Enhancing Energy System Modelling with advanced mathematical decomposition techniques: feasibility of coupling SMS++ and PyPSA

Davide Fioriti, DESTEC, University of Pisa, Pisa, Italy, davide.fioriti@unipi.it

Alessandro Pampado, DESTEC, University of Pisa, Pisa, Italy

Claudio Scarpelli, DESTEC, University of Pisa, Pisa, Italy

Gianluca Pasini, DESTEC, University of Pisa, Pisa, Italy

Giovanni Lutzemberger, DESTEC, University of Pisa, Pisa, Italy

Stefano Barsali, DESTEC, University of Pisa, Pisa, Italy

Davide Poli, DESTEC, Università di Pisa, Pisa, Italy

Donato Meoli, Dipartimento di Informatica, University of Pisa, Pisa, Italy

Luca Mencarelli, Dipartimento di Informatica, University of Pisa, Pisa, Italy

Antonio Frangioni, Dipartimento di Informatica, University of Pisa, Pisa, Italy, antonio.frangioni@unipi.it

***Abstract*—Large-scale high-resolution energy system modelling is essential to support long-term decarbonization strategies. Open-source frameworks such as PyPSA-Eur and PyPSA-Earth enable high-resolution techno-economic analyses but face computational bottlenecks when applied to large-scale or stochastic systems. Advanced mathematical decomposition techniques can alleviate these challenges, yet they are rigid to specific problems and difficult to scale to different applications. SMS++ is a high-performance, C++-based modelling framework designed to support flexible, nested decomposition strategies through a structured block-based approach. However, its use has so far remained isolated from mainstream energy system modelling.**

This paper presents a novel methodology to couple energy modelling tools with mathematical decomposition frameworks, focusing on integrating PyPSA with SMS++, to facilitate the wide adoption of advanced decompositions into energy applications. A formal interface is developed by mapping PyPSA components to SMS++ blocks. A fully reproducible prototype workflow is then implemented and validated on six case studies, demonstrating numerical equivalence within tight tolerances. The successful integration confirms the feasibility of applying advanced decomposition techniques within a user-friendly modelling environment. This approach opens new pathways for scaling energy system models to sector-coupled and uncertainty-aware applications with enhanced computational efficiency.

***Index Terms*—PyPSA; SMS++; Mathematical Decomposition; Mixed-Integer Linear Programming (MILP)**

I. Introduction

A. Motivation

Decarbonizing large-scale energy systems require unprecedented investments that need to withstand long lifetime and be cost-effective to ensure affortable energy for populations to prosper. To meet this goal, policy makers require appropriate techno-economic tools to estimate the optimal planning of their future energy systems that also allow to effectively capture uncertainties to ensure resilience. However, modeling large-scale areas is notoriously computationally hard especially when considering uncertainties, which has led scholars to develop advanced mathematical decomposition techniques to break computational requirements down. Mathematical tools are generally highly tailored to their applications and complex to use, unlike some energy modeling tools such as PyPSA, which generally makes them hard to use. For these reasons, in this study we deemed useful to investigate the possible coupling of energy modelling tools and advanced mathematical frameworks that allow mathematical decomposition, with the promise of unlocking unprecedented computational efficiency for continental energy modelling.

B. Energy System Models

Energy system models are essential analytical tools used to simulate, analyze, and optimize the interactions among different components within energy systems, needed to guide policy making [1]. Typical energy system models range from detailed engineering tools to broader macroeconomic frameworks. Integrated assessment models (IAMs), such as REMIND [2], provide insights into the broader economic implications and interactions between energy, economy, and environment. However, to achieve higher representation, such as PyPSA [3], Calliope [4], TIMES [5] and OSeMOSYS [6], provide higher technical details suitable for techno-economic planning and dispatch analyses, supporting capacity expansion and detailed technical operation.

Various software tools and frameworks are available for executing energy system models. Some, including TIMES [5], generally focus on the overall energy mix. Accordingly, they generally have a lower spatial resolution, e.g. one or few buses per country, to focus on multi-year planning over long horizons. Other tools, such as PyPSA [3] or Calliope [4], support custom high spatial and temporal resolution, facilitating combined capacity expansion and dispatch analyses. The tools facilitate the creation of Linear or Mixed-Integer Linear Problems that can be solved by state-of-the-art solvers like Gurobi [7] or HiGHS [8].

Several continental models have been developed using high-resolution energy tools. For example, Euro-Calliope and PyPSA-Eur [9] supports the European energy system and PyPSA-Earth extends the scope to the globe [10], [11], among others [12]. Nevertheless, these models have large computational requirements that usually require approximations to make the numerical problem tractable.

C. Mathematical decomposition in energy

Energy system models are inherently complex due to several factors. Technological constraints such as unit commitment introduce non-linearities [13], while spatial resolution requires power flow constraints [14], and temporal granularity increases computational size, especially when uncertainty is included [15]. Solving such non-linear, large-scale problems often requires specialized solvers. When integer variables are involved, computational requirements can grow exponentially with model resolution.

To address this, researchers have developed mathematical decomposition methods to reformulate problems into equivalent, more tractable subproblems. Benders decomposition is widely used in two-stage stochastic optimisation [16], while Stochastic Dual Dynamic Programming is increasingly applied to multistage formulations [17]. Column and row generation techniques are also common [18]. Furthermore, recent dynamic programming-based approaches have enabled linearization of the unit commitment problem, without loss of generality [19].

Despite their effectiveness, these decomposition strategies are often tailored to specific problems and are not easily transferable. To address this gap, the Structured Modelling System (SMS++) [20] was developed as a general framework that supports advanced, nested decomposition techniques. While SMS++ offers robust performance thanks to its C++ architecture, its usability could be greatly enhanced through integration with widely used energy system modelling frameworks, enabling more accessible application of advanced decomposition methods via user-friendly interfaces.

D. Contributions and organization of the paper

This paper develops an approach for coupling energy models with the unconventional mathematical decomposition framework SMS++, to prove the possible simple application of decomposition techniques to arbitrary energy systems. In particular, beyond the coupling procedure, we also propose a prototype implementation, including a case study, to prove the feasibility of the approach.

The remainder of the paper is organized as follows. Section II and Section III provide a description of the framework PyPSA for Energy System Modelling and the tool SMS++ suitable for advanced decomposition. Then, Section IV describes the proposed methodology for executing the coupling of the two models. Section V details the case study adopted to validate the proposed procedure, whose results are discussed in Section VI. Finally, conclusions are drawn.

II. The PyPSA energy framework

A. Description

PyPSA (Python for Power System Analysis) is an open-source Python framework for simulating and optimizing energy systems. It supports both power flow and optimal capacity expansion studies, enabling a wide range of applications including dispatch optimization, network expansion, decarbonization strategies, and security-constrained analyses. Thanks to its modular structure, solver integration, and ease of customization, PyPSA has become a widely adopted tool in both academia and industry.

PyPSA models energy systems using abstract components, including: Buses (network nodes), Lines (network connections), Generators (e.g., wind, gas, nuclear), Storage Units and Stores (for hydro, battery and hydrogen storage), and Links (DC converters or sector-coupled units). These components offer flexibility for representing complex energy systems with varying levels of detail.

B. Mathematical model

1) Objective function: PyPSA transforms the description of the model into a Mixed-Integer Quadratic Problem whose objective function generally focuses on minimizing the Annuity Costs in (1), where $C^{CAP}_{n,a}$ and $C^{OP}_{n,a,t}$ denote the investment and operating costs of the asset a in node n and time step t, respectively. Investment costs are modelled in linear form as detailed in (2), where X_a denote the investment variable for component a and c^{cap}_a is the per-unit investment costs. Operating charges are mainly modelled with the quadratic expression shown in (3), where $p_{n,a,t}$ denotes the dispatch of the asset a subject to linear ($c^{op}_{n,a,t}$) and quadratic ($c^{op,q}_{n,a,t}$) operating charge, $c^{op,s}_a$ denotes the stand-by cost for every time-step the asset is dispatched using binary variable $u_{n,a,t}$. Shut-down and start-up charges are also supported by means of variables $w_{n,a,t}$ and $v_{n,a,t}$ with coefficients $c^{op,su/sd}_{n,a,t}$. For storages, additional cost terms proportional to the state of charge $e_{n,a,t}$ and the spillage of hydro units $p^{spill}_{n,a,t}$.

$$\min AC = \textstyle\sum_{n,a} \left(C^{CAP}_{n,a} + \sum_t w_t C^{OP}_{n,a,t}\right) \tag{1}$$

$$C^{CAP}_{n,a} = c^{cap}_{n,a} X_{n,a} \tag{2}$$

$$\begin{aligned} C^{OP}_{n,a,t} =& c^{op}_{n,a,t} p_{n,a,t} + c^{op,q}_{n,a,t} p^2_{n,a,t} + c^{op,s}_{n,a,t} u_{n,a,t} \\ &+ c^{op,su}_{n,a,t} v_{n,a,t} + c^{op,sd}_{n,a,t} w_{n,a,t} + c^{e}_{n,a,t} e_{n,a,t} \\ &+ c^{e}_{n,a,t} p^{spill}_{n,a,t} \end{aligned} \tag{3}$$

2) Constraints: PyPSA includes a comprehensive set of operational constraints for all system components. It supports bounds on both investment (4) and dispatch (5) variables, enforces ramp rate limits (6), and optionally allows for unit commitment modelling (7)–(11). When enabled, unit commitment includes binary on/off status variables, start-up (8) and shut-down (9) tracking, and minimum up/down time constraints (10)–(11).

$$\underline{X_{n,a}} \le X_{n,a} \le \overline{X_{n,a}} \tag{4}$$

$$\underline{p_{n,a,t}} X_{n,a} \le p_{n,a,t} \le \overline{p_{n,a,t}} X_{n,a} \tag{5}$$

$$-\underline{rd_{n,a,t}} \hat{X}_{n,a} \le p_{n,a,t} - p_{n,a,t-1} \le \overline{ru_{n,a,t}} \hat{X}_{n,a} \tag{6}$$

$$\underline{p_{n,a,t}} \hat{X}_{n,a} u_{n,a,t} \le p_{n,a,t} \le \overline{p_{n,a,t}} \hat{X}_{n,a} s_{n,a,t} \tag{7}$$

$$v_{n,a,t} \ge u_{n,a,t} - u_{n,a,t-1} \tag{8}$$

$$w_{n,a,t} \ge u_{n,a,t-1} - u_{n,a,t} \tag{9}$$

$$\textstyle\sum_{t'=t}^{t+T_u} \delta_{i,r,t'} \ge T_u \left(\delta_{i,r,t} - \delta_{i,r,t-1}\right) \tag{10}$$

$$\textstyle\sum_{t'=t}^{t+T_d} \left(1 - \delta_{i,r,t'}\right) \ge T_d \left(\delta_{i,r,t-1} - \delta_{i,r,t}\right) \tag{11}$$

Finally, the nodal energy balance is regulated in (12) that account for the dispatch variables for generators $p_{n,a,t}, a \in A^p$, storages ($p^{+/-}_{n,a,t}, a \in A^b$), and links and lines $p_{a,t}, a \in A^l$. The matrix $K_{n,a,t}$ is a three-dimensional quantity that describes whether the link or line l is connected to the node n. When $K_{n,a,t} = -1$, it means that the line or link a conventionally departs from the bus n and when $K_{n,a,t} = 1$ the line arrives at the node. Links support values different from -1 and 1 to support conversion efficiencies. Finally, the energy balance for storage is denoted in (13), accounting for self-discharging $\eta^{self}_{n,a,t}$, charging and discharging efficiency ($\eta_{i,s,+/-}$), energy inflow $p^{inflow}_{i,s,t}$ for hydro units and possible spillage $p^{spill}_{i,s,t}$.

$$\sum_{a\in A^p} p_{n,a,t} + \sum_{a\in A^b} \left[p^+_{n,a,t} - p^-_{n,a,t}\right] + \sum_{a\in A^l} K_{n,a,t} p_{a,t} = \sum_{a\in A^d} d_{n,a,t} \tag{12}$$

$$e_{n,a,t} = (1-\eta^{self}_{n,a,t}) e_{i,s,t-1} + w^S_t \Big(p^+_{i,s,t}\eta_{i,s,+} + p^-_{i,s,t}/\eta_{i,s,-} - p^{spill}_{i,s,t} + p^{inflow}_{i,s,t} \Big) \tag{13}$$

III. SMS++ TOOL FOR ADVANCED MATHEMATICAL DECOMPOSITION

A. Description

The Structured Modelling System (SMS++) [20] is an open-source, general-purpose C++ framework designed for formulating and solving complex optimization problems. Unlike most other existing mathematical optimization frameworks, SMS++ is focused on preserving the intrinsic mathematical structure of optimization problems. This enables a hierarchical application of advanced solutions, mainly decomposition techniques such as Lagrangian relaxation, Benders' decomposition, and Stochastic Dual Dynamic Programming.

B. Nested block structure

A typical SMS++ problem is described by a `Block` that can (recursively) contain multiple sub-`Block`s, each describing a part of the problem. Each `Block` is designed so that it can be either solved by a specialized `Solver` that exploits its specific structure to implement bespoke (and, hence, efficient) solution techniques, or by general-purpose solvers that use its "abstract representation" in terms of variables and (algebraic) constraints. In particular, SMS++ is interfaced with many current MILP solvers (all subclasses of the base `MILPSolver` object) such as HiGHS. Any number of `Solver`, be them specialised or general-purpose, can be attached to a `Block` to solve it, possibly in parallel exploiting specific (and unique) SMS++ features that allow safe concurrent access to the `Block` data structures.

In energy system applications, a common structure is the Unit Commitment [13] block (`UCBlock`), sketched in Fig. 1, that coordinates the operational scheduling of multiple generators, represented by the general class `UnitBlocks`. Specific generator types are then implemented as derived classes; among them, `ThermalUnitBlock` represents advanced unit commitment of thermal units, `HydroBlock` and `HydroSystemBlock` represent cascading hydro units and whole hydro systems with complex future-value-of-water functions, `BatteryUnitBlocks` represent batteries and `IntermittentUnitBlocks` model intermittent generations. This allows to develop specific solution approaches exploiting each unit's characteristics, such as the sophisticated Dynamic Programming procedure implemented in `DPThermalUnitSolver` [21] for `ThermalUnitBlock`. Besides generating units, another fundamental component of energy systems is the interconnection network(s) allowing energy exchanges. `UCBlock` caters for this by defining the general class `NetworkBlock`, which is then specialised according to the features of the energy model by means of the derived classes such as `DCNetworkBlock` (DC and HVDC electrical networks), `ACNetworkBlock` (AC electrical networks [14]), and `ECNetworkBlock` (energy community [18]).

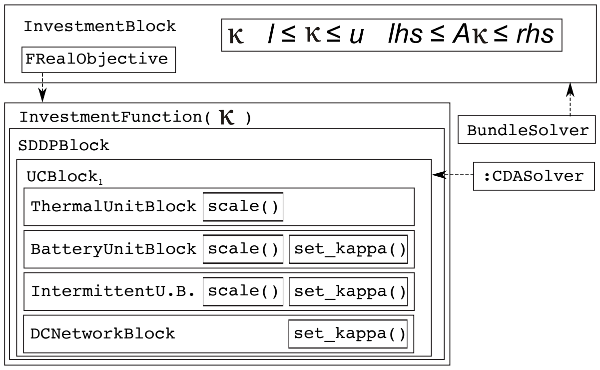

Fig. 1: A schematic of the energy system investment block (`InvestmentBlock`).

With `UCBlock` providing a general and flexible description of energy system operational problems, SMS++ allows to define general and powerful models for strategic problems—optimal energy system design and expansion—by means of `InvestmentBlock`. This exploits the nested structure of the capacity expansion problem by having a `UCBlock` as sub-`Block`, and defining a subset of the units and lines that can be designed, as depicted in Fig. 1.

The solution of the capacity expansion problem requires algorithms for non-smooth optimization [22], since the expansion value function is not everywhere differentiable, for which SMS++ provides the advanced `BundleSolver` implementation of bundle-type algorithms [23]. This requires the repeated solution of the inner `UCBlock`, that can be obtained either with general-purpose solvers like `HiGHSMILPSolver` and `GurobiMILPSolver`, or by approaches exploiting the block structure of the (inner) problem. Chiefly among this is `LagrangianDualSolver` that implements advanced Lagrangian decomposition (using again `BundleSolver` to drive the solution of the Lagrangian dual), allowing the (parallel) implementation of nested decomposition approaches. Although not extensively described in this report, the further module `SDDPBlock` can be "slotted in" between `InvestmentBlock` and `UCBlock` to represent the design of energy systems with significant stochastic components (such as year-long management of hydro reservoirs), exploiting the dedicated `SDDPSolver` implementation of the Stochastic Dual Dyanmic Programming approach to allow even more sophisticated multilevel decomposition schemes.

C. Mathematical representation

This section describes the representation of selected SMS++ `Blocks`, to facilitate the understanding of the methodology focused in the following paragraphs.

1) `ThermalUnitBlock`: In SMS++, this *Block* supports the most advanced mathematical representation of thermal units, including 7 different formulations of fuel-fired units with different number of variables and constraints and therefore various size-to-bound strength ratios (among which an "exact" DP-based one that does not need, in isolation, integrality constraints [19]). The nonlinear objective function is tightly described by dinamically-separated perspective cuts [24]. For the sake of clarity, we describe the closest formulation with respect to PyPSA, referred to as *3bin*.

The objective function, described in (14), is a quadratic function of the dispatch power p_t of the thermal unit, accounting for the stand-by cost $c_t^{op,s}$ and start-up charges $c_t^{op,su}$.

$$\min \sum_t c_t^{op,su} v_t + \sum_t (c_t^{op} p_t^2 + c_t^{op,q} p_t + c_t^{op,s} u_t) \quad (14)$$

This is similar to the model described in Section II, and similarities extend to the technical constraints, as shown in (15)–(21). Minimum up- and down-time requirements are accounted for in (15)–(17). Ramping limits are modeled using (18) for upward ramp and (19) for downard one. Minimum and maximum power limits are regulated by (20) and (21).

$$u_t - u_{t-1} = v_t - w_t \quad (15)$$
$$\sum_{s \in [t-T_u, t]} v_s \leq u_t \quad (16)$$
$$\sum_{s \in [t-T_d, t]} w_s \leq 1 - u_t \quad (17)$$
$$p_{t+1} - p_t \leq ru_t^s u_t + ru_t v_{t+1} \quad (18)$$
$$p_t - p_{t+1} \leq rd_t^s u_{t+1} + ru_t w_{t+1} \quad (19)$$
$$\underline{p}_t \leq p_t \quad (20)$$
$$p_t \leq \bar{p}_t u_t + (\bar{u}_t - \bar{p}_t) w_{t+1} + (\bar{l}_t - \bar{p}_t) v_t \quad (21)$$

The PyPSA model for generators and the *3bin* formulation of SMS++ are very close, which facilitates the tools coupling.

2) `DCNetworkBlock`: The energy balance in SMS++ is regulated by means of `NetworkBlocks` that support the modelling of (among others) DC and HVDC networks. The power flow of DC networks can be modelled in the linear formulation using equation (22), where $B_{(l,n)}$ is the Power Transfer Distribution Factor matrix, S_n is the generation at node n and D_n^{ac} is the demand. The modelling of HVDC lines is instead different as the power flow of each line can be controlled independently thanks to the converters located at their ends. Thus, the power flow of each line F_l can be controlled simply as in (23), and the energy balance can be formulated as (24).

$$P_l^{mn} \leq \sum_{n \in \mathcal{N}} B_{(l,n)} (S_n - D_n^{ac}) \leq P_l^{mx} \quad l \in L^{AC} \quad (22)$$
$$P_l^{mn} \leq F_l \leq P_l^{mx} \quad l \in L^{DC} \quad (23)$$
$$\sum_{l=(n,n')} F_l - \sum_{l=(n',n)} F_l = S_n - D_n \quad n \in \mathcal{N} \quad (24)$$

Finally, the objective function of the `DCNetworkBlock`, if any, is minimizing the costs incurred in the flow of energy across the grid. This can be obtained by introducing the auxiliary variable V_l that represents the absolute value of F_l, to appear in the objective function (25), and to be added to the constraints (26).

$$\min \sum_l nc_l V_l \quad (25)$$
$$-V_l \leq F_l \leq V_l \quad l \in L^{DC} \quad (26)$$

Note that there is a separate `NetworkBlock` for each of the time instant in which the time horizon is subdivided.

3) `InvestmentBlock`: This is the SMS++ component responsible to model capacity expansion. Its objective function, shown in (27), accounts for investment costs that are proportional to the size of each asset X_a and the optimal cost of the subproblem described by the inner `Block` for the given choice of the assets' dimensioning. Bounds on the capacity expansion, detailed in (28), are also given (and general linear constraints on the expansion variables are possible as well).

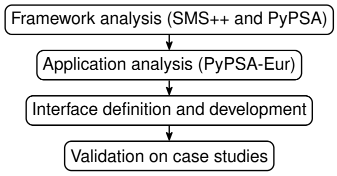

Fig. 2: Methodology.

$$\min \sum_a ic_a X_a + f(\text{Inner } \texttt{Block}(X)) \quad (27)$$
$$\underline{X_a} \leq X_a \leq \overline{X_a} \quad (28)$$

When the inner `Block` is a Unit Commitment (*UCBlock*), then the overall problem automatically inherits all objective functions of its `UnitBlocks` and `NetworkBlocks`, such as those shown in the previous subsections (14) and (25).

IV. METHODOLOGY

We propose the methodology described in Fig. 2 to successfully couple energy system models with tools for advanced mathematical decomposition, having in scope the application of the tool on PyPSA-Eur/Earth models and SMS++. The proposed approach is composed by a preliminary framework analysis focused on the selected tools (e.g. PyPSA and SMS++) and the target model (e.g. PyPSA-Eur/Earth), then the definition of the interface is executed and developed. Finally, the validation on selected numerical case studies is executed.

First, the framework (e.g. PyPSA) of the model (e.g. PyPSA-Eur) and the mathematical tool (SMS++) are analyzed with the goal of identifying (a) the abstract components being represented, (b) their mathematical formulation, (c) the input parameters and (d) the supported interfaces for input-output. This enables identifying the boundaries of action and the options suitable for coupling, namely which components of a framework are most similar to the other one, which formulations and approximations are best suited, what procedure is required to adapt the parameters, and finally how to operatively execute the coupling, e.g. which specific API calls to launch, if any.

Subsequently, the target final application is analyzed to identify which components to prioritize and their modelling data. Indeed, the complete coupling of all features is an extremely difficult task and focus must be guaranteed on the specific requirements. Accordingly, we propose to investigate the reference model to identify (a) how many sectors are represented, (b) what components and options are most frequently used and (c) the mesh-ness of the network, with special focus on the sector-coupled components that allow the combined coupling of multiple sectors. After this stage, it is possible to investigate what options to focus on and to develop specific interfaces.

The last two steps consist in the definition of the actual interface between the two tools and in its validation on selected case studies. These latter should consider the components of step 2 and the objective function of the two tools must be within a given optimization tolerance.

V. CASE STUDY

Given its wide adaptability, policy relevance and project focus, in this paper we investigate the feasibility of coupling the

Case	Type	Nodes [#]	PyPSA components [#]					
			Gen.	Stor. Unit	Store	Line	Link	Load
DT1	Dispatch	1	1					1
DT2	Dispatch	2	1			1		2
DS2	Dispatch	3	1		1	1	1	2
DALL5	Dispatch	5	3*	2	1	3	1	4
DNoHS1	Dispatch	1	3*	1				1
INoHS1	Investment	1	3*	1				1

*: 2 renewable units

TABLE I: Test cases of the methodology.

PyPSA-Eur model with the tool SMS++ for advanced mathematical decomposition. Accordingly, we apply the methodology discussed in Section IV to the two frameworks under consideration, namely PyPSA and SMS++, on the PyPSA-Eur model.

For the sake of brevity, we refer to sections II and III to describe the results of the investigation aimed to focus the mathematical framework, input parameters, and input/output interfaces. This analysis has been extended to all major SMS++ components, namely *BatteryUnitBlock*, *HydroUnitBlock*, *IntermittentUnitBlock*, *ThermalUnitBlock*, *SlackUnitBlock*, *DCNetworkBlock*, *UCBlock* and *InvestmentBlock*.

Subsequently, we analyzed the structure of the PyPSA-Eur model aiming to capture major components to develop the interface. While the analysis covered the whole sector-coupled model, for the sake of simplicity in this study we focused on the power-only version.

The preliminary analysis of the PyPSA-Eur model identified over 40 different networks and more than 150 different component types connecting them, sketched in the PyPSA-Eur-draw-io repository for simplicity [25]. It was found that the most recurrent PyPSA components are *Generators*, *Stores*, *StorageUnits*, *Links*, *Buses* and *Lines*, which are in focus of this study. In particular, in reference to Section II, the most important techno-economic factors being considered are the capital costs and marginal costs. For conversion technologies, efficiencies are also relevant, alongside water inflows for hydro units.

Given the PyPSA-Eur components, we defined the selected case studies in Table I to successfully perform the validation, as defined in the proposed methodology (Fig. 2). The case studies involve the optimization of microgrids of up to 5 nodes, including a diesel generator, renewable sources (wind and solar), battery storages, and hydro units with numerical parameters based on [26]. The proposed set aims to capture the main objects and validate that the procedure successfully converts PyPSA models into SMS++ to allow advanced mathematical decomposition. The case studies are available open-source in [27].

The results of the analysis and the prototype implementation are the focus of the following section.

VI. Results

A. Interface definition

According to the methodology outlined in Fig. 2, we performed a detailed comparison of the PyPSA and SMS++ frameworks, as introduced in Section II and Section III, respectively. The primary outcome of this comparison is the coupling matrix presented in Table II, which maps PyPSA components to their most appropriate counterparts in SMS++. This mapping provides the foundation for aligning the mathematical structures of the two frameworks and enables the definition of model equivalence through the consistent coupling of coefficients.

TABLE II: Coupling between SMS++ and PyPSA objects

Physical component	PyPSA	SMS++
Fuel-fired unit	Generator	ThermalUnitBlock
Renewable unit	Generator	IntermittentUnitBlock
Batteries	Stores	BatteryUnitBlock
Hydro units	StorageUnit	HydroUnitBlock
Converters	Link	DCNetworkBlock
Line	Line	DCNetworkBlock
Bus	Bus	DCNetworkBlock

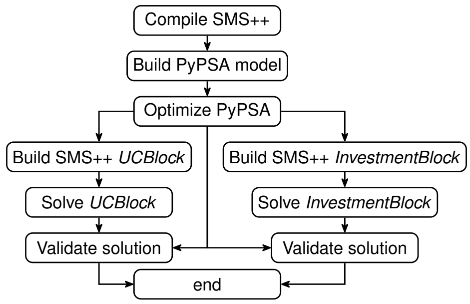

Fig. 3: Prototype flowchart implemented in [27]

In particular, parameters in the PyPSA objective and operational constraints ((1)–(3)) are put in correspondence with their best match in SMS++ equations ((14), (26), and (28)). Notably, for specific blocks such as *HydroUnitBlock* and *IntermittentUnitBlock*, marginal cost support was originally absent in SMS++ and has been added to ensure compatibility with the PyPSA formulation. It is also worth noting that converter assets in SMS++ can currently be represented using HVDC lines, although efficiency losses are not yet supported in this representation. For modelling capacity expansion problems, the use of *InvestmentBlocks*, as illustrated in Fig. 1, is appropriate. Conversely, for dispatch-only analyses, *UCBlocks* alone are sufficient to represent the problem structure.

B. Prototype implementation

Accordingly, we developed the workflow depicted in Fig. 3, which is available in the open-source GitHub repository *SMSpp_builder* [27] and implemented using GitHub Actions to ensure full open reproducibility. The procedure begins by compiling the necessary SMS++ tools, followed by generating the corresponding PyPSA models for each test case listed in Table I. Subsequently, the models are translated into SMS++ representations: *UCBlock* for dispatch analyses and *InvestmentBlock* for capacity expansion.

The entire workflow is orchestrated using the Snakemake tool, which facilitates the decomposition of the process into modular subtasks, each corresponding to a block in Fig. 3. The key conversion scripts from PyPSA to SMS++ are written in Python, contributing to the accessibility and transparency of the interface. This prototype demonstrates the successful integration of advanced mathematical decomposition techniques from SMS++ into the PyPSA modelling framework. The validation procedure compares the objective function values computed by SMS++ with those obtained from PyPSA, confirming the correctness of the conversion process.

C. Validation

Finally, the implemented procedure was tested on the selected case studies, with results summarized in Table III.

TABLE III: Validation results

Case	PyPSA [€]	SMS++ [€]	Difference [€]	Difference [%]
DT1	40669.01	40669.01	$2 \cdot 10^{-4}$	$< 10^{-4}$
DT2	82717.38	82717.38	$4 \cdot 10^{-4}$	$< 10^{-4}$
DS2	82717.38	82717.38	$4 \cdot 10^{-4}$	$< 10^{-4}$
DALL5	0.0	0.0	$< 10^{-5}$	-1.21
DNoHS1	0.0	0.0	-	0.0
INoHS1	27808.70	27808.80	0.099	$3.6 \cdot 10^{-4}$

All test cases were successfully validated, exhibiting absolute and relative differences well within the prescribed tolerances of €0.01 and 10^{-4}%, respectively. These outcomes confirm both the validity of the proposed coupling methodology and the practical feasibility of interfacing PyPSA and SMS++. This integration enables the application of advanced decomposition techniques within the user-friendly PyPSA framework.

VII. Conclusions

This paper presents a novel methodology for coupling energy modelling frameworks, specifically integrating the advanced mathematical decomposition techniques provided by SMS++ into the well-accessible PyPSA energy modelling tool. In addition to detailing the theoretical framework, a prototype implementation has been developed and validated in six case studies, covering both dispatch optimization and capacity investment analyses. The successful validation confirms the robustness of the proposed coupling approach and demonstrates the practical feasibility of the developed interface. Moreover, an in-depth theoretical comparison of the detailed mathematical structures of both tools was crucial in determining an effective strategy for their integration.

While the current study establishes the foundational feasibility of coupling SMS++ with PyPSA, future research should perform comprehensive analyses involving large-scale, sector-coupled energy systems. Further exploration of SMS++'s advanced capabilities, particularly in stochastic optimization contexts, would also significantly enrich the coupling methodology. Such advancements promise to significantly enhance the application of sophisticated decomposition methods, thereby improving the accuracy and reliability of future energy system analyses.

VIII. Acknowledgments

University of Pisa was supported by Italian Ministry of University and Research (MUR) under the RESILIENT project - CETPartnership – Joint Call 2022 TRI1 - Call module 1.1: PowerPlanningTools CUP I53C23002650007 co-funded by the European Union under grant agreement no. 101069750. We also thank colleagues of the Power System group at University of Pisa for the exchange and contributions on the paper.

References

[1] M. Brinkerink, B. Zakeri, D. Huppmann, J. Glynn, B. Ó Gallachóir, and P. Deane, "Assessing global climate change mitigation scenarios from a power system perspective using a novel multi-model framework," *Environmental Modelling & Software*, vol. 150, p. 105336, Apr. 2022.

[2] G. Luderer, N. Bauer, L. Baumstark, C. Bertram, and et al., "REMIND - regional model of investments and development," May 2025. [Online]. Available: https://www.pik-potsdam.de/research/transformation-pathways/models/remind

[3] T. Brown, J. Hörsch, and D. Schlachtberger, "PyPSA: Python for power system analysis," *Journal of Open Research Software*, vol. 6, no. 1, 2018, number: 1.

[4] S. Pfenninger and B. Pickering, "Calliope: a multi-scale energy systems modelling framework," *Journal of Open Source Software*, vol. 3, no. 29, p. 825, Sep. 2018.

[5] R. Loulou, G. Goldstein, A. Kanudia, A. Lettila, and U. Remme, "Documentation for the TIMES model - PART I 1–78," ETSAP, Tech. Rep., 2021. [Online]. Available: https://iea-etsap.org/index.php/documentation

[6] T. Barnes, A. Shivakumar, M. Brinkerink, and T. Niet, "OSeMOSYS Global, an open-source, open data global electricity system model generator," *Scientific Data*, vol. 9, no. 1, p. 623, Oct. 2022.

[7] Gurobi, "Gurobi Optimizer," 2020. [Online]. Available: https://www.gurobi.com/products/gurobi-optimizer/

[8] Q. Huangfu and J. A. J. Hall, "Parallelizing the dual revised simplex method," *Mathematical Programming Computation*, vol. 10, no. 1, pp. 119–142, Mar. 2018.

[9] J. Hörsch, F. Hofmann, D. Schlachtberger, and T. Brown, "PyPSA-Eur: An open optimisation model of the European transmission system," *Energy Strategy Reviews*, vol. 22, pp. 207–215, Nov. 2018.

[10] M. Parzen, H. Abdel-Khalek, E. Fedotova, M. Mahmood, M. M. Frysztacki, J. Hampp, L. Franken, L. Schumm, F. Neumann, D. Poli, A. Kiprakis, and D. Fioriti, "PyPSA-Earth. A new global open energy system optimization model demonstrated in Africa," *Applied Energy*, vol. 341, p. 121096, Jul. 2023.

[11] D. Fioriti, E. Fedotova, M. Parzen, D. Giubilato, M. Frysztacki, H. Abdel-Khalek, L. Schumm, S. James, and D. Poli, "Country-Wise Open Energy Planning in High-Resolution with PyPSA-Earth," 2023.

[12] D. Bogdanov, M. Ram, A. Aghahosseini, A. Gulagi, A. S. Oyewo, M. Child, U. Caldera, K. Sadovskaia, J. Farfan, L. De Souza Noel Simas Barbosa, M. Fasihi, S. Khalili, T. Traber, and C. Breyer, "Low-cost renewable electricity as the key driver of the global energy transition towards sustainability," *Energy*, vol. 227, p. 120467, Jul. 2021.

[13] W. van Ackooij, I. Danti Lopez, A. Frangioni, F. Lacalandra, and M. Tahanan, "Large-scale Unit Commitment under uncertainty: an updated literature survey," *Annals of Operations Research*, vol. 271, no. 1, pp. 11–85, 2018.

[14] B. D, M. Escobar, C. Gentile, and L. Liberti, "Mathematical programming formulations for the alternating current optimal power flow problem," *4OR*, vol. 18, no. 3, pp. 249–292, 2020.

[15] E. Vera, C. Canizares, M. Pirnia, T. Guedes, and J. Trujillo, "Two-Stage Stochastic Optimization Model for Multi-Microgrid Planning," *IEEE Transactions on Smart Grid*, vol. 14, no. 3, pp. 1723–1735, 2023.

[16] R. Zhao, Z. Hou, J. Lu, G. Lin, J. Yan, J. Chen, K. Pan, and F. Wang, "Distributed coordinated service restoration strategy for transmission and distribution networks based on distribution network partition," *Electric Power Systems Research*, vol. 246, 2025.

[17] J. Hole, A. Philpott, and O. Dowson, "Capacity planning of renewable energy systems using stochastic dual dynamic programming," *European Journal of Operational Research*, vol. 322, no. 2, pp. 573–588, 2025.

[18] D. Fioriti, A. Frangioni, and D. Poli, "Optimal sizing of energy communities with fair revenue sharing and exit clauses: Value, role and business model of aggregators and users," *Applied Energy*, vol. 299, p. 117328, 2021.

[19] T. Bacci, A. Frangioni, C. Gentile, and K. Tavlaridis-Gyparakis, "New minlp formulations for the unit commitment problem with ramping constraints," *Operations Research*, vol. 73, no. 5, pp. 2153—-2167, 2024.

[20] A. F. et.al., "SMS++, a Structured Modeling System for mathematical models." [Online]. Available: https://gitlab.com/smspp/smspp-project

[21] A. Frangioni and C. Gentile, "Solving nonlinear single-Unit Commitment problems with ramping constraints," *Operations Research*, vol. 54, no. 4, pp. 767–775, 2006.

[22] W. van Ackooij and N. Oudjane, "On supply and network investment in power systems," *4OR*, vol. 22, no. 4, pp. 465–481, 2024.

[23] A. Frangioni, *Standard Bundle Methods: Untrusted Models and Duality*. Springer, 2020, ch. 2, pp. 61–116.

[24] A. Frangioni and C. Gentile, "Perspective cuts for a class of convex 0–1 mixed integer programs," *Mathematical Programming*, vol. 106, no. 2, pp. 225–236, 2006.

[25] F. Davide, "PyPSA-Eur-drawio," 2025. [Online]. Available: https://github.com/SPSUnipi/PyPSA-Eur-drawio

[26] D. Fioriti, G. Lutzemberger, D. Poli, P. Duenas-Martinez, and A. Micangeli, "Coupling economic multi-objective optimization and multiple design options: A business-oriented approach to size an off-grid hybrid microgrid," *Int. J. Electr. Power Energy Syst.*, vol. 127, p. 106686, 2021.

[27] F. Davide, "SMSpp_builder," 2025. [Online]. Available: https://github.com/SPSUnipi/SMSpp_builder