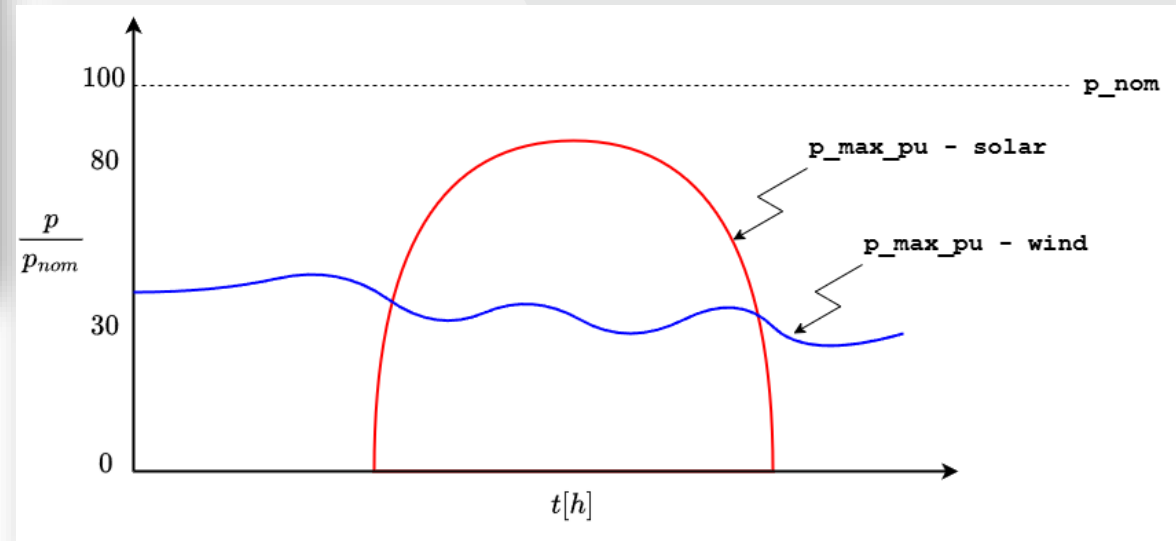
[centerwaysteel](#)
[australiansolarquotes](#)
[ethosenergy](#)
[vistechcooling](#)

Formulation of constraints



marginal_cost
[currency/MWh]

capital_cost
[currency/MW]
Additional capacity



Activity

```
import pypsa
```

```
network = pypsa.Network()
```

```
network.component.
```

Kahoot!

Python

Google Colab Interface

The training notebook is at this site:

- <https://priyeshgosai.github.io/pypsa-meets-earth-lab-2025/>
- Run the model using Google Colab. After opening a lesson, click on the rocket at the top to launch in Colab.
- At the top of each notebook, they can select whether they want to connect to:
 - Google Drive (optional)
 - This will allow users to link the sheet to their Google Drive and adjust the Excel spreadsheet using Google Sheets.
 - Files will not be lost if you close the session.
 - If you do not select this option, the notebook will use the virtual machine's storage.
 - Install packages (compulsory on first run in every session)
 - Download files (compulsory on first run in every session), only required for Parts 2 and 3.

The screenshot displays the Google Colab interface for a notebook titled "PyPSA Demystified". The interface includes a header with the notebook title, a search bar, and a "Launch on Colab" button. Below the header, the "Session Instructor" section introduces Priyesh Gosai, Director of Energy Systems at PAS (The Human Reliability Company). The notebook content is organized into parts: "Part 1: Getting started with PyPSA" and "Part 2: Developing a model using Excel". The "Part 2" section is expanded, showing three sub-sections: "Connect to Google Drive", "Install Packages", and "Download the file for this notebook". Each sub-section contains a checkbox and a "Show code" link. The "Connect to Google Drive" section has a checkbox labeled "CONNECT_TO_DRIVE:" which is checked. The "Install Packages" section has a checkbox labeled "INSTALL_PACKAGES:" which is checked. The "Download the file for this notebook" section has a checkbox labeled "DOWNLOAD_FILE:" which is checked. The code editor at the bottom shows the first line of code: `import os`.

Coding Exercise



Priyesh Gosai
Energy Modelling Consultant

Tel (UK): +44 774 109 8913
Email: pgosai@pas-sa.co.za
Location: Edinburgh, Scotland

GitHub Username: PriyeshGosai
LinkedIn: <https://www.linkedin.com/in/gosaip/>

www.pas-sa.co.za



<https://priyeshgosai.github.io/pypsa-meets-earth-lab-2025/intro.html>

