

Workshop

PyPSA-Eur: A Sector-Coupled Open Optimisation Model of the European Energy System

Dr. Fabian Neumann

Dr. Iegor Riepin

(Technische Universität Berlin)

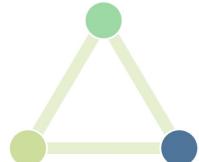
Dr. Markus Millinger

(RISE)

Supported by:



on the basis of a decision
by the German Bundestag



Gothenburg – 24-25 November 2025

github.com/pypsa/pypsa-eur
pypsa-eur.readthedocs.io
resilient-project.github.io



What is PyPSA?

Our research focus:

- Cost-effective pathways to reduce greenhouse gas emissions
- Evaluation of grid expansion, hydrogen strategies, carbon management strategies
- Co-optimisation of generation, storage, conversion and transmission infrastructure
- Algorithms to improve the tractability of models
- All open source and open data

The image shows the PyPSA website homepage. At the top, there's a navigation bar with the PyPSA logo, 'Features', 'Ecosystem', 'Models', 'Documentation', and a search icon. A banner at the top right says 'New PyPSA 1.0 is here!' with a small icon. The main title 'Welcome to the PyPSA Community!' is prominently displayed in large, bold, black and pink text. Below it, a subtitle explains: 'PyPSA stands for Python for Power System Analysis. Free. Open-Source. Batteries included.' Two buttons are visible: 'Get Started' and 'GitHub Repository'. In the bottom right corner of the page, there's a dark box containing a Python code snippet for a demonstration:

```
+ pypsa-demo python example.py

import pypsa

# Create a new network
n = pypsa.Network()

# Add components
n.add("Bus", "mybus")
n.add("Load", "myload", bus="mybus", p_set=100)
n.add("Generator", "mygen", bus="mybus", p_nom=100, marginal_cost=20)

# Optimize the network
n.optimize()

# Get statistics
n.statistics()
n.statistics.energy_balance()

# Plot results
n.plot()
n.explore()

# Start your journey here ...
```

Application examples

NGOs and international organisations



TSOs



International



PyPSA:

Python for Power System Analysis

Capabilities

Capacity expansion (linear)

- single-horizon
- multi-horizon

Market modelling (linear)

- Linear optimal power flow
- N-1 security-constraints
- Unit commitment
- Redispatch
- Rolling-horizon

Non-linear power flow

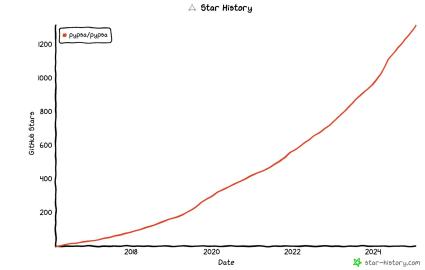
Modelling-to-Generate-Alternatives

With components for

- Electricity transmission networks and pipelines
- Generators with **unit commitment constraints**
- **Variable** generation with time series (e.g. wind, solar, hydro)
- **Storage** with efficiency losses and inflow/spillage for hydro
- **Conversion** between energy carriers (PtX, CHP, BEV, DAC)

Backend

- all data stored in **pandas**
- framework built for performance with large networks and time series
- interfaces to major **solvers** (Gurobi, CPLEX, HiGHS, Xpress), with **linopy** (*by PyPSA devs*)
- highly **customisable**, but **no GUI**
- Suitable for greenfield, brownfield & pathway studies





Economic Dispatch (ED)

Model short-term market dispatch with unit commitment, renewables, storage, multi-carrier conversion, and more.



Linear Optimal Power Flow (LOPF)

Optimize power flow in AC-DC networks using linearized KVL/KCL constraints with optional loss approximations.



Security-Constrained LOPF (SCLOPF)

Ensure system reliability by accounting for line outage contingencies under N-1 conditions.



Capacity Expansion Planning (CEP)

Plan long-term infrastructure investments for generation, storage, transmission, and conversion with continuous or discrete options.



Modular Design

Clean separation between data and modeling code enables flexible scenario development.



Pathway Planning

Co-optimize multi-period investments to design energy transition pathways with perfect foresight planning.



Rolling-Horizon Optimisation

Solve large-scale problems sequentially with myopic foresight and dynamic updates across time horizons.



Stochastic Optimisation

Two-stage stochastic programming for uncertainty handling with weighted scenarios and separate investment-dispatch decisions.



Modelling-to-Generate-Alternatives (MGA)

Explore alternative system designs with similar costs to understand trade-offs and decision flexibility.



Performance

Designed to scale with high-resolution networks, minimizing memory usage and solver overhead.



Policy Constraints

Built-in support for CO₂ limits, subsidies, resource limits, expansion limits, and growth constraints.



Custom Constraints

Impose custom objectives, variables and constraints using Linopy for specialized requirements.



Solver Flexibility

Support for wide range of LP, MILP, and QP solvers from open-source to commercial solutions.



Data Backbone

Uses pandas for data handling and linopy for optimization with efficient solver interfacing.



Resolution Control

Flexible control over temporal, spatial, and sectoral scope and detail for any analysis needs.



Sector-Coupling

Model integrated multi-carrier energy systems with heat pumps, electrolyzers, EVs, and synthetic fuels.



Standard Grid Components

Includes standard types for lines and transformers from pandapower for realistic grid modeling.



Static Power Flow Analysis

Compute non-linear and linearised load flows for AC-DC grids using Newton-Raphson method.



Visualisations

Plot statistics, time series, spatial distributions of line loadings and dispatch decisions.



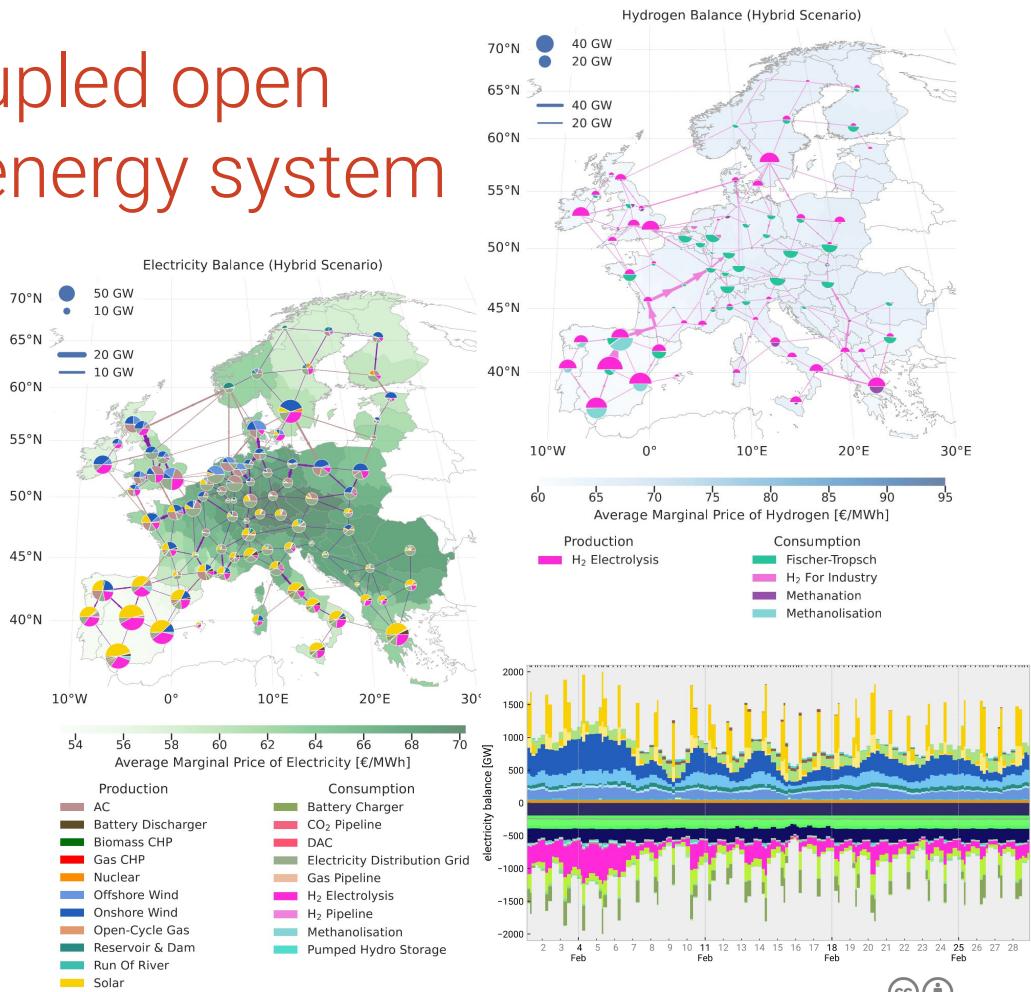
Scalability

Handle models from small prototypes to continent-scale systems on high-performance clusters.

PyPSA-Eur: A sector-coupled open model of the European energy system

Automated **workflow** to build energy system model of Europe from raw open data with high spatial and temporal resolution:

1. OSM power lines (>60 kV) + TYNDP
2. a database of existing **power plants**,
3. time series for electricity **demand**,
4. time series for wind/solar **availability**, and
5. geographic wind/solar **potentials**
6. **cost and efficiency assumptions**
7. methods for **model simplification**
8. more for sector-coupled networks like pipelines, LNG terminals, electric vehicles, industry locations,



Energy infrastructure planning in PyPSA as an optimisation problem

Find the long-term cost-optimal energy system, including investments and short-term costs:

$$\text{Min} \left[\begin{array}{c} \text{Yearly} \\ \text{system costs} \end{array} \right] = \text{Min} \left[\sum_n \left(\begin{array}{c} \text{Annualised} \\ \text{capital costs} \end{array} \right) + \sum_{n,t} \left(\begin{array}{c} \text{Marginal} \\ \text{costs} \end{array} \right) \right]$$

subject to

- meeting energy demand at each node n (e.g. region) and time t (e.g. hour of year)
- transmission constraints between nodes and linearised power flow
- wind, solar, hydro (variable renewables) availability time series $\forall n, t$
- installed capacity \leq geographical potentials for renewables
- fulfilling CO₂ emission reduction targets
- Flexibility from gas turbines, battery/hydrogen storage, HVDC links

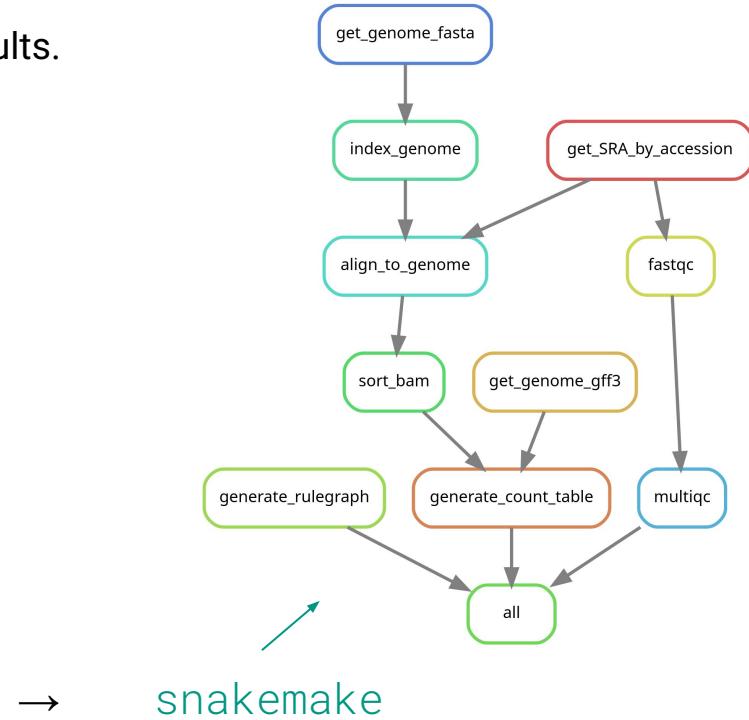


More on that later!

Challenges with data-driven modelling

Create a full pipeline of data processing from raw data to results.

- Many different data [sources](#)
- Many data sources need [cleaning](#) and [processing](#)
- Many [intermediate](#) scripts and datasets
- Data and software [dependencies](#) need to be managed
- Data and code [change](#) over time
- Want to be able to [reproduce](#) results
- Want to run many different [scenarios](#)



Requires a scalable [workflow management tool!](#)



[snakemake](#)

Originally comes from bioinformatics field.

Miniature example of snakemake

Snakefile

```
rule mytask:  
    input:  
        "data/{sample}.txt"  
    output:  
        "result/{sample}.txt"  
    script:  
        "scripts/mytask.py"
```

```
rule myplot:  
    input:  
        "result/{sample}.txt"  
    output:  
        "figures/{sample}.pdf"  
    script:  
        "scripts/myplot.py"
```

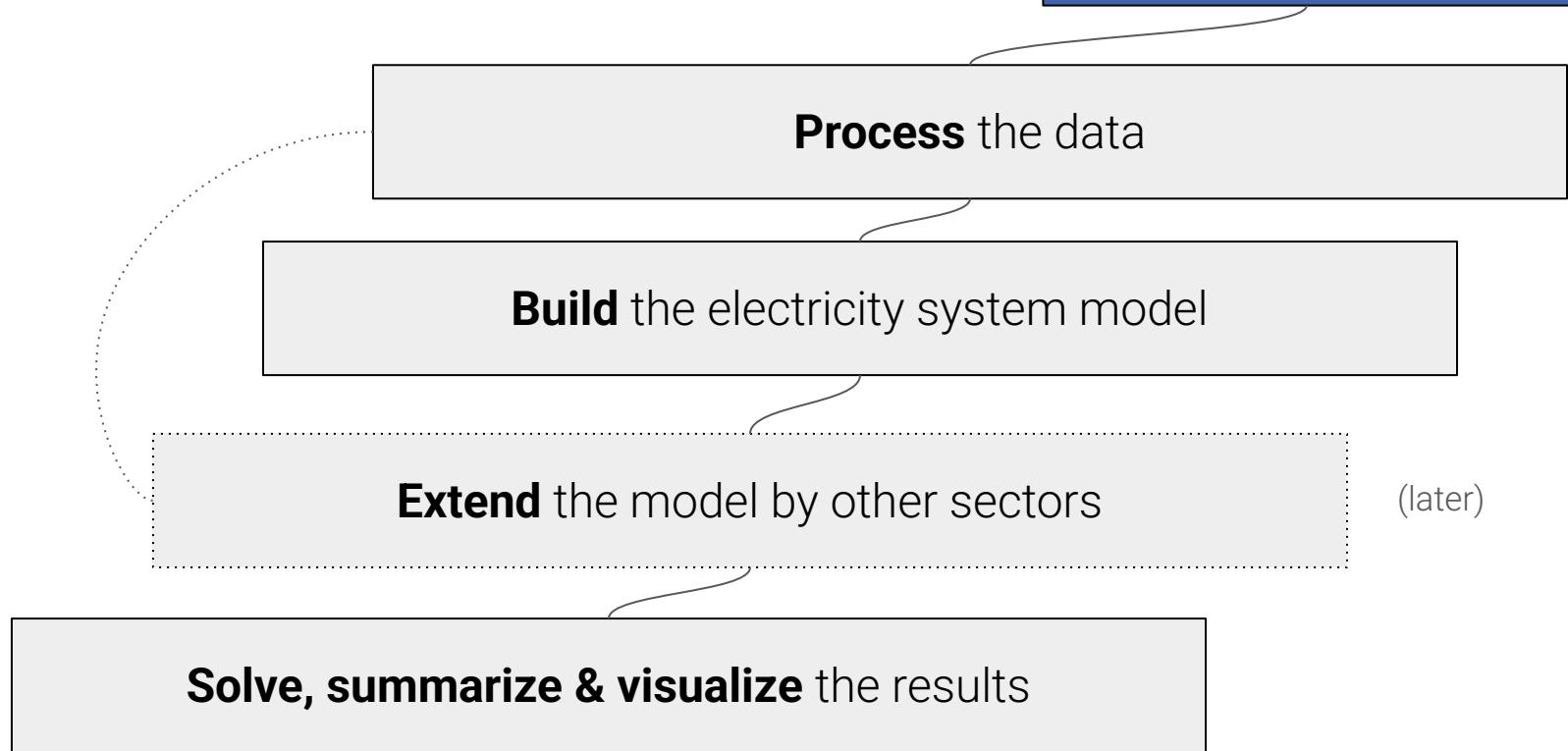
command: \$ snakemake figures/myfigure.pdf

snakemake workflow for the electricity sector



Simplified workflow structure

Automated **downloads**

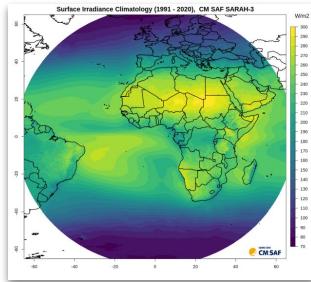


First, raw data is automatically downloaded.

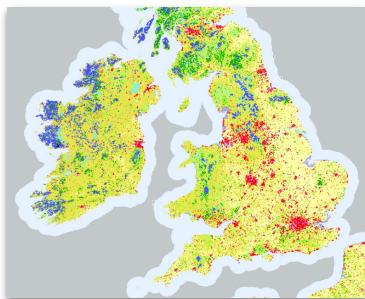
WDPA



SARAH-3



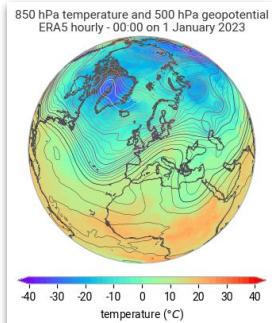
CORINE



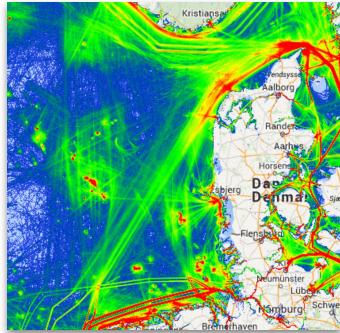
GEBCO



ERA5



World Bank



https://pypsa-eur.readthedocs.io/en/latest/data_sources.html



A screenshot of the Joint Research Centre Data Catalogue. It displays logos for the European Commission, Global Energy Monitor, and entsoe Transparency platform. The page lists various datasets related to energy generation, transportation, and consumption across Europe.

Simplified workflow structure

Automated **downloads**

Process the data

Build the electricity system model

Extend the model by other sectors

(later)

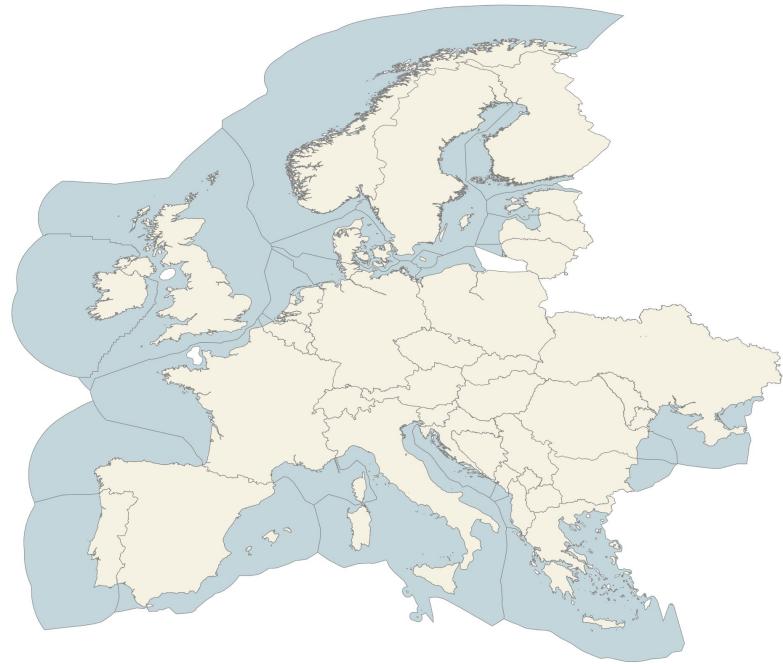
Solve, summarize & visualize the results

Steps to building PyPSA-Eur electricity system

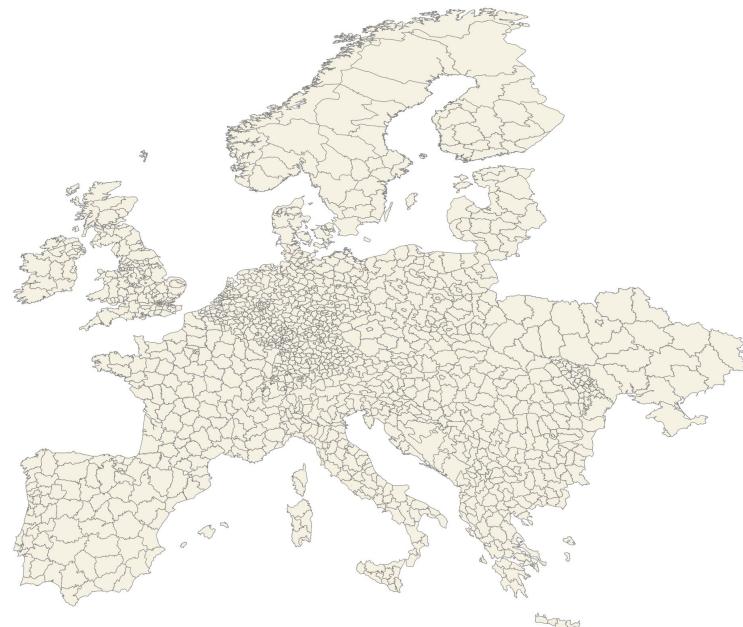
Retrieve onshore & offshore [polygons](#) for each country

`build_shapes`

Country shapes & exclusive economic zones (EEZ)



NUTS administrative regions (NUTS3)



Steps to building PyPSA-Eur electricity system

Retrieve onshore & offshore [polygons](#) for each country

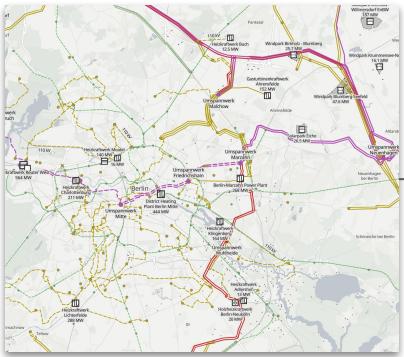
`build_shapes`

Construct a [base medium- to high-voltage network](#) with buses, transformers, AC & DC lines with DLR & TYNDP

`base_network,`
`build_transmission_projects`

Power grid topology

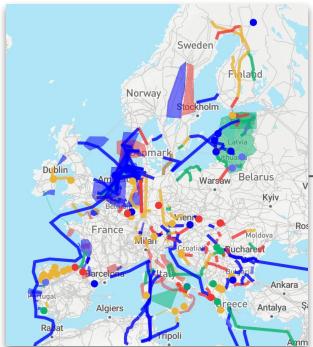
OpenStreetMap data



Apply **standard line types** for capacity and parameters.

Calculate **dynamic line rating** potential from weather data.

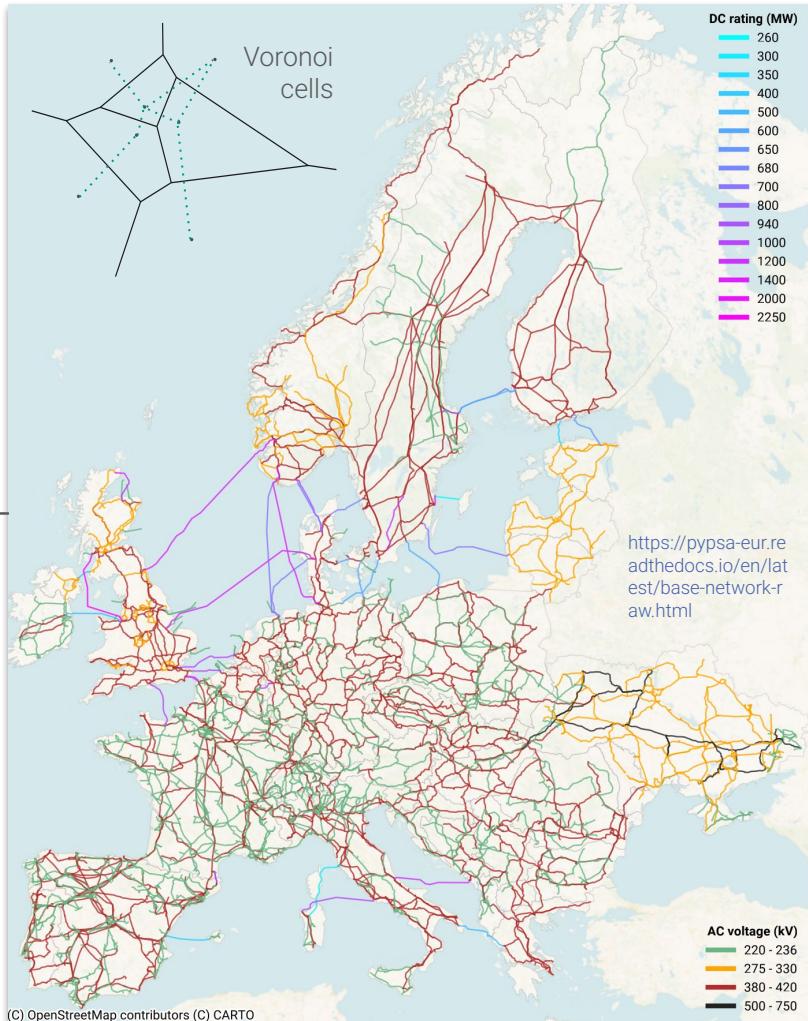
TYNDP projects



European network with

- ~5,800 buses
- ~7,300 AC lines (>220 kV)
- 36 HVDC links (+TYNDP)
- Optionally down to 60 kV

<https://www.nature.com/articles/s41597-025-04550-7>



Steps to building PyPSA-Eur electricity system

Retrieve onshore & offshore [polygons](#) for each country

`build_shapes`

Construct a [base high-voltage network](#) with
buses, transformers, AC & DC lines with DLR & TYNDP

`base_network,`
`build_transmission_projects`

Transform all transmission lines to 380 kV, remove
dead ends & cluster with [k-means](#) or [hierarchical](#) clustering

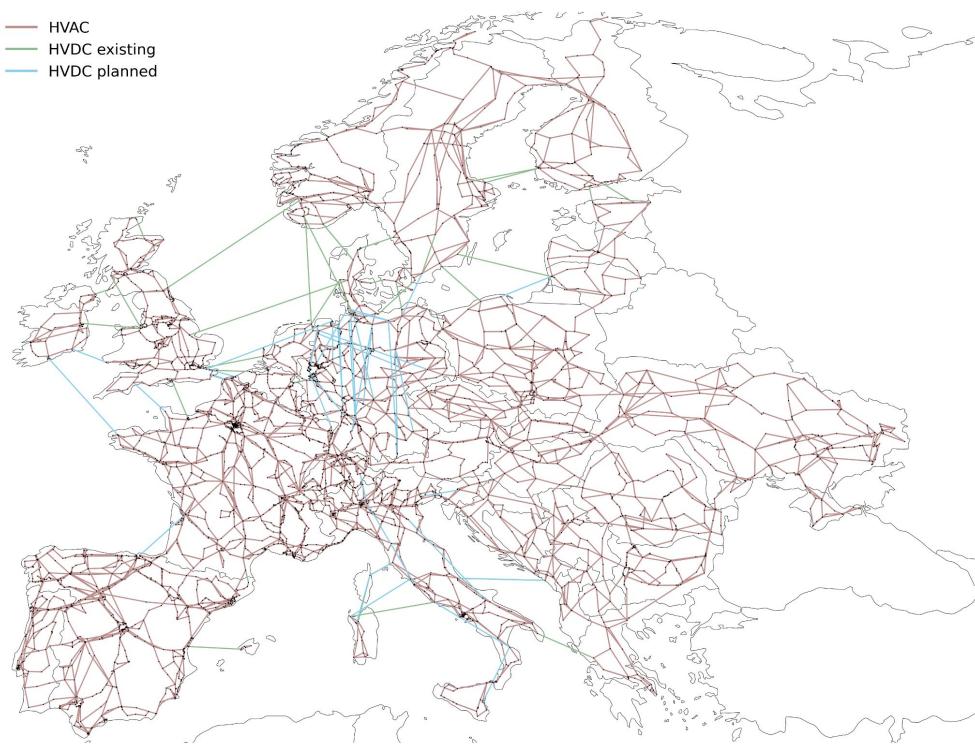
`simplify_network,`
`cluster_network`

Clustering the electricity network: simplify_network

Need to make the optimization problem
less **computationally challenging**...

...if we want to **co-optimise** generation,
storage, PtX conversion and transmission
infrastructure:

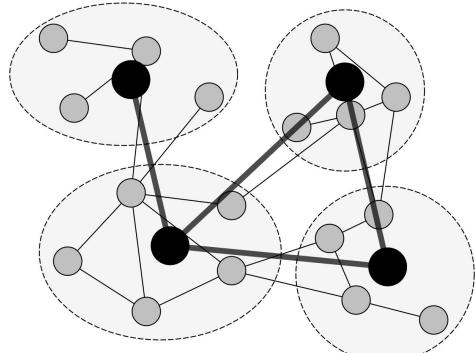
1. Lift all lines to **common voltage** level of 380 kV.
2. Remove **dead ends**.



Clustering the electricity network: cluster_network

Transformed
to 380 kV

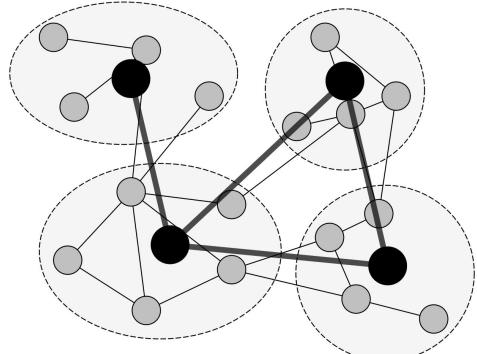
Clustered to
512 regions



Clustering the electricity network: cluster_network

Transformed
to 380 kV

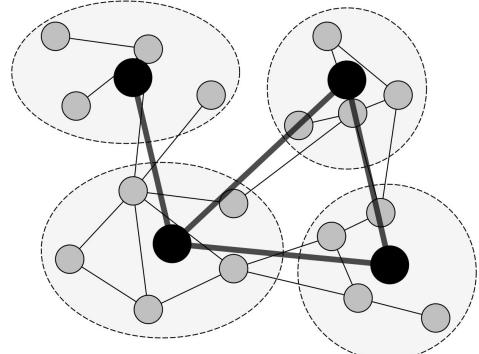
Clustered to
256 regions



Clustering the electricity network: cluster_network

Transformed
to 380 kV

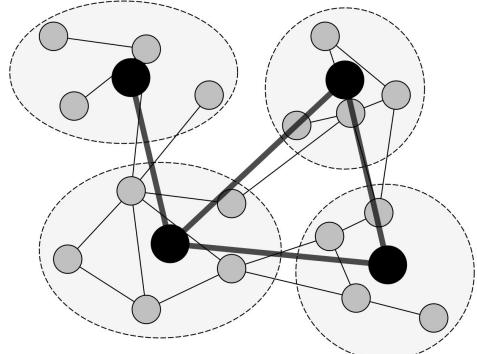
Clustered to
128 regions



Clustering the electricity network: cluster_network

Transformed
to 380 kV

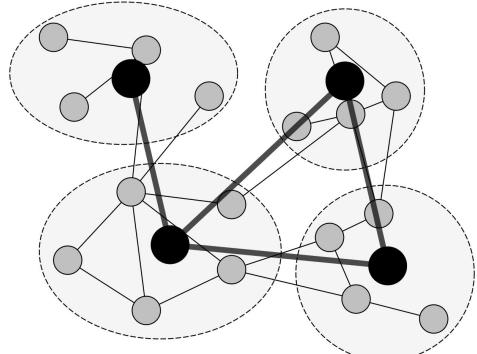
Clustered to
64 regions



Clustering the electricity network: cluster_network

Transformed
to 380 kV

Clustered to
41 regions



Steps to building PyPSA-Eur electricity system

Retrieve onshore & offshore polygons for each country	<code>build_shapes</code>
Construct a base high-voltage network with buses, transformers, AC & DC lines with DLR & TYNDP	<code>base_network,</code> <code>build_transmission_projects</code>
Transform all transmission lines to 380kV, remove dead ends & cluster with k-means or hierarchical clustering	<code>simplify_network,</code> <code>cluster_network</code>
Determine eligible areas for utility-scale PV & onshore/offshore wind park development	<code>determine_availability_matrix</code>
Build renewable capacity factor profiles for each clustered region based on land availability	<code>build_renewable_profiles,</code> <code>build_hydro_profile</code>

atlite: Convert weather data to energy systems data

Rule: build_renewable_profiles

pypi v0.4.1 conda-forge v0.4.1 Tests passing codecov 72% docs passing license MIT REUSE compliant
JOSS 10.21105/joss.03294 chat 63 online stackoverflow pypsa questions 38

Python library for converting **weather data** (e.g. wind, solar radiation, temperature, precipitation) into **energy systems data**:

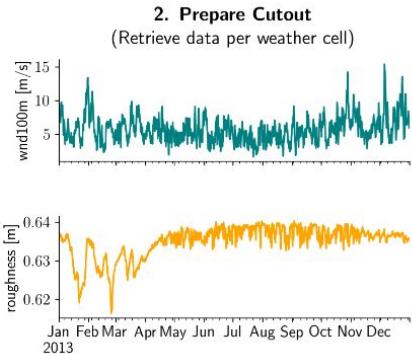
- solar photovoltaics
- solar thermal collectors
- wind turbines
- hydro run-off, reservoir, dams
- heat pump COPs
- dynamic line rating (DLR)
- heating and cooling demand (HDD/CDD)

It can also perform **land eligibility analyses**.

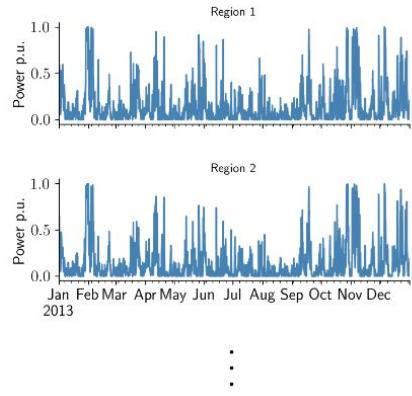
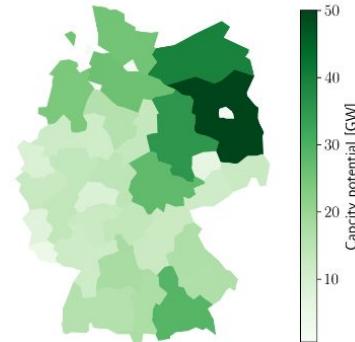
1. Create Cutout (Select spatio-temporal bounds)



2. Prepare Cutout (Retrieve data per weather cell)

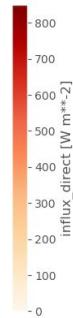
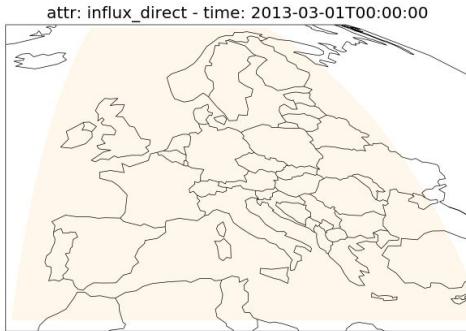


3. Convert Cutout (Calculate potentials and timeseries per region)

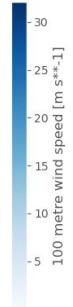
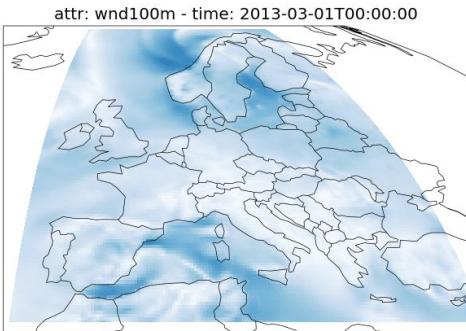


Time series for renewables

Historical meteorological weather data from ERA5 and SARAH-3
(up to 84 years, 30x30 km)



- Solar panel models
- orientation
 - material

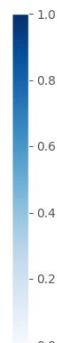
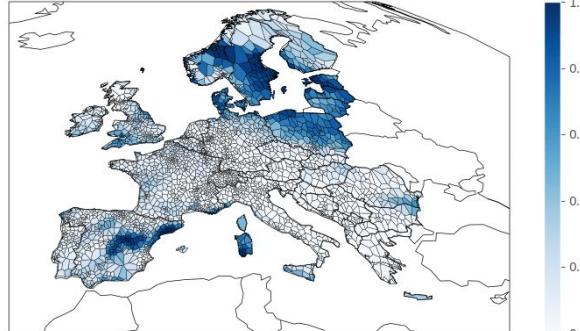
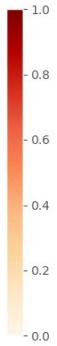
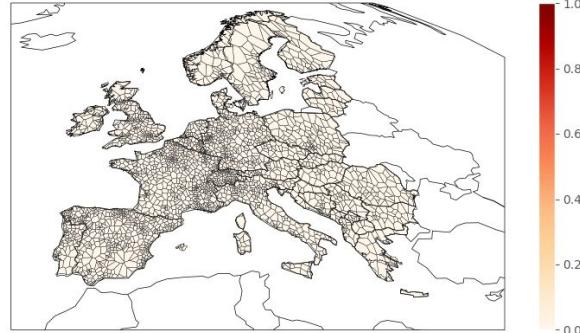


- Wind turbine models
- power curve
 - surface roughness

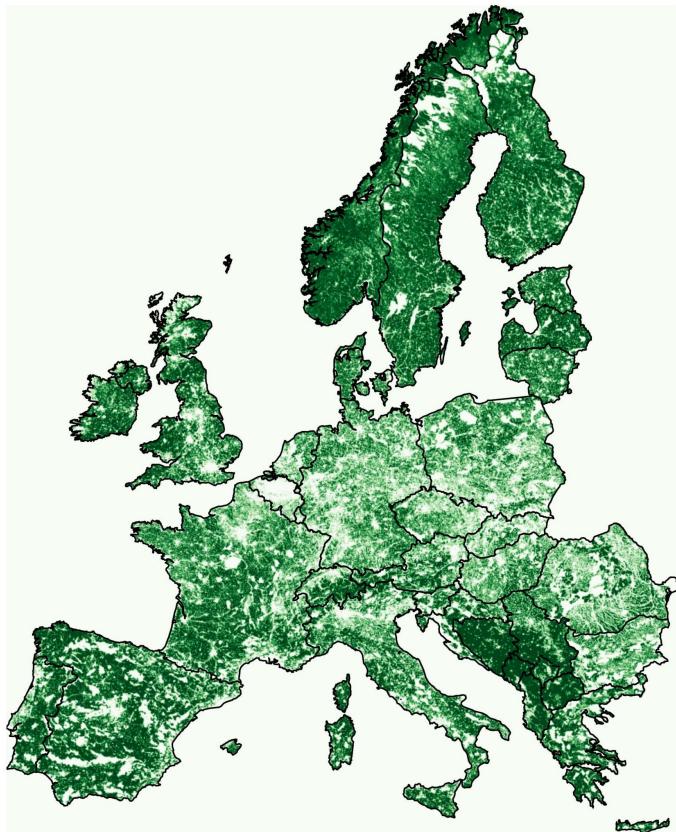
atlite: Convert weather data to energy systems data

pypi v0.4.1 conda-forge v0.4.1 Tests passing codecov 72% docs passing license MIT REUSE compliant
JOSS 10.21105/joss.03294 chat 63 online stackoverflow pypsa questions 38

Wind and solar capacity factors



Land availability for renewables



Example:

Onshore wind
in one clustered region



atlite: Convert weather data to energy systems data

[pypi v0.4.1](#) [conda-forge v0.4.1](#) [Tests passing](#) [codecov 72%](#) [docs passing](#) [license MIT](#) [REUSE compliant](#)
[JOSS 10.21105/joss.03294](#) [chat 63 online](#) [stackoverflow pypsa questions 38](#)

- CORINE / LUISA land cover
 - eligible land types
 - distance requirements
- NATURA / WDPA natural protection areas
- GEBCO bathymetry data
- Shipping lanes
- Distance to shore

Steps to building PyPSA-Eur electricity system

Retrieve onshore & offshore polygons for each country	<code>build_shapes</code>
Construct a base high-voltage network with buses, transformers, AC & DC lines with DLR & TYNDP	<code>base_network,</code> <code>build_transmission_projects</code>
Transform all transmission lines to 380kV, remove dead ends & cluster with k-means or hierarchical clustering	<code>simplify_network,</code> <code>cluster_network</code>
Determine eligible areas for utility-scale PV & onshore/offshore wind park development	<code>determine_availability_matrix</code>
Build renewable capacity factor profiles for each clustered region based on land availability	<code>build_renewable_profiles,</code> <code>build_hydro_profile</code>
Prepare existing renewables and fossil power plants	<code>build_powerplants</code>

Welcome to powerplantmatching's documentation!

pypi v0.7.1 conda-forge v0.7.1 python >=3.10 Tests failing docs passing pre-commit.ci passed Ruff
DOI 10.5281/zenodo.3358985

A toolset for cleaning, standardizing and combining multiple power plant databases.

```
import powerplantmatching as pm
df = pm.powerplants(from_url=True)
df.query("DateIn > 2000")
```

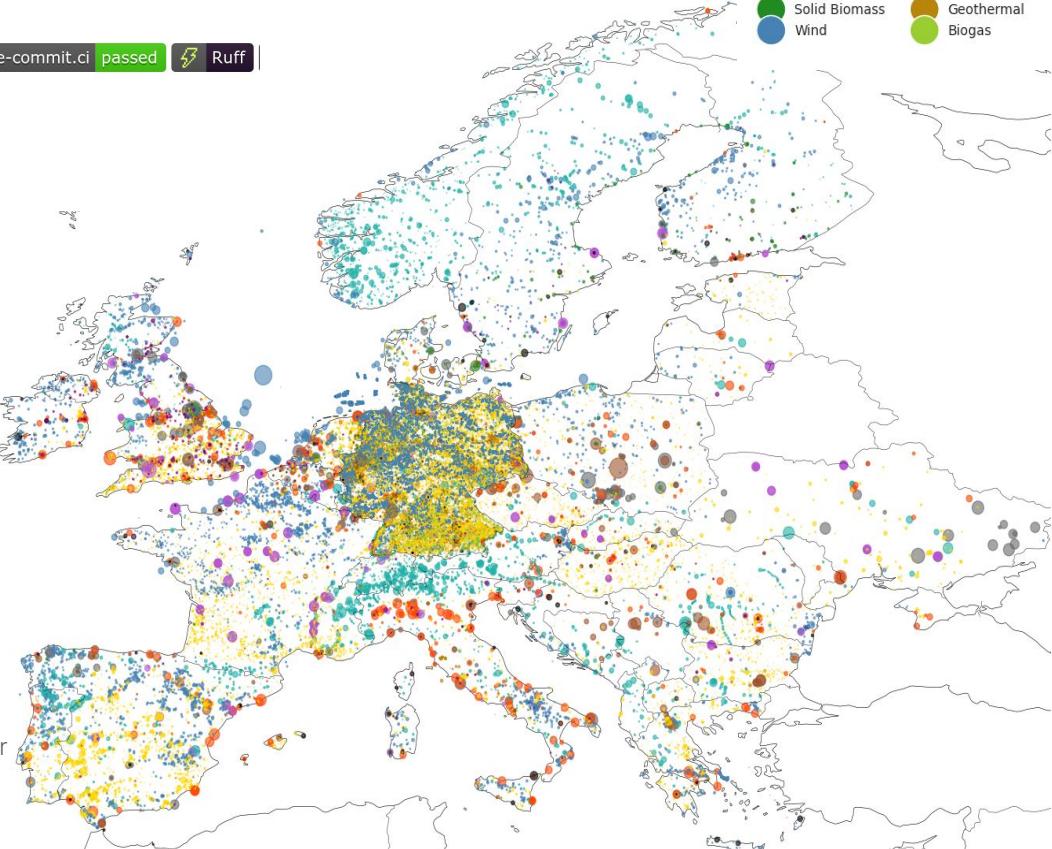
Sources

- Global Energy Monitor (GEM)
- Open Power System Data (OPSD)
- Global Energy Observatory
- World Resources Institute
- Marktstammdatenregister (MaStR)
- CARMA
- ENTSO-E, BNetzA, UBA, IRENA
- JRC for hydro power plants
- European Energy Storage Inventory

Attributes

- name / block
- fuel type
- technology
- country
- capacity
- commissioning year
- retirement year
- coordinates

<https://globalenergymonitor.org/projects/global-integrated-power-tracker/tracker-map/>



github.com/pypsa/powerplantmatching



Steps to building PyPSA-Eur electricity system

Retrieve onshore & offshore polygons for each country	<code>build_shapes</code>
Construct a base high-voltage network with buses, transformers, AC & DC lines with DLR & TYNDP	<code>base_network,</code> <code>build_transmission_projects</code>
Transform all transmission lines to 380kV, remove dead ends & cluster with k-means or hierarchical clustering	<code>simplify_network,</code> <code>cluster_network</code>
Determine eligible areas for utility-scale PV & onshore/offshore wind park development	<code>determine_availability_matrix</code>
Build renewable capacity factor profiles for each clustered region based on land availability	<code>build_renewable_profiles,</code> <code>build_hydro_profile</code>
Prepare existing renewables and fossil power plants	<code>build_powerplants</code>
Add generation, storage and demand to the network with techno-economic assumptions on costs and efficiencies, ...	<code>add_electricity,</code> <code>prepare_network</code>

Open database of techno-economic assumptions

- compiles **techno-economic assumptions** on energy system components
 - investment costs, FOM/VOM costs, efficiencies, lifetimes
 - for given years, e.g. 2020, 2030, 2040, 2050
 - from mixed sources, but prioritising **Danish Energy Agency** where available (and sensible)

Preview Code Blame 1097 lines (1097 loc) · 213 KB

Q fischer-tropsch

1	technology	parameter	value	unit	source	fu
217	Fischer-Tropsch	FOM	3.0	%/year	Agora Energiewende (2018): The Future Cost of Electricity-Based Synthetic Fuels (https://www.agora-energiewende.de/en/publications/the-future-cost-of-electricity-based-synthetic-fuels-1/), section 6.3.2.1.	
218	Fischer-Tropsch	VOM	4.4663	EUR/MWh_FT	Danish Energy Agency, data_sheets_for_renewable_fuels.xlsx	10
219	Fischer-Tropsch	capture rate	0.9	per unit	Assumption based on doi:10.1016/j.biombioe.2015.01.006	
220	Fischer-Tropsch	carbondioxide-input	0.326	t_CO2/MWh_FT	DEA (2022): Technology Data for Renewable Fuels (https://ens.dk/en/our-services/projections-and-models/technology-data/technology-data-renewable-fuels), Hydrogen to Jet Fuel, Table 10 / pg. 267.	Ir
221	Fischer-Tropsch	efficiency	0.799	per unit	Agora Energiewende (2018): The Future Cost of Electricity-Based Synthetic Fuels (https://www.agora-energiewende.de/en/publications/the-future-cost-of-electricity-based-synthetic-fuels-1/), section 6.3.2.2.	
222	Fischer-Tropsch	electricity-input	0.007	MWh_el/MWh_FT	DEA (2022): Technology Data for Renewable Fuels (https://ens.dk/en/our-services/projections-and-models/technology-data/technology-data-renewable-fuels), Hydrogen to Jet Fuel, Table 10 / pg. 267.	0.
223	Fischer-Tropsch	hydrogen-input	1.421	MWh_H2/MWh_FT	DEA (2022): Technology Data for Renewable Fuels (https://ens.dk/en/our-services/projections-and-models/technology-data/technology-data-renewable-fuels), Hydrogen to Jet Fuel, Table 10 / pg. 267.	0.
224	Fischer-Tropsch	investment	703726.4462	EUR/MW_FT	Agora Energiewende (2018): The Future Cost of Electricity-Based Synthetic Fuels (https://www.agora-energiewende.de/en/publications/the-future-cost-of-electricity-based-synthetic-fuels-1/), table 8: "Reference scenario".	W
225	Fischer-Tropsch	lifetime	20.0	years	Danish Energy Agency, Technology Data for Renewable Fuels (04/2022), Data sheet "Methanol to Power".	
956	methanation	lifetime	20.0	years	Guesstimate.	B

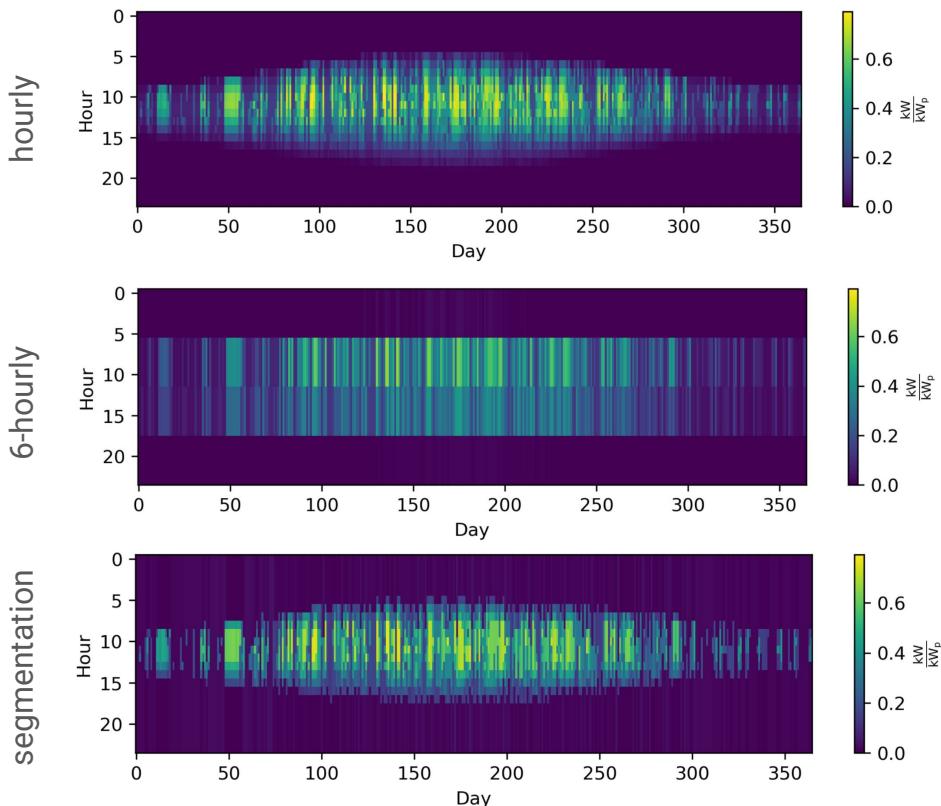
https://github.com/PyPSA/technology-data/blob/master/outputs/costs_2030.csv



Options for temporal aggregation

Multiple options:

1. averaging of every Nth hour
2. sampling every Nth hour (e.g. 3-hourly)
3. Non-equidistant segmentation with pre-defined number of segments using the `tsam` Python library from FZ Jülich



Introduction



tsam - time series aggregation module

<https://tsam.readthedocs.io/en/latest/newsDoc.html>

Simplified workflow structure

Automated **downloads**

Process the data

Build the electricity system model

Extend the model by other sectors

(later)

Solve, summarize & visualize the results

linopy: Linear optimization with N-D labeled variables



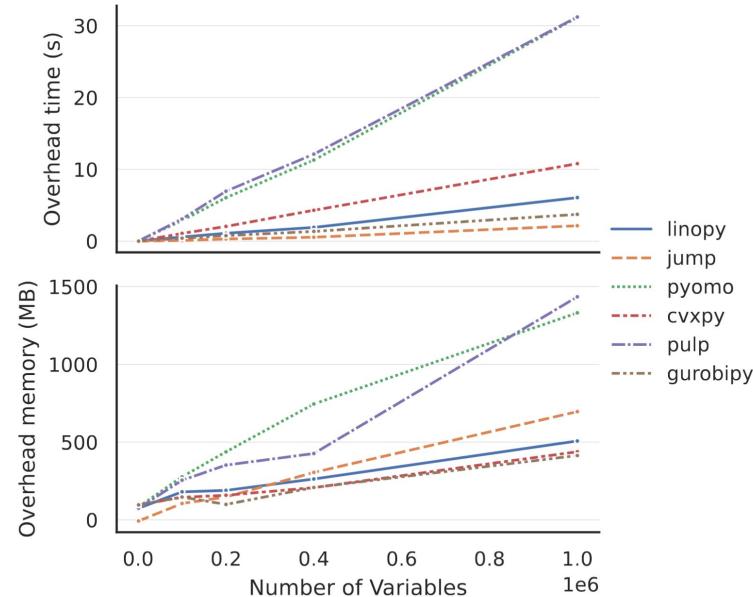
pypi v0.5.8 [CI](#) license MIT

Python library that facilitates **optimisation** with real-world, large-scale data.

It supports:

- Linear (LP),
- Mixed-Integer (MILP),
- Quadratic programming (QP).

It has been developed to make linear programming in Python easy, highly-flexible and – most importantly – **highly performant**.



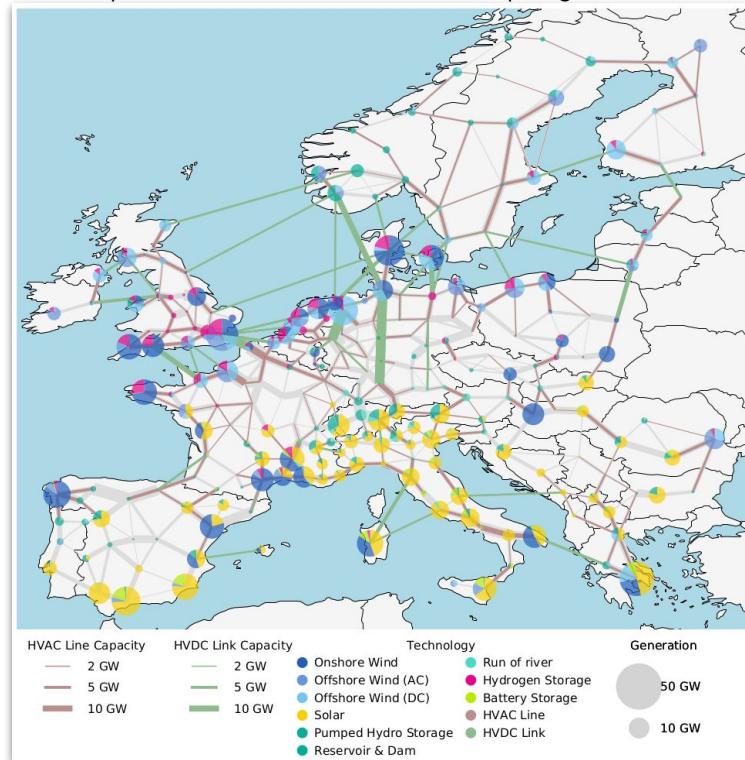
Solving and summarising networks

Hardware requirements:

- Building the model **can run locally** on most modern laptops. Very simple models can run with HiGHS solver.
- But access to a **commercial solver** and a larger **cluster/workstation** is required for solving problems (~250 GB RAM per scenario if resolution is very high)!

There is a **statistics module** in PyPSA designed to help with analysing solved networks and several figures/maps are created automatically.

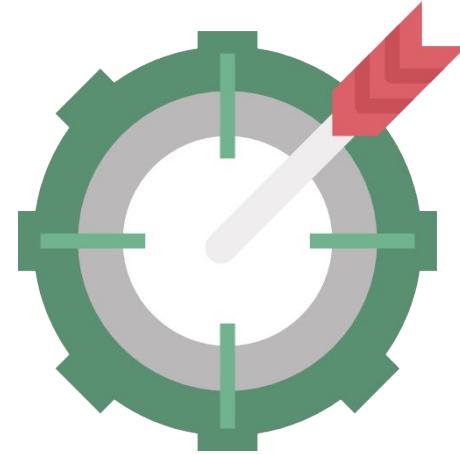
Example result without sector-coupling



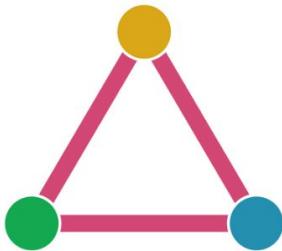
What is configurable?

electricity-only
examples

- Select subset of countries and focus countries (e.g. only DE)
- Select weather year (1940 - 2024 for ERA5)
- Specify CO₂ constraint and gas usage limit
- Tweak spatial resolution (between 41 and >1000 nodes)
- Tweak temporal resolution (from hourly to N-hourly)
- Customize cost assumptions (e.g. 2020, 2030, 2050)
- Parametrize technologies (e.g. wind turbine type, panel orientation)
- Define land use eligibility criteria (e.g. distance requirements)
- Pick a solver (HiGHS, Gurobi, CPLEX, Xpress...)
- Choose between greenfield or brownfield expansion



config.yaml



Search

Ctrl + K

Getting Started

Introduction

Installation

Tutorial: Electricity-Only

Tutorial: Sector-Coupled

Configuration

Wildcards

Configuration

Foresight Options

Techno-Economic Assumptions

Rules Overview

Retrieving Data

Building Electricity Networks



[https://pypsa-eur.readthedocs.io
/en/latest/configuration.html](https://pypsa-eur.readthedocs.io/en/latest/configuration.html)



Contents

Configuration Files

Top-level configuration

run

foresight

scenario

countries

snapshots

enable

co2 budget

electricity

atlite

renewable

conventional

lines

links

transmission projects

transformers

load

energy

biomass

solar_thermal

existing_capacities

sector

industry

costs

clustering

Configuration

PyPSA-Eur has several configuration options which are documented in this section.

Configuration Files

Any PyPSA-Eur configuration can be set in a `.yaml` file. The default configurations

`config/config.default.yaml` and `config/plotting.default.yaml` are maintained in the repository and cover all the options that are used/ can be set.

To pass your own configuration, you can create a new file, e.g. `my_config.yaml`, and specify the options you want to change. They will override the default settings and options which are not set, will be inherited from the defaults above.

Another way is to use the `config/config.yaml` file, which does not exist in the repository and is also not tracked by git. But snakemake will always use this file if it exists. This way you can run snakemake with a custom config without having to specify the config file each time.

Configuration order of precedence is as follows: 1. Command line options specified with `--config` (optional) 2. Custom configuration file specified with `--configfile` (optional) 3. The `config/config.yaml` file (optional) 4. The default configuration files `config/config.default.yaml` and `config/plotting.default.yaml`

To use your custom configuration file, you need to pass it to the `snakemake` command using the `--configfile` option:

```
$ snakemake -call --configfile my_config.yaml
```



Live Demo – Belgium / electricity-only / few days

Start with a dry-run:

```
$ snakemake -c1 solve_elec_networks --configfile config/test/config.electricity.yaml -n
```

Don't forget to activate your conda environment first!

Then execute the same command “for real” by dropping “-n” flag:

The “-c1” flag tells snakemake to run one job at a time; “-call” to use all cores.

```
$ snakemake -call solve_elec_networks --configfile config/test/config.electricity.yaml
```

To explore results, start a Jupyter notebook:

```
$ jupyter notebook
```

Practical Phase

(electricity-only)

2) Install conda environment

Installation links:

- Anaconda (bigger download):
- Miniconda (recommended):

```
$ conda update conda  
$ conda env create -f envs/environment.yaml  
$ conda activate pypsa-eur
```

4) Explore PyPSA network in a Jupyter notebook

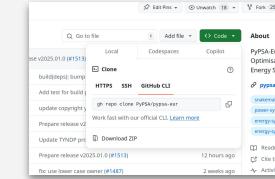
```
import pypsa  
fn = "results/test-elec/networks/base_s_5_elec_.nc"  
n = pypsa.Network(fn)  
n.statistics()  
n.plot()
```

1) Download the repository

Open a terminal / CMD / Anaconda Prompt and type:

```
$ cd ~/path/to/my/directory  
$ git clone https://github.com/PyPSA/pypsa-eur.git  
$ cd pypsa-eur
```

You can also download
the repository as a ZIP by
hand.



3) Run PyPSA-Eur tutorial with snakemake

Guide:

<https://pypsa-eur.readthedocs.io/en/latest/tutorial.html>

```
$ snakemake -call solve_elec_networks  
--configfile  
config/test/config.electricity.yaml
```

Users of Windows, add two lines to YAML:
run:
use_shadow_directory: false



Small exploratory configuration tasks

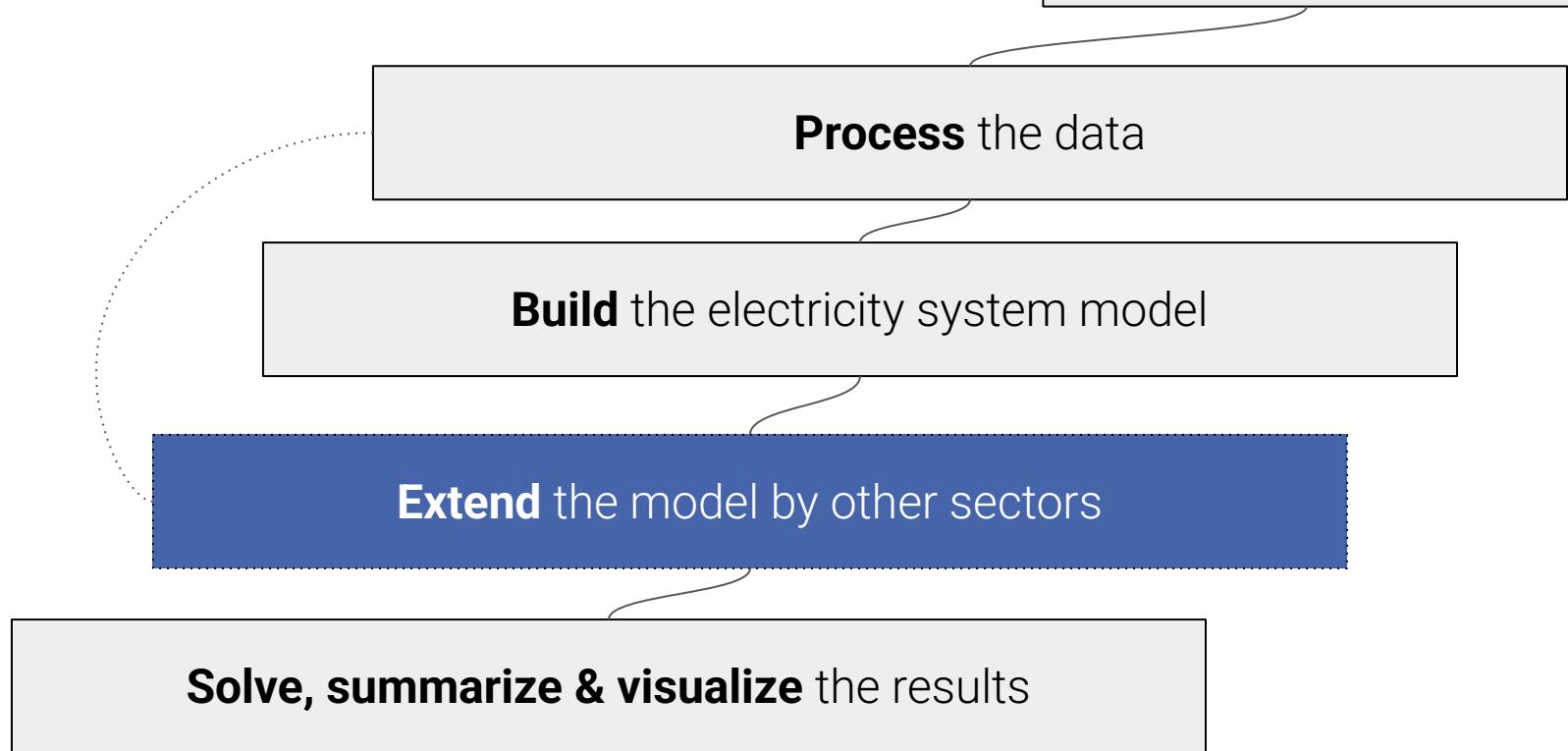
(electricity-only)

Go to <https://pypsa-eur.readthedocs.io/en/latest/configuration.html> and <https://github.com/PyPSA/pypsa-eur/blob/master/config/config.default.yaml> and try to find out how to configure some of the settings for **electricity-only models** listed below:

1. Increase the maximum line loading from 70% to 100%.
2. Disable power transmission grid reinforcements.
3. Activate dynamic line rating with default settings.
4. Activate linearised transmission loss approximation.
5. Deactivate the estimation of existing renewable capacities.
6. Change the techno-economic assumptions to the year 2020.
7. Remove the option to build hydrogen or battery storage.

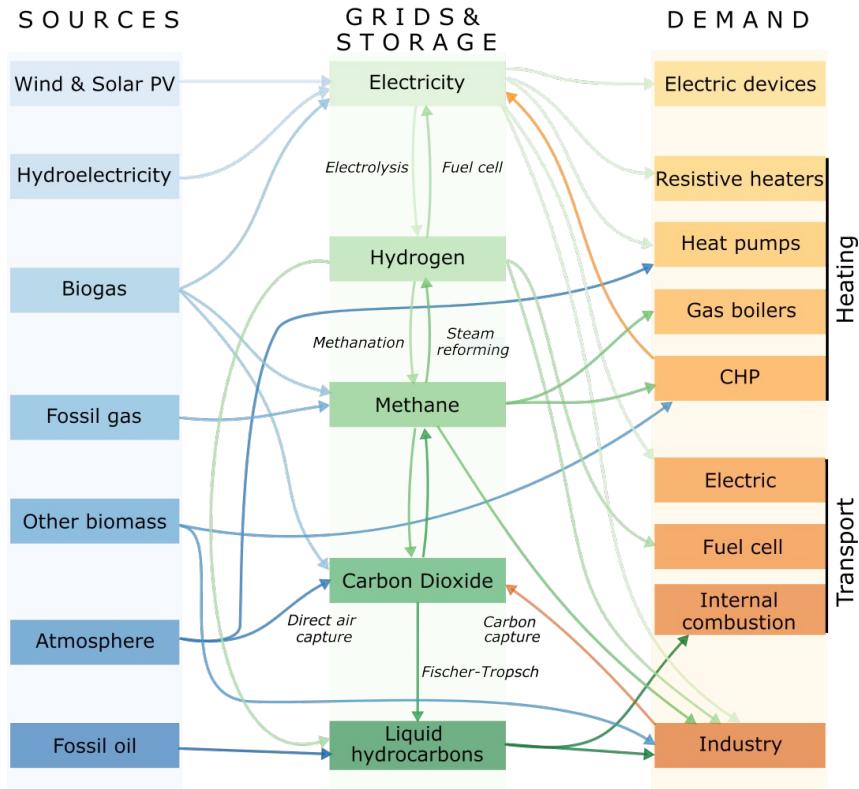
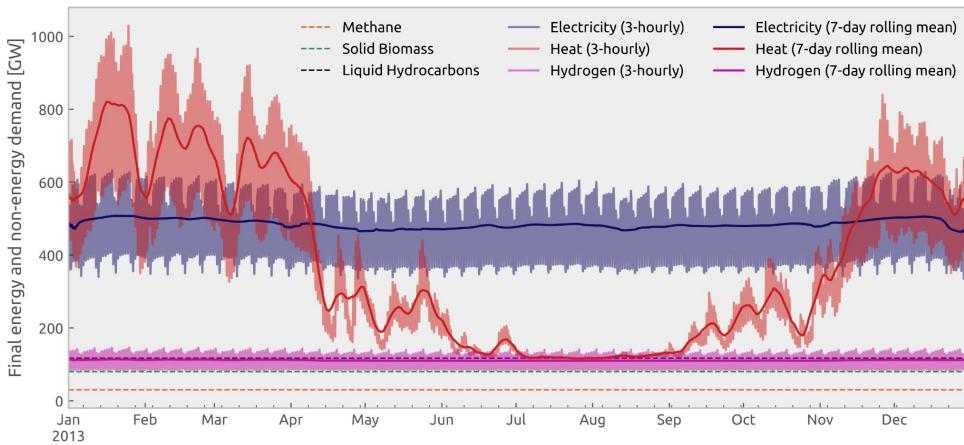
Simplified workflow structure

Automated **downloads**

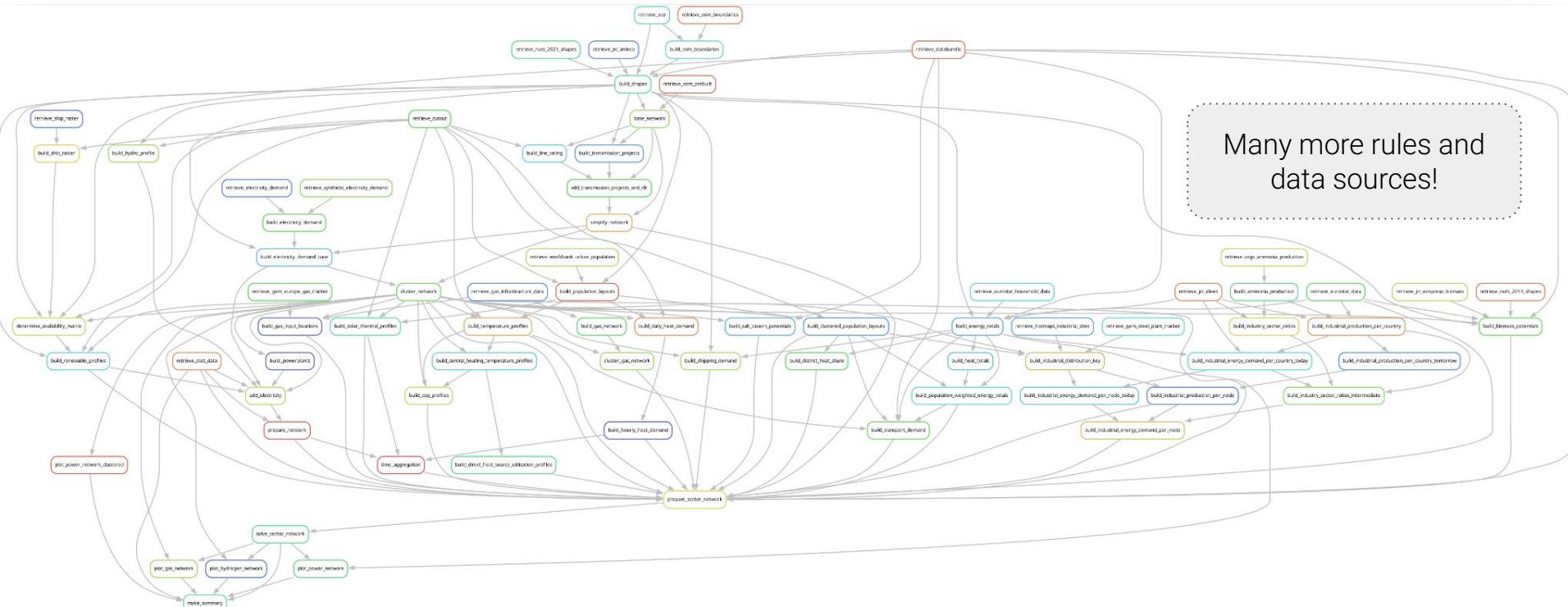


Coupling with other sectors

- transport sector (EVs, shipping, aviation)
- heating sector (district heating, individual)
- industry sector (steel, chemicals, ammonia, ...)
- carbon management (CCUTS) + biomass
- hydrogen, CO₂ and gas networks



Extension by other sectors requires more data!



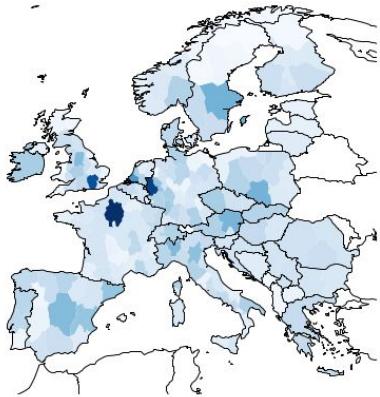
High-resolution version:

<https://tubcloud.tu-berlin.de/s/E7tx3BagXsKXLre>

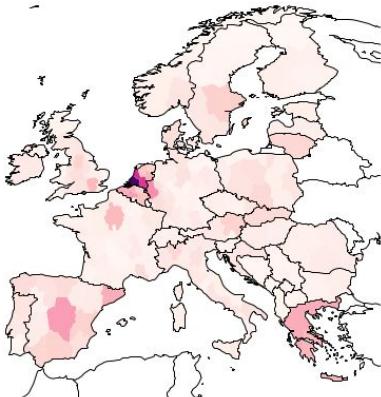


Spatial distribution of energy demands

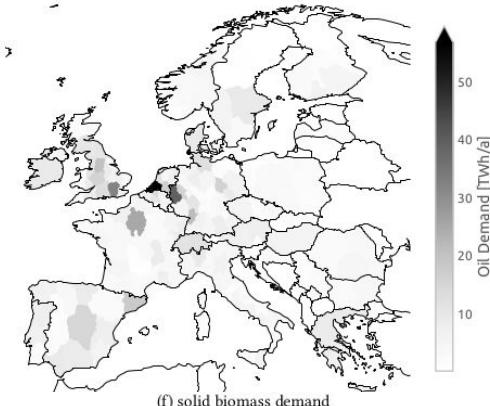
(a) electricity demand



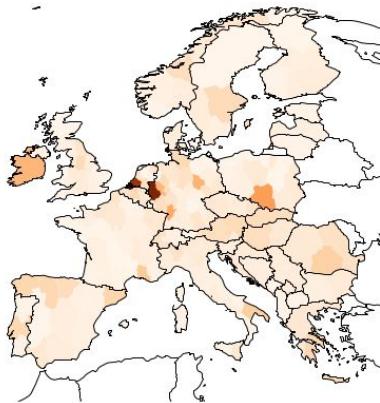
(b) hydrogen demand



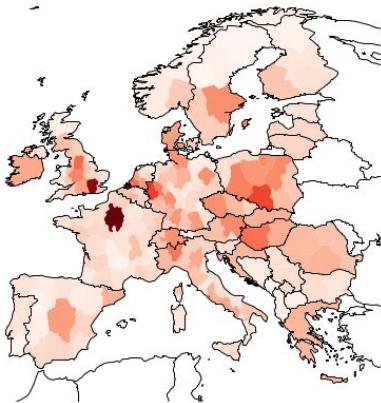
(e) oil-based product demand



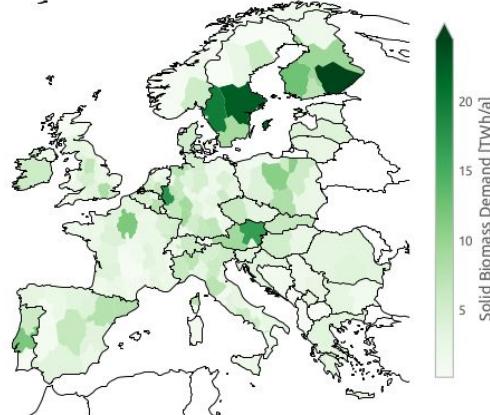
(c) methane demand



(d) heat demand



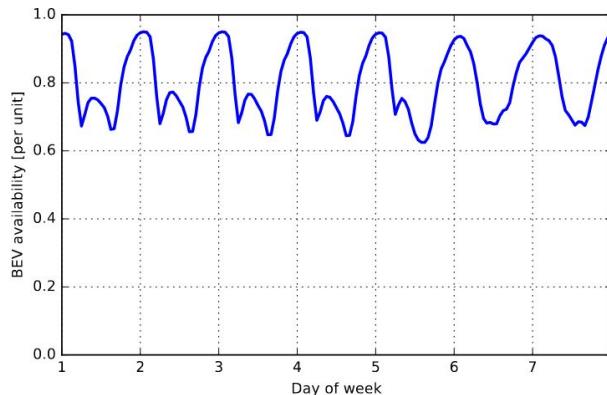
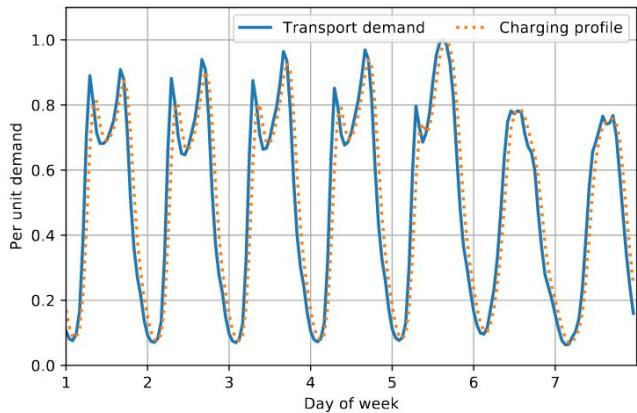
(f) solid biomass demand



Use of PyPSA components

- **Buses** impose energy balances for different energy carriers (heat, hydrogen, methane), and account for CO₂ emitted/absorbed from the atmosphere.
- **Links** represent **energy converters**, e.g. a heat pump that withdraws energy from the electricity bus and supplies heat to the heating bus with a certain efficiency.
- **Multilinks** represent more **elaborated conversion processes**, e.g. a biomass-fired CHP that withdraws biomass and supplies electricity, heat and CO₂ emissions.
- **Stores** can represent available potential (if allowed to discharge throughout the modelled period) or relax the hourly energy balancing constraint to annual (when a free store is added).

Transport - Electrification of land transport

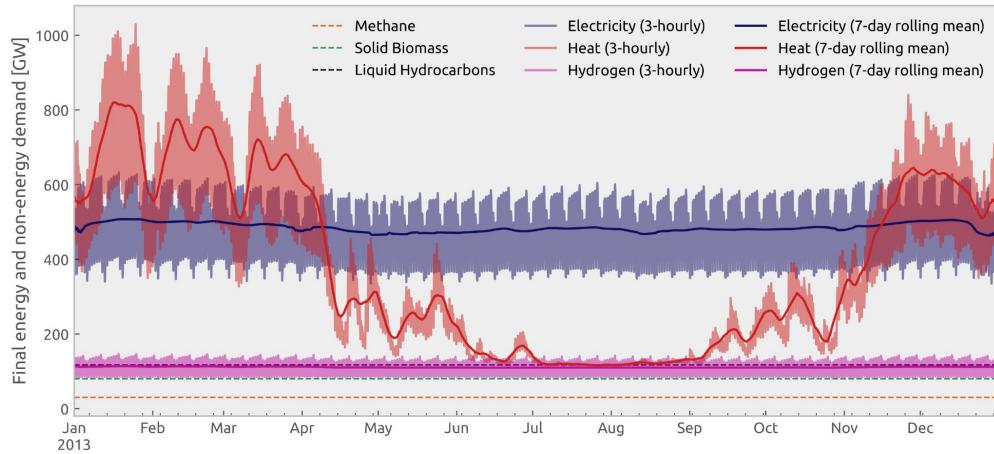


- Road and rail are **largely electrified**, ICE only during pathway to net-zero
- Time series from traffic counting
- Lumped representation of electric vehicles: a configurable share of them participates in demand-side management (**DSM**) or vehicle-to-grid (**V2G**)
- Electric vehicles have **time-varying availability** for charging depending on driving profiles
- EV battery almost fully charged in the

Modelling **land transport transformation** endogenously is currently under development!



Heating - Demand



Heating demand time series created by :

- space heating based on degree-day and daily profiles, weighted by population
- constant water heating demand
- scaled to historical annual demand
- split into individual and central systems (cooling demand already electrified)

Heating - Tech for individual & district heating

Decentral individual heating

can be supplied by:

- air- or ground-sourced heat pumps
- resistive heaters
- gas / oil / biomass / hydrogen boilers
- solar thermal
- small water tanks

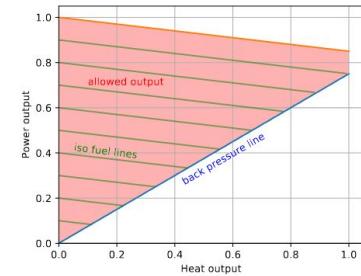
Building renovations can be co-optimised to reduce space heating demand.

District heating systems

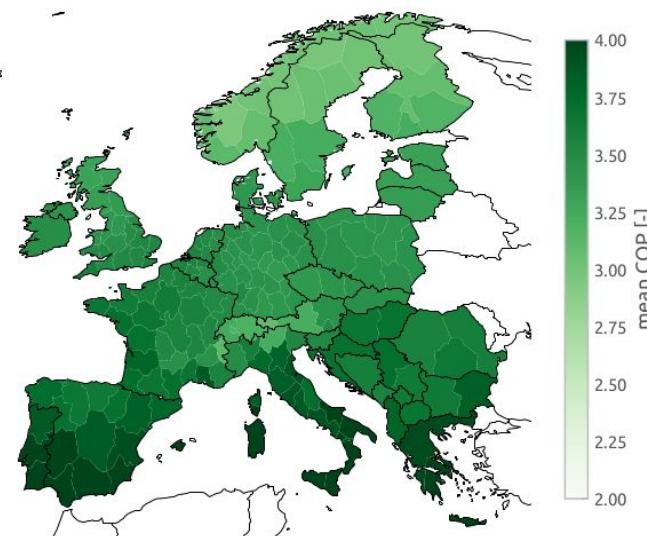
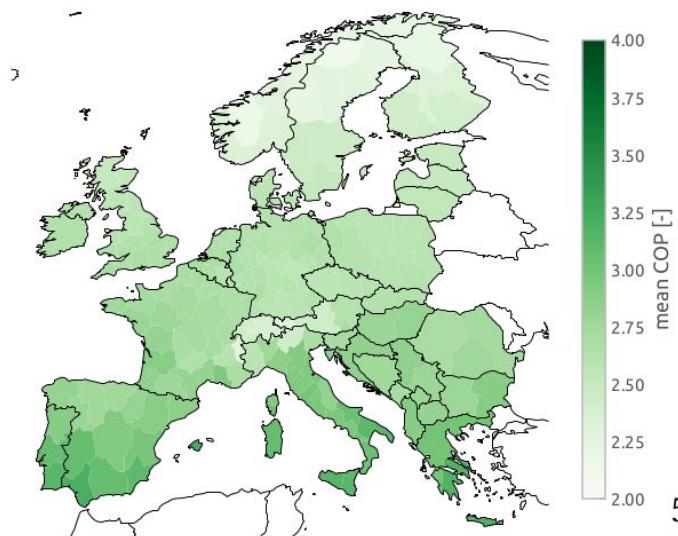
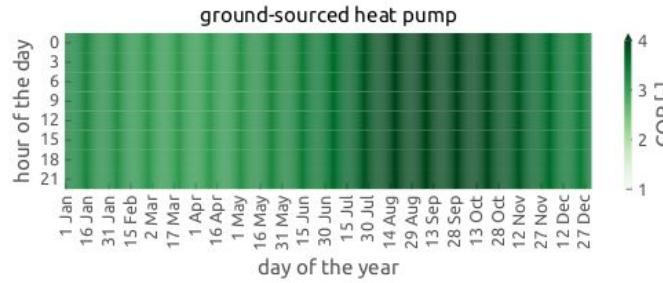
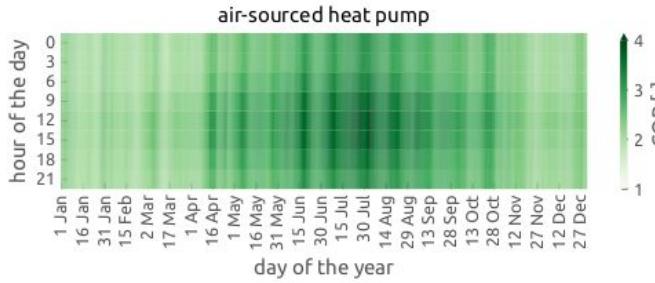
can be supplied in urban areas by:

- air-sourced heat pumps + other
- resistive heaters
- gas / hydrogen / biomass / waste CHPs
- gas / oil / biomass / hydrogen boilers
- solar thermal
- long-duration hot water storage
- waste heat from industrial processes

CHP feasible dispatch:

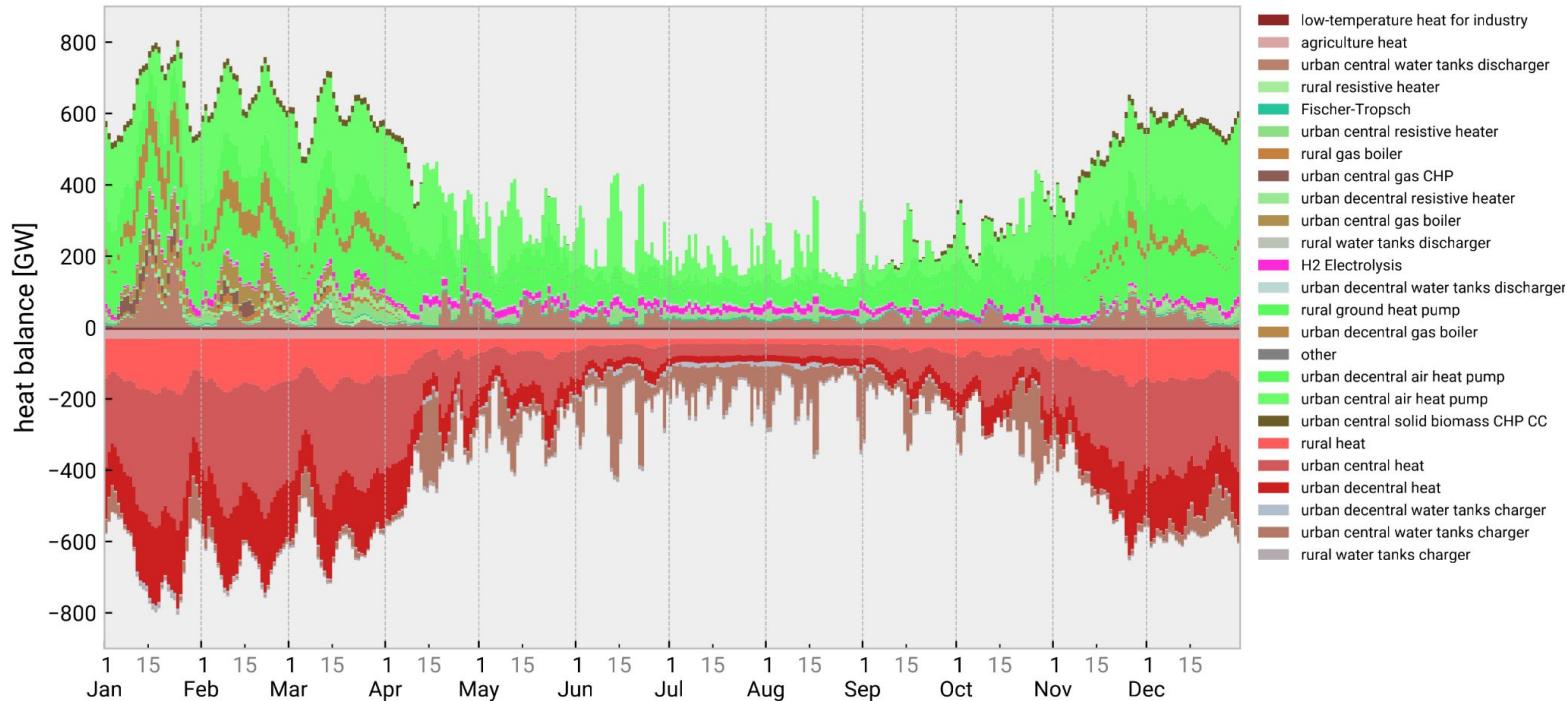


Heating - Heat pumps as new variable supply tech



Geothermal heat
sources have
been integrated
very recently!

Heating - Example daily heat system balance

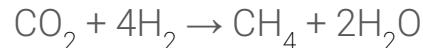


There are difficult periods in winter with **low** wind and solar, **high** space heating demand and **low** air temperatures, which are bad for air-sourced heat pump performance. In this case **gas boilers** and **CHP plants** jump in as backup.

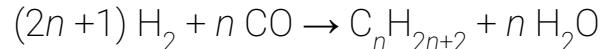
Synthetic fuels - Power-to-X

Carbonaceous fuels produced synthetically:

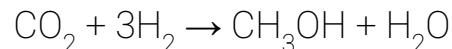
- Synthetic methane using Sabatier reaction



- Synthetic oil via Fischer-Tropsch



- Synthetic methanol in methanolisation plants

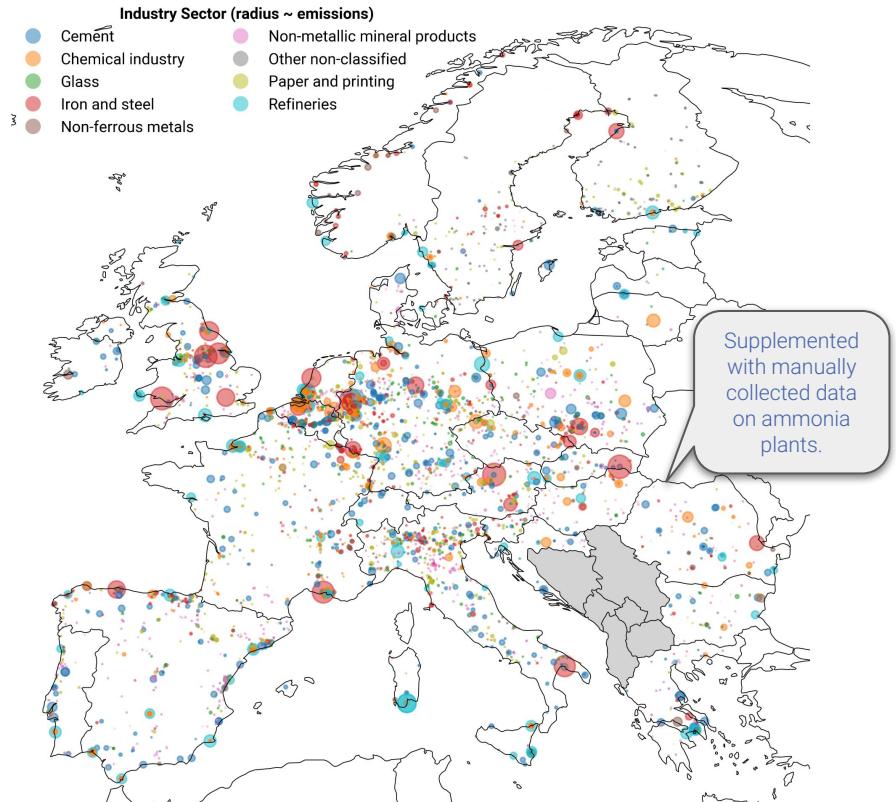


Conversion efficiencies read from open database of techno-economic assumptions

https://github.com/PyPSA/technology-data/blob/master/outputs/costs_2030.csv



Industry - Regionalisation based on Hotmaps



https://gitlab.com/hotmaps/industrial_sites/industrial_sites_Industrial_Database

Iron & Steel	Phase-out integrated steelworks; increased recycling; rest from H ₂ -DRI + EAF
Aluminium	Methane for high-enthalpy heat; increased recycling
Cement	Solid biomass; capture of CO ₂ emissions
Ceramics	Electrification
Ammonia	Gray, blue, green hydrogen
Plastics	Synthetic naphtha; MtO/MtA, increased recycling
Other industry	Electrification; process heat from biomass
Shipping	Methanol, (oil), (liquid hydrogen), (LNG)
Aviation	Kerosene from Fischer-Tropsch or methanol

Modelling **industry relocation, high-temperature heat source & shipping fuels** endogenously is currently under development!



Infrastructure - Gas network with H₂ retrofitting

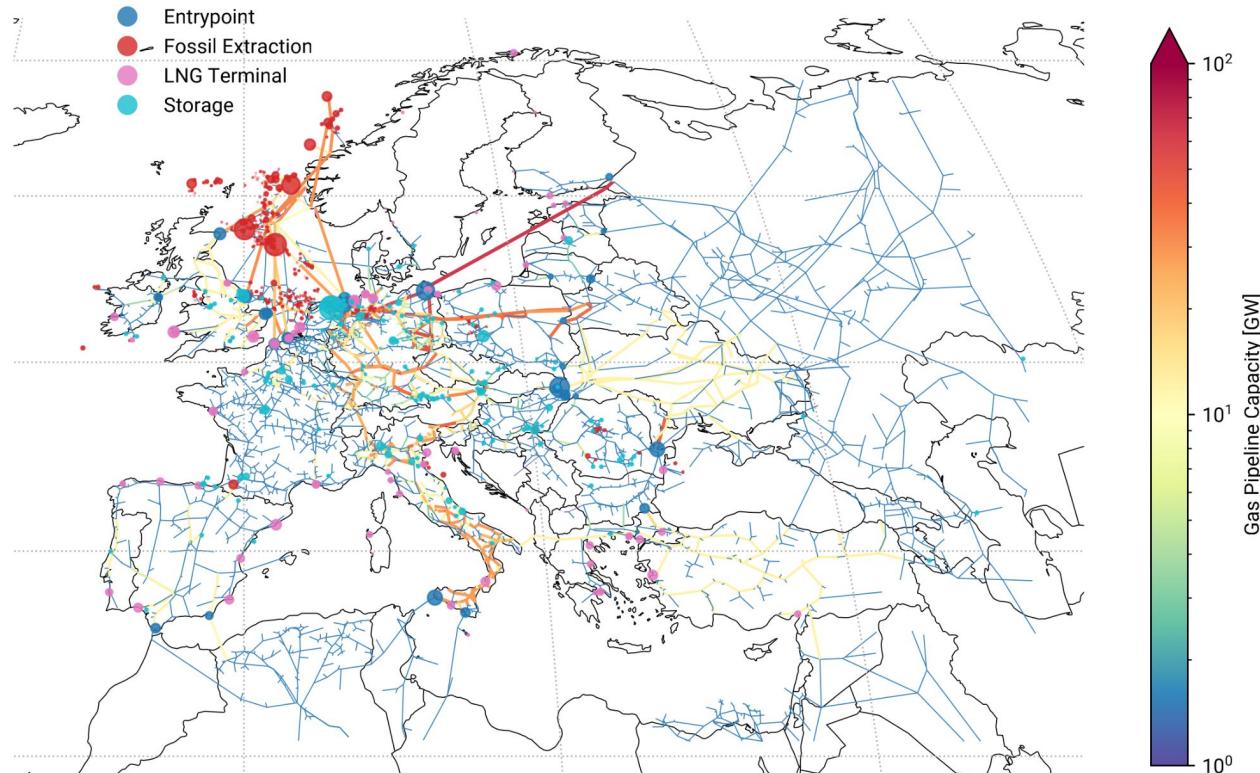
Compiled from open
SciGRID_gas dataset.

Fossil gas enters at **LNG terminals** or **gas fields**.

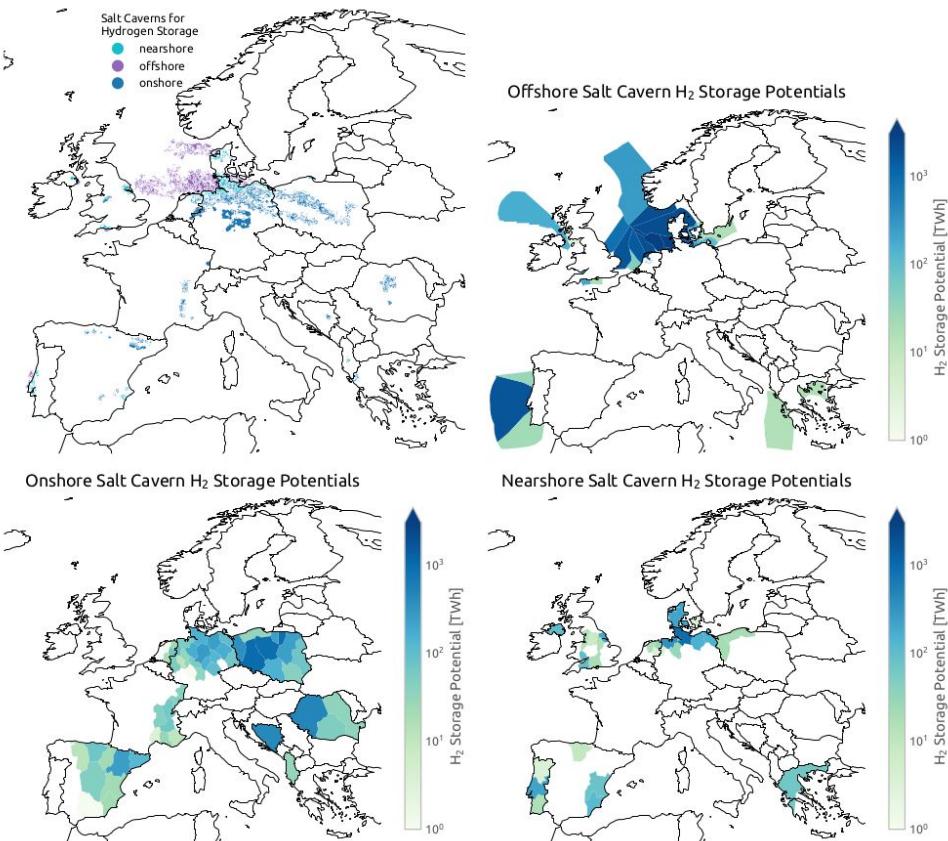
Gas flow **physics** and **valve control** neglected
transport model.

Electricity demand for
compression and **leakage**
configurable.

Pipelines can be **retrofitted**
to H₂ with costs from **EHB**.



Infrastructure - Hydrogen storage potentials



The regional distribution of **geological potential** to store hydrogen in **salt caverns** is considered.

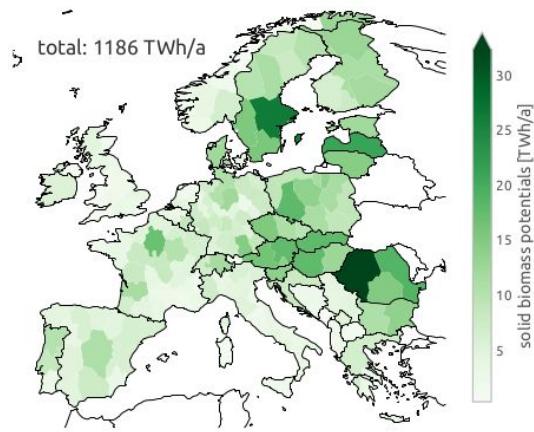
The user can **configure** if onshore and/or offshore potentials can be used.

Dilara Gulcin Caglayan, Nikolaus Weber, Heidi U. Heinrichs, Jochen Linßen, Martin Robinius, Peter A. Kukla, Detlef Stolten, *Technical potential of salt caverns for hydrogen storage in Europe*, **International Journal of Hydrogen Energy**, Volume 45, Issue 11, 2020, 6793-6805, <https://doi.org/10.1016/j.ijhydene.2019.12.161>

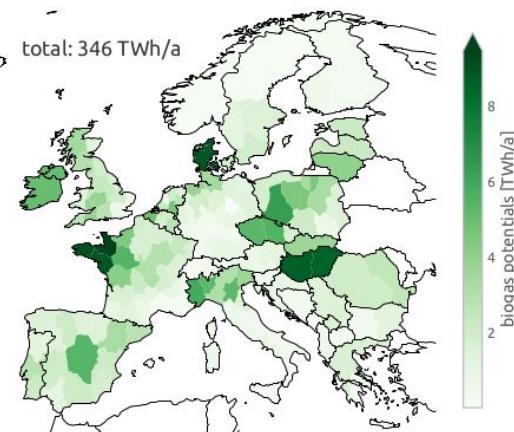


Infrastructure - Biomass from JRC ENSPRESO

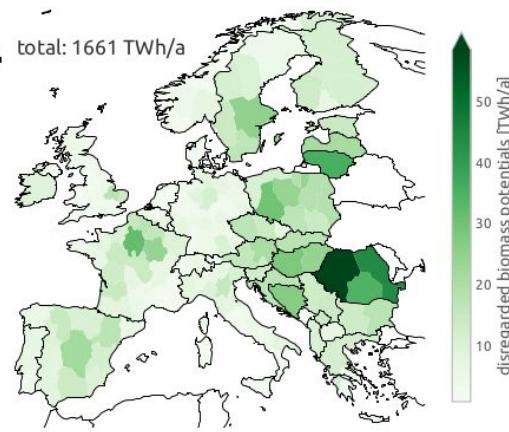
(a) solid biomass potentials



(b) biogas potentials



(c) potentials not included

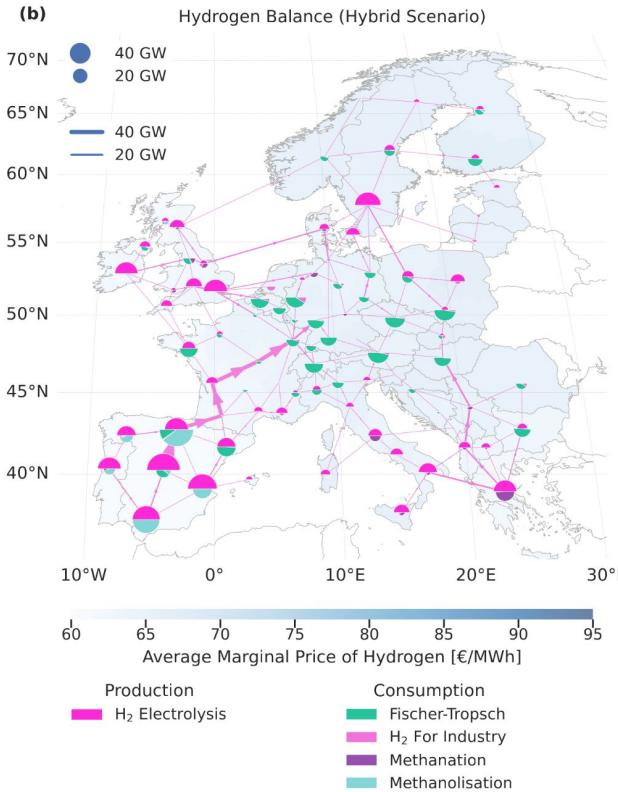
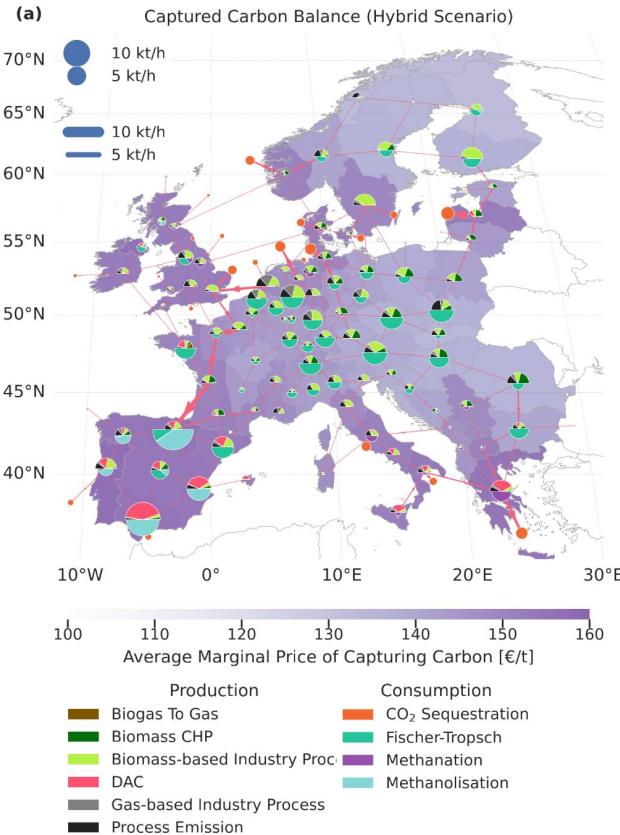


Biomass potentials are split between **solid biomass** and **biogas** (which can be, for instance, upgraded).

The user can configure low/medium/high potentials and what categories of biomass to consider (e.g. forest residues).

The default configuration only considers **residual biomass**, no energy crops.

Infrastructure - Carbon management

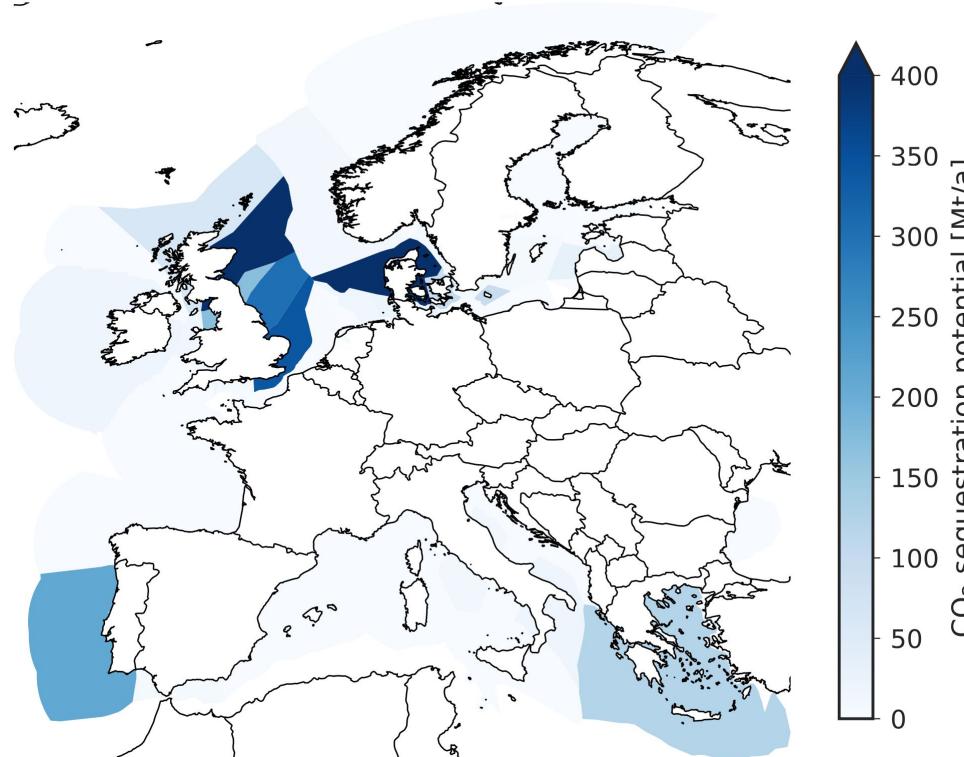


Built-in carbon flows:

- **Capture:**
DAC, process emissions, fossil / biomass CHP
- **Transport:**
CO₂ pipelines
- **Storage:**
intermediate storage and long-term geological sequestration
- **Utilization:**
for synthetic carbonaceous fuels

Infrastructure - Carbon sequestration potentials

Example: Offshore carbon sequestration potentials



The user can **configure**

- onshore/offshore sequestration,
- gas fields/oil fields/aquifer, and
- low/medium/high potentials,

as well as a **total limit** on the annual sequestration, e.g. 250 Mt per year.

Data source:

 **CO2Stop** – a project mapping both reserves and resources for CO₂ storage in Europe

By: Niels Posten (IEUS), Sam Holloway (BGS), Karen Kirk (BGS), Filip Neeli (TNO) and Nichols Ann Smith (BGS)

CO2Stop is the acronym for the "CO₂ Storage Potential in Europe" project. The CO2Stop project, which started in January 2012 and ended in December 2013, was funded by the European Commission (Project N°: ENER/CT/1154-2011-012/111000). CO2Stop has covered 26 countries in Europe. Further Latvia was covered by Estonia.

The results of the study are provided as a database of CO₂ storage locations throughout Europe, a Database of Geological Structures, a Database of Reservoirs and a Database of Injectors to compute storage capacities and injection rates (StoreP).

The database is now available on the website of the Joint Research Centre (JRC) of the European Commission, Petten, the Netherlands (<https://store.jrc.ec.europa.eu/european-co2-storage-database/>) and will be available as an interactive web-based application on the website of the Geological Data Infrastructure (EGDI) (www.eurogeogrid.eu/).

The CO2Stop database will be the first step towards a European Storage Atlas. Other databases are available for UK, Norway and the Nordic countries.

(www.co2stopuk.co.uk/home/index.html) and the Nordic CO₂ storage atlas is produced by NORCCS – the Nordic CCS Coordinating Centre (<http://www.nordicccslabutum.net/>). Spain has published Atlas of Geological Structures that uphold a CO₂ storage potential in Spain (ISBN: 978-84-7846-928-0 Mo 2014).

88 Countries covered by the project
27 Countries covered by the boundary project
project included 26 countries and 27
Latvia was covered by Estonia



Technology choices - endogenous vs. exogenous

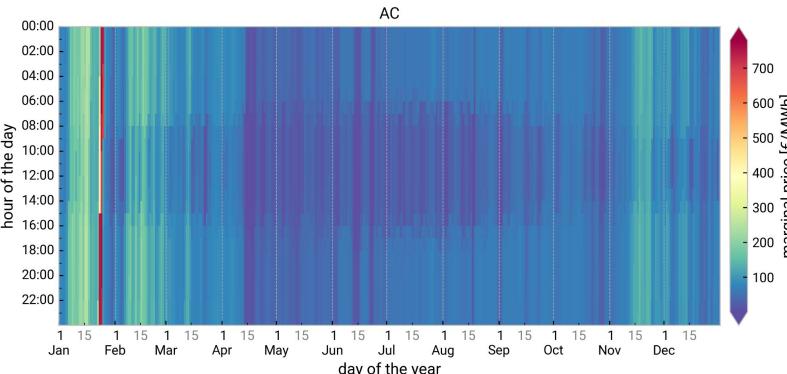
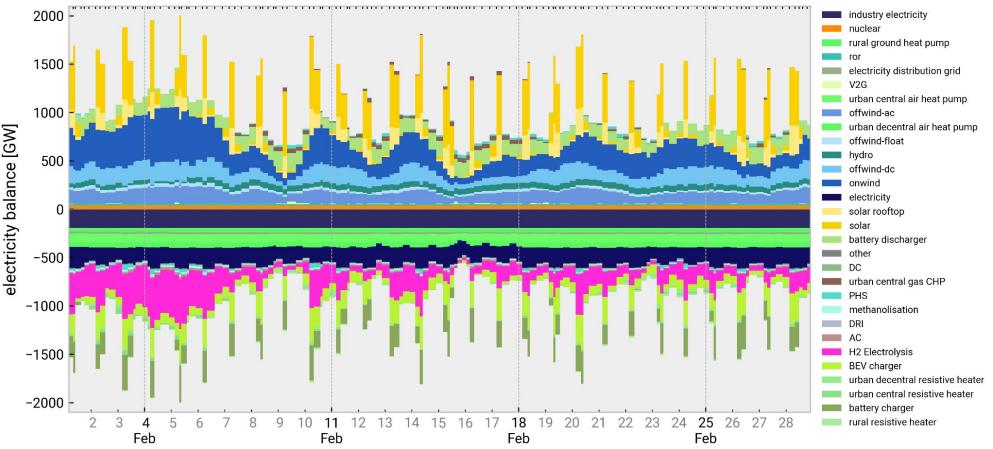
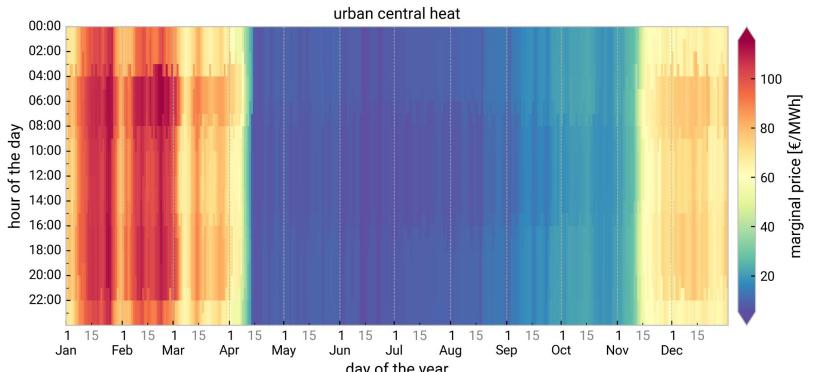
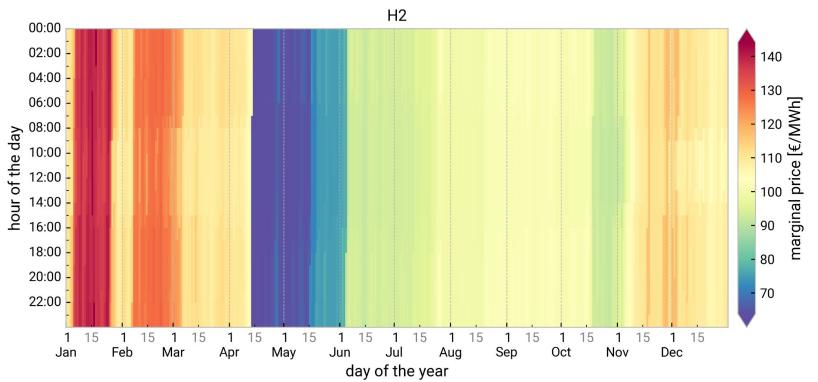
Exogenous assumptions (modeller chooses):

- energy services demand (e.g. heat)
- district heating shares
- energy carrier shares for road transport
- kerosene for aviation
- methanol for shipping
- electrification & recycling in industry
- steel production with DRI + EAF

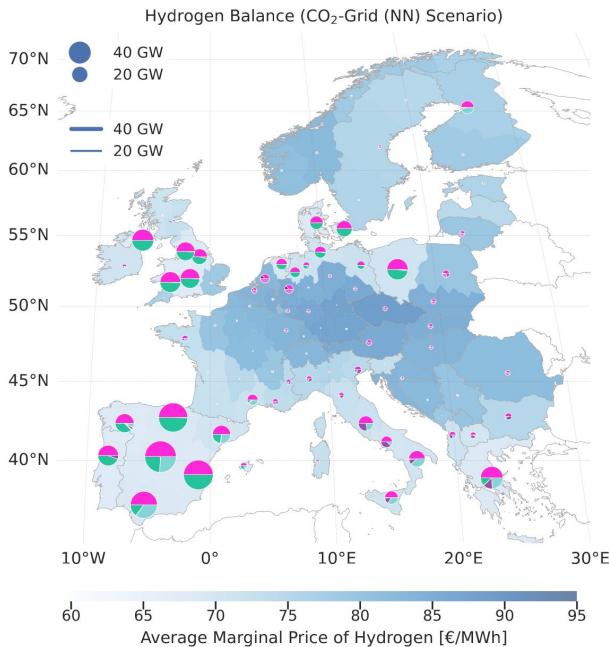
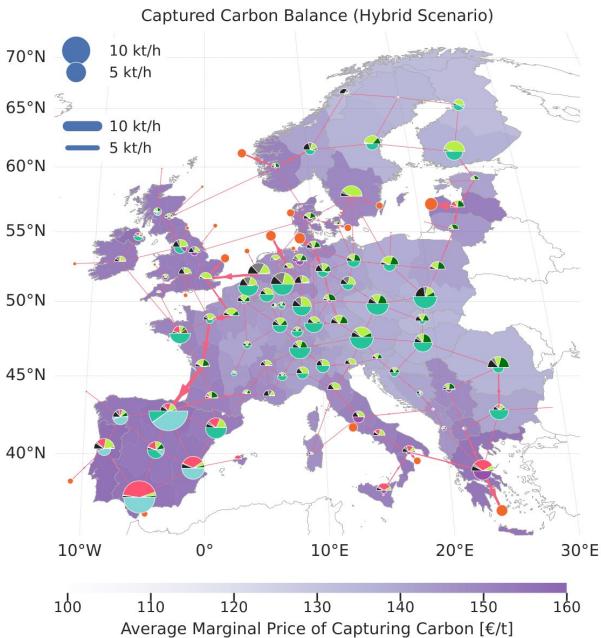
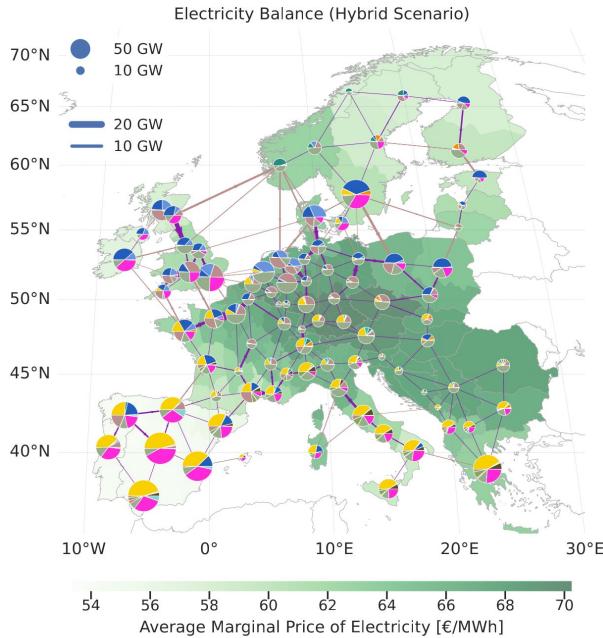
Endogenous choices (model optimizes):

- change in electricity generation fleet
- transmission reinforcement
- capacities and locations of short and long-duration energy storage
- space and water heating technologies (including building renovations)
- all P2G/L/H/C
- supply of process heat for industry
- carbon capture (e.g. CHP, industry, DAC)

Prices in the model



Prices in the model

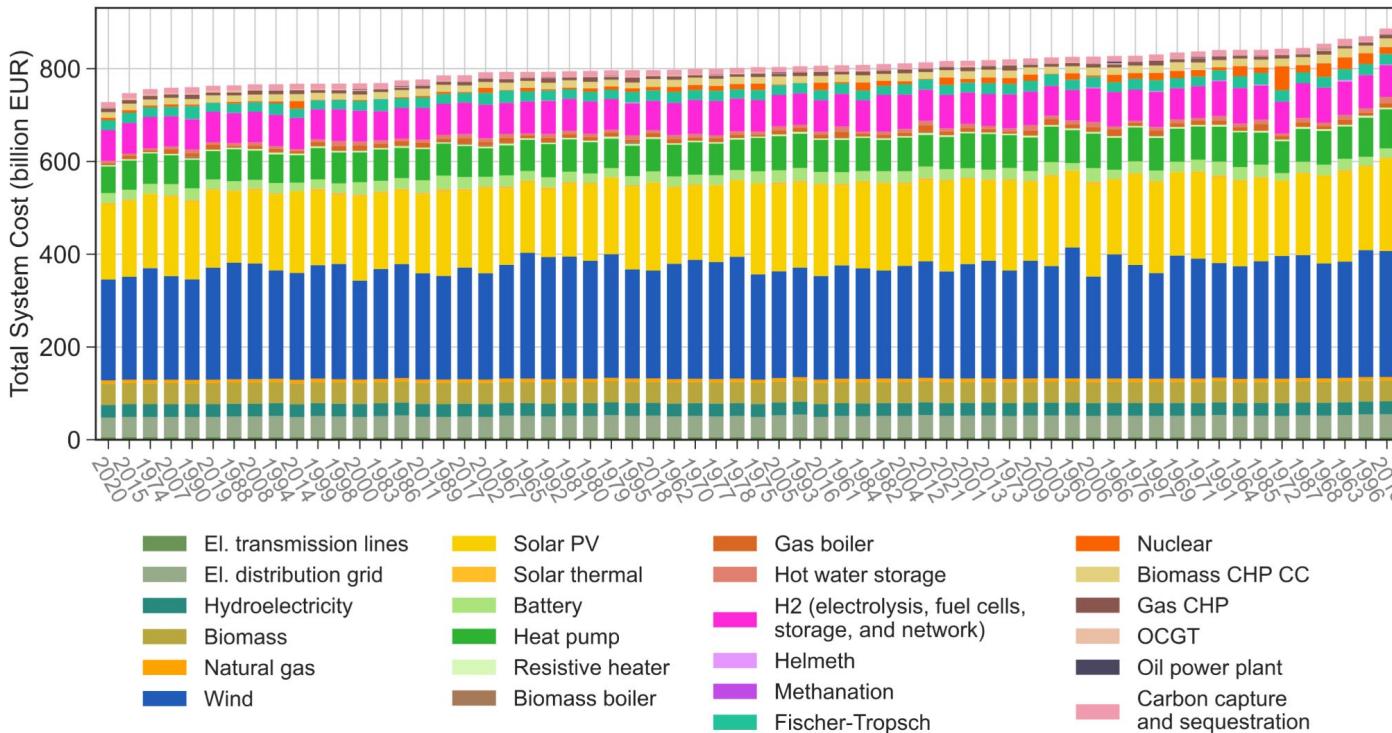


Supply, consumption and storage options by carrier

Supply, consumption and storage options by carrier

Heat (115 regions)		CO2 atmosphere (not spatially resolved)		CO2 commodity (not spatially resolved)					
Supply	Withdrawal	Supply	Withdrawal	Supply	Withdrawal				
air-sourced heat pump	residential heat	kerosene for aviation	solid biomass for industry (w CC)	direct air capture	Fischer-Tropsch				
ground-sourced heat pump (only rural)	services heat	diesel for agriculture	solid biomass CHP (w CC)	biogas upgrading (w CC)	methanolisation				
resistive heater	agriculture heat	methanol for shipping	biogas upgrading (w CC)	gas CHP (w CC)	sequestration				
gas boiler	low-T industry heat	methanol for industry	direct air capture	biomass CHP (w CC)	Sabatier				
biomass boiler	direct air capture	naphtha for industry	electrobiofuels	steam methane reforming (w CC)					
solar thermal	water tank charger	gas boiler		process emissions (w CC)					
water tank discharger		gas CHP (w/wo CC)		solid biomass for industry (w CC)					
biomass CHP (w/wo CC, only DH)		gas turbine (OCGT)		gas for high-T industry heat (w CC)					
gas CHP (w/wo CC, only DH)		methanol turbine (OCGT)							
hydrogen fuel cell CHP (only DH)		process emissions (w/wo CC)							
electrolysis (only DH)		fossil oil refining							
Haber-Bosch (only DH)		gas for high-T industry heat (w/wo CC)							
Sabatier (only DH)		steam methane reforming (w/wo CC)							
Fischer-Tropsch (only DH)									
methanolisation (only DH)									
Storage		long-duration thermal storage (only DH) hot water tank							
		intermediate storage in steel tank long-term geological sequestration							

PyPSA-Eur can be run on different **weather years**!



The years **2010, 2013, 2019** and **2023** are currently available "out of the box".

Other years **1940-2024** require a few more steps.

We are planning to expand the number of "plug-and-play" years.

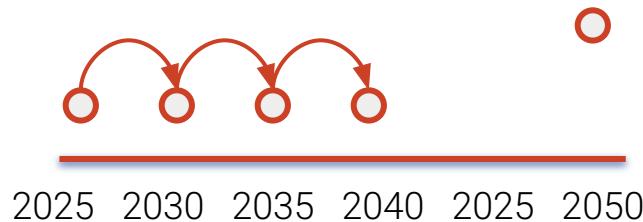
Overnight and pathway optimisation

- **Overnight** optimisation builds the complete system in one go. Already existing capacities can be included (**brownfield**) or the system can be built from scratch (**greenfield**)
- The choice of **investment years** is arbitrary.
- **Pathway optimisation with myopic foresight** (next slide). The CO₂ emission **reduction path** is exogenous.
- **Perfect foresight pathway planning** is currently experimental (i.e. use the CO₂ budget endogenously).

Overnight

Myopic pathway

Perfect-foresight pathway



Myopic pathway optimization

- The CO₂ emission **reduction path** is exogenous.
- Optimise **start network** for e.g. 2025, starting with existing energy infrastructure.
- Take results from **2025 as input** for 2030 infrastructure optimisation, take 2030 results for next iteration, etc.
- Infrastructure decommissioned when reaching end of lifetime
- Configurable social discount rate and number of planning horizons



Scenario management

PyPSA-Eur has integrated & scalable scenario management.

config/config.yaml

```
run:
  name: all
  scenarios:
    enable: true

scenario:
  clusters: [90]

sector:
  H2_network: true
  gas_network: true
  H2_retrofit: true

electricity:
  transmission_limit: vopt
```

With these two files configured, run:

\$ snakemake all -n

and

\$ snakemake all

config/scenarios.yaml

```
no-h2-network:
  sector:
    H2_network: false

no-grid-expansion:
  electricity:
    transmission_limit: v1.0

no-to-both:
  sector:
    H2_network: false
  electricity:
    transmission_limit: v1.0

yes-to-both:
  sector:
    H2_network: true
  electricity:
    transmission_limit: vopt
```

Live Demo – very similar to electricity-only case

Start with a dry-run:

```
$ snakemake -call --configfile config/test/config.overnight.yaml -n
```

Then execute the same command “for real” by dropping “-n” flag:

```
$ snakemake -call all --configfile config/test/config.overnight.yaml
```

And for myopic pathway optimisation:

```
$ snakemake -call all --configfile config/test/config.myopic.yaml
```

To explore results, start a Jupyter notebook:

```
$ jupyter notebook
```

Practical Phase

(sector-coupled)

1) Run PyPSA-Eur sector-coupling tutorial with `snakemake`

Guide:

https://pypsa-eur.readthedocs.io/en/latest/tutorial_sector.html

```
snakemake -call all --configfile config/test/config.overnight.yaml
```

2) Explore CSV files and images in `results` directory.

Go to <https://pypsa-eur.readthedocs.io/en/latest/configuration.html> and try to find out how to configure some of the settings for **sector-coupled models** listed below:

1. Disable vehicle-to-grid discharging.
2. Disable hydrogen network expansion.
3. Increase the carbon sequestration potential to 800 Mt/a.
4. Allow hydrogen underground storage also onshore.
5. Reduce the primary production of plastics by increasing recycling rates.