OPEN: An Open-Source Platform for Developing Smart Local Energy System Applications

Thomas Morstyn[a,*], Katherine A. Collett[a], Avinash Vijay[a], Matthew Deakin[b], Scot Wheeler[a], Sivapriya M. Bhagavathy[a], Filiberto Fele[a], Malcolm D. McCulloch[a]

[a]*Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, United Kingdom*
[b]*School of Engineering, Merz Court, Newcastle University, Newcastle upon Tyne, NE1 7RU, United Kingdom*

Abstract

This paper presents OPEN, an open-source software platform for integrated modelling, control and simulation of smart local energy systems. Electric power systems are undergoing a fundamental transition towards a significant proportion of generation and flexibility being provided by distributed energy resources. The concept of 'smart local energy systems' brings together related strategies for localised management of distributed energy resources, including active distribution networks, microgrids, energy communities, multi-energy hubs, peer-to-peer trading platforms and virtual power plants. OPEN provides an extensible platform for developing and testing new smart local energy system management applications, helping to bridge the gap between academic research and industry translation. OPEN combines features for managing smart local energy systems which are not provided together by existing energy management tools, including multi-phase distribution network power flow, energy market modelling, nonlinear energy storage modelling and receding horizon optimisation. The platform is implemented in Python with an object-oriented structure, providing modularity and allowing it to be easily integrated with third-party packages. Case studies are presented, demonstrating how OPEN can be used for a range of smart local energy system applications due to its support of multiple model fidelities for simulation and control.

Highlights

- Presents the Open Platform for Energy Networks (OPEN), github.com/EPGOxford/OPEN
- Integrated modelling, control & simulation framework for smart local energy systems
- The object-oriented approach offers modularity, code reuse & extensibility
- Development has been motivated by four industry–academic demonstration projects
- Case studies demonstrate how OPEN can be extended for new applications

Keywords: Distributed energy resource, distribution network, modelling, Python, open-source software, smart local energy system.

1. Introduction

Electric power systems are undergoing a fundamental transition, away from the traditional model of centralised generation, towards a significant proportion of generation and flexibility being provided locally by distributed energy resources (DERs), including renewable sources, electric vehicles (EVs) and heat pumps. Coordinating DERs on a localised basis could offer significant value by reducing upstream power flows and losses, alleviating the need to curtail renewable generation and enabling the deferral of distribution, transmission and generation infrastructure upgrades [1]. There is also the potential for aggregated groups of DERs to offer ancillary services upstream as a 'virtual power plant' [2], or for DER clusters to support autonomous 'microgrid' operation [3].

The concept of 'smart local energy systems' brings together related strategies for localised management of DERs [4], including active distribution networks [5], microgrids [6], energy communities [7], multi-energy hubs [8], peer-to-peer trading platforms [9], distribution flexibility markets [10], virtual power plants [11] and federated power plants [12]. These strategies are distinct, but they also overlap and have the potential to be combined. Smart local energy systems have a com-

*Corresponding author
Email address: thomas.morstyn@eng.ox.ac.uk (Thomas Morstyn)

mon set of elements: a set of DERs; an interconnecting local power network; an upstream energy market; and a digital coordination platform providing sensing, communications and control.

The UK Government has invested £102.5m in the *Prospering from the Energy Revolution Challenge*, towards the design, development and demonstration of smart local energy systems [13]. The objective is to make the best use of cheaper renewables, energy storage, EVs, energy efficiency, low carbon heat and digital infrastructure. International programmes supporting smart local energy system research and development include *EN SGplusRegSys* [14] and *IO.Energy* in the EU [15], the *Brooklyn Microgrid* in the US [16] and the *Decentralised Energy Exchange* (deX) in Australia [17].

The authors are contributing to four smart local energy system innovation projects involving industry–academic collaboration [18]:

(i) *Vehicle-to-Grid Oxfordshire* (V2GO): an optimisation platform demonstration for coordinating EV delivery fleets to charge at lowest cost and provide ancillary services upstream [19].

(ii) *Multi-Sites, Actors, Vectors, Energy Services* (Multi-SAVES): a smart building demonstration of flexible heating ventilation and air conditioning (HVAC) and solar generation to reduce energy costs and carbon emissions [20].

(iii) *Power Energy Technology Efficiency* (PETE): a virtual power plant demonstration made up of 500 smart electric hot water tanks and 100 home battery systems [21].

(iv) *Local Energy Oxfordshire* (LEO): a demonstration of a county-wide market platform for distribution system flexibility bringing together and coordinating 90 plug-in projects (including solar generation, microgrids and EV hubs) [22].

These projects are diverse, focusing on a range of different DER technologies, including EVs, flexible HVAC, residential batteries and solar generation; and are based on different coordination strategies, including site-level optimisation, virtual power plant aggregation and distribution flexibility markets. Despite this variety, they each require a common set of management tools: modelling, to understand the characteristics and constraints of DERs and the local network; control, to achieve the system objectives within operating constraints; and detailed simulation, to verify performance ahead of implementation.

Recent academic interest in smart local energy systems has led to the application of advanced modelling and control techniques. These include technology-specific nonlinear DER modelling (including for battery storage [23], low-carbon heat [24] and power electronics [25]), computationally scalable optimal power flow for unbalanced multi-phase distribution networks [26] and receding horizon model predictive control (MPC) to address model approximations and uncertainty when scheduling DERs [27].

Despite this, current energy management software tools do not adequately support smart local energy system research and development. The capability to model, control and simulate distribution systems with embedded DERs is divided between multiple tools, with a lack of extensibility and interoperability, and advanced modelling and control techniques are not offered. This creates challenges for collaboration and replication between laboratories, slowing the development and testing of new methods and the translation of these methods to industry application. Another important gap is the inability of existing tools to support a range of model fidelities for DERs and networks.

To address these challenges, this paper presents the Open Platform for Energy Networks (OPEN). OPEN is an open-source Python platform for developing and testing smart local energy system management applications. It provides an extensible object-oriented platform for integrated modelling, control and simulation. The development of OPEN has been motivated by gaps identified with existing energy management tools, along with the increasing importance of smart local energy systems as an architecture for renewable integration, off-grid electrification, and the electrification of heat and transport. The latest version of OPEN is available for download [28], along with full documentation [29] .

The rest of the paper is organised as follows: First, a detailed comparison between the features offered by OPEN and those offered by existing energy management tools is provided. Then, OPEN's object-oriented structure and program flow are described. Two case studies are presented, demonstrating OPEN applications for building energy management and EV smart charging. The Appendices detail the modelling and optimisation techniques used by OPEN's classes.

2. Comparison with Existing Tools

Table 1 provides a comparison between OPEN and other popular energy management software tools. Existing software tools can be broadly divided into three groups: power network simulation tools; energy system management tools; and transmission system management tools. A more comprehensive review of software tools for energy system modelling and management is presented in [45].

Power network simulation tools focus on detailed modelling and simulation of transmission or distribution power flows, which are important for system planning and dispatch. Examples include pandapower [30], MATPOWER [31], DiSC [32], PowerFactory [33], GridLAB-D [34] and OpenDSS [35]. These tools are focused on traditional generation, and do not consider

Table 1: Comparison between OPEN and other energy management software tools.

Software	Reference	Open Source	Linear Storage Models	Nonlinear Storage Models	Upstream Energy Market	Demand Charges	Power Flow Simulation*	Power Flow Optimisation*	Separate Opt. & Sim. Models	Multi-Period Optimisation	Receding Horizon Opt.
pandapower	[30]	✓	✗	✗	✗	✗	BN	BN	✗	✗	✗
MATPOWER	[31]	✓	✗	✗	✗	✗	BN	BN	✗	✗	✗
DiSC	[32]	✓	✓	✗	✗	✗	BN	✗	✗	✗	✗
PowerFactory	[33]	✗	✗	✗	✗	✗	MN	BN	✗	✗	✗
GridLAB-D	[34]	✓	✓	✗	✓	✗	MN	✗	✗	✗	✗
OpenDSS	[35]	✓	✓	✗	✗	✗	MN	✗	✗	✗	✗
Calliope	[36]	✓	✓	✗	✓	✗	ET	ET	✗	✓	✓
oemof-solph	[37]	✓	✓	✗	✗	✗	ET	ET	✗	✓	✗
OSeMOSYS	[38]	✓	✓	✗	✓	✗	ET	ET	✗	✓	✗
urbs	[39]	✓	✓	✗	✓	✗	ET	ET	✗	✓	✗
PLEXOS	[40]	✗	✓	✗	✓	✗	BL	BL	✗	✓	✓
PyPSA	[41]	✓	✓	✗	✓	✗	BN	BL	✗	✓	✗
TIMES	[42]	✗	✓	✗	✓	✗	BL	BL	✗	✓	✗
Switch 2.0	[43]	✓	✓	✗	✓	✗	BL	BL	✗	✓	✗
PowerGAMA	[44]	✓	✓	✗	✓	✗	BL	BL	✗	✗	✗
OPEN	[28]	✓	✓	✓	✓	✓	MN	ML	✓	✓	✓

*Power flow models used for simulation/optimisation: (ET) energy transfer; (BL) balanced linear; (BN) balanced nonlinear; (ML) multi-phase linear; or (MN) multi-phase nonlinear.

multi-period scheduling relevant for modern systems with energy storage and flexible loads. Energy system management tools focus on higher-level planning. Calliope [36], oemof-solph [37], OSeMOSYS [38] and urbs [39] are examples of energy system management tools. These tools include multi-period scheduling, but use simplified energy transfer network models, rather than detailed electrical power flow modelling. Finally, transmission system management tools address the adoption of renewable generation and energy storage within electric power systems, by combining multi-period optimal scheduling with balanced power flow models suitable for transmission systems. Examples of transmission system management tools include PLEXOS [40], PyPSA [41], TIMES [42], Switch 2.0 [43] and PowerGAMA [44].

An important insight from the smart local energy system projects the authors are contributing to is the need for an integrated platform that can support a range of DER and network models, varying in detail and complexity, for simulation and control. OPEN has been designed with a modular structure to support this, enabling it to offer features relevant to smart local energy systems which are not provided by other tools, including:

(i) Multi-period scheduling of DERs combined with multi-phase distribution network modelling. The balanced power flow models used by the transmission system management tools do not address voltage constraints and losses when DERs are connected on different phases, which is an important consideration for distribution systems [46].

(ii) Receding horizon optimisation, where scheduling is updated during operation based on new forecasts and a predictive system model [47]. This is important for addressing uncertainty (e.g. associated with upstream prices, load, generation, DER availability) and DER/network model approximations required to reduce computational complexity for optimisation. Of the transmission system management tools, only PLEXOS (which is not open source and does not include nonlinear power flow) offers receding horizon optimisation.

(iii) Nonlinear energy storage modelling for simulation (e.g. battery efficiency depends nonlinearly on output power [23]). This is an important feature for comparing different energy storage technology options. Linearised models for storage systems are used for optimisation to maintain computational scalability.

(iv) Separate model fidelities for control and simulation, allowing DERs to be scheduled based on simplified resource and network models. Optimisation based on a full distribution system model is sometimes considered impractical due to computational complexity or a lack of network information. The suitability of a simplified control model can be tested by coupling it with a higher fidelity nonlinear system model for simulation. If this results in constraint violations, designers can then consider more accurate power flow optimisation strategies.

(v) Demand charges (per-kW charges on the maximum demand during a particular time period). Demand charges are an important local energy market pricing mechanism which are commonly imposed to incentivise local supply–demand balancing [48].

3. Platform Structure

OPEN is implemented in Python, which is a widely used programming language for open-source scientific computing [49]. This allows OPEN to be easily integrated with a range of open-source third-party packages. It has been developed using an object-oriented

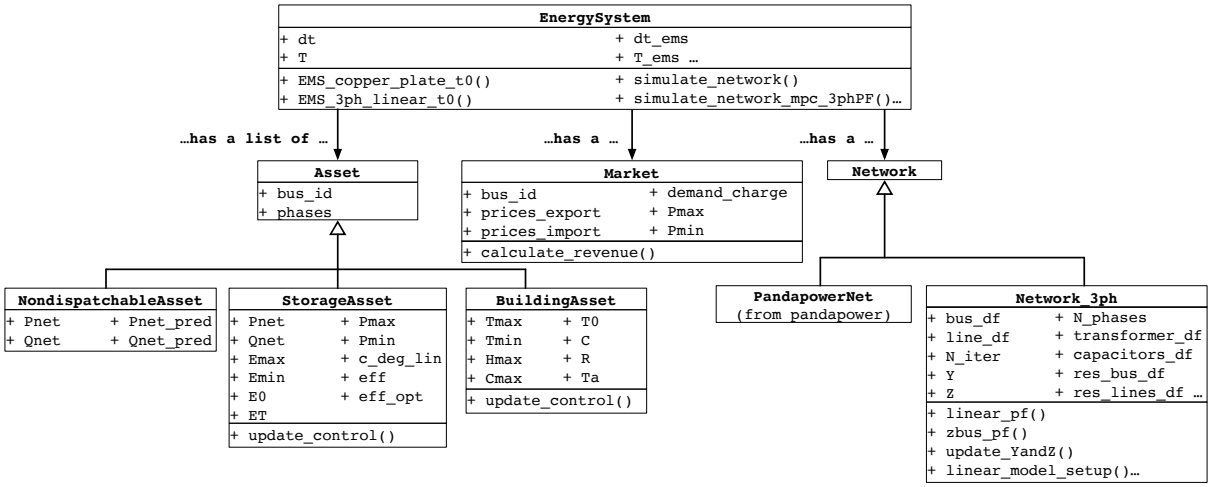

Figure 1: A UML class diagram of OPEN, showing the platform's main classes, attributes, methods, inheritance relationships and associations.

programming approach, providing modularity, code reuse and extensibility.

Fig. 1 shows a universal modelling language (UML) class diagram of OPEN. OPEN has four important base classes: `Asset`, `Network`, `Market` and `EnergySystem`.

3.1. EnergySystem Class

In OPEN, a smart local energy system application is built around an `EnergySystem` object, which has a list of `Asset` objects defining the loads and DERs, a `Network` which the `Asset` objects are embedded within, and an upstream `Market` which the `Network` is connected to.

The `EnergySystem` class has two main types of methods: (i) energy management system (EMS) methods and (ii) simulation methods. EMS methods implement algorithms to calculate `Asset` control references. Simulation methods first call an EMS method to generate control references for `Asset` objects, then update the state of the `Asset` objects by calling their `update_control()` method, and finally update the state of the `Network` by calling its power flow method. An `EnergySystem` has two separate time-series, one for the EMS, and the other for simulation. The resolution of the simulation time-series sets the update rate of the `Asset` states and `Network` power flows, while the resolution of the EMS time-series sets the update rate of `Asset` references. Separating these time-series allows OPEN to simulate intra-interval variability between EMS updates [50].

OPEN includes two EMS methods for controllable `Asset` objects: (i) one for multi-period optimisation with a simple copper plate network model, and (ii) the other for multi-period optimisation with a linear multi-phase distribution network model from [46] which includes voltage and current flow constraints. OPEN has simulation methods for: (i) open-loop optimisation, where the EMS method is run ahead of operation to obtain controllable `Asset` references over the EMS time-series; and (ii) MPC, where the EMS method is implemented with a receding horizon so that the flexible `Asset` references are updated at each step of the EMS time-series.

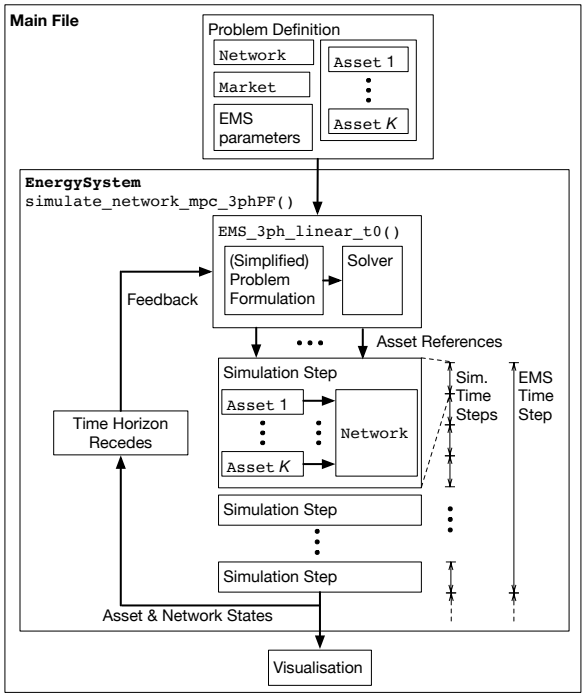

Figure 2: A high-level program flow diagram for an MPC OPEN application. The instantiated objects, method calls and information flows are shown.

Fig. 2 shows a high-level program flow diagram for an example MPC OPEN application. In a new main file, the user first initialises a `Network` object, `Market` object, a list of `Asset` objects and parameters associ-

ated with energy management, which together can be considered the problem definition for the application. The user then initialises an `EnergySystem` object and calls the desired simulation method. Within the MPC simulation method, an EMS method is called which solves an optimisation problem formulated to generate `Asset` control references for the first EMS interval. For each simulation interval within the EMS interval, the states of the `Asset` objects are updated based on the control references, and then the `Asset` output powers are used to update the state of the `Network`. Then, the EMS time horizon recedes by a step, and the EMS method is called again. A more detailed description is provided in Appendix D. Once the simulation is complete, it returns the `Asset` and `Network` states over the simulation time-series, which the user can plot using a data visualisation package, such as Matplotlib [51].

The `EnergySystem` class can be extended by defining new EMS methods. For example, EMS methods could be used to implement more advanced optimisation strategies which account for detailed asset characteristics, decentralised coordination algorithms or market-based scheduling with peer-to-peer energy trading. The requirement for a new EMS method to be interoperable is that it returns a Python dictionary with `Asset` references that can be read by the `EnergySystem` simulation methods that call it. There is no restriction on the scheduling algorithms that are implemented by EMS methods. An EMS method could also be implemented to serve as an interface between OPEN and third-party software for DER coordination, with the EMS method sending system information to the third-party software and receiving control references. In this way, OPEN could be used to test new coordination platforms prior to implementation.

3.2. Asset Class

The `Asset` class is used to define DERs and loads. Attributes include network location, phase connection, and real and reactive output power profiles over the simulation time-series. Flexible `Asset` classes have an `update_control()` method, which is called by `EnergySystem` simulation methods with control references as inputs, and updates the `Asset` object's output power profiles and state variables. Note that an `update_control()` method can include constraints which limit the implementation of control references (e.g. a battery storage system which has reached a minimum energy level will not export power).

OPEN includes the following `Asset` subclasses: `NondispatchableAsset` for uncontrollable loads and generation sources, `StorageAsset` for storage systems and `BuildingAsset` for buildings with flexible HVAC. New `Asset` subclasses can be defined which may have additional attributes and different `update_control()` method implementations.

3.3. Market Class

The `Market` class is used to define an upstream market which the `EnergySystem` is connected to. Attributes include the network location, prices of imports and exports over the simulation time-series, the demand charge paid on the maximum demand over the simulation time-series and import/export power limits. The market class has a method which calculates the total revenue associated with a particular set of real and reactive power profiles.

3.4. Network Class

OPEN offers two options for network modelling. For balanced power flow analysis, the `PandapowerNet` class from the open-source Python package pandapower can be used by OPEN. Balanced power flow is generally acceptable for transmission system studies, but is often not suitable for distribution systems, since they may have single/double-phase spurs, untransposed lines with coupling between sequence impedances and singe-phase sources/loads. For unbalanced multi-phase power flow analysis, OPEN offers the `Network_3ph` class.

The `PandapowerNet` class offers methods for balanced nonlinear power flow using a Newton–Raphson solution method, and balanced linear power flow based on the DC approximation. OPEN's `Network_3ph` class offers nonlinear multi-phase power flow using the Z-Bus method [52], as well as linear multi-phase power flow using the fixed-point linearisation from [46]. Wye and delta connected constant power loads/sources, constant impedance loads and capacitor banks can be modelled. Lines are modelled as π-equivalent circuits. Transformers with any combination of wye, wye-grounded or delta primary and secondary connections can also be modelled. Features that are planned to be added in future include voltage regulators and constant current loads.

Datasets associated with the initialisation methods of the `Network_3ph` class have been included with OPEN for a number of multi-phase networks, namely the IEEE 13 Node Test Feeder and IEEE European Low Voltage Test Feeder from [53], and 30 low voltage feeders from [54]. A range of balanced distribution and transmission network datasets are also available for the `PandapowerNet` class [55]. Like the `PandapowerNet` class, `Network_3ph` has been designed with its input and output variables organised into pandas dataframes, which provide a tabular data structure that can be easily accessed and modified [56]. The initialisation methods for the multi-phase networks included with OPEN have been written with line-by-line comments so they can be straightforwardly adapted for third-party network datasets.

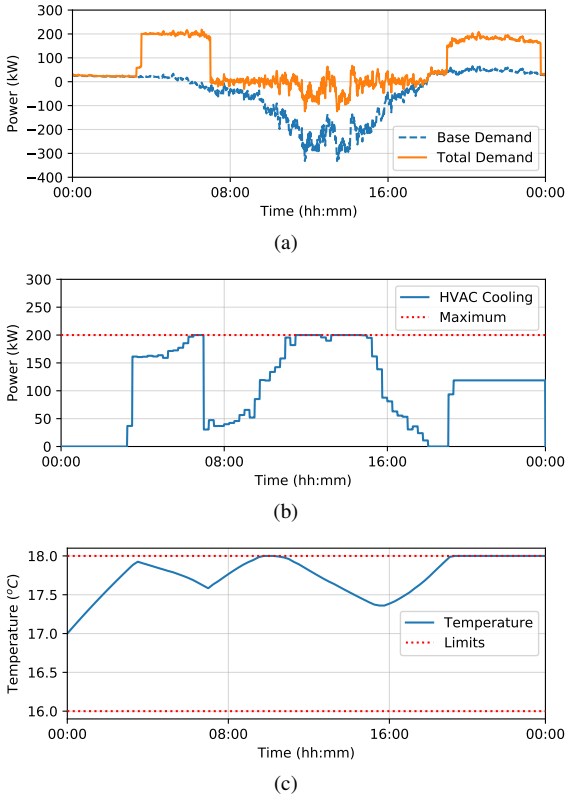

Figure 3: Summer building EMS case study. (a) The base demand (net of inflexible load and PV generation) and total demand (including HVAC). (b) The flexible HVAC output power used for cooling. (c) The internal temperature of the building.

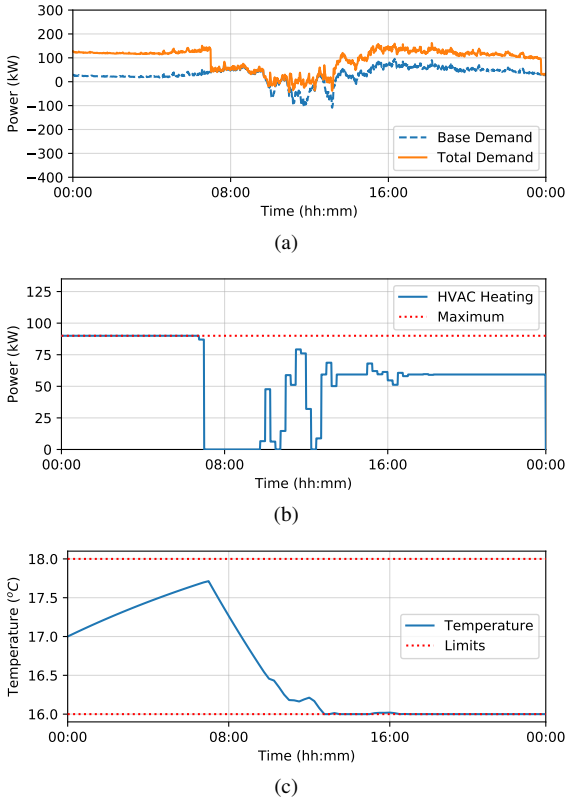

Figure 4: Winter building EMS case study. (a) The base demand (net of inflexible load and PV generation) and total demand (including HVAC). (b) The flexible HVAC output power used for heating. (c) The internal temperature of the building.

4. Case Studies

Two case study applications are included with OPEN to demonstrate its capabilities, and to provide templates to help users build their own applications. The first considers building energy management for a site with PV generation, flexible electric HVAC and time-of-use energy prices. The second considers smart EV charging within an unbalanced three-phase distribution network.

4.1. Building Energy Management Case Study

The building energy management case study focuses on a building with PV generation and a flexible HVAC unit which is controlled in order to minimise costs, with the constraint that the internal temperature remains between 16°C and 18°C.

4.1.1. Setup

The building operates under an Economy 7 tariff, which is a standard retail electricity supply option in the UK that charges a higher price for energy used between 7 am and 12 am, and a lower price for usage between 12 am and 7 am [57]. A lower feed-in price is received for excess generation. Summer and winter solar generation profiles with 1-minute resolution from the Customer-led Network Revolution trial are used [58]. The case study parameters are summarised in Table 2.

Table 2: Building energy management case study parameters.

Allowed temperature range	16 to 18°C
Initial temperature	17°C
Ambient temp. (Summer / Winter)	22°C / 10°C
PV generation capacity	400 kWp
HVAC cooling capacity	200 kW
HVAC heating capacity	90 kW
Cooling coefficient of performance	1
Heating coefficient of performance	3
Building heat transfer	0.0337°C/kW
Building thermal mass	500 kWh/°C
Energy price (7 am to 12 am)	£0.15/kWh
Energy price (12 am to 7 am)	£0.075/kWh
Feed-in price	£0.04/kWh
EMS time-series resolution	15 min.
Simulation time-series resolution	1 min.

The case study demonstrates the `BuildingAsset` class for modelling flexible HVAC, the `Market` class for modelling variable energy prices and open-loop optimal control within the `EnergySystem` class. The

Table 3: Electric vehicle smart charging case study parameters.

Number of electric vehicles	80
Electric vehicle battery sizes	36 kWh
Electric vehicle charger capacities	6.6 kW
PV generation capacities	200 kWp
Phase voltage magnitude limits	0.95 to 1.05 pu
Import price	£0.15/kWh
Export price	£0.05/kWh
Demand charge	£0.10/kW
EMS time-series resolution	30 min.
Simulation time-series resolution	5 min.

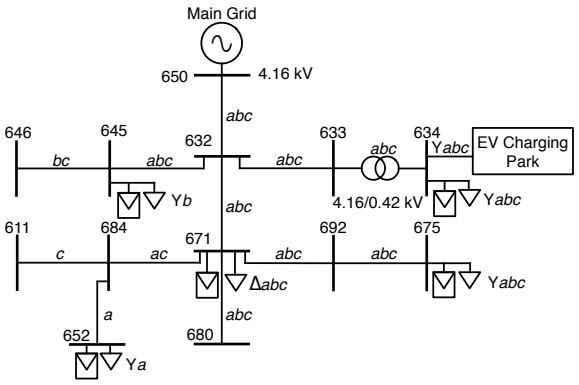

Figure 5: The IEEE 13 Node Test Feeder which has been adapted for the EV smart charging case study. The connected phases for each line, load and source are indicated (e.g. *abc* for a three-phase connection), as well as the connection configurations of the loads and sources (Y for wye and Δ for delta).

EMS is formulated with a 15-minute resolution time-series, and the simulation time-series has 1-minute resolution. Since network modelling is not the focus of this case study, a simple network with two buses has been implemented using the `PandapowerNet` class. The building is connected to the main grid via a 100 m cable and a 400 kVA 20 kV / 415 V transformer.

4.1.2. Results

Fig. 3 presents results for operation in summer, assuming an initial internal temperature of 17°C and an external ambient temperature of 22°C. Fig. 3a shows the base demand (the net of the inflexible building load and PV generation), as well as the total demand which also includes the flexible HVAC demand. As shown in Fig. 3b, the HVAC unit is used for cooling between 3 am and 7 am due to the lower energy price during this period. The HVAC consumption increases again during the middle of the day, since the low price received for energy exports incentivises local self-consumption of PV generation. As shown in Fig. 3c, the building temperature reaches but does not exceed the upper limit of 18°C.

Fig. 4 presents results for operation in winter, assuming the same initial internal temperature and an external ambient temperature of 10°C. As shown in Fig. 4a, the base demand is rarely negative, since there is significantly less solar generation in winter than summer. Fig. 4b shows that the HVAC unit is controlled to heat the building before 7 am when the import price of energy is lower. As shown in Fig. 4c, the building temperature reaches but does not fall below the lower limit of 16°C.

4.2. Electric Vehicle Smart Charging Case Study

This case study considers smart charging of EVs within an unbalanced three-phase distribution network.

4.2.1. Setup

The case study uses an adapted version of the IEEE 13 Node Test Feeder, shown in Fig. 5. Bus 634 has a business park where 80 EVs are charged at 6.6 kW controllable charge points. The objective is to charge all of the vehicles to their maximum energy level prior to departure, at lowest cost. This needs to be done without violating the phase voltage magnitude limits of 0.95 pu and 1.05 pu. The case study parameters are summarised in Table 3.

It is assumed that vehicle arrival and departure times are uniformly distributed between 6 am to 10 am, and 3 pm to 9 pm, respectively. The vehicles have 36 kWh batteries and arrive with uniformly distributed energy levels between 20% and 90%. The nonlinear relationship between charging power and efficiency is modelled using a stepwise approximation of empirical observations from charging a Nissan Leaf [59].

The network has five solar generation sources, which are each rated at 200 kWp. These are modelled as having the same generation profile (using a 24-hour profile form [58]) due to their proximity to one another. The price of energy imports is £0.15/kWh, and £0.05/kWh is paid for energy exports. In addition, there is a demand charge of £0.10/kW for the maximum power import over the day. Three-phase substation load profiles from the Customer Led Network Revolution trial are used to model the load at each bus of the distribution network [60].

The case study demonstrates the `StorageAsset` class for modelling EVs, the `Market` class for modelling demand charges and the `Network_3ph` class for multi-phase power flow. The case study was completed using open-loop optimisation and MPC within the `EnergySystem` class to compare the performance of these approaches. The EMS is formulated with a 30-minute resolution time-series, and the simulation time-series has 5-minute resolution. For the solar generation and loads, a smoothed version of the data (the 100-minute rolling average) is used as the day-ahead predicted generation and load for optimisation. Under MPC there is a receding horizon, so that a new optimisation problem is solved at each 30 minute sam-

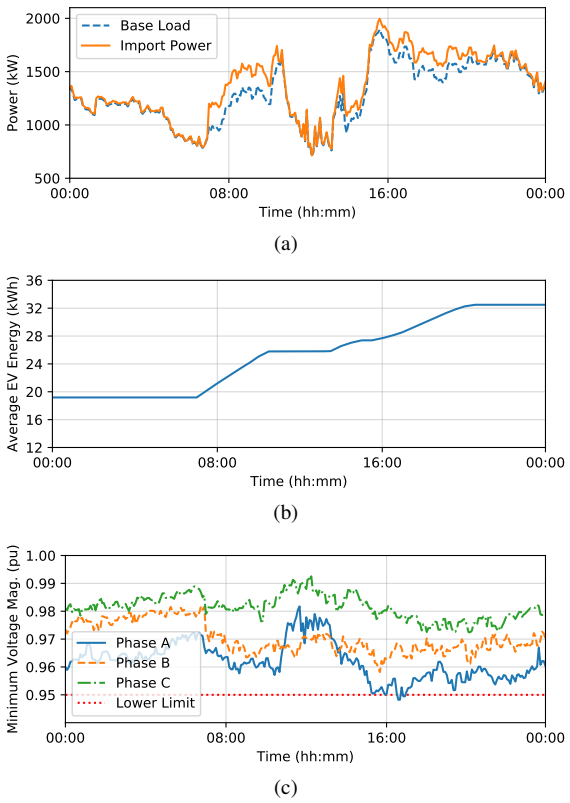

Figure 6: EV smart charging case study, under open-loop optimisation. (a) The base demand (net of inflexible load and PV generation) and total imported power (including EV charging). (b) The average energy level of the EVs. (c) The minimum voltage magnitude across the network for each phase.

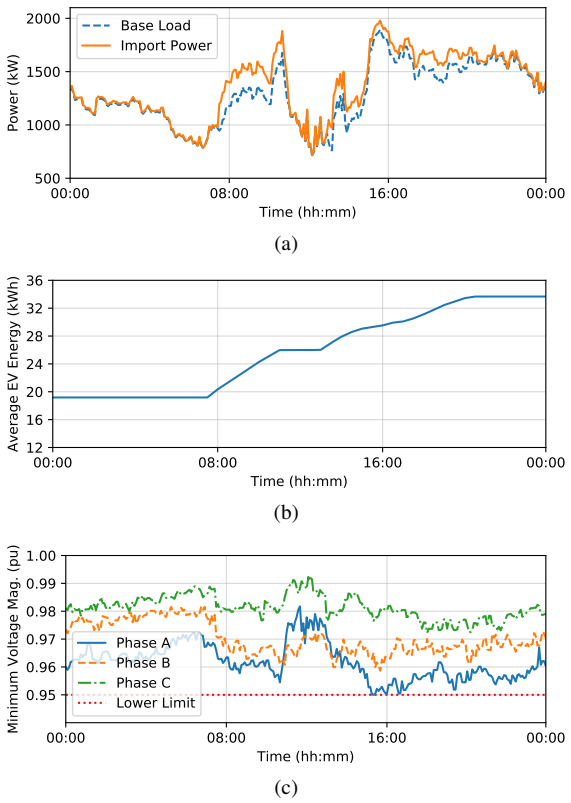

Figure 7: EV smart charging case study, under MPC. (a) The base demand (net of inflexible load and PV generation) and total imported power (including EV charging). (b) The average energy level of the EVs. (c) The minimum voltage magnitude across the network for each phase.

pling interval based on updated predictions of the load and renewable generation, as well as the current energy levels of the EVs.

4.2.2. Results

Fig. 6 shows the results for open-loop optimisation and Fig. 7 shows the results for MPC. Under both strategies, EV charging is scheduled to prevent an increase in the maximum demand, which would result in higher demand charges. Fig. 7a shows that the maximum demand under MPC is 1980 kW, which is slightly lower than the maximum demand of 1997 kW shown in Fig 6a under open-loop optimisation. Comparing Fig. 6b and Fig. 7b, it can be seen that the vehicles reach a higher average energy level under MPC (33.7 kWh) compared with open-loop optimisation (32.5 kWh). Note that the optimisation problem formulations treat battery efficiency as constant to obtain linear models. However, under MPC the EV charging power schedules are updated at each EMS sampling interval based on the current EV energy levels, which helps account for the nonlinear relationship between battery efficiency and charging power. Fig. 6c shows that under the open-loop optimisation the lower 0.95 pu phase voltage limit is violated by a small amount just after 4pm. Fig. 7c shows that there are no voltage violations under MPC, since the control reference updates at each EMS interval account for the current demand.

5. Conclusion

This paper has presented the Open Platform for Energy Networks (OPEN), an open-source Python platform for developing smart local energy system applications. OPEN addresses the need for integration between software tools for modelling, control and simulation, and combines important features for managing smart local energy systems, including multi-phase distribution network power flow, nonlinear energy storage modelling and receding horizon optimisation.

A key aim for OPEN is to enhance how academic research is done, and thereby increase the speed at which new methods are translated to industry application. Standardising interfaces between components (distributed energy resources, networks and markets) and between tools (for modelling, control and simulation) is important for helping teams retain knowledge and leverage previous work from project to project. Predefined interfaces between components and tools

make it easier for laboratories specialising in different areas to collaborate effectively. By providing modular smart local energy system case studies, OPEN allows users to focus on the novel aspects of their particular application.

Since it is open source, users can extend OPEN by implementing new inheriting classes, creating interfaces to other open-source tools and expanding its library of distributed energy resource models, networks and market arrangements. Areas of particular interest for future development include multi-energy vector modelling, network investment planning and DC microgrid modelling.

Appendix A. Nonlinear Energy Storage Modelling

Energy storage systems can be modelled using the `StorageAsset` class. Within the EMS methods of the `EnergySystem` class, a linear energy storage model is used to maintain a convex optimisation problem formulation which can be solved in polynomial time by standard solvers [61]. The discrete time linear energy storage model is given by [62]

$$E_{it+1} = \begin{cases} E_{it} - \frac{1}{\eta_i}\Delta t p_{it}^{dis}, & \text{for } p_{it}^{dis} \geq 0, p_{it}^{ch} = 0 \\ E_{it} + \eta_i \Delta t p_{it}^{ch}, & \text{for } p_{it}^{ch} \geq 0, p_{it}^{dis} = 0 \end{cases} \quad \text{(A.1)}$$

For storage system i and time interval t, E_{it} is the energy level, p_{it}^{ch} is the charging power, p_{it}^{dis} is the discharging power and Δt is the time series interval duration. The model is based on the approximation of constant efficiency $\eta_i \in (0, 1]$ over the range of output powers. Constraints are imposed on the maximum charging and discharging power, and on the maximum and minimum energy level. `StorageAsset` also includes a coefficient $c_i^{deg} \geq 0$ specifying the cost of degradation associated with energy throughput, which is used as a cost term by the EMS optimisation methods.

The `StorageAsset.update_control()` method has been implemented to allow nonlinear relationships between efficiency and output power to be modelled during simulations. This is particularly relevant for battery energy storage systems, since battery and converter efficiency are output power dependent [59]. The nonlinear relationship between output power and efficiency is modelled with stepwise approximations. Practically, this is implemented with a 100-element vector which specifies efficiencies over the range of possible output powers (e.g. the 70th element specifies the efficiency when the output power is between 69% and 70% of the maximum output power).

`StorageAsset` has been implemented in a generic manner so that it can be used to model a variety of energy storage technologies. The object-oriented structure of OPEN allows more advanced technology-specific storage system efficiency and degradation models to be implemented by creating new subclasses which inherit from `StorageAsset`.

Appendix B. Building Thermal Modelling

Buildings with flexible HVAC can be modelled using the `BuildingAsset` class. The thermal characteristics of a building are modelled by a first order discrete time temperature model (analogous to an RC electrical circuit) [63]

$$\tau_{jt+1} = \tau_{jt} + \frac{\Delta t}{R_j C_j}(\tau_{jt}^a - \tau_{jt}) + \frac{\Delta t}{C_j}(\eta_j^{he} p_{jt}^{he} - \eta_j^{co} p_{jt}^{co}). \quad \text{(B.1)}$$

For building j and time interval t, τ_{jt} is the internal building temperature, τ_{jt}^a is the ambient temperature, p_{jt}^{he} is the forced heating power and p_{jt}^{co} is the forced cooling power. R_j and C_j are constants which respectively model the heat transfer and thermal mass of the building. η_j^{he} and η_j^{co} are the coefficients of performance for heating and cooling.

Appendix C. Multi-Phase Power Flow Modelling and Validation

OPEN's `Network_3ph` class allows three-phase unbalanced power systems to be accurately modelled. The `Network_3ph.update_YandZ()` method updates the admittance matrix of the network based on pandas dataframes specifying the characteristics of lines, transformers and capacitor banks. Consider a three-phase network with phases $\{a, b, c\}$, slack bus voltage $v_0/|v_0| = (1, e^{-j\frac{2\pi}{3}}, e^{j\frac{2\pi}{3}})$ and N load buses. $Y \in \mathbb{C}^{3(N+1)\times 3(N+1)}$ is the three-phase admittance matrix of the network, which can be partitioned into $Y = \begin{bmatrix} Y_{00} & Y_{0N} \\ Y_{N0} & Y_{NN} \end{bmatrix}$, where $Y_{00} \in \mathbb{C}^{3\times 3}$ and $Y_{NN} \in \mathbb{C}^{3N\times 3N}$.

`Network_3ph.zbus_pf()` implements the Z-Bus nonlinear power flow method to obtain complex phase voltages and line currents. This involves iteratively applying the following fixed-point equation for a user-specified number of iterations [46]

$$\begin{aligned} v_{[k+1]} &= Y_{NN}^{-1}(\text{diag}(v_{[k]})^{-1} s^Y \\ &\quad + H^\top \text{diag}(H v_{[k]})^{-1} s^\Delta) - Y_{NN}^{-1} Y_{N0} v_0, \quad \text{(C.1)} \\ H &= \text{blockdiag}\left(\begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix}\right). \end{aligned}$$

$v_{[k]} = (v_{1[k]}, \ldots, v_{N[k]})$ is the vector of complex phase voltages calculated at iteration k, where $v_{i[k]} = (v_{i[k]}^a, v_{i[k]}^b, v_{i[k]}^c)$. $s^Y = (s_1^Y, \ldots, s_N^Y)$ and $s^\Delta = (s_1^\Delta, \ldots, s_N^\Delta)$ are the vectors of wye and delta connected complex power injections on each bus and phase, where $s_i^Y = (s_i^a, s_i^b, s_i^c)$ and $s^\Delta = (s_i^{ab}, s_i^{bc}, s_i^{ca})$.

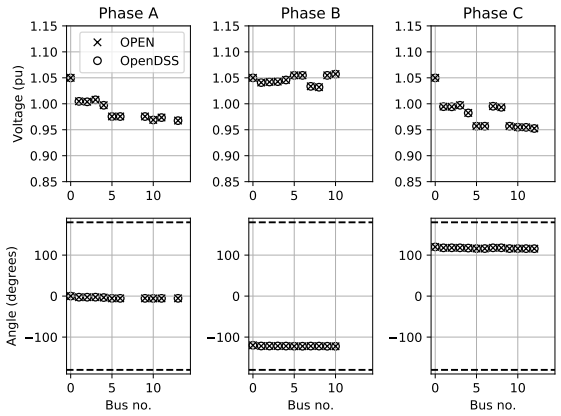

Figure C.8: Voltage magnitude and angle for each bus and phase obtained for the IEEE 13 Node Test Feeder using OPEN and OpenDSS.

To validate the accuracy of the Z-Bus method used by `Network_3ph`, power flow simulations have been completed for each of the multi-phase networks included within OPEN (the IEEE 13 Node Test Feeder, IEEE European Low Voltage Test Feeder [53], and 30 low voltage feeders from [54]). The solutions given by OPEN have been compared with the solutions from OpenDSS [35], a widely used power network simulation tool which offers nonlinear multi-phase power flow. The relative voltage error ϵ_v is calculated as

$$\epsilon_v = \frac{\|v_{\text{OPEN}} - v_{\text{OpenDSS}}\|_2}{\|v_{\text{OpenDSS}}\|_2}, \tag{C.2}$$

where v_{OPEN} and v_{OpenDSS} are the vectors of complex voltages calculated for each phase across the network using OPEN and OpenDSS respectively.

Fig. C.8 shows the voltage magnitude and angle for each bus and phase obtained for the IEEE 13 Node Test Feeder using OPEN and OpenDSS. In this case, the relative voltage error ϵ_v is 7.54×10^{-6}. Across the multi-phase networks which were tested ϵ_v was found to be less than 3.3×10^{-5} in all cases.

Appendix D. Energy Management System Optimisation

The `EnergySystem` class has EMS methods for open-loop optimisation and MPC. These methods generate references for controllable `Asset` objects for each interval of the EMS time-series. The EMS time-series has T^{ems} intervals, each of duration Δt^{ems}. There is a separate simulation time-series with T intervals, each of duration Δt. The time-series should be defined so that they have the same duration ($T\Delta t = T^{ems}\Delta t^{ems}$), and so that the EMS schedules assets at the same timescale, or slower, than the simulation ($\Delta t \leq \Delta t^{ems}$). The EMS methods which are included with OPEN use the open-source Python package PICOS for optimisation. PICOS provides a high-level portable interface for a range of conic and integer programming solvers [64].

Consider a smart local energy system with a set of energy storage systems $\mathcal{S}$, buildings with flexible HVAC $\mathcal{B}$, a three-phase network with slack bus 0, load buses $\mathcal{N} = \{1, \ldots, N\}$ and lines $\mathcal{L}$, and an upstream market at the network's slack bus. Combining the linear energy storage system model (A.1), first order building heat model (B.1) and linear three-phase power flow model from [46], the multi-period smart local energy system optimisation can be formulated as a linear program.

$$\begin{aligned}
\min\ & \bar{\lambda}\bar{p}_0 + \textstyle\sum_{t\in\mathcal{T}} \Delta t^{ems}(\tilde{\lambda}_t^{imp} p_{0t}^{imp} - \tilde{\lambda}_t^{exp} p_{0t}^{exp} \\
& + \textstyle\sum_{i\in\mathcal{S}} c_i^{deg}(p_{it}^{dis} + p_{it}^{ch})) && \text{(D.1a)}\\
\text{s.t. } & 0 \leq p_{it}^{ch} \leq \bar{p}_{it}^{ch},\ 0 \leq p_{it}^{dis} \leq \bar{p}_{it}^{dis}, && \text{(D.1b)}\\
& E_{it+1} = E_{it} + \Delta t^{ems}(\eta_i p_{it}^{ch} - \tfrac{1}{\eta_i} p_{it}^{dis}), && \text{(D.1c)}\\
& \underline{E}_{it} \leq E_{it} \leq \bar{E}_{it}, i \in \mathcal{S} && \text{(D.1d)}\\
& 0 \leq p_{jt}^{he} \leq \bar{p}_{jt}^{he}, 0 \leq p_{jt}^{co} \leq \bar{p}_{jt}^{co} && \text{(D.1e)}\\
& \tau_{jt+1} = \tau_{jt} + \tfrac{\Delta t^{ems}}{R_j C_j}(\tilde{\tau}_{jt}^{a} - \tau_{jt}) \\
& \qquad + \tfrac{\Delta t^{ems}}{C_j}(\eta_j^{he} p_{jt}^{he} - \eta_j^{co} p_{jt}^{co}), && \text{(D.1f)}\\
& \underline{\tau}_{jt} \leq \tau_{jt} \leq \bar{\tau}_{jt}, j \in \mathcal{B}, && \text{(D.1g)}\\
& \Delta p_{lt}^{Y} = \textstyle\sum_{i\in\mathcal{S}} m_{li}^{Y\mathcal{S}}(p_{it}^{dis} - p_{it}^{ch}) \\
& \qquad - \textstyle\sum_{j\in\mathcal{B}} m_{lj}^{Y\mathcal{B}}(p_{jt}^{he} + p_{jt}^{co}), l \in \mathcal{N} && \text{(D.1h)}\\
& \Delta p_{lt}^{\Delta} = \textstyle\sum_{i\in\mathcal{S}} m_{li}^{\Delta\mathcal{S}}(p_{it}^{dis} - p_{it}^{ch}) \\
& \qquad - \textstyle\sum_{j\in\mathcal{B}} m_{lj}^{\Delta\mathcal{B}}(p_{jt}^{he} - p_{jt}^{co}), l \in \mathcal{N} && \text{(D.1i)}\\
& |\underline{v}| \leq K_t^{Y}\Delta p_t^{Y} + K_t^{\Delta}\Delta p_t^{\Delta} + |\tilde{v}_t| \leq |\bar{v}|, && \text{(D.1j)}\\
& J_{lmt}^{Y} p_t^{Y} + J_{lmt}^{\Delta}\Delta p_t^{\Delta} + |\tilde{i}_{lmt}| \leq |\bar{i}_{lm}|, (l,m) \in \mathcal{L} && \text{(D.1k)}\\
& p_{0t}^{exp} - p_{0t}^{imp} = G_t^{Y}\Delta p_t^{Y} + G_t^{\Delta}\Delta p_t^{\Delta} - \tilde{p}_{0t}, && \text{(D.1l)}\\
& 0 \leq p_{0t}^{imp} \leq \bar{p}_{0t}^{imp}, 0 \leq p_{0t}^{exp} \leq \bar{p}_{0t}^{exp}, && \text{(D.1m)}\\
& p_{0t}^{imp} - p_{0t}^{exp} \leq \bar{p}_0 + \bar{p}_0^{pre}, 0 \leq \bar{p}_0. && \text{(D.1n)}
\end{aligned}$$

$\mathcal{T} = \{t_0^{ems}, \ldots, t_0^{ems} + T^{ems} - 1\}$ is the optimisation time horizon. The objective (D.1a) is to minimise the combined cost associated with importing/exporting energy upstream, demand charges and degradation. $\bar{\lambda}$ is the demand charge price and $\bar{p}_0$ is the maximum power imported over the day at the slack bus. For time interval $t \in \mathcal{T}$, $p_{t0}^{imp}, p_{t0}^{exp}$ are the import and export powers, and $\tilde{\lambda}_{t0}^{imp}, \tilde{\lambda}_{t0}^{exp}$ are the predicted prices of energy imports and exports, where $\tilde{\lambda}_{t0}^{imp} \geq \tilde{\lambda}_{t0}^{exp}$.

The energy storage system constraints are given by (D.1b)–(D.1d). For energy storage system $i \in \mathcal{S}$ and interval $t \in \mathcal{T}$, $\bar{p}_{it}^{ch}, \bar{p}_{it}^{dis}$ are the maximum charging and discharging powers, and $\underline{E}_{it}, \bar{E}_{it}$ are the minimum and maximum energy levels. These are time dependent (and stored as vectors in OPEN), allowing a variety of constraints to be modelled (e.g. electric vehicles which are only available for charging at certain times). The

initial energy levels $E_{it_0^{ems}}$ are inputs of the optimisation.

The building constraints are given by (D.1e)–(D.1g). For building $j \in \mathcal{B}$ and interval $t \in \mathcal{T}$, $\bar{p}^{he}_{jt}, \bar{p}^{co}_{jt}$ are the maximum HVAC powers for heating and cooling, and $\underline{\tau}_{jt}, \bar{\tau}_{jt}$ are the minimum and maximum internal temperatures. The initial internal temperatures $\tau_{jt_0^{ems}}$ and the predicted ambient temperature profiles $\tilde{\tau}^a_{jt}$ are inputs of the optimisation.

The network constraints are given by (D.1h)–(D.1n). To formulate the optimisation problem, linear network models are obtained for each interval $t \in \mathcal{T}$, based on nominal complex wye and delta power injection vectors $\tilde{s}^Y_t, \tilde{s}^\Delta_t \in \mathbb{C}^{3N}$. These include the predicted real and reactive power injections from inflexible assets throughout the network. From the fixed-point linearisation in [46], coefficient matrices can be obtained for the phase voltage magnitudes K^Y_t, K^Δ_t, line current magnitudes $J^Y_{lmt}, J^\Delta_{lmt}$, and net export power G^Y_t, G^Δ_t. Constraints (D.1h) and (D.1i) link the energy storage system and building output powers to the network power injections, based on the vectors $m^{Y\mathcal{S}}_{li}, m^{\Delta\mathcal{S}}_{li}, m^{Y\mathcal{B}}_{lj}, m^{\Delta\mathcal{B}}_{lj} \in \mathbb{R}^3$ which depend on the network locations and connection configurations (single/three-phase and wye/delta) of the flexible assets. $\bar{p}^{pre}_0$ is the maximum demand which occured prior to t^{ems}_0, and is therefore locked-in from the perspective of demand charges. Note for the copper plate optimisation methods, constraints (D.1h)–(D.1l) are replaced by a single total power balance constraint for each time interval $t \in \mathcal{T}$,

$$p^{exp}_{0t} - p^{imp}_{0t} = \textstyle\sum_{i\in\mathcal{S}}(p^{dis}_{it} - p^{ch}_{it}) - \sum_{j\in\mathcal{B}}(p^{he}_{jt} + p^{co}_{jt}) - \tilde{p}_{0t}. \tag{D.2}$$

For the open-loop EMS methods, the optimisation problem (D.1) is solved once with $t^{ems}_0 = 0$, based on the predicted price, load and generation profiles, and the initial flexible asset state variables (storage system energy levels and building internal temperatures). This generates control references for the flexible assets over the full time horizon, which are then applied to the assets during simulation.

For the MPC EMS methods, an optimisation problem is solved at each step of the time horizon, based on up-to-date information. At each time interval t^{ems}_0:

1. For the current interval t^{ems}_0, the flexible asset state variables ($E_{t^{ems}_0}, i \in \mathcal{S}$ and $\tau_{jt^{ems}_0}, j \in \mathcal{B}$) are updated. For $t \in \mathcal{T}$, the predicted energy import and export prices $\tilde{\lambda}^{imp}_t, \tilde{\lambda}^{exp}_t$, nominal power injections $\tilde{s}^Y_t, \tilde{s}^\Delta_t$ and building ambient temperatures $\tilde{\tau}^a_{jt}, j \in \mathcal{B}$ are also updated.
2. The optimisation problem (D.1) is solved.
3. Each flexible asset implements its control references obtained for the current interval t^{ems}_0 ($p^{ch*}_{it^{ems}_0}, p^{dis*}_{it^{ems}_0}, i \in \mathcal{S}$ and $p^{he*}_{jt^{ems}_0}, p^{co*}_{jt^{ems}_0}, j \in \mathcal{B}$) during the associated simulation time-steps $t^{sim} \in \{\frac{\Delta t^{ems}}{\Delta t} t^{ems}_0, \frac{\Delta t^{ems}}{\Delta t} t^{ems}_0 + 1, \ldots, \frac{\Delta t^{ems}}{\Delta t}(t^{ems}_0 + 1) - 1\}$. Also, the locked-in maximum demand is updated, $\bar{p}^{pre}_0 \leftarrow \max\{\bar{p}^{pre}_0, \bar{p}^*_0\}$
4. The optimisation time horizon recedes by a step $t^{ems}_0 \leftarrow t^{ems}_0 + 1$ and is shortened by an interval $T^{ems} \leftarrow T^{ems} - 1$. Done when $T^{ems} = 0$.

Acknowledgements

The authors would like to acknowledge the Engineering and Physical Sciences Research Council (project references EP/S000887/1 and EP/S031901/1) and Innovate UK (project references 104229 and 104781) for supporting this work.

References

[1] D. Pudjianto, Chin Kim Gan, V. Stanojevic, M. Aunedi, P. Djapic, G. Strbac, Value of Integrating Distributed Energy Resources in the UK electricity system, in: IEEE PES General Meeting, 2010, pp. 1–6.

[2] D. Pudjianto, C. Ramsay, G. Strbac, Virtual Power Plant and System Integration of Distributed Energy Resources, Renewable Power Generation, IET 1 (1) (2007) 10–16.

[3] R. Lasseter, MicroGrids, in: IEEE Power Engineering Society Winter Meeting, Vol. 1, 2002, pp. 305–308.

[4] R. Ford, C. Maidment, M. Fell, C. Vigurs, M. Morris, A framework for understanding and conceptualising smart local energy systems, EnergyREV, University of Strathclyde Publishing, Strathclyde, UK, 2019.

[5] L. F. Ochoa, C. J. Dent, G. P. Harrison, Distribution Network Capacity Assessment: Variable DG and Active Networks, IEEE Transactions on Power Systems 25 (1) (2010) 87–95.

[6] T. Morstyn, B. Hredzak, V. G. Agelidis, Control Strategies for Microgrids With Distributed Energy Storage Systems: An Overview, IEEE Transactions on Smart Grid 9 (4) (2018) 3652–3666.

[7] F. Moret, P. Pinson, Energy Collectives: a Community and Fairness Based Approach to Future Electricity Markets, IEEE Transactions on Power Systems.

[8] F. Kienzle, P. Ahčin, G. Andersson, Valuing Investments in Multi-Energy Conversion, Storage, and Demand-Side Management Systems Under Uncertainty, IEEE Transactions on Sustainable Energy.

[9] T. Morstyn, A. Teytelboym, M. D. Mcculloch, Bilateral Contract Networks for Peer-to-Peer Energy Trading, IEEE Transactions on Smart Grid 10 (2) (2019) 2026–2035.

[10] T. Morstyn, A. Teytelboym, M. D. McCulloch, Designing Decentralized Markets for Distribution System Flexibility, IEEE Transactions on Power Systems 34 (3) (2019) 2128–2139.

[11] D. Pudjianto, C. Ramsay, G. Strbac, Microgrids and Virtual Power Plants: Concepts to Support the Integration of Distributed Energy Resources, Proceedings of the Institution of Mechanical Engineers Part a-Journal of Power and Energy 222 (A7) (2008) 731–741.

[12] T. Morstyn, N. Farrell, S. J. Darby, M. D. McCulloch, Using Peer-to-Peer Energy-Trading Platforms to Incentivize Prosumers to Form Federated Power Plants, Nature Energy 3 (2) (2018) 94–101.

[13] Prospering From the Energy Revolution, accessed Sep. 2019. URL ukri.org/innovation/industrial-strategy-challenge-fund/prospering-from-the-energy-revolution

[14] EN SGplusRegSys, accessed Sep. 2019. URL cordis.europa.eu/project/rcn/212898/factsheet/en

[15] IO.Energy, accessed Sep. 2019. URL ioenergy.eu

[16] E. Mengelkamp, J. Gärttner, K. Rock, S. Kessler, L. Orsini, C. Weinhardt, Designing Microgrid Energy Markets. A Case Study: The Brooklyn Microgrid, Applied Energy 210 (2017) 870–880.
[17] Decentralised Energy Exchange (deX) Program, accessed Sep. 2019. URL arena.gov.au/projects/decentralised-energy-exchange
[18] Oxford Energy and Power Group – Current Research Projects, accessed May 2020. URL epg.eng.ox.ac.uk/our-research
[19] Vehicle-to-Grid Oxford (V2GO), accessed May 2020. URL v2go.org
[20] Multi- Sites, Actors, Vectors, Energy Services (Multi-SAVES), accessed May 2020. URL gtr.ukri.org/projects?ref=133702
[21] Power, Energy, Technology, Efficiency (PETE), accessed May 2020. URL peteproject.com
[22] Local Energy Oxfordshire (LEO), accessed May 2020. URL project-leo.co.uk
[23] J. M. Reniers, G. Mulder, S. Ober-Blöbaum, D. A. Howey, Improving Optimal Control of Grid-Connected Lithium-Ion Batteries Through More Accurate Battery and Degradation Modelling, Journal of Power Sources 379 (2018) 91–102.
[24] N. Holjevac, T. Capuder, I. Kuzle, N. Zhang, C. Kang, Modelling Aspects of Flexible Multi-Energy Microgrids, in: Power Systems Computation Conference (PSCC), 2018, pp. 1–7.
[25] C. Crozier, M. Deakin, T. Morstyn, M. D. McCulloch, Incorporating Charger Efficiency into Electric Vehicle Charging Optimisation, in: IEEE Innovative Smart Grid Technologies Europe Conference (ISGT), 2019.
[26] S. S. Guggilam, E. Dall'Anese, Y. C. Chen, S. V. Dhople, G. B. Giannakis, Scalable Optimization Methods for Distribution Networks with High PV Integration, IEEE Transactions on Smart Grid 7 (4) (2016) 2061–2070.
[27] P. Wolfs, G. S. Reddy, A Receding Predictive Horizon Approach to the Periodic Optimization of Community Battery Energy Storage Systems, Australasian Universities Power Engineering Conference (AUPEC) (2012) 1–6.
[28] Open Platform for Energy Networks (OPEN) Code, accessed May 2020. URL github.com/EPGOxford/OPEN
[29] Open Platform for Energy Networks (OPEN) Documentation, accessed May 2020. URL open-platform-for-energy-networks.readthedocs.io
[30] L. Thurner, A. Scheidler, F. Schafer, J.-H. Menke, J. Dollichon, F. Meier, et al., Pandapower–An Open-Source Python Tool for Convenient Modeling, Analysis, and Optimization of Electric Power Systems, IEEE Transactions on Power Systems 33 (6) (2018) 6510–6521.
[31] R. D. Zimmerman, C. E. Murillo-Sanchez, R. J. Thomas, MATPOWER: Steady-State Operations, Planning, and Analysis Tools for Power Systems Research and Education, IEEE Transactions on Power Systems 26 (1) (2011) 12–19.
[32] R. Pedersen, C. Sloth, G. B. Andresen, R. Wisniewski, DiSC: A simulation framework for distribution system voltage control, European Control Conference (ECC) (2015) 1056–1063.
[33] F. M. Gonzalez-Longatt, J. Luis Rueda (Eds.), PowerFactory Applications for Power System Analysis, Power Systems, Springer International Publishing, Cham, 2014.
[34] D. P. Chassin, J. C. Fuller, N. Djilali, GridLAB-D: An Agent-Based Simulation Framework for Smart Grids, Journal of Applied Mathematics (2014) 1–12.
[35] R. C. Dugan, T. E. McDermott, An Open Source Platform for Collaborating on Smart Grid Research, IEEE Power and Energy Society General Meeting (2011) 1–7.
[36] S. Pfenninger, B. Pickering, Calliope: a Multi-Scale Energy Systems Modelling Framework, Journal of Open Source Software 3 (29) (2018) 825.
[37] S. Hilpert, C. Kaldemeyer, U. Krien, S. Günther, C. Wingenbach, G. Plessmann, The Open Energy Modelling Framework (oemof) - A New Approach to Facilitate Open Science in Energy System Modelling, Energy Strategy Reviews 22 (0) (2018) 16–25.
[38] M. Howells, H. Rogner, N. Strachan, C. Heaps, H. Huntington, S. Kypreos, et al., OSeMOSYS: The Open Source Energy Modeling System. An Introduction to its Ethos, Structure and Development., Energy Policy 39 (10) (2011) 5850–5870.
[39] urbs, accessed Sep. 2019. URL urbs.readthedocs.io
[40] PLEXOS, accessed Sep. 2019. URL energyexemplar.com
[41] T. Brown, J. Hörsch, D. Schlachtberger, PyPSA: Python for Power System Analysis, Journal of Open Research Software 6.
[42] R. Loulou, G. Goldstein, A. Kanudia, A. Lettila, U. Remme, Documentation for the TIMES Model: Part 1, IEA Energy Technology Systems Analysis Programme (2005) 1–78.
[43] J. Johnston, R. Henriquez-Auba, B. Maluenda, M. Fripp, Switch 2.0: A modern platform for planning high-renewable power systems, SoftwareX 10 (2019) 1–32.
[44] H. G. Svendsen, O. C. Spro, PowerGAMA: A new simplified modelling approach for analyses of large interconnected power systems, applied to a 2030 Western Mediterranean case study, Journal of Renewable and Sustainable Energy 8 (5) (2016) 055501.
[45] H. K. Ringkjøb, P. M. Haugan, I. M. Solbrekke, A Review of Modelling Tools for Energy and Electricity Systems with Large Shares of Variable Renewables, Renewable and Sustainable Energy Reviews 96 (2018) 440–459.
[46] A. Bernstein, C. Wang, E. Dall'Anese, J.-Y. Le Boudec, C. Zhao, Load Flow in Multiphase Distribution Networks: Existence, Uniqueness, Non-Singularity and Linear Models, IEEE Transactions on Power Systems 33 (6) (2018) 5832–5843.
[47] T. Morstyn, B. Hredzak, R. P. Aguilera, V. G. Agelidis, Model Predictive Control for Distributed Microgrid Battery Energy Storage Systems, IEEE Transactions on Control Systems Technology 26 (3) (2018) 1107–1114.
[48] J. Neubauer, M. Simpson, Deployment of Behind-The-Meter Energy Storage for Demand Charge Reduction, National Renewable Energy Laboratory (2015) 1–30. URL nrel.gov/docs/fy15osti/63162.pdf
[49] G. Van Rossum, F. L. Drake Jr, Python Tutorial, Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995.
[50] L. Che, X. Liu, X. Zhu, Y. Wen, Z. Li, Intra-Interval Security Based Dispatch for Power Systems with High Wind Penetration, IEEE Transactions on Power Systems 34 (2) (2019) 1243–1255.
[51] J. D. Hunter, Matplotlib: A 2d graphics environment, Computing in Science & Engineering 9 (3) (2007) 90–95.
[52] M. Bazrafshan, N. Gatsis, Comprehensive Modeling of Three-Phase Distribution Systems via the Bus Admittance Matrix, IEEE Transactions on Power Systems 33 (2) (2018) 2015–2029.
[53] K. Schneider, B. Mather, B. Pal, C.-W. Ten, G. Shirek, H. Zhu, et al., Analytic Considerations and Design Basis for the IEEE Distribution Test Feeders, IEEE Transactions on Power Systems 33 (3) (2017) 3181–3188.
[54] A. N. Espinosa, Low Voltage Networks Models and Low Carbon Technology Profiles, ENWL LV Network Solutions Dissemination Document.
[55] Pandapower – Power System Test Cases, accessed Sep. 2019. URL pandapower.readthedocs.io/en/v1.4.1/networks.html
[56] W. McKinney, Data Structures for Statistical Computing in Python, in: S. van der Walt, J. Millman (Eds.), Proceedings of the 9th Python in Science Conference, 2010, pp. 51 – 56.
[57] A Complete Guide to Economy 7 and How it Works, accessed May 2020. URL uswitch.com/gas-electricity/guides/economy-7
[58] Customer-Led Network Revolution, Dataset: Enhanced Profiling of Domestic Customers With Solar Photovoltaics, accessed May 2020 (2014). URL networkrevolution.co.uk
[59] A. Thingvad, C. Ziras, J. Hu, M. Marinelli, Assessing the Energy Content of System Frequency and Electric Vehicle Charging Efficiency for Ancillary Service Provision, in: International Universities Power Engineering Conference (UPEC), 2017, pp. 1–6.

[60] Customer-Led Network Revolution, Dataset: Urban Distribution Substation Transformer, Thermal Dataset, accessed May 2020 (2014). URL networkrevolution.co.uk
[61] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, Cambridge, 2004.
[62] P. Fortenbacher, J. L. Mathieu, G. Andersson, Modeling and Optimal Operation of Distributed Battery Storage in Low Voltage Grids, IEEE Transactions on Power Systems 32 (6) (2017) 4340–4350.
[63] E. McKenna, M. Thomson, High-Resolution Stochastic Integrated Thermal-Electrical Domestic Demand Model, Applied Energy 165 (2016) 445–461.
[64] G. Sagnol, M. Stahlberg, PICOS, accessed Sep. 2019. URL picos-api.gitlab.io/picos