Need for rapid results! Ideas for speedy optimisation?

Elif Turhan

18 May 2022

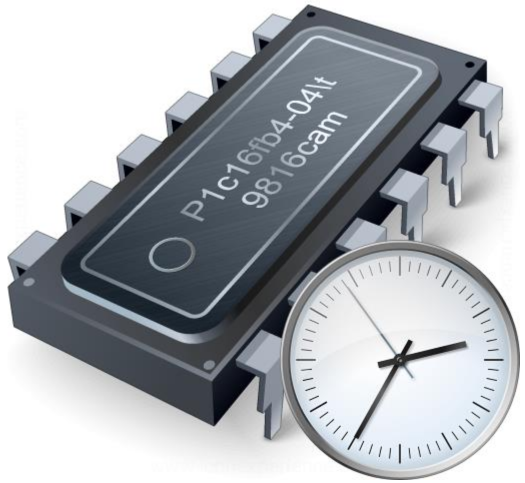
oemof user meeting

DLR Institute of Networked Energy Systems

DLR

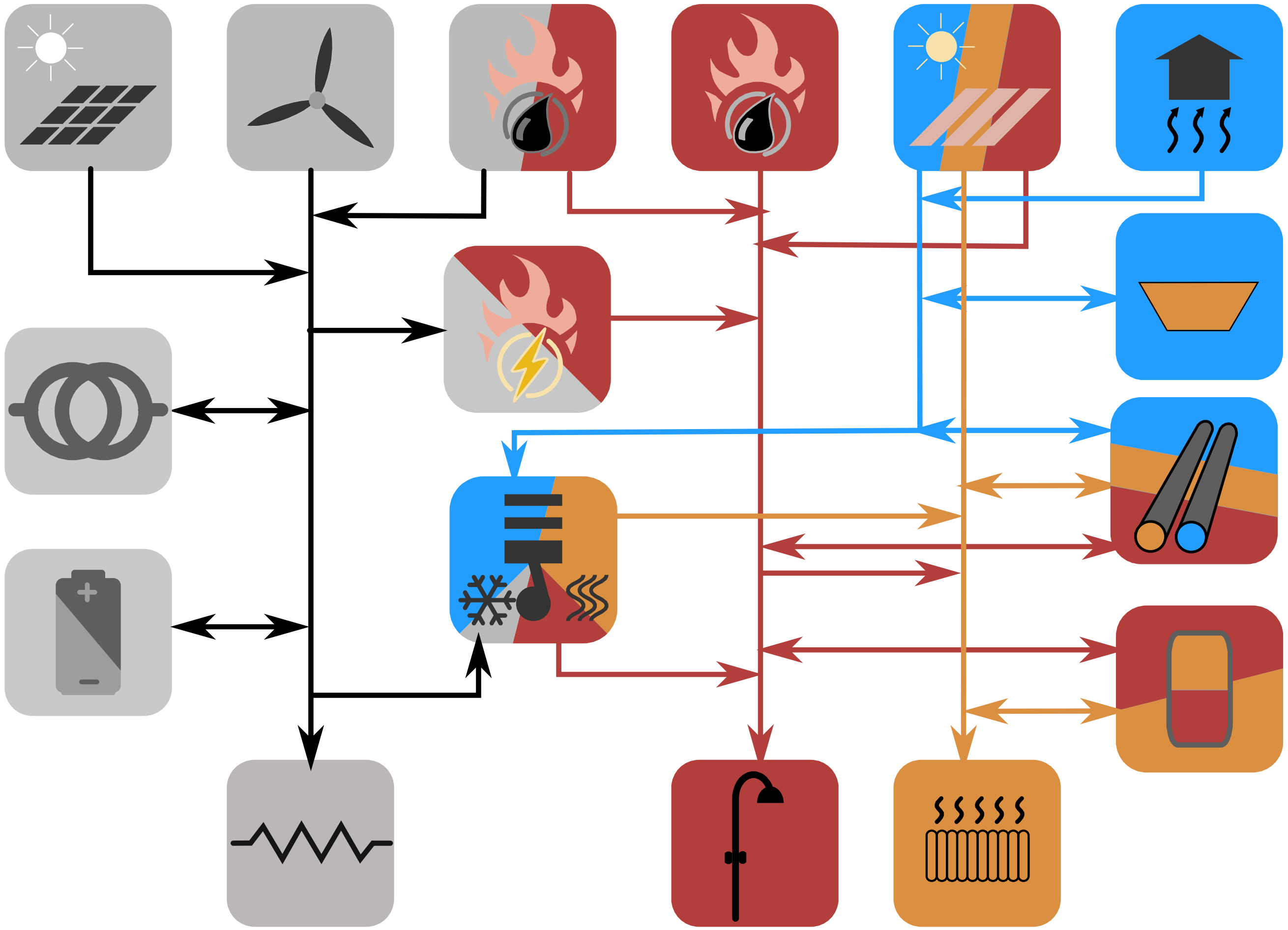
Motivation

- Complexity of energy system optimisation models and/or analysis
  - memory demand computing time
- Computations could take up to weeks
- Common and transferable methodologies reduce computational time, based on similar characteristics of energy system optimisations.

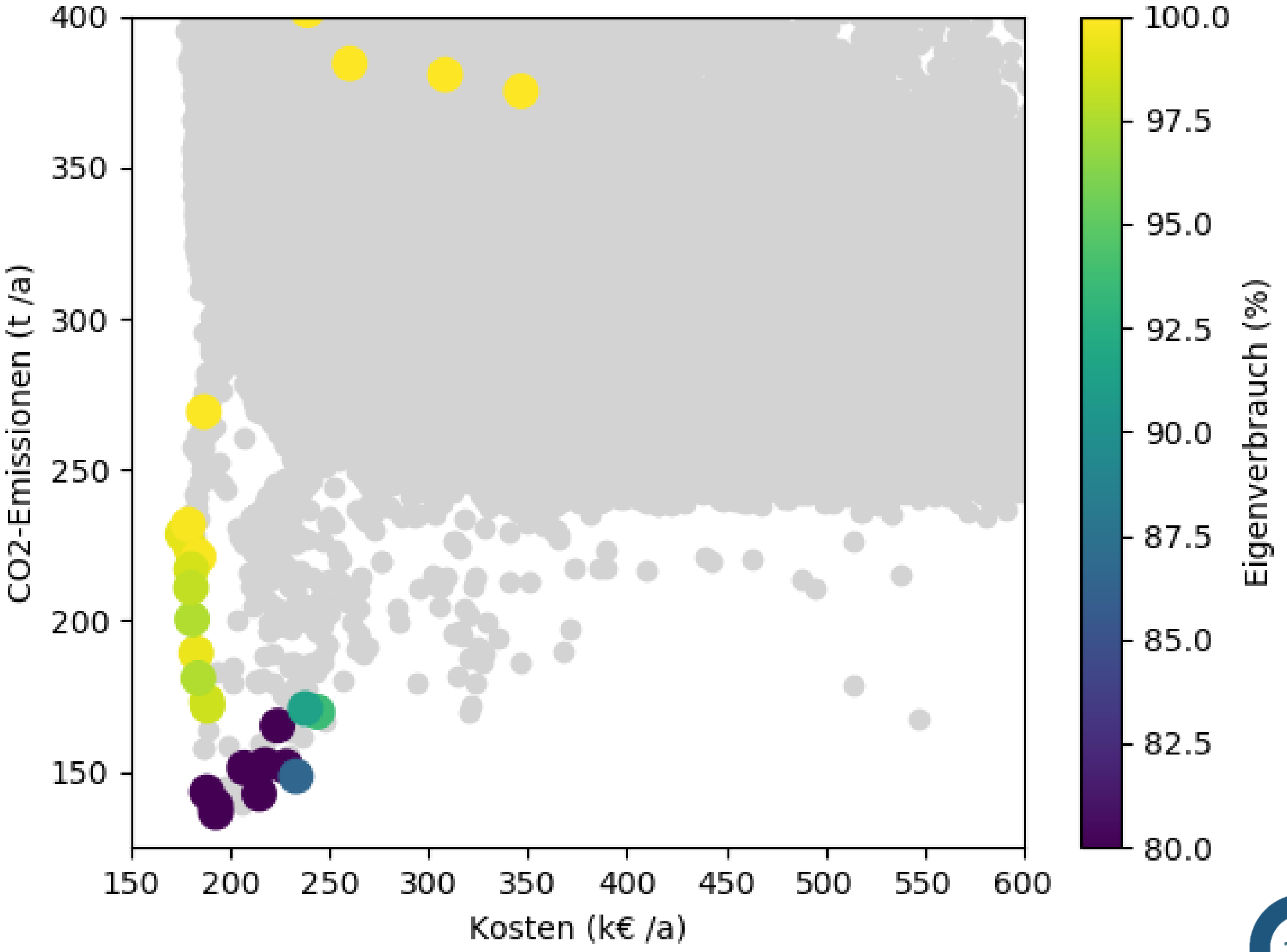


Model template for residential energy supply systems

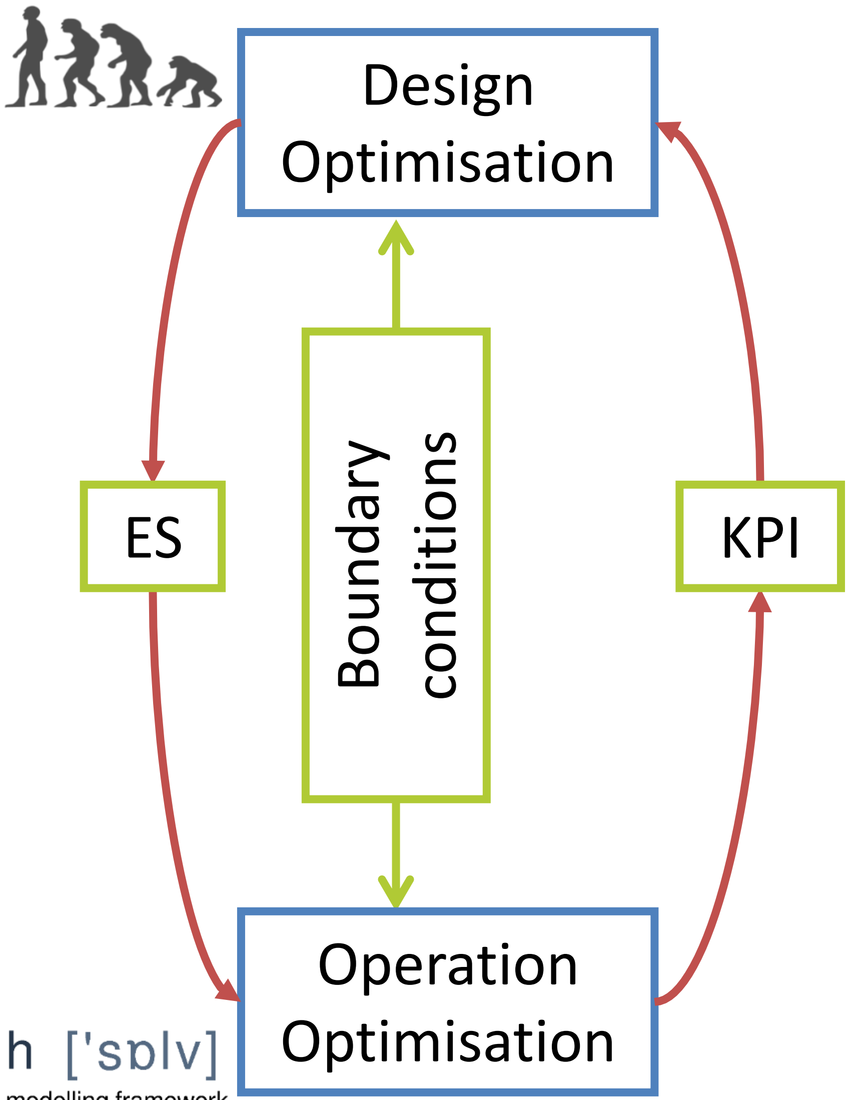
- Implemented using oemof.solph & oemof.thermal
- Fixed demands (electricity, heat, DHW)
- Network predefined, techs sized by parameters
- WIP: Connecting locations using electricity and heating grid(s)



Pareto-optimisation



PyGMO

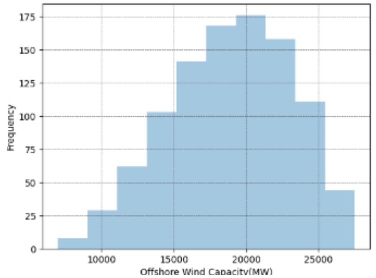


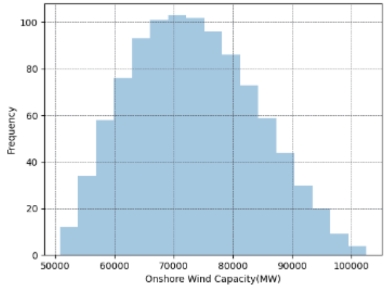
solph ['sɒlv]
open energy modelling framework

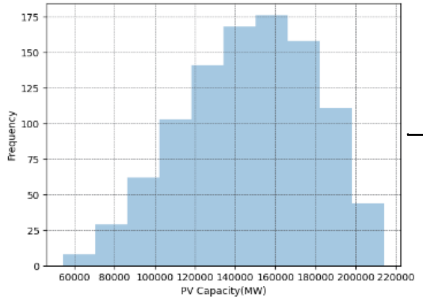
DLR

Use-case: Monte Carlo Simulation

2030 input pdf



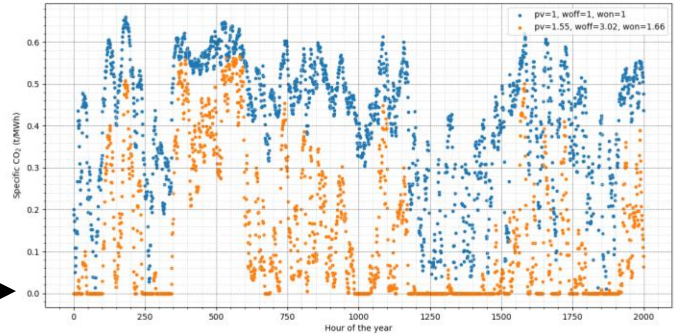


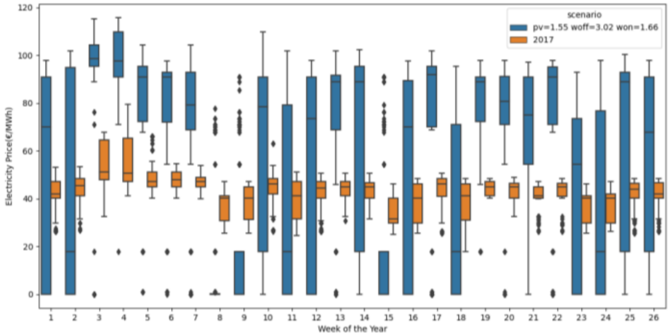


DEFLEX

National grid model

Intermediate Time Series

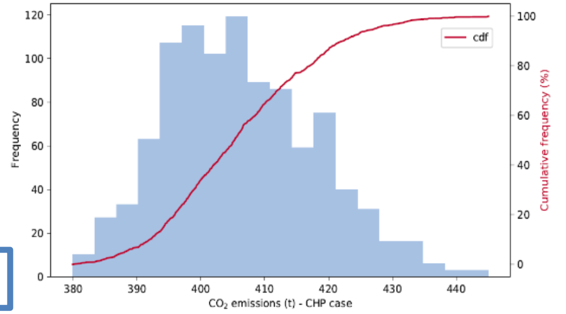


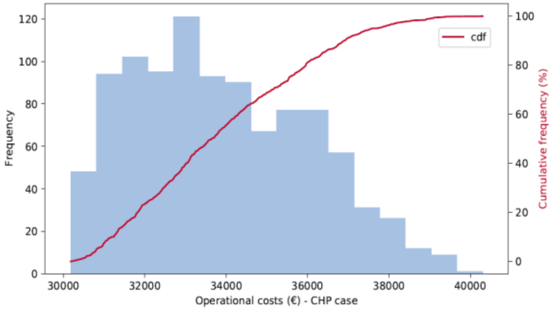


MTRESS

Residential supply model

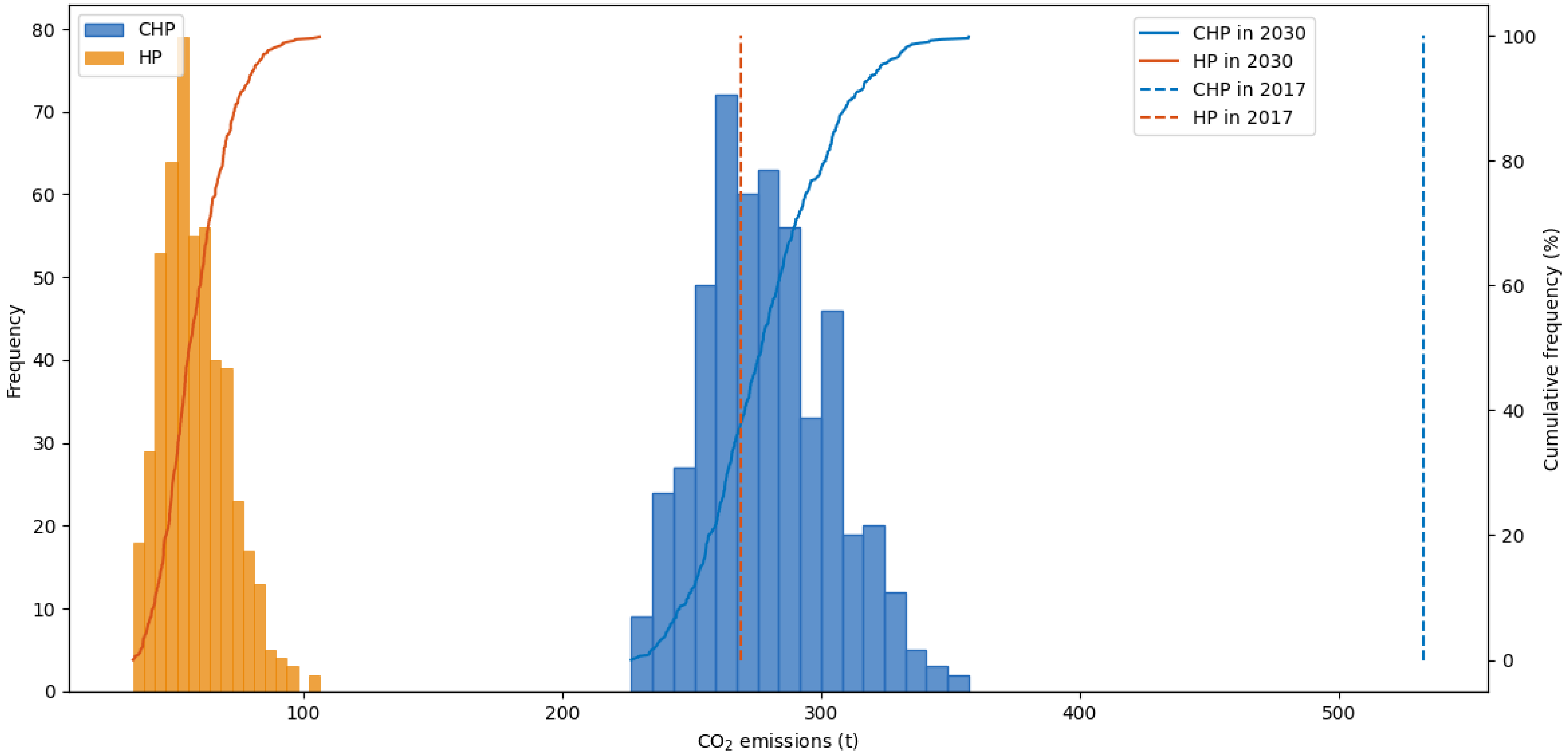
2030 performance pdf





DLR

Example Monte Carlo results



Overview of Different Methodologies

Methodology	Brief Description	Problems
Time Slicing	Focusing on a specific section of time	• Significant deviations of the results compared to the global optimum of full optimisation
Spatial Aggregation	Reducing spatial fidelity of the model	• Pre-calculation for majority of network equivalents of the model could be time costly, if this data is not available.
Temporal Aggregation	Down sampling of highly detailed data set	• Hard to capture the dynamic behaviour of the renewable energy power outputs • No clear best practice
Technological Aggregation	Reducing technological modelling accuracy	• No clear best practice
Rolling Horizon	Solving smaller individual time slices sequentially	• Hard to account for long term variables or constraints • May not meet the global optimum of the original problem
Temporal Zooming	Time slices optimised using info from coarser time scale optimisation.	• Requirement for an additional model run • May not be as fast as rolling horizon

Conclusions

- Need for quicker optimisation because of broader analysis or more complicated energy system optimisation
- Various options to reduce the computational time
- Each of them have their drawbacks

- Did you experience a need to speed up your optimisation?
- What difficulties did you experience and how did you overcome these?

Thank you for your attention!

PS: Please send presentations to patrik.schoenfeldt@dlr.de

Backup

Motivation

- Deregulation and growing decentralization lead to an increasing complexity of energy systems and this trend can be expected to continue.
- often investigated with the help of linear optimization models . increasing complexity of the system to be modelled results in energy system optimization models (ESOM) that reach their limits in terms of memory demand and reasonable computing time.
  - For example, very complex models on single servers could achieve or exceed computing times of weeks to a month with memory consumptions of 100 GB and more.
  - The mathematical optimization problems formulated in the models reached orders of magnitude that pushed the then most powerful solvers such as CPLEX, Gurobi, XPress, or SCIP to their limits.
- Since the used models show similarities in essential characteristics (e. g. with regard to fundamental equations or applied solver software packages), it can be assumed that effective speed-up strategies for energy system models are transferable.

Modeling based performance enhancement methods

- enhancement are related to the content and formulation of the model.

- Model reduction.
  - usually manipulate input data in a pre-processing step, instead of changing the way how an ESOM is solved
  - i) slicing and ii) aggregation

Model reduction-Slicing, Spatial,Temporal and Technological Aggregation

Slicing:

- focusing to a specific sub-problem by ignoring existing interdependencies, and therefore only a part of the input data that could be analyzed is used
- These sub-sets represent either critical situations, such as the peak load hour, or typical time periods are defined which are supposed to be characteristic for the entire set of operational time steps
- can lead to significant deviations of results compared to the global optimum of the full OP as they do not ensure that the relevant information within the available data is captured.
- if for the selection of specific slices a pre-analysis is conducted, we refer to this process as part of an aggregation as this approaches aim to take into account all input data

Model reduction-Slicing, Spatial,Temporal and Technological Aggregation

Spatial Aggregation:

- corresponding to the area of responsibility of system operators, methods for power networks were developed to study certain slices of the entire interconnected network

- In particular,
  they are usually characterized by a summation of demand and generation capacities, whereas intra-regional flows are neglected and regions are grouped based on administrative areas, such as market or country borders

Model reduction-Slicing, Spatial,Temporal and Technological Aggregation

- Temporal aggregation refers to representative time periods or the process of data down sampling derived from a highly resolved initial data set.
- time series based input data is coarsen to a lower temporal resolution (e.g. by averaging from 1-hourly to 6-hourly)
- down sampling typically affects demand profiles (e.g. electric or heat load) or the feed-in from vRES power
- Although the approach is an effective way to reduce computing times { (Stefan Pfenninger, 2017) for example shows a reduction of CPU time up to 80% (scenario 90% 2014){the method is rarely applied.
- ESOMs typically rely on their highest resolved data and often use hourly input
- More common is the combination of down sampling and the selection of representative time periods
- Moreover, also challenges exist to account for the chronological relationship between hours which in particular becomes important if time-linking constraints are incorporated in an ESOM.

- the selection of time-slices is either based on a clustering algorithm, such as k-means (Green, Staffell, & Vasilakos, 2014) or hierarchical clustering (Paul Nahmmacher, Eva Schmid, Lion Hirth,

Heuristics, nested approaches: Rolling horizon, temporal zooming

- methods that usually find near-optimal solutions
  rather than a theoretically guaranteed exact optimum.
- after the solution of the first reduced model is obtained, certain outputs are used as boundary conditions (e.g. in the form of additional constraints) for the following model(s) to be solved.

- Temporal zooming: First a large geographical coverage is considered in a coarse
  spatial resolution to study macroscopic interdependencies. In a second step, these interdependencies, such
- as transnational power flows, can be fixed in order to conduct a detailed analysis of the region of interest

- Comparing the different reduced models used in a nested approach, typically, a decrease of resolution on one
  scale is often accompanied by an increase on another.
  Therefore, first, a power plant portfolio is developed over
  the analyzed planning horizon using a simplified dispatch model and pre-defined time-slices to simulate the operation. In order to check whether the derived power plant portfolio performs well for a selected target year, UC constraints are added and capacities are fixed in the subsequent model run(s)

- A similar method call (Babrowski, Heffels,

Heuristics, nested approaches: Rolling horizon, temporal zooming

- split up the temporal scale (temporal slicing) into
  smaller intervals to obtain multiple reduced ESOMs to be solved sequentially
- Mostly in operational planning

- the total
  computing time for solving individual reduced problems stays below the one for obtaining a solution for the original problem
- a wide variety of speed-up achievements
  ranging from 15 up to 100 times.

- In the context of energy system
  analysis, this overlap is important to emulate the continuing global planning horizon. Especially the dispatch of seasonal storage units is strongly affected by this as, without any countermeasures, it is more costefficient to fully discharge the storage until the end of an operational period. Also time-linking variables
  and constraints, such as annual limits on emissions, can only barely be considered in this way since global information regarding the temporal scale can only be roughly estimated for each time window
- a trend to overestimate the total system costs (Marquant et al., 2015), the
  aggregation of interval solutions does not necessarily end up at the global optimum of the original problem.

Heuristics, nested approaches: Rolling horizon, temporal zooming

- rolling horizon approaches have one particular disadvantage. Since each partial solution is updated by a subsequent one, the reduced ESOM instances are sequentially coupled. This prevents parallel solving
- temporal zooming, overcomes this issue and allows for solutions closer to the exact optimum of the original problem.

- time-linking information is gathered from the solution of an additional ESOM instance which is reduced on the temporal scale. But, in contrast to the reduced ESOMs which consider specific intervals within the full operational horizon, the temporal resolution is down sampled. This in turn allows optimizing the dispatch of the original problem for the full planning period. Values of variables from this first model run can subsequently be used to tune the consideration of global time-linking variables and constraints within the intervals. Despite the need for an additional model run, total computing times for obtaining a final solution can be expected to be at least competitive compared to rolling horizon approaches. This is due to the fact that, on the one hand side, overlaps are not required and, on the other hand, the temporally sliced ESOMs can be solved in parallel.

- **Replace list with set to check whether an element is in a sequence**
- **Replace list comprehension with generator expressions**
- **Replace global variables with local variables**
- **Avoid function access**
- **Avoid class property access**
- Replace + with join() when concatenating strings
- Replace while with for
- Replace explicit for loop with implicit for loop
- **Reduce the calculation of inner for loop**
- **Use numba.jit**
  - https://towardsdatascience.com/10-techniques-to-speed-up-python-runtime-95e213e925dc
- To check if membership of a list, it’s generally faster to use the “in” keyword.
- You can load the modules only when you need them. This technique helps distribute the loading time for modules more evenly, which may reduce peaks of memory usage.
- **Exit early.:** You can test the input in a few ways before carrying out your actions. Another approach is to raise the exception early and to carry out the main action in the else part of the loop.
- a decorator function takes another function and extends its functionality. We denote these functions with the @ symbol. In the example above, I’ve used the decorator functools.lru_cache function provided by the functools module
- **Use keys for sorts. my_list.sort(key=operator.itemgetter(0))**
- **Don’t construct a set for a conditional.**