PyPSA-Eur-Sec Documentation

Release 0.6.0

author

Nov 24, 2021

Getting Started

1 Documentation 3

2 Warnings 21

3 Licence 23

repo size 5.14 MB

docs passing

repo size 5.14 MB

repo size 5.14 MB

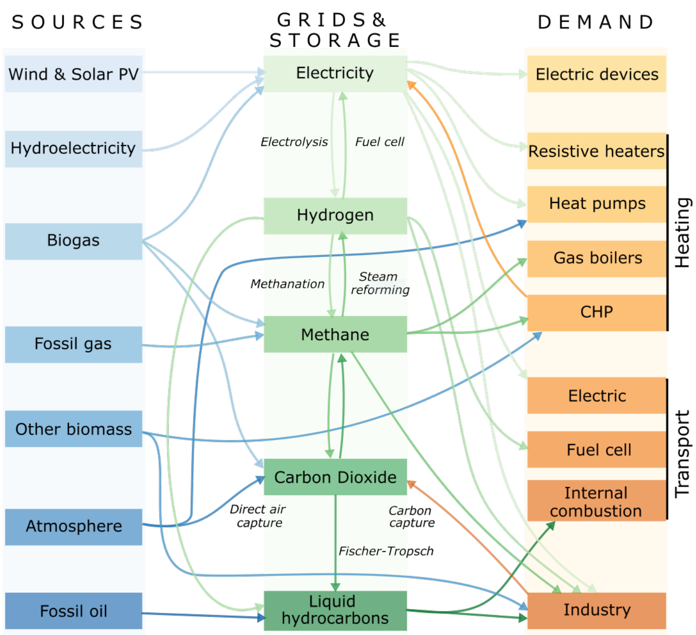
chat on gitter

PyPSA-Eur-Sec is an open model dataset of the European energy system at the transmission network level that covers the full ENTSO-E area.

PyPSA-Eur-Sec builds on the electricity generation and transmission model PyPSA-Eur to add demand and supply for the following sectors: transport, space and water heating, biomass, industry and industrial feedstocks. This completes the energy system and includes all greenhouse gas emitters except waste management, agriculture, forestry and land use.

Note: More about the current model capabilities and preliminary results can be found in a recent presentation at EMP-E and the the following preprint with a description of the industry sector.

This diagram gives an overview of the sectors and the links between them:



PyPSA-Eur-Sec was initially based on the model PyPSA-Eur-Sec-30 described in the paper Synergies of sector coupling and transmission reinforcement in a cost-optimised, highly renewable European energy system (2018) but it differs by being based on the higher resolution electricity transmission model PyPSA-Eur rather than a one-node-per-country model, and by including biomass, industry, industrial feedstocks, aviation, shipping, better carbon management, carbon capture and usage/sequestration, and gas networks.

PyPSA-Eur-Sec includes PyPSA-Eur as a snakemake subworkflow. PyPSA-Eur-Sec uses PyPSA-Eur to build the clustered transmission model along with wind, solar PV and hydroelectricity potentials and time series. Then PyPSA-Eur-Sec adds other conventional generators, storage units and the additional sectors.

Currently the scripts to solve and process the resulting PyPSA models are also included in PyPSA-Eur-Sec, although they could in future be better integrated with the corresponding scripts in PyPSA-Eur. A stumbling block to sharing solve_network.py between PyPSA-Eur and PyPSA-Eur-Sec is the different extra_functionality required to build storage and CHP constraints.

PyPSA-Eur-Sec is designed to be imported into the open toolbox PyPSA for which documentation is available as well.

This project is currently maintained by the Department of Digital Transformation in Energy Systems at the Technical University of Berlin. Previous versions were developed by the Energy System Modelling group at the Institute for Automation and Applied Informatics at the Karlsruhe Institute of Technology which was funded by the Helmholtz Association, and by the Renewable Energy Group at FIAS to carry out simulations for the CoNDyNet project, financed by the German Federal Ministry for Education and Research (BMBF) as part of the Stromnetze Research Initiative.

CHAPTER 1

Documentation

Getting Started

- *Installation*

1.1 Installation

The subsequently described installation steps are demonstrated as shell commands, where the path before the `%` sign denotes the directory in which the commands following the `%` should be entered.

1.1.1 Install PyPSA-Eur and its data

First install PyPSA-Eur and all its dependencies. Clone the repository:

```
projects % git clone https://github.com/PyPSA/pypsa-eur.git
```

then download and unpack all the PyPSA-Eur data files by running the following snakemake rule:

```
projects/pypsa-eur % snakemake -j 1 retrieve_databundle
```

1.1.2 Clone technology-data repository

Next install the technology assumptions database technology-data by creating a parallel directory:

```
projects % git clone https://github.com/PyPSA/technology-data.git
```

1.1.3 Clone PyPSA-Eur-Sec repository

Create a parallel directory for PyPSA-Eur-Sec with:

```
projects % git clone https://github.com/PyPSA/pypsa-eur-sec.git
```

1.1.4 Environment/package requirements

The requirements are the same as PyPSA-Eur. For `solve_network.py` in addition you need `gurobipy`. If you have xarray version >= 0.15.1, you will need the latest master branch of atlite version 0.0.2.

You can create an enviroment using the environment.yaml file in pypsa-eur/envs:

```
.../pypsa-eur % conda env create -f envs/environment.yaml

.../pypsa-eur % conda activate pypsa-eur
```

See details in PyPSA-Eur Installation

1.1.5 Data requirements

Small data files are included directly in the git repository, while larger ones are archived in a data bundle on zenodo (10.5281/zenodo.5546517). The data bundle's size is around 640 MB.

To download and extract the data bundle on the command line:

```
projects/pypsa-eur-sec/data % wget "https://zenodo.org/record/5546517/files/pypsa-eur-
↪sec-data-bundle.tar.gz"
projects/pypsa-eur-sec/data % tar -xvzf pypsa-eur-sec-data-bundle.tar.gz
```

The data licences and sources are given in the following table.

description	file/folder	licence	source
JRC IDEES database	jrc-idees-2015/	CC BY 4.0	https://ec.europa.eu/jrc/en/potencia/jrc-idees
urban/rural fraction	urban_percent.csv	unknown	unknown
JRC biomass potentials	biomass/	unknown	https://doi.org/10.2790/39014
JRC ENSPRESO biomass potentials	remote	CC BY 4.0	https://data.jrc.ec.europa.eu/dataset/74ed5a04-7d74-4807-9eab-b94774309d9f
EEA emission statistics	eea/UNFCCC_v23.csv	EEA standard re-use policy	https://www.eea.europa.eu/data-and-maps/data/national-emissions-reported-to-the-unfccc-and-to-the-eu-greenhouse-ga
Eurostat Energy Balances	eurostat-energy_balances-*/	Eurostat	https://ec.europa.eu/eurostat/web/energy/data/energy-balances
Swiss energy statistics from Swiss Federal Office of Energy	switzerland-sfoe/	unknown	http://www.bfe.admin.ch/themen/00526/00541/00542/02167/index.html?dossier_id=02169
BASt emobility statistics	emobility/	unknown	http://www.bast.de/DE/Verkehrstechnik/Fachthemen/v2-verkehrszaehlung/Stundenwerte.html?nn=626916
BDEW heating profile	heat_load_profile_BDEW.csv	unknown	https://github.com/oemof/demandlib
heating profiles for Aarhus	heat_load_profile_DK_AdamJensen.csv	unknown	Adam Jensen MA thesis at Aarhus University
George Lavidas wind/wave costs	WindWaveWEC_GLTB.xlsx	unknown	George Lavidas
country codes	Country_codes.csv	CC BY 4.0	Marta Victoria
co2 budgets	co2_budget.csv	CC BY 4.0	https://arxiv.org/abs/2004.11009
existing heating potentials	existing_infrastructure/existing_heating_raw.csv	unknown	https://ec.europa.eu/energy/studies/mapping-and-analyses-current-and-future-2020-2030-heatingcooling-fu en?redir=1
IRENA existing VRE capacities	existing_infrastructure/{solar|onwind|offwind}_capcity_IRENA.csv	unknown	https://www.irena.org/Statistics/Download-Data
USGS ammonia production	myb1-2017-nitro.xls	unknown	https://www.usgs.gov/centers/nmic/nitrogen-statistics-and-information
hydrogen salt cavern potentials	hydrogen_salt_cavern_potentials.csv	CC BY 4.0	https://doi.org/10.1016/j.ijhydene.2019.12.161
hotmaps industrial site database	Industrial_Database.csv	CC BY 4.0	https://gitlab.com/hotmaps/industrial_sites/industrial_sites_Industrial_Database
Hotmaps building stock data	data_building_stock.csv	CC BY 4.0	https://gitlab.com/hotmaps/building-stock
U-values Poland	u_values_poland.csv	unknown	https://data.europa.eu/euodp/de/data/dataset/building-stock-observatory
Floor area missing in hotmaps building stock data	floor_area_missing.csv	unknown	https://data.europa.eu/euodp/de/data/dataset/building-stock-observatory
Comparative level investment	comparative_level_investment.csv	Eurostat	https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Comparative_price_levels_for_investment
Electricity taxes	electricity_taxes_eu.csv	Eurostat	https://appsso.eurostat.ec.europa.eu/nui/show.do?dataset=nrg_pc_204&lang=en
Building topologies and corresponding standard values	tabula-calculator-calcsetbuilding.csv	unknown	https://episcope.eu/fileadmin/tabula/public/calc/tabula-calculator.xlsx
Retrofitting thermal	retro_cost_germany.csv	unkown	https://www.iwu.de/forschung/handlungslogiken/

1.1.6 Set up the default configuration

First make your own copy of the `config.yaml` based on `config.default.yaml`. For example:

```
projects/pypsa-eur-sec % cp config.default.yaml config.yaml
```

1.1.7 Getting started

In `config.yaml` you can control the settings for the scenarios you want to run, such as the number of nodes, the CO2 limit, the installable potentials for solar and wind, which technologies are activated, etc.

To run the full optimization with your settings:

```
projects/pypsa-eur-sec % snakemake -j1
```

Warning: you may need a computer cluster for this (with e.g. 10-100 GB of RAM and several processors).

To only prepare the networks, you can run the scripts up to the point before optimization:

```
projects/pypsa-eur-sec % snakemake -j1 prepare_sector_networks
```

Implementation details

- *Spatial resolution*
- *Supply and demand*

1.2 Spatial resolution

The default nodal resolution of the model follows the electricity generation and transmission model PyPSA-Eur, which clusters down the electricity transmission substations in each European country based on the k-means algorithm. This gives nodes which correspond to major load and generation centres (typically cities).

The total number of nodes for Europe is set in the `config.yaml` file under `clusters`. The number of nodes can vary between 37, the number of independent countries / synchronous areas, and several hundred. With 200-300 nodes the model needs 100-150 GB RAM to solve with a commerical solver like Gurobi.

Not all of the sectors are at the full nodal resolution, and some demand for some sectors is distributed to nodes using heuristics that need to be corrected. Some networks are copper-plated to reduce computational times.

For example:

Electricity network: nodal.

Electricity residential and commercial demand: nodal, distributed in each country based on population and GDP.

Electricity demand in industry: based on the location of industrial facilities from HotMaps database.

Building heating demand: nodal, distributed in each country based on population.

Industry demand: nodal, distributed in each country based on locations of industry from HotMaps database.

Hydrogen network: nodal.

Methane network: single node for Europe, since future demand is so low and no bottlenecks are expected.

Solid biomass: choice between single node for Europe and nodal where biomass potential is regionally disaggregated (currently given per country, then distributed by population density within) and transport of solid biomass is possible.

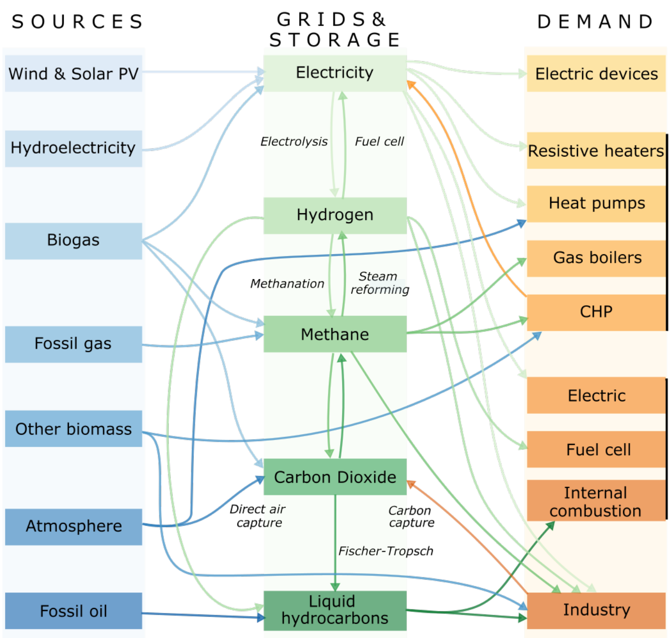
CO2: single node for Europe, but a transport and storage cost is added for sequestered CO2. Optionally: nodal, with CO2 transport via pipelines.

Liquid hydrocarbons: single node for Europe, since transport costs for liquids are low.

1.3 Supply and demand

An initial orientation to the supply and demand options in the model PyPSA-Eur-Sec can be found in the description of the model PyPSA-Eur-Sec-30 in the paper Synergies of sector coupling and transmission reinforcement in a cost-optimised, highly renewable European energy system (2018). The latest version of PyPSA-Eur-Sec differs by including biomass, industry, industrial feedstocks, aviation, shipping, better carbon management, carbon capture and usage/sequestration, and gas networks.

The basic supply (left column) and demand (right column) options in the model are described in this figure:



1.3.1 Electricity supply and demand

Electricity supply and demand follows the electricity generation and transmission model PyPSA-Eur, except that hydrogen storage is integrated into the hydrogen supply, demand and network, and PyPSA-Eur-Sec includes CHPs.

Unlike PyPSA-Eur, PyPSA-Eur-Sec does not distribution electricity demand for industry according to population and GDP, but uses the geographical data from the Hotmaps Industrial Database.

Also unlike PyPSA-Eur, PyPSA-Eur-Sec subtracts existing electrified heating from the existing electricity demand, so that power-to-heat can be optimised separately.

The remaining electricity demand for households and services is distributed inside each country proportional to GDP and population.

1.3.2 Heat demand

Heat demand is split into:

- `urban central`: large-scale district heating networks in urban areas with dense heat demand
- `residential/services urban decentral`: heating for individual buildings in urban areas
- `residential/services rural`: heating for individual buildings in rural areas, agriculture heat uses

1.3.3 Heat supply

Oil and gas boilers

Heat pumps

Either air-to-water or ground-to-water heat pumps are implemented.

They have coefficient of performance (COP) based on either the external air or the soil hourly temperature.

Ground-source heat pumps are only allowed in rural areas because of space constraints.

Only air-source heat pumps are allowed in urban areas. This is a conservative assumption, since there are many possible sources of low-temperature heat that could be tapped in cities (waste water, rivers, lakes, seas, etc.).

Resistive heaters

Large Combined Heat and Power (CHP) plants

A good summary of CHP options that can be implemented in PyPSA can be found in the paper Cost sensitivity of optimal sector-coupled district heating production systems.

PyPSA-Eur-Sec includes CHP plants fuelled by methane, hydrogen and solid biomass from waste and residues.

Hydrogen CHPs are fuel cells.

Methane and biomass CHPs are based on back pressure plants operating with a fixed ratio of electricity to heat output. The methane CHP is modelled on the Danish Energy Agency (DEA) "Gas turbine simple cycle (large)" while the solid biomass CHP is based on the DEA's "09b Wood Pellets Medium".

The efficiencies of each are given on the back pressure line, where the back pressure coefficient `c_b` is the electricity output divided by the heat output. The plants are not allowed to deviate from the back pressure line and are implement as `Link` objects with a fixed ratio of heat to electricity output.

NB: The old PyPSA-Eur-Sec-30 model assumed an extraction plant (like the DEA coal CHP) for gas which has flexible production of heat and electricity within the feasibility diagram of Figure 4 in the Synergies paper. We have switched to the DEA back pressure plants since these are more common for smaller plants for biomass, and because the extraction plants were on the back pressure line for 99.5% of the time anyway. The plants were all changed to back pressure in PyPSA-Eur-Sec v0.4.0.

Micro-CHP for individual buildings

Optional.

Waste heat from Fuel Cells, Methanation and Fischer-Tropsch plants

Solar thermal collectors

Thermal energy storage using hot water tanks

Small for decentral applications.

Big water pit storage for district heating.

1.3.4 Retrofitting of the thermal envelope of buildings

Co-optimising building renovation is only enabled if in the `config.yaml` the option `retro_endogen: True`. To reduce the computational burden default setting is

```
existing_capacities:
  grouping_years: [1980, 1985, 1990, 1995, 2000, 2005, 2010, 2015, 2019]
```

Renovation of the thermal envelope reduces the space heating demand and is optimised at each node for every heat bus. Renovation measures through additional insulation material and replacement of energy inefficient windows are considered.

In a first step, costs per energy savings are estimated in `build_retro_cost.py`. They depend on the insulation condition of the building stock and costs for renovation of the building elements. In a second step, for those cost per energy savings two possible renovation strengths are determined: a moderate renovation with lower costs and lower maximum possible space heat savings, and an ambitious renovation with associated higher costs and higher efficiency gains. They are added by step-wise linearisation in form of two additional generations in `prepare_sector_network.py`.

Settings in the config.yaml concerning the endogenously optimisation of building renovation

```
  threshold_capacity: 10
  conventional_carriers:
    - lignite
    - coal
    - oil
```

Further information are given in the publication

Mitigating heat demand peaks in buildings in a highly renewable European energy system, (2021).

1.3.5 Hydrogen demand

Stationary fuel cell CHP.

Transport applications.

Industry (ammonia, precursor to hydrocarbons for chemicals and iron/steel).

1.3.6 Hydrogen supply

Steam Methane Reforming (SMR), SMR+CCS, electrolysers.

1.3.7 Methane demand

Can be used in boilers, in CHPs, in industry for high temperature heat, in OCGT.

Not used in transport because of engine slippage.

1.3.8 Methane supply

Fossil, biogas, Sabatier (hydrogen to methane), HELMETH (directly power to methane with efficient heat integration).

1.3.9 Solid biomass demand

Solid biomass provides process heat up to 500 Celsius in industry, as well as feeding CHP plants in district heating networks.

1.3.10 Solid biomass supply

Only wastes and residues from the JRC ENSPRESO biomass dataset.

1.3.11 Oil product demand

Transport fuels, agriculture machinery and naphtha as a feedstock for the chemicals industry.

1.3.12 Oil product supply

Fossil or Fischer-Tropsch.

1.3.13 Industry demand

Based on materials demand from JRC-IDEES and other sources such as the USGS for ammonia.

Industry is split into many sectors, including iron and steel, ammonia, other basic chemicals, cement, non-metalic minerals, alumuninium, other non-ferrous metals, pulp, paper and printing, food, beverages and tobacco, and other more minor sectors.

Inside each country the industrial demand is distributed using the Hotmaps Industrial Database.

1.3.14 Industry supply

Process switching (e.g. from blast furnaces to direct reduction and electric arc furnaces for steel) is defined exogenously.

Fuel switching for process heat is mostly also done exogenously.

Solid biomass is used for up to 500 Celsius, mostly in paper and pulp and food and beverages.

Higher temperatures are met with methane.

1.3.15 Carbon dioxide capture, usage and sequestration (CCU/S)

Carbon dioxide can be captured from industry process emissions, emissions related to industry process heat, combined heat and power plants, and directly from the air (DAC).

Carbon dioxide can be used as an input for methanation and Fischer-Tropsch fuels, or it can be sequestered underground.

Foresight options

- *Overnight (greenfield) scenarios*
- *Myopic transition path*

1.4 Overnight (greenfield) scenarios

The default is to calculate a rebuilding of the energy system to meet demand, a so-called overnight or greenfield approach.

For this, use `foresight : 'overnight'` in `config.yaml`, like the example in `config.default.yaml`.

In this case, the `planning_horizons : [2030]` scenario parameter can be set to use the year from which cost and other technology assumptions are set (forecasts for 2030 in this case).

1.5 Myopic transition path

The myopic code can be used to investigate progressive changes in a network, for instance, those taking place throughout a transition path. The capacities installed in a certain time step are maintained in the network until their operational lifetime expires.

The myopic approach was initially developed and used in the paper Early decarbonisation of the European Energy system pays off (2020) but the current implementation complies with the pypsa-eur-sec standard working flow and is compatible with using the higher resolution electricity transmission model PyPSA-Eur rather than a one-node-per-country model.

The current code applies the myopic approach to generators, storage technologies and links in the power sector and the space and water heating sector.

The transport sector and industry are not affected by the myopic code. In essence, the electrification of road and rail transport, the percentage of electric vehicles that allow demand-side management and vehicle-to-grid services, and the transformation in the different industrial subsectors do not evolve with time. They are kept fixed at the values specified in the configuration file. Including the transport sector and industry in the myopic code is planned for the near future.

See also other outstanding issues.

1.5.1 Configuration

PyPSA-Eur-Sec has several configuration options which are collected in a config.yaml file located in the root directory. For myopic optimization, users should copy the provided default configuration `config.default.yaml` and make their own modifications and assumptions in the user-specific configuration file (`config.yaml`).

The following options included in the config.yaml file are relevant for the myopic code.

To activate the myopic option select `foresight: 'myopic'` in `config.yaml`.

To set the investment years which are sequentially simulated for the myopic investment planning, select for example `planning_horizons : [2020, 2030, 2040, 2050]` in `config.yaml`.

existing capacities

Grouping years indicates the bins limits for grouping the existing capacities of different technologies

grouping_years: [1980, 1985, 1990, 1995, 2000, 2005, 2010, 2015, 2019]

threshold capacity

if for a technology, node, and grouping bin, the capacity is lower than threshold_capacity, it is ignored threshold_capacity: 10

conventional carriers

conventional carriers indicate carriers used in the existing conventional technologies

conventional_carriers: [‘lignite’, ‘coal’, ‘oil’, ‘uranium’]

1.5.2 Wildcards

{planning_horizons} wildcard

The {planning_horizons} wildcard indicates the timesteps in which the network is optimized, e.g. planning_horizons: [2020, 2030, 2040, 2050]

1.5.3 Options

The total carbon budget for the entire transition path can be indicated in the `scenario.sector_opts` in `config.yaml`. The carbon budget can be split among the `planning_horizons` following an exponential or beta decay. E.g. `'cb40ex0'` splits the a carbon budget equal to 40 GtCO_2 following an exponential decay whose initial linear growth rate r is zero

$e(t) = e_0 (1+ (r+m)t) e^(-mt)$

See details in Supplementary Note 1 of the paper Early decarbonisation of the European Energy system pays off (2020)

1.5.4 Rules overview

1.5.5 General myopic code structure

The myopic code solves the network for the time steps included in `planning_horizons` in a recursive loop, so that:

1.The existing capacities (those installed before the base year are added as fixed capacities with p_nom=value, p_nom_extendable=False). E.g. for baseyear=2020, capacities installed before 2020 are added. In addition, the network comprises additional generator, storage, and link capacities with p_nom_extendable=True. The non-solved network is saved in `results/run_name/networks/prenetworks-brownfield`.

The base year is the first element in `planning_horizons`. Step 1 is implemented with the rule add_baseyear for the base year and with the rule add_brownfield for the remaining planning_horizons.

2.The 2020 network is optimized. The solved network is saved in `results/run_name/networks/postnetworks`

3.For the next planning horizon, e.g. 2030, the capacities from a previous time step are added if they are still in operation (i.e., if they fulfil planning horizon <= commissioned year + lifetime). In addition, the network comprises additional generator, storage, and link capacities with p_nom_extendable=True. The non-solved network is saved in `results/run_name/networks/prenetworks-brownfield`.

Steps 2 and 3 are solved recursively for all the planning_horizons included in `config.yaml`.

1.5.6 rule add_existing baseyear

The rule add_existing_baseyear loads the network in ‘results/run_name/networks/prenetworks’ and performs the following operations:

1.Add the conventional, wind and solar power generators that were installed before the base year.

2.Add the heating capacities that were installed before the base year.

The existing conventional generators are retrieved from the powerplants.csv file generated by pypsa-eur which, in turn, is based on the powerplantmatching database.

Existing wind and solar capacities are retrieved from IRENA annual statistics and distributed among the nodes in a country proportional to capacity factor. (This will be updated to include capacity distributions closer to reality.)

Existing heating capacities are retrieved from the report Mapping and analyses of the current and future (2020 - 2030) heating/cooling fuel deployment (fossil/renewables)

The heating capacities are assumed to have a lifetime indicated by the parameter lifetime in the configuration file, e.g 25 years. They are assumed to be decommissioned linearly starting on the base year, e.g., from 2020 to 2045.

Then, the resulting network is saved in `results/run_name/networks/prenetworks-brownfield`.

1.5.7 rule add_brownfield

The rule add_brownfield loads the network in `results/run_name/networks/prenetworks` and performs the following operation:

1.Read the capacities optimized in the previous time step and add them to the network if they are still in operation (i.e., if they fulfill planning horizon < commissioned year + lifetime)

Then, the resulting network is saved in `results/run_name/networks/prenetworks_brownfield`.

References

- *Release Notes*

1.6 Release Notes

1.6.1 Future release

Note: This unreleased version currently may require the master branches of PyPSA, PyPSA-Eur, and the technology-data repository.

1.6.2 PyPSA-Eur-Sec 0.6.0 (4 October 2021)

This release includes improvements regarding the basic chemical production, the addition of plastics recycling, the addition of the agriculture, forestry and fishing sector, more regionally resolved biomass potentials, CO2 pipeline transport and storage, and more options in setting exogenous transition paths, besides many performance improvements.

This release is known to work with PyPSA-Eur Version 0.4.0, Technology Data Version 0.3.0 and PyPSA Version 0.18.0.

Please note that the data bundle has also been updated.

General

- With this release, we change the license from copyleft GPLv3 to the more liberal MIT license with the consent of all contributors.

New features and functionality

- Distinguish costs for home battery storage and inverter from utility-scale battery costs.
- Separate basic chemicals into HVC (high-value chemicals), chlorine, methanol and ammonia [#166].
- Add option to specify reuse, primary production, and mechanical and chemical recycling fraction of platics [#166].
- Include energy demands and CO2 emissions for the agriculture, forestry and fishing sector. It is included by default through the option `A` in the `sector_opts` wildcard. Part of the emissions (1.A.4.c) was previously assigned to "industry non-elec" in the `co2_totals.csv`. Hence, excluding the agriculture sector will now lead to a tighter CO2 limit. Energy demands are taken from the JRC IDEES database (missing countries filled with eurostat data) and are split into electricity (lighting, ventilation, specific electricity uses, pumping devices (electric)), heat (specific heat uses, low enthalpy heat) machinery oil (motor drives, farming machine drives, pumping devices (diesel)). Heat demand is assigned at "services rural heat" buses. Electricity demands are added to low-voltage buses. Time series for demands are constant and distributed inside countries by population [#147].
- Include today's district heating shares in myopic optimisation and add option to specify exogenous path for district heating share increase under `sector: district_heating:` [#149].
- Added option for hydrogen liquefaction costs for hydrogen demand in shipping. This introduces a new `H2 liquid` bus at each location. It is activated via `sector: shipping_hydrogen_liquefaction: true`.
- The share of shipping transformed into hydrogen fuel cell can be now defined for different years in the `config.yaml` file. The carbon emission from the remaining share is treated as a negative load on the atmospheric carbon dioxide bus, just like aviation and land transport emissions.
- The transformation of the Steel and Aluminium production can be now defined for different years in the `config.yaml` file.
- Include the option to alter the maximum energy capacity of a store via the `carrier+factor` in the `{sector_opts}` wildcard. This can be useful for sensitivity analyses. Example: `co2 stored+e2` multiplies the `e_nom_max` by factor 2. In this example, `e_nom_max` represents the CO2 sequestration potential in Europe.
- Use JRC ENSPRESO database to spatially disaggregate biomass potentials to PyPSA-Eur regions based on overlaps with NUTS2 regions from ENSPRESO (proportional to area) (#151).
- Add option to regionally disaggregate biomass potential to individual nodes (previously given per country, then distributed by population density within) and allow the transport of solid biomass. The transport costs are determined based on the JRC-EU-Times Bioenergy report in the new optional rule `build_biomass_transport_costs`. Biomass transport can be activated with the setting `sector: biomass_transport: true`.
- Add option to regionally resolve CO2 storage and add CO2 pipeline transport because geological storage potential, CO2 utilisation sites and CO2 capture sites may be separated. The CO2 network is built from zero based on the topology of the electricity grid (greenfield). Pipelines are assumed to be bidirectional and lossless. Furthermore, neither retrofitting of natural gas pipelines (required pressures are too high, 80-160 bar vs <80 bar) nor other modes of CO2 transport (by ship, road or rail) are considered. The regional representation of CO2 is activated with the config setting `sector: co2_network: true` but is deactivated by default. The global limit for CO2 sequestration now applies to the sum of all CO2 stores via an `extra_functionality` constraint.
- The myopic option can now be used together with different clustering for the generators and the network. The existing renewable capacities are split evenly among the regions in every country [#144].
- Add optional function to use `geopy` to locate entries of the Hotmaps database of industrial sites with missing location based on city and country, which reduces missing entries by half. It can be activated by setting

`industry: hotmaps_locate_missing: true`, takes a few minutes longer, and should only be used if spatial resolution is coarser than city level.

Performance and Structure

- Extended use of `multiprocessing` for much better performance (from up to 20 minutes to less than one minute).
- Handle most input files (or base directories) via `snakemake.input`.
- Use of `mock_snakemake` from PyPSA-Eur.
- Update `solve_network` rule to match implementation in PyPSA-Eur by using `n.ilopf()` and remove outdated code using `pyomo`. Allows the new setting to skip iterated impedance updates with `solving: options: skip_iterations: true`.
- The component attributes that are to be overridden are now stored in the folder `data/override_component_attrs` analogous to `pypsa/component_attrs`. This reduces verbosity and also allows circumventing the `n.madd()` hack for individual components with non-default attributes. This data is also tracked in the Snakefile. A function `helper.override_component_attrs` was added that loads this data and can pass the overridden component attributes into `pypsa.Network()`.
- Add various parameters to `config.default.yaml` which were previously hardcoded inside the scripts (e.g. energy reference years, BEV settings, solar thermal collector models, geomap colours).
- Removed stale industry demand rules `build_industrial_energy_demand_per_country` and `build_industrial_demand`. These are superseded with more regionally resolved rules.
- Use simpler and shorter `gdf.sjoin()` function to allocate industrial sites from the Hotmaps database to onshore regions. This change also fixes a bug: The previous version allocated sites to the closest bus, but at country borders (where Voronoi cells are distorted by the borders), this had resulted in e.g. a Spanish site close to the French border being wrongly allocated to the French bus if the bus center was closer.
- Retrofitting rule is now only triggered if endogeneously optimised.
- Show progress in build rules with `tqdm` progress bars.
- Reduced verbosity of `Snakefile` through directory prefixes.
- Improve legibility of `config.default.yaml` and remove unused options.
- Use the country-specific time zone mappings from `pytz` rather than a manual mapping.
- A function `add_carrier_buses()` was added to the `prepare_network` rule to reduce code duplication.
- In the `prepare_network` rule the cost and potential adjustment was moved into an own function `maybe_adjust_costs_and_potentials()`.
- Use `matplotlibrc` to set the default plotting style and backend.
- Added benchmark files for each rule.
- Consistent use of `__main__` block and further unspecific code cleaning.
- Updated data bundle and moved data bundle to zenodo.org (10.5281/zenodo.5546517).

Bugfixes and Compatibility

- Compatibility with `atlite>=0.2`. Older versions of `atlite` will no longer work.
- Corrected calculation of "gas for industry" carbon capture efficiency.
- Implemented changes to `n.snapshot_weightings` in PyPSA v0.18.0.
- Compatibility with `xarray` version 0.19.

- New dependencies: `tqdm`, `atlite>=0.2.4`, `pytz` and `geopy` (optional). These are included in the environment specifications of PyPSA-Eur v0.4.0.

Many thanks to all who contributed to this release!

1.6.3 PyPSA-Eur-Sec 0.5.0 (21st May 2021)

This release includes improvements to the cost database for building retrofits, carbon budget management and wildcard settings, as well as an important bugfix for the emissions from land transport.

This release is known to work with PyPSA-Eur Version 0.3.0 and Technology Data Version 0.2.0.

Please note that the data bundle has also been updated.

New features and bugfixes:

- The cost database for retrofitting of the thermal envelope of buildings has been updated. Now, for calculating the space heat savings of a building, losses by thermal bridges and ventilation are included as well as heat gains (internal and by solar radiation). See the section *Retrofitting of the thermal envelope of buildings* for more details on the retrofitting module.
- For the myopic investment option, a carbon budget and a type of decay (exponential or beta) can be selected in the `config.yaml` file to distribute the budget across the `planning_horizons`. For example, `cb40ex0` in the `{sector_opts}` wildcard will distribute a carbon budget of 40 GtCO2 following an exponential decay with initial growth rate 0.
- Added an option to alter the capital cost or maximum capacity of carriers by a factor via `carrier+factor` in the `{sector_opts}` wildcard. This can be useful for exploring uncertain cost parameters. Example: `solar+c0.5` reduces the `capital_cost` of solar to 50% of original values. Similarly `solar+p3` multiplies the `p_nom_max` by 3.
- Rename the bus for European liquid hydrocarbons from `Fischer-Tropsch` to `EU oil`, since it can be supplied not just with the Fischer-Tropsch process, but also with fossil oil.
- Bugfix: The new separation of land transport by carrier in Version 0.4.0 failed to account for the carbon dioxide emissions from internal combustion engines in land transport. This is now treated as a negative load on the atmospheric carbon dioxide bus, just like aviation emissions.
- Bugfix: Fix reading in of `pypsa-eur/resources/powerplants.csv` to PyPSA-Eur Version 0.3.0 (use column attribute name `DateIn` instead of old `YearDecommissioned`).
- Bugfix: Make sure that `Store` components (battery and H2) are also removed from PyPSA-Eur, so they can be added later by PyPSA-Eur-Sec.

Thanks to Lisa Zeyen (KIT) for the retrofitting improvements and Marta Victoria (Aarhus University) for the carbon budget and wildcard management.

1.6.4 PyPSA-Eur-Sec 0.4.0 (11th December 2020)

This release includes a more accurate nodal disaggregation of industry demand within each country, fixes to CHP and CCS representations, as well as changes to some configuration settings.

It has been released to coincide with PyPSA-Eur Version 0.3.0 and Technology Data Version 0.2.0, and is known to work with these releases.

New features:

- The Hotmaps Industrial Database is used to disaggregate the industrial demand spatially to the nodes inside each country (previously it was distributed by population density).

- Electricity demand from industry is now separated from the regular electricity demand and distributed according to the industry demand. Only the remaining regular electricity demand for households and services is distributed according to GDP and population.
- A cost database for the retrofitting of the thermal envelope of residential and services buildings has been integrated, as well as endogenous optimisation of the level of retrofitting. This is described in the paper Mitigating heat demand peaks in buildings in a highly renewable European energy system. Retrofitting can be activated both exogenously and endogenously from the `config.yaml`.
- The biomass and gas combined heat and power (CHP) parameters `c_v` and `c_b` were read in assuming they were extraction plants rather than back pressure plants. The data is now corrected in Technology Data Version 0.2.0 to the correct DEA back pressure assumptions and they are now implemented as single links with a fixed ratio of electricity to heat output (even as extraction plants, they were always sitting on the backpressure line in simulations, so there was no point in modelling the full heat-electricity feasibility polygon). The old assumptions underestimated the heat output.
- The Danish Energy Agency released new assumptions for carbon capture in October 2020, which have now been incorporated in PyPSA-Eur-Sec, including direct air capture (DAC) and post-combustion capture on CHPs, cement kilns and other industrial facilities. The electricity and heat demand for DAC is modelled for each node (with heat coming from district heating), but currently the electricity and heat demand for industrial capture is not modelled very cleanly (for process heat, 10% of the energy is assumed to go to carbon capture) - a new issue will be opened on this.
- Land transport is separated by energy carrier (fossil, hydrogen fuel cell electric vehicle, and electric vehicle), but still needs to be separated into heavy and light vehicles (the data is there, just not the code yet).
- For assumptions that change with the investment year, there is a new time-dependent format in the `config.yaml` using a dictionary with keys for each year. Implemented examples include the CO2 budget, exogenous retrofitting share and land transport energy carrier; more parameters will be dynamised like this in future.
- Some assumptions have been moved out of the code and into the `config.yaml`, including the carbon sequestration potential and cost, the heat pump sink temperature, reductions in demand for high value chemicals, and some BEV DSM parameters and transport efficiencies.
- Documentation on *Supply and demand* options has been added.

Many thanks to Fraunhofer ISI for opening the hotmaps database and to Lisa Zeyen (KIT) for implementing the building retrofitting.

1.6.5 PyPSA-Eur-Sec 0.3.0 (27th September 2020)

This releases focuses on improvements to industry demand and the generation of intermediate files for demand for basic materials. There are still inconsistencies with CCS and waste management that need to be improved.

It is known to work with PyPSA-Eur v0.1.0 (commit bb3477cd69), PyPSA v0.17.1 and technology-data v0.1.0. Please note that the data bundle has also been updated.

New features:

- In previous version of PyPSA-Eur-Sec the energy demand for industry was calculated directly for each location. Now, instead, the production of each material (steel, cement, aluminium) at each location is calculated as an intermediate data file, before the energy demand is calculated from it. This allows us in future to have competing industrial processes for supplying the same material demand.
- The script `build_industrial_production_per_country_tomorrow.py` determines the future industrial production of materials based on today’s levels as well as assumed recycling and demand change measures.
- The energy demand for each industry sector and each location in 2015 is also calculated, so that it can be later incorporated in the pathway optimization.

- Ammonia production data is taken from the USGS and deducted from JRC-IDEES's "basic chemicals" so that it ammonia can be handled separately from the others (olefins, aromatics and chlorine).
- Solid biomass is no longer allowed to be used for process heat in cement and basic chemicals, since the wastes and residues cannot be guaranteed to reach the high temperatures required. Instead, solid biomass is used in the paper and pulp as well as food, beverages and tobacco industries, where required temperatures are lower (see DOI:10.1002/er.3436 and DOI:10.1007/s12053-017-9571-y).
- National installable potentials for salt caverns are now applied.
- When electricity distribution grids are activated, new industry electricity demand, resistive heaters and micro-CHPs are now connected to the lower voltage levels.
- Gas distribution grid costs are included for gas boilers and micro-CHPs.
- Installable potentials for rooftop PV are included with an assumption of 1 kWp per person.
- Some intermediate files produced by scripts have been moved from the folder `data` to the folder `resources`. Now `data` only includes input data, while `resources` only includes intermediate files necessary for building the network models. Please note that the data bundle has also been updated.
- Biomass potentials for different years and scenarios from the JRC are generated in an intermediate file, so that a selection can be made more explicitly by specifying the biomass types from the `config.yaml`.

1.6.6 PyPSA-Eur-Sec 0.2.0 (21st August 2020)

This release introduces pathway optimization over many years (e.g. 2020, 2030, 2040, 2050) with myopic foresight, as well as outsourcing the technology assumptions to the technology-data repository.

It is known to work with PyPSA-Eur v0.1.0 (commit bb3477cd69), PyPSA v0.17.1 and technology-data v0.1.0.

New features:

- Option for pathway optimization with myopic foresight, based on the paper Early decarbonisation of the European Energy system pays off (2020). Investments are optimized sequentially for multiple years (e.g. 2020, 2030, 2040, 2050) taking account of existing assets built in previous years and their lifetimes. The script uses data on the existing assets for electricity and building heating technologies, but there are no assumptions yet for existing transport and industry (if you include these, the model will greenfield them). There are also some outstanding issues on e.g. the distribution of existing wind, solar and heating technologies within each country. To use myopic foresight, set `foresight : 'myopic'` in the `config.yaml` instead of the default `foresight : 'overnight'`. An example configuration can be found in `config.myopic.yaml`. More details on the implementation can be found in *Myopic transition path*.
- Technology assumptions (costs, efficiencies, etc.) are no longer stored in the repository. Instead, you have to install the technology-data database in a parallel directory. These assumptions are largely based on the Danish Energy Agency Technology Data. More details on the installation can be found in *Installation*.
- Logs and benchmarks are now stored with the other model outputs in `results/run-name/`.
- All buses now have a `location` attribute, e.g. bus `DE0 3 urban central heat` has a `location` of `DE0 3`.
- All assets have a `lifetime` attribute (integer in years). For the myopic foresight, a `build_year` attribute is also stored.
- Costs for solar and onshore and offshore wind are recalculated by PyPSA-Eur-Sec based on the investment year, including the AC or DC connection costs for offshore wind.

Many thanks to Marta Victoria for implementing the myopic foresight, and Marta Victoria, Kun Zhu and Lisa Zeyen for developing the technology assumptions database.

1.6.7 PyPSA-Eur-Sec 0.1.0 (8th July 2020)

This is the first proper release of PyPSA-Eur-Sec, a model of the European energy system at the transmission network level that covers the full ENTSO-E area.

It is known to work with PyPSA-Eur v0.1.0 (commit bb3477cd69) and PyPSA v0.17.0.

We are making this release since in version 0.2.0 we will introduce changes to allow myopic investment planning that will require minor changes for users of the overnight investment planning.

PyPSA-Eur-Sec builds on the electricity generation and transmission model PyPSA-Eur to add demand and supply for the following sectors: transport, space and water heating, biomass, industry and industrial feedstocks. This completes the energy system and includes all greenhouse gas emitters except waste management, agriculture, forestry and land use.

PyPSA-Eur-Sec was initially based on the model PyPSA-Eur-Sec-30 (Version 0.0.1 below) described in the paper Synergies of sector coupling and transmission reinforcement in a cost-optimised, highly renewable European energy system (2018) but it differs by being based on the higher resolution electricity transmission model PyPSA-Eur rather than a one-node-per-country model, and by including biomass, industry, industrial feedstocks, aviation, shipping, better carbon management, carbon capture and usage/sequestration, and gas networks.

PyPSA-Eur-Sec includes PyPSA-Eur as a snakemake subworkflow. PyPSA-Eur-Sec uses PyPSA-Eur to build the clustered transmission model along with wind, solar PV and hydroelectricity potentials and time series. Then PyPSA-Eur-Sec adds other conventional generators, storage units and the additional sectors.

1.6.8 PyPSA-Eur-Sec 0.0.2 (4th September 2020)

This version, also called PyPSA-Eur-Sec-30-Path, built on PyPSA-Eur-Sec 0.0.1 (also called PyPSA-Eur-Sec-30) to include myopic pathway optimisation for the paper Early decarbonisation of the European energy system pays off (2020). The myopic pathway optimisation was then merged into the main PyPSA-Eur-Sec codebase in Version 0.2.0 above.

This model has its own github repository and is archived on Zenodo.

1.6.9 PyPSA-Eur-Sec 0.0.1 (12th January 2018)

This is the first published version of PyPSA-Eur-Sec, also called PyPSA-Eur-Sec-30. It was first used in the research paper Synergies of sector coupling and transmission reinforcement in a cost-optimised, highly renewable European energy system (2018). The model covers 30 European countries with one node per country. It includes demand and supply for electricity, space and water heating in buildings, and land transport.

It is archived on Zenodo.

1.6.10 Release Process

- Finalise release notes at `doc/release_notes.rst`.
- Update version number in `doc/conf.py` and `*config.*.yaml`.
- Make a `git commit`.
- Tag a release by running `git tag v0.x.x`, `git push`, `git push --tags`. Include release notes in the tag message.
- Make a GitHub release, which automatically triggers archiving by zenodo.
- Send announcement on the PyPSA mailing list.

To make a new release of the data bundle, make an archive of the files in `data` which are not already included in the git repository:

```
data % tar pczf pypsa-eur-sec-data-bundle.tar.gz eea/UNFCCC_v23.csv switzerland-sfoe␣
↪biomass eurostat-energy_balances-* jrc-idees-2015 emobility WindWaveWEC_GLTB.xlsx␣
↪myb1-2017-nitro.xls Industrial_Database.csv retro/tabula-calculator-calcsetbuilding.
↪csv nuts/NUTS_RG_10M_2013_4326_LEVL_2.geojson
```

CHAPTER 2

Warnings

WARNING: This model is under construction and contains serious problems that distort the results. See the github repository issues for some of the problems (please feel free to help or make suggestions). There is neither documentation nor a paper yet, but we hope to have a preprint out by summer 2020. We cannot support this model if you choose to use it.

CHAPTER 3

Licence

The code in PyPSA-Eur-Sec is released as free software under the MIT license, see LICENSE. However, different licenses and terms of use may apply to the various input data.