

---

# **DispaSET Documentation**

***Release 2.2***

**Sylvain Quoilin**

August 31, 2018



<b>1</b>	<b>Model description and philosophy</b>	<b>3</b>
<b>2</b>	<b>Downloading Dispa-SET</b>	<b>5</b>
<b>3</b>	<b>Documentation</b>	<b>7</b>
<b>4</b>	<b>Contents</b>	<b>9</b>
<b>5</b>	<b>Indices and tables</b>	<b>51</b>



The Dispa-SET model is an open-source unit commitment and optimal dispatch model focused on the balancing and flexibility problems in European grids. Its pre and post-processing tools are written in Python and the main solver can be called via GAMS or via PYOMO. The selected Mixed-Integer Linear Programming (MILP) solver is CPLEX.

Dispa-SET is mainly developed within the Joint Research Centre of the EU Commission, in close collaboration with the University of Liège (Belgium).



---

# Model description and philosophy

---

A comprehensive description of the model is available in the 2017 JRC technical report: [Modelling Future EU Power Systems Under High Shares of Renewables](#).







---

### Downloading Dispa-SET

---

The public version of Dispa-SET can be downloaded in the [Releases](#) section or from its github repository (using the Clone or Download button on the right side of the screen): <https://github.com/energy-modelling-toolkit/Dispa-SET>



---

### Documentation

---

The model documentation is available by running sphinx in the Docs folder of the project or by consulting the online documentation. This documentation corresponds to the latest available public version of Dispa-SET: <http://www.dispaset.eu/latest/index.html>

Main contributors:

- Sylvain Quoilin (University of Liège, Belgium))
- Konstantinos Kavvadias (Joint Research Centre, EU Commission)
- Andreas Zucker (Joint Research Centre, EU Commission)



---

## Contents

---

### Overview

**Organization** Joint Research Centre, European Commission,

**Version** 2.2

**Date** August 31, 2018

The Dispa-SET model is mainly developed within the “Joint Research Centre” of the European Commission and focused on the balancing and flexibility problems in European grids <sup>1</sup>.

It is written in GAMS and Python (Pyomo) and uses csv files for input data handling. The optimisation is defined as a Linear Programming (LP) or Mixed-Integer Linear Programming (MILP) problem, depending on the desired level of accuracy and complexity. Continuous variables include the individual unit dispatched power, the shedded load and the curtailed power generation. The binary variables are the commitment status of each unit. The main model features can be summarized as follows:

### Features

- Minimum and maximum power for each unit
- Power plant ramping limits
- Reserves up and down
- Minimum up/down times
- Load Shedding
- Curtailment
- Pumped-hydro storage
- Non-dispatchable units (e.g. wind turbines, run-of-river, etc.)
- Start-up, ramping and no-load costs
- Multi-nodes with capacity constraints on the lines (congestion)
- Constraints on the targets for renewables and/or CO2 emissions
- Yearly schedules for the outages (forced and planned) of each units

---

<sup>1</sup> Quoilin, S., Hidalgo Gonzalez, I., & Zucker, A. (2017). Modelling Future EU Power Systems Under High Shares of Renewables: The Dispa-SET 2.1 open-source models. Publications Office of the European Union.

- CHP power plants and thermal storage

The demand is assumed to be inelastic to the price signal. The MILP objective function is therefore the total generation cost over the optimization period.

## Libraries used

- `pyomo` Optimization object library, interface to LP solver (e.g. CPLEX)
- `pandas` for input and result data handling
- `matplotlib` for plotting
- `GAMS_api` for the communication with GAMS

## Ongoing developments

The Dispa-SET project is relatively recent, and a number of improvements will be brought to the project in a close future:

- Grid constraints (DC power-flow)
- Stochastic scenarios
- Extension of the modeled areas
- Modeling of the ancillary markets
- User interface

## Public administration reference

This software is primarily developed and used within the Institute for Energy and Transport, which is one of the 7 scientific institutes of the Joint Research Centre (JRC) of the European Commission. The IET is based both in Petten, the Netherlands, and Ispra, Italy. The Dispa-SET model is developed in the framework of the “Energy Systems Modelling” (ESM) project.

## Licence

Dispa-SET is a free software licensed under the “European Union Public Licence” EUPL v1.1. It can be redistributed and/or modified under the terms of this license.

## Main Developers

- Sylvain Quoilin (University of Liège)
- Andreas Zucker (European Commission, Institute for Energy and Transport)
- Konstantinos Kavvadias (European Commission, Institute for Energy and Transport)

## References

## Getting Started

This short tutorial describes the main steps to get a practical example of Dispa-SET running.

### Prerequisites

Install Python 2.7, with full scientific stack. The [Anaconda](#) distribution is recommended since it comprises all the required packages. If Anaconda is not used, the following libraries and their dependencies should be installed manually:

- numpy
- pandas (> v0.18.1)
- matplotlib
- xlrd
- pickle

This can be achieved using the pip installer (example for numpy):

```
pip install numpy
```

NB: For Windows users, some packages might require the installation of a C++ compiler for Python. This corresponds to the typical error message: “Unable to find vcvarsall.bat”. This can be solved by installing the freely available “Microsoft Visual C++ Compiler for Python 2.7”. In some cases the path to the compiler must be added to the PATH windows environment variable (e.g. C:\Program Files\Common Files\Microsoft Visual C++ for Python\9.0)

### Using Dispa-SET with GAMS:

Dispa-SET is primarily designed to run with GAMS and therefore requires GAMS to be installed with a valid user licence. Currently, only the 64-bit version of GAMS is supported in Dispa-SET!

The GAMS api for python has been pre-compiled in the “Externals” folder and is usable with most operating systems. If the pre-compiled binaries are not available or could not be loaded, the system exits with an error message. In that case, the gams python api should be compiled from the source provided in the GAMS installation folder (e.g. “C:\GAMS\win64\24.3\apifiles\Python\api”):

```
python gdxsetup.py install
python gamsssetup.py install
```

The api requires the path to the gams installation folder. The “get\_gams\_path()” function of dispa-set performs a system search to automatically detect this path. If it is not successful, the user is prompted for the proper installation path.

### Using Dispa-SET with PYOMO:

- Install pyomo

```
pip install pyomo
```

- Install a solver and add it to the PATH environment variable (e.g. if cplex is installed, the “cplex” command should be callable from any command prompt).

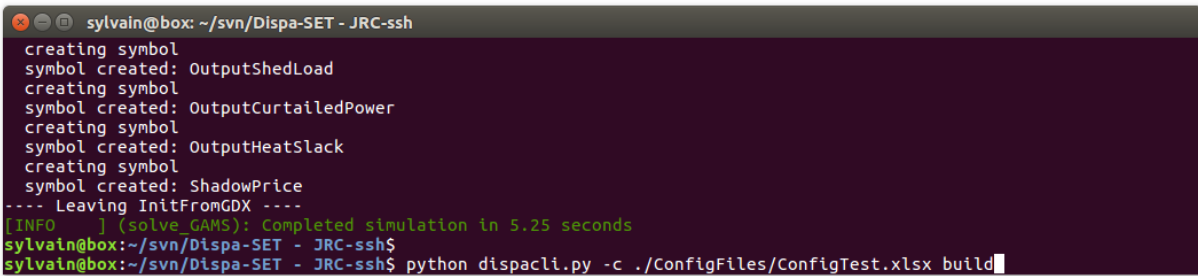
## Step-by-step example of a Dispa-SET run

This section describes the pre-processing and the solving phases of a Dispa-SET run. Three equivalent methods are described in the next sections:

- Using the command line interface
- Using the Dispa-SET API
- Using GAMS

### 1. Using the command line interface

Dispa-SET can be run from the command line. To that aim, open a terminal window and change the directory to the Dispa-SET root folder.



```
sylvain@box: ~/svn/Dispa-SET - JRC-ssh
creating symbol
symbol created: OutputShedLoad
creating symbol
symbol created: OutputCurtailedPower
creating symbol
symbol created: OutputHeatSlack
creating symbol
symbol created: ShadowPrice
---- Leaving InitFromGDx ----
[INFO ] (solve_GAMS): Completed simulation in 5.25 seconds
sylvain@box:~/svn/Dispa-SET - JRC-ssh$
sylvain@box:~/svn/Dispa-SET - JRC-ssh$ python dispacli.py -c ./ConfigFiles/ConfigTest.xlsx build
```

#### 1.1. Check the configuration file

Dispa-SET runs are defined in dedicated excel configuration files stored in the “ConfigFiles” folder. The configuration file “ConfigTest.xlsx” is provided for testing purposes. It generates a 10-days optimisation using data relative a fictitious power system composed of two zones Z1 and Z2.

#### 1.2. Pre-processing

From the command line, specify the configuration file to be used as an argument, the solver (Pyomo or GAMS) and the actions to be performed. Within the “Dispa-SET” folder, run:

```
python dispacli.py -c ./ConfigFiles/ConfigTest.xlsx build
```

NB: The command line interface `dispacli.py` is designed to run with the Python interpreter, which should be the one selected. It might present some problems when run in Ipython.

#### 1.3. Check the simulation environment

The simulation environment folder is defined in the configuration file. In the test example it is set to “Simulations/simulation\_test”. The simulation inputs are written in three different formats: excel (34 excel files), Python (Inputs.p) and GAMS (Inputs.gdx).

#### 1.4. Run the optimisation

Using the GAMS api, the simulation can be started directly from the main DispaSet python file after the pre-processing phase. From the “Dispa-SET” folder, run:



```
python dispacli.py -g -c ./ConfigFiles/ConfigTest.xlsx simulate
```

This generates the simulation environment, runs the optimisation, and stores the results in the same folder. Note that this can only work if the simulation has been pre-processed before (step 1.2). It is possible to combine the pre-processing and simulation step in one command:

```
python dispacli.py -g -c ./ConfigFiles/ConfigTest.xlsx build simulate
```

The same action can be performed using the PYOMO solver. In that case, the “-g” argument must be changed into “-p”:

```
python dispacli.py -p -c ./ConfigFiles/ConfigTest.xlsx build simulate
```

## 2. Using the Dispa-SET API.

The steps to run a model can be also performed directly in python, by importing the Dispa-SET library. An example file (“build\_and\_run.py”) is available in the “scripts/” folder. After checking the configuration file “ConfigTest.xlsx” (in the “ConfigFiles” folder). Run the following python commands:

2.1 Import Dispa-SET:

```
import DispaSET as ds
```

2.2 Load the configuration file:

```
config = ds.load_config_excel('ConfigFiles/ConfigTest.xlsx')
```

2.3 Build the simulation environment (Folder that contains the input data and the simulation files required for the solver):

```
SimData = ds.build_simulation(config)
```

2.4a Solve using PYOMO:

```
r = ds.solve_pyomo(config['SimulationDirectory'])
```

2.4b Solve using GAMS:

```
r = ds.solve_GAMS(config['SimulationDirectory'], config['GAMS_folder'])
```

A more detailed description of the Dispa-SET functions is available in the API section.

## 3. Using GAMS

It is sometimes useful to run the dispa-SET directly in GAMS (e.g. for debugging purposes). In that case, the pre-processing must be run first (steps 1.2 or 2.1, 2.2 and 2.3) and the gams file generated in the simulation folder can be used to run the optimization.

### Using the GAMS graphical user interface:

From the simulation folder (defined in the config file), the Dispa-SET model can be run following the instruction below:

1. Open the UCM.gpr project file in GAMS
2. From GAMS, open the UCM\_h.gmx model file

3. Run the model in GAMS.

The result file is written in the.gdx format and stored in the Simulation folder, together with all input files.

### Using the GAMS command line:

GAMS can also be run from the command line (this is the only option for the Linux version).

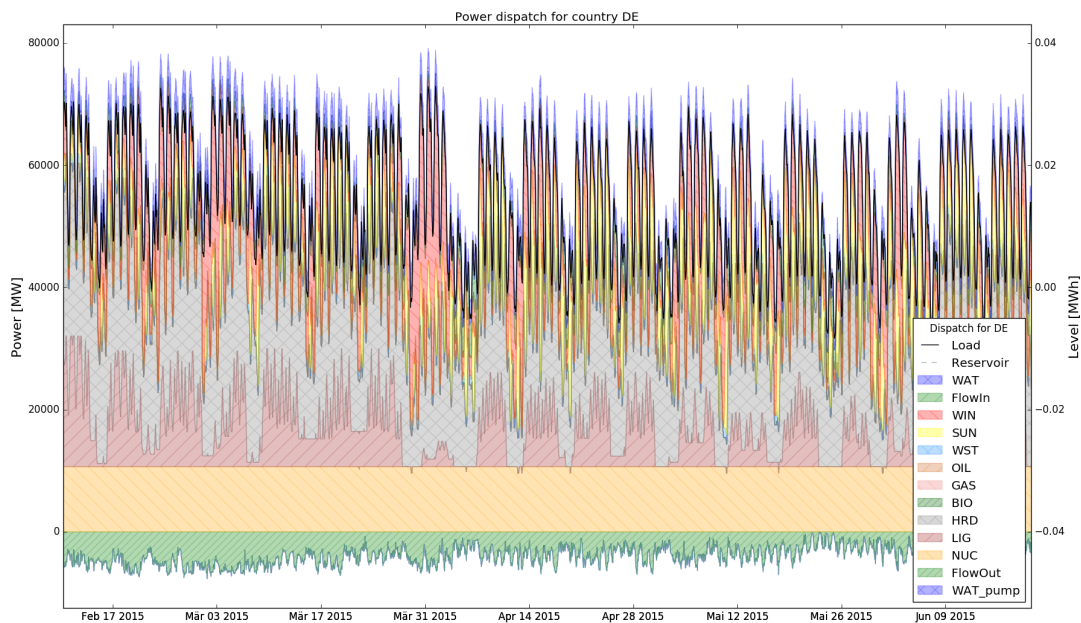
1. Make sure that the gams binary is in the system PATH
2. From the simulation environment folder, run:

```
gams UCM_h.gms
```

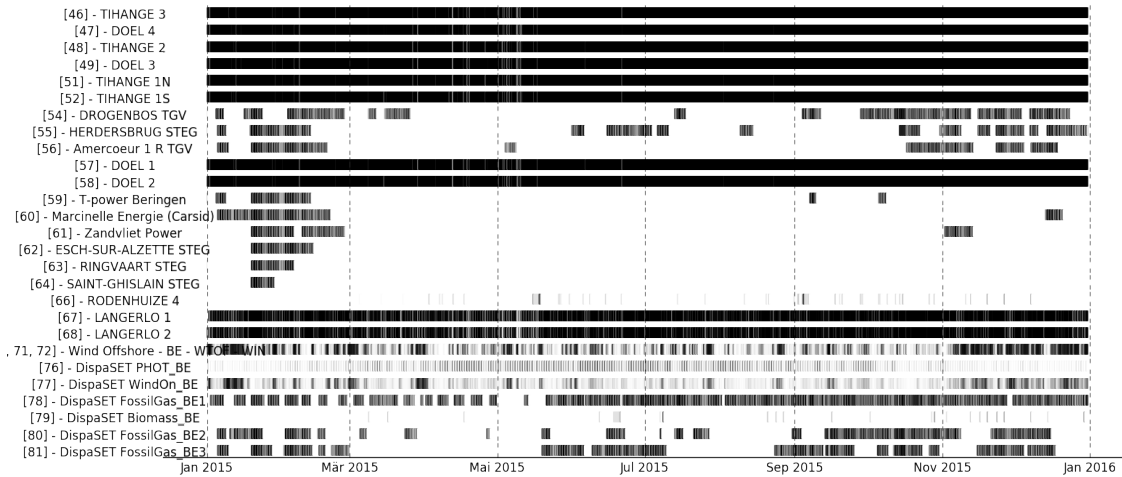
## Postprocessing

Various functions and tools are provided within the PostProcessing.py file to load, analyse and plot the simulation results. The use of these functions is illustrated into the the “Read\_results\_notebook.ipynb” Notebook or in the “read\_results.py” script, which can be run by changing the path to the simulation folder. The type of results provided by the post-processing is illustrated hereunder.

The power dispatch can be plotted for each simulated zone. In this plot, the units are aggregated by fuel type. The power consumed by storage units and the exportations are indicated as negative values.



It is also interesting to display the results at the unit level to gain deeper insights regarding the dispatch. In that case, a plot is generated, showing the commitment status of all units in a zone at each timestep. Both the dispatch plot and the commitment plot can be called using the CountryPlots function.



Some aggregated statistics on the simulations results can also be obtained, including the number of hours of congestion in each interconnection line, the yearly energy balances for each zone, the amount of lost load, etc.

Aggregated statistics for the considered area:

Total consumption: 1227.07310992 TWh

Peak load: 203182.461067 MW

Net importations: -42.20072928 TWh

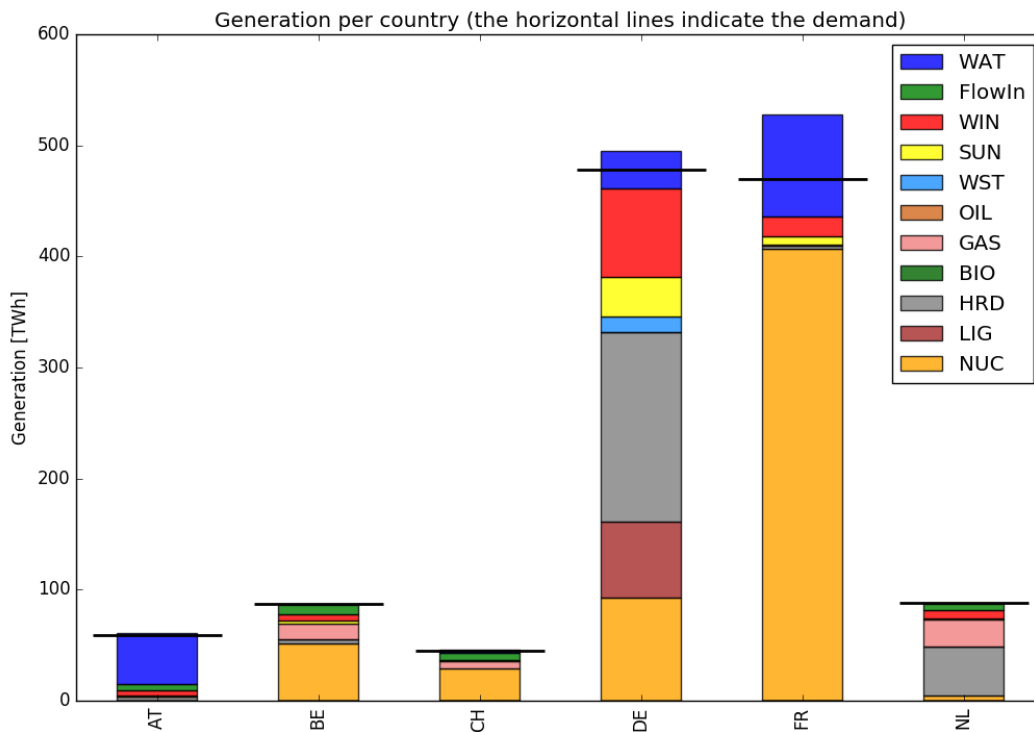
Country-Specific values (in TWh or in MW):

	Demand	PeakLoad	NetImports	LoadShedding	Curtailment
AT	59.375448	10144.000000	5.144132	NaN	NaN
BE	86.971154	13632.250000	8.911190	NaN	NaN
CH	44.694098	7794.262468	7.199527	NaN	NaN
DE	478.030824	76212.250000	-17.260122	NaN	NaN
FR	470.075612	90588.000000	-51.878128	NaN	NaN
NL	87.925973	16285.500000	5.682672	NaN	NaN

Number of hours of congestion on each line:

```
{'AT -> CH': 5917,
 'AT -> DE': 430,
 'BE -> FR': 62,
 'BE -> NL': 344,
 'CH -> AT': 720,
 'CH -> DE': 15,
 'CH -> FR': 56,
 'DE -> AT': 1522,
 'DE -> CH': 4378,
 'DE -> NL': 2803,
 'FR -> BE': 2689,
 'FR -> CH': 7665,
 'NL -> BE': 1403,
 'NL -> DE': 60}
```

The yearly energy balance per fuel or per technology is also useful to compare the energy mix in each zone. This can be plotted using the EnergyBarPlot function, with the following results:



## Model Description

The model is expressed as a MILP or LP problem. Continuous variables include the individual unit dispatched power, the shedded load and the curtailed power generation. The binary variables are the commitment status of each unit. The main model features can be summarized as follows:

## Variables

## Sets

Name	Description
f	Fuel types
h	Hours
i	Time step in the current optimization horizon
l	Transmission lines between nodes
mk	{ DA: Day-Ahead, 2U: Reserve up, 2D: Reserve Down }
n	Zones within each country (currently one zone, or node, per country)
p	Pollutants
t	Power generation technologies
tr	Renewable power generation technologies
u	Units
s	Storage units (including hydro reservoirs)
chp(u)	CHP units

## Parameters

Name	Units	Description
AvailabilityFactor(u,i)	%	Percentage of nominal capacity available
CHPPowerLossFactor(u)	%	Power loss when generating heat
CHPPowerToHeat(u)	%	Nominal power-to-heat factor
CHPMaxHeat(chp)	MW	Maximum heat capacity of chp plant
CHPType	n.a.	CHP Type
CommittedInitial(u)	n.a.	Initial commitment status
CostFixed(u)	EUR/h	Fixed costs
CostLoadShedding(n,h)	EUR/MWh	Shedding costs
CostRampDown(u)	EUR/MW	Ramp-down costs
CostRampUp(u)	EUR/MW	Ramp-up costs
CostShutDown(u)	EUR/h	Shut-down costs
CostStartUp(u)	EUR/h	Start-up costs
CostVariableH(u,i)	EUR/MWh	Variable costs
Curtailment(n)	n.a.	Curtailment {binary: 1 allowed}
Demand(mk,n,i)	MW	Hourly demand in each zone
Efficiency(u)	%	Power plant efficiency
EmissionMaximum(n,p)	EUR/tP	Emission limit per zone for pollutant p
EmissionRate(u,p)	tP/MW	Emission rate of pollutant p from unit u
FlexibilityDown(u)	MW/h	Available fast shut-down ramping capacity
FlexibilityUp(u)	MW/h	Available fast start-up ramping capacity
Fuel(u,f)	n.a.	Fuel type used by unit u {binary: 1 u uses f}
LineNode(l,n)	n.a.	Line-zone incidence matrix {-1,+1}
LoadMaximum(u,h)	%	Maximum load for each unit
LoadShedding(n,h)	MW	Load that may be shed per zone in 1 hour
Location(u,n)	n.a.	Location {binary: 1 u located in n}
OutageFactor(u,h)	%	Outage factor (100 % = full outage) per hour
PartLoadMin(u)	%	Percentage of minimum nominal capacity
PowerCapacity(u)	MW	Installed capacity
PowerInitial(u)	MW	Power output before initial period
PowerMinStable(u)	MW	Minimum power for stable generation
PowerMustRun(u)	MW	Minimum power output

Continued on next page

Table 4.1 – continued from previous page

Name	Units	Description
PriceTransmission(l,h)	EUR/MWh	Price of transmission between zones
RampDownMaximum(u)	MW/h	Ramp down limit
RampShutDownMaximum(u)	MW/h	Shut-down ramp limit
RampStartUpMaximum(u)	MW/h	Start-up ramp limit
RampUpMaximum(u)	MW/h	Ramp up limit
Reserve(t)	n.a.	Reserve provider {binary}
StorageCapacity(s)	MWh	Storage capacity (reservoirs)
StorageChargingCapacity(s)	MW	Maximum charging capacity
StorageChargingEfficiency(s)	%	Charging efficiency
StorageDischargeEfficiency(s)	%	Discharge efficiency
StorageInflow(s,h)	MWh	Storage inflows
StorageInitial(s)	MWh	Storage level before initial period
StorageMinimum(s)	MWh	Minimum storage level
StorageOutflow(s,h)	MWh	Storage outflows (spills)
StorageProfile(u,h)	MWh	Storage long-term level profile
Technology(u,t)	n.a.	Technology type {binary: 1: u belongs to t}
TimeDownInitial(u)	h	Hours down before initial period
TimeDownLeftInitial(u)	h	Time down remaining at initial time
TimeDownLeftJustStopped(u,i)	h	Time down remaining if started at time i
TimeDownMinimum(u)	h	Minimum down time
TimeDown(u,h)	h	Number of hours down
TimeUpInitial(u)	h	Number of hours up before initial period
TimeUpLeftInitial(u)	h	Time up remaining at initial time
TimeUpLeftJustStarted(u,i)	h	Time up remaining if started at time i
TimeUpMinimum(u)	h	Minimum up time
TimeUp(u,h)	h	Number of hours up
VOLL ()	EUR/MWh	Value of lost load

## Optimization Variables

Name	Units	Description
Committed(u,h)	n.a.	Unit committed at hour h {1,0}
CostStartUpH(u,h)	EUR	Cost of starting up
CostShutDownH(u,h)	EUR	Cost of shutting down
CostRampUpH(u,h)	EUR	Ramping cost
CostRampDownH(u,h)	EUR	Ramping cost
CurtailedPower(n,h)	MW	Curtailed power at node n
Flow(l,h)	MW	Flow through lines
Heat(chp,h)	MW	Heat output by chp plant
HeatSlack(chp,h)	MW	Heat satisfied by other sources
MaxRamp2U(u,h)	MW/h	Maximum 15-min Ramp-up capability
MaxRamp2D(u,h)	MW/h	Maximum 15-min Ramp-down capability
Power(u,h)	MW	Power output
PowerMaximum(u,h)	MW	Power output
PowerMinimum(u,h)	MW	Power output
ShedLoad(n,h)	MW	Shed load
StorageInput(s,h)	MWh	Charging input for storage units
StorageLevel(s,h)	MWh	Storage level of charge
Spillage(s,h)	MWh	Spillage from water reservoirs
SystemCostD	EUR	Total system cost for one optimization period
LostLoadMaxPower(n,h)	MW	Deficit in terms of maximum power
LostLoadRampUp(u,h)	MW	Deficit in terms of ramping up for each plant
LostLoadRampDown(u,h)	MW	Deficit in terms of ramping down
LostLoadMinPower(n,h)	MW	Power exceeding the demand
LostLoadReserve2U(n,h)	MW	Deficit in reserve up

## Optimisation model

The aim of this model is to represent with a high level of detail the short-term operation of large-scale power systems solving the so-called unit commitment problem. To that aim we consider that the system is managed by a central operator with full information on the technical and economic data of the generation units, the demands in each node, and the transmission network.

The unit commitment problem considered in this report is a simplified instance of the problem faced by the operator in charge of clearing the competitive bids of the participants into a wholesale day-ahead power market. In the present formulation the demand side is an aggregated input for each node, while the transmission network is modelled as a transport problem between the nodes (that is, the problem is network-constrained but the model does not include the calculation of the optimal power flows).

The unit commitment problem consists of two parts: i) scheduling the start-up, operation, and shut down of the available generation units, and ii) allocating (for each period of the simulation horizon of the model) the total power demand among the available generation units in such a way that the overall power system costs is minimized. The first part of the problem, the unit scheduling during several periods of time, requires the use of binary variables in order to represent the start-up and shut down decisions, as well as the consideration of constraints linking the commitment status of the units in different periods. The second part of the problem is the so-called economic dispatch problem, which determines the continuous output of each and every generation unit in the system. Therefore, given all the features of the problem mentioned above, it can be naturally formulated as a mixed-integer linear program (MILP).

Since our goal is to model a large European interconnected power system, we have implemented a so-called tight and compact formulation, in order to simultaneously reduce the region where the solver searches for the solution and increase the speed at which the solver carries out that search. Tightness refers to the distance between the relaxed and integer solutions of the MILP and therefore defines the search space to be explored by the solver, while compactness is

related to the amount of data to be processed by the solver and thus determines the speed at which the solver searches for the optimum. Usually tightness is increased by adding new constraints, but that also increases the size of the problem (decreases compactness), so both goals contradict each other and a trade-off must be found.

## Objective function

The goal of the unit commitment problem is to minimize the total power system costs (expressed in EUR in equation ), which are defined as the sum of different cost items, namely: start-up and shut-down, fixed, variable, ramping, transmission-related and load shedding (voluntary and involuntary) costs.

$$\begin{aligned}
\min \sum_{u,n,i} & \left[ \text{CostStartUp}_{u,i} + \text{CostShutDown}_{u,i} + \text{CostFixed}_u \cdot \text{Committed}_{u,i} \right. \\
& + \text{CostVariable}_{u,i} \cdot \text{Power}_{u,i} + \text{CostRampUp}_{u,i} + \text{CostRampDown}_{u,i} \\
& + \text{PriceTransmission}_{i,l} \cdot \text{Flow}_{i,l} + (\text{CostLoadShedding}_{i,n} \cdot \text{ShedLoad}_{i,n}) \\
& + \sum_{chp} \text{CostHeatSlack}_{chp,i} \cdot \text{HeatSlack}_{chp,i} \\
& + \sum_{chp} \text{CostVariable}_{chp,i} \cdot \text{CHPPowerLossFactor}_{chp} \cdot \text{Heat}_{chp,i} \\
& + \text{VOLL}_{Power} \cdot (\text{LostLoadMaxPower}_{i,n} + \text{LostLoadMinPower}_{i,n}) \\
& + \text{VOLL}_{Reserve} \cdot (\text{LostLoadReserve2U}_{i,n} + \text{LostLoadReserve2D}_{i,n}) \\
& \left. + \text{VOLL}_{Ramp} \cdot (\text{LostLoadRampUp}_{u,i} + \text{LostLoadRampDown}_{u,i}) \right]
\end{aligned}$$

The costs can be broken down as:

- Fixed costs: depending on whether the unit is on or off.
- Variable costs: stemming from the power output of the units.
- Start-up costs: due to the start-up of a unit.
- Shut-down costs: due to the shut-down of a unit.
- Ramp-up: emerging from the ramping up of a unit.
- Ramp-down: emerging from the ramping down of a unit.
- Load shed: due to necessary load shedding.
- Transmission: depending of the flow transmitted through the lines.
- Loss of load: power exceeding the demand or not matching it, ramping and reserve.

The variable production costs (in EUR/MWh), are determined by fuel and emission prices corrected by the efficiency (which is considered to be constant for all levels of output in this version of the model) and the emission rate of the unit (equation ):

$$\begin{aligned}
& \text{CostVariable}_{u,h} = \\
& \text{Markup}_{u,h} + \sum_{n,f} \left( \frac{\text{Fuel}_{u,f} \cdot \text{FuelPrice}_{n,f,h} \cdot \text{Location}_{u,n}}{\text{Efficiency}_u} \right) \\
& + \sum_p (\text{EmissionRate}_{u,p} \cdot \text{PermitPrice}_p)
\end{aligned}$$

The variable cost includes an additional mark-up parameter that can be used for calibration and validation purposes.



The start-up and shut-down costs are positive variables, active when the commitment status between two consecutive time periods is modified:

$$\begin{aligned}
 i = 1 : \\
 & CostStartUp_{u,i} \geq CostStartUp_u \cdot (Committed_{u,i} - CommittedInitial_u) \\
 & CostShutDown_{u,i} \geq CostShutDown_u \cdot (CommittedInitial_u - Committed_{u,i}) \\
 i > 1 : \\
 & CostStartUp_{u,i} \geq CostStartUp_u \cdot (Committed_{u,i} - Committed_{u,i-1}) \\
 & CostShutDown_{u,i} \geq CostShutDown_u \cdot (Committed_{u,i-1} - Committed_{u,i})
 \end{aligned}$$

In the previous equation, as in some of the following, a distinction is made between the equation for the first and subsequent periods. The equation for the first period takes into account the commitment status of the unit before the beginning of the simulation, which is part of the information fed into the model.

Ramping costs are computed in the same manner:

$$\begin{aligned}
 i = 1 : \\
 & CostRampUp_{u,i} \geq CostRampUp_u \cdot (Power_{u,i} - PowerInitial_u) \\
 & CostRampDown_{u,i} \geq CostRampDown_u \cdot (PowerInitial_u - Power_{u,i}) \\
 i > 1 : \\
 & CostRampUp_{u,i} \geq CostRampUp_u \cdot (Power_{u,i} - Power_{u,i-1}) \\
 & CostRampDown_{u,i} \geq CostRampDown_u \cdot (Power_{u,i-1} - Power_{u,i})
 \end{aligned}$$

It should be noted that in case of start-up and shut-down, the ramping costs are added to the objective function. Using start-up, shut-down and ramping costs at the same time should therefore be performed with care.

In the current formulation all other costs (fixed and variable costs, transmission costs, load shedding costs) are considered as exogenous parameters.

As regards load shedding, the model considers the possibility of voluntary load shedding resulting from contractual arrangements between generators and consumers. Additionally, in order to facilitate tracking and debugging of errors, the model also considers some variables representing the capacity the system is not able to provide when the minimum/maximum power, reserve, or ramping constraints are reached. These lost loads are a very expensive last resort of the system used when there is no other choice available. The different lost loads are assigned very high values (with respect to any other costs). This allows running the simulation without infeasibilities, thus helping to detect the origin of the loss of load. In a normal run of the model, without errors, all these variables are expected to be equal to zero.

### Demand-related constraints

The main constraint to be met is the supply-demand balance, for each period and each zone, in the day-ahead market (equation ). According to this restriction, the sum of all the power produced by all the units present in the node (including the power generated by the storage units), the power injected from neighbouring nodes, and the curtailed power from intermittent sources is equal to the load in that node, plus the power consumed for energy storage, minus the load interrupted and the load shed.

$$\begin{aligned}
 & \sum_u (Power_{u,i} \cdot Location_{u,n}) \\
 & + \sum_l (Flow_{l,i} \cdot LineNode_{l,n}) \\
 & = Demand_{DA,n,h} + \sum_r (StorageInput_{s,h} \cdot Location_{s,n}) \\
 & \quad - ShedLoad_{n,i} \\
 & \quad - LostLoadMaxPower_{n,i} + LostLoadMinPower_{n,i}
 \end{aligned}$$

Besides that balance, the reserve requirements (upwards and downwards) in each node must be met as well. In DispaSET, the reserve requirements are defined as an aggregation of secondary and tertiary reserves, which are typically brought online in periods shorter than an hour, the time step of this model. Therefore, additional equations and constraints are defined to account for the up/down ramping requirements, by computing the ability of each unit to adapt its power output within a period of 15 min.

For each power plant, the ability to increase its power (in MW/h) is the ramp-up capability if it is already committed or the nominal power if it is stopped and its starting time is lower than 15 minutes. This is to take into account that fast starting units could provide reserve (hydro units for secondary reserve, gas turbine for tertiary reserve).

$$\begin{aligned} &MaxRamp2U_{u,i} \\ &\leq RampUpMaximum_u \cdot Committed_{u,i} \\ &\quad + FlexibilityUp_u \cdot (1 - Committed_{u,i}) \end{aligned}$$

where FlexibilityUp is the maximum flexibility (in MW/h) that can be provided by the unit in 15 min in case of cold start:

$$\begin{aligned} &If RampStartUpMaximum_u \geq PowerMinStable_u \cdot 4 \\ &\quad Then FlexibilityUp_u = RampStartUpMaximum_u \\ &\quad Else FlexibilityUp_u = 0 \end{aligned}$$

where the factor 4 is used to convert the ramping rate from MW/15min to MW/h.

The maximum ramping rate is also limited by the available capacity margin between current and maximum power output:

$$\begin{aligned} MaxRamp2U_{u,i} &\leq (PowerCapacity_u \cdot AvailabilityFactor_{u,i} \\ &\quad \cdot (1 - OutageFactor_{u,i}) - Power_{u,i}) \cdot 4 \end{aligned}$$

The same applies to the 15 min ramping down capabilities:

$$\begin{aligned} &MaxRamp2D_{u,i} \\ &\leq \max(RampDownMaximum_u, FlexibilityDown_u) \cdot Committed_{u,i} \end{aligned}$$

The parameter FlexibilityDown is defined as the maximum ramp down rate at which the unit can shut down in 15 minutes. In case the unit cannot be shut-down in 15 minutes (and only in this case) the maximum ramping down capability is limited by the capacity margin between actual and minimum power:

$$\begin{aligned} &If RampShutDownMaximum_u < PowerMinStable_u \cdot 4 : \\ &MaxRamp2D_{u,i} \leq (Power_{u,i} - PowerMinStable_u \cdot Committed_{u,i}) \cdot 4 \\ &Else : \\ &MaxRamp2D_{u,i} \leq Power_{u,i} \cdot 4 \end{aligned}$$

The reserve requirements are defined by the users. In case no input is provided a default formula is used to evaluate the needs for secondary reserves as a function of the maximum expected load for each day. The default formula is described by:

$$Demand_{2U,n,i} = \sqrt{10 \cdot \max_h (Demand_{DA,n,h}) + 150^2} - 150$$

Downward reserves are defined as 50% of the upward margin:

$$Demand_{2D,n,h} = 0.5 \cdot Demand_{2U,n,h}$$

The reserve demand should be fulfilled at all times by all the plants allowed to participate in the reserve market:

$$\begin{aligned}
 & Demand_{2U,n,h} \\
 & \leq \sum_{u,t} (MaxRamp_{2U,u,i} \cdot Technology_{u,t} \cdot Reserve_t \cdot Location_{u,n}) \\
 & \quad + LostLoadReserve_{2UH_{n,i}}
 \end{aligned}$$

The same equation applies to downward reserve requirements (2D).

### Power output bounds

The minimum power output is determined by the must-run or stable generation level of the unit if it is committed:

$$\begin{aligned}
 & PowerMustRun_{u,i} \cdot Committed_{u,i} \\
 & \leq Power_{u,i}
 \end{aligned}$$

On the other hand, the output is limited by the available capacity, if the unit is committed:

$$\begin{aligned}
 & Power_{u,i} \\
 & \leq PowerCapacity_u \cdot AvailabilityFactor_{u,i} \\
 & \quad \cdot (1 - OutageFactor_{u,i}) \cdot Committed_{u,i}
 \end{aligned}$$

The availability factor is used for renewable technologies to set the maximum time-dependent generation level. It is set to one for the traditional power plants. The outage factor accounts for the share of unavailable power due to planned or unplanned outages.

The power output in a given period also depends on the output levels in the previous and the following periods and on the ramping capabilities of the unit. If the unit was down, the ramping capability is given by the maximum start up ramp, while if the unit was online the limit is defined by the maximum ramp up rate. Those bounds are given with respect to the previous time step by the equation :

$$\begin{aligned}
 & i = 1 : \\
 & Power_{u,i} \leq \\
 & PowerInitial_u \\
 & \quad + CommittedInitial_u \cdot RampUpMaximum_u \\
 & \quad + (1 - CommittedInitial_u) \cdot RampStartUpMaximum_u \\
 & \quad + LostLoadRampUp_{u,i} \\
 & i > 1 : \\
 & Power_{u,i} \leq \\
 & Power_{u,i-1} \\
 & \quad + Committed_{u,i-1} \cdot RampUpMaximum_u \\
 & \quad + (1 - Committed_{u,i-1}) \cdot RampStartUpMaximum_u \\
 & \quad + LostLoadRampUp_{u,i}
 \end{aligned}$$

Where the LoadMaximum parameter is calculated taking into account the availability factor and the outage factor:

$$LoadMaximum_{u,h} = AvailabilityFactor_{u,h} \cdot (1 - OutageFactor_{u,h})$$

Similarly, the ramp down capability is limited by the maximum ramp down or the maximum shut down ramp rate:

$$\begin{aligned}
& i = 1 : \\
& \quad PowerInitial_u - Power_{u,i} \leq \\
& \quad Committed_{u,i} \cdot RampDownMaximum_u \\
& + (1 - Committed_{u,i}) \cdot RampShutDownMaximum_u \\
& \quad + LostLoadRampDown_{u,i} \\
& i > 1 : \\
& \quad Power_{u,i-1} - Power_{u,i} \leq \\
& \quad Committed_{u,i} \cdot RampDownMaximum_u \\
& + (1 - Committed_{u,i}) \cdot RampShutDownMaximum_u \\
& \quad + LostLoadRampDown_{u,i}
\end{aligned}$$

### Minimum up and down times

The operation of the generation units is also limited as well by the amount of time the unit has been running or stopped. In order to avoid excessive ageing of the generators, or because of their physical characteristics, once a unit is started up, it cannot be shut down immediately. Reciprocally, if the unit is shut down it may not be started immediately.

That is, the value of the time counter with respect to the minimum up time and down times determines the commitment status of the unit. In order to model these constraints linearly, it is necessary to keep track of the number of hours the unit must be online at the beginning of the simulation:

$$\begin{aligned}
& TimeUpLeftInitial_u = \\
& \min \{N, (TimeUpMinimum_u - TimeUpInitial_u) \cdot CommittedInitial_u\}
\end{aligned}$$

where N is the number of time steps in the current optimisation horizon.

If the unit is initially started up, it has to remain committed until reaching the minimum up time:

$$\sum_{i=1}^{TimeUpLeftInitial_u} (1 - Committed_{u,i}) = 0$$

If the unit is started during the considered horizon, the time it has to remain online is TimeUpMinimum, but cannot exceed the time remaining in the simulated period. This is expressed in equation and is pre-calculated for each time step of the period.

$$\begin{aligned}
& TimeUpLeftJustStarted_{u,i} = \\
& \min \{N - i + 1, TimeUpMinimum_u\}
\end{aligned}$$

The equation imposing the unit to remain committed is written:

$$\begin{aligned}
& i = 1 : \\
& \quad \sum_{ii=i}^{i+TimeUpLeftJustStarted_{u,i}-1} Committed_{u,ii} \geq \\
& \quad TimeUpLeftJustStarted_{u,i} \cdot (Committed_{u,i} - CommittedInitial_u) \\
& i > 1 : \\
& \quad \sum_{ii=i}^{i+TimeUpLeftJustStarted_{u,i}-1} Committed_{u,ii} \geq \\
& \quad TimeUpLeftJustStarted_{u,i} \cdot (Committed_{u,i} - Committed_{u,i-1})
\end{aligned}$$

The same method can be applied to the minimum down time constraint:

$$\begin{aligned} TimeDownLeft_u = \\ \min\{N, (TimeDownMinimum_u - TimeDownInitial_u) \\ \cdot (1 - CommittedInitial_u)\} \end{aligned}$$

Related to the initial status of the unit:

$$\sum_{i=1}^{TimeDownLeft_u} Committed_{u,i} = 0$$

The TimeDownLeftJustStopped parameter is computed by:

$$\begin{aligned} TimeDownLeftJustStopped_{u,i} = \\ \min\{N - i + 1, TimeDownMinimum_u\} \end{aligned}$$

Finally, the equation imposing the time the unit has to remain de-committed is defined as:

$$\begin{aligned} i = 1 : \\ i + TimeDownLeftJustStopped_{i,u} - 1 \\ \sum_{ii=i} (1 - Committed_{u,i}) \geq \\ TimeDownLeftJustStopped_{u,i} \cdot (CommittedInitial_u - Committed_{u,i}) \\ i > 1 : \\ i + TimeDownLeftJustStopped_u - 1 \\ \sum_{ii=i} (1 - Committed_{u,i}) \geq \\ TimeDownLeftJustStopped_{u,i} \cdot (Committed_{u,i-1} - Committed_{u,i}) \end{aligned}$$

This formulation avoids the use of additional binary variables to describe the start-up and shut-down of each unit.

### Storage-related constraints

Generation units with energy storage capabilities (mostly large hydro reservoirs and pumped hydro storage units) must meet additional restrictions related to the amount of energy stored. Storage units are considered to be subject to the same constraints as non-storage power plants. In addition to those constraints, storage-specific restrictions are added for the set of storage units (i.e. a subset of all units). These restrictions include the storage capacity, inflow, outflow, charging, charging capacity, charge/discharge efficiencies, etc. Discharging is considered as the standard operation mode and is therefore linked to the Power variable, common to all units.

The first constrain imposes that the energy stored by a given unit is bounded by a minimum value:

$$StorageMinimum_s \leq StorageLevel_{s,i}$$

In the case of a storage unit, the availability factor applies to the charging/discharging power, but also to the storage capacity. The storage level is thus limited by:

$$StorageLevel_{s,i} \leq StorageCapacity_s \cdot AvailabilityFactor_{s,i}$$

The energy added to the storage unit is limited by the charging capacity. Charging is allowed only if the unit is not producing (discharging) at the same time (i.e. if Committed, corresponding to the `normal` mode, is equal to 0).

$$\begin{aligned} StorageInput_{s,i} \leq StorageChargingCapacity_s \\ \cdot AvailabilityFactor_{s,i} \cdot (1 - Committed_{s,i}) \end{aligned}$$

Discharge is limited by the level of charge of the storage unit:

$$\begin{aligned} & \frac{Power_{i,s}}{StorageDischargeEfficiency_s} + StorageOutflow_{s,i} \\ & + Spillage_{s,i} - StorageInflow_{s,i} \\ & \leq StorageLevel_{s,i} \end{aligned}$$

Charge is limited by the level of charge of the storage unit:

$$\begin{aligned} & StorageInput_{s,i} \cdot StorageChargingEfficiency_s \\ & - StorageOutflow_{s,i} - Spillage_{s,i} \\ & + StorageInflow_{s,i} \\ & \leq StorageCapacity_s - StorageLevel_{s,i} \end{aligned}$$

Besides, the energy stored in a given period is given by the energy stored in the previous period, net of charges and discharges:

$$\begin{aligned} & i = 1 : \\ & StorageLevelInitial_s + StorageInflow_{s,i} \\ & + StorageInput_{s,i} \cdot StorageChargingEfficiency_s \\ & = StorageLevel_{s,i} + StorageOutflow_{s,i} + \frac{Power_{s,i}}{StorageDischargeEfficiency_s} \\ & i > 1 : \\ & StorageLevel_{s,i-1} + StorageInflow_{s,i} \\ & + StorageInput_{s,i} \cdot StorageChargingEfficiency_s \\ & = StorageLevel_{s,i} + StorageOutflow_{s,i} + \frac{Power_{s,i}}{StorageDischargeEfficiency_s} \end{aligned}$$

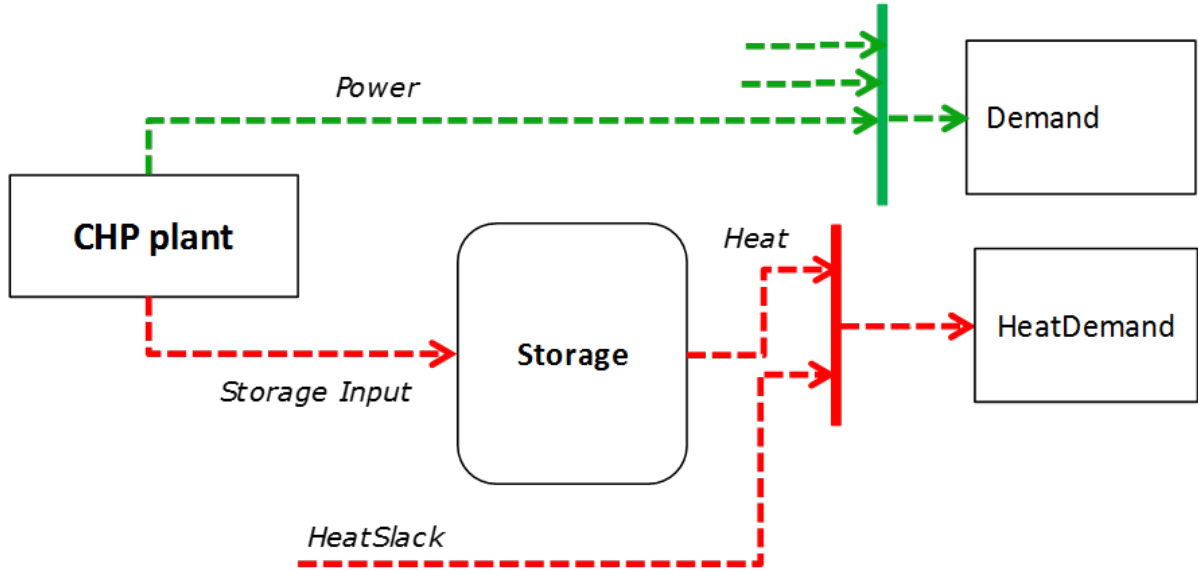
Some storage units are equipped with large reservoirs, whose capacity at full load might be longer than the optimisation horizon. Therefore, a minimum level constraint is required for the last hour of the optimisation, which otherwise would systematically tend to empty the reservoir as much as possible. An exogenous minimum profile is thus provided and the following constraint is applied:

$$\begin{aligned} & StorageLevel_{s,N} \geq \min(StorageProfile_{s,N} \\ & \cdot AvailabilityFactor_{s,N} \cdot StorageCapacity_s, \\ & StorageLevel_{s,0} + \sum_{i=1}^N InFlows_{s,i}) \end{aligned}$$

where StorageProfile is a non-dimensional minimum storage level provided as an exogenous input. The minimum is taken to avoid unfeasibilities in case the provided inflows are not sufficient to comply with the imposed storage level at the end of the horizon.

### Heat production constraints (CHP plants only)

In DispaSET Power plants can be indicated as CHP satisfying one heat demand. Heat Demand can be covered either by a CHP plant or by alternative heat supply options (Heat Slack).



The following two heat balance constraints are used for any CHP plant type.

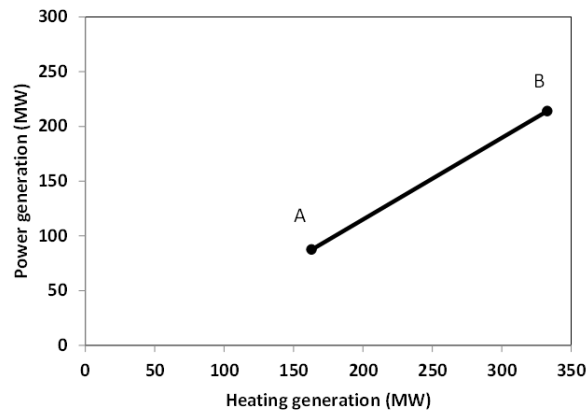
$$Heat(chp, i) + HeatSlack(chp, i) = HeatDemand(chp, i)$$

$$StorageInput_{chp,i} \leq CHPMaxHeat_{chp}$$

The constraints between heat and power production differ for each plant design and explained within the following subsections.

#### Steam plants with Backpressure turbine

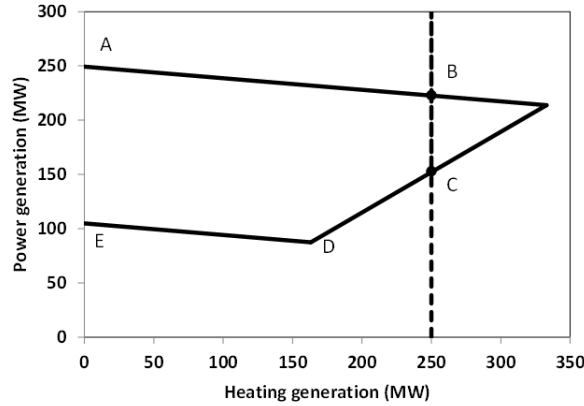
This options includes steam-turbine based power plants with a backpressure turbine. The feasible operating region is between AB. The slope of the line is the heat to power ratio.



$$Power_{chp,i} = StorageInput_{chp,i} \cdot CHPPowerToHeat_{chp}$$

### Steam plants with Extraction/condensing turbine

This options includes steam-turbine based power plants with an extraction/condensing turbine. The feasible operating region is within ABCDE. The vertical dotted line BC corresponds to the minimum condensation line (as defined by  $CHPMaxHeat$ ). The slope of the DC line is the heat to power ratio and the slope of the AB line is the inverse of the power penalty ratio.

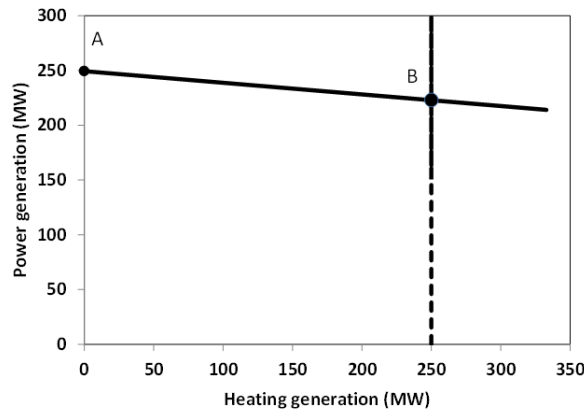


$$Power_{chp,i} \geq StorageInput_{chp,i} \cdot CHPPowerToHeat_{chp}$$

$$Power_{chp,i} \leq PowerCapacity_{chp} - StorageInput_{chp,i} \cdot CHPPowerLossFactor_{chp}$$

### Power plant coupled with any power to heat option

This option includes power plants coupled with resistance heater or heat pumps. The feasible operating line is between AB. The slope of the line is the inverse of the COP or efficiency. The vertical dotted line corresponds to the heat pump (or resistance heater) thermal capacity (as defined by  $CHPMaxHeat$ )



$$Power_{chp,i} = PowerCapacity_{chp} - StorageInput_{chp,i} \cdot CHPPowerLossFactor_{chp}$$



## Heat Storage

Heat storage is modeled in a similar way as electric storage as follows:

Heat Storage balance:

$$\begin{aligned}
 i = 1 : \\
 & StorageInitial_{chp} + StorageInput_{chp,i} = \\
 & StorageLevel_{chp,i} + Heat_{chp,i} + StorageSelfDischarge_{chp} \cdot StorageLevel_{chp,i}/24 \\
 i > 1 : \\
 & + StorageLevel_{chp,i-1} + StorageInput_{chp,i} = \\
 & StorageLevel_{chp,i} + Heat_{chp,i} + StorageSelfDischarge_{chp} \cdot StorageLevel_{chp,i}/24
 \end{aligned}$$

Storage level must be above a minimum and below storage capacity:

$$StorageMinimum_{chp} \leq StorageLevel_{chp,i} \leq StorageCapacity_{chp}$$

## Emission limits

The operating schedule also needs to take into account any cap on the emissions (not only CO2) from the generation units existing in each node:

$$\begin{aligned}
 \sum_u (Power_{u,i} \cdot EmissionRate_{u,p} \cdot Location_{u,n}) \\
 \leq EmissionMaximum_{n,p}
 \end{aligned}$$

It is important to note that the emission cap is applied to each optimisation horizon: if a rolling horizon of one day is adopted for the simulation, the cap will be applied to all days instead of the whole year.

## Network-related constraints

The flow of power between nodes is limited by the capacities of the transmission lines:

$$\begin{aligned}
 FlowMinimum_{l,i} &\leq Flow_{l,i} \\
 Flow_{l,i} &\leq FlowMaximum_{l,i}
 \end{aligned}$$

In this model a simple Net Transfer Capacity (NTC) between countries approach is followed. No DC power flow or Locational Marginal Pricing (LMP) model is implemented.

## Curtailment

If curtailment of intermittent generation sources is allowed in one node, the amount of curtailed power is bounded by the output of the renewable (tr) units present in that node:

$$\begin{aligned}
 & CurtailedPower_{n,i} \\
 & \leq \sum_{u,tr} (Power_{u,i} \cdot Technology_{u,tr} \cdot Location_{u,n}) \cdot Curtailment_n
 \end{aligned}$$

## Load shedding

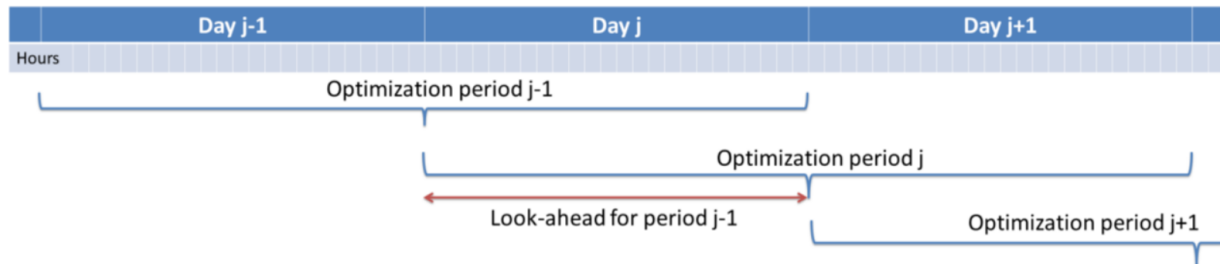
If load shedding is allowed in a node, the amount of shed load is limited by the shedding capacity contracted on that particular node (e.g. through interruptible industrial contracts)

$$ShedLoad_{n,i} \leq LoadShedding_n$$

## Rolling Horizon

The mathematical problem described in the previous sections could in principle be solved for a whole year split into time steps of one hour, but with all likelihood the problem would become extremely demanding in computational terms when attempting to solve the model with a realistically sized dataset. Therefore, the problem is split into smaller optimization problems that are run recursively throughout the year.

The following figure shows an example of such approach, in which the optimization horizon is one day, with a look-ahead (or overlap) period of one day. The initial values of the optimization for day  $j$  are the final values of the optimization of the previous day. The look-ahead period is modelled to avoid issues related to the end of the optimization period such as emptying the hydro reservoirs, or starting low-cost but non-flexible power plants. In this case, the optimization is performed over 48 hours, but only the first 24 hours are conserved.



Although the previous example corresponds to an optimization horizon and an overlap of one day, these two values can be adjusted by the user in the Dispa-SET configuration file. As a rule of thumb, the optimization horizon plus the overlap period should be at least twice the maximum duration of the time-dependent constraints (e.g. the minimum up and down times). In terms of computational efficiency, small power systems can be simulated with longer optimization horizons, while larger systems should reduce this horizon, the minimum being one day.

## Power plant clustering

For computational efficiency reasons, it is useful to cluster some of the original units into larger units. This reduces the number of continuous and binary variables and can, in some conditions, be performed without significant loss of simulation accuracy.

The clustering occurs at the beginning of the pre-processing phase (i.e. the units in the Dispa-SET database do not need to be clustered).

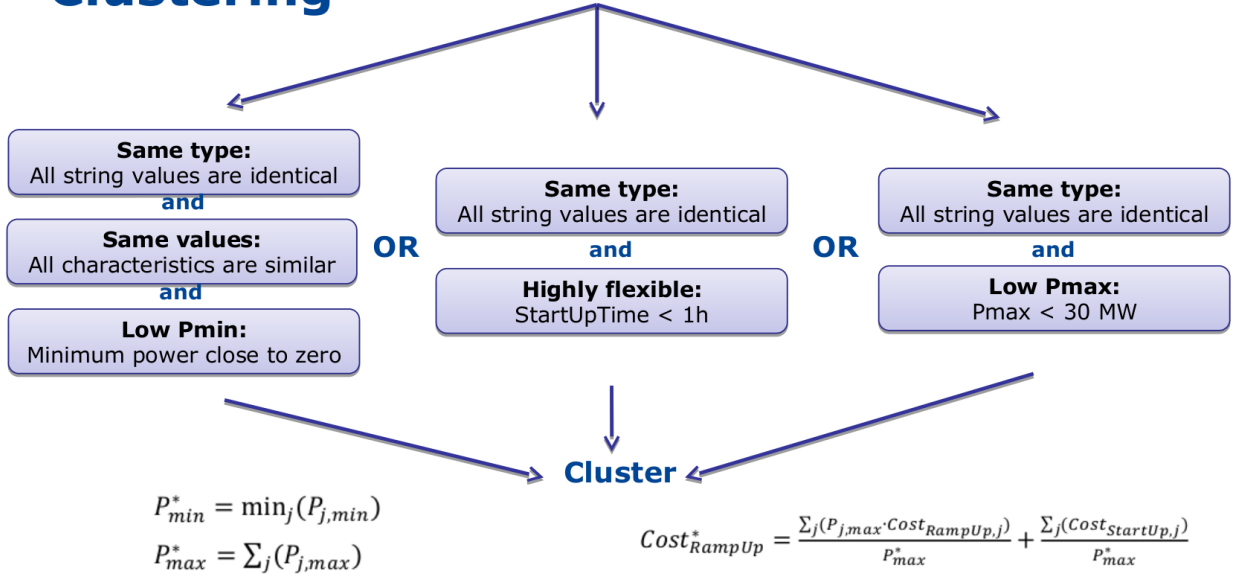
In Dispa-SET, different clustering options are available and can be automatically generated from the same input data. They are described in the two next sections.

### MILP clustering

In this formulation, the units that are either very small or very flexible are aggregated into larger units. Some of these units (e.g. the turbojets) indeed present a low capacity or a high flexibility: their output power does not exceed a few MW and/or they can reach full power in less than 15 minutes (i.e. less than the simulation time step). For these units, a unit commitment model with a time step of 1 hour is unnecessary and computationally inefficient. They are therefore merged into one single, highly flexible unit with averaged characteristics.

The condition for the clustering of two units is a combination of subconditions regarding their type, maximum power, flexibility and technical similarities. They are summarized in the figure below (NB: the thresholds are for indicative purpose only, they can be user-defined).

## Clustering



When two units are clustered, the minimum and maximum capacities of new aggregated units (indicated by the star) are given by:

$$P_{min}^* = \min(P_{j,min})$$

$$P_{max}^* = \sum_j(P_{j,max})$$

The last equation is also applied for the storage capacity or for the storage charging power.

The unit marginal (or variable cost) is given by:

$$Cost_{Variable}^* = \frac{\sum_j(P_{j,max} \cdot Cost_{Variable,j})}{P_{max}^*}$$

The start-up/shut-down costs are transformed into ramping costs (example with ramp-up):

$$Cost_{RampUp}^* = \frac{\sum_j(P_{j,max} \cdot Cost_{RampUp,j})}{P_{max}^*} + \frac{\sum_j(Cost_{StartUp,j})}{P_{max}^*}$$

Other characteristics, such as the plant efficiency, the minimum up/down times or the CO2 emissions are computed as a weighted averaged:

$$Efficiency^* = \frac{\sum_j(P_{j,max} \cdot Efficiency_j)}{P_{max}^*}$$

It should be noted that only very similar units are aggregated (i.e. their quantitative characteristics should be similar), which avoids errors due to excessive aggregation.

### LP clustering

Dispa-SET provides the possibility to generate the optimisation model as an LP problem (i.e. without the binary variables). In that case, the following constraints are removed since they can only be expressed in an MILP formulation:

- Minimum up and down times

- Start-up costs
- Minimum stable load

Since the start-up of individual units is not considered anymore, it is not useful to disaggregate them in the optimisation. All units of a similar technology, fuel and zone can be aggregated into a single unit using the equations proposed in the previous section.

## References

## Implementation and interface

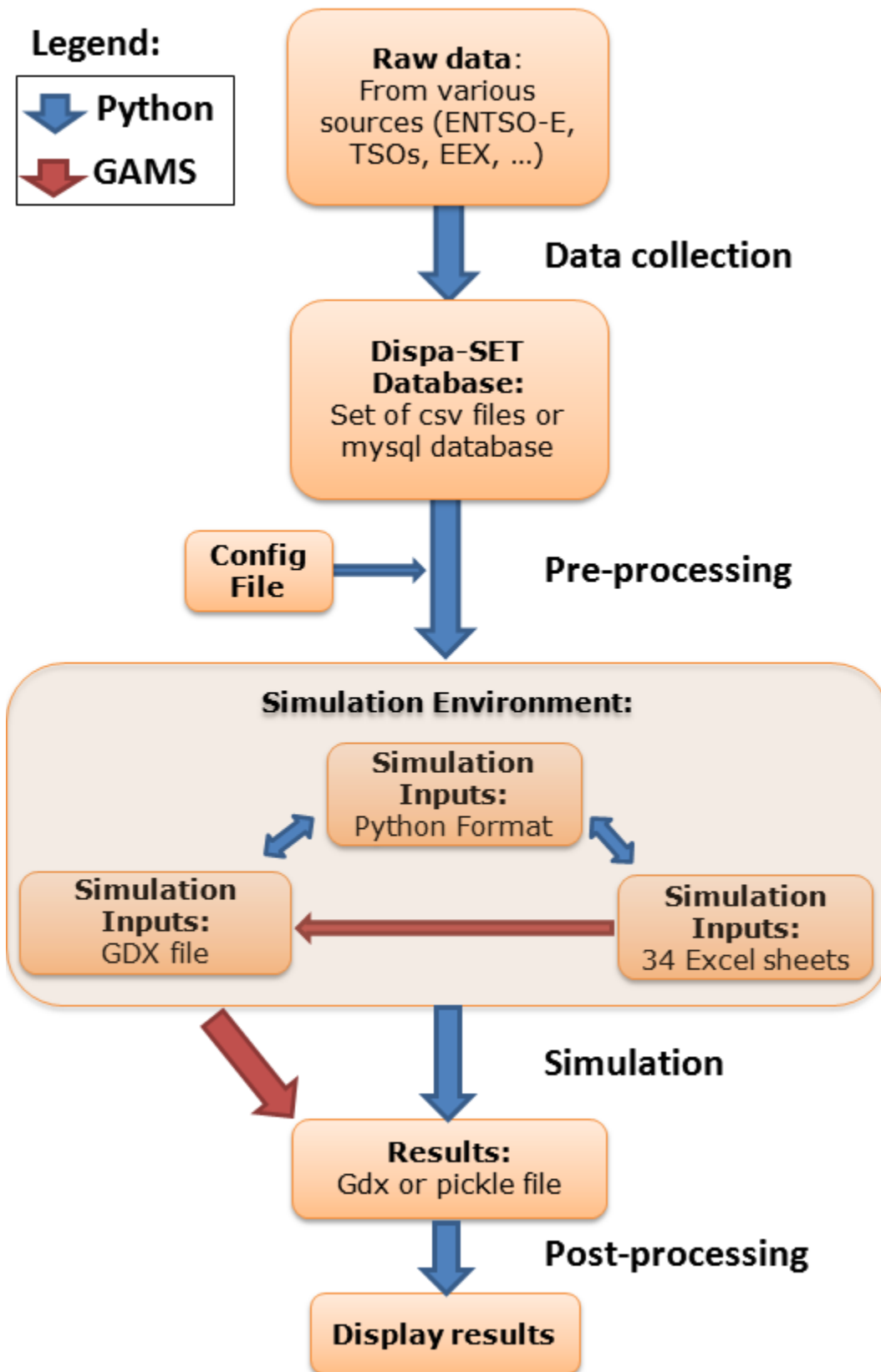
The typical step-by-step procedure to parametrize and run a DispaSET simulation is the following:

1. Fill the Dispa-SET database with properly formatted data (time series, power plant data, etc.)
2. Configure the simulation parameters (rolling horizon, data slicing) in the configuration file.
3. Generate the simulation environment which comprises the inputs of the optimisation
4. Open the GAMS simulation files (project: UCM.gpr and model: UCM\_h.gms) and run the model.
5. Read and display the simulation results.

This section provides a detailed description of these steps and the corresponding data entities.

## Resolution Flow Chart

The whole resolution process for a dispa-SET run is defined from the processing and formatting of the raw data to the generation of aggregated result plots and statistics. A flow chart of the consecutive data entities and processing steps is provided hereunder.



Each box in the flow chart corresponds to one data entity. The links between these data entities correspond to script written in Python or in GAMS. The different steps perform various tasks, which can be summarized by:

**1. Data collection:**

- Read csv sheets, assemble data
- Convert to the right format (timestep, units, etc).
- Define proper time index (duplicates not allowed)
- Connect to database
- Check if data present & write data
- Write metadata

**2. Pre-processing:**

- Read the config file
- Slice the data to the required time range
- Deal with missing data
- Check data for consistency (min up/down times, startup times, etc.)
- Calculate variable cost for each unit
- Cluster units
- Define scenario according to user inputs (curtailment, participation to reserve, amount of VRE, amount of storage, ...)
- Define initial state (basic merit-order dispatch)
- Write the simulation environment to a user-defined folder

**3. Simulation environment and interoperability:**

- **Self-consistent folder with all required files to run the simulation:**
  - Excel files
  - GDX file
  - Input files in pickle format
  - Gams model files
- Python scripts to translate the data between one format to the other.
- Possibility to modify the inputs manually and re-generate a GDX file from the excel files

**4. Simulation:**

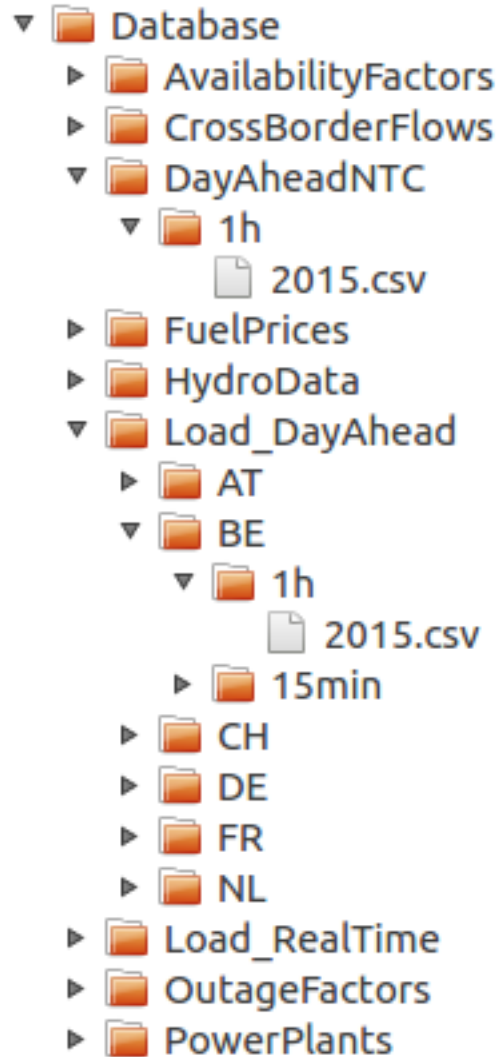
- The GAMS simulation file is run from the simulation environment folder
- Alternatively the model is run with the PYOMO solver
- All results and inputs are saved within the simulation environment

**5. Post-processing:**

- Reads the simulation results saved in the simulation environment
- Aggregates the power generation and storage curves
- Computes of yearly statistics
- Generates plots

## Dispa-SET database

Although two versions of the database are available (mysql and csv), the public version of Dispa-SET only comes with the latter. The Dispa-SET input data is stored as csv file in directory structure. A link to the required data is then provided by the user in the configuration file.



The above figure shows a partially unfolded view of the database structure. In that example, data is provided for the day-ahead net transfer capacities for all lines in the EU, for the year 2015 and with a 1h time resolution. Time series are also provided for the day-ahead load forecast for Belgium in 2015 with 1h time resolution.

## Configuration File

The excel config file is read at the beginning of the pre-processing phase. It provides general inputs for the simulation as well as links to the relevant data files in the database.

## Simulation environment

This section describes the different simulation files, templates and scripts required to run the DispaSET model. For each simulation, these files are included into a single directory corresponding to a self-sufficient simulation environ-

ment.

A more comprehensive description of the files included in the simulation environment directory is provided hereunder.

### **UCM\_h.gms and UCM.gpr**

UCM\_h.gms is the main GAMS model described in Chapter 1. A copy of this file is included in each simulation environment, allowing keeping track of the exact version of the model used for the simulation. The model must be run in GAMS and requires a proper input file (Inputs.gdx).

Requires:	Inputs.gdx	Input file for the simulation.
Generates:	Results.gdx	Simulation results in.gdx format
.	Results.xlsx	Simulation results in.xlsx format.

UCM.gpr is the GAMS project file which should be opened before UCM\_h.gms.

### **make\_gdx.gms**

GAMS file that reads the different template excel files and generates the Inputs.gdx file. This file should be opened in GAMS.

Requires:	InputDispa-SET – xxx.xlsx	DispaSET template files
Generates:	Inputs.gdx	Input file for the simulation

### **makeGDX.bat**

Batch script that generates the input file from the template without requiring opening GAMS. The first time it is executed, the path of the GAMS folder must be provided.

Requires:	InputDispa-SET – xxx.xlsx	DispaSET template files
.	make_gdx.gms	GAMS file to generate Inputs.gdx
Generates:	Inputs.gdx	Input file for the simulation

### **writeresults.gms**

GAMS file to generate the excel Results.xlsx file from the Results.gdx generated by GAMS (in case the write\_excel function was deactivated in GAMS).

Requires:	Results.gdx	Simulation results in.gdx format
Generates:	Results.xlsx	Simulation results in.xlsx format

### **Inputs.gdx**

All the inputs of the model must be stored in the Inputs.gdx file since it is the only file read by the main GAMS model. This file is generated from the DispaSET template.

Requires:	InputDispa-SET – xxx.xlsx	DispaSET template files
Generates:		

### **InputDispa-SET - [ParameterName].xlsx**

Series of 42 excel files, each corresponding to a parameter of the DispaSET model (see Chapter 1). The files must be formatted according to section 2.2.



### InputDispa-SET - Sets.xlsx

Single excel file that contains all the sets used in the model in a column format.

### InputDispa-SET - Config.xlsx

Single excel file that contains simulation metadata in the form of a Table. This metadata allows setting the rolling horizon parameter and slicing the input data to simulate a subset only.

Table 4.2: Config

FirstDay	2012	10	1	First day of the simulation in the template data
LastDay	2013	9	30	Last day of the simulation in the template data
RollingHorizon Length	0	0	3	Length of the rolling horizons
RollingHorizon LookAhead	0	0	1	Overlap period of the rolling horizon

### Structure of the Excel template

The name of the input files are “Input Dispa-SET – [Parameter name].xlsx”. These files contain the data to be read by the model, after conversion into a GDX file.

The structure of all input files follows the following rules:

1. There is one file per model parameter
2. Each file contains only one sheet
3. The first row is left blank for non-time series data (i.e. data starts at A2)
4. **For time series data, the rows are organized as follows:**
  - (a) The first row is left blank
  - (b) Rows 2 to 5 contains the year, month, day and hour of each data
  - (c) Row 6 contains the time index of the data, which will be used in DispaSET
  - (d) The data therefore starts at A6
5. If one of the input sets of the data is u (the unit name), it is always defined as the first column of the data (column A)
6. If one of the input sets of the data is h (the time index), it is always defined as the only horizontal input in row 6

In the case of the file “Input Dispa-SET – Sets.xlsx”, all the required sets are written in columns with the set name in row 2.

### Post-processing

Post-processing is implemented in the form of a series of functions to read the simulation inputs and results, to plot them, and to derive statistics.

The following values are computed:

- The total energy generated by each fuel, in each country.
- The total energy curtailed
- the total load shedding

- The overall country balance of the interconnection flows
- The total hours of congestion in each interconnection line
- The total amount of lost load, indicating (if not null) that the unit commitment problem was unfeasible at some hours
- The number of start-ups of power plants for each fuel

The following plots can be generated:

- A dispatch plot (by fuel type) for each country
- A commitment status (ON/OFF) plot for all the unit in a given country
- The level (or state of charge) of all the storage units in a given country
- The overall power generation by fuel type for all countries (bar plot)

An example usage of these funciones is provided in the “Read\_Results.ipynb” notebook.

## Input Data

In this section, “Input Data” refers to the data stored in the Dispa-SET database. The format of this data is pre-defined and imposed, in such a way that it can be read by the pre-processing tool.

Two important preliminary comments should be formulated:

- All the time series should be registered with their timestamps (e.g. ‘2013-02-20 02:00:00’) relative to the UTC timezone.
- Although the optimisation model is designed to run with any technology or fuel name, the pre-processing and the post-processing tools of Dispa-SET use some hard-coded values. The Dispa-SET database should also comply with this convention (described in the next sections). Any non-recognized technology or fuel will be discarded in the pre-processing.

## Technologies

The Dispa-SET input distinguishes between the technologies defined in the table below. The VRES column indicates the variable renewable technologies (set “tr” in the optimisation) and the Storage column indicates the technologies which can accumulate energy.

Table 4.3: Dispa-SET technologies

Technology	Description	VRES	Storage
COMC	Combined cycle	N	N
GTUR	Gas turbine	N	N
HDAM	Conventional hydro dam	N	Y
HROR	Hydro run-of-river	Y	N
HPHS	Pumped hydro storage	N	Y
ICEN	Internal combustion engine	N	N
PHOT	Solar photovoltaic	Y	N
STUR	Steam turbine	N	N
WTOF	Offshore wind turbine	Y	N
WTON	Onshore wind turbine	Y	N
CAES	Compressed air energy storage	N	Y
BATS	Stationary batteries	N	Y
BEVS	Battery-powered electric vehicles	N	Y
THMS	Thermal storage	N	Y
P2GS	Power-to-gas storage	N	Y

## Fuels

Dispa-SET only considers a limited number of different fuel types. They are summarised in the following table, together with some examples.

Table 4.4: Dispa-SET fuels

Fuel	Examples
BIO	Bagasse, Biodiesel, Gas From Biomass, Gasification, Biomass, Briquettes, Cattle Residues, Rice Hulls Or Padi Husk, Straw, Wood Gas (From Wood Gasification), Wood Waste Liquids Excl Blk Liq (Incl Red Liquor, Sludge, Wood, Spent Sulfite Liquor And Oth Liquids, Wood And Wood Waste
GAS	Blast Furnace Gas, Boiler Natural Gas, Butane, Coal Bed Methane, Coke Oven Gas, Flare Gas, Gas (Generic), Methane, Mine Gas, Natural Gas, Propane, Refinery Gas, Sour Gas, Synthetic Natural Gas, Top Gas, Voc Gas & Vapor, Waste Gas, Wellhead Gas
GEO	Geothermal steam
HRD	Anthracite, Other Anthracite, Bituminous Coal, Coker By-Product, Coal Gas (From Coal Gasification), Coke, Coal (Generic), Coal-Oil Mixture, Other Coal, Coal And Pet Coke Mi, Coal Tar Oil, Anthracite Coal Waste, Coal-Water Mixture, Gob, Hard Coal / Anthracite, Imported Coal, Other Solids, Soft Coal, Anthracite Silt, Steam Coal, Subbituminous, Pelletized Synthetic Fuel From Coal, Bituminous Coal Waste)
HYD	Hydrogen
LIG	Lignite black, Lignite brown, lignite
NUC	U, Pu
OIL	Crude Oil, Distillate Oil, Diesel Fuel, No. 1 Fuel Oil, No. 2 Fuel Oil, No. 3 Fuel Oil, No. 4 Fuel Oil, No. 5 Fuel Oil, No. 6 Fuel Oil, Furnace Fuel, Gas Oil, Gasoline, Heavy Oil Mixture, Jet Fuel, Kerosene, Light Fuel Oil, Liquefied Propane Gas, Methanol, Naphtha, Gas From Fuel Oil Gasification, Fuel Oil, Other Liquid, Orimulsion, Petroleum Coke, Petroleum Coke Synthetic Gas, Black Liquor, Residual Oils, Re-Refined Motor Oil, Oil Shale, Tar, Topped Crude Oil, Waste Oil
PEA	Peat Moss
SUN	Solar energy
WAT	Hydro energy
WIN	Wind energy
WST	Digester Gas (Sewage Sludge Gas), Gas From Refuse Gasification, Hazardous Waste, Industrial Waste, Landfill Gas, Poultry Litter, Manure, Medical Waste, Refused Derived Fuel, Refuse, Waste Paper And Waste Plastic, Refinery Waste, Tires, Agricultural Waste, Waste Coal, Waste Water Sludge, Waste

Different fuels may be used to power a given technology, e.g. steam turbines may be fired with almost any fuel type. In Dispa-SET, each unit must be defined with the pair of values (technology,fuel). The next tables is derived from a commercial power plant database and indicates the number of occurrences of each combination. It appears clearly that, even through some combinations are irrelevant, both characteristics are needed to define a power plant type.

f/t	COMC	GTUR	HDAM	HPHS	HROR	ICEN	PHOT	STUR	WTOF	WTON	To- tal
BIO		2				10		79			91
GAS	485	188				28		97			798
GEO								10			10
HRD	4							389			393
HYD		1						1			2
LIG								249			249
NUC								138			138
OIL	7	94				27		146			274
PEA								17			17
SUN							20	7			27
UNK		2				1		1			4
WAT			33	23	21			1			78
WIN									9	27	36
WST		3				7		46			56
To- tal	496	290	33	23	21	73	20	1181	9	27	2173

## Unit-specific or technology-specific inputs

Some parameters, such as the availability factor, the outage factor or the inflows may be defined at the unit level or at the technology level. For that reason, the pre-processing tool first lookups the unit name in the database to assign it a value, and then lookups the technology if no unit-specific information has been found.

## Demand

Electricity demand is given per zone/country and the first row of each column should reflect that name.

Heat demand timeseries is needed where CHP plants are used. In the current formulation, each CHP plant is covering a heat load. In other words, one power plant is connected to a single district heating network. Therefore, in the heat demand input file, the first column has to be a time index and the following columns the heat demand in MW. The first row should contain the exact name of the power plant that will cover this demand.

## Countries

Although the nodes names can be freely user-defined in the database, for the Dispa-SET EU model, the ISO 3166-1 standard has been adopted to describe each country at the NUTS1 level. The list of countries is defined as:

Code	Country
AT	Austria
BE	Belgium
BG	Bulgaria
CH	Switzerland
CY	Cyprus
Continued on next page	

Table 4.5 – continued from previous page

Code	Country
CZ	Czech Republic
DE	Germany
DK	Denmark
EE	Estonia
EL	Greece
ES	Spain
FI	Finland
FR	France
GB	Great Britain
HR	Croatia
HU	Hungary
IE	Ireland
IT	Italy
LT	Lithuania
LU	Luxembourg
LV	Latvia
MT	Malta
NL	Netherlands
NO	Norway
PL	Poland
PT	Portugal
RO	Romania
SE	Sweden
SI	Slovenia
SK	Slovakia

It should be noted that ‘UK’ (United Kingdom) has been replaced by ‘GB’ (Great Britain) in this list, i.e. northern Ireland is not considered and is instead included within the ‘IE’ node.

## Power plant data

The power plant database may contain as many fields as desired, e.g. to ensure that the input data can be traced back, or to provide the id of this plant in another database. However, some fields are required by Dispa-SET and must therefore be defined in the database.

### Common fields

The following fields must be defined for all units:

Table 4.6: Common fields for all units

Description	Field name	Units
Unit name	Unit	
Commissioning year	Year	
Technology	Technology	
Primary fuel	Fuel	
Zone	Zone	
Capacity	PowerCapacity	MW
Efficiency	Efficiency	%
Efficiency at minimum load	MinEfficiency	%
CO2 intensity	CO2Intensity	TCO2/MWh
Minimum load	PartLoadMin	%
Ramp up rate	RampUpRate	%/min
Ramp down rate	RampDownRate	%/min)
Start-up time	StartUPTime	h
Minimum up time	MinUpTime	h
Minimum down time	MinDownTime	h
No load cost	NoLoadCost	EUR/h
Start-up cost	StartUpCost	EUR
Ramping cost	RampingCost	EUR/MW
Presence of CHP	CHP	y/n

NB: the fields indicated with % as unit must be entered in a non-dimensional way (i.e. 90% should be written 0.9).

### Storage units

Some parameters must only be defined for the units equipped with storage. They can be left blank for all other units.

Table 4.7: Specific fields for storage units

Description	Field name	Units
Storage capacity	STOCapacity	MWh
Self-discharge rate	STOSelfDischarge	%/h
Maximum charging power	STOMaxChargingPower	MW
Charging efficiency	STOChargingEfficiency	%

In the case of a storage unit, the discharge efficiency should be assigned to the common field “Efficiency”. Similarly, the common field “PowerCapacity” is the nominal power in discharge mode.

### CHP units

Some parameters must only be defined for the units equipped with CHP. They can be left blank for all other units.

Table 4.8: Specific fields for CHP units

Description	Field name	Units
CHP Type	CHPType	extraction/back-pressure/p2h
Power-to-heat ratio	CHPPowerToHeat	•
Power Loss factor	CHPPowerLossFactor	•
Maximum heat production	CHPMaxHeat	MW(th)
Capacity of heat Storage	STOCapacity	MWh(th)
% of storage heat losses per timestep	STOSelfDischarge	%

In the current version of DispaSet three type of combined heat and power units are supported:

- Extraction/condensing units
- Backpressure units
- Power to heat

For each of the above configurations the following fields must be filled:

Table 4.9: Mandatory fields per type of CHP unit (X: mandatory, o:optional)

Description	Extraction	Backpressure	Power to heat
CHPType	X	X	X
CHPPowerToHeat	X	X	
CHPPowerLossFactor	X		X
CHPMaxHeat	o	o	X
STOCapacity	o	o	o
STOSelfDischarge	o	o	o

There are numerous data checking routines to ensure that all data provided is consistent.

**Warning:** For extraction/condensing CHP plants, the power plant capacity (*PowerCapacity*) must correspond to the nameplate capacity in the maximum heat and power mode. Internal DispaSet calculations will use the equivalent stand-alone plants capacity based on the parameters provided.

## Renewable generation

Variable renewable generation is defined as power generation from renewable source that cannot be stored: it is either fed to the grid or curtailed. The technologies falling under this definition are the ones described in the subset “tr” in the model definition.

The time-dependent generation of for these technologies must be provided as an exogenous time series in the form of an “availability factor”. The latter is defined as the proportion of the nominal power capacity that can be generated at each hour.

In the database, the time series are provided as column vectors with the technology name as header. After the pre-processing, an availability factor is attributed to each unit according to their technology. Non-renewable technologies are assigned an availability factor of 1.

## Storage and hydro data

Storage units are an extension of the regular units, including additional constraints and parameters. In the power plant table, four additional parameters are required: storage capacity (in MWh), self-discharge (in %/h), discharge power (in MW) and discharge efficiency (in %).

Some other parameters must be introduced in the form of time series in the “HydroData” section of the Dispa-SET database. There are described hereunder.

It should be noted that the nomenclature adopted for the modeling of storage units refers to the characteristics of hydro units with water reservoirs. However, these parameters (e.g. inflows, level) can easily be transposed to the case of alternative storage units such as batteries or CAES.

### Inflows

The Inflows are defined as the contribution of exogenous sources to the level (or state of charge) or the reservoir. They are expressed in MWh of potential energy. If the inflows are provided as m<sup>3</sup>/h, they must be converted.

The input to dispa-set is defined as “ScaledInflows”. It is the normalized values of the inflow with respect to the nominal power of the storage unit (in discharge mode). As an example, if the inflow value at a certain time is 100MWh/h and if the turbinning capacity of the hydro plant is 200 MW, the scaled inflow value must be defined as 0.5.

Scaled inflows should be provided in the form of time series with the unit name or the technology as columns header.

### Storage level

Because emptying the storage has a zero marginal cost, a non-constrained optimization tends to leave the storage completely empty at the end of the optimisation horizon. For that reason, a minimum storage level is imposed at the last hour of each horizon. In Dispa-SET, a typical optimisation horizon is a few days. The model is therefore not capable of optimising the storage level e.g. for seasonal variations. The minimum storage level at the last hour is therefore an exogenous input. It can be selected from a historical level or obtained from a long-term hydro scheduling optimization.

The level input in the Dispa-SET database is normalized with respect to the storage capacity: its minimum value is zero and its maximum is one.

### Variable capacity storage

In special cases, it might be necessary to simulate a storage unit whose capacity varies in time. A typical example is the simulation of the storage capacity provided by electric vehicles: depending on the time of the day, the connected battery capacity varies.

This special case can be simulated using the “AvailabilityFactor” input. In the case of a storage unit, reduces the available capacity by a factor varying from 0 to 1.

## Power plant outages

In the current version, Dispa-SET does not distinguish planned outages from unplanned outages. They are characterized for each unit by the “OutageFactor” parameter. This parameter varies from 0 (no outage) to 1 (full outage). The available unit power is thus given by its nominal capacity multiplied by (1-OutageFactor).

The outages are provided in the dedicated section of the Database for each unit. They consist of a time series with the unit name as columns header.



## Interconnections

Two case should be distinguished when considering interconnections:

- Interconnections occurring between the simulated zones
- Interconnections occurring between the simulated zones and the Rest of the World (RoW)

These two cases are addresses by two different datasets described here under.

### Net transfer capacities

Dispa-SET indogenously models the internal exchanges between countries (or zones) using a commercial net transfer caapcity (NTC). It does not consider (yet) DC power flows or more complex grid simulations.

Since the NTC values might vary in time, they must be supplied as time series, whose header include the origin country, the string ' -> ' and the destination country. As an example, the NTC from belgium to france must be provided with the header 'BE -> FR'.

Because NTCs are not necessarily symetrical, they must be provided in both directions (i.e. 'BE -> FR' and 'FR -> BE'. Non-provided NTCs are considered to be zero (i.e. no interconnection).

### Historical physical flows

In Dispa-SET, the flows between internal zones and the rest of the world cannot be modeled endogenously. They must be provided as exogenous inputs. These inputs are referred to as "Historical physical flows", although they can also be user-defined.

In the input table of historical flows, the headers are similar to those of the NTCs (ie. 'XX -> YY'). All flows occurring an internal zone of the simulation and outside zones are considered as external flows and summed up. As an example, the historical flows 'FR -> XX', 'FR -> YY' and 'FR -> ZZ' will be aggregated in to a single interconnection flow 'FR -> RoW' if XX, YY and ZZ are not simulated zones.

These aggregated historical flows are then imposed to the solver as exogenous inputs.

In Dispa-SET, the flows are defined as positive variables. For each zone, there will thus be a maximum of two vectors defining its exchanges with the rest of the world (e.g. 'FR -> RoW' and 'RoW -> FR').

As for the NTCs, undefined historical flows are considered to be zero, i.e. not provided any historical flows is equivalent to consider the system as islanded.

## Fuel Prices

Fuel prices vary both geographically and in time. They must therefore be provided as a time series for each simulated zone. One table is provided per fuel type, with as column header the zone to which it applies. If no header is provided, the fuel price is applied to all the simulated zones.

## Developers' section

### Folders organization

- GAMS code and scripts are included within the "GAMS-files" folder. The code should only be modified in that folder!
- Python code and scripts are included within the "DispaSET" folder.

- DispaSET configuration files are included within the “ConfigFiles” folder (one file per simulation).
- Input files for each country are stored in the “Database” folder
- Simulation directories can be written into the “Simulations” folder, which is not tracked by git

A sample Simulation Environment Folder is available at `SimulationReferenceTestCase`. The files in this folders are generated automatically by the pre-processing scripts. They should not be modified. The folder should be replaced and updated at every major Dispa-SET release.

By default, the pre-processing scripts are set to generate the simulation environment within the “Simulations” folder. This folder is excluded in `.gitignore` and can therefore be used for testing purposes.

## Math equations in the Docs

- To use online mathjax (default), there is nothing to do but displaying the equation requires an internet connection. In linux, the Makefile call to build with mathjax is written:

```
$(SPHINXBUILD) -b html $(ALLSPHINXOPTS) $(BUILDDIR)/html
```

In `conf.py`, the only math equation should be: `'sphinx.ext.mathjax'`

- To use pngmath (for Linux):

```
sudo apt-get install dvipng
```

In `conf.py`, add `'sphinx.ext.pngmath'` in the extensions

in Makefile: `$(SPHINXBUILD) -D pngmath_latex=latex -b html $(ALLSPHINXOPTS) $(BUILDDIR)/html`

- To use mathjax in offline mode, download the latest release from the mathjax github, copy it to `Docs/_static/`, and include it in the `conf.py`:

```
mathjax_path = "Mathjax/MathJax.js"
```

## Using Autodoc

The “API” section of the Docs uses the sphinx autodoc extension to scan the source code of Dispa-SET and display the relevant functions together with their description, parameters and outputs. In the Sphinx “`conf.py`”, the path to the source file must be added:

```
sys.path.insert(0, os.path.abspath('../DispaSET'))
```

If the API documentation is generated with sphinx-apidoc, from the Docs folder, use:

```
sphinx-apidoc -o . ../DispaSET/
```

Add a link to “DispaSET” in the table of content of `index.rst` and include the root folder in `conf.py`:

```
sys.path.insert(0, os.path.abspath('../'))
```

## Clone git repository to svn

- Install git svn

```
sudo apt-get install git-svn
```

- Create svn branch

```
git branch svn
```

- Connect to svn repository

```
git svn init -s --prefix=svn/ --username <user> https://joinup.ec.europa.eu/svn/dispaaset
```

- Checkout svn branch

```
git checkout svn
```

- Fetch remote content

```
git svn fetch
```

- Reset repository

```
git reset --hard remotes/svn/trunk
```

- Merge master into svn

```
git merge master
```

- Commit to the remote repository

```
git svn dcommit
```

The folder can finally optionally be deleted to avoid any confusion.

## Issue with the compilation of the GAMS API

First, check that the installed version of GAMS is the 64 bit. 32 bit versions tend to generated compatibility issues.

When the pre-compiled libraries do not work, they must be re-compiled from the GAMS apifile folder. In Windows, this generally raises the issue of the missing vcvarsall.bat file. If the issue persists after installing the Microsoft C++ compiler for Python 2.7, try the following:

1. Enter MSVC for Python command prompt
2. SET DISTUTILS\_USE\_SDK=1
3. SET MSSdk=1
4. python.exe gdxsetup.py install

## Public version of Dispa-SET

Because some input files are subject to intellectual property and copyrights, some folders available on the private repository cannot be uploaded to the public GitHub repository. The script DispaSync.sh in the root folder has been written to synchronize the subset of publicly available folders and files with an external folder. This folder can then be committed and pushed to the public repository.

By default, all file and folders are synchronized. In order to add a private path (to a file or to a folder), edit the DispaSync.sh file and add an entry to the “--exclude” argument.

The “rsync” software is required for the synchronization and must be installed on the local machine. The script can be run in any UNIX terminal (i.e. it cannot be run in Windows).

## DispaSET package

### Subpackages

#### DispaSET.misc package

##### Submodules

DispaSET.misc.colorstreamhandler module

DispaSET.misc.gdx\_handler module

DispaSET.misc.str\_handler module

##### Module contents

#### DispaSET.postprocessing package

##### Submodules

DispaSET.postprocessing.postprocessing module

##### Module contents

#### DispaSET.preprocessing package

##### Submodules

DispaSET.preprocessing.data\_check module

DispaSET.preprocessing.data\_handler module

DispaSET.preprocessing.preprocessing module

DispaSET.preprocessing.utils module

##### Module contents

#### DispaSET.pyomo package

##### Submodules

DispaSET.pyomo.model module

DispaSET.pyomo.utils module

**Module contents**

**Submodules**

**DispaSET.solve module**

**Module contents**

## Releases

Major stable releases:

- [Dispa-SET v2.2](#)
- [Dispa-SET v2.1](#)
- [Dispa-SET v2.0](#)

## Changelog

### Version 2.2

- Inclusion of CHP, power2heat and thermal storage (these new features can be tested by running the config file for Cyprus: 'ConfigFiles/ConfigCY.xlsx')
- Bug fixes
- Improved user interface

### Version 2.1

- Major refactoring of the folder structure
- New data included in the database
- Inclusion of the LP formulation (in addition to the MILP)

### Version 2.0

First public version of the Dispa-SET model.



---

## Indices and tables

---

- `genindex`
- `modindex`
- `search`