Università degli Studi di Padova

DIPARTIMENTO DI INGEGNERIA INDUSTRIALE
Corso di Laurea Magistrale in Ingegneria dell'Energia Elettrica

Stochastic Optimization Model for a Smart Retailer

Candidato:
Paolo Nervosi
Matricola 1076467

Relatore:
Ch.mo Prof. Arturo Lorenzoni

Correlatore:
Nicolò Mazzi

Anno Accademico 2015–2016

Abstract

Nowadays, the relevance of distributed renewable generation is growing and, given their unpredictable nature, the management of this typology of power plants is absolutely important. This thesis takes into consideration a retailer that has the challenging task of operating an energy community composed of a number of domestic loads, a photovoltaic power plant and a storage system. This community is connected to the main grid, therefore the retailer can also interact with the electricity market.
A model has been realized to help the aggregator (or the retailer) to take the best decisions concerning the bids in the Day-ahead Market; the aim is to provide electricity to the loads minimizing the costs. At 12 p.m. the energy to purchase or sell in each hour of the following day has to be scheduled and each player of the electricity market has to fulfill the programs of injection. In addition, since the retailer has to balance the instantaneous power flow, it is possible to buy/sell electricity in a Balancing Market, but this is more expensive/less profitable because different prices are applied to the two markets mentioned. The participation to the Balancing Market is seen as an unbalance for the main grid, therefore the aggregator has to be responsible for the power flow coming from its community.
Dealing with renewable sources is a challenge, in particular for their unpredictable nature. For this reason a stochastic approach has been implemented, considering the uncertainty of solar production and, of course, of energy demand and electricity price. To deal with this stochasticity a data analysis aimed at the scenarios creation have been realized. The model is based on *two-stage stochastic programming*. To show its efficiency the stochastic model is finally compared with another model based on a naive deterministic approach. The analysis showed that in the simulation performed, a significant cost reduction can be achieved when implementing a stochastic optimization compared to a deterministic one. Further improvement of the model could go in the direction of an expansion of the data analysis in order to create more accurate scenarios.

Contents

List of Figures

List of Tables

Nomenclature

Set

S	Set of photovoltaic pruduction scenarios	
R	Set of electricity price scenarios	
U	Set of load scenarios	
T	Set of time steps	

Parameters

$PV_{t,s}$	Photovoltaic production at time t in scenario s	[kW]
$D_{t,u}$	Demand of load at time t in scenario u	[kW]
$C_{t,r}^{sell}$	Sales price of electricity in the day-ahead market at time t in scenario r	[€/kWh]
$C_{t,r}^{buy}$	Purchasing price of electricity in the day-ahead market at time t in scenario r	[€/kWh]
$C_{t,r}^{sell,B}$	Sales price of electricity in the balancing market at time t in scenario r	[€/kWh]
$C_{t,r}^{buy,B}$	Purchasing price of electricity in the balancing market at time t in scenario r	[€/kWh]
$C_{avg,r}$	Average daily price in scenario r	[€/kWh]
$\pi_{s,t}$	Probability of scenario s at time t	
$\pi_{r,t}$	Probability of scenario r at time t	
$\pi_{u,t}$	Probability of scenario u at time t	
η	Storage efficiency	
α	Price ratio $\frac{C_{t,r}^{buy}}{C_{t,r}^{sell}}$	
β	Unbalance penalty for the Balancing Market	
E^{max}	Maximum energy storage level	[kWh]
E^{min}	Minimum energy storage level	[kWh]
P_{sell}^{max}	Maximum power sold to the grid each time step	[kW]
P_{buy}^{max}	Maximum power purchased from the grid each time step	[kW]
$\Delta P_{sell,B}^{max}$	Maximum power sold to the grid in the balancing market each time step	[kW]
$\Delta P_{buy,B}^{max}$	Maximum power purchased from the grid in the balancing market each time step	[kW]
P_{ch}^{max}	Maximum power charge each time step	[kW]
P_{disch}^{max}	Maximum power discharge each time step	[kW]
S^{max}	Photovoltaic power plant capacity	[kW]
D^{max}	Maximum power demand	[kW]

Variables

p_t^{ch}	Power charge at time t	[kW]
p_t^{disch}	Power discharge at time t	[kW]
e_t^{level}	Storage energy level at time t	[kWh]
$\Delta p_t^{sell,B}$	Power sold to the grid in the Balancing market at time t	[kW]
$\Delta p_t^{buy,B}$	Power purchased from the grid in the Balancing market at time t	[kW]
p_t^{sell}	Power sold to the grid in the Day-ahead market at time t	[kW]
p_t^{buy}	Power purchased from the grid in the Day-ahead market at time t	[kW]

Part I

Introduction

1

Research motivations

1.1 The role of renewable energy sources

The International Renewable Energy Agency (IRENA) mentioned in the *Roadmap for a Renewable Energy Future* [1] that

> *the world can reach its sustainable energy and climate change objectives by doubling the share of renewable energy by 2030.*

This is one of the biggest challenges that the world has been facing in the past years and will face in the next decades. According to the International Energy Agency (IEA)[2] one of the main sector in which the consumption of renewable sources is growing is the electricity generation therefore the importance of this sector is intended to grow in the next years. Figure 1.1 shows the growth of the electricity generated from renewable sources in Europe and it is clear that among all the wind and photovoltaic production are responsible of this growing.

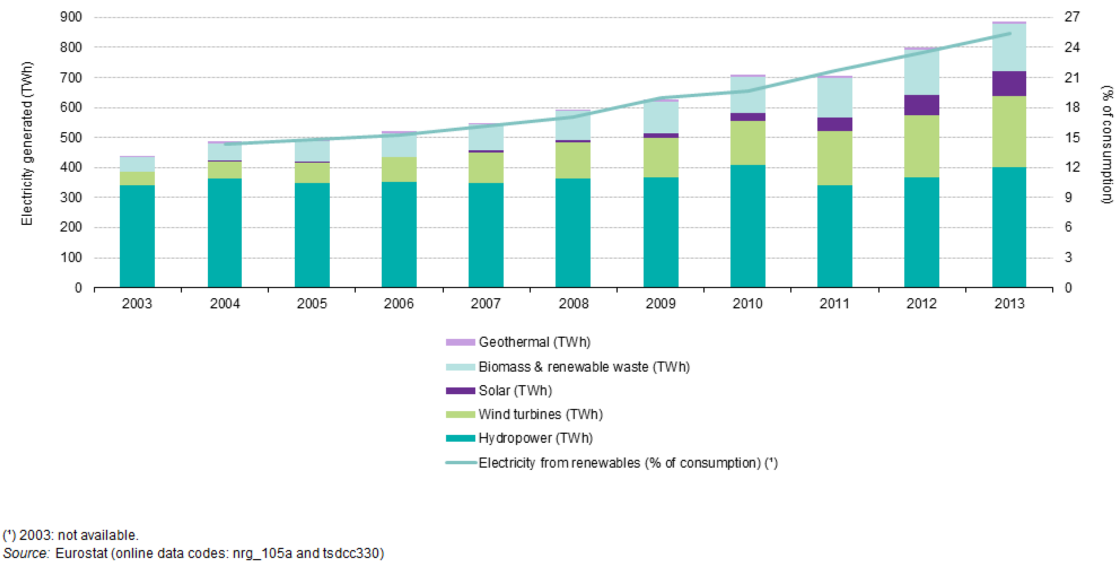

Figure 1.1: Electricity generated from renewable sources in Europe (2003-2013) [3]

To follow these trends of renewable sources the energy system has to be reconsidered in all its features. First of all concerning the generation: from the centralized generation of large power plants to distributed generation in which many small production poles are dislocated in the territory. Associated with this, there are many technical difficulties that have to be considered [4] [5], for example the flow of energy in the network since the production poles often correspond to consumption poles. The small-scale decentralized energy installations coincides with the emergence of so-called *prosumer* [6], an entity that is both a consumer and a producer because owns a small power plant connected to the main grid. Moreover, what makes the renewable sources different from the traditional fossil sources is their unpredictable nature [7]: it is not possible to know certainly the future electricity production from renewable energy. For this reason in the last years the storage systems have been growing for relevance as a technology that can partially solve the problem of unpredictable production [8]. Along with the technical issues related to the distributed generation, also the framework of the electricity market plays an important role since the *prosumer* has to communicate with it.

All this aspects contribute to the growth of the concept of *Smart Grid* ([9] and [10]) in which energy production from small power plant and storage system are considered together and properly managed in order to satisfy the energy demand. At a lower level the idea of *Micro Grid* can be investigated, in which a small network that may be connected to the main grid has to be operated [11]. The optimization performs a key task in this background in which minimizing the costs for power purchasing (or maximizing the revenues from power selling) is the main objective [12]. Nevertheless, considering the increasing penetration of renewable energy, a new approach in the optimization field is requested; not only the energy demand but also the production has a degree of uncertainty that has to be forecasted therefore a stochastic approach has to be implemented [13].

1.2 Research questions

It is in the framework mentioned that this study takes part. One of the most important issues related to the increasing penetration of renewable energy system into the electric power grid is dealing with the intermittent nature of the sources; another issue concerns the participations of the consumer in the electricity market aiming at minimizing the costs for obtaining electric energy. Therefore the objective of this study is trying to answer the following questions:

- how can an energy community grid-connected, formed of loads, renewable power plants and storage system, be properly operated?
- how can the *prosumer* (and therefore the energy community) interact with an electricity market?
- how can the retailer face the issue of the unpredictable nature of renewable sources?

2

Background

2.1 Previous studies

In literature many studies focused on the control of *Micro Grid*. All this works offer optimization models that need the basis of Operational Reseach to build the mathematical formulation of the real problem. This branch of mathematics helps to make better decisions and to define which strategy has to be followed.
For the issue facing in this studies the linear programming and integer-linear programming (also called linear optimization) is the most popular; in which an objective function has to be maximized (or minimized) and a number of linear constraints have to be satisfied.

In [14] a centralized control for microgrids is presented. The microgrid is a low-voltage distribution network composed of distributed generators, storage systems and controllable loads that can be operated either isolated or connected to the main grid. In this study two market policies are considered, in the first one the microgrid controller aims to serve the total energy demand using its distributed generators as much as possible, without exporting to the main grid, the objective function to minimize is the total cost; in the second policy the microgrid participates to the electricity market buying and selling active and reactive power, the objective function is maximizing the revenue. The approach adopted does not consider any stochasticity. In a similar way in [15] the minimization of the costs of residential households in a smartgrid is investigated; each household may have a renewable generations and a part of the load that can be controlled. A figure called *load serving entity* aims to coordinate the energy consumption of the smartgrid; the model is here formulated as a stochastic programming problem. [16] offers a comparison between a stochastic and a deterministic approach to optimize the management of a microgrid; in the stochastic approach the aleatory inputs have been divided into two types: market-related inputs and power-related inputs. The market-related inputs are treated through a scenario approach while the power-related inputs are treated by mean-risk approach. The results show that the optimization is better when the uncertainty is taken into account. Paper [17] considers a system formed of two microgrids and a wind farm connected to the main distribution grid. Each microgrid is assumed to have a total load capacity of 10 MW with 2 MW of controllable loads coming from water heaters, refrigerators and climate control of houses; this study focus on the economic benefits that can be achieved from peak shaving of the aggregated microgrids.
Other papers, such as [18], consider only the figure of a load aggregator that has to provide

energy to a number of loads, taking advantage of an electric energy storage, without considering local production from distributed generators. In this case the point is focused more on the best bids in the electricity market: the optimal scheduling in the day-ahead market and the optimal operation in the real-time balancing market.

2.2 The Electricity Market today

This section is dedicated to a brief presentation of the main electricity markets existing in Italy.
The Italian Electricity Market (figure 2.1) consists of the Spot Electricity Market (MPE), of the Forward Electricity Market(MTE) and of the Platform for physical delivery of financial contracts concluded on IDEX (CDE) [19].

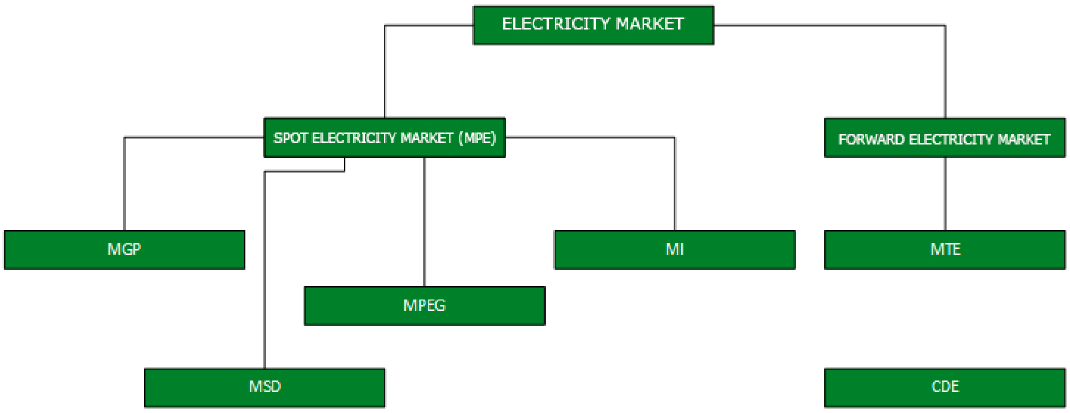

Figure 2.1: Italian Electricity Market structure [19]

The most of energy is treaded on the MPE and it is mainly formed of:

- Day-ahead Market (MGP)
- Intra-day Market (MI)
- Ancillary Service Market (MSD)

In MGP hourly energy blocks are traded for the next day. Participants submit offers/bids where they specify the quantity and the minimum/maximum price at which they are willing to sell/purchase. Bids/offers are accepted after the closure of the market sitting based on the economic merit-order criterion and the capacity limits between zones. The price is determined, for each hour, by the intersection of demand and supply curves and whenever the transmission capacity limits are saturated the price is differentiated from zone to zone. These zones are identified by Terna S.p.A. (the italian Transmission System Operator) and they are:

- North
- North-Central
- South-Central
- Sardegna
- South
- Sicilia

The accepted demand bids pertaining to consuming units are valuated at the PUN ("Prezzo Unico Nazionale") that is the average of the zonal prices weighted for the quantities purchased in each zone. The MGP sitting opens at 8.00 a.m. of the ninth day before the day of delivery and closes at 12 p.m. of the day before the day of delivery. The final results of the MGP are made known at 12.55 p.m. of the day before the day of delivery.
The Intra-day Market (MI) allows the players to adjust the scheduling of the MGP by submitting additional supply offers or demand bids. The MI takes in five different sessions: MI1, MI2, MI3, MI4, MI5. The offers are selected with the same method of the MGP: the economic merit-order criterion. Unlike in MGP, the accepted demand bids are valuated at the zonal price. In figure 2.2 the timeline of MGP and MI is shown.

	MGP	MI1	MI2	MSD1	MB1	MI3	MSD2	MB2	MI4	MSD3	MB3	MI5	MSD4	MB4	MB5
Reference Day	**D–1**				**D**										
Preliminary information	11.30	15.00	16.30	n.a.	n.a.	3.45	n.a.	n.a.	7.45	n.a.	n.a.	11.30	n.a.	n.a.	n.a.
Opening of sitting	8.00*	12.55	12.55	12.55	°	17.30**	°	22.30**	17.30**	°	22.30**	17.30**	°	22.30**	22.30**
Closing of sitting	12.00	15.00	16.30	17.30	°	3.45	°	7.00	7.45	°	11.00	11.30	°	15.00	21.00
Provisional results	12.42	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
Final results	12.55	15.30	17.00	21.10	#	4.15	6.15	#	8.15	10.15	#	12.00	14.15	#	#

* the time refers to the day D-9
**the time refers to the day D-1
° Use is made of bids/offers entered into the MSD1
Dispatching Rules

Figure 2.2: Timeline of the Spot Electricity Market (MPE) [19]

The Ancillary Service Market (MSD) is the venue where Terna S.p.A. procures the resources for the real-time balancing, for managing the system relief of intra-zonal congestion and for the creation of energy reserve. Terna acts as central counterparty and the offers accepted are remunerated at the price offered (pay-as-bid). The MSD consists of:

- ex-ante MSD
- Balancing Market (MB)

In the ex-ante MSD, Terna accepts energy demand bids and supply offers in order to relieve residual congestions and to create reserve margins. The ex-ante MSD is made of four sessions: MSD1, MSD2, MSD3, MSD4. Moreover, through the Balancing Market (MB), Terna accepts energy demand bids and supply offers to provide its service of secondary control and to balance real-time the power flow. The MB consists of five sessions.

2.3 Introduction to stochastic programming

In this section a brief introduction to stochastic programming is given. In many applications a deterministic approach, in which all parameters are fixed and supposed to be known, is not good enough because a random event has to be taken into accont. For example, to perform a good optimization in energy systems with high penetration of renewable energy it is important to deal with the unpredictable nature of this sources. Therefore to achieve this goal a new approach has been considered whenever the parameters are unknown at the time the decisions have to be made [13].

In probability theory a *probability distribution function (pdf)* is a function that describes the density of probability in each point of the sample space [20]. For the sake of semplicity, in figure 2.3 an example of uniform distribution function is reported in which the value of the function within the range$[a, b]$ is costant and equal to

$$f(x) = \frac{1}{b-a} = \frac{1}{8-2} = 0.1667 \tag{2.1}$$

This means that the random variable x has the same probability to assume one of the values in the range $[a, b]$. It's important to note that the range $[a, b]$ is formed of infinite numbers, since the variable x can assume an infinite number of values.

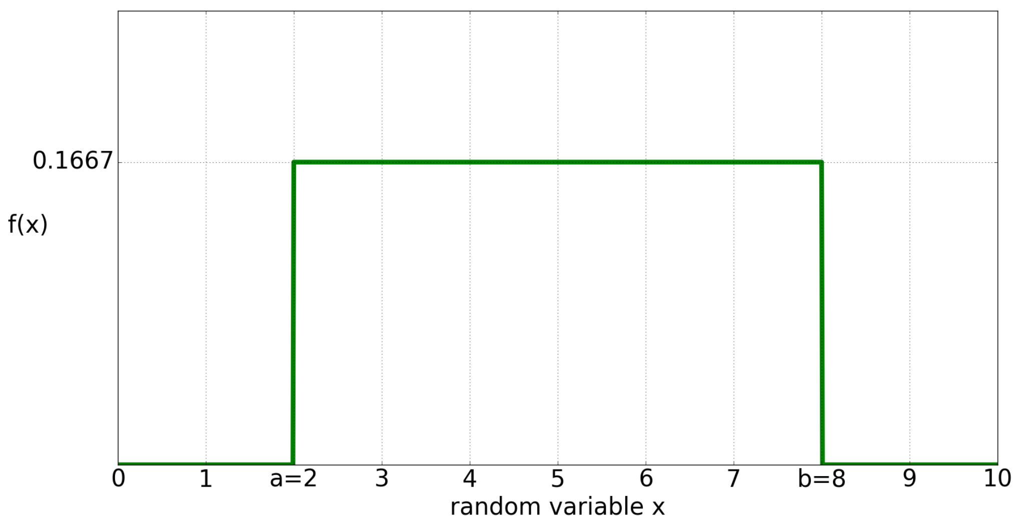

Figure 2.3: Uniform probability distribution function

The probability that the variable x falls within a particular range is the integral of the distribution function over the range considered. For instance, the probability that the random variable x falls within the range [4,5] is equal to the yellow area in figure 2.4 and it can be evaluated through the following integral:

$$\int_4^5 f(x)dx = \int_4^5 \frac{1}{8-2}dx = 0.1667 \tag{2.2}$$

Obviously the integral over the whole range (i.e. the whole area under the function) gives 1 as result.
On the other hand, if the random variable can fall only on a discrete number of values, the function that gives to each value the probability of realization is called *probability mass function (pmf)* [20]. The easiest example of *pms* is that of a fair die (figure 2.5) in which each value has the same probability to be actualized. The sum of all the probabilities must be 1.

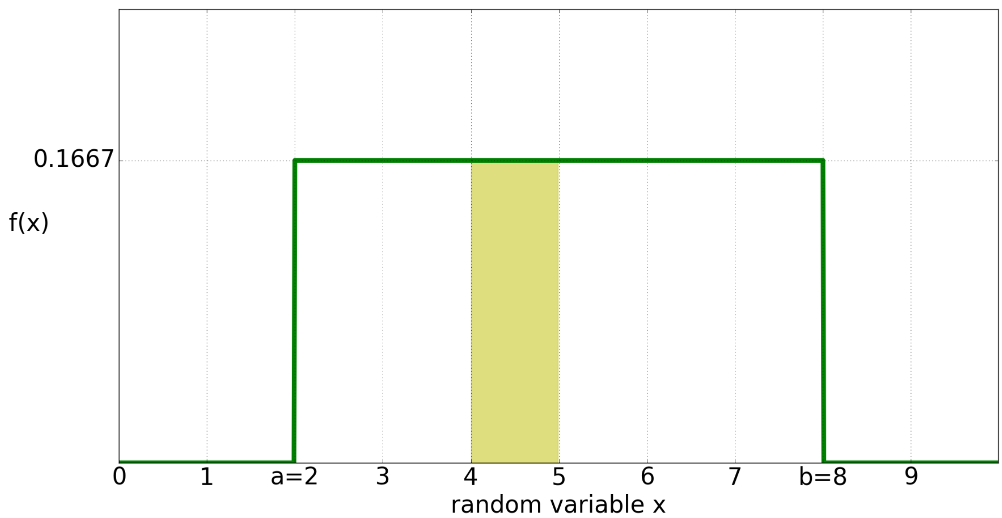

Figure 2.4: Uniform probability distribution function

Stochastic programming deals with *probability mass function* [21] [22]. Therefore, if in the problem considered the random variable can assume infinite values within a specific range, stochastic programming provides a discretization, making the assumption that only a fixed number of values can be realized. If the *probability distribution function* is uniform, the discretization can be random since each value has the same probability to be realized.

In the following example a simple optimization problem in which stochastic programming can be applied is presented.

> *Be δ the unknown demand of coal for tomorrow. To satisfy δ a company can extract today a quantity of coal equal to x. Tomorrow, after the value of δ is shown, the company can adjust the production of coal with the extraction of the quantity y. The time sequence of the events is shown below.*
>
> $\boxed{\text{dedide on } x} \Rightarrow \boxed{\delta \text{ is shown}} \Rightarrow \boxed{\text{decide on } y}$
>
> *Data: δ has the same probability to fall within the range [20,30]. The extraction of coal today costs $10x$. The extraction of coal tomorrow costs $15y$.*

In the problem presented the aim is to find the best value of x that minimize the costs. Once the uncertain parameter δ will be show, the value of y can be evaluated.
To solve this problem with a stochastic programming approach the first idea is to chose a fixed number of values of δ within the range $[20, 30]$. A reasonable choice can be: $\delta_i \in [21, 24, 25, 26, 29]$; in other words five possible scenarios have been created. One way (method A) to formulate the optimization problem is the following:

$$\operatorname*{Min}_{x,y} \quad 10x + 15y \tag{2.3a}$$

$$\text{s.t.} \quad x + y \geq \delta_i \, , \ \forall i \tag{2.3b}$$

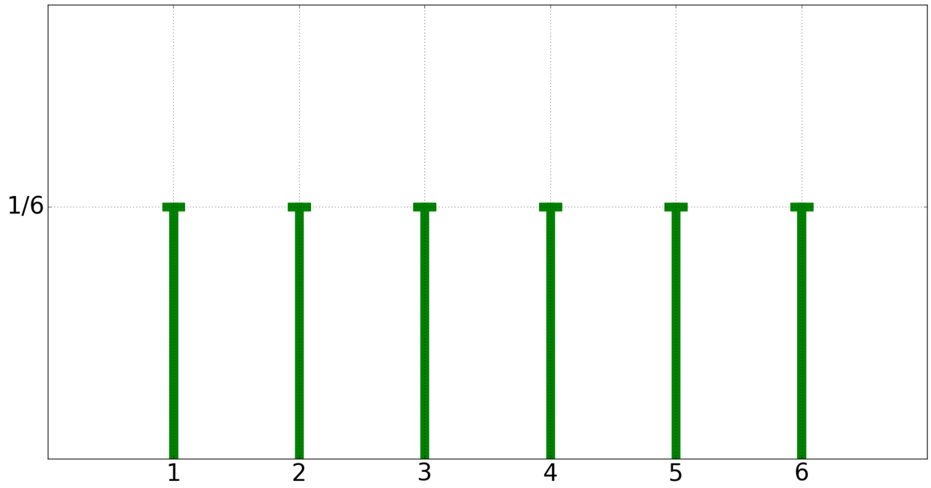

Figure 2.5: Probability mass function of a fair die

where 2.3a is the objective function (i.e. the expectation of the costs) and 2.3b is the coal-demand constraint that must be satisfied for each value of δ_i. The result of this optimization can be easily evaluated noting that $\delta_i = 29$ is the worst case hence the result is:

$$x = 29; y = 0; f_{opt} = 290 \tag{2.4}$$

where f_{opt} is the optimal value of the objective function. Method A considers the variables x and y in the same way, even if y is chosen after the uncertain parameter δ is revealed. Another approach of the stochastic programming makes a distinction between two typologies of decisions: the first-stage decisions, that have to be taken immediately (the variable x in the previous problem), and the second-stage decisions also called recourse actions (the variable y in the previous problem), that can be deferred. Another way to call these two sets of actions that reflects the time sequence of the decisions is *here-and-now* and *wait-and-see* actions. The difference between this two typologies of decisions is that the first-stage decisions are scenario-indipendent while the second-stage decisions are scenario-dependent. In the case considered the five scenarios have the same probability of realization: $\pi_1 = \pi_2 = ... = \pi_5 = 0.2$. This approach is also known as *two-stage stochastic programming with recourse* where the term *recourse* refers to the action taken after the realization of the uncertain data. This approach (method B) is presented below.

$$\underset{x,y}{\text{Min}} \quad 10x + 15 \sum_i \pi_i y_i \tag{2.5a}$$

$$\text{s.t.} \quad x + y_i \geq \delta_i \, , \; \forall i \tag{2.5b}$$

The result of method B is:

$$x = 24 \tag{2.6}$$

$$y_1 = 0; y_2 = 0; y_3 = 1; y_4 = 2; y_5 = 5 \tag{2.7}$$

$$f_{opt} = 264 \tag{2.8}$$

With this formulation the result of the first-stage decision (variable x) is unique while the second-stage decision (variable y) has a different solutions depending on the scenario considered. In 2.5a the expectation of the costs is formulated weighting each scenario according to its probability of realization. It is clear that it's more advisable to solve the problem with the method B since the objective function is better minimized; method A is a simpler stochastic approach that does not consider any recourse action.

Summing up what just explained, *two-stage stochastic programming with recourse* provides a good strategy whenever a decision has to be taken before the realization of some random events. A more formal mathematical formulation is here shown.

$$\underset{x \in \mathbf{R}^n}{\text{Min}} \quad g(x) := c^T x + \mathbb{E}[Q(x, \xi)] \tag{2.9a}$$

$$\text{s.t.} \quad Ax = b \tag{2.9b}$$

$$x \geq 0 \tag{2.9c}$$

where $Q(x, \xi)$ is the optimal value of the second stage problem:

$$\underset{y \in \mathbf{R}^m}{\text{Min}} \quad q^T y \tag{2.10a}$$

$$\text{s.t.} \quad Tx + Wy = h \tag{2.10b}$$

$$y \geq 0 \tag{2.10c}$$

The consequence of taking decisions before some random events occur is that some constraints may not be satisfied for some specific scenarios of the random events. The feasibility is restored by means of recourse actions after the realization of the uncertain data. With the letters x and y, first-stage and second-stage decisions are respectively indicated. The vectors $c \in \mathbf{R}^n$, $b \in \mathbf{R}^m$ and the matrix $A \in \mathbf{R}^{m \times n}$ are supposed to be known. $\xi := (q, h, T, W)$ are the data of the second stage problem and therefore they are characterized by uncertainty. Optimization problem 2.9 states that in the first stage what is minimized is the cost of the *here and now* decisions plus the expected cost of the optimal second-stage decisions. Expectation operator is denoted by $\mathbb{E}[\cdot]$. The second-stage problem 2.10 can be seen as a description of the optimal behaviour after the uncertain data are revealed; similarly, the term Wy can be seen as a correction to restore the feasibility violated in the first-stage problem. The expectation operator in the first-stage problem is taken in respect to the probability distribution of each element of the vector ξ; therefore stochastic programming is based on the assumption of knowing the probability distribution of random variables. It is assumed that the uncertain data ξ can be represented by a set of scenarios $\xi_1 ... \xi_K$ along with their probability of realization $\pi_1 ... \pi_K$. In this way the expectation problem in 2.9a can be formulated as:

$$\mathbb{E}[Q(x, \xi)] = \sum_{k \in K} \pi_k Q(x, \xi_k) \tag{2.11}$$

and the two-stage stochastic problem can be seen in the following way:

$$\min_{x,y} \quad c^T x + \sum_{k \in K} \pi_k q_k^T y_k \tag{2.12a}$$

$$\text{s.t.} \quad T_k x + W_k y_k = h_k \,, \ \forall k \in K \tag{2.12b}$$

$$x \in \mathbf{R}^n \tag{2.12c}$$

$$y_k \in \mathbf{R}^m \,, \ \forall k \in K \tag{2.12d}$$

The result of 2.12 is the optimal solution of the first-stage problem $\overline{x}$, and the optimal solution of the second-stage problem $\overline{y_k}$ for each scenario.

The optimization model presented in this work is based on *two-stage stochastic programming with recourse.*

Part II

Modeling

3

Model set-up

In this study the role of a new figure, a smart retailer (in literature also known as energy aggregator), growing for relevance in the framework mentioned, will be presented. The aggregator leads an energy community and has to provide electricity to the loads that belong to it. To achieve this objective it can make use of a photovoltaic power plant and can interact with the main grid, buying or selling electricity depending on the solar energy available. Furthermore the loads are connected to an energy storage system that can be properly managed in order to satisfy the power balance. In Figure 3.1 and 3.2 it is shown how each element of the energy community is related to the others.

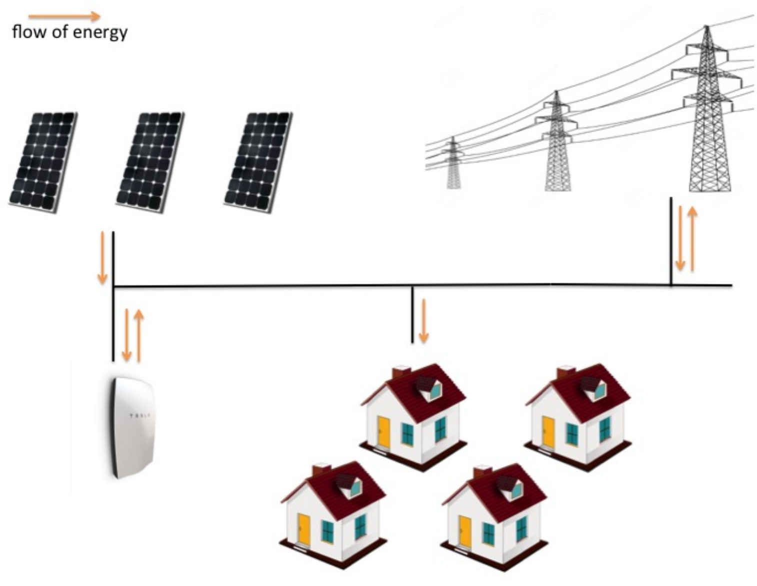

Figure 3.1: Energy community structure

The aim of the aggregator is to find the best strategy in order to minimize the costs of power purchasing. For instance, it can decide to buy electricity to charge the battery when prices are low and, on the other hand, it can sell electricity to the grid in case of high prices. Therefore the flow of energy will depend both on the electricity price and on the energy available from the solar production and the storage.

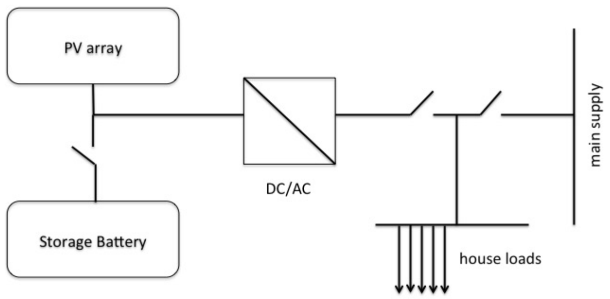

Figure 3.2: Energy community connections

The smart retailer we consider in this study interacts with the electricity market making daily bids for the 24 hours of the following day (i.e. it takes part to the day-ahead market). For this purpose, a stochastic model has been built. This model is made of the Day-ahead model that defines the bids in the day-ahead market and the Real-time model that simulates a possible storage strategy. Once the behaviour of the battery is defined, the energy to buy/sell in the balancing market to balance the power flow of the energy community is consequently set.
Figure 3.3 clarifies better the time sequence with which the two models are run. It can be noted that the Day-ahead model is run once a day (at midday) to decide how much energy to buy or to sell for the 24 hours of the next day, while the Real-time model is run every hour of the day to decide how to manage the storage system in the following hour. This strategy also shows how to resort to the balancing market for the power balance that must be satisfied each instant. In the following figures hourly intervals are indicated with $t0, t1, t2...t23$ where $t0$ means the interval from 12 a.m. to 1 a.m.

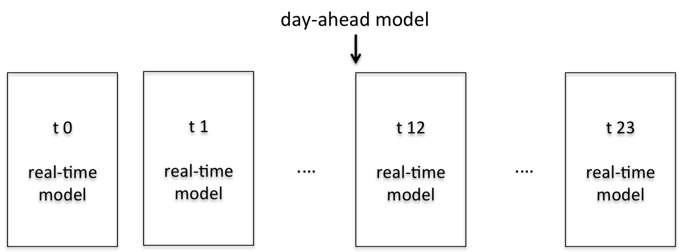

Figure 3.3: Time sequence with which the Day-ahead model and the Real-time model are run

The overall result of the optimization is a combination of the two models because, as can be seen in Figure 3.4, some terms that in the Day-ahead model play the role of decision variables (i.e. p_t^{sell} and p_t^{buy}), in the Real-time model are considered input parameters.
Finally in Figure 3.5 the overall algorithm is presented.

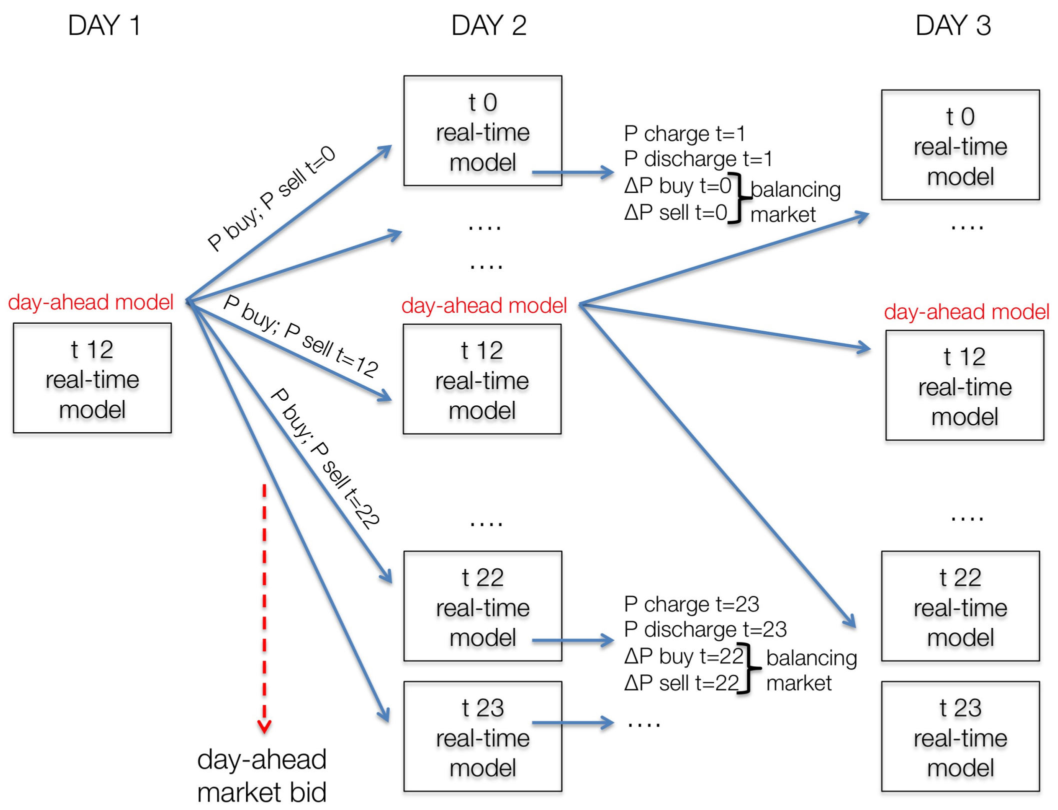

Figure 3.4: Explanation of the relationship between the two models

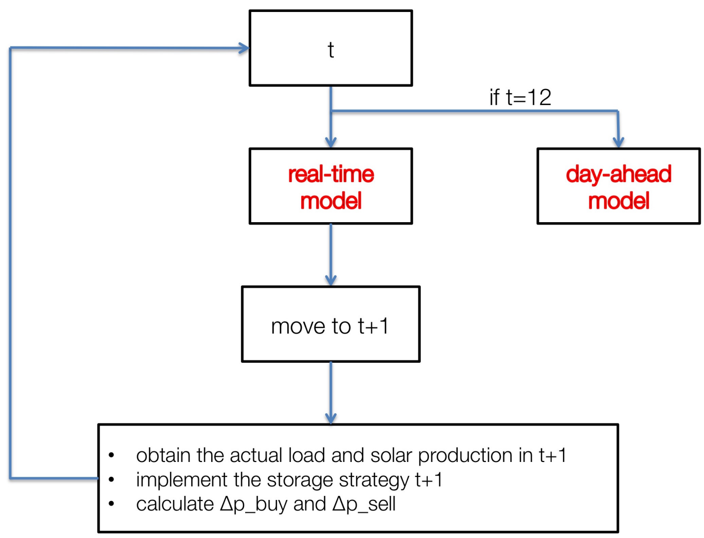

Figure 3.5: Algorithm to set the power scheduling and the storage strategy

The result obtained following this scheme will be finally compared with another model, called naive model, that is based on a deterministic approach and on a simpler storage strategy. After having presented in detail the Day-ahead model and the Real-time model, the naive model will be described. Finally, this part concludes with the data analysis and the explanation of scenarios creation.

3.1 Assumptions

As just explained, the main objective of this study is to realize a model that helps the retailer to take the decisions for the participation in the electricity market.
This market is made of a daily session in which the players make bids for the 24 hours of the following day, this session is called Day-ahead Market and each bid concerns the energy to buy/sell in a time interval of one hour. No negotiations are expected but the participants are responsible for the bids made and they must satisfy them: the following day they have to buy/sell what stated the day before. The closure of the sitting is at 12 p.m. and, as a result, the day-ahead price for the following day is shown. The Intra-day market has not been taken into consideration in this electricity market, therefore the retailer has not the possibilities to adjust its offers/bids. Concerning the power necessary to balance the instantaneous power flow, each participant can buy/sell on the Balancing Market. No bids on the balancing market are considered, this market is just seen as a possibility for the player to adjust any infeasibility with the power balance.
Since for the main network the participation in the Balancing market is seen as an imbalance in relation to the program of injection concluded in the Day-ahead Market, the player that causes this imbalance has to pay a penalty. This means that the energy purchased in the Balancing Market is more expensive than the energy purchased in the Day-ahead Market and, similarly, selling energy in the Day-ahead Market is more profitable than selling in the Balancing Market. It is assumed that this penalty is linked to the day-ahead price through a fixed parameter.
In this sense the electricity market gives high responsibility to the players and this can be well integrated in the framework mentioned in 1.1 in which the distributed generation is growing and the issue of the unpredictable nature of renewable sources has to be faced.

Considering the solar production, no possibilities of photovoltaic curtailment are taken into account, therefore the retailer has to make use of the whole production, providing electricity to the loads, charging the battery or selling energy to the grid. Similarly, load shedding is not considered as well.
Some simplifications have been adopted for the storage systems, in particular it is assumed that the power charge/discharge is always constant (no dependence on the state of charge). Moreover the charge/discharge cycles are not considered. The transmission in each connection line of the scheme 3.2 is assumed to be ideal.

4

Stochastic Model

In this chapter the main model (Day-ahed model) will be first presented and following the storage strategy (Real-time model) will be explained.

4.1 Day-ahead Model

The Day-ahead model is an optimization model that the energy aggregator uses to schedule the power to buy or to sell in the day-ahead market for the following day. The aim is to find the best strategy in order to minimize the costs.
The inputs for this model are the parameters PV_t, D_t, C_t that correspond to the solar production, the energy consumption and the electricity price, while the output are the power scheduled p_t^{sell} and p_t^{buy}. Time steps of one hour are considered.
Even if this work is based on a stochastic model, a first deterministic approach is presented in which the smart retailer supposes to have only one profile available for each input parameters. Following we accept to have a prediction of solar production, demand and prices hence the model turns into a stochastic problem and a set of scenarios for each of these parameters has to be considered.

As already mentioned it is supposed to run the Day-ahead model at 12 p.m. (i.e. at the end of $t11$) to schedule the power to sell or buy for the 24 hours of the next day. In Figure 4.1 the idea of the day-ahead model can be visualized. It's important to notice that between the instant in which this model is run end the next day, there are 12 time steps that have to be considered in the optimization (from $t12$ to $t23$ of the current day). This because at the beginning of each day the energy level in the storage system is not fixed but depends on the storage strategy adopted in the previous hours. It's for this reason that the number of time steps T in the model is 36 but, for the first 12 time steps, the quantity of energy to be sold or bought is already fixed and corresponds to the result of the optimization run the day before of the current day.

In the following descriptions capital letters have been used to identify parameters, while lowercase to identify variables. To avoid misunderstandings it's important to clarify that quantities indicated with the letter p, that refers to the concept of power, are often referred to quantity of energy as well; this is allowed because the time step considered is always one hour and in this period the power is assumed to be constant.

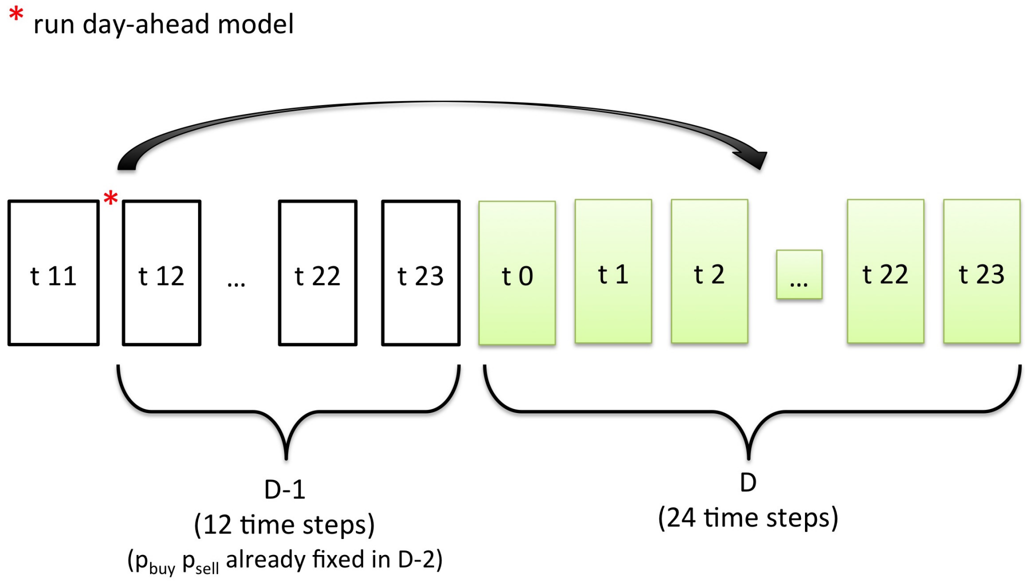

Figure 4.1: Visualization of Day-ahead model

Deterministic approach

$$\Theta = p_t^{buy}, p_t^{sell}$$
$$T = [1, ..., 36]$$
$$T_0 = [1, ..., 12]$$

$$\underset{\Theta}{\text{Min}} \quad \sum_{t \in T} C_t^{buy} p_t^{buy} - C_t^{sell} p_t^{sell} - C_{avg} e_{t=36}^{level} \tag{4.1a}$$

$$\text{s.t.} \quad D_t = PV_t + p_t^{buy} - p_t^{sell} + p_t^{disch} - p_t^{ch} \; , \; \forall t \in T \tag{4.1b}$$

$$E^{min} \leq e_t^{level} \leq E^{max} \; , \; \forall t \in T \tag{4.1c}$$

$$p_t^{buy} \leq P_{buy}^{max} \; , \; \forall t \in T \tag{4.1d}$$

$$p_t^{sell} \leq P_{sell}^{max} \; , \; \forall t \in T \tag{4.1e}$$

$$p_t^{ch} \leq P_{ch}^{max} \; , \; \forall t \in T \tag{4.1f}$$

$$p_t^{disch} \leq P_{disch}^{max} \; , \; \forall t \in T \tag{4.1g}$$

$$e_t^{level} = e_{t-1}^{level} + \eta p_t^{ch} - \frac{p_t^{disch}}{\eta} \; , \; \forall t \in T \tag{4.1h}$$

$$C_{avg} = \frac{\sum_{t \in T} C_t}{T} \tag{4.1i}$$

$$p_t^{buy} = p_{fixed,t}^{buy} \; , \; \forall t \in T_0 \tag{4.1j}$$

$$p_t^{sell} = p_{fixed,t}^{sell} \; , \; \forall t \in T_0 \tag{4.1k}$$

The objective function 4.1a shows the cost to minimize, that is the cost of purchasing energy minus the revenue obtained from the selling of electricity to the grid. Two different prices are applied to the energy bought and sold, according to the relation:

$$\alpha = \frac{C_t^{buy}}{C_t^{sell}} \tag{4.2}$$

where α is bigger then one. Hence for the aggregator it is not convenient to sell the energy that it has bought form the grid but it is incentivized to use the energy produced from the photovoltaic power plan. The last addend of the cost funtion, C_{avg}, represents the value given to the energy in the storage system at the end of the day. It's important to introduce this addend otherwise we should expect, as result of the optimisation, the storage system at its minimum level at the end of the day, as the consumption of the energy in the storage has no price.

Concerning the constraints of the model, 4.1b is the power balance that has to be satisfied each time step. On the same side of the power flow, there are the solar production, the energy purchased from the grid and the energy discharged from the battery; while, on the other hand, there are the consumption, the energy sold to the grid and the energy used to charge the storage. 4.1c guarantees that the energy level is always included in the technical limits of minimum and maximum level of the battery. As it will be shown in the next sections the result of the model is analyzed for different types of storage system, therefore E^{min} and E^{max} change with the battery considered.

The constraints 4.1d and 4.1e refer to the maximum energy that can be purchased and sold each time step and they have been evaluated in the following way:

$$p_t^{buy} \leq P_{buy}^{max} = P_{ch}^{max} + 1.2D^{max} \tag{4.3}$$

$$p_t^{sell} \leq P_{sell}^{max} = P_{disch}^{max} + S^{max} \tag{4.4}$$

Equation 4.3 states that the maximum energy purchased is equal to the maximum energy that can be charged in the battery plus the maximum energy demand, 20% increased; while equation 4.4 states that the maximum energy that can be sold is equal to the solar power plant capacity plus the maximum energy that can be discharged from the battery. Two constraints have been introduced to model the maximum energy that can be charged and discharged each time step (4.1f and 4.1g). An accurate approach would consider this limit depending on the state of charge of the battery but, in this study, a simpler approach has been adopted, dealing with constant value of P_{ch}^{max} and P_{disch}^{max} depending on the typology of battery considered.

4.1h defines the energy level in the storage at each time step; this value is evaluated according to the charging/discharging decision in the same time step and the energy level in the previous time step. When $t = 1$ the term e_{t-1}^{level} refers to the storage level when the model is run. The charging (as well as the discharging) is consider not to be ideal but an efficiency η is taken into account. Because of the non-ideal behavior of the battery, the effective energy stored is $p_t^{ch}\eta$, less than the power charged p_t^{ch}; while in the discharging fase, the energy withdrown is p_t^{disch}/η, more than the effective energy available p_t^{disch} for the loads or for selling.

Finally, the last two constraints 4.1j and 4.1k, as already mentioned, guarantee that p_t^{buy} and p_t^{sell} in the first 12 time steps (that belong to the current day of model running) match the p_t^{buy} and p_t^{sell} already scheduled in the previous day.

Stochastic approach

In the stochastic model a set of scenarios is introduced for each parameter: with the letter S the set of solar producion scenarios is indicated, R refers to the set of electricity price scenarios and U is the set for energy consumption scenarios.
With this approach the model takes the chatacteristics of a two-stage problem and the decision variables are partitioned into two sets. The first-stage variables are those that have to be decided before the actual realization of the uncertain parameters becomes available; once the random events occur, the value of the second stage or recourse variables can be decided. The objective is to choose the first-stage variables is order to minimize the sum of of the first-stage costs and the expected value of the random second-stage or recourse costs. In the case considered the first-stage variables (*here-and-now* varaibles, scenario independent as alredy expleined) are p_t^{sell} and p_t^{buy} that define the energy to buy/sell the following day in period t, while the recourse variables (*wait-and-see* variables, scenario dependent) are $\Delta p_{t,s,r,u}^{sell,B}$ $\Delta p_{t,s,r,u}^{buy,B}$,$p_{t,s,r,u}^{ch}$,$p_{t,s,r,u}^{disch}$,$e_{t,s,r,u}^{level}$. The second-stage variables $\Delta p_{t,s,r,u}^{sell,B}$ $\Delta p_{t,s,r,u}^{buy,B}$, that define the energy to sell/buy in the real time market (balancing market), can also be interpreted as *correction actions* as they are used to compensate any infeasibility from the first-stage variables.
The final model is obtained adding the stochasticity of the parameters one by one: at the beginning only solar production is considered unkown, then price uncertainty is added and finally also power consumption stochasticity is taken into account.

PV stochasticity

$$\Theta = p_t^{buy}, p_t^{sell}, \Delta p_{t,s}^{sell,B}, \Delta p_{t,s}^{buy,B}$$
$$T = [1, ..., 36]$$
$$T_0 = [1, ..., 12]$$

$$\underset{\Theta}{\text{Min}} \quad \sum_{t\in T} \cdot \sum_{s\in S} \pi_{s,t} \cdot \Big[\Big(C_t^{buy} \cdot p_t^{buy} - C_t^{sell} \cdot p_t^{sell} \Big) + + \Big(C_t^{buy,B} \cdot \Delta p_{t,s}^{buy,B} - C_t^{sell,B} \cdot \Delta p_{t,s}^{sell,B} \Big) - C_{avg} e_{t=36}^{level} \Big] \tag{4.5a}$$

subject to
$$D_t = PV_{t,s} + p_t^{buy} - p_t^{sell} + + p_{t,s}^{disch} - p_{t,s}^{ch} + \Delta p_{t,s}^{buy,B} - \Delta p_{t,s}^{sell,B} \,, \forall t \in T \,, \forall s \in S \tag{4.5b}$$
$$E^{min} \leq e_{t,s}^{level} \leq E^{max} \,, \ \forall t \in T \,, \forall s \in S \tag{4.5c}$$
$$p_t^{buy} \leq P_{buy}^{max} \,, \ \forall t \in T \tag{4.5d}$$
$$p_t^{sell} \leq P_{sell}^{max} \,, \ \forall t \in T \tag{4.5e}$$
$$p_{t,s}^{ch} \leq P_{ch}^{max} \,, \ \forall t \in T \,, \forall s \in S \tag{4.5f}$$
$$p_{t,s}^{disch} \leq P_{disch}^{max} \,, \ \forall t \in T \,, \forall s \in S \tag{4.5g}$$
$$e_{t,s}^{level} = e_{t-1,s}^{level} + \eta p_{t,s}^{ch} - \frac{p_{t,s}^{disch}}{\eta} \,, \ \forall t \in T \,, \forall s \in S \tag{4.5h}$$
$$C_{avg} = \frac{\sum_{t\in T} C_t}{T} \tag{4.5i}$$
$$p_t^{buy} = p_{fixed,t}^{buy} \,, \ \forall t \in T_0 \tag{4.5j}$$
$$p_t^{sell} = p_{fixed,t}^{sell} \,, \ \forall t \in T_0 \tag{4.5k}$$

PV and price stochasticity

$$\begin{gathered}\Theta = p_t^{buy}, p_t^{sell}, \Delta p_{t,s,r}^{sell,B}, \Delta p_{t,s,r}^{buy,B} \\ T = [1, ..., 36] \\ T_0 = [1, ..., 12]\end{gathered}$$

$$\begin{aligned}\underset{\Theta}{\text{Min}} \quad & \sum_{t \in T} \cdot \sum_{s \in S} \pi_{s,t} \cdot \sum_{r \in R} \pi_{r,t} \cdot \Big[\Big(C_{t,r}^{buy} \cdot p_t^{buy} - C_{t,r}^{sell} \cdot p_t^{sell} \Big) + \\ & + \Big(C_{t,r}^{buy,B} \cdot \Delta p_{t,s,r}^{buy,B} - C_{t,r}^{sell,B} \cdot \Delta p_{t,s,r}^{sell,B} \Big) - C_{avg,r} e_{t=36}^{level} \Big] \qquad (4.6a)\end{aligned}$$

$$\begin{aligned}\text{subject to} \quad & D_t = PV_{t,s} + p_t^{buy} - p_t^{sell} + p_{t,s,r}^{disch} - p_{t,s,r}^{ch} + \qquad (4.6b) \\ & \quad + \Delta p_{t,s,r}^{buy,B} - \Delta p_{t,s,r}^{sell,B}, \forall t \in T, \forall s \in S, \forall r \in R \\ & E^{min} \le e_{t,s,r}^{level} \le E^{max}, \ \forall t \in T, \forall s \in S, \forall r \in R \qquad (4.6c) \\ & p_t^{buy} \le P_{buy}^{max}, \ \forall t \in T \qquad (4.6d) \\ & p_t^{sell} \le P_{sell}^{max}, \ \forall t \in T \qquad (4.6e) \\ & p_{t,s,r}^{ch} \le P_{ch}^{max}, \ \forall t \in T, \forall s \in S \forall r \in R \qquad (4.6f) \\ & p_{t,s,r}^{disch} \le P_{disch}^{max}, \ \forall t \in T, \forall s \in S \forall r \in R \qquad (4.6g) \\ & e_{t,s,r}^{level} = e_{t-1,s,r}^{level} + \eta p_{t,s,r}^{ch} - \frac{p_{t,s,r}^{disch}}{\eta}, \ \forall t \in T, \forall s \in S, \forall r \in R \qquad (4.6h) \\ & C_{avg,r} = \frac{\sum_{t \in T} C_{t,r}}{T}, \forall r \in R \qquad (4.6i) \\ & p_t^{buy} = p_{fixed,t}^{buy}, \ \forall t \in T_0 \qquad (4.6j) \\ & p_t^{sell} = p_{fixed,t}^{sell}, \ \forall t \in T_0 \qquad (4.6k)\end{aligned}$$

PV price and load stochasticity

$$\Theta = p_t^{buy}, p_t^{sell}, \Delta p_{t,s,r,u}^{sell,B}, \Delta p_{t,s,r,u}^{buy,B}$$
$$T = [1, ..., 36]$$
$$T_0 = [1, ..., 12]$$

$$\underset{\Theta}{\text{Min}} \quad \sum_{t \in T} \cdot \sum_{s \in S} \pi_{s,t} \cdot \sum_{r \in R} \pi_{r,t} \cdot \sum_{u \in U} \pi_{u,t} \cdot \Big[\Big(C_{t,r}^{buy} \cdot p_t^{buy} - C_{t,r}^{sell} \cdot p_t^{sell} \Big) + \\ + \Big(C_{t,r}^{buy,B} \cdot \Delta p_{t,s,r,u}^{buy,B} - C_{t,r}^{sell,B} \cdot \Delta p_{t,s,r,u}^{sell,B} \Big) - C_{avg,r} e_{t=36}^{level} \Big] \tag{4.7a}$$

$$\text{subject to} \quad D_{t,u} = PV_{t,s} + p_t^{buy} - p_t^{sell} + p_{t,s,r,u}^{disch} - p_{t,s,r,u}^{ch} + \\ + \Delta p_{t,s,r,u}^{buy,B} - \Delta p_{t,s,r,u}^{sell,B}, \forall t \in T, \forall s \in S, \forall r \in R, \forall u \in U \tag{4.7b}$$

$$E^{min} \leq e_{t,s,r,u}^{level} \leq E^{max}, \ \forall t \in T, \forall s \in S, \forall r \in R, \forall u \in U \tag{4.7c}$$

$$p_t^{buy} \leq P_{buy}^{max}, \quad \forall t \in T \tag{4.7d}$$

$$p_t^{sell} \leq P_{sell}^{max}, \quad \forall t \in T \tag{4.7e}$$

$$\Delta p_{t,s,r,u}^{buy,B} \leq \Delta P_{buy,B}^{max}, \quad \forall t \in T, \forall s \in S \forall r \in R, \forall u \in U \tag{4.7f}$$

$$\Delta p_{t,s,r,u}^{sell,B} \leq \Delta P_{sell,B}^{max}, \quad \forall t \in T, \forall s \in S \forall r \in R, \forall u \in U \tag{4.7g}$$

$$p_{t,s,r,u}^{ch} \leq P_{ch}^{max}, \ \forall t \in T, \forall s \in S \forall r \in R, \forall u \in U \tag{4.7h}$$

$$p_{t,s,r,u}^{disch} \leq P_{disch}^{max}, \ \forall t \in T, \forall s \in S \forall r \in R, \forall u \in U \tag{4.7i}$$

$$e_{t,s,r,u}^{level} = e_{t-1,s,r,u}^{level} + \eta p_{t,s,r,u}^{ch} - \frac{p_{t,s,r,u}^{disch}}{\eta}, \ \forall t \in T, \forall s \in S, \forall r \in R, \forall u \in U \tag{4.7j}$$

$$C_{avg,r} = \frac{\sum_{t \in T} C_{t,r}}{T}, \forall r \in R \tag{4.7k}$$

$$p_t^{buy} = p_{fixed,t}^{buy}, \ \forall t \in T_0 \tag{4.7l}$$

$$p_t^{sell} = p_{fixed,t}^{sell}, \ \forall t \in T_0 \tag{4.7m}$$

Dealing with a stochastic apporach, each scenario of each set is weighted accordig to its probability of realizaion (i.e. $\pi_{s,t}, \pi_{r,t}, \pi_{u,t}$); since the receding optimization horizon covers more the one day, it is possible that the 36 time steps belong to days in which the scenarios considered have a different probability associated. From this the requirement to give a temporal dependence to each probability. The main difference with the deterministic model is the introduction of a new term in the cost function. This term is introduced to model the action of the energy aggregator in the balancing market. The power scheduled p_t^{sell} and p_t^{buy} must be the same for all the possible scenarios bacause the retailer has to make a bid in the day-ahead market, therefore to follow the power balance the aggregator has to resort to the balancing market buying or selling $\Delta p_{t,s,r,u}^{buy,B}, \Delta p_{t,s,r,u}^{sell,B}$ according to the scenarios that will be realized. The only result of this optimization that will be implemented is the value of the *here-and-now* variables. The *wait-and-seen* variables depend on the scenarios, therefore only with the Real-time model will be disclosed.
In the balancing market a different price has been adopted:

$$C_{t,r}^{buy,B} = C_{t,r}^{buy} \cdot \beta \tag{4.8}$$

$$C_{t,r}^{sell,B} = \frac{C_{t,r}^{sell}}{\beta} \tag{4.9}$$

Since β is bigger than 1 it's much more advantageous for the aggregator to buy and sell in the day-ahead market and to resort to the balancing market only if necessary.
Two new constraints have been introduced in the stochastic model to define the maximum energy that can be purchased or sold in the balancing market: 4.7f and 4.7g. They can be interpreted as a correction of the worst case of power unbalanced caused by a non-optimal power scheduling and storage strategy. Tables 4.1 and 4.2 (in which variables and parameters in the same column belong to the same direction of the power flow) show when the two worst cases can happen.

Table 4.1: Evaluation of the maximum energy purchased in the balancing market (Worst case A)

$p_t^{buy} = 0$ $p_t^{disch} = 0$	$p_t^{sell} = P_{sell}^{max}$ $p_t^{ch} = P_{ch}^{max}$
$PV_t = 0$	$D_t = D^{max}$
$\Downarrow$	$\Downarrow$
$\Delta p_t^{buy,B} = P_{sell}^{max} + P_{ch}^{max} + D^{max}$	$\Delta p_t^{sell,B} = 0$

Table 4.2: Evaluation of the maximum energy sold in the balancing market (Worst case B)

$p_t^{buy} = P_{buy}^{max}$ $p_t^{disch} = P_{disch}^{max}$	$p_t^{sell} = 0$ $p_t^{ch} = 0$
$PV_t = S^{max}$	$D_t = 0$
$\Downarrow$	$\Downarrow$
$\Delta p_t^{buy,B} = 0$	$\Delta p_t^{sell,B} = P_{buy}^{max} + P_{disch}^{max} + S^{max}$

Therefore, from equations 4.3 and 4.4, $\Delta P_{buy,B}^{max}$and$\Delta P_{sell,B}^{max}$ can be written in the following way:

$$\Delta P_{buy,B}^{max}, \Delta P_{sell,B}^{max} = P_{buy}^{max} + P_{sell}^{max} = P_{ch}^{max} + P_{disch}^{max} + S^{max} + D^{max} \tag{4.10}$$

The others constraints are equal to those explained in the deterministic approach, with the only difference that they must be satisfied for each scenario of the sets S, R and U.

4.2 Real-time Model

To run the Day-ahead model, an important role is played by the level of energy in the storage system at the end of $t11$ (i.e. at the time in which the model is run). For this reason another model has been built to simulate what the behaviour of the storage could be. This model (called Real-time) is run each hour of the day to set the storage sytem strategy for the following hour. Once the solar production and the energy consumption is known, also the quantity of energy to buy and sell in the balancing market is defined. In figure 4.2 the model can be visalizaed. The receding optimization horizon is N time steps, but only the result of $t+1$ (i.e the time step called i) is implemented. This model can be better represented with the algorithm shown in Figure 3.5 and here presented.

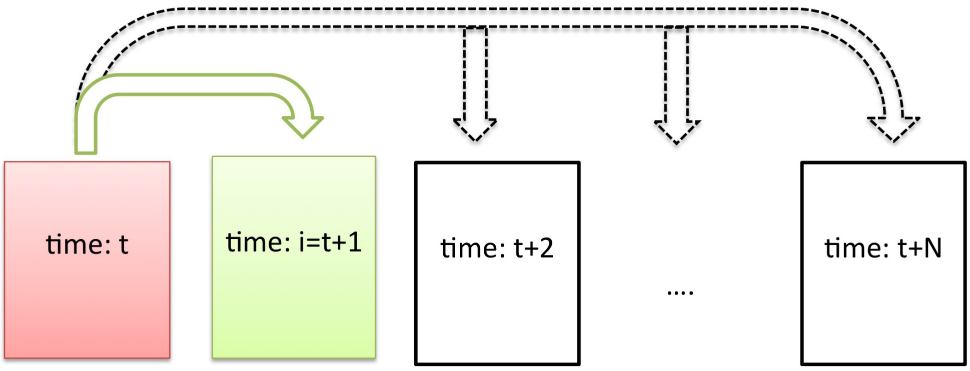

Figure 4.2: Visualization of Real-time model

Inizialization:

- Actual period : t
- Obtain the actual load and solar production in period t

Algorithm:

1. Select the receding optimization horizon N (e.g. 10 hours)
2. Obtain the scheduled P_t^{buy}, P_t^{sell} from the day-ahead optimization and the electricity price $C_t^{buy,B}, C_t^{sell,B}, t \in [i, ..., t+N]$
3. Obtain the forecasted load and solar production for $t \in [i, ..., t+N]$
4. Modify the probability of the scenarios according to the load and solar production in period t
5. Solve the minimization problem:

$$\Theta = \Delta p_{t,s,u}^{sell,B}, \Delta p_{t,s,u}^{buy,B} \quad t \in [i, ..., t+N], s \in S, u \in U$$

$$\underset{\Theta}{\text{Min}} \quad \sum_{t=i}^{i+N} \sum_{s \in S} \pi_{s,t} \cdot \sum_{u \in U} \pi_{u,t} \left[\left(\Delta p_{t,s,u}^{buy,B} C_t^{buy,B} - \Delta p_{t,s,u}^{sell,B} C_t^{sell,B} \right) - C_{avg} \gamma e_{t=N}^{level} \right] \tag{4.11a}$$

subject to

$$D_{t,u} = PV_{t,s} + P_t^{buy} - P_t^{sell} + p_{t,s,u}^{disch} - p_{t,s,u}^{ch} + \tag{4.11b}$$
$$+ \Delta P_{t,s,u}^{buy,B} - \Delta P_{t,s,u}^{sell,B} \,, \forall t \in [i, ..., t+N] \,, \forall s \in S \,, \forall u \in U$$

$$E^{min} \leq e_{t,s,u}^{level} \leq E^{max} \,, \; \forall t \in [i, ..., t+N] \,, \forall s \in S \,, \forall u \in U \tag{4.11c}$$

$$\Delta p_{t,s,u}^{buy,B} \leq \Delta P_{buy,B}^{max} \,, \quad \forall t \in T \,, \forall s \in S \forall r \in R \,, \forall u \in U \tag{4.11d}$$

$$\Delta p_{t,s,u}^{sell,B} \leq \Delta P_{sell,B}^{max} \,, \quad \forall t \in T \,, \forall s \in S \forall r \in R \,, \forall u \in U \tag{4.11e}$$

$$p_{t,s,u}^{ch} \leq P_{ch}^{max} \,, \; \forall t \in T \,, \forall s \in S \forall r \in R \,, \forall u \in U \tag{4.11f}$$

$$p_{t,s,u}^{disch} \leq P_{disch}^{max} \,, \; \forall t \in T \,, \forall s \in S \forall r \in R \,, \forall u \in U \tag{4.11g}$$

$$e_{t,s,u}^{level} = e_{t-1}^{level} + \eta p_{t,s,u}^{ch} - \frac{p_{t,s,u}^{disch}}{\eta} \,, \; t = i \,, \forall s \in S \,, \forall u \in U \tag{4.11h}$$

$$e_{t,s,u}^{level} = e_{t-1,s,u}^{level} + \eta p_{t,s,u}^{ch} - \frac{p_{t,s,u}^{disch}}{\eta} \,, \; \forall t \in [i+1, ..., t+N] \,, \forall s \in S \,, \forall u \in U \tag{4.11i}$$

$$C_{avg} = \frac{\sum_{t \in T} C_t}{T} \,, \; \forall t \in [i, ..., t+N] \tag{4.11j}$$

$$p_{t,s,u}^{ch} = p^{ch,i} \,, t = i \,, \forall s \in S \,, \forall u \in U \tag{4.11k}$$

$$p_{t,s,u}^{disch} = p^{disch,i} \,, t = i \,, \forall s \in S \,, \forall u \in U \tag{4.11l}$$

$$e_{t,s,u}^{level} = e^{level,i} \,, t = i \,, \forall s \in S \,, \forall u \in U \tag{4.11m}$$

6. Move to the next period $t+1$ and show the real load and solar production

7. Implement the period operation i of the storage system: $p^{ch,i}, p^{disch,i}$

8. Obtain the actual $\Delta p_{t+1}^{sell,B}, \Delta p_{t+1}^{buy,B}$ for the power balance

9. Repeat the algorithm from step 1

Again, it may happen that the receding optimization horizon belongs to two different days in which the probability associated to each scenario is not the same, hence the probabilities $\pi_{s,t}, \pi_{u,t}$ have a temporal dipendence. Since this model is run each hour, the probabiliy associeted to each scenario of the following time steps can be properly improved acording to the actual value of solar production and energy consumption. A probability called π_t^{real} is evaluated for each time step of the optimization horizon and for each scenario, comparing the distance between the actual value in time t and the values from $t+1$ to $t+N$ of each scenario (step 4 of the algorithm). A higher probability will be given to the scenario that has a value closer to the actual value. Finally, a new probability called π_t^{new} is evaluated taking into account both π_t^{real} and π_t^{old} where π_t^{old} is that one used in the previous optimization. This method is used both for S and U. The probability $\pi_{s,t}, \pi_{u,t}$ in 4.11a is intended to be π_t^{new}. Below the steps to update the probabilities of solar production scenarios are reported.

1. $a_{t,s} = \mid PV_{now} - PV_{t,s} \mid + k \qquad \forall s \in S, \forall t \in [i, ..., t+N]$
   $b_t = \frac{1}{\sum_{s \in S} 1/a_{t,s}} \qquad \forall t \in [i, ..., t+N]$
   $\pi^{real}_{t,s} = \frac{b_t}{a_{t,s}} \qquad \forall s \in S, \forall t \in [i, ..., t+N]$
2. $\pi^{new}_{t,s} = \lambda \pi^{old}_{t,s} + (1-\lambda)\pi^{real}_{t,s} \qquad \forall s \in S, \forall t \in [i, ..., t+N]$
3. $\pi^{old}_{t,s} = \pi^{new}_{t,s} \qquad \forall s \in S, \forall t \in [i, ..., t+N]$

The distance between two values is shifted through a constant k to avoid any infeasibility in the following steps. The new probability is obtained giving two different weights to $\pi^{real}_{t,s}$ and $\pi^{old}_{t,s}$. Higher is the parameter λ, higher is the weight given to $\pi^{old}_{t,s}$.
Since at the end of the algorithm the probabilities $\pi^{new}_{t,s}$ becomes the $\pi^{old}_{t,s}$ for the next optimization and the temporal horizon is shifted of one time step, it turns out that another value has to be used for $\pi^{old}_{t=i+N,s}$. This is the meaning of the red box in Figure 4.3 in which $N = 10$ has been used. The value that has been given to $\pi^{old}_{t=i+N,s}$ is the one evaluated at the scenario creation time.

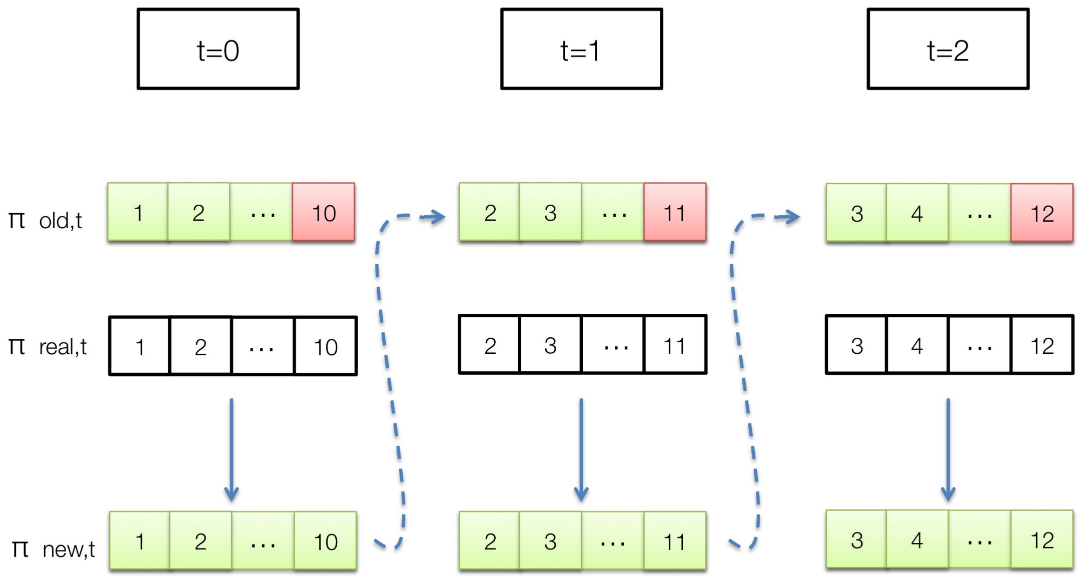

Figure 4.3: Probability update in the Real-time model

Step 5 of the algorithm includes the optimization problem; in the objective function 4.11a the cost of power purchasing in the balancing market has to be minimized. The last term, as arealdy explained in the previous model, gives the value to the energy in the storage system at the end of the receding optimization horizon. The parameter γ has been introduced because the value given to the energy in the storage can be differently set depending on the conditions of use of the battery to implement. The constraints 4.11b to 4.11j have already been explained, while, 4.11k to 4.11m are introduced to force that in the following step $t+1$ the decision variables $p^{ch,i}, p^{disch,i}, e^{level,i}$ are scenario indipendent, since they rapresent the storage system strategy to be applied. It can be noted that in this case no scenarios for electricity prices are considered. This because, as already explained, it is assumed that the penalty for any unbalance is linked to the day-ahead price through the fixed parameter β (see 4.8 and 4.9). The electricity market shows the day-ahead price at 12 p.m., therefore the retailer knows the purchasing/selling price in the Balancing market for the whole receding optimization horizon (obviously if N<12).

With this assumption this model does not consider any stochasticty for the price hence the computational complexity is much lower compared with the Day-ahead model. Once the optimization is concluded it is possible to move to the period $t + 1$ and to implement the storage strategy for this period. The solar production and the energy consumption are shown as well and finally it is possible to evaluate the energy to buy or sell in the balancing market to balance the power flow. Now it is possible to repeat the algorithm from step 1 to decide how to charge/discharge the battery in the following time step. Moving forward in this way, after $t11$ (i.e. 12 p.m.) the Day-ahead model can be run knowing the current battery level.

5

Naive Model

Another model has been built to make a comparison with the stochastic optimization model previously presented. Also in these case the aim is to operate an energy community that has the same characteristics already mentioned.

The retailer has to schedule the energy to buy or sell for the following day and to accomplish this it decides to bid according to the solar production and energy demand forcasted for the next day; in coherence with the model presented in chapter 4, this model is called *Day-ahead naive model* and it is run once a day at 12.00. Morever, as already explained, to get more significant results a secondary model is run every hour to define the storage strategy for the following hour and to show the partecipation in the balancing market; this model is called *Real-time naive model.*
Differently from the stochastic model in which a cost function is intended to be minimized, the *Day-ahead naive model* is based on a strategy much simpler. A detrministic approach has been considered and to define the value of solar production and energy consumption a persistence model has been adopted. The idea of the persistence model is that for the following day the profile of these two parameters is the same of the profile in the prevoius day. Moreover, for the power scheduling strategy, the electrcity price doesn't play any role; according to the value $PV_t^{forecast}$ and $D_t^{forecast}$, the energy to buy or sell in the following day is evaluated, without taking into account the storage system. In figure 5.1 the bidding stretegy in the day-ahead market can be better visualized. The figure reports the profile of the solar production and energy consumption forcasted for the following day; if in the considered time step the load is higher than the photovoltaic, a purchasing offer correspondig to $p_t^{buy} = D_t^{forecast} - PV_t^{forecast}$ will be done for that period. In the same way if the solar production is higher the energy $p_t^{sell} = PV_t^{forecast} - D_t^{forecast}$ will be sold in that period.

Consistently with the optimization model, a storage strategy to implement each hour is performed in the comparison model as well. To decide how to charge/discharge the battery during the following hour, the algorithm shown below has been implemented.
Inizialization:

- Actual period : t
- Obtain the actual load and solar production in period t: PV_t, D_t

Algorithm:

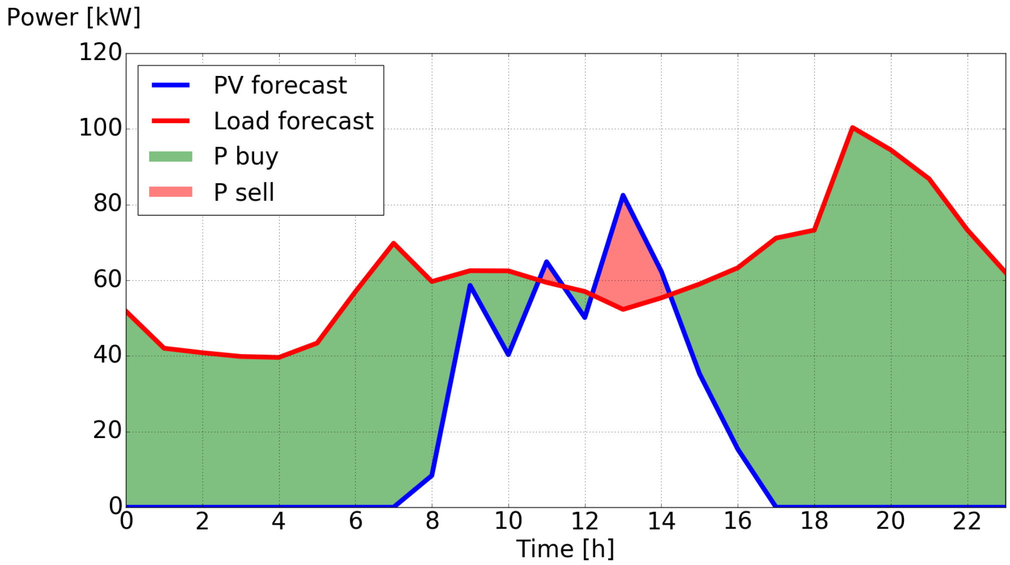

Figure 5.1: Day-ahead naive model strategy

1. Obtain the scheduled $P_{t+1}^{buy}, P_{t+1}^{sell}$ from the Day-ahead comparison model

2. Obtain the forcasted load and solar production for $t+1$: $PV_{t+1}^{forecast}, D_{t+1}^{forecast}$ where $PV_{t+1}^{forecast} = PV_t$ and $D_{t+1}^{forecast} = D_t$

3. Evaluate $\gamma = D_{t+1}^{forecast} + P_{t+1}^{sell} - PV_{t+1}^{forecast} - P_{t+1}^{buy}$

If $\gamma > 0$:

If $e_t^{level} - \frac{\gamma}{\eta} \geq E^{min}$:

$$p_{t+1}^{ch} = 0$$

$$p_{t+1}^{disch} = min(P_{disch}^{max}, \gamma)$$

$$e_{t+1}^{level} = e_t^{level} - \frac{p_{t+1}^{disch}}{\eta}$$

else :

$$p_{t+1}^{ch} = 0$$

$$p_{t+1}^{disch} = min(P_{disch}^{max}, (e_t^{level} - E^{min}) \cdot \eta)$$

$$e_{t+1}^{level} = E^{min}$$

If $\gamma < 0$:

If $e_t^{level} + |\gamma| \cdot \eta \geq E^{max}$:

$$p_{t+1}^{ch} = min(P_{ch}^{max}, |\gamma|)$$

$$p_{t+1}^{disch} = 0$$

$$e_{t+1}^{level} = e_t^{level} + p_{t+1}^{ch} \cdot \eta$$

else :

$$p_{t+1}^{ch} = min(P_{ch}^{max}, \frac{E^{max} - e_t^{level}}{\eta})$$

$$p_{t+1}^{disch} = 0$$

$$e_{t+1}^{level} = E^{max}$$

4. Move to the next period $t+1$ and show the real load and solar production:$PV_{t+1} - D_{t+1}$
5. Implement the strategy $t+1$ of the storage system: $p_{t+1}^{ch}, p_{t+1}^{disch}$
6. Obtain the actual $\Delta p_{t+1}^{sell,B}, \Delta p_{t+1}^{buy,B}$ for the power balance
7. Repeat the algorithm from step 1

In step 2 the idea of persistence model is adopted: the retailer foresees for the next time step the same solar production and energy demand of the current time step. Knowing the bid in the day-ahead market for $t+1$ (P_t^{buy} and P_t^{sell}) it is possible to evaluate γ which meaning is the unbalance foresight for the next hour. The storage stretegy for $t+1$ is set according to γ and in step 3 it is shown: the retailer uses the battery to restore the balance respecting its operational limits. Once moved to the next time step, the real load and production are shown and the storage strategy is implemented. Finally it is possible to evaluate the effective partecipation in the balancing market to balance the power flow in the following way:

$$\Delta p_{t+1}^{sell,B} = max(PV_{t+1} + P_{t+1}^{buy} + P_{t+1}^{disch} - D_{t+1} - P_{t+1}^{sell} - P_{t+1}^{ch}, 0) \tag{5.1}$$

$$\Delta p_{t+1}^{buy,B} = max(D_{t+1} + P_{t+1}^{sell} + P_{t+1}^{ch} - PV_{t+1} - P_{t+1}^{buy} - P_{t+1}^{disch}, 0) \tag{5.2}$$

6

Data analysis

In the stochastic model previously presented an important role is played by the profile of each stochastic parameter within the time frame considered. The uncertain parameters are:

- the solar production: PV
- the price of electricity: C
- the energy demand: D

Stochastic programming deals the uncertainty considering for each stochastic parameter a set of scenarios with a probability of realization associated. Letters *S,R,U* have been used to indicate solar production,electricity price and energy demand respectively. $S = [s_1, s_2, ..., s_k]$ is the set of solar production scenarios, $R = [r_1, r_2, ..., r_k]$ is the set of electricity price scenarios and finally $U = [u_1, u_2, ..., u_k]$ is the set of energy demand scenarios.
From the data available it has been important to realize a data analysis in order to create the scenarios mentioned. All data available are hourly aggregated and they concern:

- the production of a photovoltaic power plant
- the price of electricity in the italian market (PUN)
- the consumption of a domestic load ($P \leq 3kW$)

6.1 Data sources

PV

Concerning the solar production an on-line platform called *Renewables Ninja* [23] has been used. This web portal allows the user to run simulations of the hourly power output from solar power plants. The input used for the simulation are:

- Peak power: $400kW$
- Latitude: 45.536 ; Longitude: 11.564 (Vicenza)
- System loss: 10 %
- Tilt angle: 35°
- Azimuth angle: perfectly south oriented

The simulation has been run for one year and for each day the daily profile has been evaluated and plotted. In figure 6.1 the profile of a representative day for each season has been reported, while in figure 6.2 the solar production profile for five random days of January can be seen.

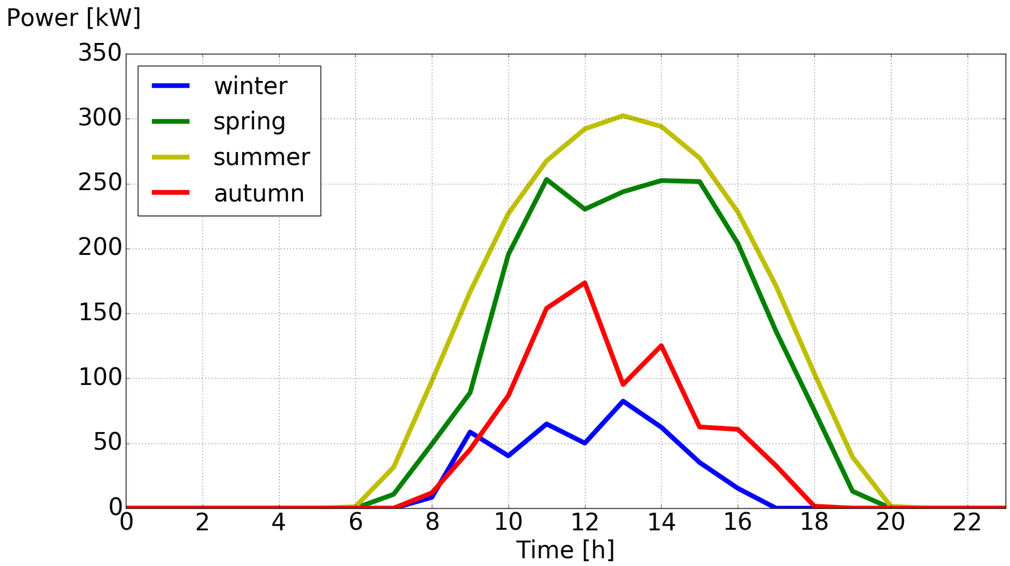

Figure 6.1: Solar production in 4 representative days of the year

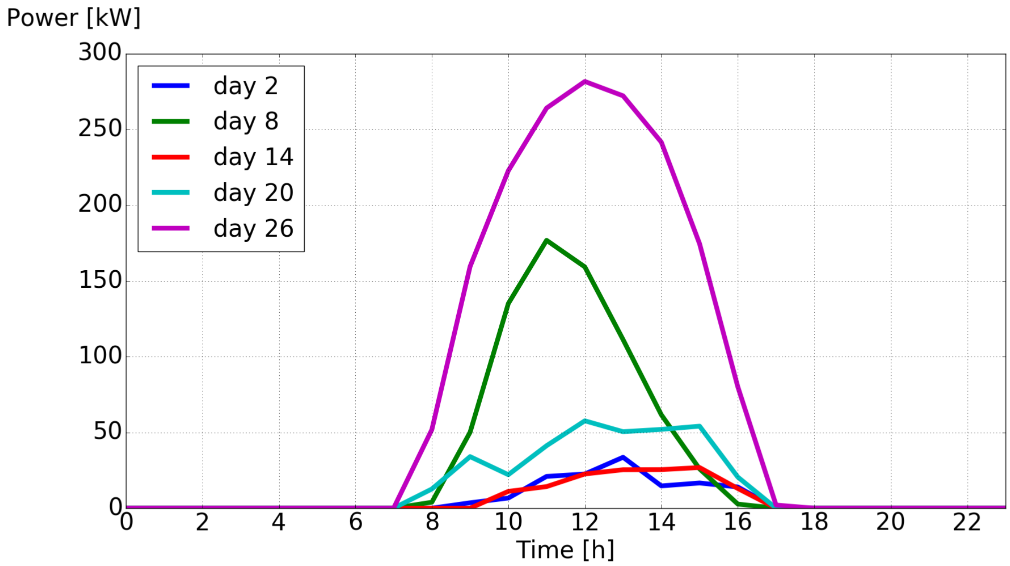

Figure 6.2: Solar production in five random days of January

For the completeness of the analysis in figure 6.3, on the left side, the average daily solar production of January is reported, while on the right side the variance of the data belonging to the same hour has been shown.
Afterwards the curves obtained have been compared with the curves of maximum production for the same power plant. These curves were calculated by means of PVGIS [24]. Having the global clear-sky irradiance and considering a constant performance ratio ($PR = 0.8$) we obtain:

$$PR = \frac{E_{out}/P_{peak}}{Irr/Istc} \qquad E_{out} = PR \cdot P_{peak} \cdot \frac{Irr}{Istc} \tag{6.1}$$

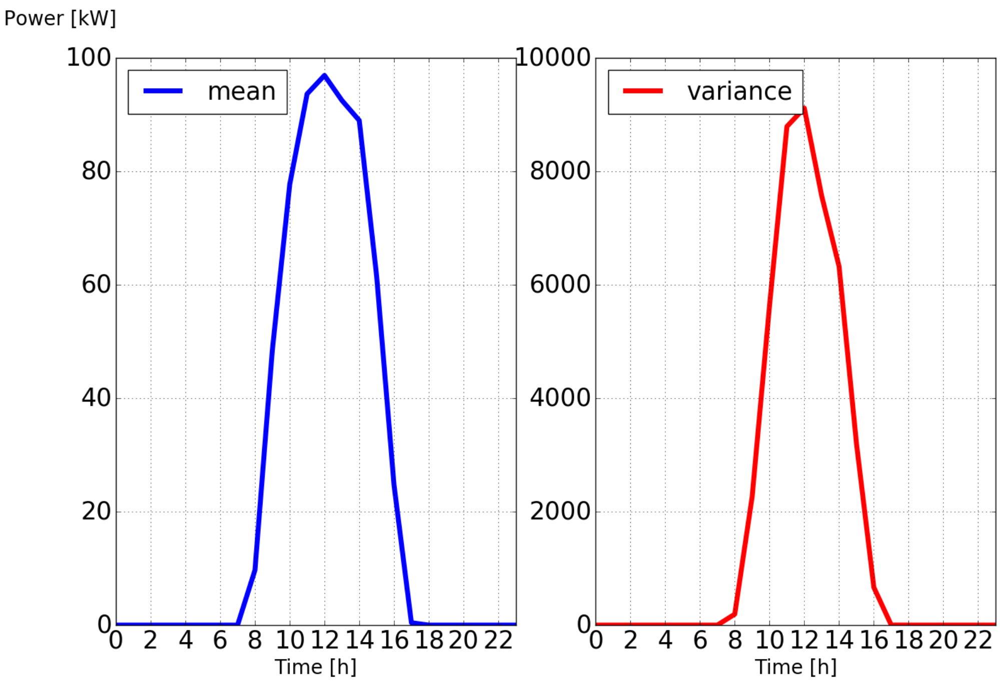

Figure 6.3: Average solar production and variance in January

It's important to notice that PVGIS provides the value of irradiance just for a typical day for each month. Hence, to find the irradiance for every day of the year, the linear interpolation between two consecutive month has been adopted.

In Figure 6.4 is pointed out the difference between the maximum production obtained with PVGIS and real production obtined with *Renewables Ninja* in a random day of the year.

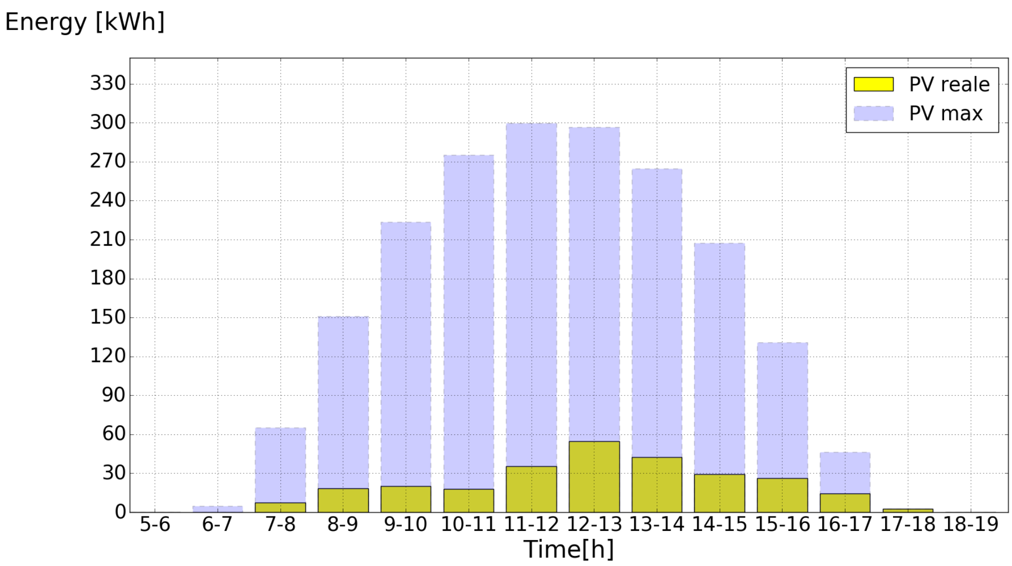

Figure 6.4: Comparison between real production evaluated through *Renewables Ninja* and maximum production evaluated through PVGIS

Electricity price

Concerning the price of electricity, the data of PUN 2013 (*Prezzo Unico Nazionale*) have been downloaded directly from the GME website (*Gestore del Mercato Elettrico*) [19]. In figure 6.5 the daily profile for four representative days of the year has been reported.

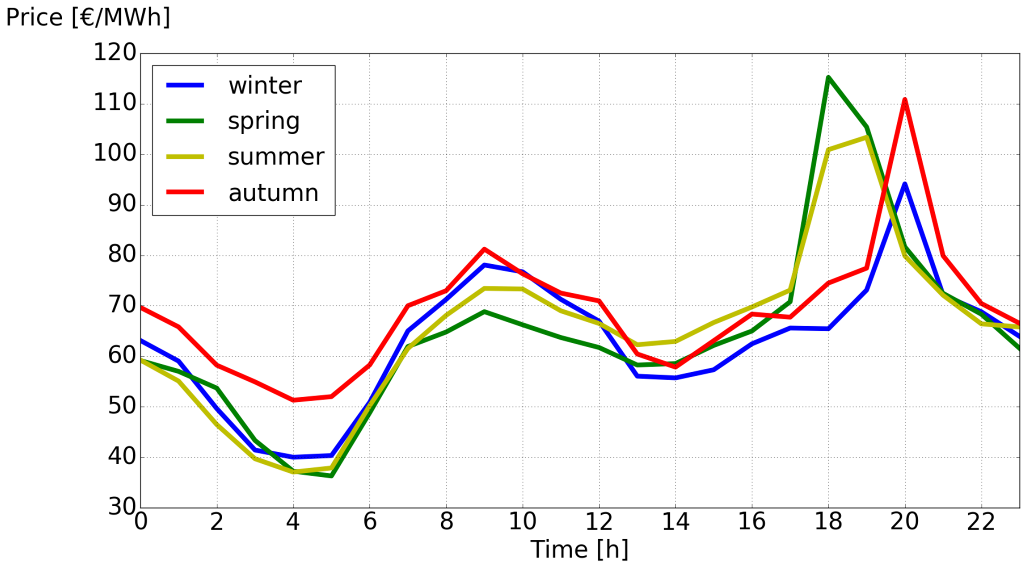

Figure 6.5: Electricity price in 4 representative days of the year

It can be seen how the price is different according to the season. In figure 6.6 the price in two working days of January has been compared with the price during two non-working days of the same month. When the national load is higher (working days) the electricity price is higher.

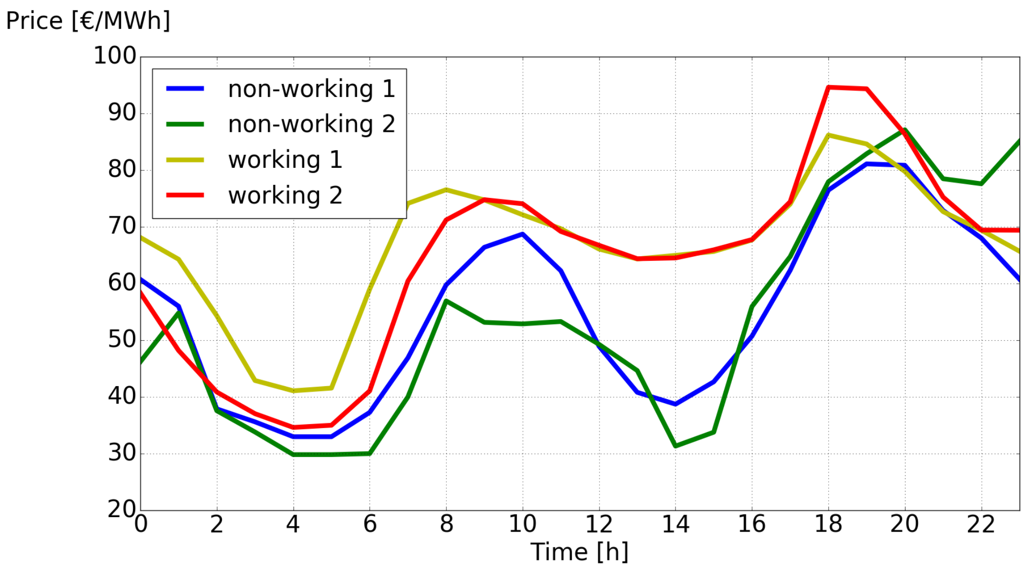

Figure 6.6: Electricity price in two working days and two non-working days

As for the solar production, the averave electricity price in the first month of the year has been evaluate as well as the variance of each hour. (Figure 6.7)

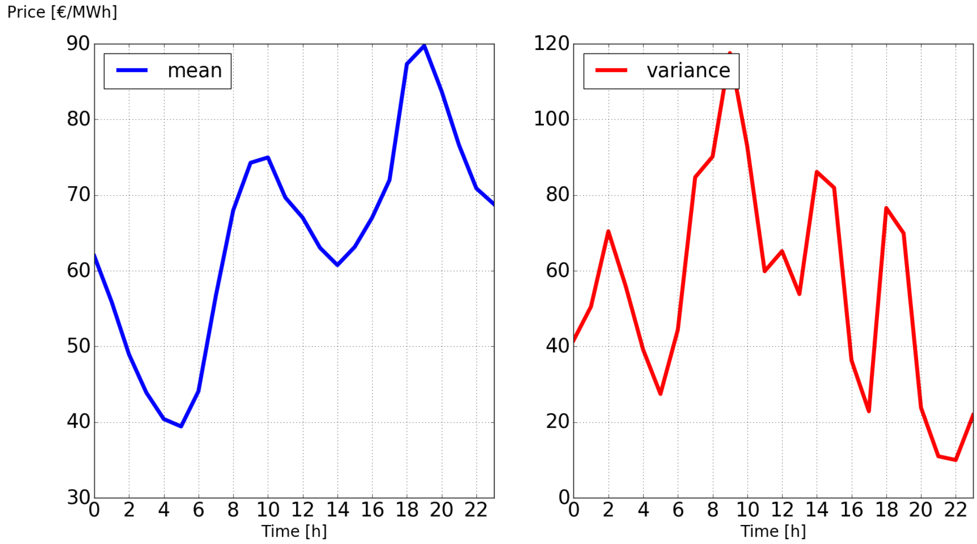

Figure 6.7: Average electricity price and variance in January

Energy demand

Finally, the average energy consumption of a domestic load ($P \leq 3kW$), hourly aggregated for one year, was provided by AIM Vicenza Spa [25]. Be P_t the load of a single electricity consumption at time t, the energy consumption of n load at time t can be evaluate as:

$$P_{n,t} = n \cdot P_t \tag{6.2}$$

In figure 6.8 the daily profile of a representative day for each season has been reported. The daily profile is different according to the season but even more significant is the figure

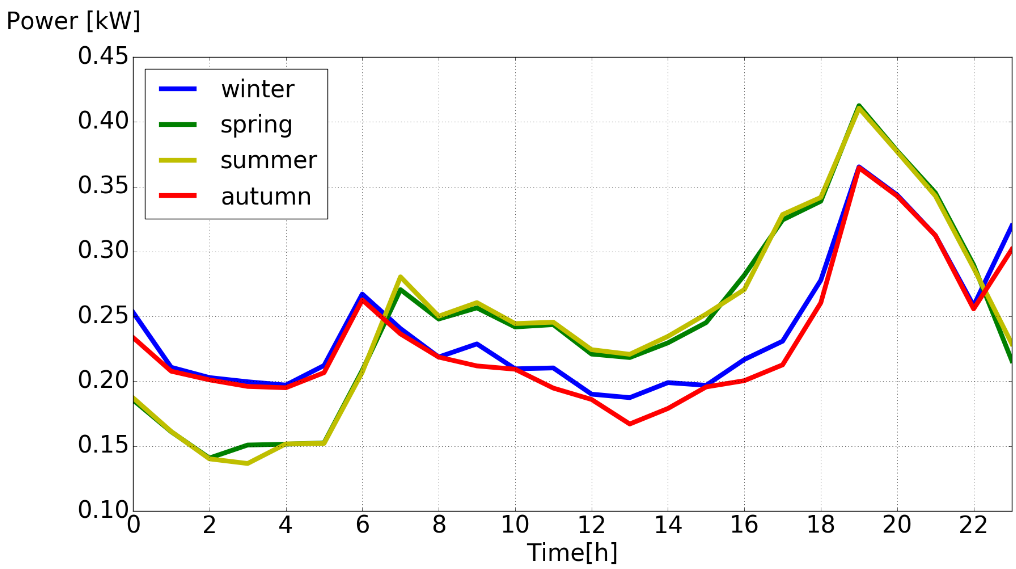

Figure 6.8: Power consumption in 4 representative days of the year

6.9 in which the energy consumption in two working days and two non-working days of the same month (January) has been compared. It can be easily seen that in a non working-days the a domestic load in much higher.

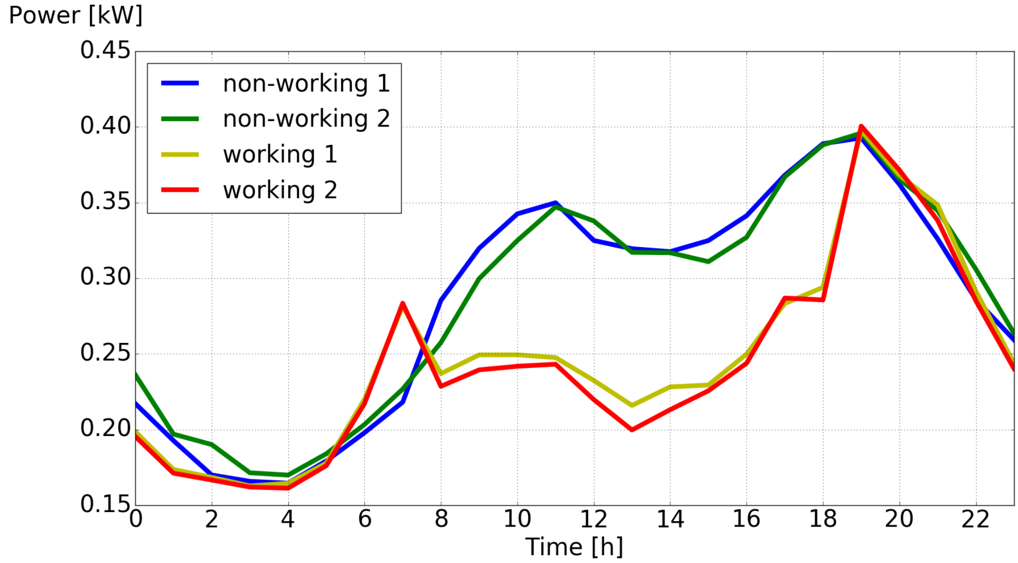

Figure 6.9: Power consumption in two working days and two non-working days

6.2 Creation of scenarios

In this section the method adopted for the creation of scenarios will be described. Starting from a set Γ made of n curves, a clustering has been performed aiming at obtaining k representative curves of Γ, each of them with a probability of realization associated. The algorithm adopted is called *k-means* [26] and it is here presented.

Inizialization: Set a number f of parameters that characterize each curve of Γ, hence each curve can be identified by a point in fD-space

1. k curves are randomly elected from Γ as initial representative of k clusters. The f parameters of each representative curve define the "centroid" of the cluster.

2. The euclidean distance of each curve of Γ from each of these six centroids is evaluated.

3. All curves in Γ are divided and grouped in the k clusters according to the centroid they are most close to.

4. New centroids are evaluated for each cluster as the average of the parameters of the curves grouped to the same cluster. Afterwards, the curve with the closest distance to this new centroid becomes the new representative curve of the cluster.

5. Having k new centroids and k new representative curves, the process starts again from point 2 until the composition of the clusters is not changing during two consecutive iterations.

At the end of the process the representative curve for each gruop is evaluated as the average of all the profiles belonging to that group. In this way k curves of Γ are elected. The probability of realization of the k profiles is evaluated as the ratio between the numbers of curves associated the cluster and the total number of curves belonging to Γ.

PV

The realistic daily profiles of solar production, obtained with the simulation, have been normalized in respect to the maximum production evaluated with PVGIS for the same day. Therefore for each day of the year an α curve has been obtained in the following way:

$$\alpha_t = \frac{PV_t^{max} - PV_t^{real}}{PV_t^{max}}; \forall t \in [0, 1, .., 23] \tag{6.3}$$

The 365 α-profiles have been divided into four groups according to the metereological seasons:

- winter: from 1 December to 28 February ($n = 90$)
- spring: from 1 March to 31 May ($n = 92$)
- summer: from 1 June to 31 August ($n = 92$)
- autumn: from 1 September to 30 November ($n = 91$)

For each group, 8 representative curves $[\alpha_{t,s_1}, \alpha_{t,s_2}, ..., \alpha_{t,s_{k=8}}]$ have been selected through the *k-means* algorithm. The initialization of the algorithm has been done considering 24 paramteres for each curve (i.e. the values in each hour). Therefore, for each day, knowing its maximum production and the α-curves, it is possible to get the 8 scenarios of solar production in the following way:

$$PV_{t,s} = PV_t^{max} - \alpha_{t,s} \cdot PV_t^{max}; \forall t \in [0, 1, .., 23], \forall s \in S = [s_1, s_2, ..., s_{k=8}] \tag{6.4}$$

In figure 6.10 the 8 scenarios of possible production refering to a specific winter day (11th January) can be visualized. The legend reports the probability of realization associated to each curve.

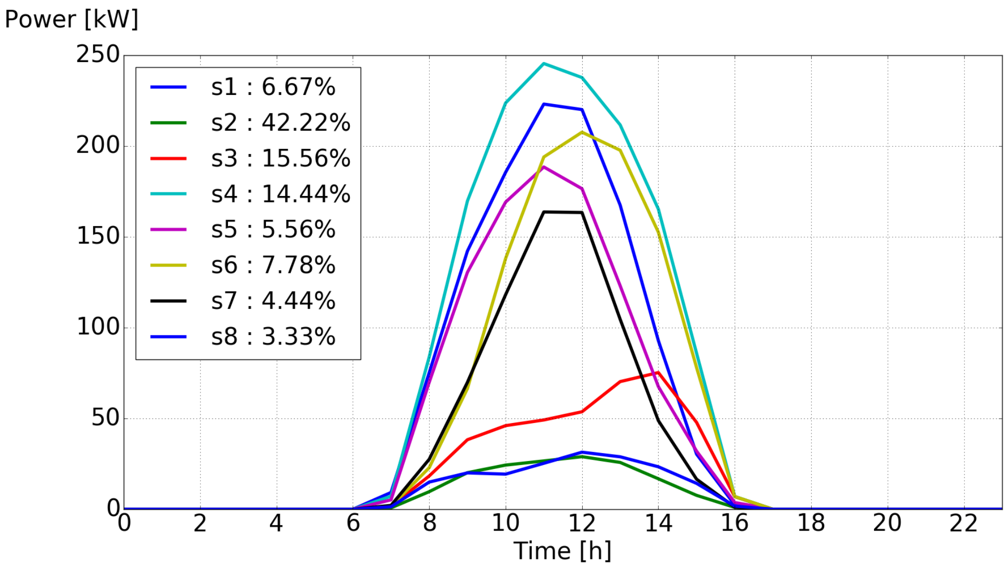

Figure 6.10: Solar production scenarios for a specific winter day (11th January)

Electricity price

Concerning electricity prices, as explained, the profile is different depending on the season and the tipology of the day (working or non-working day). Hence the 365 profiles have been divided in the following way:

working days:

- winter $(n = 62, k = 8)$
- spring $(n = 63, k = 8)$
- summer $(n = 64, k = 8)$
- autumn $(n = 64, k = 8)$

non-working days:

- winter $(n = 28, k = 5)$
- spring $(n = 29, k = 5)$
- summer $(n = 28, k = 5)$
- autumn $(n = 27, k = 5)$

For each of these groups the *k-means* algorithm has been implemented and the result is a set of 5 scenarios $R = [r_1, r_2, ..., r_5]$ if it refers to a non-working day, or a set of 8 scenarios $R = [r_1, r_2, ..., r_8]$ if it rerefers to a working day. In figure 6.11 and 6.12 the scenarios for winter days can be visualized.

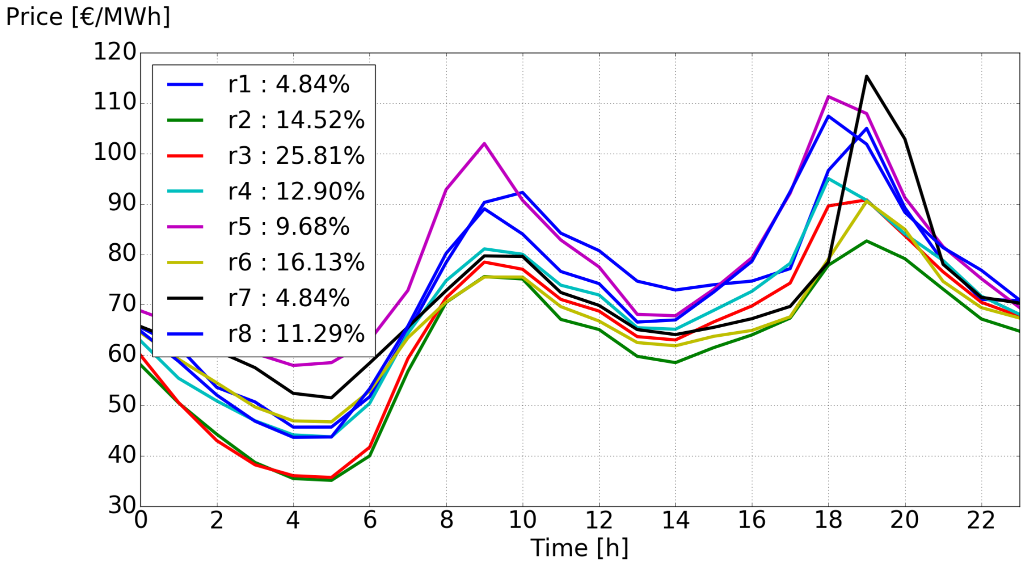

Figure 6.11: Electricity price scenarios for a winter working day

Especially during the night and during the price peaks the scenarios are very different one to another. With the stochastic approach it is important to take into consideration all these possibilities. In figure 6.11 the scenario r_3 is the most likely to be realized with a probability associated of 25%.

Concerning the electricity price scenarios for a winter non-working day, it can be noted that the scenarios r_1, r_3, r_4, r_5 have the same trends, while r_2 has a different profile especially at 8 a.m. and 8 p.m.. Therefore it can reasonably be expected that r_2 has the lowest probability of realization: 3.5%

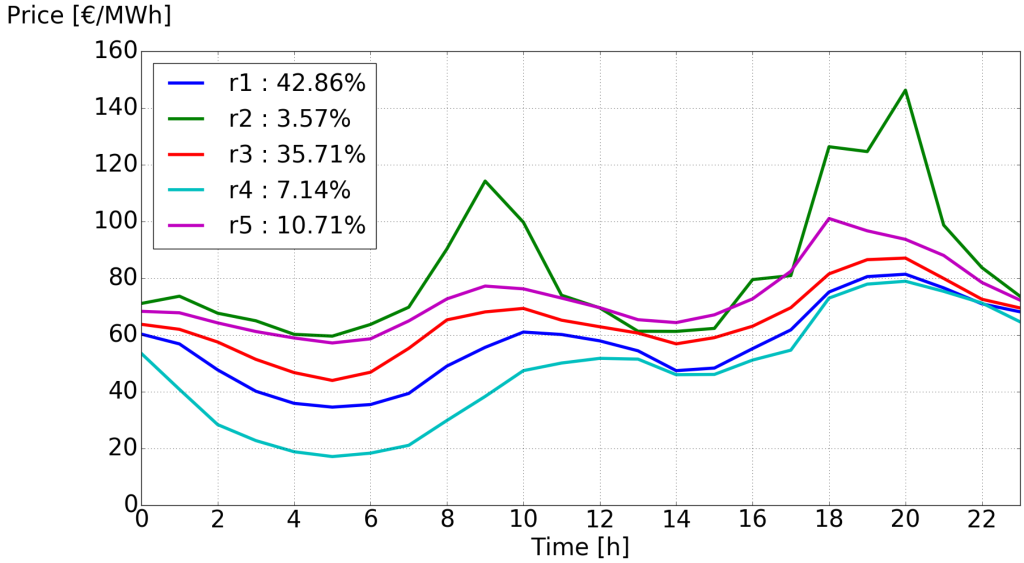

Figure 6.12: Electricity price scenarios for a winter non-working day

Energy demand

As for the electricity price, the 365 load profiles have been divided considering the season and the typoloy of the day. The *k-means algorithm* has been implemented and the set $U = [u_1, u_2, ..., u_k]$ has been evaluated where $k = 5$ if the day is holiday or $k = 8$ if it is working. In figure 6.13 and 6.14 the scenarios for winter days can be visualized.

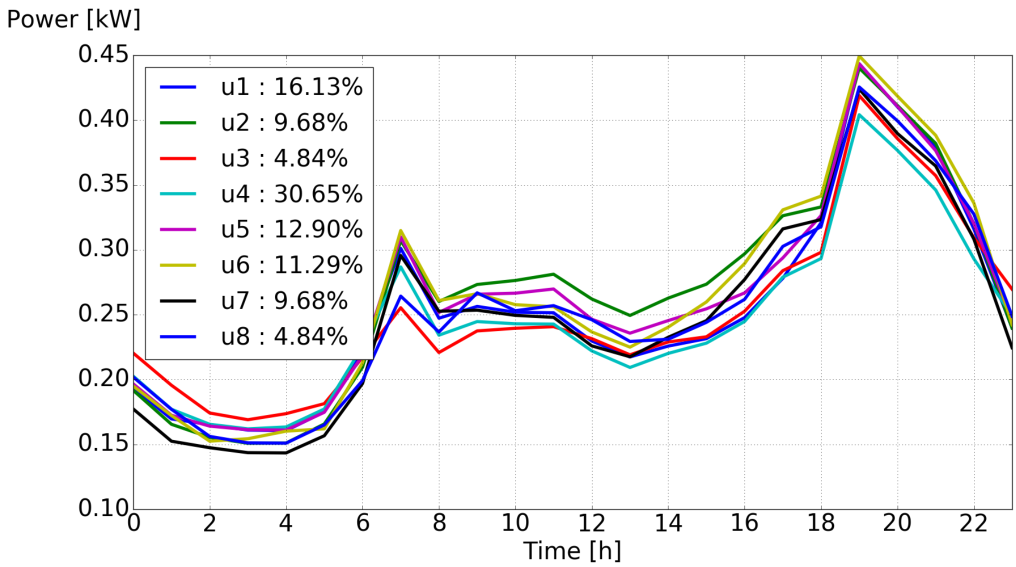

Figure 6.13: Power consumption scenarios for a winter working day

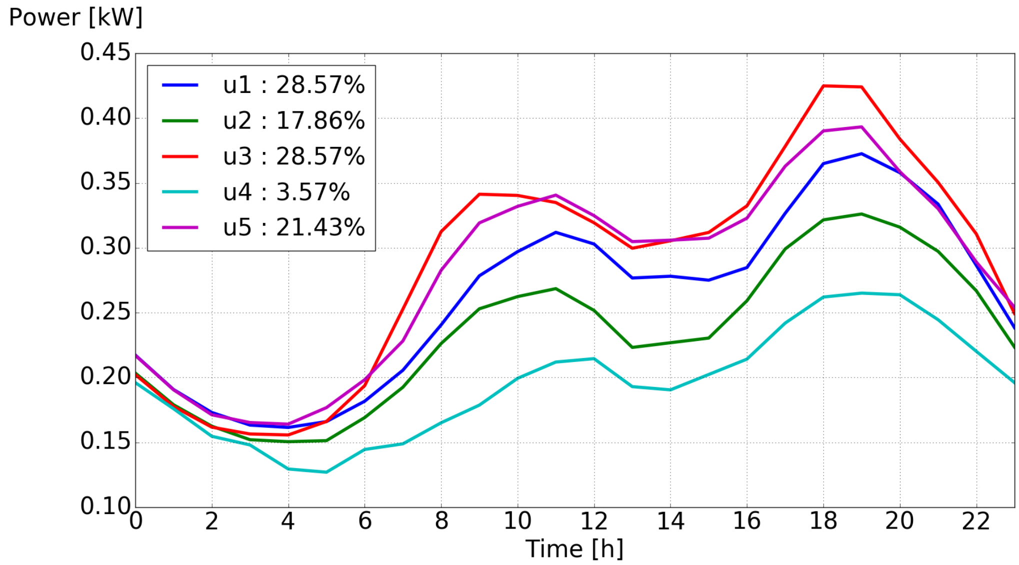

Figure 6.14: Power consumption scenarios for a winter non-working day

For a winter working day, the 8 scenarios created are very similar to one another, while this can not be stated for a winter non-working day. In figure 6.14 the 5 scenarios have the same evolution but the energy demand is very different. There are two daily peaks, one in the morning (7 a.m. if working day, 9 a.m. if holiday) and one in the evening (7 p.m.). These peaks are also reflected in the electricity price profile: high demand, means high price.

Part III

Results and Discussion

7

Model Validation

The two models are implemented in Python and for the optimizaton the solver used is Guorbi. In Appendix it is possible to analyze the main scripts realized.
In this chapter the stochastic model and the naive model will be validated and the results will be compared.

7.1 Stochastic model

Since the data analysis holds an important role in the results of the optimization model, it is first shown the model validation, i.e. it is shown that the model is efficient independently from how the scenarios have been created. This is shown because the main task of this study is to realize a valuable Day-ahead model, therefore minor effort has been focused on the creation of scenarios.

For the sake of simplicity 2 scenarios have been considered for each unknown parameter:

- solar production: $S = [s_1, s_2]$
- electricity price: $R = [r_1, r_2]$
- energy demand: $U = [u_1, u_2]$

and 50% probability of realization has been given to each scenario. The Day-ahead model is run with these input parameters. As already explained, the variables can be divided in two sets:

- *here-and now*: p_t^{sell} and $p_t^{buy}, \forall t \in T$

- *wait-and-see*: $\Delta p_{t,s,r,u}^{sell,B}$ $\Delta p_{t,s,r,u}^{buy,B}, p_{t,s,r,u}^{ch}, p_{t,s,r,u}^{disch}, e_{t,s,r,u}^{level}, \forall t \in T, \forall s \in S, \forall r \in R, \forall u \in U$

The *wait-and-see* variables are scenario dependent. Indicating with $W = [w_1, w_2, ..., w_8]$ the 8 possible scenarios foreseen, there are 8 possible values for each *wait-and-see* variables. The set W is made of the following scenarios:

- $w_1 = s_1, r_1, u_1$
- $w_2 = s_1, r_1, u_2$
- $w_3 = s_1, r_2, u_2$
- $w_4 = s_1, r_2, u_1$

- $w_5 = s_2, r_1, u_1$
- $w_6 = s_2, r_1, u_2$
- $w_7 = s_2, r_2, u_2$
- $w_8 = s_2, r_2, u_1$

and for each $w \in W$ the probability of realization is: $\pi_1 = ... = \pi_8 = 0.5 \cdot 0.5 \cdot 0.5 = 0.125$.

Once the optimization is completed, p_t^{sell} and $p_t^{buy}, \forall t \in T$ are fixed because they are scenario indipendent. On the other hand, from all the possible values of $\Delta p_{t,w}^{sell,B}$, $\Delta p_{t,w}^{buy,B}$, $p_{t,w}^{ch}$, $p_{t,w}^{disch}$, $e_{t,w}^{level}$, $\forall t \in T, \forall w \in W$ one scenario $w \in W$ is randomly selected, therefore also the *wait-and-see* variables are fixed. In this way there is no need to run the Real-time model since one of the possible scenarios belonging to W will be realized. Consequently also the participation in the balancing market is shown.

The model has been run for 365 times and the same scenarios have been adopted for each simulation. This simulation is called *in-sample* since one of the scenarios expected is actually realized.
The simulations have been run for different storage typology and, as expected, the costs are low when bigger capacity is adopted. This is because bigger capacity means higher flexibility therefore more possibilities to manage the energy. Figure 7.1 shows the cumulative operative costs. No investment costs are taken into account. For each time step the operative cost is evaluated in the following way:

$$cost_t = p_t^{buy} C_t^{buy} - p_t^{sell} C_t^{sell} + \Delta p_t^{buy,B} C_t^{buy,B} - \Delta p_t^{sell,B} C_t^{sell,B} \tag{7.1}$$

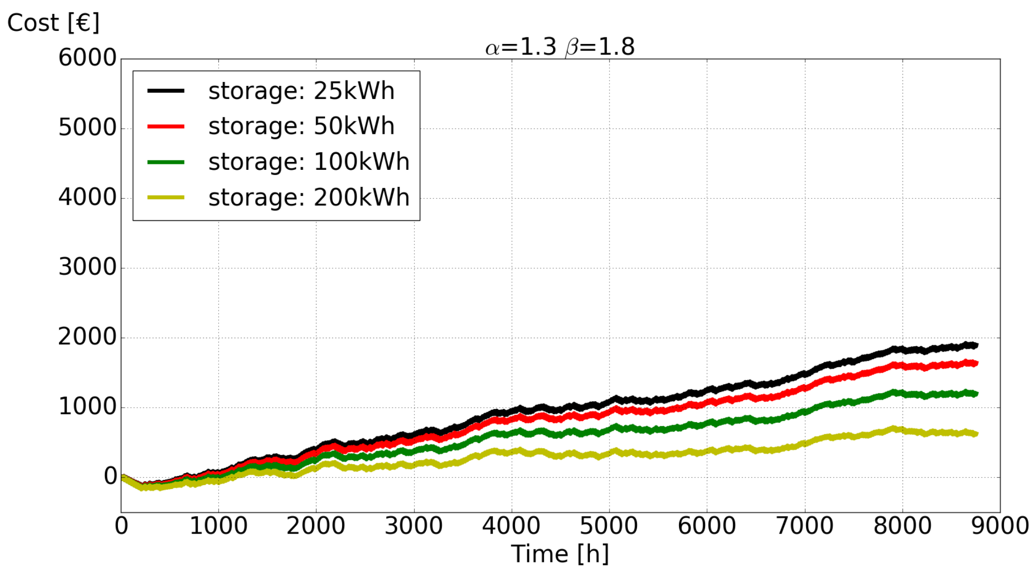

Figure 7.1: *In-sample* simulation of the stochastic model with different storage capacity ($\alpha = 1.3$ and $\beta = 1.8$)

The characteristics of the batteries considered in the simulations are here reported:

- $E^{min} = 5kWh; E^{max} = 25kWh; P^{max}_{ch} = P^{max}_{disch} = 10kWh$
- $E^{min} = 7kWh; E^{max} = 50kWh; P^{max}_{ch} = P^{max}_{disch} = 20kWh$
- $E^{min} = 10kWh; E^{max} = 100kWh; P^{max}_{ch} = P^{max}_{disch} = 40kWh$
- $E^{min} = 20kWh; E^{max} = 200kWh; P^{max}_{ch} = P^{max}_{disch} = 60kWh$

These choices have been done according to the batteries now available on the market [27]. For all the simulations the number of load considered is 250.
To obtain the curves shown in 7.1 the value of α and β mentioned in 4.2 and 4.8 was set in the following way:

- $\alpha = 1.3$
- $\beta = 1.8$

It's reasonable to expect that increasing the value of α and β, the cost for the retailer increases. So, for example, considering $\alpha = 1.5$ and $\beta = 2.0$ the costs are shown in 7.2.

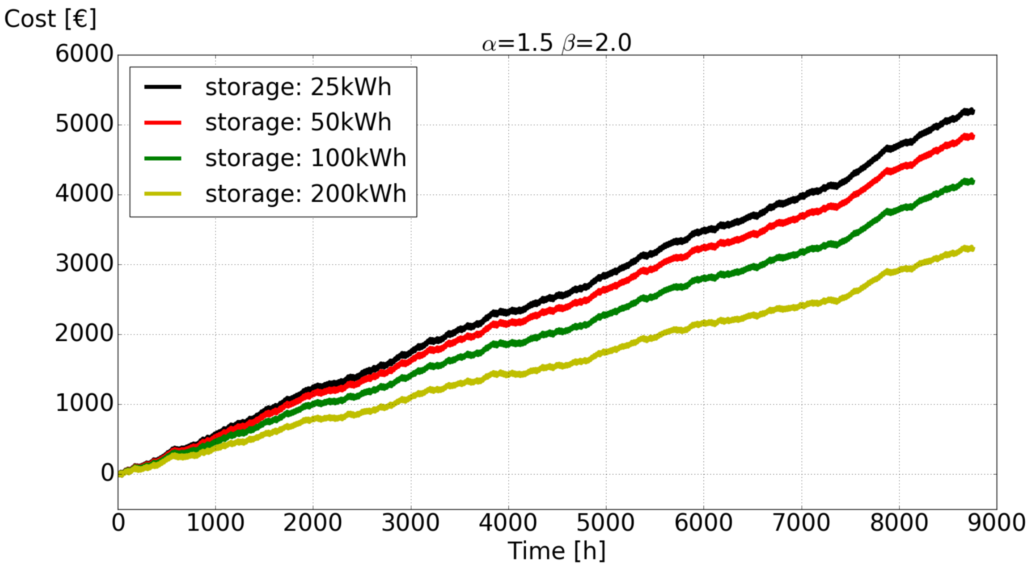

Figure 7.2: *In-sample* simulation of the stochastic model with different storage capacity($\alpha = 1.5$ and $\beta = 2.0$)

Different values of α and β have been tried and in table 7.1 it is possible to compare the final costs.

Table 7.1: Operative costs with the stochastic model. Simulations *in-sample* [€]

Storage [kWh]	$\alpha = 1.3; \beta = 1.8$	$\alpha = 1.5; \beta = 1.8$	$\alpha = 1.5; \beta = 2.0$
25	1905	4484	5197
50	1647	4115	4837
100	1208	3476	4190
200	628	2533	3227

7.2 Naive model

Also the naive model has been run with the *in-sample* mode. The profiles of solar production, consumption and electricity price that can be actually realized after each simulation are the same of those mentioned for the optimization model. The strategy for the power scheduling has already been explained in chapter 5. Concerning the storage strategy, the scheme of chapter 5 is followed, but in this case the value of p_t^{ch} and p_t^{disch} is decided after the solar production and energy demand are revealed; otherwise the comparison with the stochastic model would be unfair.
In figure 7.3 it is possible to see the outcome of the *in-sample* simulations for the naive model, while in table 7.2 the result for different values of α and β is shown. Also in this case, it can be noted that increasing the size of the battery, the operative costs can be reduced; the same typology of battery adopted for the stochastic model have been used for the naive model. Of course, changing the characteristics of the electricity market, the results are different: if the penalty for the unbalance is higher, the final costs are higher. As it will be shown in the next section, the costs reached with the deterministic approach are much higher compared with the final costs of the stochastic approach.

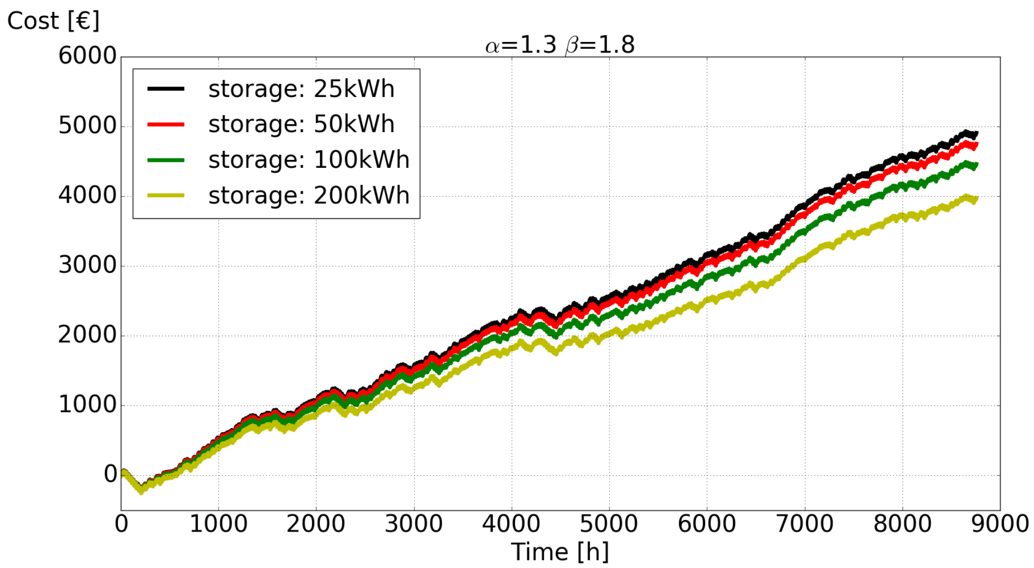

Figure 7.3: *In-sample* simulation of the naive model with different storage capacity ($\alpha = 1.3$ and $\beta = 1.8$)

Table 7.2: Operative costs with the naive model. Simulations *in-sample* [€]

Storage [kWh]	$\alpha = 1.3; \beta = 1.8$	$\alpha = 1.5; \beta = 1.8$	$\alpha = 1.5; \beta = 2.0$
25	4914	8875	11376
50	4758	8658	11117
100	4467	8258	10618
200	3981	7627	9836

7.3 Comparison between the two models

To compare the results of the two models it has been important to build a vector of 365 elements made of the random scenarios realized each simulation. The same vector has been applied to both models.

In the following graphics (7.4 and 7.5) it is shown for different storage typology that the result obtained with the stochastic model is much satisfying. The meaning of Δend is the difference between the final cost of the naive model and the final cost of the stochastic model.

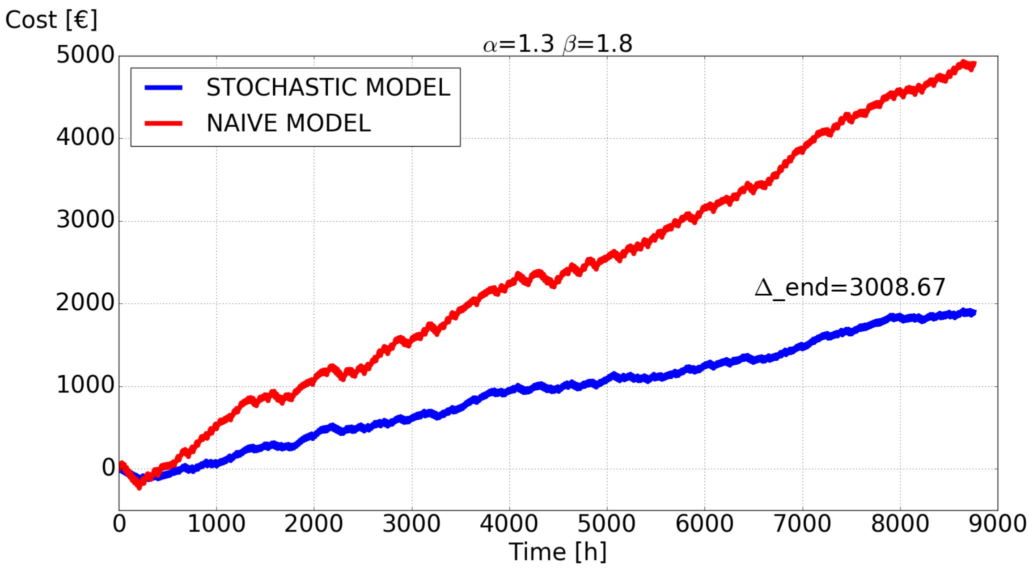

Figure 7.4: Optimization Model vs Comparison Model. Storage: 25 kWh ($\alpha = 1.3$ and $\beta = 1.8$)

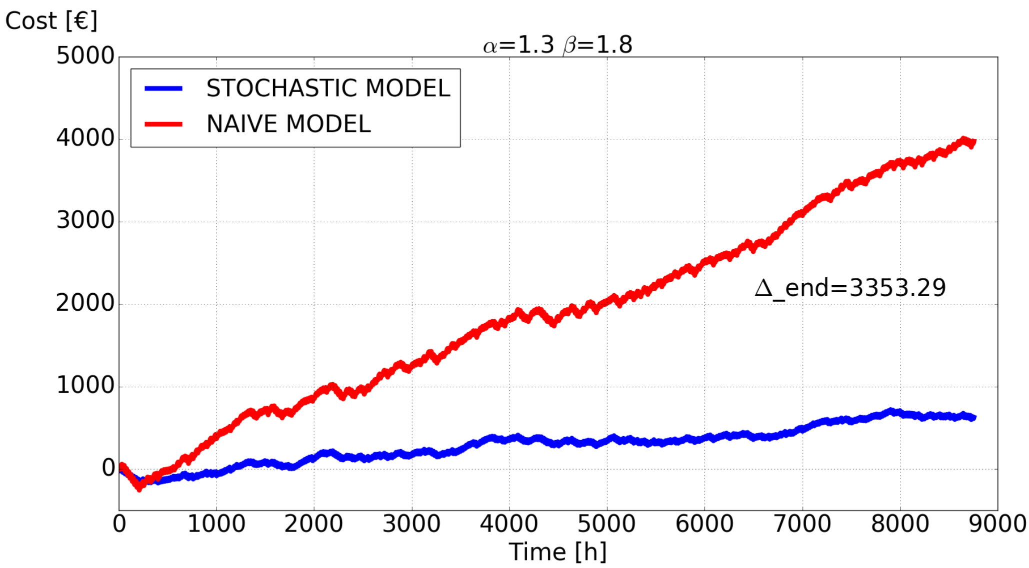

Figure 7.5: Optimization Model vs Comparison Model. Storage: 200 kWh ($\alpha = 1.3$ and $\beta = 1.8$)

It is interesting to notice how the final gap between the two models increases adopting a larger battery. This means that the optimization performs better as the capacity of the storage increases because the retailer can optimize a higher quantity of energy.
The value of α and β affects the gap between the two models as well. Increasing α and β increases the gap. In figure 7.6 and in table 7.3 it is possible to see how the gap changes when α and β are modified.

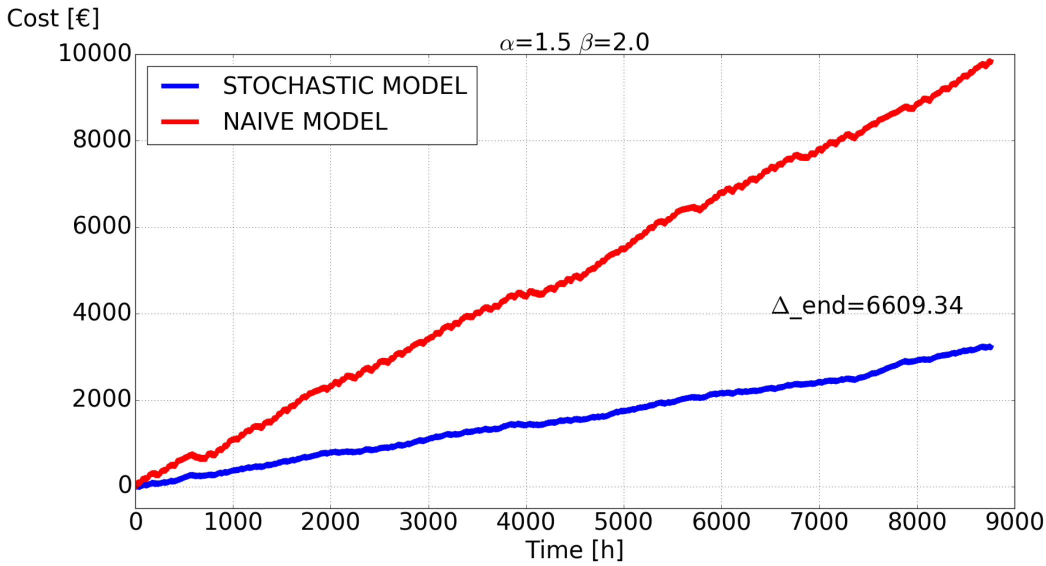

Figure 7.6: Optimization Model vs Comparison Model. Storage: 200 kWh ($\alpha = 1.5$ and $\beta = 2.0$)

Table 7.3: Difference between the final operative costs of the naive model and the final operative costs of the stochastic model [€]

Storage [kWh]	$\alpha = 1.3; \beta = 1.8$	$\alpha = 1.5; \beta = 1.8$	$\alpha = 1.5; \beta = 2.0$
25	3008	4391	6179
50	3110	4542	6279
100	3258	4782	6427
200	3353	5093	6609

Considering the graphics above illustrated, it is clear that the stochastic model performs always much better than the naive model. It is possible to reach a high cost reduction if the power scheduling in the Day-ahead market is properly decided. An optimized participation in the electricity market is much more important when the penalty for any imbalance caused to the main grid (i.e. the meaning of the parameter β) is high. In the same way when a big difference is set between the purchasing price and the selling price (i.e. the meaning of the parameter α) the stochastic optimization results much more convenient.

8
Results

In this chapter the combination of the Day-ahead model and the Real-time model will be shown and the result will be compared with the naive model.
The results are obtained considering the following characteristic of the energy community:

- loads: $n = 250$
- storage: $E^{min} = 10kWh; E^{max} = 100kWh; P_{ch}^{max} = P_{disch}^{max} = 40kWh$
- photovoltaic power plant: $P_p = 400kW$ (see chapter 6.1)

while the parameters that characterize the electricity market have been defined in the following way:

- $\alpha = 1.3$
- $\beta = 1.8$

The stochastic model has been implemented for one year, therefore the Day-ahead model has been run 365 time while the Real-time model has been run for 8760 times. The Real-time model has been implemented with a receding optimization horizon of 10 hours (N=10). Considering the values of electriciy price and energy demand actually realized, the year 2013 has been taken as reference. PUN 2013 has been downloaded from GME website while the energy consumption was provided by AIM Vicenza Spa. For a realistic solar production, the platform *Renewables Ninja* was adopted. This simulation is called *out-of-sample* since the profile actually realized is different from any scenarios forecasted and the time computing is much longer than the *in-sample* simulation. Each Real-time model takes $0.07s$ on average while each Day-ahead model takes $10s$ on average.

The results have been organized like in the table 8.1. In the first column the day of the year and the time step is indicated, while in the other columns the variables and parameters for each hour are reported. The meaning of the color of each column is the direction of the power flow: same color means same direction.

	P_ch	P_disch	P_level	delta_sell	delta_buy	P_sell_sched	P_buy_sched	PV	load	price
67_t6	2,014836806	0	38,09233977	0	13,27726015	0	31,25457665	0	42,517	0,041892
67_t7	0	0	38,09233977	0	47,599	0	0	16,812	64,411	0,054101
67_t8	0	0	38,09233977	0	8,824324628	59,09457463	0	102,002	51,73175	0,067196
67_t9	0	0	38,09233977	23,60023684	0	107,9153882	0	186,43	54,91438	0,07135
67_t10	0	0,1237743	37,93762189	49,87204849	0	147,1011008	0	252,143	55,29363	0,075738
67_t11	0	0	37,93762189	82,62489493	0	153,9737301	0	292,087	55,48838	0,067838
67_t12	0	0	37,93762189	144,879972	0	112,686028	0	307,094	49,528	0,06683
67_t13	0	0	37,93762189	159,1375858	0	86,11441424	0	293,919	48,667	0,060084
67_t14	12,24501804	0	47,73363633	129,2069823	0	78,47912463	0	268,429	48,49788	0,060542
67_t15	10,50345932	0	56,13640378	118,9593498	0	34,68931591	0	216,709	52,55688	0,062583
67_t16	4,231838501	0	59,52187459	84,2340365	0	0	0	141,935	53,46913	0,063436
67_t17	0	0	59,52187459	38,7296261	0	0	40,7393761	54,068	56,07775	0,065152
67_t18	0	0	59,52187459	0	13,97676389	0	52,90423611	0,441	67,322	0,071085
67_t19	0	0	59,52187459	0	22,13879412	0	76,64045588	0	98,77925	0,086298
67_t20	0	6,383966921	51,54191593	0	12,42771165	0	73,23844643	0	92,05013	0,090314
67_t21	0	31,61960814	12,01740577	0	8,495419643	0	43,75047222	0	83,8655	0,073164
67_t22	0	0	12,01740577	0	2,117227941	0	67,64427206	0	69,7615	0,068113
67_t23	0	0,103188502	11,88842014	0,911230168	0	0	50,50516667	0	49,69713	0,066679
68_t0	0	1,510736111	10	0	1,708333333	0	39,64130556	0	42,86038	0,051886
68_t1	0	0	10	0	1,307444445	0	35,82830556	0	37,13575	0,04533
68_t2	0,443815972	0	10,35505278	5,705614583	0	0	37,92443056	0	31,775	0,039338
68_t3	4,172568278	0	13,6931074	2,164946429	0	0	37,52313971	0	31,18563	0,035083
68_t4	6,171354167	0	18,63019073	0,965208333	0	0	38,0813125	0	30,94475	0,030894
68_t5	21,71226158	0	36	0	4,118426293	0	52,55146029	0	34,95763	0,03009
68_t6	34,2756722	0	63,42053776	0	42,45474815	0	30,61204905	0	38,79113	0,03202
68_t7	9,282531678	0	70,8465631	0	49,29165668	0	0	19,118	59,12713	0,041846
68_t8	0	0	70,8465631	0	19,63217312	60,06979812	0	108,267	67,82938	0,057271
68_t9	0	8,677250482	60	16,59784001	0	108,9189105	0	192,382	75,5425	0,064239
68_t10	0	17,78540809	37,76823989	46,37184361	0	148,1205645	0	253,828	77,121	0,065
68_t11	0	11,38775	23,53355239	73,40464086	0	154,9354841	0	292,131	75,17863	0,0599
68_t12	0	0	23,53355239	120,6733864	0	114,6078636	0	307,923	72,64175	0,053
68_t13	24,12741206	0	42,83548204	114,5089319	0	86,79765604	0	292,018	66,584	0,041349
68_t14	24,11549229	0	62,12787587	90,88167386	0	81,03145885	0	265,457	69,42838	0,038126
68_t15	31,45564745	0	87,29239383	80,57728766	0	35,54656489	0	216,275	68,6955	0,040887
68_t16	0	0	87,29239383	69,039375	0	0	0	141,84	72,80063	0,048015
68_t17	0	14,93664912	68,62158242	34,88866133	0	0	40,24476221	55,998	76,29075	0,055497
68_t18	0	15,42278618	49,3430997	0	23,43160271	0	52,90423611	0,64	92,39863	0,067168
68_t19	1,206862132	0	50,30858941	0	19,38915625	0	76,64045588	0	94,82275	0,077779

Figure 8.1: Table of results in an Excel workbook

In the table above, it can be noted that the power flow is always balanced:

$$p_t^{buy} + PV_t + \Delta p_t^{buy,B} + p_t^{disch} = p_t^{sell} + D_t + \Delta p_t^{sell,B} + p_t^{ch}; \forall t \in [t0, t1, ..., t23] \tag{8.1}$$

Moreover, it is important to underline that the optimization never chooses to buy and sell in the same time step. This because different prices have been considered for purchasing and selling the energy from/to the main grid, therefore the objective function is minimized if p_t^{buy} or p_t^{sell} are equal to zero (the same is true for $\Delta p_t^{buy,B}$ and $\Delta p_t^{sell,B}$). Similarly, it never happens that the storage is charged and discharged in the same time step. In this case the reason is the efficiency η that has been considered; since the energy level of the storage plays a significant role in the optimization, it is appropriate that or p_t^{disch} or p_t^{ch} is equal to zero.

Also the comparison model has been run considering the same energy community, the same electricity market and the same realization of solar production, energy demand and prices. In figure 8.2 the cumulative costs are plotted and it is possible to see the difference between the two models. In the same picture it is also possible to observe the costs if a perfect forecast was available and they can be obtained running the day-ahead model presented in chapter 4.1 (deterministic approach) considering perfect profiles of each input parameter.

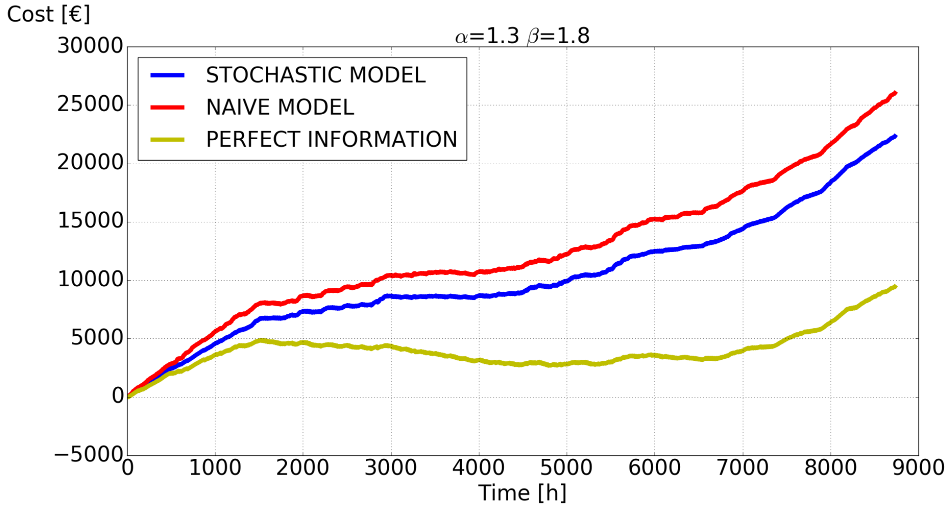

Figure 8.2: Comparison between the stochastic model and the naive model. Simulation *out-of-sample*

As expected the costs achievable with a perfect forecast would be much lower; in this case there is no need to run the Real-time model since the retailer knows in advance how to charge/discharge the battery. When the yellow curve has a negative derivative it means that the aggregator is selling more energy than what it is buying.
The most important result concerns the cost reduction that is possible to reach when a stochastic approach is implemented and in figure 8.3 the savings that can be achieved each day is shown. When they are negative, the naive model is performing better but, on average, the stochastic model can guarantee a savings of 10€each day. To have a better performance of the stochastic optimization model a more accurate scenarios creations should have be realized.

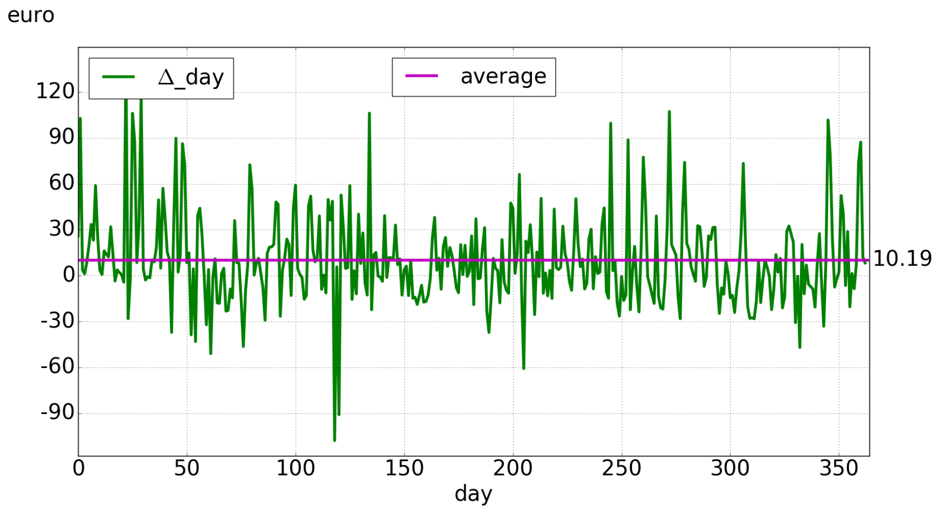

Figure 8.3: Difference between the operative costs of the naive model and the operative costs of the stochastic model at the end of each day

8.1 Electricity market participation

In this section it is possible to analyze how the retailer participates at the electricity market. A summer day has been taken as reference and, in the following graphics, the day-ahead bids and the balancing market participation can be observed for both models. In figure 8.4 it is shown the quantity of energy purchased in the day-ahead market and it is clear that with the stochastic model the retailer buys more energy when the prices are low (during the night). In the evening, when the prices are high, it is expected that the aggregator buys less energy but this is not always happening; therefore this point can be set as a level of criticality. If the forecasting was more accurate, it is clear that this would happen. On the other hand, figure 8.5 shows the energy sold to the main grid in the

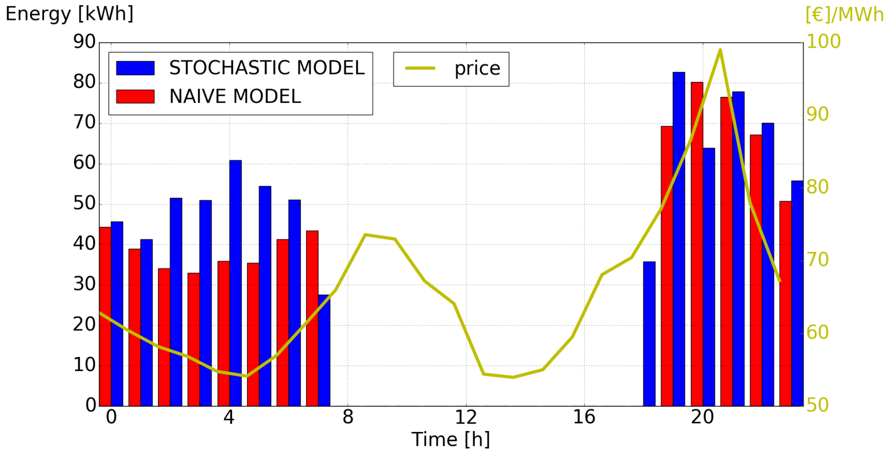

Figure 8.4: Energy purchased in the Day-ahead market (p_t^{buy})

day-ahead market. With the naive model the retailer sells much more energy but, if the solar production is less than the one expected, it has to buy on the balancing market to restore the power flow. This can be seen in figure 8.6 where the naive model shows all its weakness since more energy is purchased real-time compared with the stochastic model. This is what, most of all, makes the stochastic model performing better.

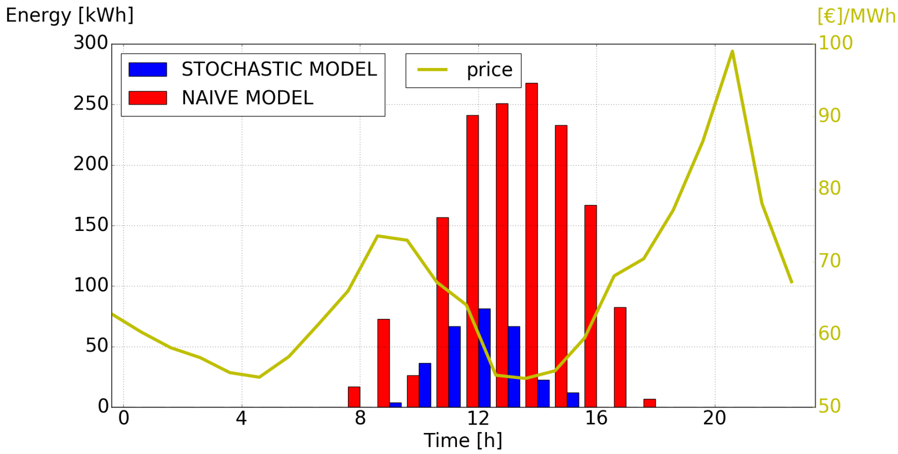

Figure 8.5: Energy sold in the Day-ahead market (p_t^{sell})

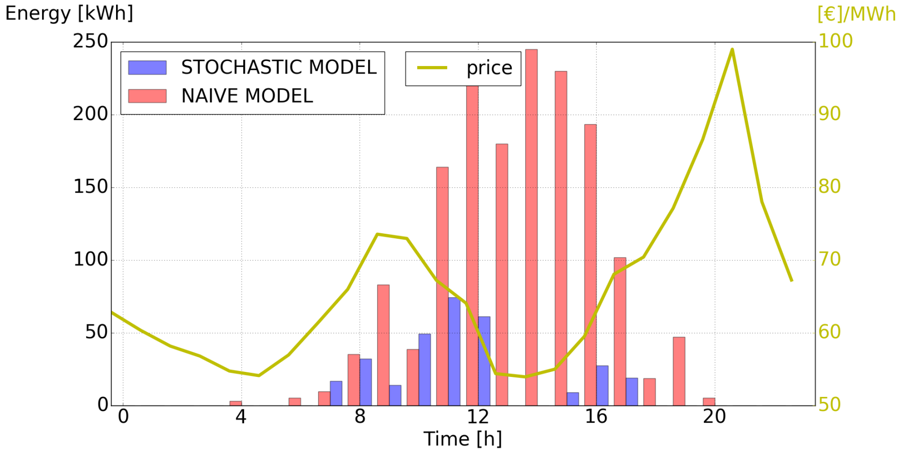

Figure 8.6: Energy purchased in the Balancing market ($\Delta p_t^{buy,B}$)

8.2 Storage strategy

In this section the charging/discharging operations of the storage system will be compared for two consecutive summer days. Figures 8.8 and 8.9 show that the strategy adopted with the stochastic model is more convenient since the state of charge, indicated on the right y-axis, is always within the maximum and minimum level. On the other hand, the battery of the naive model is often full discharged; therefore the retailer must turn to the balancing market to buy energy, because it can not further discharge the storage. This means that adopting a stochastic approach the storage capacity can be better managed and this gives higher flexibility to the retailer. Moreover it is clear that an accurate management of the battery is linked both to the solar production and the electricity price profile (figure 8.7). This can be seen in figure 8.8 where, from 3 a.m. to 7 a.m. when the price is low, the battery is charged while, at 8 p.m. when the price is high, the battery is discharged. This does not happen with the naive model because also when the price is low the storage is discharged.

It can be noted that the stochastic model does not always take the best decision, since it has not perfect information. In the case considered, at 8 p.m. when the price is high the energy discharged is $p_t^{disch} \leq P_{disch}^{max}$ but this is because it is preferable to have a safe margin for the following hours. Of course the storage strategy can be improved through a more accurate forecasting. Moreover, also the receding optimization horizon (N time steps) implemented in the Real-time model plays an important role in these kind of decisions; increasing the number of time steps the retailer can set the value of p_t^{ch} and p_t^{disch} in a different and probably better way. The parameter γ mentioned in 4.11a is significant as well; this simulation has been run setting $\gamma = 0.2$. Increasing this value, the optimization tends to keep the state of charge higher, while choosing γ smaller, less energy would be in the storage at the end of the optimization horizon.

All these aspects have to be taken into consideration to properly operate the storage system. Since the aim of this thesis is to present an efficient model for a smart retailer that has to take part to the day-ahead market, only one simulation has been done. Changing the values of the parameters mentioned above the results of the stochastic model may be different but it will always perform better than the naive model.

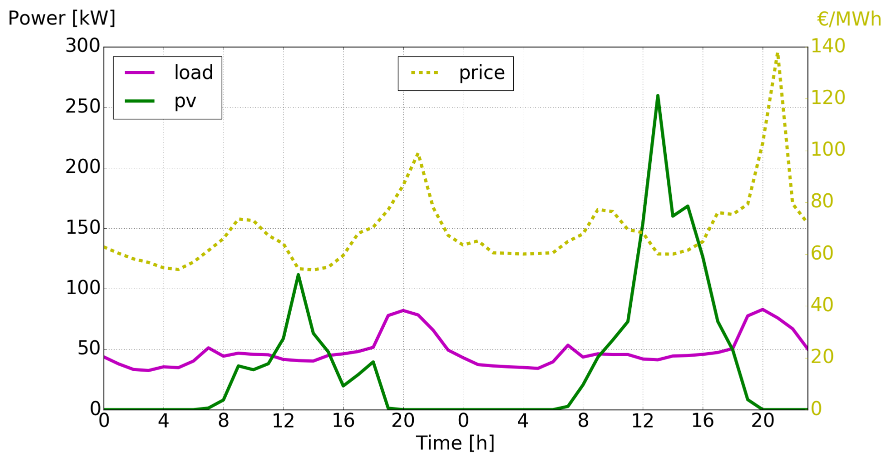

Figure 8.7: PV load and price in two consecutive summer days

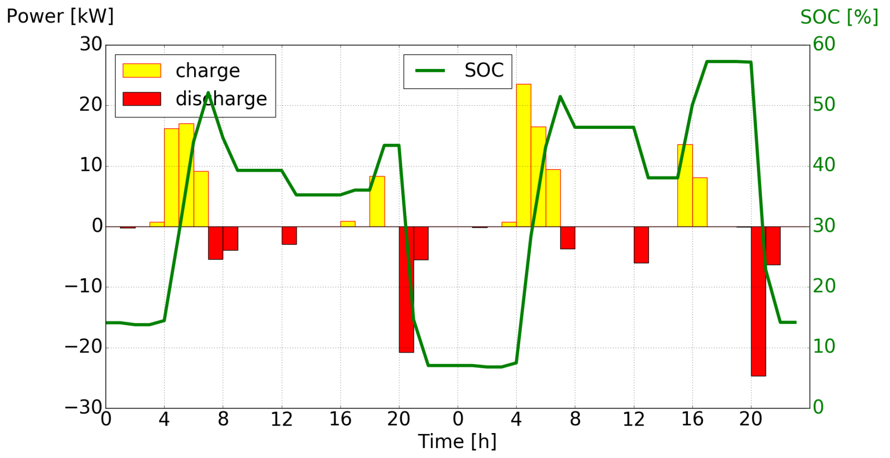

Figure 8.8: Storage strategy with stochastic model in two consecutive summer days

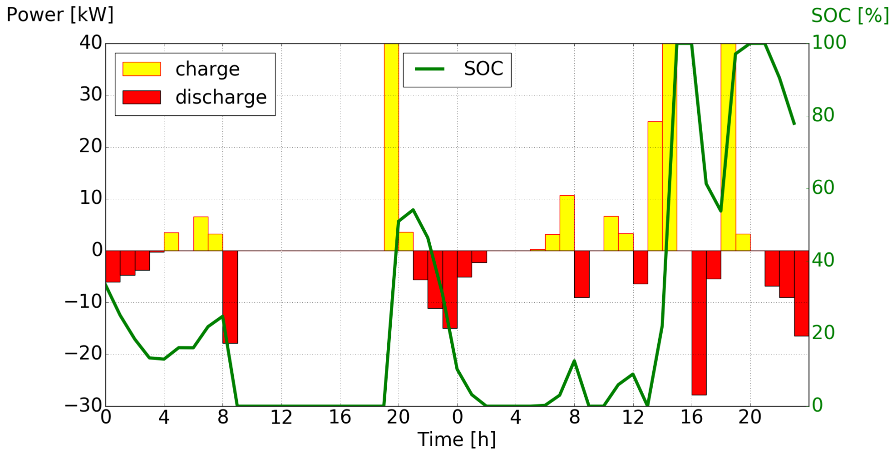

Figure 8.9: Storage strategy with naive model in two consecutive summer days

8.3 Lower bound

The last step that has been considered concerns the Real-time model. When it was previously presented, a restrictive assumption has been made: to decide the storage strategy to implement for the whole following hour. This is clearly a simplification since the management of the storage can be decided for a period much shorter (for instance 30 minutes, or 15 minutes), but for computational reasons time step of one hour have been adopted.
This section wants to show the hypothetical costs that would have reached if a perfect foresight of solar production and energy demand was available for the following hour. These costs are represented with a dotted line in figure 8.10. Even in this case, the

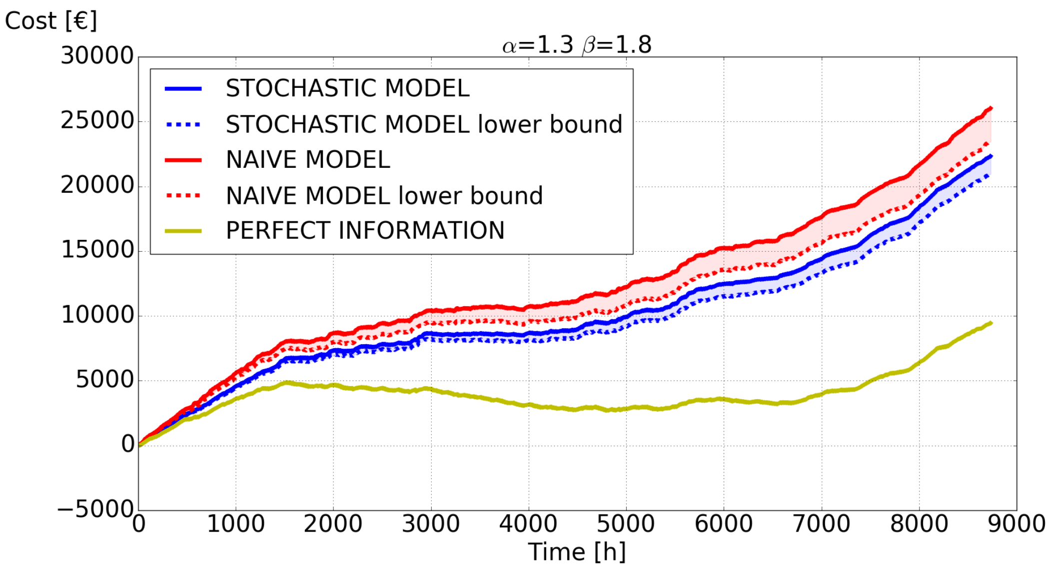

Figure 8.10: Comparison between the two models considering the lower bound

stochastic model shows its advantages.
Another interpretation that can be given to these lines is the lower bound of the costs for the retailer. If the storage strategy is set for a period shorter than one hour, the costs fall within the highlighted region. It results that to reduce the operative costs is convenient to choose how to charge/discharge the battery for a period as short as possible.

8.4 Revenues

The last section of this chapter takes a look to the revenues of the retailer. Since the aim of the aggregator is providing electric energy to its loads, the revenues come from selling electricity to them. It is supposed that a Time-of-use pricing is applied, with two different rates:

- peak (F1): working days from monday to friday from 8.00 a.m. to 7.00 p.m.;
- off-peak (F2): non-working days and working days from 7.00 p.m. to 8.00 a.m.

In F1 0.07€/kWh has been applied, while 0.05€/kWh in F2.
In figure 8.11 the revenues obtained with the two different models are presented. Obviously the curve representing the stochastic model shows higher revenues. In can be noted that

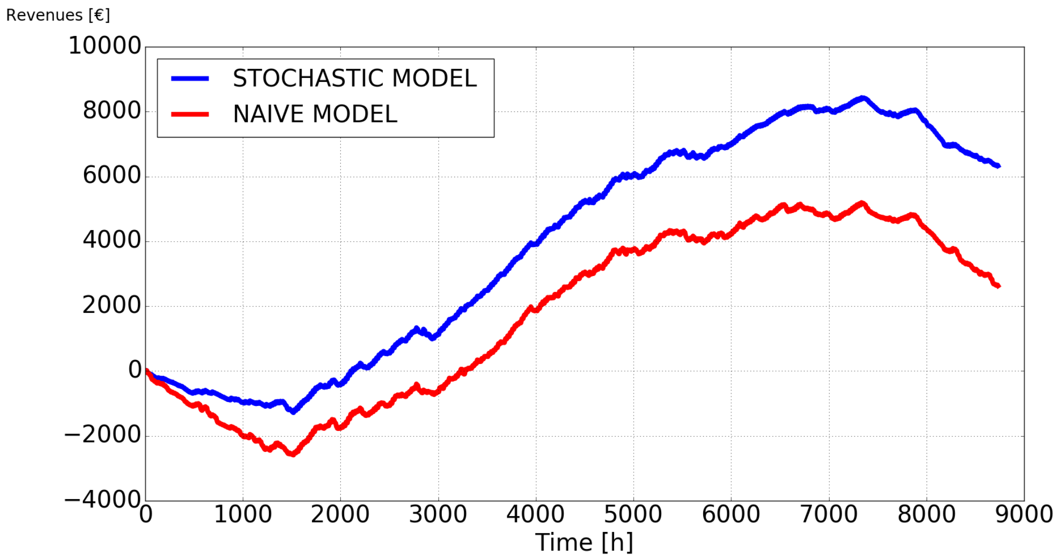

Figure 8.11: Comparison between the revenues of the retailer with the stochastic model and the naive model

in the first and in the last part of the year the derivative is negative, this because in that period (winter and autumn) less solar production is available and much more energy must be purchased in the electricity market.

9 Conclusions

This work draws on a background in which small renewable power plants are growing in number and they have to be integrated in the current power system. It is realistic that in the future also the owner of these power plants will be responsible for the energy injected in the main grid, therefore he will have to face the issue of the unpredictable nature of renewable sources.
This thesis presents a model for a retailer that has to take smart decisions pertaining to its participation in an electricity market. The retailer is seen from the main grid as a *prosumer* since it operates an energy community grid-connected formed of:

- loads;
- photovoltaic power plants;
- storage system.

The aim of the retailer is to provide electricity to the loads, minimizing the operative costs, while no investment costs have been considered. To achieve this goal, a model has been realized taking into account the stochasticity of each unknown parameter: energy demand, solar production, electricity price.

The results show that if an optimization strategy is implemented, it is possible to operate an energy community obtaining a significant cost reduction. Realizing a data analysis has been essential to run the stochastic model but, finally, this model showed its advantages compared to a deterministic approach. Dealing with uncertain parameters, the naive model, based on a deterministic approach, does not always take the best decisions for the power scheduling while the stochastic model, having different scenarios as input, can guarantee more precise choices. It has been shown how the creation of scenarios plays an important role in the performance of the model, therefore this is certainly a crucial matter that the retailer has to face in order to realize an optimal scheduling.
Also a possible storage strategy has been presented through the Real-time model and it can be observed that following a stochastic approach the charging/discharging decisions are more advisable. The retailer charges the battery when the prices are low and discharges the battery when the prices are high. This strategy is less clear with the naive model. Increasing the storage dimensions means more flexibility therefore a more considerable saving can be reached. All these considerations contribute to answer the research questions expressed in 1.2.

In the particular case considered, the retailer can reduce the operative costs of one year from 26026 € to 22326 € (14% reduction) implementing the stochastic model. This reduction depends on the components of the energy community and their characteristics. Moreover, also the features of the electricity market can modify the results: a stochastic approach is even more preferable if higher penalty are taken into account for any unbalance caused by the retailer.

This thesis showed how it is possible to introduce renewable sources into the electric power system. Optimizing the aggregated loads and the storage system, the energy requirements of a community grid-connected can be properly managed by means of a stochastic approach.

9.1 Further improvement

In this thesis the main focus was on the model realization, but to achieve better results a more accurate forecasting can be implemented. This is the first improvement that can be implemented in order to help the retailer in taking the right decisions. Concerning the model, some features can be introduced to better describe the reality, such as the charging/discharging limits depending on the state of charge and the charge/discharge cycles.
The most important improvement that can be realized refers to a potential *demand response*. In this work all loads have been considered fixed but a strategy that can be implemented to reduce the costs of energy supply is the load shifting. When few solar production or high electricity price are forecast, the retailer can decide to shift some loads (for example 20 % of the total load) to a following period. This ways, the purchasing of expensive electricity or an excessive use of the storage can be avoided.

Bibliography

[1] IRENA. Roadmap for a Renewable Energy Future. Technical report, 2016.

[2] IEA International Energy Agency. `www.iea.org/topics/electricity/`. [Online; accessed 1-October-2016].

[3] EUROSTAT. `www.ec.europa.eu/eurostat`. [Online; accessed 1-October-2016].

[4] K Purchala, R Belmans, K U Leuven, L Exarchakos, and a D Hawkes. Distributed generation and the grid integration issues. *Imperial College London*, page 9.

[5] J.A. Peças Lopes, N. Hatziargyriou, J. Mutale, P. Djapic, and N. Jenkins. Integrating distributed generation into electric power systems: A review of drivers, challenges and opportunities. *Electric Power Systems Research*, 77(9):1189–1203, 2007.

[6] Iliana Shandurkova, Bernt A. Bremdal, Rainer Bacher, Stig Ottesen, and Andreas Nilsen. A Prosumer Oriented Energy Market. Technical report, Halden, 2012.

[7] Anne Sjoerd Brouwer, Machteld van den Broek, Ad Seebregts, and André Faaij. Impacts of large-scale Intermittent Renewable Energy Sources on electricity systems, and how these can be modeled. *Renewable and Sustainable Energy Reviews*, 33:443–466, 2014.

[8] Guishi Wang, Mihai Ciobotaru, and Vassilios G. Agelidis. Power smoothing of large solar PV plant using hybrid energy storage. *IEEE Transactions on Sustainable Energy*, 5(3):834–842, 2014.

[9] R. Bayindir, I. Colak, G. Fulli, and K. Demirtas. Smart grid technologies and applications. *Renewable and Sustainable Energy Reviews*, 66:499–516, 2016.

[10] Hassan Farhangi. The path of the smart grid. *IEEE Power and Energy Magazine*, 8(1):18–28, 2010.

[11] M.S. Mahmoud, S. Azher Hussain, and M.A. Abido. Modeling and control of microgrid: An overview. *Journal of the Franklin Institute*, 351(5):2822–2859, 2014.

[12] Aqeel Ahmed Bazmi and Gholamreza Zahedi. Sustainable energy systems: Role of optimization modeling techniques in power generation and supply - A review. *Renewable and Sustainable Energy Reviews*, 15(8):3480–3500, 2011.

[13] Juan M. Morales, Antonio J. Conejo, Henrik Madsen, Pierre Pinson, and Marco Zugno. *Integrating Renewables in Electricity Markets*. Springer, New York, 2012.

[14] Antonis G Tsikalakis, St Member, Nikos D Hatziargyriou, and Senior Member. Centralized Control for Optimizing Microgrids Operation. *Power and Energy Society General Meeting, 2011 IEEE*, 23(1):1–8, 2011.

[15] Yuanxiong Guo, Miao Pan, Yuguang Fang, and Pramod P. Khargonekar. Decentralized coordination of energy utilization for residential households in the smart grid. *IEEE Transactions on Smart Grid*, 4(3):1341–1350, 2013.

[16] Luis M. Costa and George Kariniotakis. A Stochastic Dynamic Programming Model for Optimal Use of Local Energy Resources in a Market Environment. *2007 IEEE Lausanne Power Tech*, pages 449–454, 2007.

[17] Eric Sortomme and M A El-Sharkawi. Optimal power flow for a system of microgrids with controllable loads and battery storage BT - 2009 IEEE/PES Power Systems Conference and Exposition, PSCE 2009, March 15, 2009 - March 18, 2009. pages 1–5, 2009.

[18] Yixing Xu, Le Xie, and C. Singh. Optimal scheduling and operation of load aggregator with electric energy storage in power markets. *North American Power Symposium 2010*, pages 1–7, 2010.

[19] GME Gestore del Mercato Elettrico. `www.mercatoelettrico.org/It/`. [Online; accessed 1-October-2016].

[20] Charles M Grinstead and J Laurie Snell. *Introduction to Probability*, volume 23. American Mathematical Society, 2006.

[21] Trine Krogh Kristoffersen. *Stochastic programming with applications to power systems*. PhD thesis, University of Aarhus, Denmark., 2007.

[22] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczynski. *Lectures on Stochastic Programming*. SIAM, 2014.

[23] Renewables Ninja. `www.renewables.ninja/`. [Online; accessed 1-October-2016].

[24] PVGIS Photovoltaic Geographical Information System. `re.jrc.ec.europa.eu/pvgis/`. [Online; accessed 1-October-2016].

[25] AIM Vicenza S.p.A. `www.aimvicenza.it/`. [Online; accessed 1-October-2016].

[26] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining*. 2006 edition, 2005.

[27] Tesla Energy. `www.tesla.com/it_IT/energy`. [Online; accessed 1-October-2016].

Part IV

Appendix

DAY-AHEAD STOCHASTIC MODEL

```
# -*- coding: utf-8 -*-
"""
Created on Mon May 23 16:17:13 2016

@author: paolo
"""
import pandas as pd
import gurobipy as gb
import xlrd
import numpy as np
import natsort as nt


#%% Build Model

class AGG_base(object):
    pass

class AGG_model:
    def __init__(self):
        self.data = AGG_base()
        self.variables = AGG_base()
        self.constraints = AGG_base()
        self._load_data()
        self._build_model()

    def optimize(self):
        self.model.optimize()

    def _load_data(self):
        #Load Sets
        self.data.time_1=time_1
        self.data.time = time
        self.data.scen_pv = scen_pv
        self.data.scen_price = scen_price
        self.data.scen_load = scen_load
        #Load Parameters
        self.data.PV_scen = PV_scen
        self.data.PV_prob = PV_prob
        self.data.Load_scen = Load_scen
        self.data.Load_prob = Load_prob
        self.data.Prices_scen = Prices_scen
        self.data.Prices_prob = Prices_prob
        self.data.eff_batt = eff_batt
        self.data.buy_ratio = buy_ratio
        self.data.E_max_batt = E_max_batt
        self.data.P_max_sell = P_max_sell
        self.data.P_max_buy = P_max_buy
        self.data.P_max_delta_sell=P_max_delta_sell
        self.data.P_max_delta_buy=P_max_delta_buy
        self.data.E_min_batt = E_min_batt
        self.data.P_max_ch = P_max_ch
        self.data.P_max_disch = P_max_disch
        self.data.a = a
        self.data.c_avg = c_avg
        self.data.initial_level=initial_level
        self.data.P_sell_fixed=P_sell_fixed
```

```
        self.data.P_buy_fixed=P_buy_fixed



    def _build_model(self):
        self.model = gb.Model()
        self._build_variables()
        self._build_objective()
        self._build_constraints()

    def _build_variables(self):
        time = self.data.time
        scen_pv = self.data.scen_pv
        scen_price = self.data.scen_price
        m = self.model

        self.variables.P_ch = {} #Charge Battery
        self.variables.P_disch = {} #Discharge Battery
        self.variables.P_level = {} #Level of Battery
        self.variables.P_sell = {} #Sell to the grid day-ahead
        self.variables.P_buy = {} #Buy from the grid day-ahead
        self.variables.delta_sell = {} #Sell to the grid real time
        self.variables.delta_buy = {} #Buy from the grid real time



        for s in scen_pv:
            for r in scen_price:
                for u in scen_load:
                    for t in time:
                        self.variables.P_ch[t,s,r,u]    = m.addVar()
                        self.variables.P_disch[t,s,r,u] = m.addVar()
                        self.variables.P_level[t,s,r,u]   = m.addVar()
                        self.variables.delta_sell[t,s,r,u] = m.addVar()
                        self.variables.delta_buy[t,s,r,u] = m.addVar()

        for t in time:
            self.variables.P_buy[t] = m.addVar()
            self.variables.P_sell[t]  = m.addVar()


        m.update()

    def _build_objective(self):
        scen_pv = self.data.scen_pv
        scen_price = self.data.scen_price
        scen_load = self.data.scen_load
        time=self.data.time
        m = self.model


        m.setObjective(

    gb.quicksum(self.data.PV_prob[s][t]*self.data.Prices_prob[r][t]*self.data.Load_p
    rob[u][t]* \

    ((self.data.Prices_scen[r][t]*self.data.buy_ratio*self.variables.P_buy[t]- \
                            self.data.Prices_scen[r][t] * self.variables.P_sell[t] )
    + \
```

```
                        (self.data.Prices_scen[r][t] * self.data.buy_ratio *
self.data.a * self.variables.delta_buy[t,s,r,u] - \
                        self.data.Prices_scen[r][t] / self.data.a *
self.variables.delta_sell[t,s,r,u]))\
                        for t in time for s in scen_pv for r in scen_price for u
in scen_load)- \

gb.quicksum(self.data.PV_prob[s][time[34]]*self.data.Prices_prob[r][time[34]]*se
lf.data.Load_prob[u][time[34]]*\
                        (self.data.c_avg[r] *
self.variables.P_level[time[34],s,r,u])for s in scen_pv for r in scen_price \
                        for u in scen_load),

            gb.GRB.MINIMIZE)


    def _build_constraints(self):
        scen_pv = self.data.scen_pv
        scen_price = self.data.scen_price
        scen_load = self.data.scen_load
        time=self.data.time
        time_1=self.data.time_1
        m=self.model


        #power balance
        self.constraints.power_balance = {}
        for u in scen_load:
            for r in scen_price:
                for s in scen_pv:
                    for t in time:
                        self.constraints.power_balance[t,s,r,u] = m.addConstr(
                            self.data.Load_scen[u][t],
                            gb.GRB.EQUAL,
                            self.data.PV_scen[s][t] + \
                            self.variables.P_buy[t] - \
                            self.variables.P_sell[t] + \
                            self.variables.P_disch[t,s,r,u] - \
                            self.variables.P_ch[t,s,r,u] + \
                            self.variables.delta_buy[t,s,r,u] -\
                            self.variables.delta_sell[t,s,r,u])

        #maximum battery level
        self.constraints.batt_lev_max = {}
        for u in scen_load:
            for r in scen_price:
                for s in scen_pv:
                    for t in time:
                        self.constraints.batt_lev_max[t,s,r,u] = m.addConstr(
                            self.variables.P_level[t,s,r,u],
                            gb.GRB.LESS_EQUAL,
                            self.data.E_max_batt)

        #minimum battery level
        self.constraints.batt_lev_min = {}
        for u in scen_load:
            for r in scen_price:
                for s in scen_pv:
                    for t in time:
                        self.constraints.batt_lev_min[t,s,r,u] = m.addConstr(
```

```
                    self.variables.P_level[t,s,r,u],
                    gb.GRB.GREATER_EQUAL,
                    self.data.E_min_batt)

#maximum energy sell day-ahead
self.constraints.max_sell = {}
for t in time:
    self.constraints.max_sell[t] = m.addConstr(
        self.variables.P_sell[t],
        gb.GRB.LESS_EQUAL,
        self.data.P_max_sell)

#maximum energy buy day-ahead
self.constraints.max_buy = {}
for t in time:
    self.constraints.max_buy[t] = m.addConstr(
        self.variables.P_buy[t],
        gb.GRB.LESS_EQUAL,
        self.data.P_max_buy)

#maximum energy delta_sell
self.constraints.max_delta_sell = {}
for u in scen_load:
    for r in scen_price:
        for s in scen_pv:
            for t in time:
                self.constraints.max_delta_sell[t,s,r,u] = m.addConstr(
                    self.variables.delta_sell[t,s,r,u],
                    gb.GRB.LESS_EQUAL,
                    self.data.P_max_delta_sell)


#maximum energy delta_buy
self.constraints.max_delta_buy = {}
for u in scen_load:
    for r in scen_price:
        for s in scen_pv:
            for t in time:
                self.constraints.max_delta_buy[t,s,r,u] = m.addConstr(
                    self.variables.delta_buy[t,s,r,u],
                    gb.GRB.LESS_EQUAL,
                    self.data.P_max_delta_buy)






#maximum energy charge
self.constraints.max_ch = {}
for u in scen_load:
    for r in scen_price:
        for s in scen_pv:
            for t in time:
                self.constraints.max_ch[t,s,r,u] = m.addConstr(
                    self.variables.P_ch[t,s,r,u],
                    gb.GRB.LESS_EQUAL,
                    self.data.P_max_ch)
```

```
        #maximum energy discharge
        self.constraints.max_disch = {}
        for u in scen_load:
            for r in scen_price:
                for s in scen_pv:
                    for t in time:
                        self.constraints.max_disch[t,s,r,u] = m.addConstr(
                            self.variables.P_disch[t,s,r,u],
                            gb.GRB.LESS_EQUAL,
                            self.data.P_max_disch)



        #initial level battery
        self.constraints.init = {}
        for u in scen_load:
            for r in scen_price:
                for s in scen_pv:
                    self.constraints.init[s,r,u] = m.addConstr(
                        self.variables.P_level[time[0],s,r,u],
                        gb.GRB.EQUAL,
                        self.data.initial_level +
self.variables.P_ch[time[0],s,r,u] * self.data.eff_batt -
self.variables.P_disch[time[0],s,r,u] / self.data.eff_batt)



        #delta level battery
        self.constraints.delta_lev_batt = {}
        for u in scen_load:
            for r in scen_price:
                for s in scen_pv:
                    for t1,t2 in zip(time[1:], time[:-1]):
                        self.constraints.delta_lev_batt[t1,s,r,u] = m.addConstr(
                            self.variables.P_level[t1,s,r,u],
                            gb.GRB.EQUAL,
                            self.variables.P_level[t2,s,r,u] +
self.variables.P_ch[t1,s,r,u] * self.data.eff_batt -
self.variables.P_disch[t1,s,r,u] / self.data.eff_batt)



        # buy and sell fixed in time_1
        self.constraints.sell_fixed = {}
        for u in scen_load:
            for r in scen_price:
                for s in scen_pv:
                    for t in time_1:
                        self.constraints.sell_fixed[t,s,r,u] = m.addConstr(
                            self.variables.P_sell[t],
                            gb.GRB.EQUAL,
                            self.data.P_sell_fixed[t])



        self.constraints.buy_fixed = {}
        for u in scen_load:
            for r in scen_price:
```

```
                for s in scen_pv:
                    for t in time_1:
                        self.constraints.buy_fixed[t,s,r,u] = m.addConstr(
                            self.variables.P_buy[t],
                            gb.GRB.EQUAL,
                            self.data.P_buy_fixed[t])



#%% Solve
AGG_optim = AGG_model()
AGG_optim.optimize()
```

REAL-TIME STOCHASTIC MODEL

```
# -*- coding: utf-8 -*-
"""
Created on Wed Jul 13 09:14:09 2016

@author: paolo
"""

import pandas as pd
import gurobipy as gb
import xlrd
import numpy as np
import natsort as nt


#%% Build Model

class AGG_real_time(object):
    pass

class AGG_model:
    def __init__(self):
        self.data = AGG_real_time()
        self.variables = AGG_real_time()
        self.constraints = AGG_real_time()
        self._load_data()
        self._build_model()

    def optimize(self):
        self.model.optimize()

    def _load_data(self):
        #Load Sets
        self.data.time_now = time_now
        self.data.time = time
        self.data.scen_pv = scen_pv
        self.data.scen_load = scen_load
        #Load Parameters
        self.data.price = price
        self.data.PV_scen = PV_scen
        self.data.PV_prob = PV_prob_new
        self.data.Load_scen = Load_scen
        self.data.Load_prob = Load_prob_new
        self.data.P_sell = P_sell
        self.data.P_buy = P_buy
        self.data.eff_batt = eff_batt
        self.data.buy_ratio = buy_ratio
        self.data.E_max_batt = E_max_batt
        self.data.P_max_delta_sell = P_max_delta_sell
        self.data.P_max_delta_buy = P_max_delta_buy
        self.data.E_min_batt = E_min_batt
        self.data.P_max_ch = P_max_ch
```

```
        self.data.P_max_disch = P_max_disch
        self.data.a = a
        self.data.c_avg = c_avg
        self.data.P_level_now = P_level_now


    def _build_model(self):
        self.model = gb.Model()
        self._build_variables()
        self._build_objective()
        self._build_constraints()


    def _build_variables(self):
        time = self.data.time
        scen_pv = self.data.scen_pv
        scen_load_fest = self.data.scen_load
        m = self.model

        self.variables.P_ch = {} #Charge Battery
        self.variables.P_disch = {} #Discharge Battery
        self.variables.P_level = {} #Level of Battery
        self.variables.delta_sell = {} #Sell to the grid real time
        self.variables.delta_buy = {} #Buy from the grid real time

        self.variables.P_ch_i = {} #Charge Battery
        self.variables.P_disch_i = {} #Discharge Battery
        self.variables.P_level_i = {} #Level of Battery


        for s in scen_pv:
            for u in scen_load:
                for t in time:
                    self.variables.P_ch[t,s,u]    = m.addVar()
                    self.variables.P_disch[t,s,u] = m.addVar()
                    self.variables.P_level[t,s,u]   = m.addVar()
                    self.variables.delta_sell[t,s,u] = m.addVar()
                    self.variables.delta_buy[t,s,u] = m.addVar()


        self.variables.P_ch_i        = m.addVar()
        self.variables.P_disch_i     = m.addVar()
        self.variables.P_level_i     = m.addVar()

        m.update()


    def _build_objective(self):
        scen_pv = self.data.scen_pv
        scen_load = self.data.scen_load
        time=self.data.time
        m = self.model
```

```
        m.setObjective(
            gb.quicksum(self.data.PV_prob[s][t]*self.data.Load_prob[u][t]*\
                        (self.data.price[t] *self.data.a * self.data.buy_ratio * self.variables.delta_buy[t,s,u] -\
                        self.data.price[t] / self.data.a * self.variables.delta_sell[t,s,u]) for t in time for s in scen_pv \
                        for u in scen_load) -
gb.quicksum(self.data.PV_prob[s][time[9]]*self.data.Load_prob[u][time[9]]*\
                        (self.data.c_avg * self.variables.P_level[time[9],s,u])for s in scen_pv for u in scen_load),
            gb.GRB.MINIMIZE)

    def _build_constraints(self):
        scen_pv = self.data.scen_pv
        scen_load = self.data.scen_load
        time=self.data.time
        time_now = self.data.time_now
        m=self.model


        #power balance
        self.constraints.power_balance = {}
        for u in scen_load:
            for s in scen_pv:
                for t in time:
                    self.constraints.power_balance[t,s,u] = m.addConstr(
                        self.data.Load_scen[u][t],
                        gb.GRB.EQUAL,
                        self.data.PV_scen[s][t] + \
                        self.data.P_buy[t] - \
                        self.data.P_sell[t] + \
                        self.variables.P_disch[t,s,u] - \
                        self.variables.P_ch[t,s,u] + \
                        self.variables.delta_buy[t,s,u] -\
                        self.variables.delta_sell[t,s,u])

        #maximum battery level
        self.constraints.batt_lev_max = {}
        for u in scen_load:
            for s in scen_pv:
                for t in time:
                    self.constraints.batt_lev_max[t,s,u] = m.addConstr(
                        self.variables.P_level[t,s,u],
                        gb.GRB.LESS_EQUAL,
                        self.data.E_max_batt)

#minimum battery level
        self.constraints.batt_lev_min = {}
        for u in scen_load:
            for s in scen_pv:
                for t in time:
                    self.constraints.batt_lev_min[t,s,u] = m.addConstr(
                        self.variables.P_level[t,s,u],
                        gb.GRB.GREATER_EQUAL,
                        self.data.E_min_batt)
```

```
        #maximum energy delta_sell
        self.constraints.max_sell = {}
        for u in scen_load:
            for s in scen_pv:
                for t in time:
                    self.constraints.max_sell[t,s,u] = m.addConstr(
                        self.variables.delta_sell[t,s,u],
                        gb.GRB.LESS_EQUAL,
                        self.data.P_max_delta_sell)


        #maximum energy delta_buy
        self.constraints.max_buy = {}
        for u in scen_load:
            for s in scen_pv:
                for t in time:
                    self.constraints.max_buy[t,s,u] = m.addConstr(
                        self.variables.delta_buy[t,s,u],
                        gb.GRB.LESS_EQUAL,
                        self.data.P_max_delta_buy)

        #maximum energy charge
        self.constraints.max_ch = {}
        for u in scen_load:
            for s in scen_pv:
                for t in time:
                    self.constraints.max_ch[t,s,u] = m.addConstr(
                        self.variables.P_ch[t,s,u],
                        gb.GRB.LESS_EQUAL,
                        self.data.P_max_ch)

        #maximum energy discharge
        self.constraints.max_disch = {}
        for u in scen_load:
            for s in scen_pv:
                for t in time:
                    self.constraints.max_disch[t,s,u] = m.addConstr(
                        self.variables.P_disch[t,s,u],
                        gb.GRB.LESS_EQUAL,
                        self.data.P_max_disch)


        #initial level battery
        self.constraints.init = {}
        for u in scen_load:
            for s in scen_pv:
                self.constraints.init[s,u] = m.addConstr(
                    self.variables.P_level[time[0],s,u],
                    gb.GRB.EQUAL,
                    self.data.P_level_now[time_now] + self.variables.P_ch[time[0],s,u] * self.data.eff_batt -
    self.variables.P_disch[time[0],s,u] / self.data.eff_batt)
```

```
        #delta level battery
        self.constraints.delta_lev_batt = {}
        for u in scen_load:
            for s in scen_pv:
                for t1,t2 in zip(time[1:], time[:-1]):
                    self.constraints.delta_lev_batt = m.addConstr(
                        self.variables.P_level[t1,s,u],
                        gb.GRB.EQUAL,
                        self.variables.P_level[t2,s,u] + self.variables.P_ch[t1,s,u] * self.data.eff_batt -
self.variables.P_disch[t1,s,u] / self.data.eff_batt)

        #P_ch_i P_disch_i P_level_i indipendent from scenario

        self.constraints.P_ch_i = {}
        for u in scen_load:
            for s in scen_pv:
                self.constraints.P_ch_i[s,u] = m.addConstr(
                    self.variables.P_ch[time[0],s,u],
                    gb.GRB.EQUAL,
                    self.variables.P_ch_i)

        self.constraints.P_disch_i = {}
        for u in scen_load:
            for s in scen_pv:
                self.constraints.P_disch_i[s,u] = m.addConstr(
                    self.variables.P_disch[time[0],s,u],
                    gb.GRB.EQUAL,
                    self.variables.P_disch_i)

        self.constraints.P_level_i = {}
        for u in scen_load:
            for s in scen_pv:
                self.constraints.P_level_i[s,u] = m.addConstr(
                    self.variables.P_level[time[0],s,u],
                    gb.GRB.EQUAL,
                    self.variables.P_level_i)

#%% Solve
AGG_optim = AGG_model()
AGG_optim.optimize()
print AGG_optim.model.SolCount
```

Ringraziamenti

Desidero innanzitutto ringraziare il Prof. Arturo Lorenzoni per l'interesse che mi ha trasmesso riguardo le tematiche trattate in questa tesi e per essere stato sempre disponibile in questi mesi.

Voglio ringraziare in particolar modo Nicolò Mazzi, per avermi costantemente seguito in questo lavoro. Grazie soprattutto per la fiducia che mi hai dato e per essere, prima che correlatore, un grande amico.

Un grazie sincero è per l'amico più caro, Alberto. Per raggiungere questo importante obiettivo la tua presenza è stata fondamentale. Ti ringrazio perchè mi stimoli a dare il massimo e perchè ci sei sempre.

Un pensiero va ai miei amici. Grazie a Ranchi, Cesare, Davide, Ester, Anto, Marca, Pedro, Fenzi, Clemens, Bau, Nicola, Mazzu. Grazie a Casa Liberi e alla 5F. Grazie a Giacomo, ai ragazzi di Sommacampagna e a tutti quelli che non ho citato. Grazie perchè in questi anni, sicuramente avrò anche un po' studiato, ma principalmente mi sono divertito con voi.

Grazie di cuore alla mia famiglia perchè mi ha sempre sostenuto. Come immagino voi siate orgogliosi di me, io lo sono di voi.

Ringrazio infine Laura, perchè conoscerti è stata la svolta. Grazie perchè mi fai stare bene e mi fai sorridere. Sei la compagna ideale.