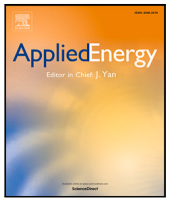
Applied Energy 373 (2024) 123848

Contents lists available at ScienceDirect

Applied Energy

journal homepage: www.elsevier.com/locate/apenergy



OPLEM: Open Platform for Local Energy Markets

Chaimaa Essayeh [a,c,*], Thomas Morstyn [a,b]

[a] School of Engineering, University of Edinburgh, UK
[b] Department of Engineering Science, University of Oxford, UK
[c] Department of Engineering, Nottingham Trent University, UK

ARTICLE INFO

Keywords:
Local energy market
Distribution networks
Distributed energy resources
Flexibility
Demand response

ABSTRACT

Local Energy Markets (LEMs) have been proposed as an effective solution to coordinate distributed energy resources within emerging smart local energy systems. However, LEMs are still at an early stage of development and are not widely implemented yet. Extensible open-source software tools are needed to simulate different market designs and evaluate their feasibility and effectiveness. This paper presents OPLEM, an open-source Python package for modelling and testing LEM designs. It offers a modular and flexible framework to create and test market designs adapted for distribution networks. OPLEM integrates two distinct models—one for designing and simulating local energy markets, and another for network operation. Unlike existing tools, OPLEM platform offers a comprehensive solution that bridges the gap between market design and network operation assessment, enabling stakeholders to gain deeper insights into the interactions between market dynamics and grid performance. Additionally, the inclusion of participant and asset models further enhances the tool's versatility, allowing for a holistic analysis. Thanks to its modular structure, OPLEM allows the development of customised LEM designs and initially incorporates four popular LEM designs: Time-of-Use pricing market, centralised dispatch with distribution locational marginal pricing, bilateral peer-to-peer market and auction market. By combining market design and network modelling functionalities, the tool empowers users to explore various scenarios and optimise market clearings to enhance both economic efficiency and grid reliability.

1. Introduction

The race towards net-zero raises two major technical challenges: how to ensure supply reliability while relying on clean intermittent generation [1] and how to make the grid ready to support electrification of heating and transportation, which involves the massive integration of energy-intensive appliances such as heat pumps (HPs) and electric vehicles (EVs) [2]. Local Energy Markets (LEMs) [3] have been proposed as an effective solution to coordinate distributed energy resources (DERs) within emerging smart local energy systems (SLES). LEMs can be broadly defined as markets that enable localised trading of electricity among prosumers (consumers who proactively manage their electricity demand/generation/storage) and other participants within a specific geographic area or community. These markets facilitate peer-to-peer energy transactions, allowing participants to buy, sell, or exchange electricity based on their needs, preferences, and available resources.

Several overarching classifications of LEM designs are delineated in the existing literature [4–6] with the most common being dynamic pricing, centralised economic dispatch (CED) and peer-to-peer (P2P) trading. Dynamic pricing involves adjusting the price of electricity based on various factors such as demand, supply, and market conditions. It implicitly encourages customers to shift their consumption from grid-stress periods to less-stressed periods of the day. In this context, energy management systems (EMS) [7] can be used to schedule the customer's assets in response to dynamic pricing. An extensive body of literature covers dynamic pricing business model and puts forth various methods, including linear programming [8], game theory [9] and reinforcement learning [4] to design dynamic pricing and EMS strategies.

Centralised economic dispatch is widely used for resources allocation at the transmission level [10], and can be extended to the distribution level [11]. It involves optimising the generation of power from various sources to meet the electricity demand at the lowest possible cost and considers factors such as fuel prices and operating constraints. In contrast to the transmission level, centralised economic dispatch in distribution networks encounters a significant computational burden due to the breakdown of the linear DC approximation, leading to a non-linear problem. Various solutions have been suggested

* Corresponding author at: Department of Engineering, Nottingham Trent University, UK.
E-mail address: chaimaa.essayeh@ntu.ac.uk (C. Essayeh).

https://doi.org/10.1016/j.apenergy.2024.123848
Received 17 May 2024; Received in revised form 19 June 2024; Accepted 2 July 2024
Available online 11 July 2024

Nomenclature

Indices and sets

$t, \mathcal{T}$	Index and Set for time steps, $t \in \mathcal{T}$.
$m, \mathcal{M}$	Index and Set for flexible assets, $m \in \mathcal{M}$.
$n, \mathcal{N}$	Index and Set for network nodes, $n \in \mathcal{N}$.
u	Index of user u.
$m_n, \mathcal{M}_n$	Index and Set for flexible assets at node n, $m_n \in \mathcal{M}_n$.
$m_u, \mathcal{M}_u$	Index and Set for flexible assets of user u, $m_u \in \mathcal{M}_u$.

Parameters

$\lambda^t_{imp}, \lambda^t_{exp}$	Import and export prices at the slack bus level at time step t.
$\lambda^t_{ToU}, \lambda^t_{FiT}$	Import and export prices at the customer side at time step t.
C^t_m	Operational cost of flexible asset m at time step t.
A_m, b_m	Matrix and vector of polytope representation of flexible asset m.
A_{agg}, b_{agg}	Matrix and vector of polytope representation of an aggregation of flexible assets.
$\mathbf{G}^t$, $g^{t,0}$, $\mathbf{K}^t$, $k^{t,0}$, $\mathbf{J}^t$ and $j^{t,0}$	Coefficients of the approximate linear model at time step t for multiphase distribution networks.
$v_{min}, v_{max}, c_{max}$	Minimum and maximum voltage limits, and thermal limit.

Variables

p^t_m	power output of flexible asset m at time step t.
p^t_n	power output at node n at time step t.
p^t_{imp}, p^t_{exp}	import and export power with the grid at time step t.
$\lambda^t_{s,b}, fee^t_{s,b}$	Agreed price of energy exchange between buyer b and seller s in P2P market, fee that buyer b and seller s should pay to network operator for trading energy at time step t.
$\lambda^t_{p,n}, \underline{\lambda}^t_{v,n}, \overline{\lambda}^t_{v,n}, \underline{\lambda}^t_{c,n}, \overline{\lambda}^t_{c,n}$	Dual variables associated to equality and inequality constraints for network operation.

by researchers to address this computational challenge. Ref. [5] for example, presented two formulations of economic dispatch in a distribution network, a non-linear exact formulation and a second-order cone formulation based on convex relaxation. For both formulations, the study analysed the interpretability of distribution locational marginal prices (DLMPs) by decomposing them into explainable terms.

P2P trading [12–14] is another concept suggested for LEMs that allows customers to trade and share energy with each other directly, often facilitated by blockchain [15] or other decentralised technologies. This approach allows for decentralised negotiation and prosumer autonomy.

It is worth noting that some papers combine two or more concepts to cope with the challenges of the emerging energy systems. In [16] for instance, a fusion of two designs, namely centralised economic dispatch and peer-to-peer (P2P) mechanisms, was undertaken. The study extended the concept of economic dispatch allocating grid costs to the P2P market participants exogenously in order to account for common infrastructure and services costs. The paper [17] merged two different pricing schemes: DLMPs and a fixed tariff to ensure social fairness of price allocation between customers who opt for a dynamic pricing scheme and those who remain using a fixed tariff.

Despite the abundance of literature in this field, researchers employ varied settings (e.g., network topology, number of assets and their technical parameters) and the source code of the designs are generally unavailable. This hinders the fair comparison between the various proposed designs and makes benchmarking a difficult process. To allow replication, refinement and comparison between existing LEM designs, extensible open-source software tools will be essential. The tools must enable the simulation of different market designs and evaluation of their feasibility and effectiveness. The existing tools consider the markets at transmission level only and provide solutions for spot markets, reserve markets and balancing markets [18].

This paper extends the existing Open Platform for Energy Networks (OPEN [19]) with advanced features tailored for LEMs. OPEN was initially designed to integrate modelling, control and simulation under the same platform, but lacked an LEM component, which is a core concept in future local energy systems. The platform has undergone a significant transformation and has been meticulously restructured and enriched with new features specifically designed to model intricate interactions with energy markets.

OPLEM represents a significant advancement in the field by seamlessly integrating two distinct models—one for designing and simulating local energy markets, and another for assessing their impact on network operation. Unlike existing tools, OPLEM platform bridges the gap between market design and network operation assessment, enabling stakeholders to gain deeper insights into the interactions between market dynamics and grid performance. By combining market design and network modelling functionalities, the platform empowers users to explore various scenarios and market mechanisms to enhance both economic efficiency and grid reliability. Additionally, the inclusion of participant and asset classes further enhances the tool's versatility, allowing users to define and model customised LEM designs. This novel approach not only facilitates more informed decision-making in LEM design but also supports the transition towards more sustainable and resilient energy systems. To demonstrate its capability, OPLEM initially incorporates four popular LEM designs with the initial release, (1) ToU market (2) CED market, (3) P2P market and (4) auction market. Additionally, it integrates EV and HP models, representing two of the primary flexible devices found at the distribution level.

The paper starts by reviewing existing market modelling and simulation tools for energy systems in Section 2 to identify the gaps that OPLEM is addressing. The platform structure is introduced in Section 3, and descriptions are given for the key components of the proposed software. Section 4 is dedicated to showcase three case studies of different LEM designs built using OPLEM. Section 5 concludes the paper and presents future directions for extension.

2. Review of existing tools

Simulation tools for energy systems can be classified into two broad categories: tools for network modelling and tools for market modelling. Table 1 gives a summary of the existing tools. The first category covers software tools used for modelling and simulating electrical transmission and distribution networks. Tools in this category include pandapower [20], MATPOWER [21], PowerFactory [22] and GridLab-D [23]. Each of these tools provides rich libraries to model, design and simulate network infrastructure and consider the physical constraints of the system with limited to no focus on the operational constraints of the resources connected to the network. These tools, while crucial for understanding the physical aspects of energy systems,

do not incorporate market dynamics, which are essential components for comprehensive system analysis and planning.

Market modelling tools have focused on CED at the transmission level considering network constraints using DC approximation to fulfil the supply–demand match. The DC approximation simplifies power flow equations by neglecting voltage magnitude fluctuations and focusing solely on phase angle differences between buses. This yields linearised power flow equations, rendering the optimisation problem convex. This approach offers computational simplicity in optimal power flow analysis, however, it assumes steady-state conditions with minor voltage angle deviations and reactive lines, rendering it more suitable for high-voltage transmission networks but less accurate for distribution networks. PowerGama [24], PyPSA [25] and PLEXOS [26] employ linear programming models to optimise generator dispatch and nodal prices, while integrating network constraints via a DC power flow model. Other tools such as ELMOD [27] and PRIMES [28] simulate energy consumer and producer responses to various factors or objectives such as economic development, carbon emission taxes and social fairness to achieve equilibrium in supply–demand matching. While these tools excel in optimising resource allocation and considering network limitations, it is important to note that their reliance on DC approximation renders them less suitable for addressing the complexities of dynamic market conditions within distribution networks.

Recently, with the assets integrated at the distribution level becoming more active, there has been a pressing need for the development of new tools tailored specifically for analysing local market dynamics at the distribution network level. OPLEM was created to address this gap. It offers integrated network modelling and market modelling within a single platform, with a focus on distribution networks and local markets. The key contributions of the tool are the following:

- A modular structure for market modelling: The market class simplifies the development and testing of new market mechanisms and enhances reusability between projects. It comes with four market example designs, namely the ToU market, the CED market, the bilateral P2P market and the auction market. It incorporates a receding horizon approach for market clearing, where decisions are made by considering both current information and future forecasts within a specified horizon. Additionally, it integrates a power flow simulation method to assess the network's response to the market clearing decisions, ensuring comprehensive analysis of the market's impact on the network infrastructure. Other market designs can be created with the help of the existing methods, and new functionalities can be integrated thanks to the modular structure.
- Aggregation: Collections of heterogeneous flexible devices can be represented using an equivalent virtual battery model. The method is useful when the number of flexible devices increases and incorporating their individual operational constraints becomes computationally prohibitive. The tool also comes with a method that recovers feasible disaggregated device-level setpoints from an aggregate solution.
- Modelling market participants with multiple resources: Individual market participants may have a portfolio of resources spread across multiple nodes of the network. The participant class within OPLEM can be used to model different types of participants present within LEMs.
- Modelling of assets at distribution level: OPLEM allows for the inclusion of diverse DERs. Notably, it provides ready-to-use models for EVs and HPs, which are pivotal in the ongoing efforts to decarbonise the transport and heating sectors.

3. Platform structure

The UML class diagram of the platform structure is shown in Fig. 1. The structure was designed to fit LEM features and characteristics.

Table 1
Software tools for energy markets. TN: Transmission Network, DN: Distribution Network, BL: Balanced-Linear, BN: Balanced-Nonlinear, ML: Multiphase-Linear, MN: Multiphase-Nonlinear, CED: Centralised Economic Dispatch, ToU: Time of Use, P2P: Peer-to-Peer.

Ref	Software	Level	Market	Network model
[20]	pandapower	TN, DN	–	BN
[21]	MATPOWER	TN, DN	–	BN
[22]	PowerFactory	TN, DN	–	MN, BN
[23]	GridLAB-D	TN, DN	–	MN
[29]	OpenDSS	TN, DN	–	MN
[24]	PowerGama	TN	CED	BL
[28]	PRIMES	TN	CED	BL
[27]	ELMOD	TN	CED	BL
[25]	PyPSA	TN	CED	BN, BL
[26]	PLEXOS	TN	CED	BL
[19]	OPEN	DN	–	ML, MN
[30]	OPLEM	DN	CED, ToU, Auction, P2P, Custom	MN, ML

The `Network` class encapsulates the physical infrastructure, the `Asset` class specifies DERs and `Participant` and `Market` classes are the two pivotal components for market modelling functionality. `Participant` class enables users to define the entities participating in the market while the `Market` class define markets and incorporates features for market clearing and energy exchange. The details of interactions between the different classes are demonstrated in Fig. 2. The procedure starts by creating a `network`, which defines the physical infrastructure of the power system to be modelled. We then connect `assets` to the network infrastructure. The `participants` are then declared, with each `participant` possessing a portfolio of the connected assets. The `market` is defined by its type, the list of its `participants` and other associated parameters. Depending on the selected market, a market clearing method runs and the result of the clearing is used to update the assets' schedules. After the profiles of the assets are updated, a network simulation runs to analyse the impact of the market clearing on the network. The full documentation of the tool methods and classes can be found in [31].

3.1. Network

The `Network` class models the electrical infrastructure of the system. A network is a set of nodes connected to each other through physical lines with specific characteristics. Different multi-phase low and medium voltage networks are included with OPLEM and ready-to-use, namely: IEEE13 bus feeder, IEEE123 bus feeder and European low voltage feeder (EULV). Balanced networks can be loaded using `pandapower` package. To analyse the power flow, two methods exist: a non-linear power flow method `zbus()` and a linear method `linear_pf()` that uses the approximate linear model for multiphase distribution networks from [32]. As many market clearing mechanisms use linear programming, a method `get_linear_parameters()` is included that retrieves the linear parameters of the approximate linear model.

3.2. Asset

Once the network is created, the user can connect assets to it. An `Asset` is defined by its unique identifier `id`, the bus-phases pair (β, ϕ) the asset is connected to. ϕ is a vector and its size determines if the asset is connected to a single phase or multiple phases of the bus β. Additionally, the asset is defined by the simulation time step duration `dt` and its horizon `T`, and the optimisation time step duration `dt_ems` and its horizon `T_ems`. The temporal arguments must remain consistent across all declared asset instances within the same system. Four subclasses inherit from the `Asset` class, `BuildingAsset` that

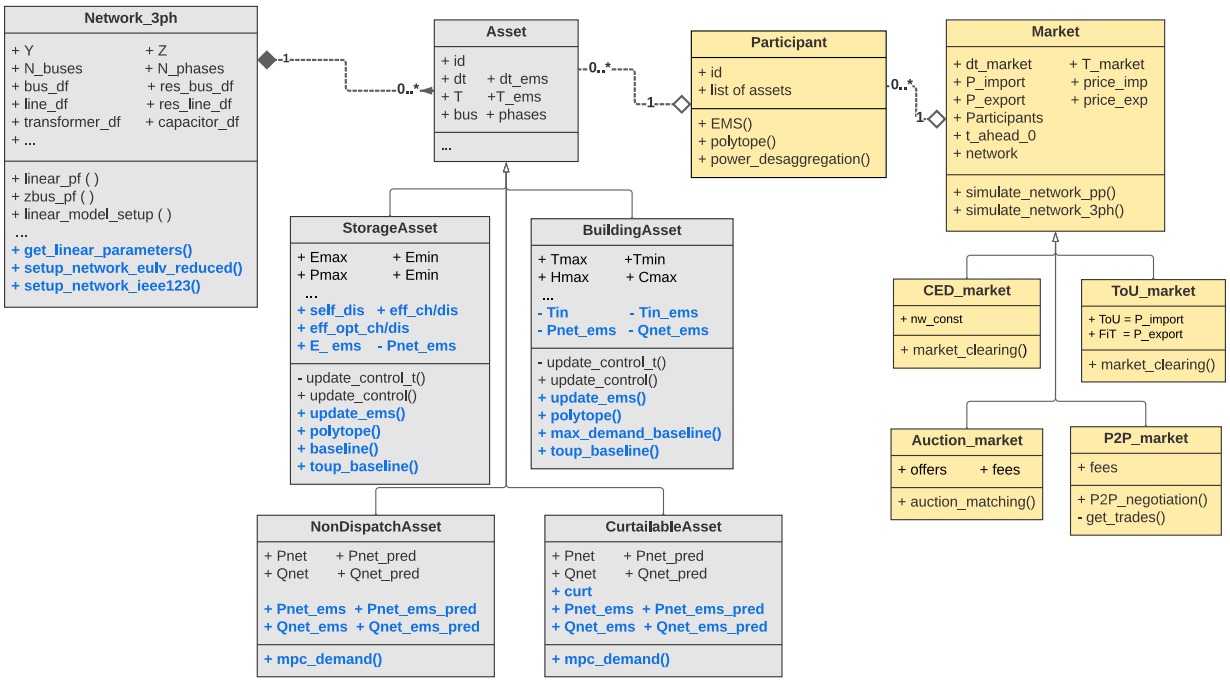


Fig. 1. UML class diagram of OPLEM. New market-oriented classes are yellow, and the attributes/methods from OPEN that were amended to accommodate the new market-based features are highlighted in blue.

models buildings with flexible heating, `StorageAsset` that models energy storage assets such as batteries, `CurtailableAsset` for loads and resources that can be curtailed such as PVs and wind turbines and `NonDispatchableAsset` for inflexible loads and non-dispatchable generation.

In contrast to OPEN, OPLEM incorporates additional methods to accommodate the new market features including `polytope()` method for flexible assets, baseline methods for the storage and the building assets, and `mpc_demand()` for non-dispatchable and curtailable assets.

- `update_ems()`: This method calls the `update_control()` method that takes control references as inputs, and updates an `Asset` object's output power profiles and state variables. The `update_ems()` takes the input references sent by the scheduling algorithms and converts them to the simulation time scale for the purpose of the control. The update methods have an optional argument `enforce_const` (set by default to `True`) that allows the user to decide whether the update method should enforce the asset operational constraints. This option when set to `False` will be useful for applications that model the operational constraints of the assets as soft constraints and allow the user to capture the deviations to evaluate them. A popular example of such applications is reinforcement learning (e.g., [33,34]). When applied on curtailable assets, the method takes the curtailment schedule as an input and updates the `Pnet_ems` accordingly. The method accepts an optional argument t_0 (set by default to 0) to define the starting time of the update. It is useful for applications that use a receding horizon strategy.
- `polytope()`: A polytope representation of asset operation constraints is a geometric way of describing the limits and conditions under which an asset can operate, using a multi-dimensional shape made up of linear inequalities. This method was defined for storage and the building assets. It follows the model presented in [35] and returns the matrix A_m and vector b_m of the polytope representation of the asset operational constraints: $\{p_m | A_m p_m \leq b_m\}$, with p_m is the power schedule of the asset m. This polytope representation includes power and energy (resp. indoor temperature) constraints for storage (resp. thermal) assets. Matrix A_m for instance include parameters such as charging/discharging efficiency and self-discharging rates for storage asset and building thermal resistance and coefficient of performance for building asset. While b_m integrates parameters such as maximum/minimum capacity, maximum/minimum charging rates and initial/final state of charge (SoC) for storage asset and minimum/maximum allowed indoor temperature and maximum power for building asset. It is useful in the sense that it facilitates the incorporation of the assets' operational constraints in optimisation strategies. The method accepts an optional argument t_0 (set by default to 0) to define the starting time step of the representation, which is useful if the method is called to generate the assets constraints for an optimisation working under a receding horizon.
- `mpc_demand()`: returns the up-to-date schedule of the non-dispatchable and curtailable assets composed of the observed value of the current time step t_0 and the predicted values for the rest of the horizon.
- `HP_maxdemand_baseline()`: runs a linear programming optimisation that returns the baseline schedule of a building asset, with the objective of minimising the peak demand under the operation limits of the asset.
- `toup_baseline()`: depending on the asset type, this method runs a linear programming optimisation that returns the optimal schedule of the flexible asset with the objective of minimising the costs in response to a time of use (ToU) tariff.
- `flexibility()`: computes the (upward or downward) flexibility that the flexible asset (building or storage asset) can provide for a flexibility window T_{flex}, depending on which type of assets it was called for.
- `EV_baseline()`: returns the charging schedule of the EV under the strategy of charging as soon as plugged in. The connection of the EV to the charger starts the charging process.

3.3. Participant

Participants can be declared following the initialisation of assets, with each participant owning a portfolio of assets. The participant is one of the core elements of the market concept. In future energy

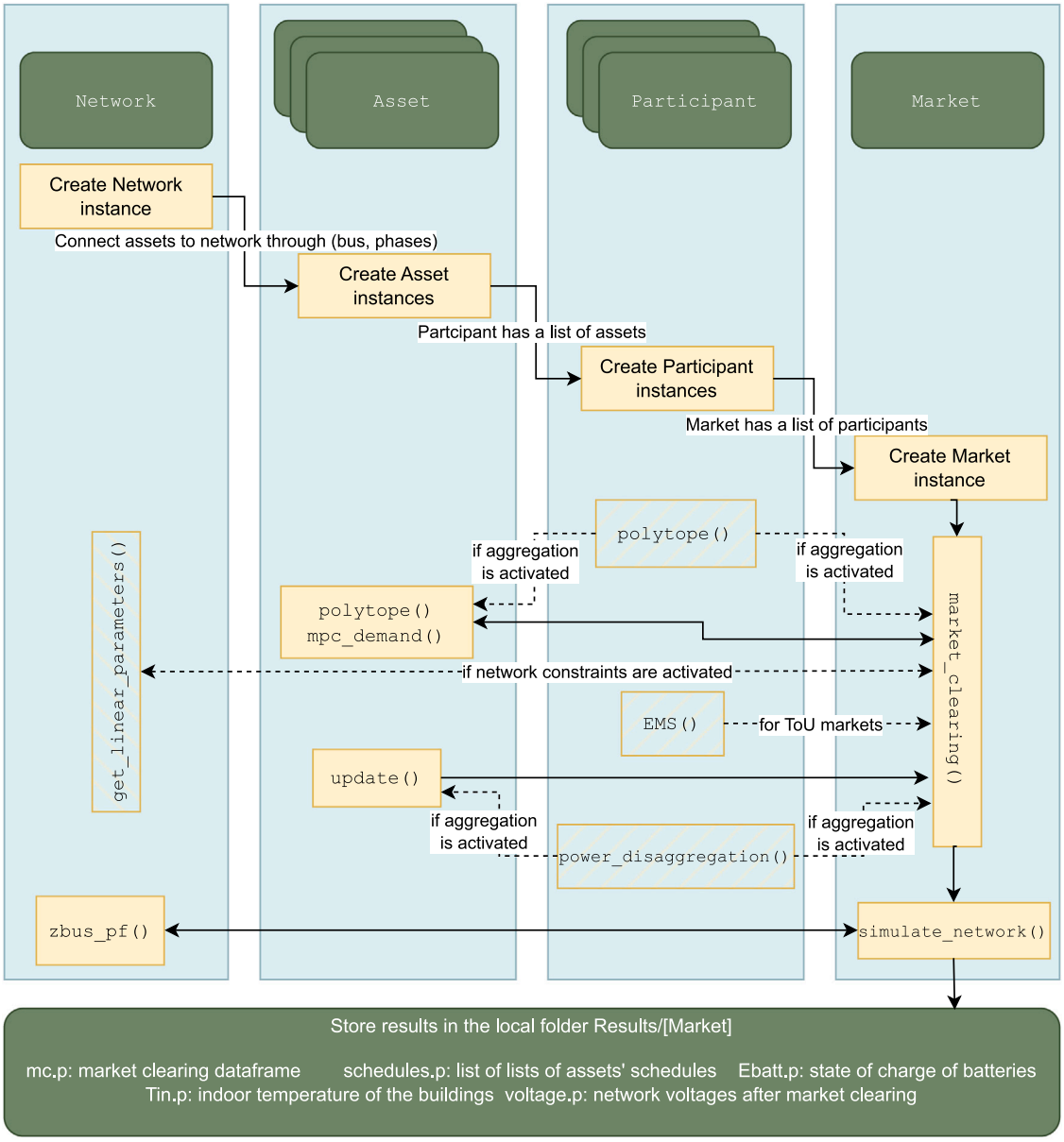


Fig. 2. Interaction diagram of OPLEM displaying the different interaction flows between the tool classes.

markets, different types of participants will be involved, including the active participation of the end consumers and the emergence of new commercial roles such as aggregators. The `Participant` class was conceived to be inclusive and able to model different roles.

3.3.1. Attributes

A participant in OPLEM is characterised by its identifier `id` and a list of its assets. Examples of different entities that can be modelled as participants using this class are:

- A homeowner: if all the assets in the list are connected to the same bus.
- An aggregator: if the assets in the list are connected to different buses.
- An energy retailer: if all the assets in the list are of type generation and/or storage.

This structure allows the co-existence of heterogeneous assets owned by entities with different interests. For example, it can model:

- A retailer with generation and centralised storage at bus $b_{id} = 0$, who has the objective of maximising profits, and
- Customers with different assets, including storage assets, at different buses $b_{id} \neq 0$, who have the objective of minimising costs or maximising autonomy.

Using the tool, we can also model the local energy community (LEC)[1] concept effectively. For instance, it is possible for a household to make their PV part of the community while excluding their EV. This can be achieved as the following:

- Asset creation: Assets such as EVs and PVs located at the household level are created with the b_{id} of the bus to which the house is connected.
- Household participant: A Participant instance is created for the household, the `assets` attribute for this instance includes the

[1] A local energy community (LEC) is a group of energy consumers, producers, and prosumers that operates under a shared framework and collectively optimises energy resources within a localised area.

EV and other assets that the house owner does not wish to make part of the community.
- Community participant: A participant instance will be created for the community, the `assets` attribute for this attribute includes the PV of the above household and assets contributed by various households. This community instance will have different assets located at different nodes of the network.

3.3.2. Energy management system

The `EMS()` method runs an optimisation programme that minimises the energy bill of the participant subject to the assets' operational constraints. A compact formulation is presented in system of Eqs. (4) in the next subsection. The programme returns the optimal dispatch of the participant's resources under a specific pricing scheme. Its arguments include the import/export prices and import/export power limits that can differ from one bus to another.

3.3.3. Aggregation

`Participant` class includes a method for asset aggregation. The feature will be useful to accelerate the computation in the case where multiple flexible resources are connected per bus. Two methods are added:

- `polytope()`: returns the outer approximation of the aggregated matrix A_{agg} and vector b_{agg} of the polytope representation for aggregation of flexible assets following the method presented in [35]:

  $$\{p_{agg} = [p^1_{agg}, \dots, p^t_{agg}, \dots, p^{T_{ems}}_{agg}] | A_{agg} p_{agg} \leq b_{agg}\},$$

  where $p^t_{agg} = \sum_{m \in \mathcal{M}} p^t_m$ is the aggregated power output of the set of assets at time step t, $\mathcal{M}$ set of aggregated assets and p^t_m is the power output of asset m at time t.
- `power_disaggregation()`: recovers a feasible disaggregated power schedule for each asset in the aggregation from the aggregated solution (Appendix).

3.4. Market

The market class was created and conceived to be general and adaptable to different types of markets. The three main attributes of the `Market` class are:

- `Participants`: Each market has a list of participants.
- `t_ahead_0`: This attribute allows for the option of running a receding time horizon simulation. If it is equal to 0, then the market will run for all the time steps of T_{market} horizon. Otherwise, the market clearing will run from the time step `t_ahead_0` to the end of the horizon.
- `network`: the network is an optional attribute to specify, and it is useful in particular cases, such as in a centralised economic dispatch (CED) market that accounts for network constraints, or to return the results of the power flow simulations after the market is cleared.

Other attributes include prices of imports and exports over the optimisation horizon and import/export power limits. The methods that are common for all the markets are the ones related to power flow simulations and receding horizon simulations:

- `simulate_network_pp()`: runs a power flow simulation for `pandapower` networks. The method returns a dictionary containing the voltage magnitude, voltage angle, real and reactive power injections at each bus and the real and reactive power injections at the slack bus. It can be called if a network instance was associated with the market and the network in question was created using Pandapower.

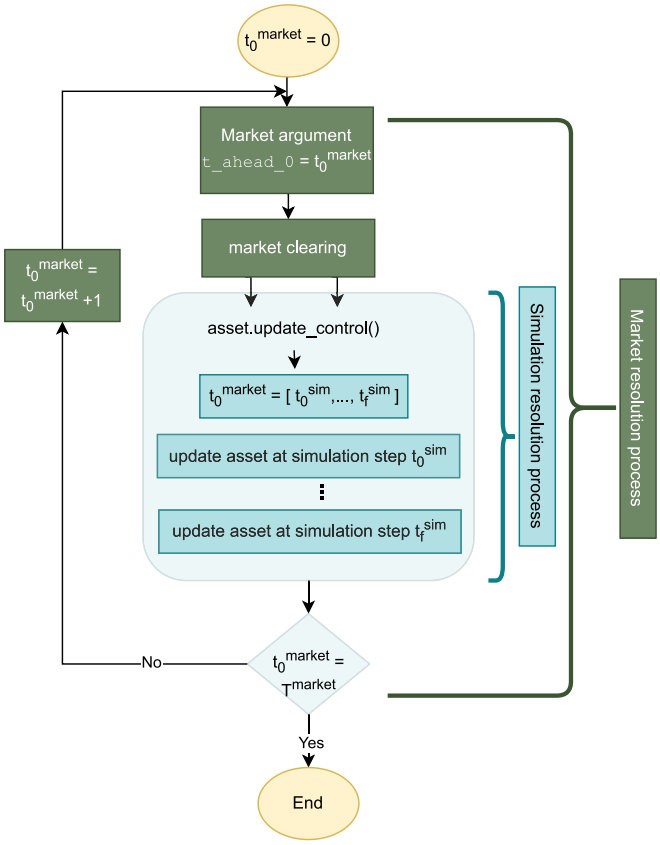


Fig. 3. Programme flow diagram for a receding horizon market application.

- `simulate_network_3ph()`: runs a power flow simulation for OPLEM three phase networks. The method returns a list of network instances (one for each time step). The network instances contain the results of the power flow in `res_bus_df`. This method can be called if a network was specified when initiating the market instance and if the specified network was created using the OPLEM `Network_3ph` class.
- `rh_market_clearing()`: runs a receding horizon version of the market clearing. Fig. 3 shows the programme flow diagram for a receding horizon market application.

The network simulation methods accept an optional argument `single_iter`, which if set to `True`, will run the power simulations for one optimisation time step, while it will run the simulations for the rest of the horizon if set to `False`. Setting the option to `False` is more suitable for day-ahead applications, as this will simulate the network flow for all the steps of the horizon. For receding horizon applications, information is updated every iteration, and thus the method can be called every iteration with the parameter set to `True` which will run the simulation for the current time step only with the most updated information.

Four market subclasses inherit from The `Market` class, they present the most popular energy markets found at the distribution level and they are detailed in the following subsections. Note that these markets were created for demonstration purposes and using the `Market` class, users can create other customisable markets and not be limited to the four markets available on the platform.

3.4.1. CED market

In the centralised economic dispatch market class, the market clearing optimises all the resources' schedules within the network to minimise the cost of energy. It returns the assets' optimal schedules and

the distributed locational marginal prices (DLMPs). DLMPs specify the marginal cost of supplying an additional unit of electricity at specific locations within a distribution system. They reflect supply–demand patterns and are influenced by the losses and physical constraints of the distribution network. This type of market accounts for network constraints and it requires complete knowledge of assets' information. The `market_clearing()` method runs the optimisation problem (1) and returns the assets' schedules and the market clearing table.

$$
\begin{aligned}
\underset{(p^t_{m_n},p^t_{imp},p^t_{exp})}{min} \quad & \sum_{t\in\mathcal{T}}\left(\lambda^t_{imp}p^t_{imp}-\lambda^t_{exp}p^t_{exp}+\sum_{n\in\mathcal{N}}\sum_{m\in\mathcal{M}_n}C^t_{m_n}(p^t_{m_n})\right)\\
\text{s.t.} \quad & p^t_n=\sum_{m_n\in\mathcal{M}_n}p^t_{m_n} \qquad (\lambda^t_{p,n})\\
& \left\{\begin{array}{ll} p^t_{imp}-p^t_{exp}=\mathbf{G}^t p^t+g^{t,0} & (\mu^t)\\ v_{min}\le \mathbf{K}^t p^t+k^{t,0}\le v_{max} & (\underline{\lambda}^t_{v,n},\overline{\lambda}^t_{v,n})\\ 0\le \mathbf{J}^t p^t+j^{t,0}\le c_{max} & (\underline{\lambda}^t_{c,n},\overline{\lambda}^t_{c,n})\end{array}\right\}\\
& A_{m_n}p_{m_n}-b_{m_n}\le 0, \forall m_n\in\mathcal{M}_n, n\in\mathcal{N}, t\in\mathcal{T}
\end{aligned}
\tag{1}
$$

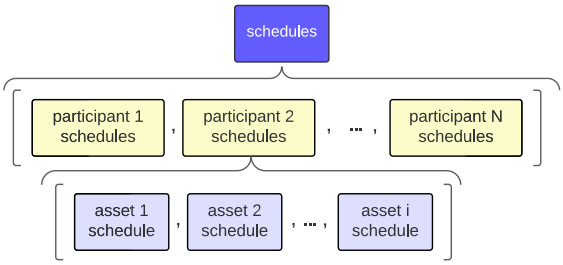
The objective of problem (1) is to minimise the cost of energy at the slack bus and the operational cost of assets while ensuring all the constraints (network constraints and assets' operational constraints) are validated. $\mathcal{N}$ is the set of connected (bus, phases) pairs in the distribution network, $\mathcal{M}_n$ is the set of flexible assets at (bus-phase) pair n, $\mathcal{T}=[t\_ahead\_0, T_{market}]$ is the clearing horizon of the modelled system, $(p^t_{imp}-p^t_{exp})$ is the net power seen at the slack bus which can be positive if there is a net import or negative if there is a net export, and $\lambda^t_{imp}/\lambda^t_{exp}$ the import/export slack bus energy price at time step t. The second term of the objective function refers to the operational costs of the assets that follow a linear pattern with $C^t_{m_n}$ is the cost function of operating asset m_n at time t. p^t_n is the power at node n at time step t and is equal to the sum of flexible assets power $p^t_{m_n}$ connected to the node n. $\mathbf{G}^t$, $g^{t,0}$, $\mathbf{K}^t$, $k^{t,0}$, $\mathbf{J}^t$ and $j^{t,0}$ are the coefficients of the approximate linear model at time step t for multiphase distribution networks [32] retrieved using the method `get_linear_parameters()`. Power balance, voltage and thermal limit constraints are presented in the respective equations inside the brackets. The power vector $p^t=[p^t_1,p^t_2,\dots,p^t_N]^T$ and the power at each node n depends on the asset connection:

$$
p^t_n=\begin{cases}[p^t_{(n,a)},p^t_{(n,b)},p^t_{(n,c)}] & \text{Y connection}\\ [p^t_{(n,ab)},p^t_{(n,ac)},p^t_{(n,bc)}] & \text{D connection}\\ [p^t_{(n,a)},0,0] & \text{Single a-connection}\end{cases}
\tag{2}
$$

The last constraint presents the set of operational constraints for the assets where the matrix and the vector A_{m_n} and b_{m_n} are obtained by calling the method `Asset.polytope()`. The DLMPs are represented by optimal dual variables of the problem $(\lambda^{t,*}_{p,n},\mu^*_t,\overline{\lambda}^{t,*}_{v,n},\underline{\lambda}^{t,*}_{v,n},\overline{\lambda}^{t,*}_{c,n},\underline{\lambda}^{t,*}_{c,n})$ using the following equation:

$$
\begin{aligned}
DLMP[t,n]=\frac{1}{\delta t}(&\lambda^{t,*}_{p,n}+\mu^{t,*}\mathbf{G}^t_n\\
&+(\overline{\lambda}^{t,*}_{v,n}-\underline{\lambda}^{t,*}_{v,n})K^t_n\\
&+(\overline{\lambda}^{n,*}_{c,t}-\underline{\lambda}^{t,*}_{c,n})J^t_n)
\end{aligned}
\tag{3}
$$

The assets' schedules are structured as a list of lists, with the first level of lists referring to the participants and the second level refers to the assets of the participant (See Fig. 4(a) for an illustration). The market clearing table is a `pandas dataframe` with six columns: index, time, seller, buyer, energy, and price (see Fig. 4(b)). Each row stores the trade (energy, price) that took place at time t between seller s_{id} and buyer b_{id}. The s_{id} and b_{id} correspond to the `id` of the participant instance. We note that for the case of the CED market, participants trade energy with the upstream market only, which means that either the seller or the buyer column will have the $id=0$ that corresponds to the upstream market. When the CED market accounts for network constraints, the prices in the market clearing table correspond to the distributed locational marginal prices (DLMPs) and are stored



(a) An illustrative example of the schedules output structure: after running a market clearing method, the assets' schedules will be stored as a list of lists, with the first level of lists referring to the participants and the second level referring to the assets of the participant. The asset schedule p_{m_n} is a numpy array storing the active power of the asset over the market horizon.

	time	seller	buyer	energy	price
24	12	0	2	2.4873	0.0958
153	12	0	8	2.9892	0.0958
452	12	0	21	0.4902	0.0958

(b) An example of a market clearing dataframe: each row stores the trade (energy, price) that took place at time t between seller s_{id} and buyer b_{id}. In this case (ToU market), participants 2,8 and 21 buy their energy from the upstream market at 12 pm at the set price of £ 0.0958/kWh.

Fig. 4. Outputs of the market clearing methods.

in the folder `Results/Central/`. The user can control whether the `market_clearing()` method accounts for the network constraints or not through the boolean argument `nw_const`. Moreover, the user can specify which lines and nodes are constrained through the `v_unconstrained_buses` and `i_unconstrained_buses` optional arguments.

3.4.2. Time of use market

The ToU market operates in contrast to the CED market. Every participant manages its resources in response to a ToU tariff with no knowledge of other participants' information and no consideration of the network constraints. The `ToU_market` calls for the `EMS()` method in the `Participant` class that runs the optimisation problem (4).

$$
\begin{aligned}
\underset{p^t_{m_u},p^t_{imp},p^t_{exp}}{min} \quad & \sum_{t\in\mathcal{T}}\left(\lambda^t_{ToU}p^t_{imp}-\lambda^t_{FiT}p^t_{exp}+\sum_{m\in\mathcal{M}_u}C^t_{m_u}(p^t_{m_u})\right)\\
\text{s.t.} \quad & A_{m_u}p_{m_u}-b_{m_u}\le 0, \quad \forall m_u\in\mathcal{M}_u\\
& p^t_{imp}-p^t_{exp}=\sum_{m_u\in\mathcal{M}_u}p^t_{m_u}+p^t_{u,inflex}, \forall t\in\mathcal{T}\\
& p^t_{imp},p^t_{exp}\ge 0, \forall t\in\mathcal{T}
\end{aligned}
\tag{4}
$$

The sub-subscript u refers to the participant and $\mathcal{M}_u$ the set of flexible loads participant u has. The output power of flexible asset m_u at time t is $p^t_{m_u}$ and $C^t_{m_u}$ is the cost function related to operating the asset, λ^t_{ToU} is the time-of-use price, λ^t_{FiT} the Feed-in-Tariff (FiT) price and $p^t_{imp}/\ p^t_{exp}$ are respectively import/export power at time step t. A_{m_u} and b_{m_u} refer to the polytope representation of the asset m_u. The aggregate inflexible load of the participant is noted $p^t_{u,inflex}$.

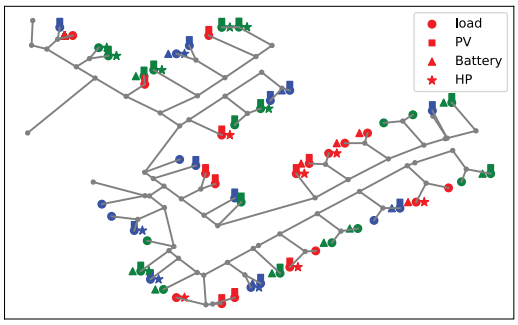


Fig. 5. EULV network is adopted for the case studies. Different icons refer to the different assets connected to a bus. Colours refer to connected phases, red for Phase A, green for Phase B and blue for Phase C.

The market clearing returns the assets' schedules and the market clearing table as was explained in Fig. 4. The price column shows the ToU value of the corresponding time step if the upstream market is the seller and the Feed Tariff (FiT) value if the upstream market is the buyer.

3.4.3. P2P market

This subclass runs a bilateral peer-to-peer energy trading as was proposed in [36]. This P2P decentralised strategy is an iterative price-adjusting mechanism that returns a stable set of bilateral contracts between peers. The process starts with each participant defining multiple potential trades with other participants; a buyer for instance will define several potential trades with several sellers, and each buyer/seller sets an initial price. The price of these trades is then adjusted by the price negotiation mechanism, based on buyers and sellers accepting or rejecting trades at their current prices at each iteration. At convergence, each contract specifies the price $\lambda^t_{s,b}$ agreed by both peers (seller and buyer) to exchange a discrete amount of energy. The final outcome accounts for the peers' preferences and maximises their individual utilities. Preferences include comfortable indoor temperatures and preferred EV charging schedules. An algorithm for the P2P negotiation mechanism can be found in ([37], Algorithm 1). The numpy matrix `fees` holds the fees that the participants should pay to trade with one another. They may vary across distinct trading contracts, and can therefore be used to discourage transactions anticipated to contribute to constraint violations. Each contract fee is shared by the participants and a seller will receive a payment of $(\lambda^t_{s,b} - fee^t_{s,b}/2)$ while the buyer will pay $(\lambda^t_{s,b} + fee^t_{s,b}/2)$. The fees are considered in the utility.

The `P2P_negotiation()` is the method that runs the iterative algorithm of P2P negotiation until convergence (i.e., reaching a stable outcome). It accepts different arguments:

- `trade_energy`: is the discrete amount of energy which is the same across all the trades.
- `price_inc`: the amount by which the prices are incremented in each iteration.
- `N_p2p_ahead_max`: the maximum number of P2P offers/bids that a peer can set for the time ahead.

The aforementioned arguments impact the results of the market clearing. A smaller `price_inc` and `trade_energy` will lead to more opportunities for P2P trading, but will take more time to converge, while a design with a large `price_inc` and/or `trade_energy` will converge faster but may miss opportunities for mutually beneficial P2P trades. Similarly to the other markets, the `P2P_negotiation()` returns the list of schedules and the market clearing table. In this case, participants can trade with each other with the price value listed in the price column, $\lambda^t_{s,b}$.

3.4.4. Auction market

The auction market is slightly different from the other markets. In the other markets, the offers are implicitly implemented within the design, while the auction market needs this information as an argument. The `offers` argument is a table with five columns: index, time, participant, energy and price. The sign of the energy determines if the participant is a buyer or a seller.

The `auction_matching()` method matches the buyers and sellers based on the list of offers. Two types of priorities are considered.

- price-based priority: the buyer with the highest bid price is matched to the seller with the lowest offer price.
- demand-based priority: the buyer with the highest bid demand is matched to the seller with the highest offer surplus.

Algorithm 1 summarises the main steps on the matching procedure in an auction market.

Algorithm 1 Auction Matching Algorithm (price-based priority)

Input: `offers`, list of buyers and sellers with their respective bids and asks.
Output: List of matched buyer-seller pairs and transaction prices.
Step 1: Sort Buyers and Sellers
Sort buyers in descending order of their bids' prices.
Sort sellers in ascending order of their asks' prices.
Step 2: Match Buyers and Sellers
for each buyer b in sorted buyers list **do**
  for each seller s in sorted sellers list **do**
    while bid of $b \geq$ ask of s **do**
      Match buyer b with seller s.
      Record the transaction price.
      if bid energy of $b \geq$ offer energy of s **then**
        Subtract the energy from total bid energy of b.
        Remove s from the list of sellers.
      else
        Subtract the energy from total offer energy of s.
        Remove b from the list of buyers.
        Break the inner loop.
      end if
    end while
  end for
end for

The method returns the market clearing table, and each row stores the trade (energy, price) between seller and buyer at time step t.

4. Case studies

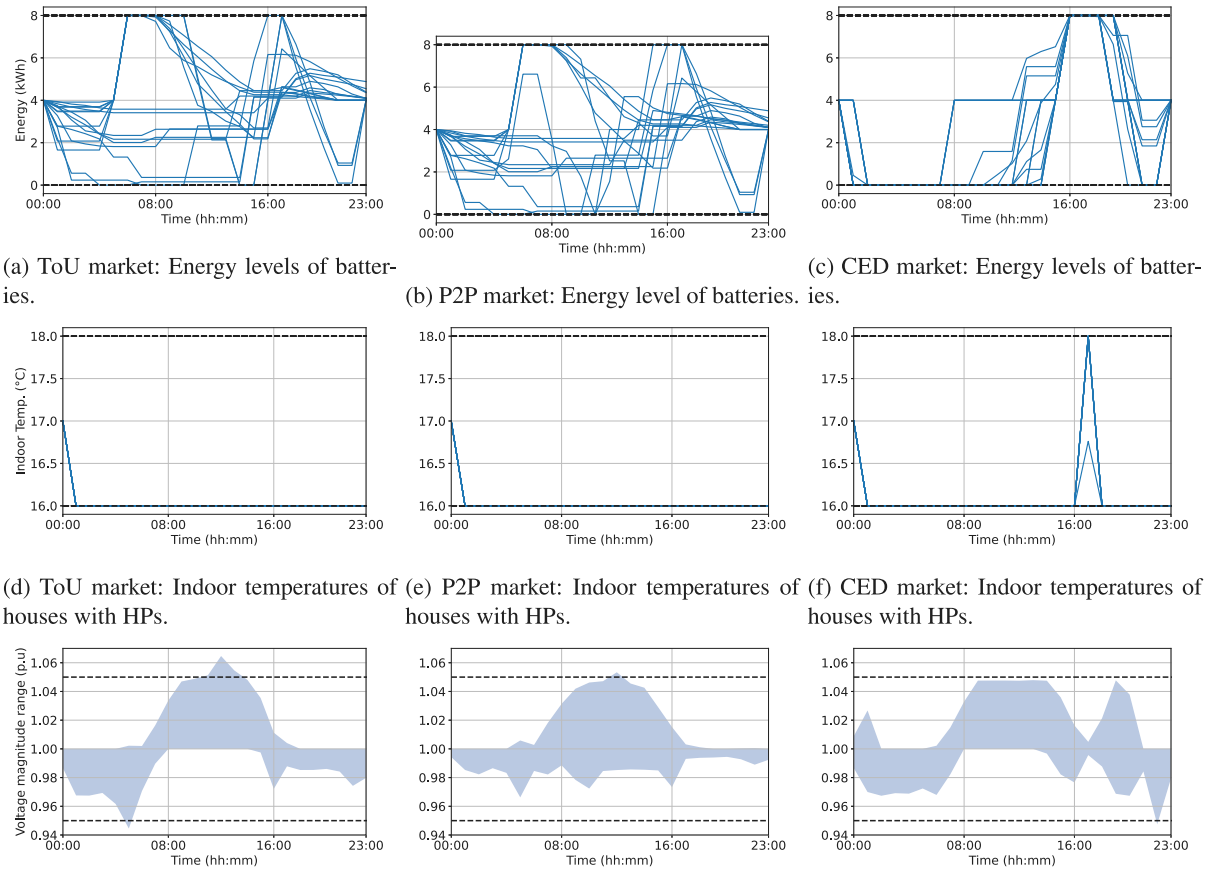
Three case studies were developed in OPLEM to demonstrate its capabilities. The first case study presents a comparison between the ToU market, the P2P market and the CED market. The second case study compares the computational requirements for a scheduling task of a charging station with and without the aggregation feature, and the last case study investigates the flexibility potential of residential HVACs to participate in a long-term local flexibility market (e.g., sustain product that aims to defer network reinforcement [38])

The simulations were run on a laptop with a Processor Intel® Core™ i7 @ 2.80 GHz, with 32 GB RAM using picos python package for solving optimisation problems.

4.1. Market design comparison

4.1.1. Setup

Three markets are compared using OPLEM: ToU, P2P and CED markets, the code source for the three market studies can be found at the root of the tool repository https://github.com/EsaLaboratory/



(a) ToU market: Energy levels of batteries.

(b) P2P market: Energy level of batteries.

(c) CED market: Energy levels of batteries.

(d) ToU market: Indoor temperatures of houses with HPs.

(e) P2P market: Indoor temperatures of houses with HPs.

(f) CED market: Indoor temperatures of houses with HPs.

(g) ToU market: Network voltages magnitude.

(h) P2P market: Network voltages magnitude.

(i) CED market: Network voltages magnitude.

Fig. 6. Energy levels of the 17 batteries stay within their limits. Indoor temperatures of the 17 houses with HPs are kept within the zone of thermal comfort. Voltage magnitudes exceed the maximum voltage limits at midday for the ToU and P2P markets and remains within an acceptable range for the CED market thanks to the incorporation of network constraints in the economic dispatch.

OPLEM. The European Low Voltage Test Feeder (EULV) with 55-load buses is selected as the network infrastructure in the study. In addition to the inflexible load (taken from [39]), 60% of the load buses are randomly selected to have photovoltaic panels, 30% have batteries and 30% have heat pumps. Fig. 5 shows the network setup and Table 2 summarises the assets parameters used for the case studies.

For the import prices, we take the hourly wholesale prices of day 29/06/2021 from the EPEX spot website [40], the export price is FiT tariff taken from OFGEM reports [41] (6.79 cents/kWh).

4.1.2. Results

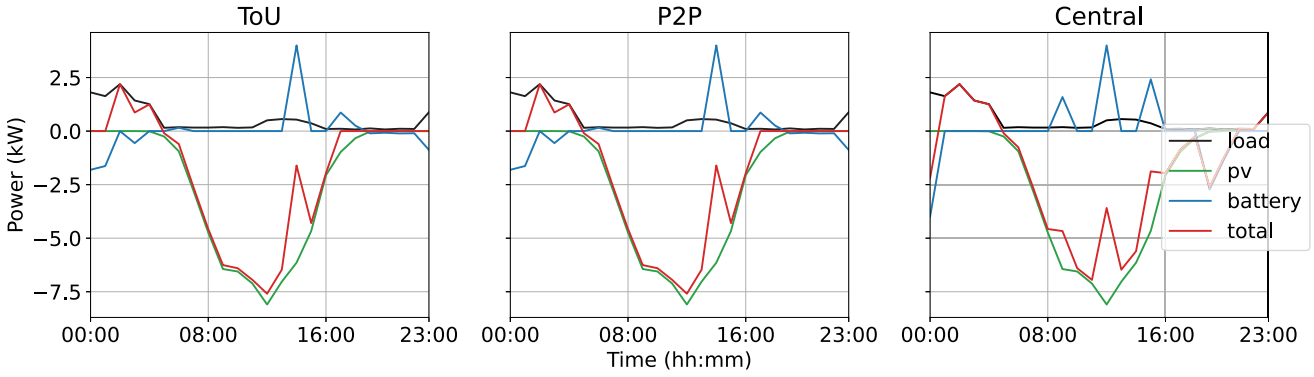
Plots in Fig. 6 give an overview of the three market outcomes. In Figs. 6(a)–6(c) we see that the operational constraints of the storage assets for the ToU, P2P and CED markets are respected. Similarly, the comfort limits constraints of the building assets (Figs. 6(d)–6(f)) were not exceeded and this demonstrates the effectiveness of the `polytope()` method. Also, we can see that the consideration of network constraints in the clearing of the CED market case study helps keep the voltages within the limits (Fig. 6(i)), while we notice a maximum limit voltage violation for both ToU (in Fig. 6(g)) and P2P (in Fig. 6(h)) markets during the period of high PV generation. In Fig. 7, we take a closer look at the assets' schedules for two participants. The participant in Fig. 7(a) owns three kinds of assets: a non-dispatchable load, a curtailable PV generation and a battery storage. For all the markets, we can see the battery charging in the period of high PV generation and discharge during the period of price increase. The battery curve in the case of the CED market shows more charging during the PV peak period to avoid curtailment or network violation due to excessive export, while for the other markets, the majority of PV generation was injected into the grid. The battery of the participant in Fig. 7(b) shows different behaviour for the three markets. In ToU market, the battery is charging overnight to benefit from the cheap energy rates, while in P2P market, the charging is delayed until midday to benefit from the cheaper P2P energy rates. In the CED market, to avoid the violations of voltage constraints resulting from the excessive injection of generation, the battery is scheduled to charge during the generation peak period.

In Fig. 8, we can see that the total exports between the ToU market and CED market are mostly the same, except for the peak period (around 10 am–1 pm) during which the CED market directs some of the excess generation into the batteries charging to avoid exceeding the voltage limits as demonstrated in Fig. 6(i). The total export of the P2P market exceeds the total export of the ToU market with fewer network violations. This is explained by the fact that P2P market enables mutually beneficial local transactions to be arranged, which increases the utilisation of energy storage.

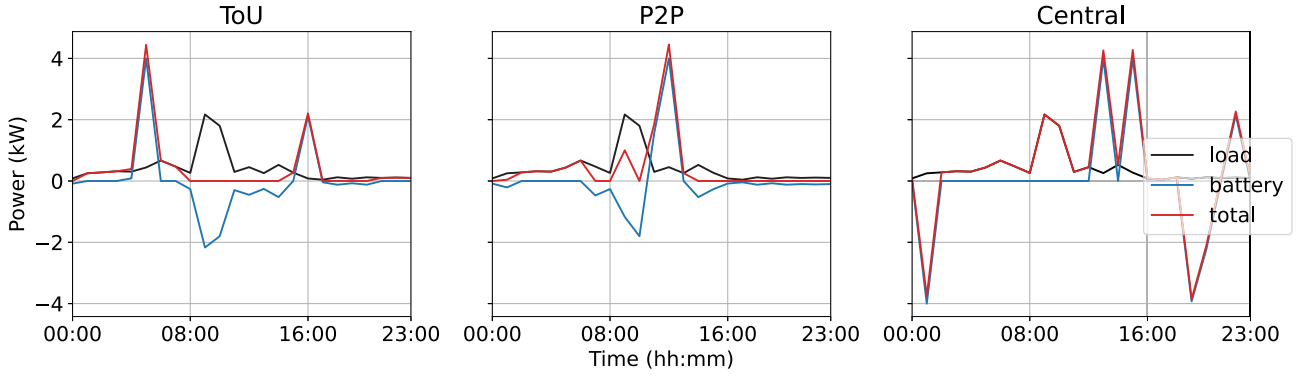
With ToU and CED markets, participants are able to trade only with the upstream market, while for P2P, trading is allowed between

Table 2
Case studies parameters.

	Parameter	Value
PV panels	Nominal power	8 kW
Batteries	Capacity	8 kWh
	Rating power	4 kW
	Degradation cost	0.005 £/kW
Heat pumps	Rated capacity	5.6 kW
	Coefficient of performance	2.5
	Building heat transfer (R)	2 °C/kW
	Building thermal mass (C)	2 kWh/°C
	Min/Max indoor comfort temperature [19]	16, 18 °C
	T_0	17 °C
P2P parameters	Price increment	0.06 £
	Number of groups	2
	Discrete amount of energy per trade	1 kWh
	Maximum number of allowable potential trades per participant	500



(a) Schedules of a participant showing the charging of the battery during the peak generation of the PV, and less export for the case of the CED market.



(b) Schedules of a participant showing the battery delaying the charging until 12 pm to take benefit of cheap P2P prices compared to ToU market.

Fig. 7. Different behaviours of batteries under the three markets.

participants. With the selected P2P parameters (shown in Table 2), 352 kWh was exchanged locally in the P2P market, and this helped reduce the network violations as can be seen in Fig. 6(h), even though no participant had information about the state of the network. The P2P market can be seen as a middle ground between the ToU and the CED market. When the parameters `trade_energy`, `price_inc` and `N_p2p_ahead_max` are selected such that they balance the trade-off between computational effort and local community autonomy, it can help mitigate network constraints, increase energy autonomy and improve community revenues without the need to centralise data and scheduling at a central point.

4.1.3. DLMPs

To demonstrate the calculation of DLMPs, we ran a CED market clearing that is slightly different than the original setup; the HPs and batteries are omitted and only loads and PV panels are kept to see the

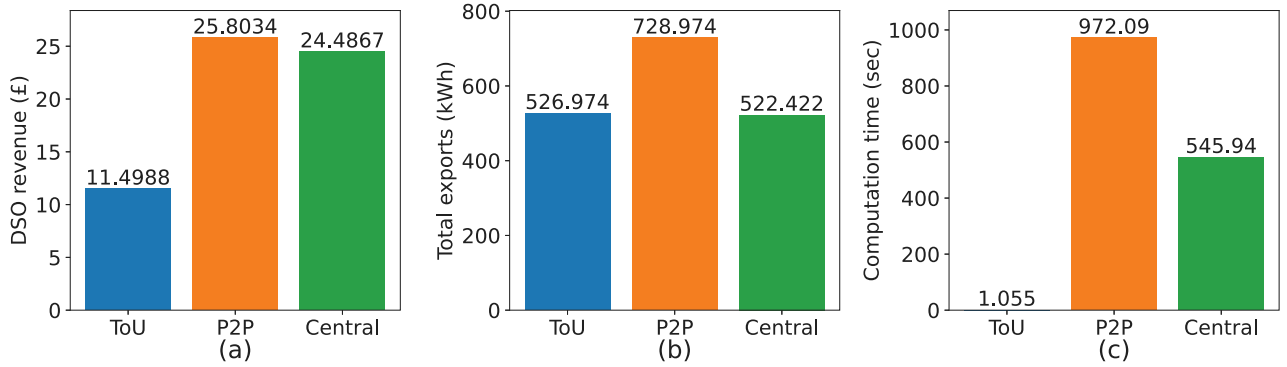


Fig. 8. DSO revenue, total exports, and computation times for the three markets.

Table 3
EV parameters for the aggregation case studies.

Parameter	Description	Values
N_{agg}	Number of aggregators	{1,2,4,5}
N_{EVs}	Number of EVs	{20,40,60,80,100}
η_{eff}	Charging/discharging efficiency	[0.8,1]
E_{max}, P_{max}	Capacity & rated power	36 kWh, 6.6 kW
SOC_{arr}	State of charge at arrival	[0.2, 0.9]
t_{arr}	Time at arrival	between 5 pm and 9 pm
t_{dep}	Time at departure	between 6 am and 9 am

impact of the excess generation on the DLMPs. Fig. 9 shows the DLMPs retrieved as a part of the CED market clearing method. They follow the import price curve in the night period when there is no generation, and vary in the period of generation (8 am to 4 pm), with prices close to 0 around midday, which is associated with curtailment to limit the export of energy that would otherwise cause voltage violations. We note that this information is available only if the optional argument `nw_const` is set to `True` when initiating a CED market instance.

4.2. Aggregation

Algorithm 2 ToU response without aggregation

A_{ev}, b_{ev} = polytope(ev), $\forall ev \in EVs$
Variables: $(p_{ev})_{ev \in EVs}$
solve optimisation problem:
min $\sum_{t \in \mathcal{T}} \sum_{ev \in EVs} p^t_{ToU} p^t_{ev}$
s.t : $A_{ev} \cdot p_{ev} \leq b_{ev}$ $\forall ev \in EVs$
return $(p_{ev})_{ev \in EVs}$

Algorithm 3 ToU response with aggregation

A_{agg}, b_{agg} = polytope(EVs)
Variables: p_{agg}, $(p_{ev})_{ev \in EVs}$
solve optimisation problem:
min $\sum_{t \in \mathcal{T}} p^t_{ToU} p^t_{agg}$
s.t : $A_{agg} \cdot p_{agg} \leq b_{agg}$
return p_{agg}
$(p_{ev})_{ev \in EVs} \leftarrow$ power_disaggregation(p_{agg})
return $(p_{ev})_{ev \in EVs}$

In this section, the application of the aggregation feature in the market design is demonstrated. EVs with different charging rates, different arrival and departure times and different states of charge at arrival are connected to different charging stations. Charging stations in this case play the role of aggregators. Table 3 summarises the parameters of the case studies. Each charging station optimally schedules the charging/discharging profiles of its portfolio of EVs in response to a ToU price signal. Two approaches are compared: the conventional optimisation in which the charging station EMS has to incorporate

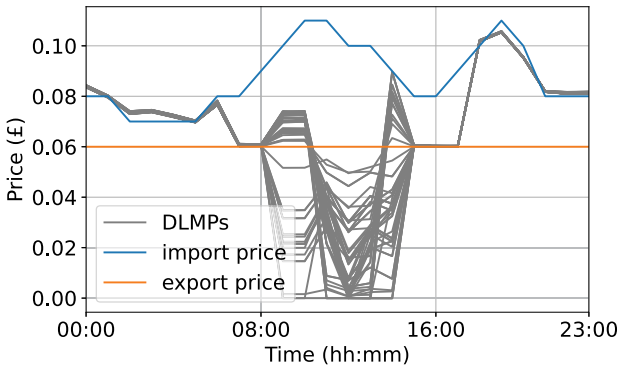


Fig. 9. Import prices, export prices and distributed locational marginal prices resulting from clearing the CED market.

the technical constraints of all the EVs as depicted in Algorithm 2, and the alternative approach that calls for the `polytope()` method, perform the optimisation and returns a disaggregated solution as outlined in Algorithm 3. The number of charging stations was varied between one and five, and the number of EVs in the range of [20, 100], i.e., for 20 EVs, if the number of charging stations is two, it implies an equal distribution of the 20 EVs between the two charging stations. Fig. 10 illustrates the time required for aggregation and optimisation as a function of the number of Electric Vehicles (EVs) participating in the aggregation. The left panel displays the time it takes to compute the aggregated (A_{agg}, b_{agg}) across different scenarios, parameterised by the number of aggregators, and the right panel depicts the optimisation time under similar scenarios and includes an additional scenario without aggregation (no agg). The plots showing the computation time of optimisation featuring aggregation include also the time it takes to recover a feasible disaggregate solution (using `power_disaggregation()` method) from the aggregated solution p_{agg}. The time required for aggregation increases as the number of EVs (N EVs) rises, with higher number of aggregators generally requiring less time. For more than 2 charging stations (e.g., aggregators), it takes up to 30 min to compute the aggregated polytope. This trend highlights the computational benefit associated with the presence of more than one aggregator. Similar to aggregation, the optimisation time also increases with the number of EVs. Notably, the 'no agg' scenario demonstrates lower optimisation time compared to the case where one aggregator exists, however, scenarios with other aggregation levels (1 to 5 agg) show progressively lower optimisation times compared to the 'no agg' scenario, emphasising the trade-off between aggregation benefits and computational efficiency. The analysis provides valuable insights into the scalability of the aggregation process, it is crucial for understanding the balance between selecting optimal number of aggregators and maintaining computational efficiency in

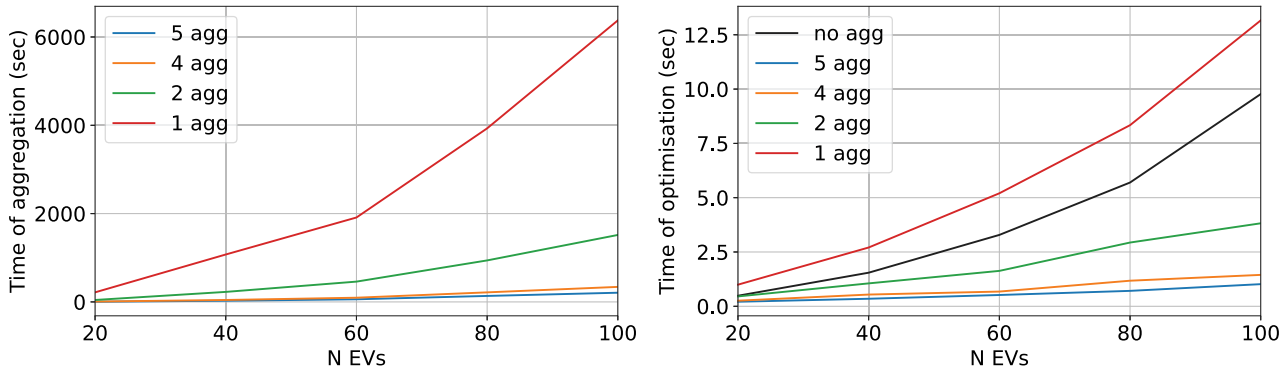


Fig. 10. Computational time for aggregation. (left) Time for computing the aggregated polytope, (right) Time for optimisation without and with aggregated polytopes, including time to recover a feasible disaggregate solution.

real-world applications. Specifically, the analysis demonstrates that utilising multiple aggregators is computationally more efficient than having no aggregator, while having no aggregator is more efficient than using a single aggregator.

4.3. Flexible capacity of residential HVACs

In this case study, the potential of a medium-sized house heat pump to participate in a local flexibility market and provide flexibility to a system operator within the time frame of [7 pm, 10 pm] is investigated. This is particularly of interest for assets managers aiming to respond to calls for tenders issued by DSOs that seek flexibility provision over a long term to defer network upgrades. The heat pump with parameters from Table 2 is selected for the study, the ambient temperature profile of Edinburgh for the year 2019 from [42], and the UK E7 Tariff as a ToU price (7 h of cheaper energy between midnight and 7 am) with night rate and day rate being 9.76 p/kWh and 20.03 p/kWh respectively. Fig. 11 shows the results of the feasibility study. The ambient temperature was split into two seasons: winter and summer. The baseline consumption was calculated using either `maxdemand_baseline()` or `toup_baseline()`, and the available flexibility using `flexibility()`. Therefore, the computation of available flexibility per sample (i.e., day) involves running two optimisation algorithms and takes an average of 0.3 s. For a one-year assessment, this totals to 109.5 s.

The results show a high variance in the available flexibility when the adopted baseline is a `maxdemand_baseline()`, especially for the summer period, while the flexibility is uniform for the `toup_baseline()`. The available flexibility has a median value of 0.18 kW, except for the summer period when using `maxdemand_baseline()` as a baseline, where the median drops to 0.11 kW. The mean values are respectively 0.17, 0.11, 0.18 and 0.14 kW for winter `maxdemand_baseline()`, summer `maxdemand_baseline()`, winter `toup_baseline()` and summer `toup_baseline()`, with the case of winter `toup_baseline()` returning the most uniform results. In the UK scenario, due to the minimum bid quantity of 10 kW, this asset cannot independently enter the market as a participant and must be aggregated with other assets. If an aggregator manages this asset among others, it is advisable to adopt the baseline schedule that minimises daily energy consumption costs rather than minimising peak demand. This approach offers reduced variation and yields consistent available flexibility across seasons.

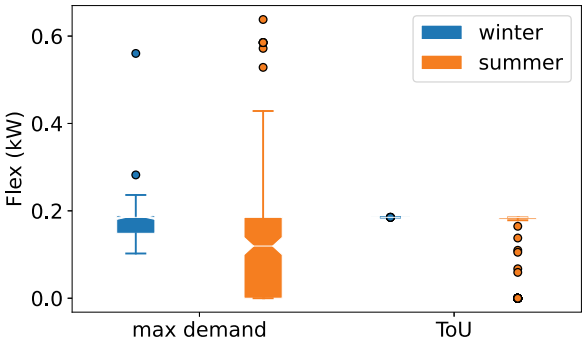


Fig. 11. Potential of a heat pump to provide flexibility under different baselines in two seasons, winter and summer.

5. Conclusion

In this paper, OPLEM, an open-source platform for LEMs is presented. The tool was developed in a modular way to accelerate the creation, simulation and testing of LEMs. It addresses the need for a standardised LEM modelling software that enables the replication, refinement and comparison of LEM designs. The platform can serve as a standard environment to benchmark existing and new LEMs and help the research community focus on innovation instead of spending time re-creating the simulation environment. The key features of OPLEM include methods for flexibility computation and energy baseline computation, a polytope representation for flexible resource operational constraints and four energy markets. The features can be used differently and extended to respond to the designer's needs. To highlight the wider potential uses of our software platform, three case studies were demonstrated. Case study 1 compared three market designs incorporated with the tool, along with an analysis of network simulation, and interaction with connected assets. This comparison illustrated the versatility of OPLEM in handling various market scenarios and their impact on network operations. Case study 2 demonstrated the use of the aggregation features for charging stations that operate multiple EVs, showing how the platform can optimise the management of DERs by aggregating assets to improve computational efficiency and market participation. Case study 3 leveraged the flexibility features incorporated within the tool by demonstrating a flexibility assessment of medium-sized home heat pump. This case study emphasised OPLEM's ability to assess and utilise the flexibility of DERs to enhance grid reliability and efficiency. These case studies collectively underscore the comprehensive capabilities of OPLEM in modelling, testing, and optimising local energy market designs, thus paving the way for its application in diverse real-world settings. For future releases of the tool, OPLEM is aimed to be made AI-compatible to allow the integration, testing and benchmarking of AI-powered LEM solutions. Another area of extension is creating a graphical user interface. It will be useful

for stakeholders from different backgrounds (policy-makers, local authorities) as it will help them use the tool without the need for a programming background.

CRediT authorship contribution statement

Chaimaa Essayeh: Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Thomas Morstyn:** Writing – review & editing, Supervision, Funding acquisition, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The links to the used data is shared in the 'References' section of the paper.

Acknowledgements

The present work has been supported by the EPSRC, United Kingdom grants EP/S031901/1 and EP/T028564/1.

Appendix. Feasible solution recovery

The formula that recovers a feasible solution **p** from an infeasible approximate aggregate solution p_{agg} is presented in [35] and is as follows:

$$\min_{p_m, m=1,\dots,M} \quad \left\| p_{agg} - \sum_{i=1}^{M} p_m \right\| \qquad \text{s.t.} \quad A_m p_m \le b_m, \quad n = 1, \dots, M \tag{5}$$

where p_{agg} refers to the outer approximation.

References

[1] Hoke Anderson, Butler Rebecca, Hambrick Joshua, Kroposki Benjamin. Maximum photovoltaic penetration levels on typical distribution feeders. Technical report, Golden, CO (United States): National Renewable Energy Lab.(NREL); 2012.
[2] Baruah Pranab J, Eyre Nicholas, Qadrdan Meysam, Chaudry Modassar, Blainey Simon, Hall Jim W, et al. Energy system impacts from heat and transport electrification. Proc Inst Civ Eng-Energy 2014;167(3):139–51.
[3] Pinson Pierre. What may future electricity markets look like? J Mod Power Syst Clean Energy 2023.
[4] Essayeh Chaimaa, Raiss El-Fenni Mohammed, Dahmouni Hamza. Cost-effective energy usage in a microgrid using a learning algorithm. Wirel Commun Mob Comput 2018;2018.
[5] Papavasiliou Anthony. Analysis of distribution locational marginal prices. IEEE Trans Smart Grid 2018;9(5):4872–82. http://dx.doi.org/10.1109/TSG.2017.2673860.
[6] Morstyn Thomas, Teytelboym Alexander, McCulloch Malcolm D. Bilateral contract networks for peer-to-peer energy trading. IEEE Trans Smart Grid 2018;10(2):2026–35.
[7] Essayeh Chaimaa, Raiss El-Fenni Mohammed, Dahmouni Hamza, Ahajjam Mohamed Aymane. Energy management strategies for smart green microgrid systems: a systematic literature review. J Electr Comput Eng 2021;2021:1–21.
[8] Luna Adriana C, Diaz Nelson L, Graells Moises, Vasquez Juan C, Guerrero Josep M. Mixed-integer-linear-programming-based energy management system for hybrid PV-wind-battery microgrids: Modeling, design, and experimental verification. IEEE Trans Power Electron 2016;32(4):2769–83.
[9] Yang Jiachen, Jiang Bin, Lv Zhihan, Choo Kim-Kwang Raymond. A task scheduling algorithm considering game theory designed for energy management in cloud computing. Future Gener Comput Syst 2020;105:985–92.
[10] Chowdhury Badrul H, Rahman Saifur. A review of recent advances in economic dispatch. IEEE Trans Power Syst 1990;5(4):1248–59.
[11] Tan Zhenfei, Cheng Tong, Liu Yuchen, Zhong Haiwang. Extensions of the locational marginal price theory in evolving power systems: A review. IET Gener Transm Distrib 2022;16(7):1277–91. http://dx.doi.org/10.1049/gtd2.12381.
[12] Zhou Yue, Wu Jianzhong, Long Chao, Ming Wenlong. State-of-the-art analysis and perspectives for peer-to-peer energy trading. Engineering 2020;6(7):739–53. http://dx.doi.org/10.1016/j.eng.2020.06.002.
[13] Soto Esteban A, Bosman Lisa B, Wollega Ebisa, Leon-Salas Walter D. Peer-to-peer energy trading: A review of the literature. Appl Energy 2021;283:116268.
[14] Suthar Sachinkumar, Cherukuri SHari Charan, Pindoriya Naran M. Peer-to-peer energy trading in smart grid: Frameworks, implementation methodologies, and demonstration projects. Electr Power Syst Res 2023;214:108907.
[15] Thukral Manish Kumar. Emergence of blockchain-technology application in peer-to-peer electrical-energy trading: a review. Clean Energy 2021;5(1):104–23. http://dx.doi.org/10.1093/ce/zkaa033.
[16] Baroche Thomas, Pinson Pierre, Latimier Roman Le Goff, Ahmed Hamid Ben. Exogenous cost allocation in peer-to-peer electricity markets. IEEE Trans Power Syst 2019;34(4):2553–64.
[17] Savelli Iacopo, Morstyn Thomas. Electricity prices and tariffs to keep everyone happy: A framework for fixed and nodal prices coexistence in distribution grids with optimal tariffs for investment cost recovery. Omega 2021;103:102450. http://dx.doi.org/10.1016/j.omega.2021.102450.
[18] Ringkjøb Hans-Kristian, Haugan Peter M, Solbrekke Ida Marie. A review of modelling tools for energy and electricity systems with large shares of variable renewables. Renew Sustain Energy Rev 2018;96:440–59. http://dx.doi.org/10.1016/j.rser.2018.08.002.
[19] Morstyn Thomas, Collett Katherine A, Vijay Avinash, Deakin Matthew, Wheeler Scot, Bhagavathy Sivapriya M, et al. OPEN: An open-source platform for developing smart local energy system applications. Appl Energy 2020;275:115397. http://dx.doi.org/10.1016/j.apenergy.2020.115397.
[20] Thurner Leon, Scheidler Alexander, Schäfer Florian, Menke Jan-Hendrik, Dollichon Julian, Meier Friederike, et al. Pandapower—An open-source python tool for convenient modeling, analysis, and optimization of electric power systems. IEEE Trans Power Syst 2018;33(6):6510–21. http://dx.doi.org/10.1109/TPWRS.2018.2829021.
[21] Zimmerman Ray Daniel, Murillo-Sánchez Carlos Edmundo, Thomas Robert John. MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education. IEEE Trans Power Syst 2011;26(1):12–9. http://dx.doi.org/10.1109/TPWRS.2010.2051168.
[22] Gonzalez-Longatt Francisco, Rueda José. PowerFactory applications for power system analysis. Springer; 2015, http://dx.doi.org/10.1007/978-3-319-12958-7.
[23] David P. Chassin, Jason C. Fuller, Ned Djilali. GridLAB-D: An agent-based simulation framework for smart grids. J Appl Math 2014;2014. http://dx.doi.org/10.1155/2014/492320.
[24] Lie Arne Ø, Rye Eirik A, Svendsen Harald G, Farahman Hossein, Korpås Magnus. Validation study of an approximate 2014 European power-flow model using powergama. IET Gener Transm Distrib 2017;11(2):392–400. http://dx.doi.org/10.1049/iet-gtd.2016.0856.
[25] Brown Thomas, Hörsch Jonas, Schlachtberger David. PyPSA: Python for power system analysis. J Open Res Software 2018. http://dx.doi.org/10.5334/jors.188.
[26] The energy analytics and decision platform for all systems. 2023, https://www.energyexemplar.com/plexos.
[27] Abrell Jan, Kunz Friedrich. Integrating intermittent renewable wind generation-a stochastic multi-market electricity model for the european electricity market. Netw Spat Econ 2015;15:117–47.
[28] Panagiotis Fragkos, Nikos Tasios, Leonidas Paroussos, Pantelis Capros, Stella Tsani. Energy system impacts and policy implications of the European intended nationally determined contribution and low-carbon pathway to 2050. Energy Policy 2017;100:216–26. http://dx.doi.org/10.1016/j.enpol.2016.10.023.
[29] Dugan Roger C, McDermott Thomas E. An open source platform for collaborating on smart grid research. In: 2011 IEEE power and energy society general meeting. IEEE; 2011, p. 1–7.
[30] OPLEM: Open platform for local energy markets. 2023, https://github.com/EsaLaboratory/OPLEM.
[31] Essayeh Chaimaa, Zhou Yihong, Morstyn Thomas. OPLEM, read the docs. 2024, https://open-new.readthedocs.io/en/latest/.
[32] Bernstein Andrey, Dall'Anese Emiliano. Linear power-flow models in multiphase distribution networks. In: 2017 IEEE PES innovative smart grid technologies conference Europe. 2017, p. 1–6. http://dx.doi.org/10.1109/ISGTEurope.2017.8260205.
[33] Biagioni David, Zhang Xiangyu, Wald Dylan, Vaidhynathan Deepthi, Chintala Rohit, King Jennifer, et al. Powergridworld: A framework for multi-agent reinforcement learning in power systems. In: Proceedings of the thirteenth ACM international conference on future energy systems. 2022, p. 565–70.
[34] Gao Yuan, Matsunami Yuki, Miyata Shohei, Akashi Yasunori. Multi-agent reinforcement learning dealing with hybrid action spaces: A case study for off-grid oriented renewable building energy system. Appl Energy 2022;326:120021. http://dx.doi.org/10.1016/j.apenergy.2022.120021, URL https://www.sciencedirect.com/science/article/pii/S0306261922012788.
[35] Barot Suhail, Taylor Josh A. A concise, approximate representation of a collection of loads described by polytopes. Int J Electr Power Energy Syst 2017;84:55–63. http://dx.doi.org/10.1016/j.ijepes.2016.05.001.
[36] Morstyn Thomas, Teytelboym Alexander, Mcculloch Malcolm D. Bilateral contract networks for peer-to-peer energy trading. IEEE Trans Smart Grid 2019;10(2):2026–35. http://dx.doi.org/10.1109/TSG.2017.2786668.

[37] Morstyn Thomas, Teytelboym Alexander, Hepburn Cameron, McCulloch Malcolm D. Integrating P2P energy trading with probabilistic distribution locational marginal pricing. IEEE Trans Smart Grid 2019;11(4):3095–106.
[38] SSEN. Flexibility service procurement. 2023, https://www.ssen.co.uk/our-services/flexible-solutions/flexibility-services/.
[39] Thomson M, Richardson I. One-minute resolution domestic electricity use data. 2008-2009, http://dx.doi.org/10.5255/UKDA-SN-6583-1, Accessed: 05/2023.
[40] EPEXSPOT. Market data. 2021, https://www.epexspot.com/en/market-data.
[41] OFGEM. Feed-in tariff (FIT): Tariff table 1 april 2023. 2023, https://www.ofgem.gov.uk/sites/default/files/2023-01/FIT%20Rates%20RPI%20Update%2023-24.xlsx.
[42] Pfenninger Stefan, Staffell Iain. Renewables.ninja. 2019, https://www.renewables.ninja/.