D2.3 How to use TradeRES optimization models database

Deliverable number:	*D2.3*
Work Package:	*WP2*
Lead Beneficiary:	*VTT*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 864276

Author(s) information (alphabetical)		
Name	Organisation	Email
Juha Kiviluoma	VTT	juha.kiviluoma@vtt.fi
Ni Wang	TNO	
Silke Johanndeiter	ENBW	

Acknowledgements/Contributions		
Name	Organisation	Email
-		

Document information

Version	Date	Dissemination Level	Description
1.0	18.11.2022	PU	Guide to using TradeRES database for optimization models

Review and approval

Prepared by	Reviewed by	Approved by
J. Kiviluoma, N. Wang, S. Jo-hanndeiter	Milos Cvetkovic	A. Estanqueiro

Executive Summary

The deliverable explains how to use the TradeRES optimization model databases to run the optimization models. It is mainly targeted at project internal use for those who have a need to understand how the scenarios have been built and how they could be modified for the purposes of the case studies.

Table of Contents

1. Introduction

TradeRES case studies are performed for five different geographical scopes. The Pan-European case study covers whole of EU, GB, Norway, Switzerland and the Balkans. Three country/region case studies: Netherlands, Germany and MIBEL with Portugal and Spain. All of these, except the local case study, which is distinct from the others, require inputs from the optimization models. WP2 has built the database in such a manner that the optimization models can run for the whole Europe, but if needed, also run for each of the country/region case studies separately. The optimization models could also be run in higher fidelity if the case studies improve the data presentation of those regions – either with the rest of Europe or separately.

This user guide aims to assist TradeRES modellers to understand and operate the optimization models. For that purpose, the current approach to input data is explained, the different data formats shortly explained (with in-depth explanations available in tool documentations), possible modifications to the scenario assumptions are introduced, and finally the results from the optimization models are outlined.

2. Common input database

TradeRES is attempting to soft-link two optimization models and four agent-based models in various ways. In the ideal case, all models could function from a common input database, which would mean that any edits to the input data would be automatically shared by every subsequent model. This would allow to freely build iterations between any of the models. While TradeRES has been working to achieve this, it has proven to be a very challenging goal. There are three main issues.

First, developing a shared ontology with shared vocabulary has been impractical – the models, especially the optimization models in relation to the agent-based models, have very different data requirements and there is much less commonality than originally anticipated.

Second, working from a common database is an extra burden for the modellers (even if it would eventually pay off as enhanced model-linking capability). It is much more convenient to work from the existing input data structures each model has, even though the data shared between the models needs to be entered multiple times. As a downside, there is an increased risk for erroneous and incomparable data.

Third, the data transformations required to enable running models from a common data format are formidable. Spine Toolbox, which assists in this task, did not have all necessary capabilities, which meant that part of the data transformations had to be done in code. As the common ontology was slow to develop, modelling began with model specific data formats and continued in that track in order to produce results in schedule. These inconveniences have been an important lesson for the further development of Spine Toolbox, which will continue in the new EU project Mopo. The required data transformations can be complex, but TradeRES has thought a lot about the extent of those complexities and how they should be resolved.

Nonetheless, TradeRES has soft-linked several models together as workflows that convert data between models often using Spine Toolbox as a conduit. Furthermore, the aim to have a common database to serve multiple models is still alive and may yet be realised towards the end of the project even if the initial results are produced with model specific data formats and workflows.

3. Backbone data format, workflow and changing scenario data

3.1 Data format and workflow

Backbone modelling framework is written in GAMS and it reads GAMS specific GDX files as input and outputs GDX files. The benefit of the filetype is a compressed binary data format which is very fast to read and write. However, it cannot be used directly by pre-processing and post-processing tool chains. Therefore, two methods for reading Backbone inputs have been developed: specially formatted Excel files or Spine Toolbox databases. Both can be converted to GDX files when the model is run.

In the TradeRES project, a Spine Toolbox workflow takes raw data from several Excel input files (developed by TradeRES Task 2.1). These are items numbered 1 in the workflow of Figure 1. Then data importers (items numbered 2) convert the data into a Spine Toolbox database.

Spine Toolbox databases are of the type "entity-attribute-value with classes and relationships" and specific classes have been developed to hold Backbone data (Figure 2). Objects (on the left) are the basic building blocks, for example nodes and units. Each object can hold parameters available to the object class (e.g., 'efficiency' for units). Relationships define the connections between the objects and contain the parameters specified for those relationships. Important relationships include grid__node__node to define transfer connections and grid__node__unit__io to define inputs and outputs of units. For example, 'existing_capacity' is can be defined for the output of the electricity generating units into a specific node. More information available in the Backbone article (Helistö et al. 2019) and in the wiki pages: https://gitlab.vtt.fi/backbone/backbone/-/wikis/home.

The Spine Toolbox data structure allows flexible construction of scenarios, since data items are divided into sets of alternatives that can be used in building of the scenarios (Figure 3). For example, alternative 'base-netherlands' contains data to run the base scenario for Netherlands. This alternative can be combined with the other 'base' alternatives to form the base scenario for the European case study. In addition, there are alternatives for the flexibilities and VRE shares that are required by the TradeRES scenarios as can be seen from Figure 3.

The result of workflow (items 5-9 in Figure 1) can then be executed with each scenario in a parallel process. The first step, items 4 and 5, converts Backbone data from the Spine database (item 3) into Backbone input files (gdx format). Item 6 is a script that executes Backbone twice: first for investments decisions using 5 representative weeks and then to schedule the system with full year time series using rolling optimization window. TradeRES relevant results are then imported to another Spine Toolbox database (item 8), which allows to deposit comparative results across scenarios in a single Excel file (item number 9 in Figure 1). This Excel file can then be tooled to serve as input data for the agent-based models.

Main input and output categories from Backbone are introduced in TradeRES D6.2. Furthermore, related data requirements are explained in more detail at https://gitlab.vtt.fi/backbone/backbone/-/wikis/home and in Helistö et al. (2019). The wiki-link also contains instructions how to start using Backbone. The TradeRES workflow (Figure 1) is available at https://github.com/TradeRES/TradeRES-Backbone-demo (it is Spine Toolbox project, so this project folder must be opened with Spine Toolbox to access the functionalities). There are some specific modifications to Backbone to run the TradeRES scenarios. TradeRES-Backbone-demo contains a submodule for Backbone as well as these TradeRES specific modifications (more specifically, they are in 'TradeRES' branch of Backbone repository: https://gitlab.vtt.fi/backbone/backbone/-/tree/TradeRES).

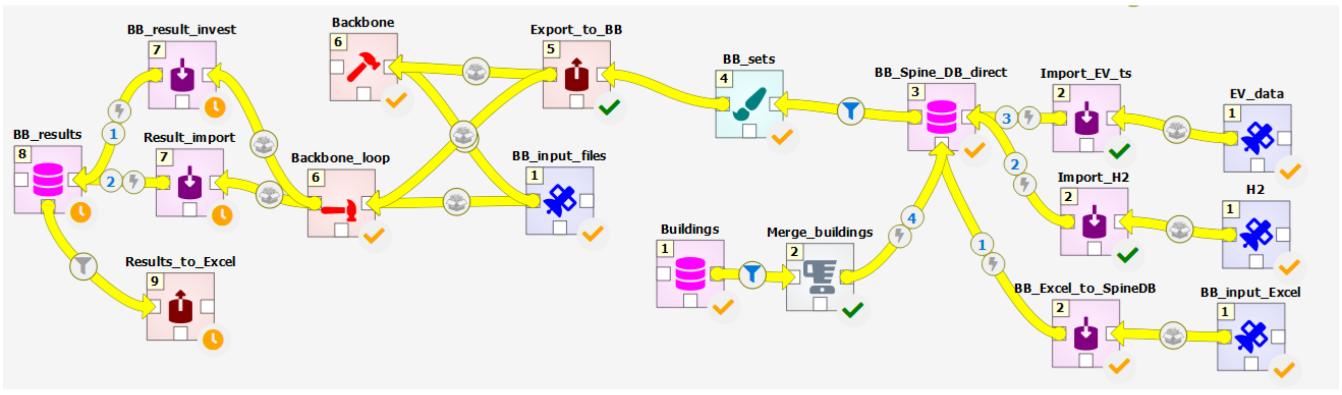
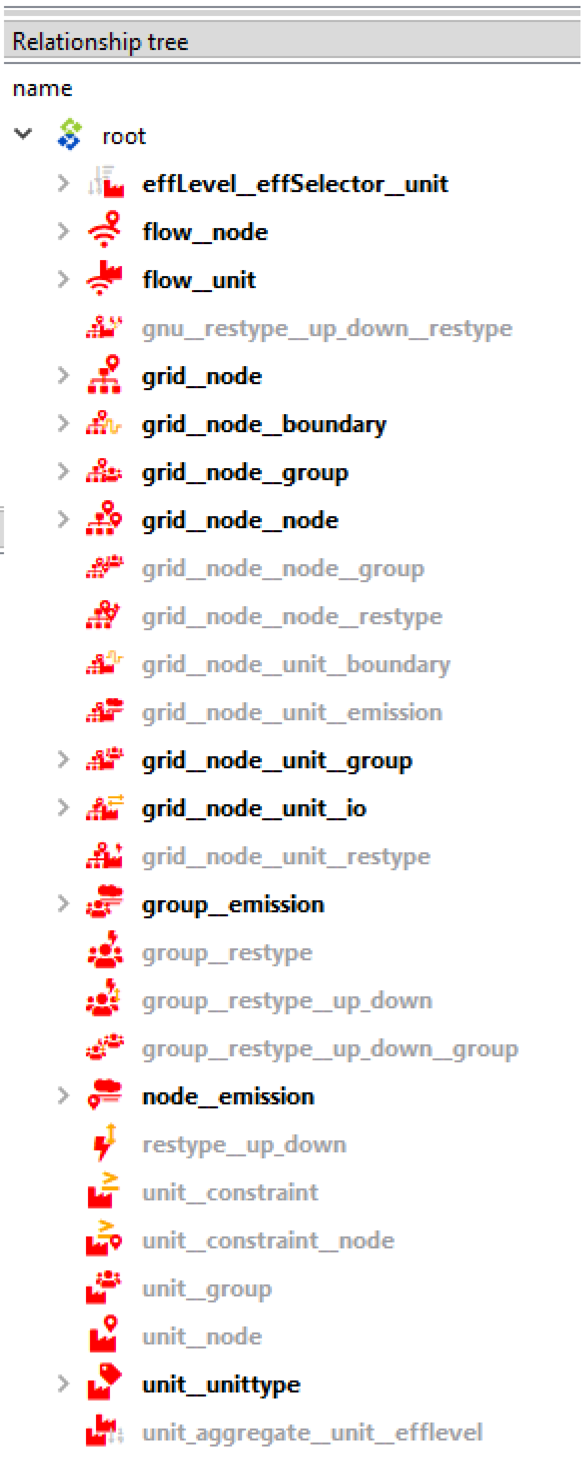


Figure 1. Spine Toolbox workflow for processing Backbone input data


Relationship tree

name

root

effLevel__effSelector__unit

flow__node

flow__unit

gnu__restype__up_down__restype

grid__node

grid__node__boundary

grid__node__group

grid__node__node

grid__node__node__group

grid__node__node__restype

grid__node__unit__boundary

grid__node__unit__emission

grid__node__unit__group

grid__node__unit__io

grid__node__unit__restype

group__emission

group__restype

group__restype__up_down

group__restype__up_down__group

node__emission

restype__up_down

unit__constraint

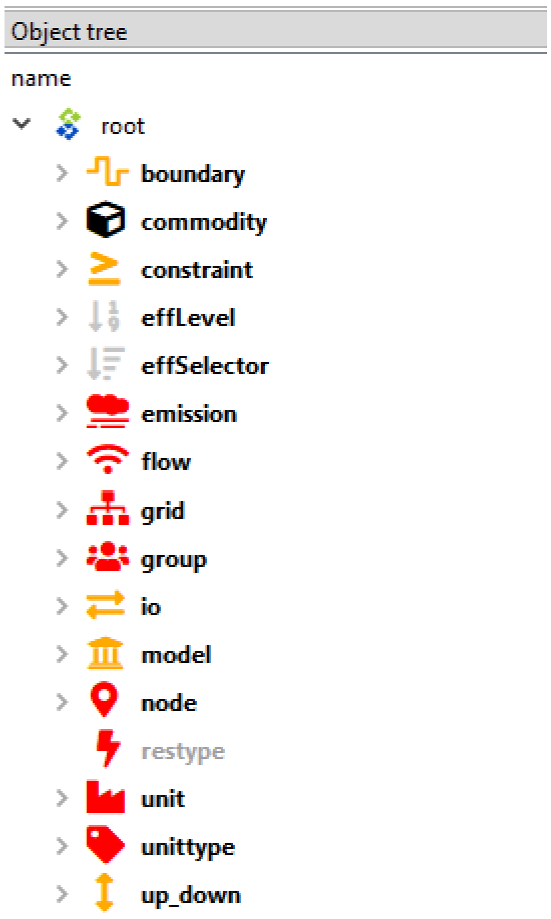
unit__constraint__node

unit__group

unit__node

unit__unittype

unit_aggregate__unit__efflevel


Object tree

name

root

boundary

commodity

constraint

effLevel

effSelector

emission

flow

grid

group

io

model

node

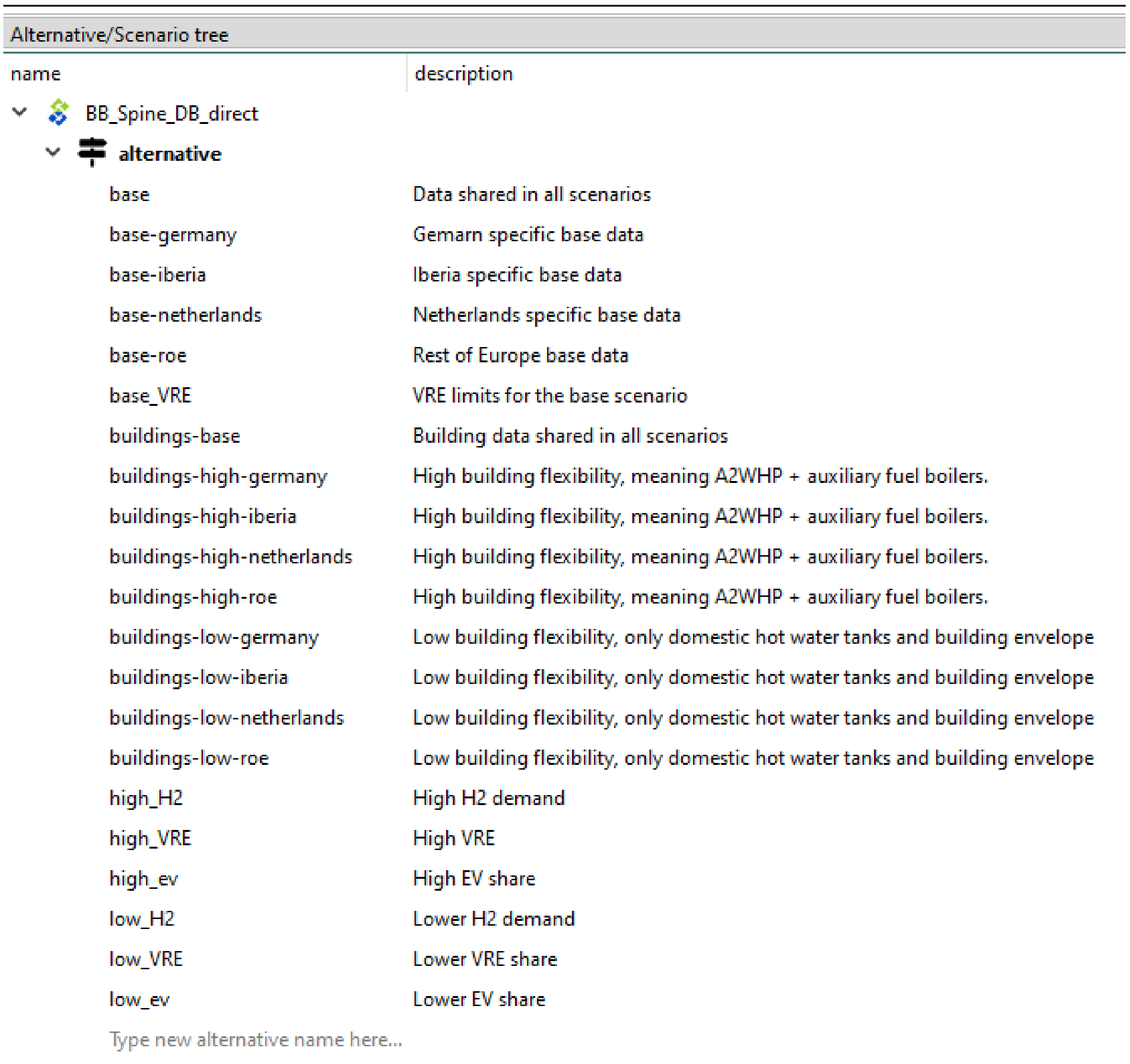
restype

unit

unittype

up_down

Figure 2. Object and relationship classes for Backbone data (classes with black text used by TradeRES runs).



Alternative/Scenario tree

name	description
BB_Spine_DB_direct	
alternative	
base	Data shared in all scenarios
base-germany	Gemarn specific base data
base-iberia	Iberia specific base data
base-netherlands	Netherlands specific base data
base-roe	Rest of Europe base data
base_VRE	VRE limits for the base scenario
buildings-base	Building data shared in all scenarios
buildings-high-germany	High building flexibility, meaning A2WHP + auxiliary fuel boilers.
buildings-high-iberia	High building flexibility, meaning A2WHP + auxiliary fuel boilers.
buildings-high-netherlands	High building flexibility, meaning A2WHP + auxiliary fuel boilers.
buildings-high-roe	High building flexibility, meaning A2WHP + auxiliary fuel boilers.
buildings-low-germany	Low building flexibility, only domestic hot water tanks and building envelope
buildings-low-iberia	Low building flexibility, only domestic hot water tanks and building envelope
buildings-low-netherlands	Low building flexibility, only domestic hot water tanks and building envelope
buildings-low-roe	Low building flexibility, only domestic hot water tanks and building envelope
high_H2	High H2 demand
high_VRE	High VRE
high_ev	High EV share
low_H2	Lower H2 demand
low_VRE	Lower VRE share
low_ev	Lower EV share
Type new alternative name here...	

Figure 3. Alternatives used to build TradeRES scenarios

3.2 Changing the main parameters in TradeRES scenarios

VRE share – Can be changed directly from the input database by selecting one of the alternatives that contain VRE share assumptions: ‘base_VRE’, ‘low_VRE’ and ‘high_VRE’. The parameters ‘energyShareMax’ and ‘energyShareMin’ can be used to limit the share of VRE (as calculated from the overall demand that includes also the flexible demands).

H2 demand – Alternatives ‘low_H2’ and ‘high_H2’ include time series for H2 demand that can be replaced (or a new alternative can be created using the existing ones as examples)

EV share – EVs require multiple time series and they are best updated in the original input data file where the share of EVs out of all vehicles can be easily changed. After changing the value, re-import the data into the input database (old values are automatically replaced).

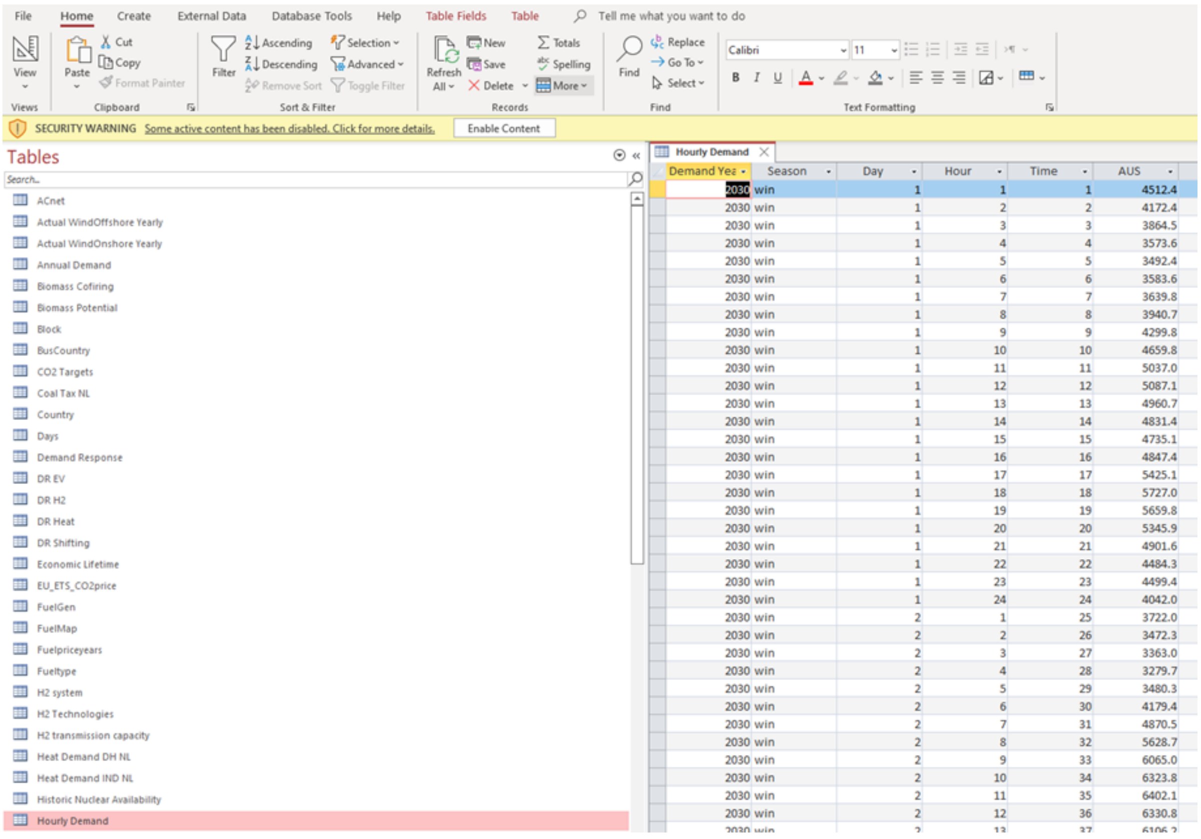
Building flexibility – the parameters and time series to represent buildings in the TradeRES Backbone model are based on a complicated data processing chain (partially published in https://cris.vtt.fi/en/publications/archetypebuildingmodeljl-a-julia-module-for-aggregating-building-, rest will be published in near future). Some of the parameters can be directly changed in the input database (e.g., conversion efficiencies) but others would re-quire re-running the whole data processing chain (e.g., structural assumptions for the build-ings).

4. COMPETES data format and workflow

4.1 Data format and workflow

COMPETES is written in AIMMS and reads data from MS Access database as input and writes in excel files as an output.

The input data is stored in an Access database (Figure 4). The advantage of the Access database is that it offers a tabular view of the data and can import external data from various data sources.



SECURITY WARNING Some active content has been disabled. Click for more details.

Tables

ACnet
Actual WindOffshore Yearly
Actual WindOnshore Yearly
Annual Demand
Biomass Cofiring
Biomass Potential
Block
BusCountry
CO2 Targets
Coal Tax NL
Country
Days
Demand Response
DR EV
DR H2
DR Heat
DR Shifting
Economic Lifetime
EU_ETS_CO2price
FuelGen
FuelMap
Fuelpriceyears
Fueltype
H2 system
H2 Technologies
H2 transmission capacity
Heat Demand DH NL
Heat Demand IND NL
Historic Nuclear Availability
Hourly Demand

Hourly Demand

Demand Yea	Season	Day	Hour	Time	AUS
2030	win	1	1	1	4512.4
2030	win	1	2	2	4172.4
2030	win	1	3	3	3864.5
2030	win	1	4	4	3573.6
2030	win	1	5	5	3492.4
2030	win	1	6	6	3583.6
2030	win	1	7	7	3639.8
2030	win	1	8	8	3940.7
2030	win	1	9	9	4299.8
2030	win	1	10	10	4659.8
2030	win	1	11	11	5037.0
2030	win	1	12	12	5087.1
2030	win	1	13	13	4960.7
2030	win	1	14	14	4831.4
2030	win	1	15	15	4735.1
2030	win	1	16	16	4847.4
2030	win	1	17	17	5425.1
2030	win	1	18	18	5727.0
2030	win	1	19	19	5659.8
2030	win	1	20	20	5345.9
2030	win	1	21	21	4901.6
2030	win	1	22	22	4484.3
2030	win	1	23	23	4499.4
2030	win	1	24	24	4042.0
2030	win	2	1	25	3722.0
2030	win	2	2	26	3472.3
2030	win	2	3	27	3363.0
2030	win	2	4	28	3279.7
2030	win	2	5	29	3480.3
2030	win	2	6	30	4179.4
2030	win	2	7	31	4870.5
2030	win	2	8	32	5628.7
2030	win	2	9	33	6065.0
2030	win	2	10	34	6323.8
2030	win	2	11	35	6402.1
2030	win	2	12	36	6330.8
2030	win	2	13	37	6106.2

Figure 4. A screenshot of the data of hourly demand in Access database

The COMPETES workflow is centred around AIMMS procedures. These procedures are pieces of the AIMMS code that perform a specific task that can be called either from the interface or the model itself. In COMPETES, procedures are used to do pre-processing (i.e., read inputs from the database and assign the values to the parameters in the model), solve the model, and post-processing (i.e., calculate specific results based on the outputs and assign the results to designated cells in the excel sheets). The procedures are part of the COMPETES model and must be used together with the main code for optimization.

It is possible to save the case data as a .data file that contains all the (raw) data in COMPETES in addition to the excel files. And it is recommended to save the data file once the model has generated outputs. The advantage of saving the data file is that it is then always possible to generate more results than what has been stored in the excel sheets.

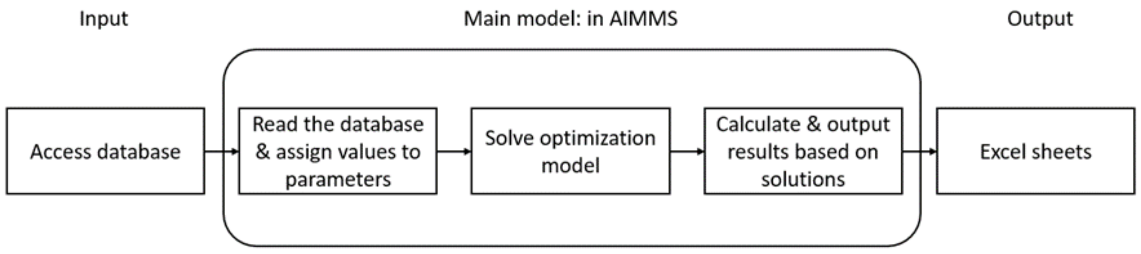

Input
Main model: in AIMMS
Output
Access database
Read the database & assign values to parameters
Solve optimization model
Calculate & output results based on solutions
Excel sheets

Figure 5. COMPETES workflow

4.2 Changing the main parameters for TradeRES scenarios

A standard workflow for changing the input is to prepare the TradeRES parameters in excel sheets using a designated template and import them to the Access database. The TradeRES parameters are taken from the common TradeRES database.

5. Common data structure for optimization model results

Both optimization models create a wide range of outputs. A subset of those may be of interest to the case studies in TradeRES and those results are separately output into Excel files that are easy to access. Consequently, following data will be available in the Excel files for each scenario:

- consumption time series (separated when possible: inflexible demand, electrolysers, buildings, electric vehicles)
- price time series for electricity nodes (based on marginal values of the nodes)
- generation time series for all conversion units
- transfer time series for all transfer connections
- existing, invested and decommissioned capacities for different technologies in all nodes
- use of demand curtailment (to indicate issues in model runs)
- total generation by unit type
- total cost of the scenario

The results from the optimization models contain a lot more data than is considered above. It may also be that the data is required by the agent-based models in a specific format and the process is to be automated. In both cases, it is possible to build further processing chains using the exporter and importer tools in Spine Toolbox and, if need be, by using Spine database API Python functions to have programmatic access to the data. There are further instructions for using the exporters in the Spine Toolbox documentation: https://spine-toolbox.readthedocs.io/en/latest/data_import_export.html

6. References

Helistö, N., Kiviluoma, J., Ikäheimo, J., Rasku, T., Rinne, E., O'Dwyer, C., Li, R., & Flynn, D. (2019). Backbone—An Adaptable Energy Systems Modelling Framework. Energies, 12(17), 3388. https://doi.org/10.3390/en12173388