Flexibility Aggregation of Temporally Coupled Resources in Real Time Balancing Markets Using Machine Learning

Georgios Tsaousoglou, Ippokratis Sartzetakis, Prodromos Makris, Nikolaos Efthymiopoulos, Emmanouel Varvarigos, Nikolaos G. Paterakis

***Abstract*— In modern power systems with high penetration of renewable energy sources, the flexibility provided by distributed energy resources is becoming invaluable. Demand aggregators offer balancing energy in the real-time balancing market on behalf of flexible resources. A challenging task is the design of the offering strategy of an aggregator. In particular, it is difficult to capture the flexibility cost of a portfolio of flexibility assets within a price-quantity offer, since the costs and constraints of flexibility resources exhibit inter-temporal dependencies. In this paper, we propose a generic method for constructing aggregated balancing energy offers that best represent the portfolio's actual flexibility costs, while accounting for uncertainty in future timeslots. For the case study presented, we use offline simulations to train and compare different machine learning algorithms that receive the information about the state of the flexible resources and calculate the aggregator's offer. Once trained, the machine learning algorithms can make fast decisions about the portfolio's balancing energy offer in the real-time balancing market. Our simulations show that the proposed method performs reliably towards capturing the flexibility of the Aggregator's portfolio and minimizing the aggregator's imbalances.**

***Index Terms*— flexibility, distributed energy resources, aggregator, balancing market, machine learning**

NOMENCLATURE

Sets

T	Set of timeslots.
N	Set of DER agents.
N_{EV}	Set of Electric Vehicles.
N_{TCL}	Set of Thermostatically Controlled Loads.
F_n	Set of local constraints of DER n.
H_n	Set of timeslots for which DER n is operating.
S	Set of possible scenarios for dispatch orders.

Indices

t	A timeslot.
τ	The current operating timeslot.
n	A DER.
i	An Electric Vehicle.
j	A Thermostatically Controlled Load.

Parameters

P_N^t	Aggregated energy demand in period t [kWh].
R^t	RES generation in period t [kWh].
$\mathrm{B}_{\mathrm{up}}^t$	Quantity for balancing energy up, in t $\left[\frac{\$}{\mathrm{kWh}}\right]$.
$\mathrm{B}_{\mathrm{down}}^t$	Quantity for balancing energy down, in t $\left[\frac{\$}{\mathrm{kWh}}\right]$.
arr_n	Arrival time of DER n.
dep_n	Departure time of DER n.
$\mathrm{x}_n^{\min}$	Minimum operating energy of DER n [kWh].
$\mathrm{x}_n^{\max}$	Maximum operating energy of DER n [kWh].
w_i	Elasticity parameter of EV i $\left[\frac{\$}{(\mathrm{kWh})^2}\right]$.
w_j	Elasticity parameter of TCL j $\left[\frac{\$}{(\mathrm{kWh})^2}\right]$.
tol_n	Tolerance parameter of DER n.
θ_{env}^t	Environment temperature at period t [F].
$\theta_{\mathrm{sp},j}^t$	Temperature setpoint of TCL j at period t [F].
con_j	Energy conversion factor of TCL j.
ins_j	Temperature increase factor of TCL j.
h_i	Charging efficiency of EV i.
E_i	Required charging energy of EV i [kWh].
$\lambda_{(\cdot)}^t$	Price for balancing energy at period t $\left[\frac{\$}{\mathrm{kWh}}\right]$.
λ_{Imb}	Imbalance price.
$\lambda_{\max}$	Upper bound on the balancing energy price.
$\lambda_{\min}$	Lower bound on the balancing energy price.

Variables

x_n^t	Energy consumption decision of DER n in t $[kWh]$.
X_N^t	Aggregated energy consumption in period t $[kWh]$.
b_{up}^t	Price offer for balancing energy up in period t $\left[\frac{\$}{kWh}\right]$.
b_{down}^t	Price offer for balancing energy down in t $\left[\frac{\$}{kWh}\right]$.
θ_j^t	Temperature of TCL j at period t [F].

Georgios Tsaousoglou received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No.754462. N. Efthymiopoulos, P. Makris, and E. Varvarigos received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 863876 in the context of the FLEXGRID project.

G. Tsaousoglou, and N.G. Paterakis are with the Eindhoven University of Technology. I. Sartzetakis, N. Efthymiopoulos, P. Makris, and E. Varvarigos are with the National Technical University of Athens.

I. Introduction

THE increasing penetration of Renewable Energy Sources (RES) in modern power systems necessitates the exploitation of flexible energy resources that can provide services towards continuously balancing supply and demand. To this end, the flexibility capability of small, distributed energy resources (DERs) is considered an important asset that needs to be utilized effectively.

Integrating DERs into the wholesale electricity markets has been a much-discussed topic in the power systems community [1]. There is a general consensus that participation of DERs should be realized via Aggregators, i.e., entities that participate in electricity markets and undertake balance responsibility on behalf of a portfolio of multiple DERs [2], [3]. The portfolio of an Aggregator may consist of small generation facilities (predominantly RES), distributed storage, and controllable electricity consuming assets such as Electric Vehicles (EVs) and Heating, Ventilation, and Air-Conditioning (HVAC) units [4].

A DER is assumed to be registered with an Aggregator, where the latter installs the necessary communication infrastructure that allows it to monitor, forecast and control the electricity profile of the DER. Each DER has a certain set of preferences towards its electricity profile, as well as a cost function that maps its electricity profile to a monetary cost. For example, an EV has an arrival time and a certain amount of energy that it needs to receive (charge) before its departure. If the Aggregator requests the EV to receive less energy than required, then the EV requests a compensation for this flexibility service.

Market participants (buyers and sellers) can trade energy in the day-ahead and/or intra-day markets. This free trade stops at a certain time before delivery time in order for the system operator to ensure that the system will be balanced in real time operation [5]. The time at which trading stops is called gate closure time. After the gate closure, each participant reports a certain energy profile (energy bought/sold) to the system operator. This profile is referred to as the participant's market program.

In real-time operation, the transmission system operator (TSO) is responsible for maintaining the balance between supply and demand. Given a market program for each market participant, the TSO receives the players' offers for providing or requesting balancing energy. A cost optimization problem is run at the TSO side, through which the balancing energy dispatch of each player is determined.

In order for the TSO to be able to solve this optimization problem in a fast and scalable way, the balancing energy offers made by the participants need to be provided in a certain bidding format, which guarantees that the optimization problem is tractable. For example, a participant is typically required to make an offer for upward balancing energy and downward balancing energy for the timeslot ahead. An offer is a mapping that relates a level of balancing energy provision to a certain monetary cost. These offers are typically required to be in a step-wise form, i.e., pairs of price-quantity (e.g. [6]).

While many studies have proposed methods for further allocating the aggregator's cleared quantities to its DERs downstream (e.g. [7]–[9]), fewer studies have coped with the issue of how to aggregate and communicate the DERs' preferences upstream by capturing the aggregated flexibility costs. The difficulty mainly lies in expressing the flexibility costs and local constraints of multiple DERs into an informative but concise offer/bid that the TSO will be able to incorporate in its dispatch optimization problem. Moreover, the dynamic nature of DERs requires that all the necessary computations must run in real-time in order for the Aggregator to dynamically adjust its bids in the balancing market. Finally, it is desirable that the aggregation method is general enough (i.e., not tailored to a specific DER model), so that different types of DERs can register and participate. These four requirements for the aggregator's bid (concise, informative, real-time, general) and their importance were described in detail in [10].

Different aggregation methods have been proposed in various studies. In [11], the energy requirements of a set of EVs are communicated to the electricity market by constructing a set of upper and lower bounds for the aggregated demand across time. Once the Aggregator receives a (aggregated) power dispatch, the power is allocated to the EVs via an auction procedure. In [12], an Aggregator is providing reserves on behalf of a DER portfolio. An inner-box method is used to aggregate DER constraints. The method takes the DER constraints as input and outputs the upper and lower bounds on aggregated net load (considering also time-coupling constraints).

Another family of studies considers the case where an Aggregator acts as a Virtual Power Plant by representing a set of DERs in the market. In electricity markets where a certain participant holds considerable market power, bi-level optimization approaches have shown that the participant's profit can be optimized. In [13], bi-level programming is used to derive the optimal offering strategy of a DER Aggregator in a day-ahead electricity market. However, the cost of flexibility for the demand-response assets is neglected in the model. In [14], a bi-level program is again used to maximize the Aggregator's profits in the day-ahead market, while also considering the cost of demand response. However, in sufficiently competitive electricity markets, it is to the best interest of each participant to bid according to its true cost for energy provision [5]. Many studies consider the Aggregator as a price-taker, that has a certain forecast of the electricity price and only bids an energy quantity. In [15], a stochastic mixed-integer linear program is solved in order to calculate the optimal bid of a DER portfolio in the day-ahead market, assuming a price forecast. In [16], an EV-Aggregator defines its optimal bidding curve, again assuming certain knowledge about electricity market prices. A similar approach is taken in [17], where the Aggregator stochastically optimizes its quantity-only bids in the DA market and real-time market. Nevertheless, it should be noted that a quantity-only bid communicates that the market participant is willing to buy/sell this quantity at any price, which is a case that is more relevant to suppliers/load serving entities/retailers. In contrast, since the real-time balancing market prices can be volatile, flexibility Aggregators could benefit from bidding price-quantity pairs.

Fewer studies have considered an Aggregator that bids in

the real-time balancing market. In [18], an EV-Aggregator is in charge of participating in the day-ahead and real-time (balancing) market on behalf of an EV fleet. However, the Aggregator only bids the desired energy quantity and not a price-quantity function. In [19], a stochastic bi-level mathematical program is used for optimizing the strategy of a price-taking DR-Aggregator in a real-time market. In [20], an EV-fleet Aggregator bids in the day-ahead and real-time market on behalf of the EVs. An optimization problem is solved based on the probability distribution of real-time electricity prices, which is assumed known. In [21], an Aggregator bids an energy quantity in the balancing market, and an event-driven mechanism is applied to the DERs in order to incentivize them to comply with the cleared aggregated quantity.

Summarizing the literature review, existing studies typically assume some type of electricity price forecast, and a price-taking Aggregator that only bids an energy quantity (much like a supplier) instead of price-quantity pairs. Creating price-quantity pairs, is a challenging task for the Aggregator, since the costs and constraints of its DERs have inter-temporal couplings, i.e., the flexibility cost of a DER in the current timeslot is dependent on how the DER flexibility will be controlled in future timeslots. Moreover, the Aggregator's bid must be decided in an online fashion, which means that the available time for computations is very limited.

To this end, learning methods have been studied as a way to facilitate fast decision making in online operation, after having been trained offline. In [22], a deep reinforcement learning method is proposed, through which a price-making Aggregator decides for its energy bids in a day-ahead electricity market. In [23], a neural network is trained to learn how the aggregated consumption of DERs changes with a set of retail prices imposed by the Aggregator to the DERs. A similar model is used in [24], where the authors also used particle swarm optimization to determine the Aggregator's optimal retail price vector that maximizes its profit. However, these studies, again, have not provided a method for the Aggregator to capture its flexibility costs in price-quantity pairs.

In this paper, we propose a generic method for capturing the Aggregator's upward and downward balancing energy cost for a set of DERs that have inter-temporal couplings in their cost functions and/or constraints. The method is not tailored to any specific DER model. Rather, it can be applied to any use case of DERs, regardless of the DER models. We use a fitting function for this purpose. In order to address the uncertainties of the DERs' parameters, we perform offline scenario-based simulations, and use these simulations to train a machine learning (ML) algorithm. Different ML methods are tested and compared. In online operation, the trained ML can be provided with the current state of the DERs, and predict the optimal Aggregator balancing energy offer (prices for given levels of balancing energy) for the next timeslot ahead very quickly and, as our simulation results indicate, with very good accuracy.

II. System Model and Problem Formulation

We consider a DER Aggregator who is responsible for submitting offers for balancing energy on behalf of its portfolio. The high-level procedure is visualized in Fig. 1. In this Section we elaborate on each step of this procedure, and define it mathematically. Our final goal is to provide a method for calculating a proper set of bids for step (3) of Fig. 1, by taking into account the given data of steps (1) and (2), while also formulating an expectation on the outcome of steps (4) and (5).

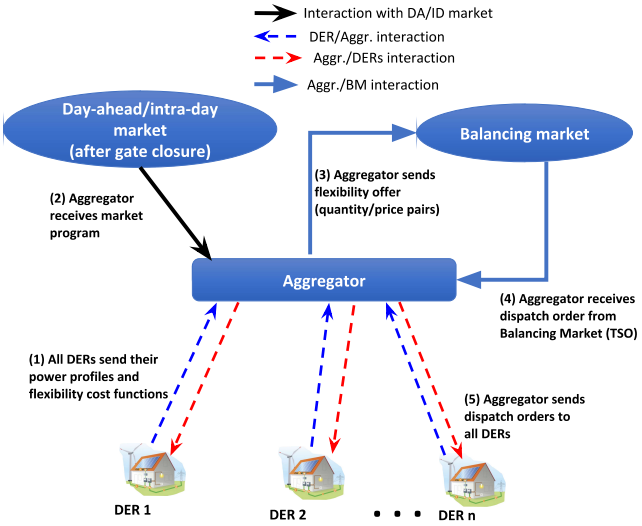

Fig. 1. High-level architecture of the system

Let N denote the set of DERs registered with the Aggregator and T denote a set of timeslots within a particular time horizon. Some DERs can offer certain flexibility with respect to their electricity demand. In particular, the electricity consumption of a flexible DER n in timeslot t can be controlled. We denote this control variable as x_n^t and the respective vector $\boldsymbol{x}_n = \{x_n^1, x_n^2, ..., x_n^{|T|}\}$ denotes the controllable consumption profile of a flexible DER across the time horizon. A certain consumption profile $\boldsymbol{x}_n$, generally comes with a cost for DER n. More specifically, a DER's cost is defined as a function $c_n(\boldsymbol{x}_n)$, representing the compensation that the Aggregator needs to pay to the DER for shaping the latter's consumption profile. Moreover, DER n bears a set of constraints regarding its profile, which for the moment are abstractly denoted as

$$\boldsymbol{x}_n \in F_n. \tag{1}$$

Note that constraints (1) may couple a variable $x_n^{t_1}$ with a variable $x_n^{t_2}$, i.e., a DER's model may exhibit intertemporal couplings. Referring to Fig. 1, step (1) is realized by communicating the $c_n(\boldsymbol{x}_n)$ and F_n of each DER to the Aggregator.

The Aggregator's market program (i.e., the energy bought in the day-ahead market) is denoted by a vector $\mathbf{P}_N = \{\mathrm{P}_N^1, \mathrm{P}_N^2, ...\mathrm{P}_N^{|T|}\}$, where an element P_N^t represents the portfolio's market program for timeslot t. Step (2) of Fig. 1 is realized when the Aggregator receives its market program $\mathbf{P}_N$ upon the gate closure of the day-ahead market.

The aggregated consumption of the Aggregator in timeslot t, in real-time operation, is denoted as X_N^t, where

$$X_N^t = \sum_{n \in N} x_n^t, \tag{2}$$

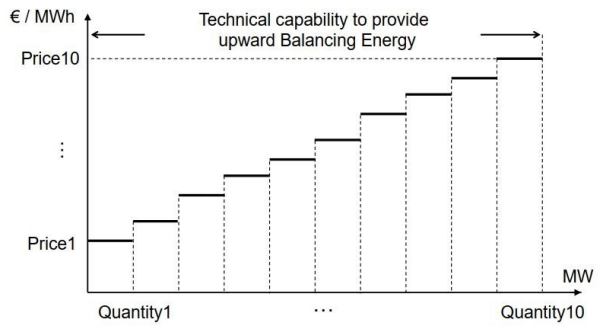

Fig. 2. A typical form of an offer for balancing energy [6].

and the respective vector $\boldsymbol{X}_N$ denotes the Aggregator's consumption profile across the horizon. The difference $\mathrm{P}_N^t - X_N^t$ is the Aggregator's provided balancing energy in timeslot t. Note that it can also take on negative values when the Aggregator absorbs more energy than P_N^t. The Aggregator has to provide a bid/offer in the timeslot τ ahead (i.e. a cost for energy injection and an offer for energy absorption). The bidding format is subject to the rules of the TSO. Typically, it has to be in a form of a step-wise function that defines pairs of balancing energy and price (as in Fig.2), in order to make the economic dispatch problem solvable by standard mixed-integer programming techniques. The above bidding format, although conducive for the TSO, is quite restrictive for the Aggregator, since it cannot fully capture the Aggregator's actual model which is comprised by the cost functions $c_n(\boldsymbol{x}_n)$ and constraints $\boldsymbol{x}_n \in F_n$ of all DERs n in the Aggregator's portfolio. Note also, that the DERs' cost functions and constraints may exhibit inter-temporal dependencies.

In order to facilitate the methodology's presentation, and without loss of generality with respect to its applicability, we assume that the Aggregator offers one price-quantity pair, i.e., a per-unit price b_{up}^τ paired with a maximum quantity $\mathrm{B}_{\mathrm{up}}^\tau$ for upward balancing energy (i.e. injecting power by curtailing electricity consumption) and bids a per-unit price b_{down}^τ and a maximum quantity $\mathrm{B}_{\mathrm{down}}^\tau$ for downward balancing energy in the next timeslot τ (i.e. buying more energy than P_N^τ). The method is generic and can be directly extended to as many pairs as desirable, as it will become clear shortly. The mathematical form of the Aggregator's bid reads as

$$q^\tau(X_N^\tau) = \begin{cases} b_{up}^\tau \left(\mathrm{P}_N^\tau - X_N^\tau\right) & , \quad \mathrm{P}_N^\tau - X_N^\tau \geq 0 \\ b_{down}^\tau \left(X_N^\tau - \mathrm{P}_N^\tau\right) & , \quad \mathrm{P}_N^\tau - X_N^\tau < 0 \end{cases} \tag{3a}$$

$$\mathrm{P}_N^\tau - X_N^\tau \leq \mathrm{B}_{\mathrm{up}}^\tau, \quad \mathrm{P}_N^\tau - X_N^\tau \geq 0 \tag{3b}$$

$$X_N^\tau - \mathrm{P}_N^\tau \leq \mathrm{B}_{\mathrm{down}}^\tau, \quad \mathrm{P}_N^\tau - X_N^\tau < 0, \tag{3c}$$

where $q^\tau(X_N^\tau)$ is the Aggregator's cost function (for upwards and downwards balancing energy) and (3b), (3c) communicate to the TSO that the Aggregator can receive a dispatch up to $\mathrm{B}_{\mathrm{up}}^\tau$ ($\mathrm{B}_{\mathrm{down}}^\tau$) for balancing energy up (down). The proposed method (to be presented later), relates to the calculation of b_{up}^τ and b_{down}^τ for step (3) of Fig. 1.

Next, we describe the balancing market process that leads to step (4) of Fig. 1. The TSO gathers the bids for balancing energy from all the balancing market participants, including the bid (3a)-(3c) of the Aggregator and the set of bids $\mathcal{B}^\tau$ of other market participants, and clears the balancing market close to real-time by solving an economic dispatch problem. The objective of the TSO's economic dispatch problem is to minimize the system's balancing energy cost SC, defined as the sum of the participants' cost functions, while satisfying the system's power balance constraint and respecting the constraints set by the participants' bids. The output of the economic dispatch problem is the balancing energy dispatch decisions for each market participant and the balancing energy prices $\lambda_{\mathrm{up}}^\tau$, $\lambda_{\mathrm{down}}^\tau$ for upwards and downwards balancing energy respectively. Note that the system dispatches balancing energy either upwards or downwards, so that only one of the two prices is non-zero. These prices are typically defined based on the dual variables of the power balance constraints. We denote the balancing energy dispatch of the Aggregator as D^τ, and the tuple of the dispatch instructions for all balancing market participants as $\boldsymbol{\mathcal{D}}^\tau$, where $D^\tau \in \boldsymbol{\mathcal{D}}^\tau$. The economic dispatch problem of the TSO reads as

$$\begin{aligned} &\min_{\boldsymbol{\mathcal{D}}^\tau} \{SC\} \\ &\text{s.t. (3a)} - \text{(3c)} \\ &\quad \text{Bids } \mathcal{B}^\tau \text{of other participants} \\ &\quad \text{Power Balance Constraints} : (\lambda_{\mathrm{up}}^\tau, \lambda_{\mathrm{down}}^\tau). \end{aligned} \tag{4}$$

Thus, the Aggregator's dispatch order D^τ depends on its bid prices $(b_{up}^\tau, b_{down}^\tau)$ through problem (4).

Upon receiving the dispatch order and the price, the Aggregator calculates the power of each DER (in step (5) of Fig. 1) so as to maximize its profit π. The Aggregator receives a revenue $\lambda_{\mathrm{up}}^\tau \cdot \max\{0, (\mathrm{P}_N^\tau - X_N^\tau)\}$ from providing balancing energy up (or a cost $\lambda_{\mathrm{down}}^\tau \cdot \max\{0, (X_N^\tau - \mathrm{P}_N^\tau)\}$ for down), while in case the Aggregator deviates from the TSO's dispatch order, it receives a penalty $\lambda_{\mathrm{Imb}} \cdot |X_N^\tau - D^\tau|$. Finally, the Aggregator pays a cost $\sum_{n\in N} c_n^\tau(x_n^\tau)$ to its DERs in order to shape their profile to $\{\boldsymbol{x}_n\}_{n\in N}$ such that (2) holds. Based on the above, the Aggregator's profit in current timeslot τ, reads as

$$\begin{aligned} \pi^\tau = \lambda_{\mathrm{up}}^\tau \cdot \max\{0, (\mathrm{P}_N^\tau - X_N^\tau)\} - \lambda_{\mathrm{down}}^\tau \cdot \max\{0, (X_N^\tau - \mathrm{P}_N^\tau)\} \\ - \lambda_{\mathrm{Imb}} \cdot |X_N^\tau - D^\tau| - \sum_{n\in N} c_n^\tau(x_n^\tau). \end{aligned} \tag{5}$$

In view of the descriptions of this Section, and towards calculating the bid prices b_{up}^τ and b_{down}^τ for step (3) of Fig. 1, the Aggregator deals with a sequential decision making problem where, in the first stage of current timeslot τ, it decides upon its bid $(b_{up}^\tau, b_{down}^\tau)$, and in the second stage (after receiving its dispatch order) it decides upon the electricity consumption of its DERs $\{x_n^\tau\}_{n\in N}$ and consequently X_N^τ. The decisions are realized and the procedure repeats in the next timeslot. In the first stage decision, the Aggregator's objective is to find the optimal bid $(b_{up}^\tau, b_{down}^\tau)$ that maximizes its expected profit over the second stage decision $\{x_n^\tau\}_{n\in N}, X_N^\tau$ and also over the expected profits of future timeslots. This

is formalized through a multistage stochastic optimization problem that takes a nested form:

$$\max_{b^{\tau}_{up},b^{\tau}_{down}} \left\{ \mathbb{E}\left[\max_{\{x^{\tau}_n\}_{n\in N}, X^{\tau}_N} \pi^{\tau} \right] + \mathbb{E}\left[\max_{b^{\tau+1}_{up},b^{\tau+1}_{down}} \left\{ \mathbb{E}\left[\max_{\{x^{\tau+1}_n\}_{n\in N}, X^{\tau+1}_N} \pi^{\tau+1} \right] + ... \right\} \right] \right\}$$
$$\text{s.t. } (1)-(5), \tag{6}$$

where the expectations are over dispatch orders D^t and prices λ^t_{up}, λ^t_{down}, that depend on decisions $b^{\tau}_{up}, b^{\tau}_{down}$ through problem (4).

Since the Aggregator has no information on the bids $\mathcal{B}^t$ of other players for the current or future timeslots, it cannot tackle problem (6) optimally, since it cannot have an expression for the dispatch orders or prices. In what follows, we propose a method through which the Aggregator can handle this uncertainty upon deciding its bids in the first stage.

Let us consider a set S of arbitrary scenarios $s \in S$ for the Aggregator's dispatch orders, constrained by (3b), (3c), over the entire horizon T. Let the sequence of dispatch orders for a certain scenario s be denoted as $\boldsymbol{D}_s = \{D^1_s, D^2_s, ..., D^{|T|}_s\}$. We consider a conservative strategy, where, given the dispatch information, the Aggregator opts for minimizing its total balancing energy and imbalance costs, as in

$$C^*_s = \min_{\boldsymbol{x}_{n,s}, \boldsymbol{X}_{N,s}} \left\{ \sum_{n\in N} c_n(\boldsymbol{x}_{n,s}) + \sum_{t\in T} \lambda_{\text{Imb}} \cdot |X^t_{N,s} - D^t_s| \right\}$$
$$\text{s.t. } (1), (2). \tag{7}$$

Using problem (7), we can obtain the optimal variables $X^{t,*}_{N,s} \in \boldsymbol{X}^*_{N,s}$ and respective optimal costs C^*_s for each scenario s. Then, for each scenario, we fix the values of $X^{t,*}_{N,s}$, $\forall t \in T$ and C^*_s for each scenario, and solve a fitting problem to decide the variables b^t_{up}, b^t_{down} for the entire horizon, such that the distance between the average Aggregator's cost given by problems (7) and the cost given by the Aggregator's bid function (3a), is minimized:

$$\min_{b^t_{up},b^t_{down}} \left\{ \sum_{s\in S} \left(\sum_{t\in T} q^t(X^{t,*}_{N,s}) - C^*_s \right)^2 \right\}$$
$$\text{s.t. (3a).} \tag{8}$$

Using the above method, the Aggregator can retrieve a mapping from input data $\mathbf{P}_N, c_n(\boldsymbol{x}_n), F_n$ to the decision for its bids $(b^{\tau}_{up}, b^{\tau}_{down})$. The bid estimation method is summarized in Algorithm 1. However, a large number of scenarios may be required before a good approximation is achieved, which can be impractical for real-time operation. A ML solution to this problem is described in the next section.

After receiving the actual dispatch D^{τ} and the balancing market prices $\lambda^{\tau}_{\text{up}}, \lambda^{\tau}_{\text{down}}$ for the current timeslot from the TSO, the Aggregator decides upon X^{τ}_N and $\{x^{\tau}_n\}_{n\in N}$ by greedily maximizing the profit in the current timeslot, assuming that its dispatch for future timeslots $t > \tau$ will be $D^t = \mathrm{P}^t_N$:

$$\max_{\boldsymbol{x}_n, \boldsymbol{X}_N} \left\{ \sum_{t\in T} \pi^t \right\}$$
$$\begin{aligned} \text{s.t. } & D^t = \mathrm{P}^t_N, && \forall t \in T : t > \tau \\ & \lambda^t_{\text{up}} = \lambda^t_{\text{down}} = 0, && \forall t \in T : t > \tau \\ & x^t_n = \widetilde{x}^t_n, && \forall t < \tau, n \in N \\ & (1), (2), \end{aligned} \tag{9}$$

where $\widetilde{x}^t_n$ denotes the decisions made in the previous timeslots (which have to be fixed as decisions about the past cannot be altered).

Algorithm 1 Bid estimation method

1: Read input $\mathbf{P}_N, c_n(\boldsymbol{x}_n), F_n$
2: Create a set S of scenarios for dispatch sequences $\boldsymbol{D}_s$
3: **for** $s \in S$
4: solve problem (7)
5: store values $\boldsymbol{X}^*_{N,s}$, C^*_s
6: Solve problem (8) using $\boldsymbol{X}^*_{N,s}$, C^*_s to estimate bids $(b^{\tau}_{up}, b^{\tau}_{down})$

III. Using Machine Learning for making fast online decisions

The bid estimation part of the method described in the previous section is computationally expensive. This is because set S has size that is exponentially large in the number of timeslots of the horizon T. This implies that many scenarios are needed, where for each scenario the Aggregator has to solve an optimization problem (namely (7)). Thus, in real-time operation, there is no sufficient time to apply the method of the previous section.

In this section, we propose the use of ML techniques to train a decision making system for the Aggregator's bids. We assume that the Aggregator knows the form of functions $c_n(\cdot)$ and constraints F_n and has statistical knowledge over their parameters in the form of probability distributions to which these parameters abide. The input data of the ML algorithm, denoted as $\mathcal{U}$, contains all the parameters necessary for defining $\mathbf{P}_N, c_n(\boldsymbol{x}_n), F_n$. We run Monte Carlo simulations to obtain a number of instances $\mathcal{U}_k$ of the input data, where each $k \in K$ is a certain sample and K is the set of samples. For each instance $\mathcal{U}_k$, we apply the method described in the previous section to obtain the estimated bids $(b^{\tau}_{k,up}, b^{\tau}_{k,down})$. Thus, using $\mathcal{U}_k$ and $(b^{\tau}_{k,up}, b^{\tau}_{k,down})$ as input and output respectively, we can train a ML algorithm. The training procedure is summarized in Algorithm 2. Once trained, the ML algorithm will be able to provide a fast decision on coefficients $(b^{\tau}_{up}, b^{\tau}_{down})$ for the next timeslot ahead, upon receiving the information on $\mathbf{P}_N, c_n(\boldsymbol{x}_n), F_n$ in online operation.

The task at hand is a regression problem. That is, given a specific input, a set of numerical values are predicted. Various ML algorithms were tested for this problem. In this paper, we present the two methods that achieved the most promising results, namely, Deep Neural Networks (DNNs) and Random Forests (RFs), among others that we also tried.

Algorithm 2 ML training using Monte Carlo simulations

1: Generate input data $\{\mathcal{U}_k\}_{k\in K}$
2: **for** $k \in K$
3: execute Algorithm 1
4: store $(b^{\tau}_{k,up}, b^{\tau}_{k,down})$
5: Train the ML algorithm using $\mathcal{U}_k$ and $(b^{\tau}_{k,up}, b^{\tau}_{k,down})$

A. Deep Neural Networks

Deep Neural Networks consist of one input layer through which the features are fed into the network. A number of hidden layers follows, each one comprised of several neurons. The large number of layers in DNNs allows the network to learn complex representations. The challenge is to define the number of hidden layers and neurons in order to balance the accuracy and the computational complexity of the model. There is no standard formula to do this, and a trial and error approach is usually required.

B. Random Forests

Random Forests is an ensemble learning method. Ensemble methods use many learning algorithms combined and can obtain better predictive performance when compared to any of the learning algorithms alone. One ensemble method is bagging of classification or regression trees, where successive trees are independently constructed using a bootstrap sample of the data set. A majority vote is taken for the final prediction. Bagging improves the accuracy and also reduces variance and over-fitting. In random forests, the best split of a given node is decided using a predictor chosen randomly from the set of predictors of that node. Depending on the specific scenario, they can outperform other regression or classification techniques based on support vector machines or neural networks. More information about random forests can be found in [25].

IV. Case Study

For the purpose of evaluating the proposed method, we consider a setting where the Aggregator represents a portfolio of 100 flexible loads and a RES generation facility. The method is evaluated for an operational horizon of 24 timeslots, A load $n \in N$ features an arrival time arr_n and a departure time dep_n. Its feasible interval for energy allocation is denoted as $H_n = [\mathrm{arr}_n, \mathrm{dep}_n] \subset T$.

The portfolio consists of two classes of loads, namely Thermostatically Controlled Loads (TCLs) $j \in N_{TCL}$, including Air-Conditioners, Water Heaters etc., and EVs $i \in N_{EV}$, where $|N_{TCL}| = |N_{EV}| = 50$ and $N = N_{TCL} \cup N_{EV}$. For each family of loads, we present the models below.

An EV $i \in N_{EV}$ is constrained by an upper and lower power consumption level

$$\mathrm{x}_i^{\min} \leq x_i^t \leq \mathrm{x}_i^{\max} \tag{10}$$

and it cannot be charged before arrival or after departure

$$x_i^t = 0, t \notin H_i. \tag{11}$$

Moreover, the EV has a certain energy requirement E_i to be fulfilled. When the total charged energy upon departure is different than E_i, the user suffers a dissatisfaction and needs to be compensated by the Aggregator. This translates to the EV's flexibility cost function, defined as:

$$c_{EV}(\boldsymbol{x}_i) = \begin{cases} 0, & \left|\sum_{t\in T} \mathrm{h}_i \cdot x_i^t - \mathrm{E}_i\right| \leq \mathrm{tol}_i \\ \mathrm{w}_i \cdot \left(\sum_{t\in H_i} \mathrm{h}_i \cdot x_i^t - \mathrm{E}_i\right)^2, & \left|\sum_{t\in T} \mathrm{h}_i \cdot x_i^t - \mathrm{E}_i\right| > \mathrm{tol}_i \end{cases} \tag{12}$$

where tol_i is a tolerance level, h_i is the EV's charging efficiency, and w_i is the load's elasticity parameter. Observe that the EV's cost function exhibits intertemporal couplings since the cost of the EV is only realized at its departure timeslot dep_i, but is, however, dependent on the charging decisions of all previous timeslots.

For TCL $j \in N_{TCL}$ let θ_j^t denote the temperature measured by the TCL's sensor. The transition function of the temperature is defined as:

$$\theta_j^t = \theta_j^{t-1} + \mathrm{ins}_j(\theta_{\mathrm{env}}^t - \theta_j^{t-1}) - \mathrm{con}_j x_j^{t-1} \tag{13}$$

where θ_{env}^t is the environment's temperature, ins_j is a parameter related to temperature decay (e.g. insulation) and con_j is a conversion factor (from electrical power to thermal energy). Similarly to constraints (10) and (11), for TCLs we have:

$$\mathrm{x}_j^{\min} \leq x_j^t \leq \mathrm{x}_j^{\min} \tag{14}$$

$$x_j^t = 0, t \notin H_j \tag{15}$$

where H_j is the TCL's operation interval. The TCL has a setpoint $\theta_{\mathrm{sp},j}^t$, which represents the user's target temperature. Similarly to EVs, the TCL's cost function is defined as

$$c_{TCL}(\boldsymbol{x}_j) = \begin{cases} 0, & \left|\theta_j^t - \theta_{\mathrm{sp},j}^t\right| \leq \mathrm{tol}_j \\ \sum_{t\in[\mathrm{arr}_j,\mathrm{dep}_j]} \mathrm{w}_j \cdot \left(\theta_j^t - \theta_{\mathrm{sp},j}^t\right)^2, & \left|\theta_j^t - \theta_{\mathrm{sp},j}^t\right| > \mathrm{tol}_j. \end{cases} \tag{16}$$

The Aggregator also features local RES generation facilities, with a generation profile $\mathbf{R} = \{\mathrm{R}^1, \mathrm{R}^2, ...\mathrm{R}^{|T|}\}$. In order to obtain realistic values for $\mathbf{P}_N$, each DER's intended demand p_n^t for each timeslot, is set to the value that incurs minimum cost to the DER (assuming no balancing actions by the Aggregator), i.e.

$$\begin{aligned} \mathrm{p}_n^t = \mathrm{argmin}_{x_n^t} \{c_n(\boldsymbol{x}_n)\} \\ \text{s.t. } (10) - (16). \end{aligned} \tag{17}$$

Thus, the Aggregator's net demand profile $\mathbf{P}_N$ under no flexibility actions is defined by

$$\mathrm{P}_N^t = \sum_{n\in N} \mathrm{p}_n^t - \mathrm{R}^t, \quad \forall t \in T. \tag{18}$$

Having defined parameters $\mathbf{P}_N$, cost functions $c_n(\boldsymbol{x}_n)$ and the feasible sets for $\boldsymbol{x}_n$, we can apply the method

TABLE I
VALUES/DISTRIBUTIONS OF SETTING'S PARAMETERS

Parameter	Comments	Value	Average Value	Standard deviation
$\mathrm{x}_n^{\min}$	$\forall n$	0	-	-
$\mathrm{x}_i^{\max}$	for EVs	-	3	0.1
$\mathrm{x}_j^{\max}$	for TCLs	-	5	0.5
arr_i	for EVs	-	4	2.5
arr_j	for TCLs	-	3	1
dep_i	for EVs	-	$\mathrm{arr}_n + 4$	1
dep_j	for TCLs	$\lvert T\rvert$	-	-
E_i	-	-	$\mathrm{x}_i^{\max}(\mathrm{dep}_i - \mathrm{arr}_i) - 2$	0.5
ins_j	-	-	0.05	0.01
$\theta^t_{\mathrm{sp},j}$	-	-	77	1
w_n	$\forall n$	-	0.5	0.1

proposed in the previous section. Specifically, parameters $\theta^t_{\mathrm{env}}, \mathrm{R}^t, \mathrm{arr}^t_n, \mathrm{dep}^t_n, \mathrm{E}_i, \theta^t_{\mathrm{sp},j}, \mathrm{w}_n$ are the features together with $\mathbf{P}_N$.

V. EVALUATION FRAMEWORK

In this section, we evaluate the proposed method for the case study presented. The EVs' charging efficiency, h_i, follows a uniform distribution between 94% and 100%, while the parameter con_j is uniformly sampled from the interval [3, 4]. The average outdoors temperature $\theta_{\mathrm{env}}(average)$ is assumed to follow the temperature of a typical summer day in southern Europe: $\theta^0_{\mathrm{env}}(average) = 83$ F and $\theta^t_{\mathrm{env}}(average) = \theta^{t-1}_{\mathrm{env}}(average) + 3$ F, assuming a simulation horizon of 24 timeslots, that represents quarterly intervals from morning to noon. The actual value for θ^t_{env} follows a normal distribution around the respective value of $\theta^t_{\mathrm{env}}(average)$, with a standard deviation of 3 F. The local RES production for each timeslot is sampled from a normal distribution with mean values starting from 2 kWh and increasing by 2 kWh in every next timeslot. The standard deviation for each timeslot is set equal to 0.25 kWh of the respective mean value. The quantities $\mathrm{B}^t_{\mathrm{up}}, \mathrm{B}^t_{\mathrm{down}}$ were set equal to $0.1\mathrm{P}^t_N$. The rest of the setting's parameters are sampled from normal distributions, as those are defined in Table I. Finally, upon solving the optimization problem of the method, the absolute values were linearized by using an auxiliary variable.

A. Wholesale Market Model

In order to evaluate the proposed method, we use a model through which the wholesale electricity market receives the offer of the Aggregator and decides whether it is going to request balancing energy (up or down) from the Aggregator. In reality, this decision is made by problem (4), where the offers from all market participants are taken into account. However, since we are only interested in the Aggregator's dispatch, for the scope of this paper we abstract away the complete market model and construct a Wholesale Electricity Market Module (WEMM) that provides decisions only on the Aggregator's dispatch and the balancing energy price λ^τ for the current timeslot τ.

In case the Aggregator is called to offer balancing energy up (reduce load), it follows that price λ^τ is higher than its offer b^τ_{up}. In this case the WEMM randomly generates a price that is within the interval $[b^\tau_{up}, \lambda_{max}]$, where λ_{max} is the administrative upper bound for the balancing energy price. In case the Aggregator is called to buy balancing energy down (increase load), it follows that price λ^τ is lower than its offer b^τ_{down}. In this case, the WEMM randomly generates a price that is within the interval $[\lambda_{min}, b^\tau_{down}]$, where λ_{min} is the administrative lower bound. Parameters $\lambda_{min}, \lambda_{max}$ are set to zero and 20 cents respectively. The model for the WEMM is described in the following procedure:

1) First, the WEMM receives the Aggregator's offers b^τ_{up} and b^τ_{down} for the timeslot ahead.
2) The module randomly decides if it is going to need upward or downward balancing energy, with equal probability unless stated otherwise.
3) a) for upward balancing energy (the Aggregator reduces its load): The Aggregator is not called, (i.e., requested to follow its market schedule, $D^\tau = \mathrm{P}^\tau_N$) with probability $\varrho_{up,out} = max\{1, b^\tau_{up}/\lambda_{max}\}$. On the other hand, the Aggregator is called to offer balancing energy $\mathrm{B}^\tau_{\mathrm{up}}$, i.e. $D^\tau = \mathrm{P}^\tau_N - \mathrm{B}^\tau_{\mathrm{up}}$, with probability $\varrho_{up,in} = 1 - \varrho_{up,out}$, at a price λ^τ_{up}, which is picked randomly from the interval $[b^\tau_{up}, \lambda_{max}]$.
   b) for downward balancing energy (the Aggregator increases its load): The Aggregator is called to offer balancing energy down $\mathrm{B}^\tau_{\mathrm{down}}$ (i.e., $D^\tau = \mathrm{P}^\tau_N + \mathrm{B}^\tau_{\mathrm{down}}$) with probability $\varrho_{down,in} = max\{1, \lambda_{min}/b^\tau_{down}\}$, at a price λ^τ_{down}, which is picked randomly from the interval $[\lambda_{min}, b^\tau_{down}]$. Finally, the Aggregator is not called, (i.e., requested to follow its market schedule $D^\tau = \mathrm{P}^\tau_N$) with probability $\varrho_{down,out} = 1 - \varrho_{down,in}$.

The procedure through which the setup is simulated is described in Algorithm 3.

Algorithm 3 Evaluation Procedure

1: set $\tau = 1$
2: **while** $\tau \in [1, 24]$
3: feed input features the ML and get ML estimation for $b^\tau_{up}, b^\tau_{down}$
4: feed $b^\tau_{up}, b^\tau_{down}$ to the WEMM and get the dispatch D^τ and price λ^τ
5: solve problem (9) to decide $\{x^\tau_n\}_{n\in N}, X^\tau_N$
6: set $\widetilde{x}^\tau_n$ equal to the solution of (9)
7: $\tau = \tau + 1$

B. Machine Learning Methods

We assumed 1000 samples to generate a single case of mapping from the defined set of features to the optimal bids $b^\tau_{up}, b^\tau_{down}$. We evaluated the ML algorithms for a total of 1000 cases. We considered 2-fold cross validation with 3 repeats. The accuracy metric is the mean absolute error and the standard deviation of the errors.

1) Deep Neural Networks: We used the Keras deep learning library along with tensorflow [26]. The architecture that we considered is the following: an input layer, seven hidden layers and one output layer. The number of neurons of the input layer and of the hidden layers is equal to the number of features. In

TABLE II
ACCURACY OF ML ALGORITHMS

Algorithm	Mean	Standard Deviation
DNN	0.27	0.005
RF	0.29	0.004

the hidden layers we assumed a dropout rate equal to 0.2. The number of neurons of the output layer is equal to the number of coefficients $b^{\tau}_{up}, b^{\tau}_{down}$. We chose the rectifier activation function for the hidden layers and the Adam optimization algorithm [27]. Finally we considered 1000 epochs.

2) Random Forests: We used the Random Forests regressor from the Scikit-Learn library. The number of trees in the forest is 100. The nodes are expanded until all leaves are pure or until all leaves contain less than 2 samples. The minimum number of samples required to be at a leaf node is 1.

VI. RESULTS

A. Comparison of Machine Learning Methods

In Table II, we present the mean and the standard deviation of the score (defined as the Mean Absolute Error) of the ML algorithms for the aforementioned scenarios. We notice that both algorithms are sufficiently accurate even with a relatively low amount of training cases. RFs perform a bit better than DNNs, but the difference is not very large to be deemed significant. Once the data is generated, DNNs require a training time of 60 seconds on a Quad Core CPU at 4 GHz, while RFs require 10 seconds. Note that these running times refer to the training phase. The resulting estimations require much less time (around 0.11 seconds) to provide a bid estimate. Thus, both models are suitable for dynamic scenarios to promptly acquire a bidding decision.

B. Aggregator Profits

Algorithm 3 was run for a number of different cases for the imbalance price λ_{Imb}. More specifically, the setting was simulated for $\lambda_{Imb} = \{0, 5, 10, ..., 40\}$. For each value of λ_{Imb}, a number of setting instances were simulated and the results on the Aggregator's profits were averaged out over all instances. Figure 3 shows the resulting average Aggregator's profits as a function of λ_{Imb}. The Aggregator's profits are always positive. This is not trivial, since if the Aggregator does not submit bids and follows its market schedule it obviously makes zero profit, and if the bidding method performed poorly (e.g. resulted in major imbalances or DER costs), the Aggregator's profit could well be negative.

As it can be observed, the Aggregator's profits decline for higher values of λ_{Imb}. However, the curve gradually stabilizes, especially after λ_{Imb} surpasses λ_{max}, which means that as λ_{Imb} increases, the profits are no longer affected significantly by λ_{Imb}. The reason for this, is that for $\lambda_{Imb} > \lambda_{max}$, the Aggregator opts for minimizing its imbalances. Thus, the fact that the profits are not affected by λ_{Imb} after a certain point, means that the Aggregator succeeds in minimizing imbalances, which in turn indicates that the proposed ML method achieves a very good capturing of the Aggregator's flexibility cost, i.e., the offers made by the ML method do not result in dispatch decisions that the Aggregator cannot follow.

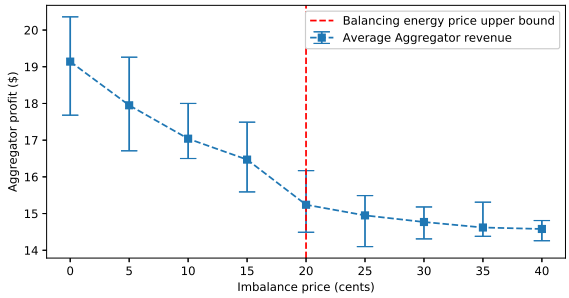

Fig. 3. Aggregator's profit as a function of the imbalance price

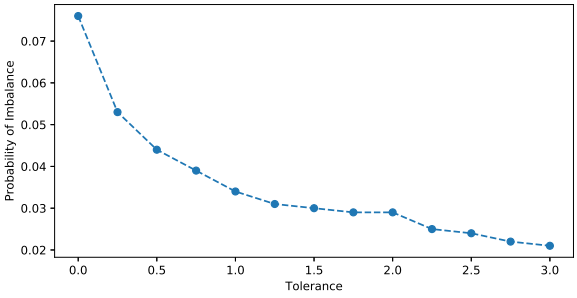

Fig. 4. Estimated probability of imbalance for different values of the tolerance level $\mathbf{tol}_n$

C. Imbalances

In order to verify the indication of the previous subsection, we estimated the probability that the Aggregator's offer results in a dispatch order which the Aggregator prefers to not follow. This can happen when the flexibility costs of the DERs for following the dispatch, are higher than the imbalance price (which, in turn, means that the estimate of flexibility costs by the ML algorithm was not good). The setting was simulated for different values of parameter tol_n (the same for all DERs). For each value of tol_n, we conducted a number of 1000 simulations and counted the number of experiments in which an imbalance occurred (no matter how small). The probability of imbalance was estimated as the number of experiments with imbalances, divided by 1000, and is depicted in Fig. 4 for different values of tol_n.

D. Flexibility aggregation

In this subsection we examine how well the proposed flexibility aggregation algorithm captures the DERs' flexibility level. In order to control the overall flexibility of the DERs via a single parameter, we use the tolerance tol_n. A lower value of tol_n means lower flexibility for the set of DERs, since their cost functions (12), (16) are activated more easily. In contrast, a high tol_n, gives the Aggregator more flexibility to shape

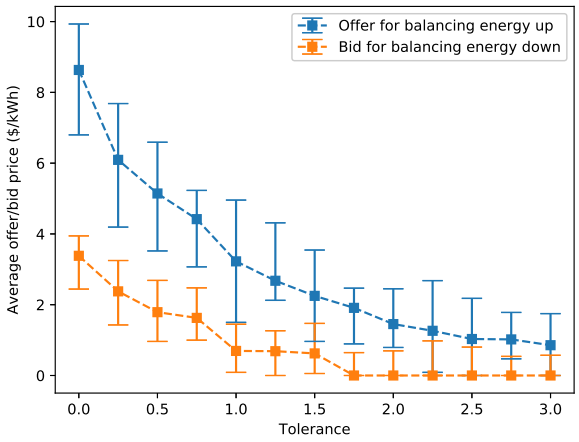

Fig. 5. Average Aggregator's offers/bids for different levels of DER flexibility

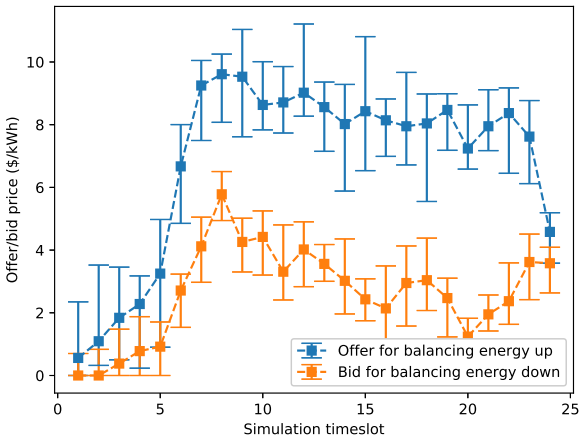

Fig. 6. Aggregator's offers/bids for each timeslot

the DERs' profiles without suffering flexibility costs. Figure 5 presents the resulted Aggregator's bids (averaged over all timeslots) for different values of tol_n.

The figure verifies that for higher values of tol_n, the method is able to capture the increased flexibility of the DERs, since it results in lower bids. Note that a lower bid means that the Aggregator is more likely to be dispatched (even for a lower price), therefore it communicates to the TSO that the Aggregator is more flexible towards offering balancing energy.

E. Bidding Behavior

For the purposes of this experiment, we modified the WEMM to always ask from the Aggregator to offer balancing energy up (curtail load). Thus, the Aggregator curtails energy consumption and in every next timeslot it is asked to curtail again. Figure 6 shows how the Aggregator's offers b^t_{up}, b^t_{down} are affected in this case along the horizon. The results indicate that when the Aggregator is asked to curtail energy in a given timeslot, the proposed method increases the requested price for further curtailment in the next timeslot and, after a certain point, increases also the Aggregator's offer to buy balancing energy. After a certain point in time, this phenomenon is counter-balanced by the departure of many EVs (and the arrival of new ones), which is why the offer/bid prices do not further increase after that point.

VII. Conclusions

In this paper we presented a generic method for aggregating the flexibility costs of distributed energy resources and constructing a per-timeslot price-quantity offer of an Aggregator that represents a portfolio of flexible assets. Machine Learning methods were used in order to enable the Aggregator to calculate its offer in an online fashion, for the real-time balancing market. Different machine learning techniques were tested and compared in a case study. Finally, the Aggregator's market participation was simulated, and the results showed that the method performs well towards capturing the portfolio's flexibility costs. In particular, the method's application does not result in dispatch decisions that are not profitable for the Aggregator to follow.

References

[1] Q. Wang, C. Zhang, Y. Ding, G. Xydis, J. Wang, and J. Østergaard, "Review of real-time electricity markets for integrating distributed energy resources and demand response," *Applied Energy*, vol. 138, pp. 695 – 706, 2015.

[2] L. Gkatzikis, I. Koutsopoulos, and T. Salonidis, "The Role of Aggregators in Smart Grid Demand Response Markets," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 7, pp. 1247–1257, 2013.

[3] G. Tsaousoglou, P. Makris, and E. Varvarigos, "Electricity market policies for penalizing volatility and scheduling strategies: The value of aggregation, flexibility, and correlation," *Sust. Energy, Grids and Networks (SEGAN)*, vol. 12, pp. 57–68, December 2017.

[4] G. Tsaousoglou, P. Pinson, and N. G. Paterakis, "Max-min fairness for demand side management under high res penetration: Dealing with undefined consumer valuation functions," in *2020 International Conference on Smart Energy Systems and Technologies (SEST)*, 2020, pp. 1–6.

[5] D. Kirschen and G. Strbac, *Participating in Markets for Electrical Energy*. John Wiley Sons, Ltd, 2005, ch. 4, pp. 73–104.

[6] "Balancing market detailed design," https://www.admie.gr/uploads/media/Balancing_Detailed_Design_-_Public_Consultation_201712.pdf.

[7] G. Tsaousoglou, K. Steriotis, N. Efthymiopoulos, K. Smpoukis, and E. Varvarigos, "Near-optimal demand side management for retail electricity markets with strategic users and coupling constraints," *Sustainable Energy, Grids and Networks*, vol. 19, 2019.

[8] G. Tsaousoglou, K. Steriotis, N. Efthymiopoulos, P. Makris, and E. Varvarigos, "Truthful, practical and privacy-aware demand response in the smart grid via a distributed and optimal mechanism," *IEEE Transactions on Smart Grid*, pp. 1–1, 2020.

[9] G. Tsaousoglou, P. Pinson, and N. G. Paterakis, "Transactive energy for flexible prosumers using algorithmic game theory," *IEEE Transactions on Sustainable Energy*, 2021.

[10] T. Li, S. H. Low, and A. Wierman, "Real-time flexibility feedback for closed-loop aggregator and system operator coordination," arXiv 2006.13814, 2020.

[11] S. Vandael, B. Claessens, M. Hommelberg, T. Holvoet, and G. Deconinck, "A scalable three-step approach for demand side management of plug-in hybrid vehicles," *IEEE Transactions on Smart Grid*, vol. 4, no. 2, pp. 720–728, 2013.

[12] X. Chen, E. Dall'Anese, C. Zhao, and N. Li, "Aggregate power flexibility in unbalanced distribution systems," *IEEE Transactions on Smart Grid*, vol. 11, no. 1, pp. 258–269, 2020.

[13] E. G. Kardakos, C. K. Simoglou, and A. G. Bakirtzis, "Optimal offering strategy of a virtual power plant: A stochastic bi-level approach," *IEEE Transactions on Smart Grid*, vol. 7, no. 2, pp. 794–806, 2016.

[14] K. Steriotis, K. Smpoukis, N. Efthymiopoulos, G. Tsaousoglou, P. Makris, and E. Varvarigos, "Strategic and network-aware bidding policy for electric utilities through the optimal orchestration of a virtual and heterogeneous flexibility assets' portfolio," *Electric Power Systems Research*, vol. 184, 2020.

[15] M. Di Somma, G. Graditi, and P. Siano, "Optimal bidding strategy for a der aggregator in the day-ahead market in the presence of demand flexibility," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 2, pp. 1509–1519, 2019.

[16] Y. Vardanyan, F. Banis, S. A. Pourmousavi, and H. Madsen, "Optimal coordinated bidding of a profit-maximizing ev aggregator under uncertainty," in *2018 IEEE International Energy Conference (ENERGYCON)*, 2018, pp. 1–6.

[17] J. Iria, F. Soares, and M. Matos, "Optimal supply and demand bidding strategy for an aggregator of small prosumers," *Applied Energy*, vol. 213, pp. 658 – 669, 2018.

[18] S. I. Vagropoulos and A. G. Bakirtzis, "Optimal bidding strategy for electric vehicle aggregators in electricity markets," *IEEE Transactions on Power Systems*, vol. 28, no. 4, pp. 4031–4041, 2013.

[19] R. Henríquez, G. Wenzel, D. E. Olivares, and M. Negrete-Pincetic, "Participation of demand response aggregators in electricity markets: Optimal portfolio management," *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 4861–4871, 2018.

[20] H. Yang, S. Zhang, J. Qiu, D. Qiu, M. Lai, and Z. Dong, "Cvar-constrained optimal bidding of electric vehicle aggregators in day-ahead and real-time markets," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 5, pp. 2555–2565, 2017.

[21] W. Pei, Y. Du, W. Deng, K. Sheng, H. Xiao, and H. Qu, "Optimal bidding strategy and intramarket mechanism of microgrid aggregator in real-time balancing market," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 2, pp. 587–596, 2016.

[22] Y. Ye, D. Qiu, J. Li, and G. Strbac, "Multi-period and multi-spatial equilibrium analysis in imperfect electricity markets: A novel multi-agent deep reinforcement learning approach," *IEEE Access*, vol. 7, pp. 130 515–130 529, 2019.

[23] N. G. Paterakis, A. Taşcıkaraoğlu, O. Erdinç, A. G. Bakirtzis, and J. P. S. Catalão, "Assessment of demand-response-driven load pattern elasticity using a combined approach for smart households," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 4, pp. 1529–1539, 2016.

[24] K. Dehghanpour, M. H. Nehrir, J. W. Sheppard, and N. C. Kelly, "Agent-based modeling of retail electrical energy markets with demand response," *IEEE Transactions on Smart Grid*, vol. 9, no. 4, pp. 3465–3475, 2018.

[25] Tin Kam Ho, "Random decision forests," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, 1995, pp. 278–282 vol.1.

[26] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265–283.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv 1412.6980, 2014.