

**U P C**

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Escola Politècnica Superior d’Enginyeria de Manresa

Gestor energètic

Control de càrregues, alimentades per autoproducció renovable i/o la xarxa, per minimitzar el cost elèctric

3 juliol de 2022

treball de fi de grau que presenta

Jordi Panadès Closes

en compliment dels requisits per assolir el

Grau d’Enginyeria en Sistemes TIC

Direcció: Francisco del Aguila Lopez i Jordi Bonet Dalmau

Aquesta obra està subjecte a la llicència de Reconeixement-NoComercial-CompartirIgual 3.0 Espanya de Creative Commons. Si voleu veure una còpia d'aquesta llicència accediu a <http://creativecommons.org/licenses/by-nc-sa/3.0/es/> o envieu una carta sol·licitant-la a Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.



Voldria donar les gràcies al Francisco i el Jordi

per cedir-me les dades reals de les seves instal·lacions d’autoconsum

i permetre instal·lar els sistemes de control reals en les mateixes

# Resum

A Espanya, l’autoproducció i autoconsum d’energies renovables està regit pel Reial Decret 244/2019 [1]. Per un costat, aquest decret permet als usuaris consumir de la xarxa elèctrica quan la seva producció no cobreix les necessitats i, per l’altre, vendre l’energia excedent a l’empresa subministradora contractada, a un preu menor però.

Gràcies al balanç net horari, l’energia injectada a la xarxa equival a la consumida, fent que la reducció del preu de factura de la llum sigui més significativa. També permet a l’usuari tenir cert joc per consumir i retornar el consumit de forma no instantània.

Aquest projecte planteja diversos algoritmes que es poden fer servir per controlar les càrregues d’una instal·lació, simplificada, per tal de reduir el cost de la factura i aprofitar al màxim l’energia produïda.

# Abstract

In Spain, the self-production and self-consumption of renewable energies is governed by the Royal Decree 244/2019 [1]. This decree allows the users to consume from the energy grid when their production doesn’t cover their needs. It also allows them to sell the surplus energy to the supply company, at a lower price be it.

Thanks to the *“Balanç net horari”* (net hourly balance), the energy injected to the grid is equivalent to the consumed, making reducing the energy bill price easier. This also allows the user some wiggle room to consume and return the consumption non-instantaneously.

This project explores some of the various algorithms that can be used to control the loads of a simplified installation in order to reduce the cost of the bill and make the most of the energy produced.

Índex

[Resum v](#_Toc107763251)

[Abstract v](#_Toc107763252)

[I. Memòria 1](#_Toc107763253)

[1. Introducció 3](#_Toc107763254)

[1.1. Punt de partida 3](#_Toc107763255)

[1.2. Objectiu/s 4](#_Toc107763256)

[1.3. Dades **Error! Bookmark not defined.**](#_Toc107763257)

[2. Model del sistema 7](#_Toc107763258)

[2.1. Model general i model ideal 7](#_Toc107763259)

[2.2. Model simplificat 7](#_Toc107763260)

[3. Terminologia 9](#_Toc107763261)

[3.1. Potències 9](#_Toc107763262)

[3.1.1 Definicions 9](#_Toc107763263)

[3.1.2 Traduccions 10](#_Toc107763264)

[3.1.3 Segmentació visual 10](#_Toc107763265)

[3.1.4 Exemples 11](#_Toc107763266)

[3.2. Energies 14](#_Toc107763267)

[3.2.1 Definicions 14](#_Toc107763268)

[3.2.2 Traduccions 16](#_Toc107763269)

[3.2.3 Segmentació visual 17](#_Toc107763270)

[3.2.4 Exemples 17](#_Toc107763271)

[3.3. Per comparació 21](#_Toc107763272)

[3.3.1 Definicions 21](#_Toc107763273)

[3.4. Traduccions 22](#_Toc107763274)

[4. Algoritmes de control 23](#_Toc107763275)

[4.1. Histèresi 23](#_Toc107763276)

[4.1.1 Una càrrega 23](#_Toc107763277)

[4.1.2 Dues càrregues 24](#_Toc107763278)

[4.1.3 Observacions 24](#_Toc107763279)

[4.2. Temps mínim engegada 27](#_Toc107763280)

[4.2.1 Una càrrega 27](#_Toc107763281)

[4.2.2 Dues càrregues 27](#_Toc107763282)

[4.2.3 Observacions 28](#_Toc107763283)

[4.3. Predictiu (Temps de consum) 32](#_Toc107763284)

[4.3.1 Com predir el balanç final 32](#_Toc107763285)

[4.3.2 Una càrrega 33](#_Toc107763286)

[4.3.2.1 Plantejament 33](#_Toc107763287)

[4.3.2.2 Versió final 35](#_Toc107763288)

[4.3.3 Dues càrregues 35](#_Toc107763289)

[5. Simulador 37](#_Toc107763290)

[5.1. Instal·lació i ús 37](#_Toc107763291)

[5.2. Llenguatge i llibreries 37](#_Toc107763292)

[5.3. Implementació del simulador (programa) 38](#_Toc107763293)

[5.4. Implementació de la simulació (simulator.py) 41](#_Toc107763294)

[6. Implementació real 45](#_Toc107763295)

[6.1. Descripció de la instal·lació 45](#_Toc107763296)

[6.2. Implementació del software 47](#_Toc107763297)

[6.2.1 Drivers 47](#_Toc107763298)

[6.2.2 Gestió (managing.py) 48](#_Toc107763299)

[6.2.3 Monitorització (monitoring.py) 48](#_Toc107763300)

[6.3. Instal·lació i ús 48](#_Toc107763301)

[7. Anàlisi 51](#_Toc107763302)

[7.1. Histèresi vs Temps mínim engegada 51](#_Toc107763303)

[8. Conclusions 53](#_Toc107763304)

[Bibliografia 55](#_Toc107763305)

Figures

[Figura 1.1. Càlcul de la factura de la llum 3](#_Toc107763306)

[Figura 2.1. Model energètic (vàlid tant per potència o energia) 8](#_Toc107763307)

[Figura 3.1. Segmentació visual de la potència del sistema 10](#_Toc107763308)

[Figura 3.2. Exemple – Potència produïda d’un dia assolellat 11](#_Toc107763309)

[Figura 3.3. Exemple - Potència produïda d’un dia ennuvolat 11](#_Toc107763310)

[Figura 3.4. Exemple - Potència produïda d’un dia plujós 12](#_Toc107763311)

[Figura 3.5. Exemple - Potència consumida i de cada càrrega 12](#_Toc107763312)

[Figura 3.6. Exemple – Potència disponible i de la xarxa – Vista completa 13](#_Toc107763313)

[Figura 3.7. Exemple - Potència del consum màxim (ideal) 13](#_Toc107763314)

[Figura 3.8. Formes de calcular el balanç 15](#_Toc107763315)

[Figura 3.9. Segmentació visual de l’energia del sistema 17](#_Toc107763316)

[Figura 3.10. Exemple – Consum màxim d’energia a partir d’àrees 17](#_Toc107763317)

[Figura 3.11. Exemple - Energia perduda a partir d’àrees 18](#_Toc107763318)

[Figura 3.12. Exemple – Energia disponible del balanç i deguda a la xarxa – Vista comportament 18](#_Toc107763319)

[Figura 3.13. Exemple – Resultats per hores de les energies (vista model) 19](#_Toc107763320)

[Figura 3.14. Exemple – Resultats per hores de les energies (1a segmentació) 19](#_Toc107763321)

[Figura 3.15. Exemple – Resultats per hores de les energies (2a segmentació) 20](#_Toc107763322)

[Figura 3.16. Exemple – Model energètic per hores (2a segmentació) 20](#_Toc107763323)

[Figura 4.1. Corba d’histèresi del control d’una càrrega 23](#_Toc107763324)

[Figura 4.2. Histèresi – Màquina d’estats per controlar una càrrega 24](#_Toc107763325)

[Figura 4.3. Histèresi – 1a càrrega amb preferència d’engegada respecta la 2a 24](#_Toc107763326)

[Figura 4.4. Histèresi – 2 càrregues amb els mateixos llindars equivalen a 1 25](#_Toc107763327)

[Figura 4.5. Histèresis – Llindar baix a 0 i càlcul de balanç projectant al passat 26](#_Toc107763328)

[Figura 4.6. Histèresis – Llindar baix a 0 i càlcul de balanç projectant al futur 26](#_Toc107763329)

[Figura 4.7. Temps mín. engegada – Solapament de comportament de les implementacions 28](#_Toc107763330)

[Figura 4.8. Temps mín. engegada – Implementacions amb el mateix comportament quan C1 < C2 29](#_Toc107763331)

[Figura 4.9. Temps mín. engegada – Implementacions amb el mateix comportament quan C1 > C2 30](#_Toc107763332)

[Figura 4.10. Histèresi (segona part) amb mateixos llindars que Temps mínim engegat (primera part) 31](#_Toc107763333)

[Figura 4.11. Predictiu – Predicció projectant la potència disponible 32](#_Toc107763334)

[Figura 4.12. Predictiu – Exemplificació de l’àrea de equivalent a la integral 32](#_Toc107763335)

[Figura 4.13. Predictiu – Predicció a partir de la potència mitjana disponible 33](#_Toc107763336)

[Figura 4.14. Predictiu base – Situació ideal 33](#_Toc107763337)

[Figura 4.15. Predictiu base – Situacions extremes que pot tenir 34](#_Toc107763338)

[Figura 4.16. Predictiu base – Solució al problema a) i nou problema que presenta 34](#_Toc107763339)

[Figura 4.17. Predictiu base – Solució al problema b) i nou problema que presenta 34](#_Toc107763340)

[Figura 4.18. Predictiu – Màquina d’estats per controlar una càrrega 35](#_Toc107763341)

[Figura 4.19. Predictiu – Màquina d’estats per controlar dues càrregues 35](#_Toc107763342)

[Figura 5.1. Simulador – Dependència entre mòduls 38](#_Toc107763343)

[Figura 6.2. Shelly1L 46](#_Toc107763344)

[Figura 6.1. Enphase Envoy-S Metered 46](#_Toc107763345)

[Figura 6.2. Raspberry Pi Model B 46](#_Toc107763346)

[Figura 6.3. Dashboard creat en Grafana que mostra les dades del sistema a temps real 46](#_Toc107763347)

[Figura 6.4. Implementació real – Relació programa-driver 47](#_Toc107763348)

Taules

[Taula 1.1. Exemple – Càlcul del balanç net horari 4](#_Toc107763349)

[Taula 1.2. Exemple – Càlcul de la factura 4](#_Toc107763350)

[Taula 1.3. Exemple – Càlcul de la factura sense balanç net horari 4](#_Toc107763351)

[Taula 3.1. Traduccions i nomenclatures – Potències 10](#_Toc107763352)

[Taula 3.2. Traduccions i nomenclatures – Energies 16](#_Toc107763353)

[Taula 3.3. Traduccions i nomenclatures – Comparació 22](#_Toc107763354)

[Taula 4.1. Temps mín. engegada – Implementacions amb el mateix comportaments quan C1 < C2 28](#_Toc107763355)

[Taula 4.2. Temps mín. engegada – Implementacions amb el mateix comportaments quan C1 > C2 28](#_Toc107763356)

[Taula 4.3. Temps mín. engegada – Implementacions amb el mateix comportaments quan C1 = C2 28](#_Toc107763357)

1. Memòria

# Introducció

## Punt de partida

A Espanya, l’autoproducció i autoconsum d’energies renovables està regit pel Reial Decret 244/2019 [1]. Aquest defineix diverses modalitats d’autoconsum a les quals els usuaris de la xarxa elèctrica es poden acollir[[1]](#footnote-2). En resum hi ha dues modalitats en funció de l’energia produïa[[2]](#footnote-3):

1. Sense excedents: L’energia produïda no consumida, si n’hi ha, no s’injecta a la xarxa. Requereix instal·lar un sistema antiabocament que impedeix la injecció d’energia excedent a la xarxa de transport i distribució.
2. Amb excedents: L’energia produïda no consumida és injectada a la xarxa de transport i distribució. Aquests excedents poden ser:
   1. Sense compensació econòmica
   2. Amb compensació econòmica simplificada

La compensació simplificada consisteix en un saldo monetari de l’energia consumida de la xarxa en el període de facturació[[3]](#footnote-4), [[4]](#footnote-5):

* El cost a pagar és la diferència entre el valor econòmic de l’energia horària consumida de la xarxa i de l’energia horària excedent/injectada. És a dir, la suma de tots els valors econòmics dels balanços nets horaris dins del període.
* Com a màxim és compensa el valor econòmic d’energia horària consumida de la xarxa. Això vol dir que el valor més baix de la factura pot ser 0.
* La compensació com a molt pot ser mensual.

El balanç net horari és el saldo energètic net hora a hora [2]. És a dir, la diferència entre l’energia consumida de la xarxa i l’excedent injectada (des del punt de vista del proveïdor).

|  |  |
| --- | --- |
|  | (.1) |

Si un resultat és positiu significa que s’ha consumit de la xarxa, si és negatiu, que s’ha injectat.

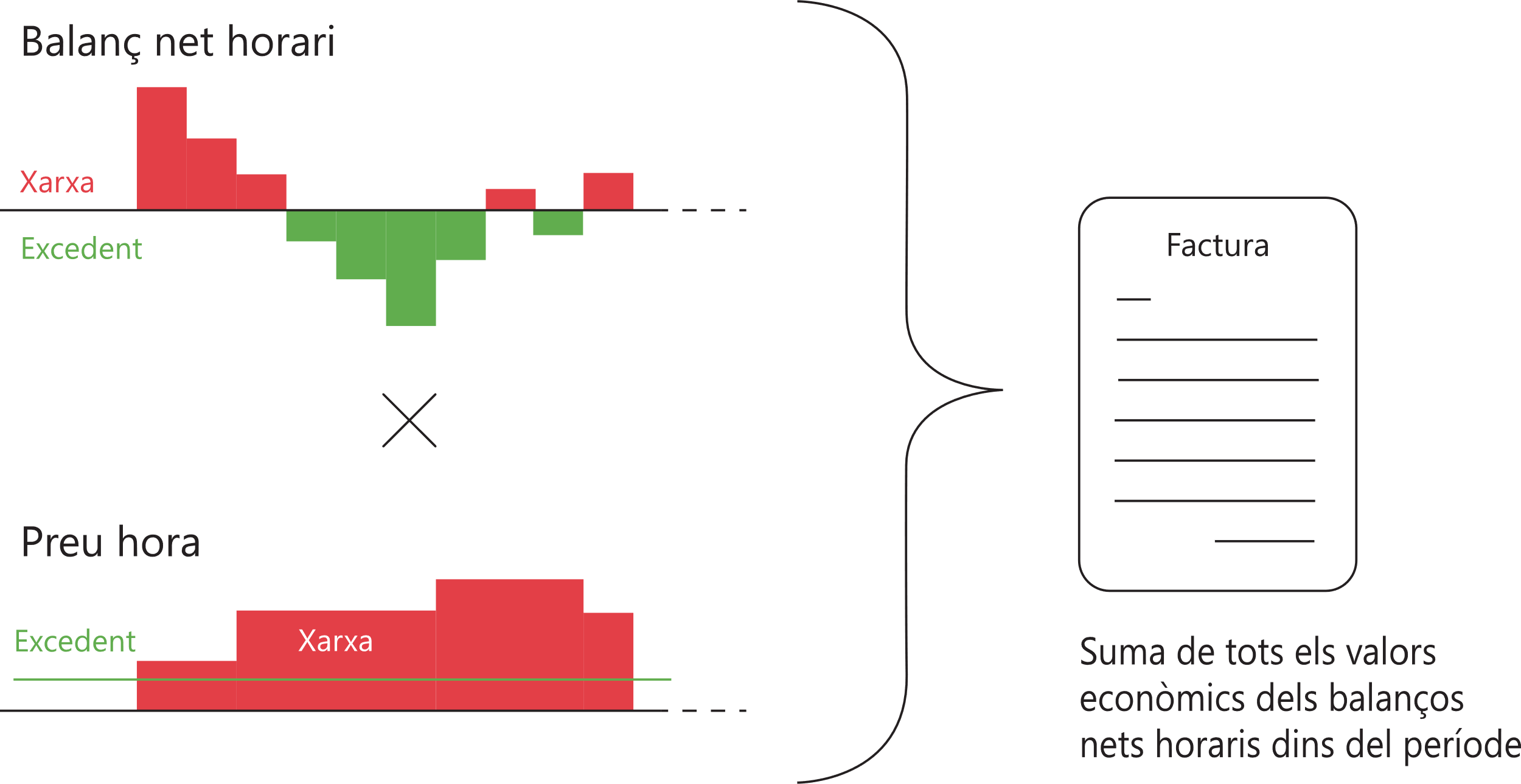


Figura 1.1. Càlcul de la factura de la llum

Exemple

Primer exemplificarem el càlcul del balanç net horari (en kWh):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| Hora | Consum instal·lació | Producció | Consum Xarxa | Injectats |
| 13:00 | 0,5 | 0,4 | 0,1 | 0 |
| 13:15 | 0,5 | 0,4 | 0,1 | 0 |
| 13:30 | 0 | 0,4 | 0 | 0,4 |
| 13:45 | 0 | 0,4 | 0 | 0,4 |
| Total | 1 | 1,6 | 0,2 | 0,8 |

Taula 1.1. Exemple – Càlcul del balanç net horari

Degut al balanç net horari, és com si només haguéssim injectat 0,6 kWh.

|  |  |
| --- | --- |
|  | (.2) |

Un cop vist el càlcul del balanç net horari, podem mirar com és calcula la factura amb compensació simplificada, si el període fos de 5 hores:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | | |  |  | |  |  |
|  |  | Energia [kWh] | | |  | Preu [€/kWh] | |  |  |
| Hora |  | Consum Xarxa | Injectats | Balanç net horari |  | Xarxa | Injectats |  | Total [€] |
| 12:00 |  | 2,0 | 0 | 2,0 |  | 0,357 | 0,187 |  | 0,714 |
| 13:00 |  | 1,0 | 3,0 | - 2,0 |  | 0,357 | 0,187 |  | - 0,374 |
| 14:00 |  | 1,0 | 2,5 | - 1,5 |  | 0,293 | 0,187 |  | - 0,2805 |
| 15:00 |  | 2,0 | 0,5 | 1,5 |  | 0,293 | 0,187 |  | 0,4395 |
| 16:00 |  | 1,0 | 1,5 | - 0,5 |  | 0,293 | 0,187 |  | - 0,0935 |
|  |  |  |  |  |  |  | Factura [€] | | 0,4055 |

Taula 1.2. Exemple – Càlcul de la factura

## Objectiu/s

Gràcies al balanç net horari podem reduir considerablement el cost de la factura, ja que és com si estiguéssim “venent” l’energia al mateix preu que el cost de compra, dins de cada interval d’hora. Per demostrar-ho podem repetir l’exemple anterior sense balanç net total:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | |  |  | |  |  | | |
|  |  | Energia [kWh] | |  | Preu [€/kWh] | |  | [€] | | |
| Hora |  | Consum Xarxa | Injectats |  | Xarxa | Preu Injectats |  | Cost Xarxa | Benefici Injectat | Total |
| 12:00 |  | 2,0 | 0 |  | 0,357 | 0,187 |  | 0,714 | 0 | 0,714 |
| 13:00 |  | 1,0 | 3,0 |  | 0,357 | 0,187 |  | 0,357 | 0,561 | - 0,204 |
| 14:00 |  | 1,0 | 2,5 |  | 0,293 | 0,187 |  | 0,293 | 0,4675 | - 0,1745 |
| 15:00 |  | 2,0 | 0,5 |  | 0,293 | 0,187 |  | 0,586 | 0,0935 | 0,4925 |
| 16:00 |  | 1,0 | 1,5 |  | 0,293 | 0,187 |  | 0,293 | 0,2805 | 0,0125 |
|  |  |  |  |  |  |  |  |  | Total Factura [€] | 0,8405 |

Taula 1.3. Exemple – Càlcul de la factura sense balanç net horari

Com podem veure, el cost de la factura sense balanç net horari és més elevat. Aquest principi fonamental és el que ha motivat la creació d’aquest projecte.

L’objectiu d’aquest projecte és dissenyar, analitzar i comparar algoritmes que controlin les càrregues elèctriques d’una instal·lació amb autoproducció renovable per tal de: reduir el preu de la factura al màxim (buscar cost 0), maximitzar l’ús d’aquesta producció i minimitzar les commutacions de les càrregues.

L’objectiu està enfocat principalment cap al model d’autoproducció amb excedents i compensació econòmica. Però la resta de models també poden aplicar els mateixos algoritmes per injectar el mínim a la xarxa i aprofitar al màxim la seva producció.

Inicialment, la idea havia sortit per ser aplicada en un habitatge, tot i així, al tractar amb conceptes genèrics, es pot aplicar a quasi qualsevol espai o instal·lació.

L’objectiu general s’ha segmentat en els següents objectius/tasques:

* Dissenyar un model el màxim simplificat del sistema.
* Dissenyar diferents algoritmes de control.
* Crear un simulador del sistema.
* Implementar el sistema a la realitat i provar els algoritmes.
* Analitzar i comprar els algoritmes.

## Codi font

Si heu obtingut la memòria del projecte sense el codi font, podeu trobar-lo a GitHub [3].

# Model del sistema

Abans de res, el que hem de fer és plantejar com serà el sistema que l’algoritme controlarà. Així podrem simular-lo, implementar-lo i sobretot comparar els algoritmes entre ells, ja que si cadascun gestiona un sistema diferent, tindrem més d’una variable no fixa i no una de sola.

## Model general i model ideal

Si generalitzem, podem descriure el sistema com un sistema que, a partir de múltiples entrades (mostres) i dades internes (preestablertes o calculades a temps real), gestioni les càrregues per assolir les necessitats que l’usuari tingui i indiqui. Això implica directament algoritmes molt complexos.

Si l’idealitzem, el sistema tindria una gran nombre d’entrades i dades internes per poder cobrir totes les necessitats:

* Entrades: producció, consum de la xarxa a temps real, temperatura de l’habitatge, temperatura de la calefacció, predicció d’irradiació solar, ...
* Dades internes:consum intern, balanç net horari a temps real, aproximació de la producció diària, quant i en quina forma cada càrrega consumeix, quines càrregues controla, ...
* Control:calefacció, rentadora, nevera, carregar el cotxe, ...
* Necessitats: aclimatació de l’espai, engegar una rentadora ja preparada al millor moment però abans del dia/hora determinada, poder fer un cafè de forma espontània, ...
* Complexitat de l’algoritme: predir els consums i l’energia a l’acabar l’hora; si volem controlar la temperatura i que la càrrega s’engegui al millor moment, també hauria de predir la temperatura; deduir i predir rutines de consum; ...

## Model simplificat

Ara bé, aquest treball és el primer o dels primers en aquest àmbit, per això partirem d’una aproximació del sistema el màxim de simplificada:

* Entrades: consum de la xarxa i producció en temps real.
* Dades internes: consum intern i balanç net horari en temps real.
* Control:dues càrregues (com poden ser dues bombes de calor).
* Necessitats:mantenir les càrregues el màxim de temps enceses, és a dir, consumir el màxim d’energia produïa possible donades les càrregues.
* Possible complexitat de l’algoritme: predir els consums i l’energia a l’acabar l’hora.

A nivell energètic, tant si parlem en termes de potència com d’energia, es pot modelar com un sistema amb entrades i sortides:

* Entrades: potència/energia produïda i/o de la xarxa.
* Ús intern: potència/energia consumida.
* Sortides: potència/energia produïda no consumida (sobrant).

I sempre complirà que:

|  |  |
| --- | --- |
|  | (2.1) |

Produïda

Xarxa

No consumida

Consumida

Figura 2.1. Model energètic (vàlid tant per potència o energia)

# 

# Terminologia

Al desenvolupar el projecte, s’han definit i segmentat diferents potències i energies per poder fer els càlculs i analitzar els resultats. En la mateixa línia també s’han definit altres conceptes orientats a la comparació d’algoritmes.

A més a més, tant la programació com la interfície gràfica del simulador s’han fet en anglès per així treballar la 3a llengua i, com a conseqüència, fer-lo més accessible a tot el món.

La finalitat d’aquesta secció és definir i exemplificar aquest termes.

## Potències

### Definicions

La classificació de potències és la següent:

* Produïda: potència total generada per una o més fonts d’energia renovable.
* Càrrega *i*: potència consumida de la càrrega *i.*
* Càrrega Base: potència consumida de totes les càrregues de la instal·lació no controlades pel sistema.
* Càrrega 1 o 2: potència consumida de la càrrega controlada 1 o 2.
* Consumida: potència consumida per la instal·lació.

|  |  |
| --- | --- |
|  | (.1) |

* Consum màxim (ideal): potència consumida per la instal·lació si aquesta consumís el màxim de potència produïda possible. Té la mateixa forma que la potència produïda, però saturant el valor a la càrrega de consum total màxima de la instal·lació (com si totes les càrregues estiguessin engegades).

|  |  |
| --- | --- |
|  | (.2) |

* Potència de consum: potència estipulada que consumeix un producte. Si sabem la potència de consum de cada càrrega podem calcular la càrrega màxima fàcilment:

|  |  |
| --- | --- |
|  | (3.3) |

Ara, si no sabem les potències de consum, les podem aproximar fent una mitjana de les mostres.

|  |  |
| --- | --- |
|  | (3.4) |
|  | (3.5) |

És important que, al fer el càlcul de la càrrega, tinguem clar què estem calculant, ja que si incloem les mostres que valen 0, estarem considerant els instants en que la càrrega ha estat apagada. Cosa que resultaria en la potència mitjana que hi ha hagut, i no en la potència que té la càrrega quan està encesa.

* Xarxa: potència consumida de la xarxa. Només s’utilitza per suplir l’energia consumida quan la produïda no cobreix la demanda.

|  |  |
| --- | --- |
|  | (.6) |

* No consumida o disponible: potència produïda després de considerar la consumida. Si ens hi fixem, *pX* i *pD* són la part positiva i negativa de calcular *pP – pC.*

|  |  |
| --- | --- |
|  | (.7) |

* Xarxa durant la producció: potència consumida de la xarxa només quan s’ha estat produint energia.

|  |  |
| --- | --- |
|  | (.8) |

### Traduccions

La traducció a l’anglès dels termes de potència definits anteriorment i els noms que es fan servir a les equacions són:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
| Terme | |  | Formula | |
| Català | Anglès |  | Català (document) | Anglès (codi i simulador) |
| Produïda | Produced |  | *pP* | *powerP* |
| Consumida | Consumed |  | *pC* | *powerC* |
| Xarxa | Grid |  | *pX* | *powerG* |
| Xarxa durant la producció | Grid while prod. |  | *pXP* | *powerGP* |
| Prod. no cons. o disponible | Available |  | *pPnC o pD* | *powerA* |
| Càrrega base | Base load |  | *pCB* | *powerLB* |
| Càrrega 1 | Load 1 |  | *pC1* | *powerL1* |
| Càrrega 2 | Load 2 |  | *pC2* | *powerL2* |
| Consum màxim | Max consumption |  | *pCM* | *powerCM* |
| De consum de la càrrega *i* | - |  | *carregai o Ci* | *-* |

Taula 3.1. Traduccions i nomenclatures – Potències

### Segmentació visual

Una bona forma visual de representar aquesta segmentació és a través de blocs, on cada bloc té la seva segmentació a la dreta.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sistema | Produïda | | Produïda no consumida | |  |
| Màxim consum |
| Consumida | Càrrega base |
| Xarxa | Xarxa fora prod. | Càrrega 1 |
| Xarxa durant prod. | Càrrega 2 |

Figura 3.1. Segmentació visual de la potència del sistema

### Exemples

Per facilitar la lectura de les gràfiques que hi ha al llarg del treball, i exemplificar els conceptes anteriors, a continuació es mostren i comenten diverses gràfiques generades amb el simulador[[5]](#footnote-6) a partir de dades reals.

Potència produïda:a través d’una gràfica de potència produïda podem determinar quina ha estat la situació meteorològica d’aquell dia.



Figura 3.2. Exemple – Potència produïda d’un dia assolellat

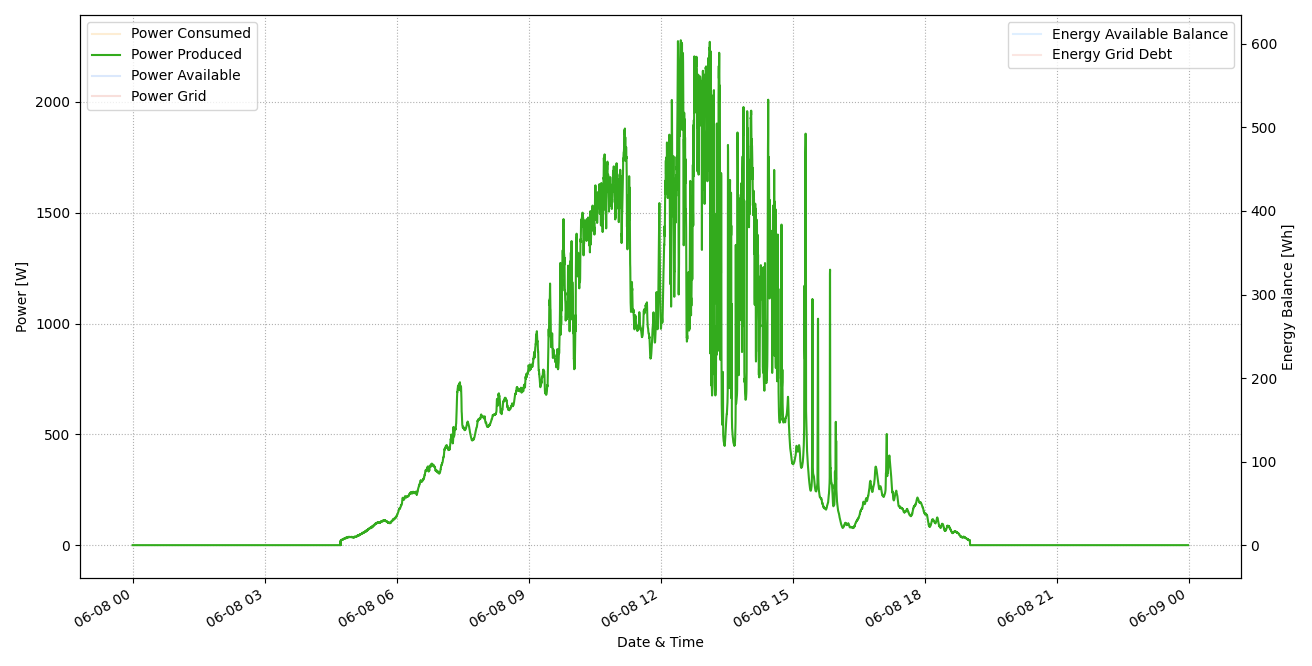


Figura 3.3. Exemple - Potència produïda d’un dia ennuvolat

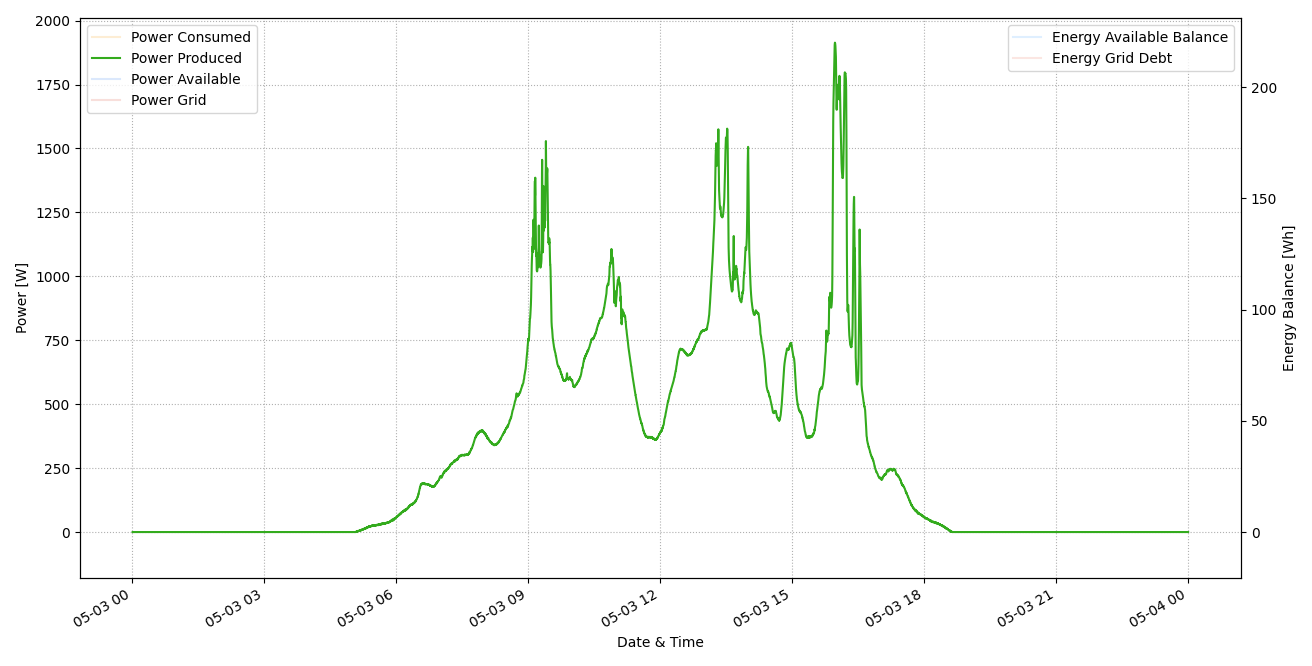


Figura 3.4. Exemple - Potència produïda d’un dia plujós

Potència consumida/càrrega: una bona forma de representar la potència consumida total i la de cada càrrega en una sola gràfica és dibuixant la total i, posteriorment, segmentar l’àrea d’aquesta en les potències de cada càrrega. D’aquesta forma estàs mostrant la potència instantània total, de cada càrrega i l’energia total consumida (les àrees).

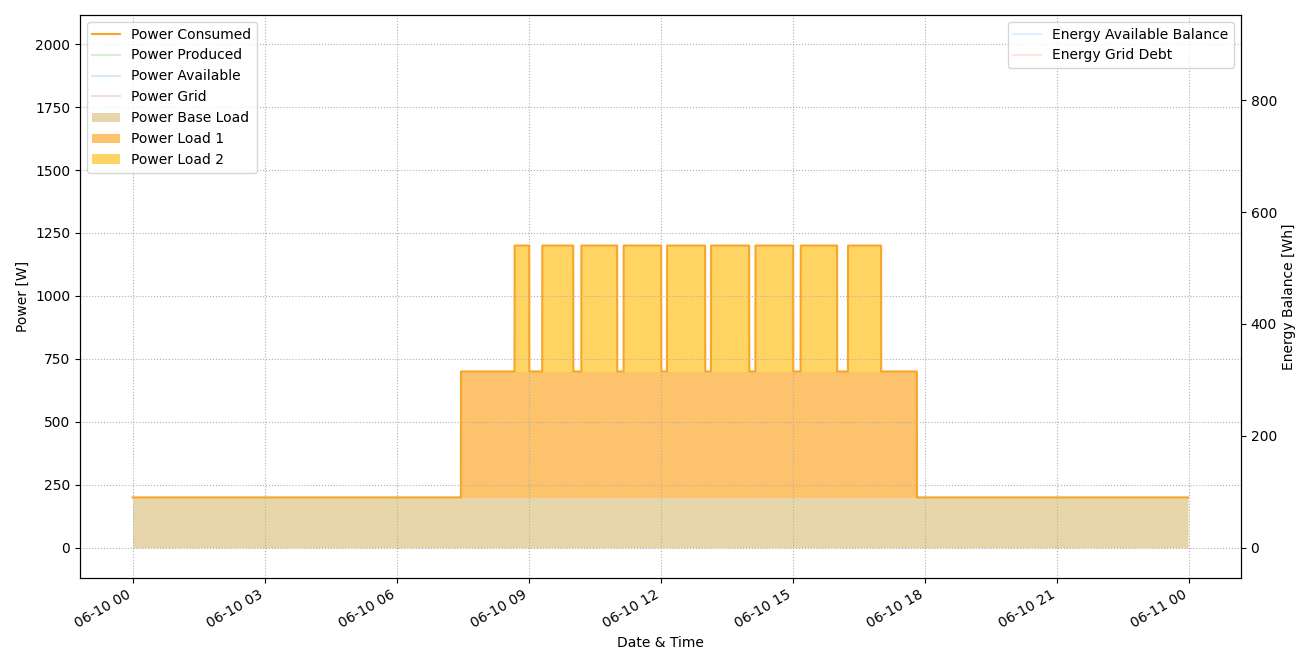


Figura 3.5. Exemple - Potència consumida i de cada càrrega

Potència de la xarxa i potència disponible:per veure clarament d’on surten els valors, va molt bé tenir al fons de la gràfica l’energia produïda i la consumida. Quan la producció està dins de l’àrea de la consumida és quan agafem de la xarxa. En canvi, quan està per sobre, és quan en tenim d’extra.

**

Figura 3.6. Exemple – Potència disponible i de la xarxa – Vista completa

Gràcies a la segmentació triada, amb una sola gràfica podem tenir tota la informació necessària per veure com ha anat el dia en termes de potència.

Potència de consum màxim:amb la gràfica, podem veure clarament el que s’ha comentat en la definició. Quan parlem del consum màxim, ens referim a aprofitar tota la potència possible que podem arribar a consumir amb les càrreges donades.

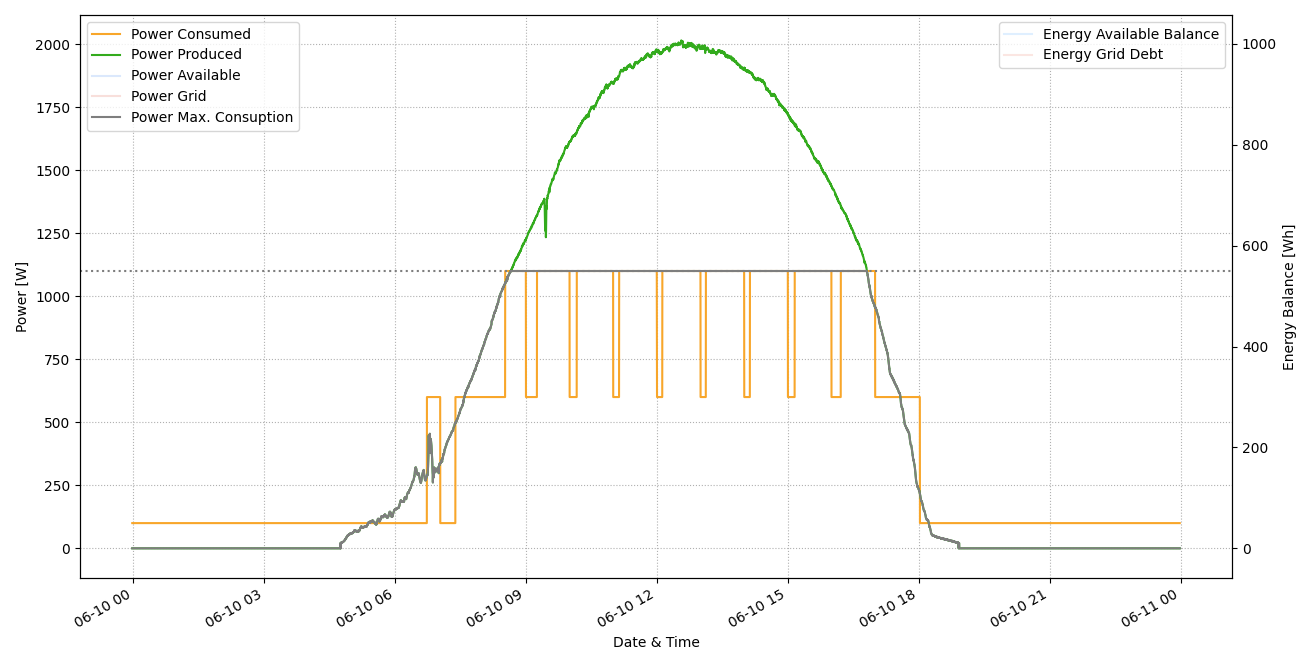
**

Figura 3.7. Exemple - Potència del consum màxim (ideal)

## Energies

### Definicions

La classificació d’energies és la següent:

* Sistema: energia que ha entrat al sistema. Aquesta ha de ser igual a la consumida més la que ha sortit.

|  |  |
| --- | --- |
|  | (.9) |

* Produïda: energia produïda per les fonts d’energia renovable.

|  |  |
| --- | --- |
|  | (.10) |

* Càrrega base: energia consumida per totes les càrregues de la instal·lació no controlades pel sistema.

|  |  |
| --- | --- |
|  | (3.11) |

* Càrrega 1 o 2: energia consumida per la càrrega controlada *i.*

|  |  |
| --- | --- |
|  | (3.12) |

* Consumida: energia consumida per la instal·lació.

|  |  |
| --- | --- |
|  | (3.13) |

* Xarxa: energia consumida de la xarxa. Si és més gran que 0, és que en algun moment no s’ha produït prou com per cobrir el consum de les càrregues.

|  |  |
| --- | --- |
|  | (3.14) |

* Xarxa durant prod.: energia consumida de la xarxa quan s’ha estat produint energia.

|  |  |
| --- | --- |
|  | (3.15) |

* Xarxa durant no prod.: energia consumida de la xarxa quan no s’ha estat produint energia.

|  |  |
| --- | --- |
|  | (3.16) |

* Prod. consumida: energia produïda consumida. Si el valor no és igual a l’energia consumida, significa que en algun moment la producció no ha cobert el consum.

|  |  |
| --- | --- |
|  | (3.17) |

* Prod. sobrant (no consumida): energia produïda no consumida. A nivell d’hora, primer es fa servir aquesta energia per compensar el deute d’energia consumida de la xarxa. La restant, és el balanç net horari on s’aplica la compensació econòmica.

|  |  |
| --- | --- |
|  | (3.18) |

* Balanç: balanç net horari.

|  |  |
| --- | --- |
|  | (.19) |

L’algoritme el fa servir per prendre les decisions, per tant, necessitarem calcular-lo a temps real. Hi ha dues formes de fer-ho.

La primera és projectant al passat. A cada mostra calcules l’energia que hi ha hagut entre l’última i l’actual. Al canviar d’hora, l’energia en aquell instant serà 0, ja que l’energia calculada amb la mostra seria de l’hora anterior.

|  |  |
| --- | --- |
|  | (3.20) |

L’altra opció és projectant al futur. A cada mostra calcules l’energia que hi haurà entre l’actual i la següent. Al canviar d’hora, l’energia en aquell instant serà .

|  |  |
| --- | --- |
|  | (3.21) |

13 *h*

t

13 *h*

t

Figura 3.8. Formes de calcular el balanç

S’ha fet servir la segona forma ja que el factor predictiu et permet anticipar-te i millorar funcionament (a la pàgina 26 hi ha un exemple clar dels efectes de cada un).

* Deguda a la xarxa: part negativa del balanç. Igual que en el balanç, és interessant calcular-ho a temps real, a part de fer el càlcul a nivell d’hora.

|  |  |
| --- | --- |
|  | (3.22) |
|  | (.23) |

* Disponible (del balanç): part positiva del balanç. Igual que en el balanç, és interessant calcular-ho a temps real, a part de fer el càlcul a nivell d’hora.

|  |  |
| --- | --- |
|  | (.24) |
|  | (3.25) |

* Retornada a la xarxa: mai serà més gran que l’energia consumida de la xarxa.

|  |  |
| --- | --- |
|  | (3.26) |

* Consum màxim (ideal): energia consumida per la instal·lació si aquesta consumís el màxim d’energia produïda possible donades les càrregues.

|  |  |
| --- | --- |
|  | (.27) |

* Sobreeixida: energia del balanç disponible que no s’hauria pogut consumir degut a la manca de càrrega. És a dir, energia produïda que excedeix el consum màxim, i que no s’ha gastat en compensar el deute amb la xarxa.

|  |  |
| --- | --- |
|  | (3.28) |

* No aprofitada: energia del balanç disponible que s’hauria pogut consumir. És a dir, energia que ha faltat per arribar al consum màxim.

|  |  |
| --- | --- |
|  | (3.29) |

A la pràctica, treballarem en el domini discret. Això vol dir que, quan haguem de calcular qualsevol de les integral anteriors, ho farem de la següent forma:

|  |  |
| --- | --- |
|  | (.30) |

### Traduccions

La traducció a l’anglès dels termes d’energia definits anteriorment i els noms que es fan servir a les equacions són:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
| Terme | |  | Formula | |
| Català | Anglès |  | Català (document) | Anglès (codi i simulador) |
| Sistema | System |  | *eSI* | *energySY* |
| Produïda | Produced |  | *eP* | *energyP* |
| Prod. consumida | Prod. consumed |  | *ePC* | *energyPC* |
| Prod. no cons. o sobrant | Prod. left |  | *ePnC o ePS* | *energyPL* |
| Xarxa | Grid |  | *eX* | *energyG* |
| Xarxa durant prod. | Grid while prod. |  | *eP* | *energyGP* |
| Xarxa durant no prod. | Grid while not prod. |  | *enP* | *energyGnP* |
| Consumida | Consumed |  | *eC* | *energyC* |
| Càrrega base | Base load |  | *eCB* | *energyLB* |
| Càrrega 1 | Load 1 |  | *eC1* | *energyL1* |
| Càrrega 2 | Load 2 |  | *eC2* | *energyL2* |
| Consum màxim | Max. consumption |  | *eCM* | *energyCM* |
| Balanç | Balance |  | *eB* | *energyB* |
| Deguda a la xarxa | Grid debt |  | *eXD* | *energyGB* |
| Disponible (del balanç) | Available balance |  | *eBD* | *energyAB* |
| Sobreeixida | Surplus |  | *eS* | *energyS* |
| No aprofitada o perduda | Lost |  | *ePe* | *energyL* |
| Retornada a la xarxa | Returned to grid |  | *eXR* | *energyGR* |

Taula 3.2. Traduccions i nomenclatures – Energies

### Segmentació visual

Una bona forma visual de representar aquesta classificació/segmentació és a través de blocs, on cada bloc té la seva segmentació a la dreta.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sistema | Produïda | Prod. no cons. | | Balanç disponible | Sobreeixida |
| No aprofitada |
| Retornada xarxa | |
| Prod. consumida | Consumida | Càrrega base | |
| Xarxa | Xarxa fora prod. | Càrrega 1 | |
| Xarxa durant prod. | Càrrega 2 | |

Figura 3.9. Segmentació visual de l’energia del sistema

### Exemples

Per facilitar la lectura de les gràfiques que hi ha al llarg del treball i exemplificar els conceptes anteriors, a continuació es mostren i comenten diverses gràfiques generades a partir de dades reals.

Consum màxim d’energia:visualment queda molt clar el càlcul, és l’àrea inferior que queda al sobreposar la potència de producció i el consum de totes les cargues engegades.

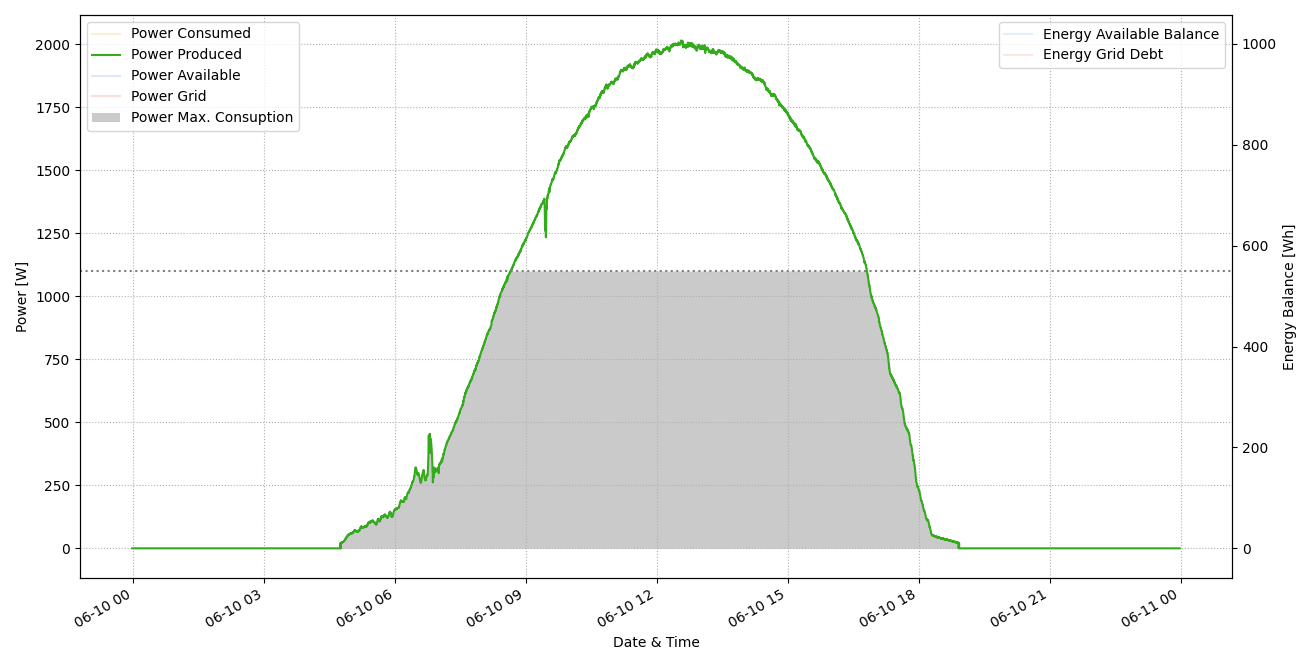


Figura 3.10. Exemple – Consum màxim d’energia a partir d’àrees

Energia sobreeixida: Amb aquesta gràfica anterior també podem veure fàcilment l’energia que ha sobreeixit. És l’àrea en blanc que queda sobre l’àrea del consum màxim.

Energia perduda:Si a la gràfica anterior hi sobreposem les àrees de les càrregues podrem veure, aproximadament, l’energia perduda (àrees grises). Diem “aproximadament” perquè s’han de considerar els trossos de les àrees de les càrregues que sobrepassen la producció, ja que aquestes poden haver estat compensades precisament per l’energia de les àrees grises.

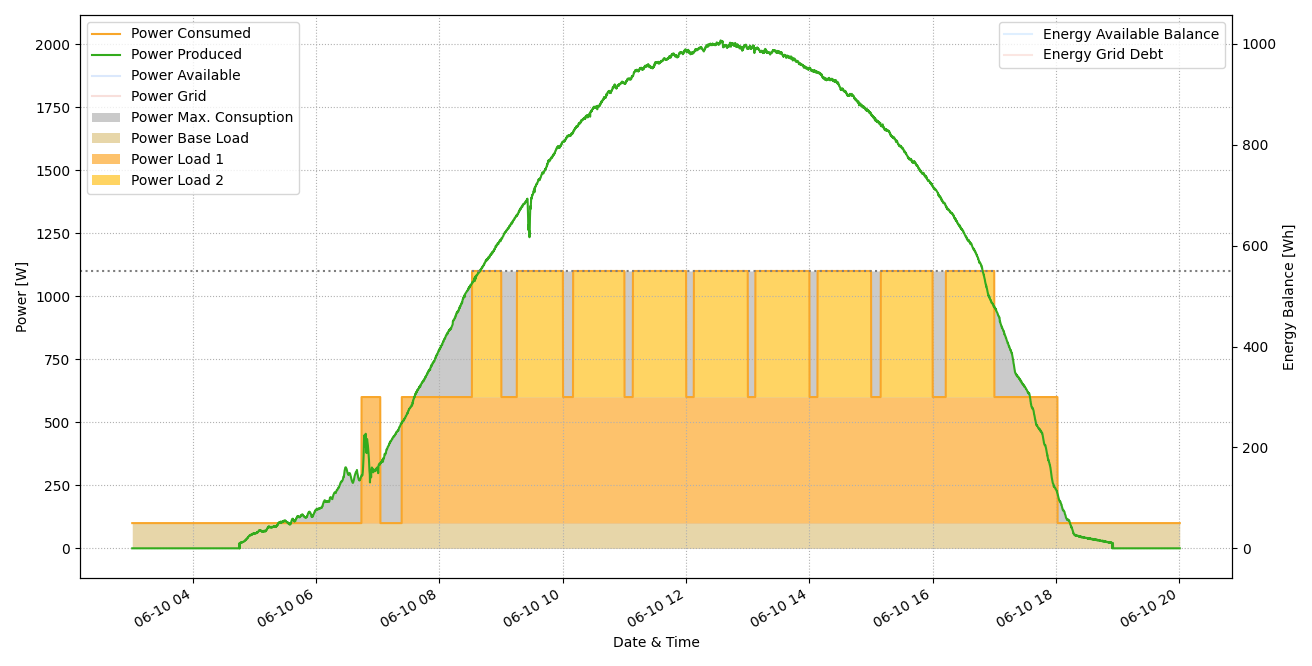


Figura 3.11. Exemple - Energia perduda a partir d’àrees

Energia disponible i energia deguda a la xarxa: La millor forma de veure aquestes dades és a temps real juntament amb les potències. A més a més amb aquesta vista podem veure fàcilment com es comporta el sistema i comparar-ho amb el que esperaríem.

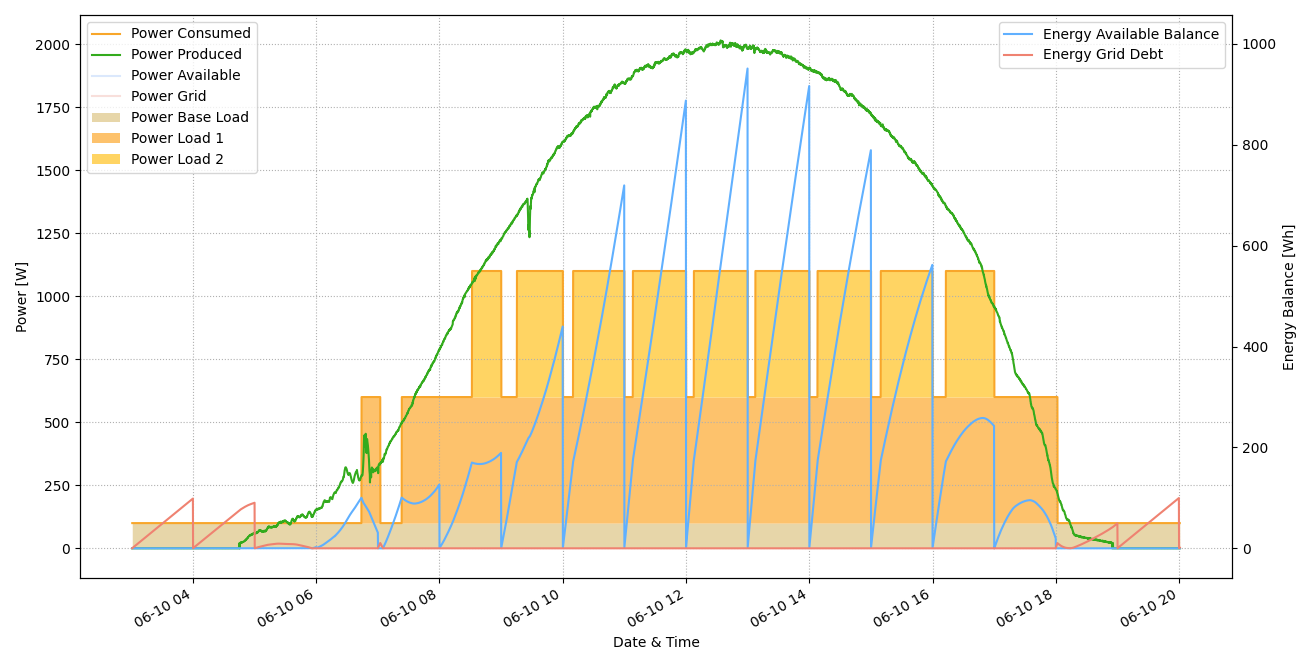


Figura 3.12. Exemple – Energia disponible del balanç i deguda a la xarxa – Vista comportament

Representació per hores: Al tenir tants conceptes d’energia, necessitem trobar una forma clara i entenedora que, alhora, mostri el màxim d’informació. L’opció triada ha estat començar representant les energies del model energètic plantejat a l’inici, Figura 2.1, i posteriorment anar-les segmentant.

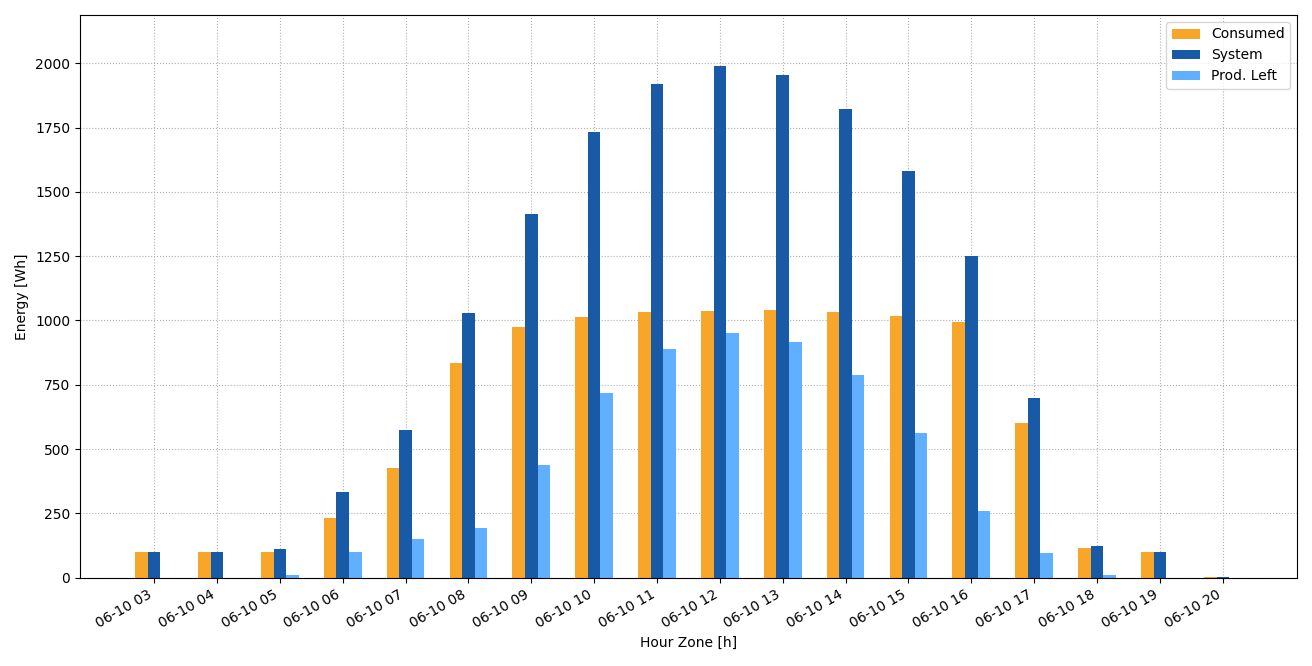


Figura 3.13. Exemple – Resultats per hores de les energies (vista model)

Amb la primera segmentació podem començar a veure coses interessants:

* Quanta energia ens consumeix cada càrrega.
* Si l’energia ha esta subministrada per la producció, la xarxa o totes dues fonts.
* Si s’ha pogut retornar tota l’energia o quanta.
* Si ens ha sobrat energia.
* Si el sistema funciona correctament i retorna el màxim d’energia possible a la xarxa abans de tenir balanç disponible.

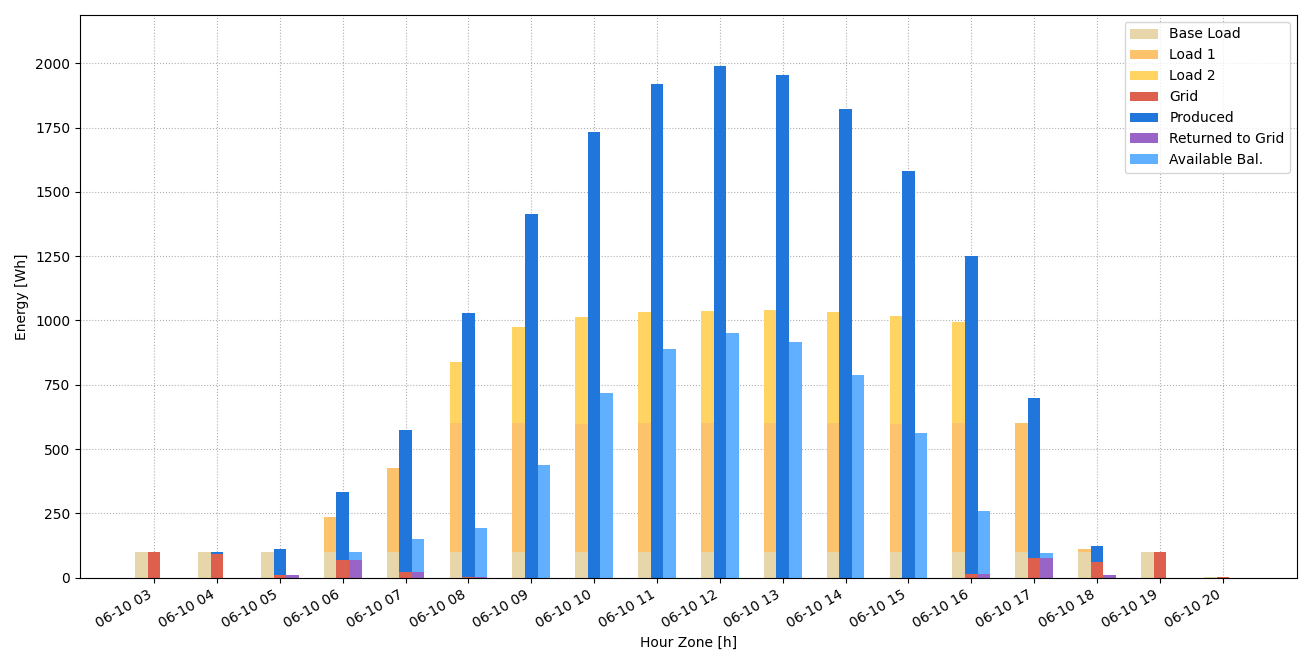


Figura 3.14. Exemple – Resultats per hores de les energies (1a segmentació)

Finalment, amb la segona segmentació podem veure el mateix que abans i algunes coses noves:

* Quanta energia no s’ha aprofitat.
* Quanta energia no ha sobreeixit.
* *Com pot ser que haguem consumit més energia de la màxima?* En els instants de no producció la càrrega base segueix consumint. Només té sentit mirar-ho en hores de producció.

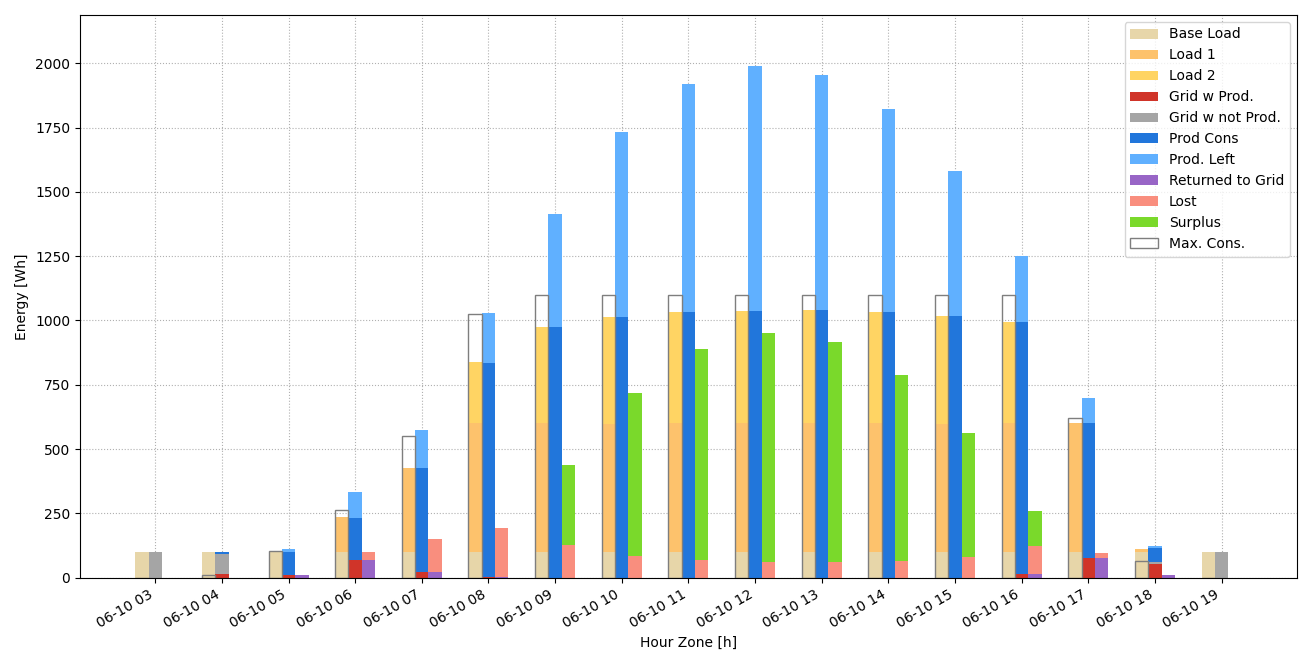


Figura 3.15. Exemple – Resultats per hores de les energies (2a segmentació)

Representació total:Igual que en la representació per hores, també podem representar el total del dia/setmana/simulació, ... Ara, en aquesta gràfica hem d’estar atents ja que, degut al balanç net horari, hi ha una cosa que pot semblar incorrecte:

* *Com pot ser que no haguem retornat tota l’energia a la xarxa si ens n’ha sobrat?* Perquè el balanç net horari no compensa l’energia consumida entre hores. És a dir, hi ha hores que no s’ha pogut retornar tota i no ens n’ha sobrat, i altres que si.

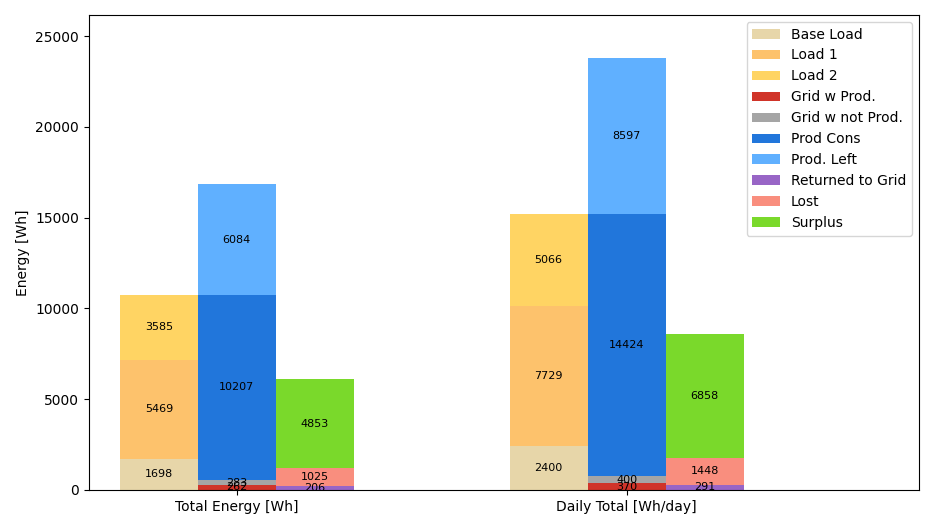


Figura 3.16. Exemple – Model energètic per hores (2a segmentació)

## Per comparació

### Definicions

Per poder compara els diversos algoritmes a partir dels objectius marcats, s’han definit els següents termes representatius[[6]](#footnote-7):

* Eficiència del consum: percentatge d’energia que s’ha aprofitat respecte el consum màxim ideal. El percentatge restant és l’energia perduda.

|  |  |
| --- | --- |
|  | (.31) |

* Eficiència del retorn a la xarxa: percentatge d’energia retornada a la xarxa respecta la consumim de la xarxa durant producció, ja que fora de producció només hi haurà consum base, el qual el sistema no controla.

|  |  |
| --- | --- |
|  | (.32) |

* Balanç monetari:balanç energètic net a nivell monetari. Quants diners suma o resta aquell interval de temps a la factura.

|  |  |
| --- | --- |
|  | (.33) |

* Commutacions: numero de vegades que la càrrega s’ha engegat i apagat.

|  |  |
| --- | --- |
|  | (.34) |

* Mostres o hores (amb la càrrega X) engegada

Quan a algun dels termes s’hi afegeix la paraula diàries, significa que representa un interval de 24h. Si no s’indica que la dada és diària, l’interval serà el transcorregut/simulat.

Observació

Un error habitual és pensar que l’eficiència total és pot calcular fent la mitjana d’eficiències de cada hora.

|  |  |
| --- | --- |
|  | (3.35) |

Però no és el cas. El càlcul d’una eficiència és pot escriure com:

|  |  |
| --- | --- |
|  | (3.36) |

I podem comprovar que no és el mateix l’eficiència total que la mitjana d’eficiències.

|  |  |
| --- | --- |
|  | (3.37) |

## Traduccions

La traducció a l’anglès dels termes definits anteriorment i els noms que es fan servir a les equacions són:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  |  | |
| Terme | |  | Formula | |
| Català | Anglès |  | Català (document) | Anglès (codi i simulador) |
| Eficiència del consum | Consumption efficiency |  | *efC* | *efficC* |
| Eficiència del retorna xarxa | Grid return efficiency |  | *efG* | *efficG* |
| Balanç monetari | Balance |  | *balanç* | *balance* |
| Commutacions | Commutations |  | *commut* | *commut* |
| Hores engegada | Hours on |  | *horesOn* | *hoursOn* |
| Commutacions diàries | Daily Commutations |  | *commutD* | *commutD* |
| Hores engegada diàries | Daily hours on |  | *horesonOnD* | *hoursOnD* |
| Mostres engegada | Samples On |  | *mostresOn* | *samplesOn* |

Taula 3.3. Traduccions i nomenclatures – Comparació

# Algoritmes de control

Partint del model simplificat explicant en l’apartat 2.2, s’han dissenyat, i posteriorment simulat i implementat, tres algoritmes diferents per controlar la instal·lació, aquests són:

* Control amb histèresi
* Control a partir del temps mínim engegada
* Control predictiu (temps de consum)

Tots tres algoritmes prenen les decisions a temps real (a cada mostra) a partir del balanç i l’estat de les càrregues (entrades) per engegar o apagar les càrregues (sortides).

## Histèresi

Ha estat el primer algoritme plantejat ja que és molt senzill, és un concepte conegut en el mon tècnic i científics [3], i es utilitzat en sistemes de control, com pot ser en termòstat.

El terme histèresi descriu el comportament d’un sistema el qual depèn tant de factors externs com de l’estat anterior d’aquest. Com a conseqüència, el sistema pot conservar l’estat en absència de l’estímul.

### Una càrrega

Per començar, plantejarem l’algoritme per controlar una sola càrrega, i posteriorment l’ampliarem a dues. El seu comportament és el següent:

* Si el balanç és més gran que el llindar superior, engega la càrrega
* Si el balanç és més petit que el llindar inferior, apaga la càrrega

Aquest comportament es pot representar gràficament amb una corba d’histèresi:

Balanç

Estat Càrrega

Figura 4.1. Corba d’histèresi del control d’una càrrega

També el podem representar el comportament amb una màquina d’estats.

Apagada

Engegada

engegar càrrega

apagar càrrega

Figura 4.2. Histèresi – Màquina d’estats per controlar una càrrega

### Dues càrregues

Per controlar dues càrregues simplement les controlem per separat, cadascuna amb els seus llindars.

### Observacions

Si volem que una càrrega que una tingui preferència, per exemple, que la segona no s’encengui fins que la primera s’hagi encès, hem de fer que els llindars de la segona càrrega siguin superiors als de la primera.

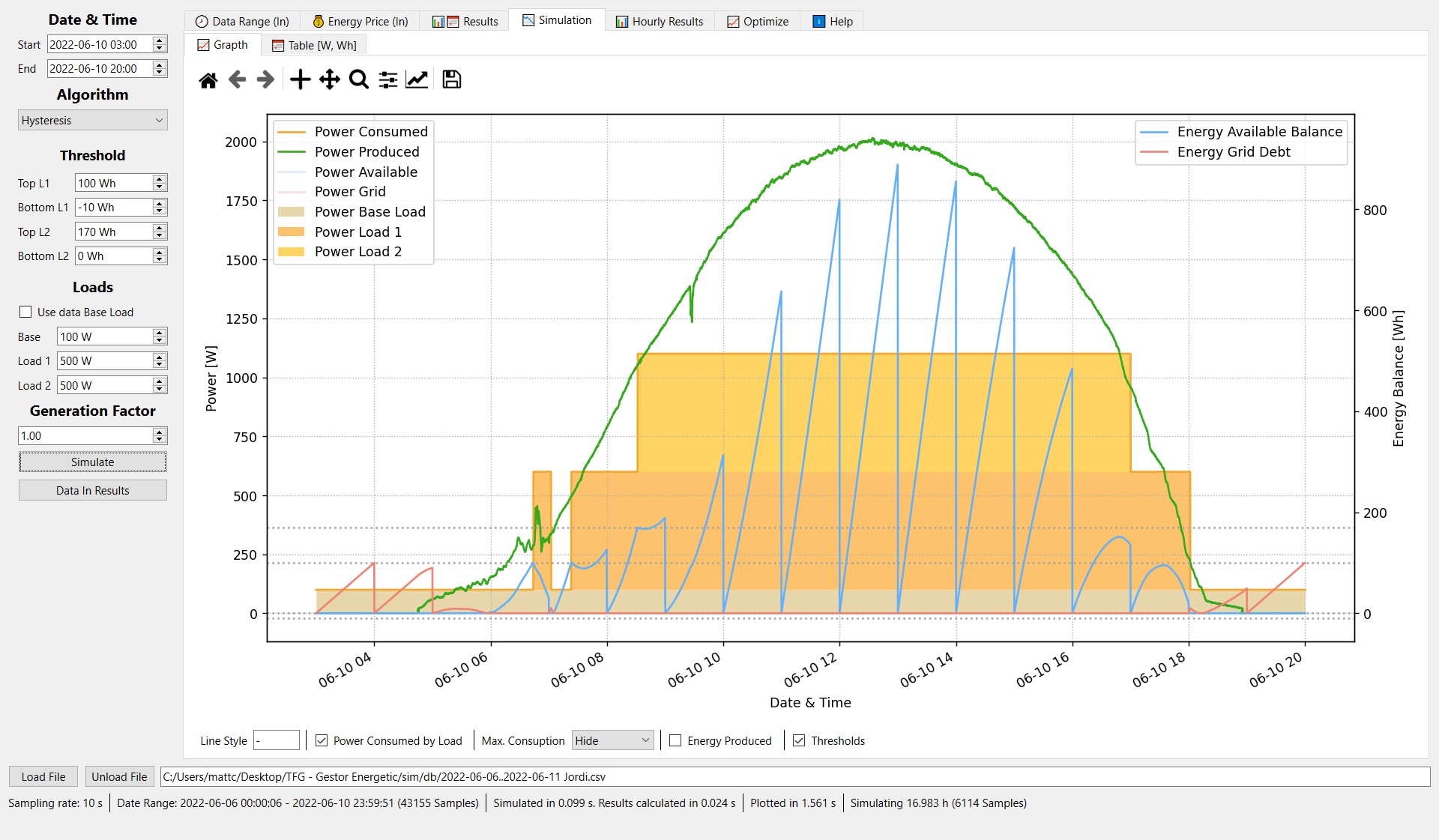


Figura 4.3. Histèresi – 1a càrrega amb preferència d’engegada respecta la 2a

En canvi, si les dues càrregues tenen el mateix llindar, és con si tinguéssim un sistema amb una sola càrrega equivalent a la suma de els dues. En les següents simulacions podem comprovar-ho.

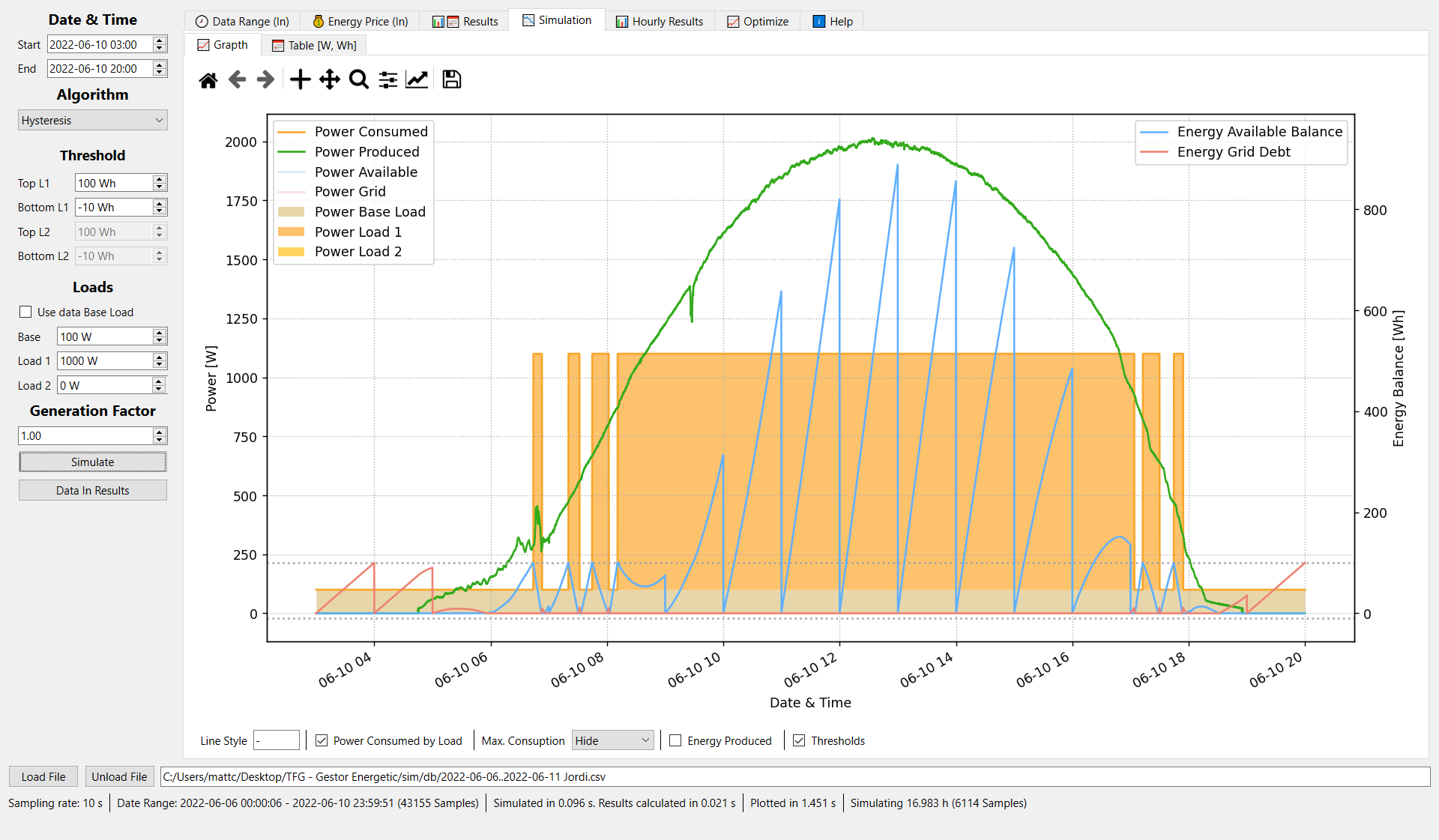
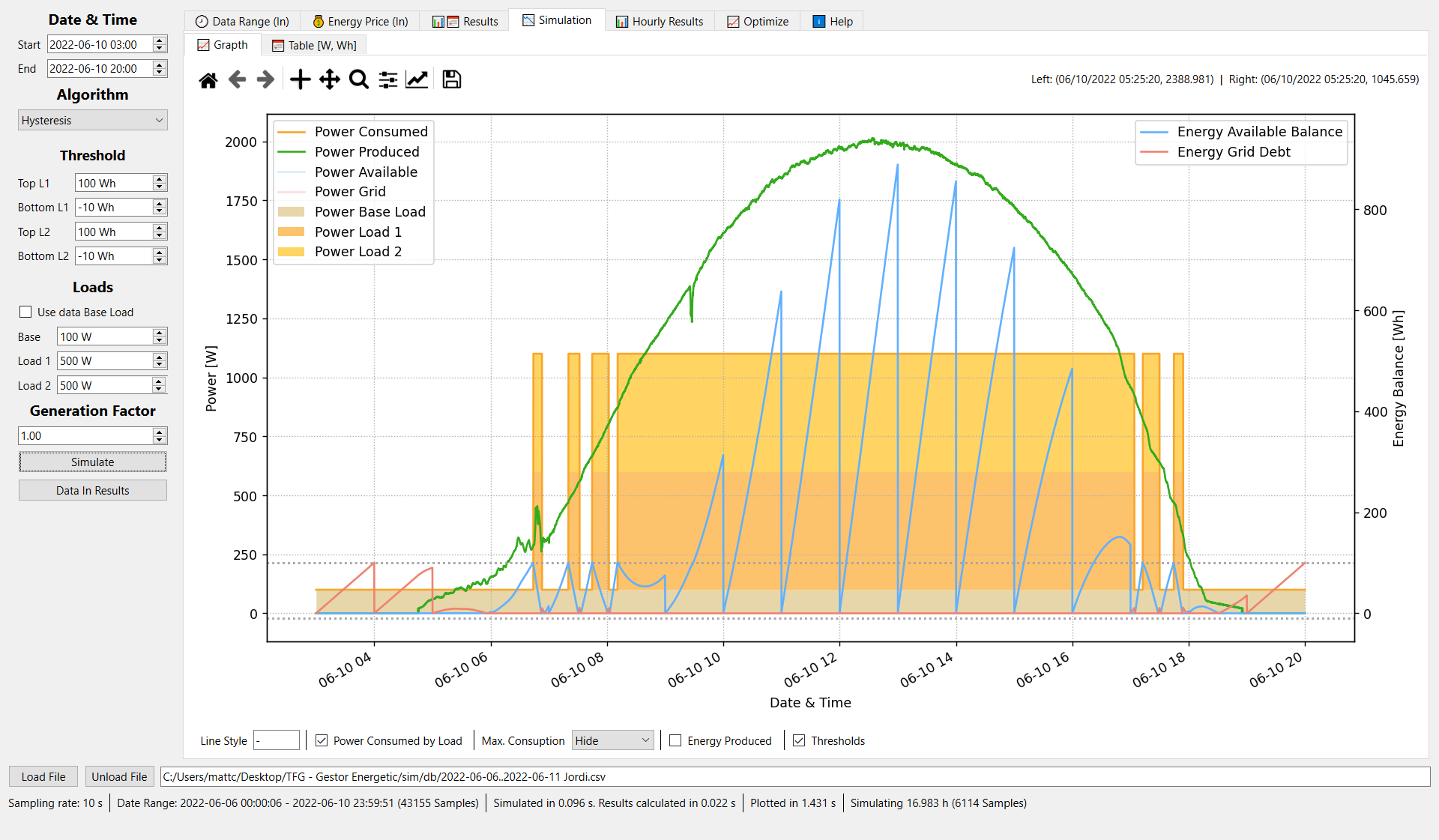


Figura 4.4. Histèresi – 2 càrregues amb els mateixos llindars equivalen a 1

A l’apartat 3.2 s’han definit dues formes de calcular el balanç a temps real i s’ha dir que s’ha fet servir la segona. Amb una histèresis on el llindar baix és 0 podem veure clarament perquè. Si féssim servir la primera forma de calcular el balanç, a cada hora tindríem commutacions degut a que el càlcul donaria 0.

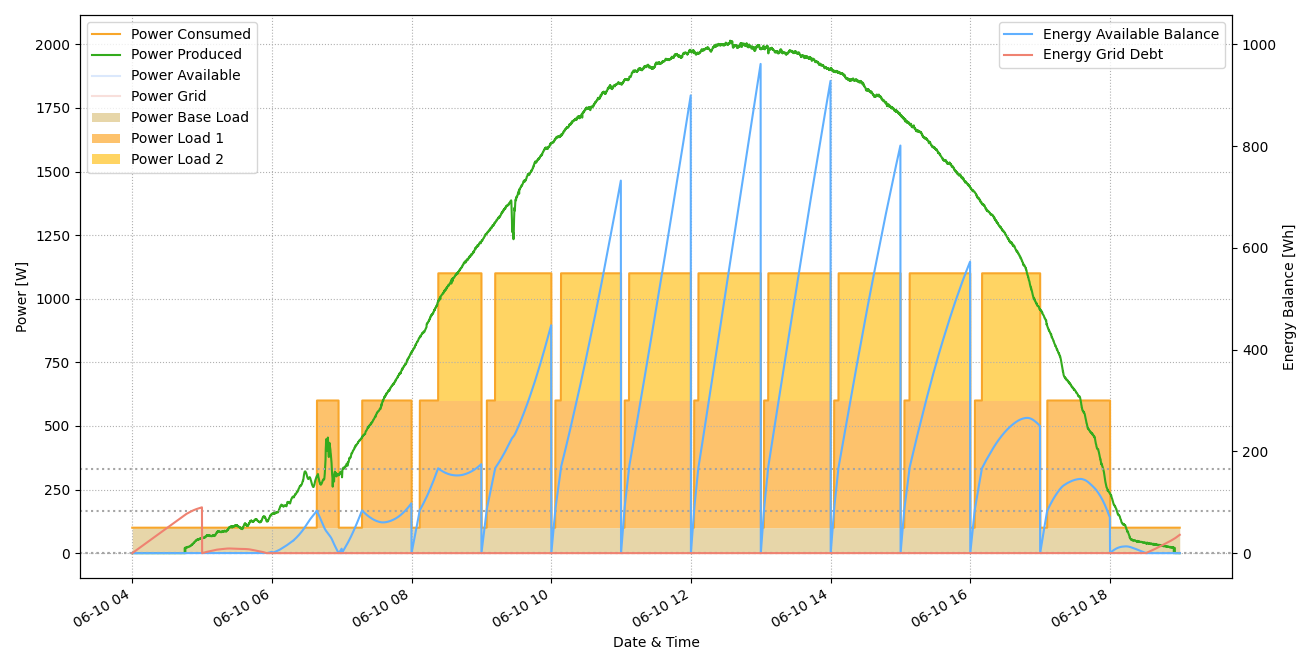


Figura 4.5. Histèresis – Llindar baix a 0 i càlcul de balanç projectant al passat

En canvi, de l’altre forma només tindrem commutacions en els canvis d’hora si preveiem que el balanç en serà 0 o negatiu.

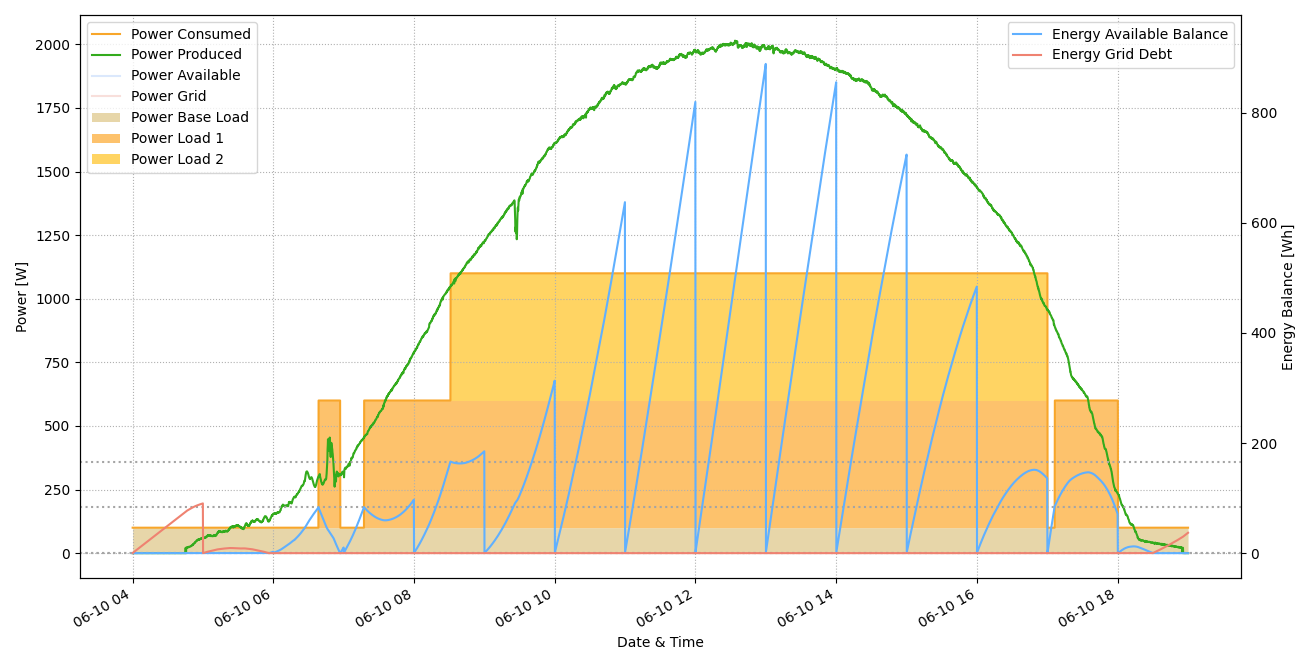


Figura 4.6. Histèresis – Llindar baix a 0 i càlcul de balanç projectant al futur

## Temps mínim engegada

Aquest algoritme l’ha proposat un professor del departament que l’estava utilitzant. Com que només l’havia plantejat per a una, s’ha ampliat a dues.

Partim de la idea que a les càrregues se’ls hi pot indicar que s’engeguin durant un certs temps, per exemple, engega’t durant 2 minuts. Si la càrrega ja està engegada, dir-li la mateixa instrucció únicament posa el temporitzador al nou valor indicat.

La idea és dir-li a la càrrega/es que s’engegui durant un temps mínim, si el temps en que tarda a consumir el balanç disponible es igual o més alt a el temps mínim. El càlcul del temps en que és tardarà a consumir el balanç disponible dependrà de les càrregues que es vulguin engegar:

|  |  |
| --- | --- |
|  | (.1) |

En el nostre cas, s’han fet servir relés Shelly que ja tenen la funcionalitat. Però en el cas en que la càrrega no tingui la funcionalitat de temporització, l’algoritme haurà d’implementar-la, que és el que s’ha hagut de fer per simular-ho.

### Una càrrega

Si només controlem una sola càrrega, és ben senzill: si i el temps per consumir el balanç disponible és major o igual al temps mínim li diem a la càrrega que s’encengui.

### Dues càrregues

En canvi, si controlem dues cargues la cosa es complica i es pot resoldre de múltiples maneres. La més simple és engegar la 2a càrrega només si la 1a està engegada, ja que ens pot interessar donar preferència a una de les dues. L’ordre d’accions seria:

1. Primer mirem si podem engegar les dues càrregues alhora.
2. Si no podem engegar les dues, mirem si podem engegar la 1a.

Però si no ens importa la preferència, una altre solució seria considerar totes les situacions que podem tenir:

1. Primer mirem si podem engegar les dues càrregues alhora.
2. Si no podem engegar les dues càrregues alhora però si qualsevol de les dues, si no engeguem l’altre, necessitarem fixar un mètode per discernir quina de les dues engegar:
   1. Tirar la càrrega a engegar per ordre.
   2. La que tardi més/menys en consumir-ho, és a dir, a la càrrega més petita/gran.
3. Finalment, quedaria el cas en que només en podem engegar una.

A aquestes dues implementacions les anomenarem: simplificada i completa. Amb la completa tenint la variant A i B en funció del mètode de tria.

### Observacions

Les tres possibles implementacions és solapen en comportament en funció de la combinació de càrregues. Si la càrrega petita és la primera, la implementació completa A equival a la simple. Podem veure-ho en les simulacions, Figura 4.8. Això és degut a que:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | |
|  | Càrrega a engegar |  | Implementació | |
| Situació |  | Completa A | Simplificada |
| 1a | 1a i 2a |  | - | - |
| 2a | 1a |  | degut al ordre | la 2a no és pot si la 1a no està engegada |
| 3a | 1a |  | única que pot consumir-ho en el temps mínim |

Taula 4.1. Temps mín. engegada – Implementacions amb el mateix comportaments quan C1 < C2

Si la càrrega petita és la segona, la implementació completa A equival a la completa B. Podem veure-ho en les simulacions, Figura 4.9. Això és degut a que:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | |
|  | Càrrega a engegar |  | Implementació | |
| Situació |  | Completa A | Completa B |
| 1a | 1a i 2a |  | - | - |
| 2a | 1a |  | degut al ordre | consumeix més ràpid |
| 3a | 1a |  | única que pot consumir-ho en el temps mínim | |

Taula 4.2. Temps mín. engegada – Implementacions amb el mateix comportaments quan C1 > C2

Finalment, si les dues càrregues són iguals, tots tres és comporten igual. Això és degut a que:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  | |  |
|  | Càrrega a engegar |  | Implementació | |  |
| Situació |  | Completa A | Completa B | Simplificada |
| 1a | 1a i 2a |  | - | - | - |
| 2a | 1a |  | degut al ordre | degut al ordre de comprovació | la 2a no és pot si la 1a no està engegada |
| 3a | 1a |  |

Taula 4.3. Temps mín. engegada – Implementacions amb el mateix comportaments quan C1 = C2

Si representem visualment el solapament queda de la següent forma:

Completa A

Completa B

Simple

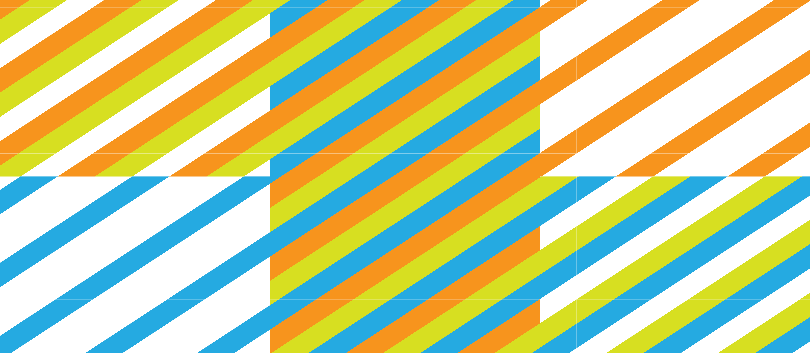


Figura 4.7. Temps mín. engegada – Solapament de comportament de les implementacions

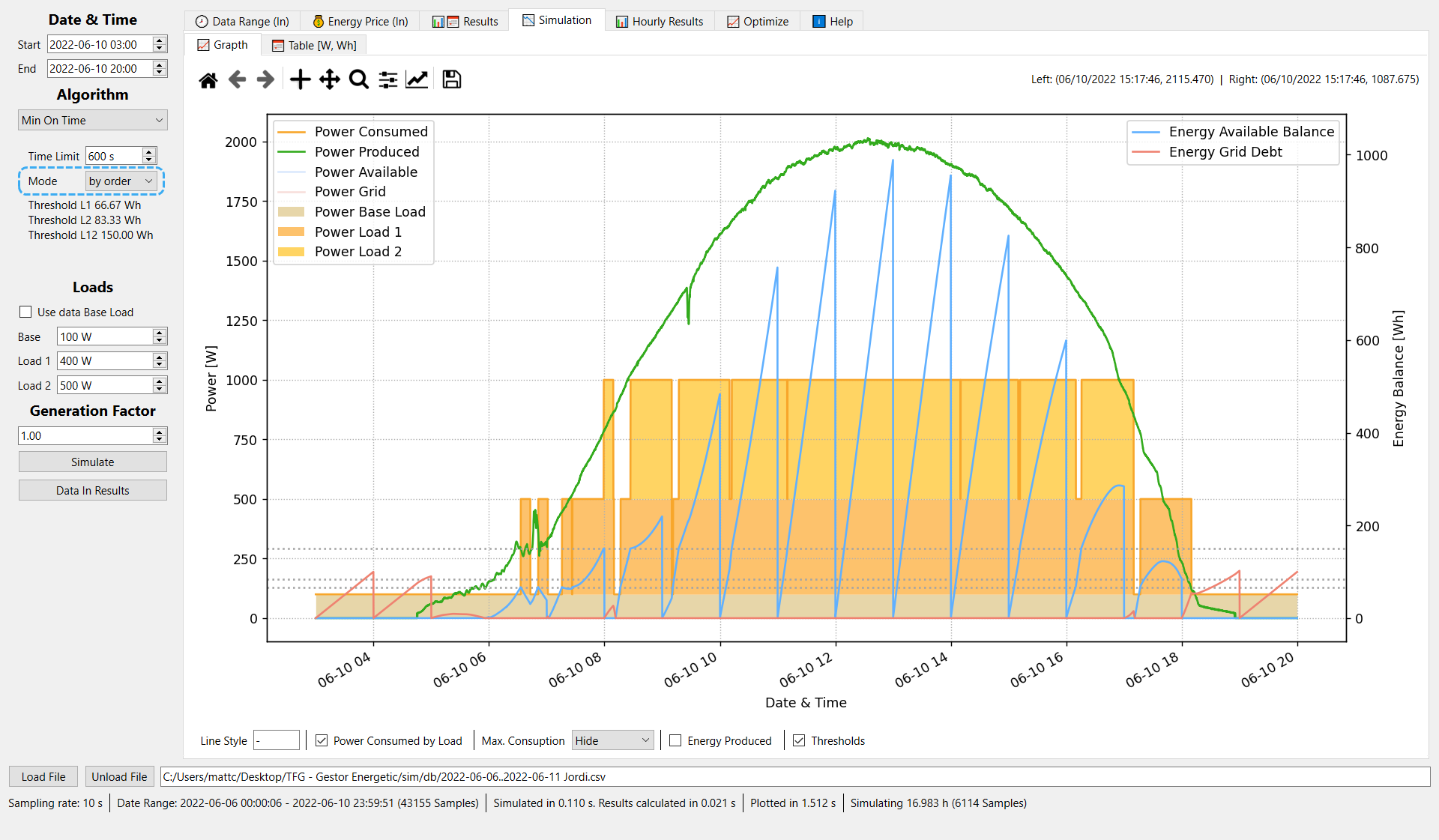
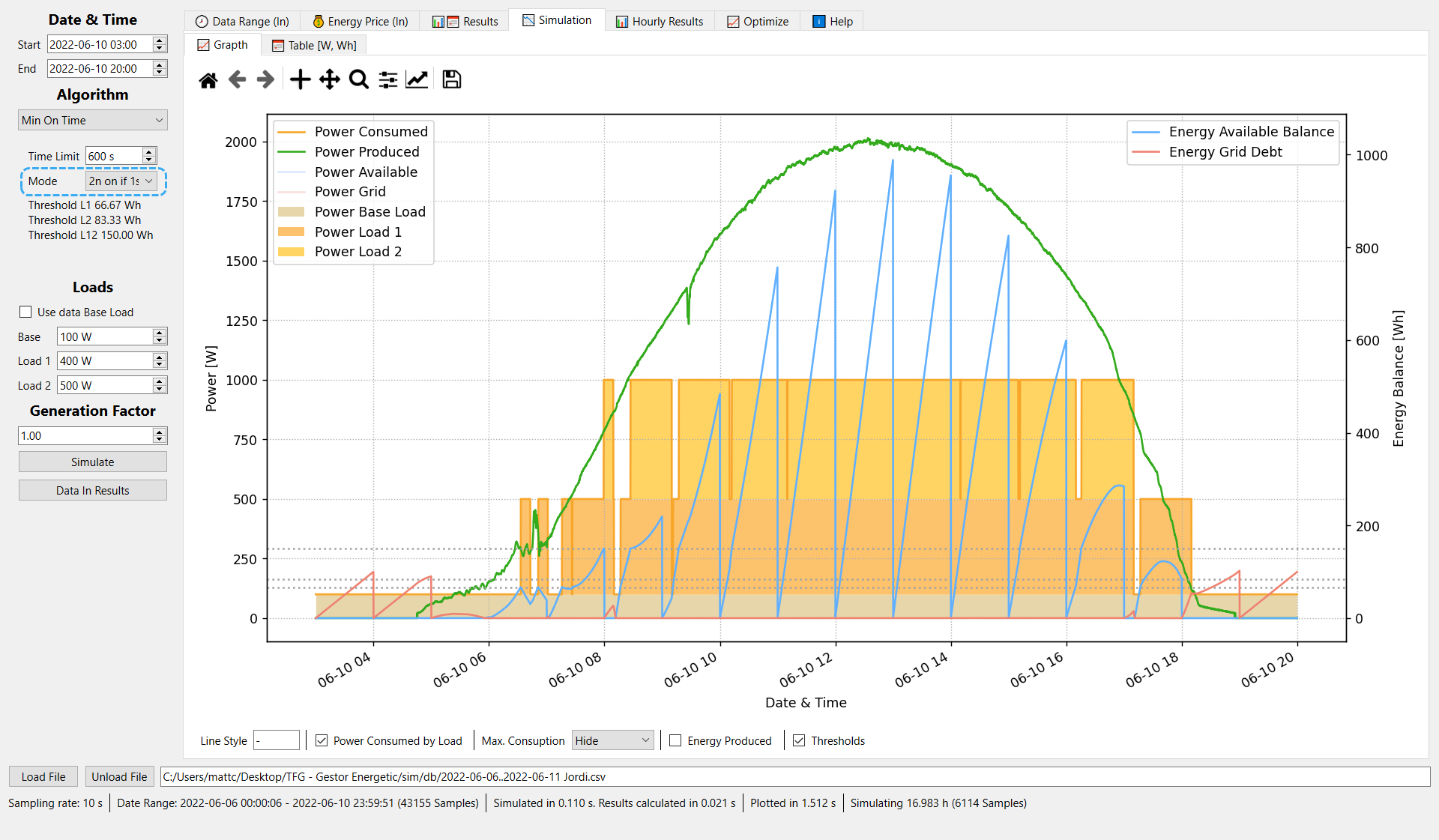


Figura 4.8. Temps mín. engegada – Implementacions amb el mateix comportament quan C1 < C2

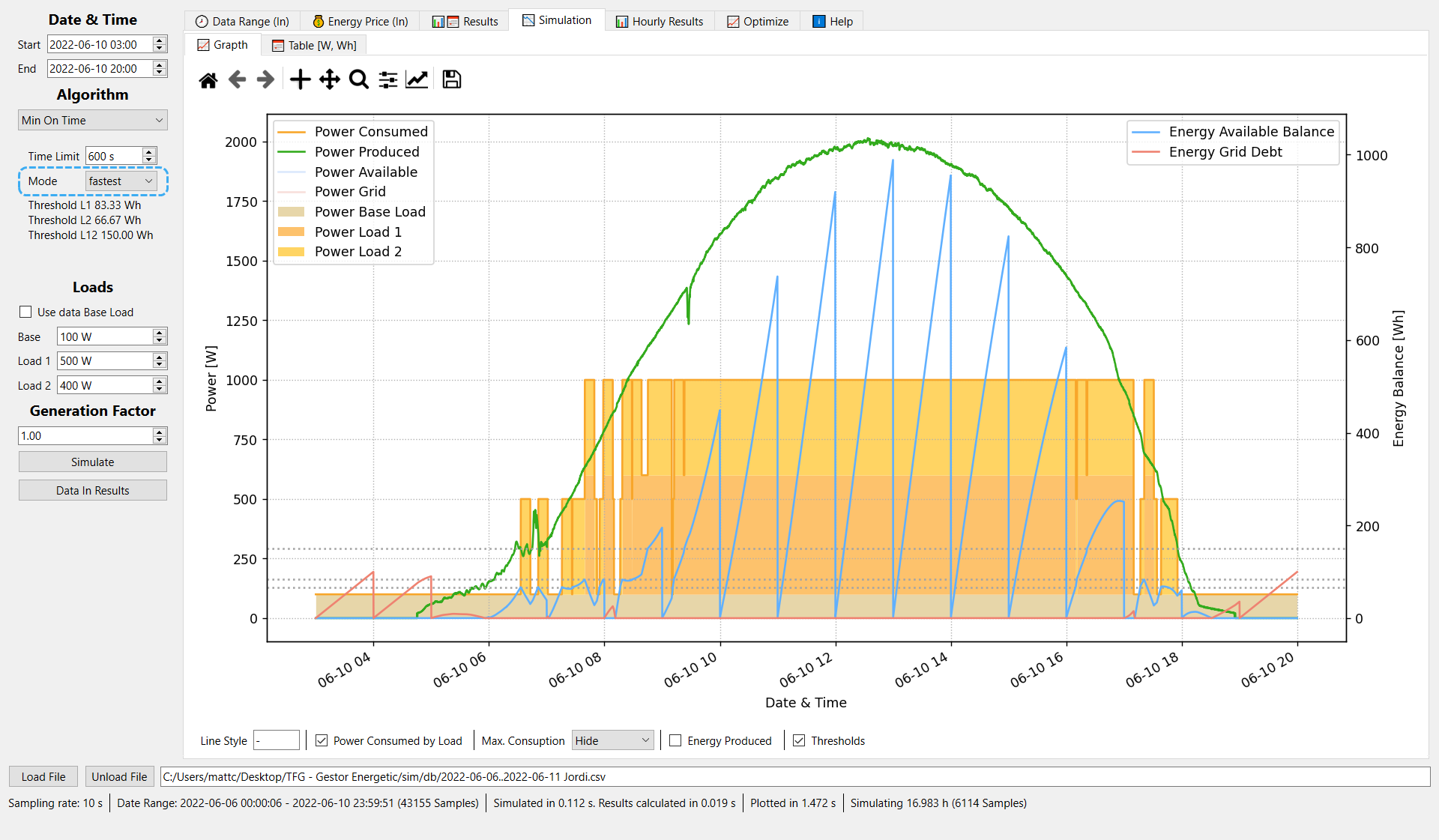
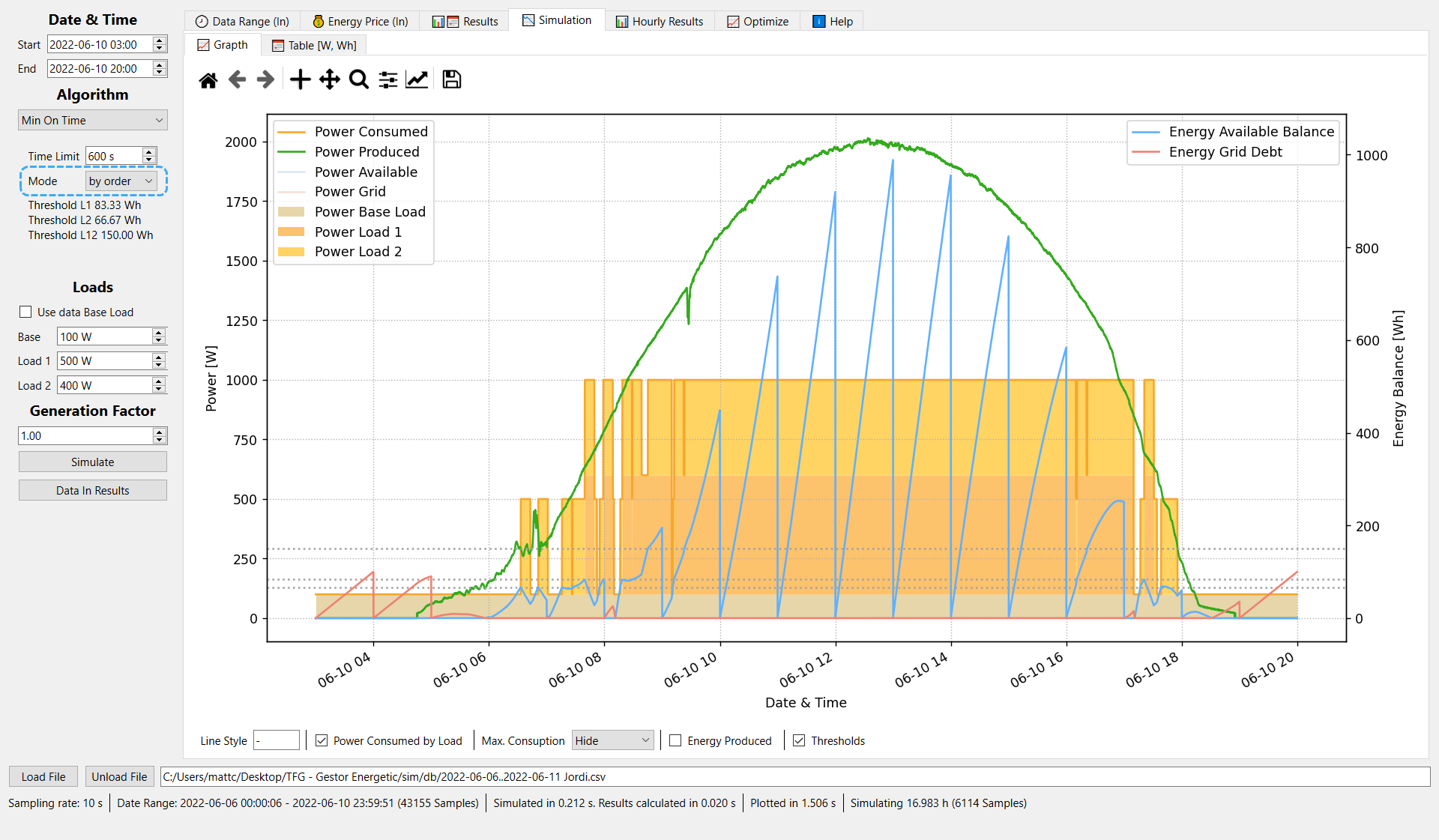
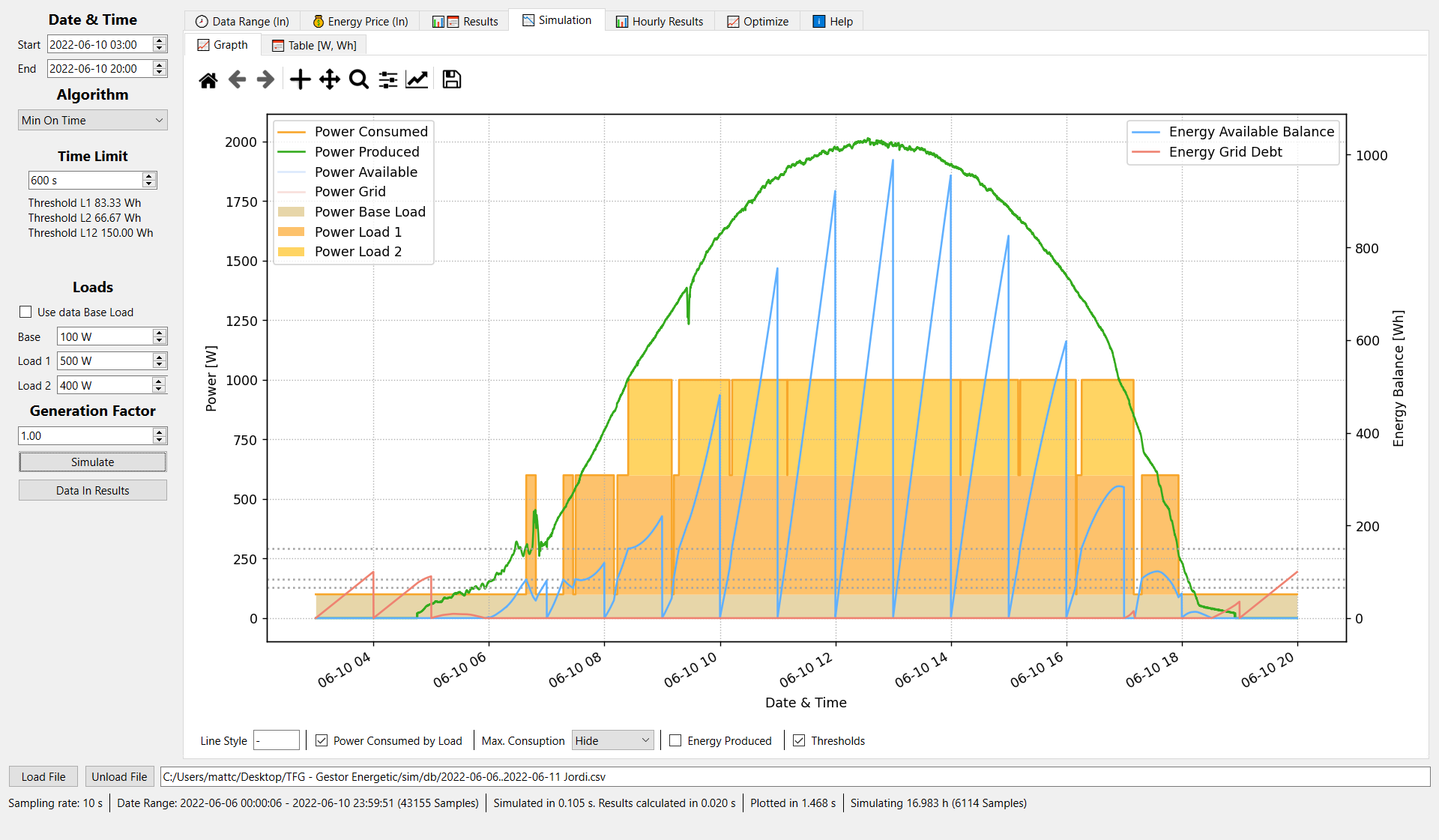


Figura 4.9. Temps mín. engegada – Implementacions amb el mateix comportament quan C1 > C2

Una altre característica és que el temps mínim és pot traduir a un llindar energètic, el qual depèn de la càrrega.

Quan fem servir l’algoritme simple, el comportament que té és com si fos una histèresi on el llindar baix i l’alt són iguals, però, un cop passat per sota del llindar inferior, la càrrega es manté engegada el temps mínim, evitant així múltiples commutacions.



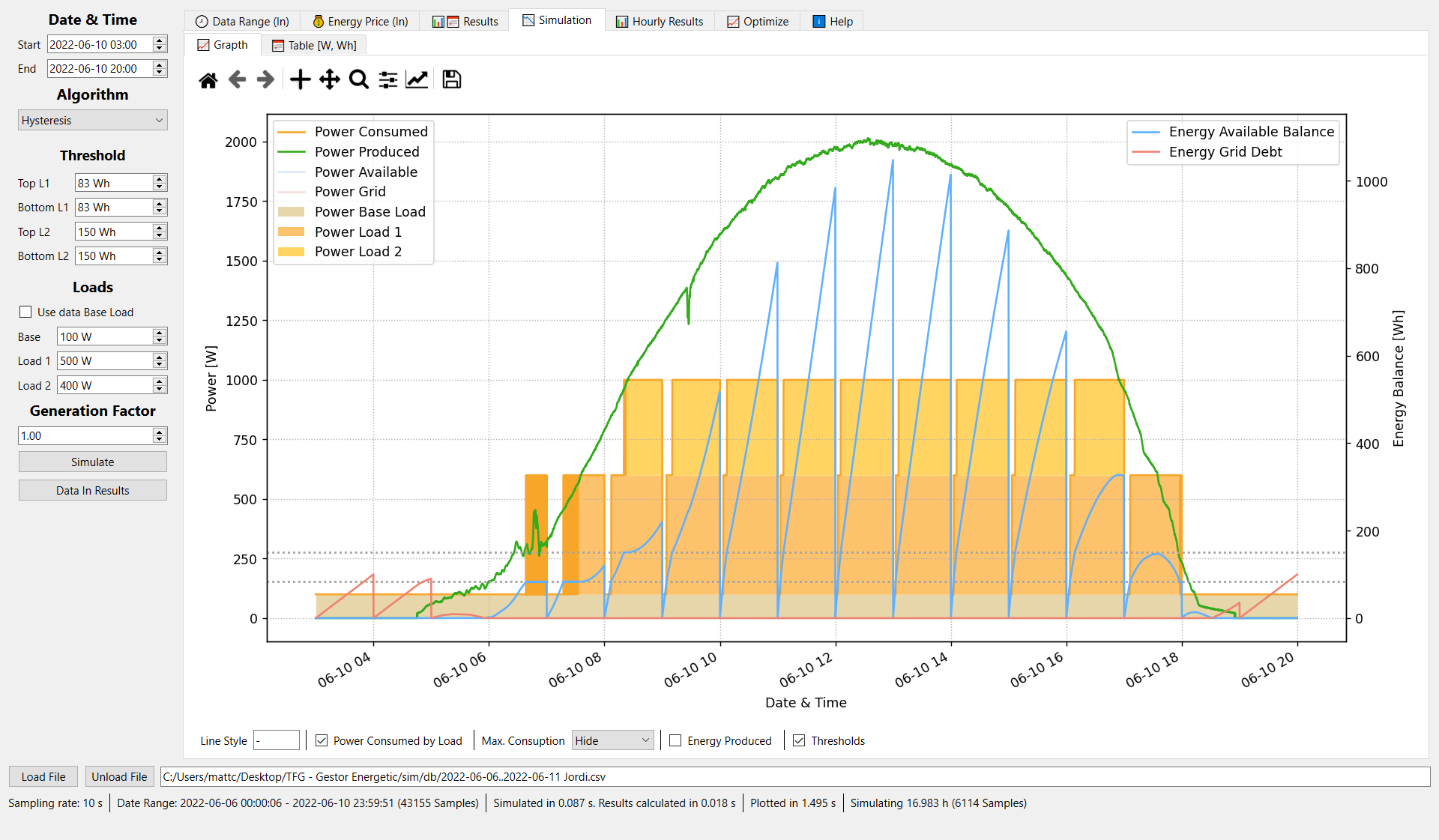


Figura 4.10. Histèresi (segona part) amb mateixos llindars que Temps mínim engegat (primera part)

## Predictiu (Temps de consum)

Ha estat l’últim dissenyat ja que primer es volien implementar algoritmes no predictius. L’algoritme controla les càrregues en funció del balanç net horari que preveu que hi haurà en acabar l’hora, i del temps que restant per d’acabar l’hora.

### Com predir el balanç final

Per predir el balanç net horari s’han pensat tres formes. Aquestes són:

1. No fer cap “predicció” i considerar que el balanç actual serà el balanç final horari.
2. Projectant la potència disponible en el que queda d’hora.
3. A partir de la potència mitjana disponible calcular el balanç final.

Addicionalment, si volem predir el balanç net horari considerant que apaguem una càrrega, simplement s’ha de sumar tota l’energia que la càrrega no consumirà al balanç net horari predit.

|  |  |
| --- | --- |
|  | (.2) |

Projectant la potència disponible

El balanç net horari és la integral de la potència disponible en aquella hora, és a dir l’àrea. Simplement suposarem que la potència disponible actual seguirà constant en el que queda de temps. En termes d’àrea afegirem un rectangle.

|  |  |
| --- | --- |
|  | (.3) |

13 *h*

14 *h*

Potència

t

Figura 4.11. Predictiu – Predicció projectant la potència disponible

A partir de la potència mitjana disponible

Una integral és pot pensar coma a suma d’àrees positives i negatives delimitades per la funció. Aquestes àrees equivalen a l’àrea delimitada per la mitjana de la funció. La demostració és la següent:

|  |  |
| --- | --- |
|  | (4.4) |
|  | (4.5) |

El balanç net horari és la integral de la potència disponible en aquella hora. Així dons, podem calcular el balanç net horari a partir de la potència mitjana disponible.

13 *h*

14 *h*

Potència

t

Figura 4.12. Predictiu – Exemplificació de l’àrea de equivalent a la integral

Com que, la potència mitjana disponible d’aquella hora no la sabrem fins que acabem l’hora, l’aproximarem a partir de la potència mitjana disponible que portem fins el moment dins del interval horari.

|  |  |
| --- | --- |
|  | (4.6) |

La potència mitjana disponible fins el moment la podem calcular a partir del balanç actual.

|  |  |
| --- | --- |
|  | (4.7) |
|  | (4.8) |

13 *h*

14 *h*

Potència

t

Figura 4.13. Predictiu – Predicció a partir de la potència mitjana disponible

### Una càrrega

#### Plantejament

Partirem de l’algoritme més senzill:

* Si el temps en que es tarda a consumir el balanç net horari predit és igual o superior al temps restant, engeguem la càrrega.
* Si el balanç és 0, apaguem la càrrega

Idealment, el comportament seria el que mostra la següent figura:

13 *h*

14 *h*

balanç

t

càrrega encesa

Figura 4.14. Predictiu base – Situació ideal

Ara bé, aquest primer plantejament presenta dos problemes molt importants, que es poden veure en dues situacions extremes:

1. Si la producció no supleix el consum, farem moltes commutacions. Ja que, sí que fem servir la xarxa com una bateria, on guardem energia i després la consumim, però no la fem servir per demanar “deute”.
2. Si la producció acaba sent més de la predita, pot ser que estiguem desaprofitant energia.

En la següent figura podem veure les dues situacions extremes.

13 *h*

14 *h*

t

**a) Producció < Consum**

13 *h*

14 *h*

t

**b) Producció > Consum**

balanç

càrrega encesa

Figura 4.15. Predictiu base – Situacions extremes que pot tenir

Solució a la situació a

En comptes d’apagar la càrrega quan el balanç arribi a 0, l’apagarem quan la predicció del balanç net horari final, contant que apaguéssim la càrrega, sigui igual o menor a 0.

Idealment, el que estem fent és aglutinar els múltiples consums en una sola commutació. Però a la realitat és un “arriscat”, si a final d’hora tenim algun consum espontani o la predicció és imprecisa, podem acabar molt probablement amb deute, és a dir balanç net horari negatiu.

13 *h*

14 *h*

t

**Ideal**

balanç

càrrega encesa

13 *h*

14 *h*

t

**Acabar amb balanç negatiu**

Figura 4.16. Predictiu base – Solució al problema a) i nou problema que presenta

Per minimitzar els cops que acabem amb deute, en comptes d’apagar la càrrega quan la predicció del balanç net horari final, contant que apaguéssim la càrrega, sigui igual o menor a 0, l’apagarem quan sigui igual o menor a un llindar positiu.

Al haver afegit el llindar positiu, també necessitarem un llindar d’engegada que sigui igual o més gran que aquest llindar d’apagada, per evitar tenir moltes commutacions a final d’hora.

Solució a la situació b

Per solucionar el segon cas, sense canviar la càrrega, el que hauria de passar és que la càrrega s’engegués abans.

En comptes d’engegar la càrrega quan el temps de consum sigui igual o major a el temps restant, l’engegarem quan sigui igual o major a un percentatge del temps restant. Això és com si tinguéssim un *offset* que es va fent petit a mesura que passa el temps.

13 *h*

14 *h*

t

13 *h*

14 *h*

t

**Amb offset**

Figura 4.17. Predictiu base – Solució al problema b) i nou problema que presenta

#### Versió final

Afegint totes les punts considerats al plantejament, l’algoritme final queda de la següent forma:

* Si el balanç net horari predit és major a un llindar, i el temps en que es tarda a consumir el balanç net horari predit és igual o major al temps que queda d’hora multiplicat per un factor entre 1 i 0, s’engega la càrrega.
* Si el balanç net horari final predit, contant que apaguéssim la càrrega, és igual o menor a un llindar mínim, s’apaga la càrrega.

La màquina d’estats és la següent:

Apagada

Engegada

engegar càrrega

apagar càrrega

Figura 4.18. Predictiu – Màquina d’estats per controlar una càrrega

### Dues càrregues

Per controlar dues càrregues, podríem pensar que és pot fer el mateix que en la histèresi i gestionar les càrregues per separat, però el fet d’engegar o apagar una càrrega altera el balanç net horari final, cosa que complicaria molt el sistema.

La solució adequada és aplicar el comportament a la 2a càrrega només si la 1a s’ha engegat. I apagar la 1a càrrega només si la 2a està apagada. La màquina d’estat queda de la següent forma:

1 Engegada

2 Apagada

engegar càrrega

apagar càrrega

1 i 2 Engegades

1 i 2 Apagades

engegar càrrega

apagar càrrega

Figura 4.19. Predictiu – Màquina d’estats per controlar dues càrregues

# Simulador

Abans d’implementar els algoritmes al món real, s’ha creat un simulador per poder veure el seu comportament i poder trobar la configuració més òptima en funció de la instal·lació.

Fer el simulador ha ocupat gran percentatge del projecte, degut a diversos factors:

* S’han fet servir llibreries mai utilitzades anterioritat.
* S’ha fet genèric, per poder afegir i emular qualsevol algoritme en un sol programa.
* La quantitat de dades i paràmetres que havia d’incloure.
* Els càlculs s’han optimitzat al màxim, ja que, al tractar amb tantes dades, la simulació podria arribar a tardar molt, depenent de la implementació.
* S’ha fet el màxim d’accessible al usuari, implementat una interfície gràfica i fent el programa el màxim de robust.
* S’ha desenvolupat de forma iterativa, implementant les noves funcionalitats i algoritmes a mesura que han anat fent falta, ja que algunes d’elles no estaven marcades a l’inici.

## Instal·lació i ús

Per fer servir el simulador només cal instal·lar les llibreries fetes servir. Dins de la carpeta *./sim* hem d’executar pip install -r requirements.txt. Un cop instal·lades ja podem iniciar el programa executant *main.py*.

El simulador té una pestanya d’ajuda amb tota la informació per aprendre a fer-lo servir. També és pot llegir sense obrir el simulador obrint el fitxer *Simulator Manual.html*.

Finalment, dir que dins de la carpeta *./sim/db* hi ha totes les dades del monitoratge del sistema real implementat. Per si es vol provar el simulador sense tenir dades pròpies.

## Llenguatge i llibreries

El simulador s’ha implementat amb l’IDE VSCode amb python 3.10, fent ús de múltiples llibreries. Les més rellevants són: Pandas, NumPy, MatPlotLib i PySide6.

Pandas

Pandas és una llibreria que proporciona estructures de dades d’alt rendiment i eines d’anàlisi de dades fàcils d’utilitzar [4]. Pandas està creada sobre la llibreria NumPy, el qual fa que sigui una llibreria molt potent i eficient. L’objecte fonamental de Pandas és el *dataframe*, és un vector bidimensional, és a dir, una taula.

A més a més, ofereix la funció .plot() que per defecte fer servir MatPlotLib com a *backend* gràfic, això vol dir que tenim tota la funcionalitat de MatPlotLib amb Pandas.

Pandas s’ha fet servir per el processar i calcular de totes les dades de la simulació.

NumPy

NumPy és una llibreria que permet crear i operar amb vectors i matrius multidimensionals de forma eficient [5]. Python no té renom de ser un llenguatge eficient, però gràcies a que el nucli de la llibreria està escrit i optimitzat en C, la llibreria és molt eficient. És tant potent que actualment que és la llibreria de Python a utilitzar per defecte al tractar amb dades.

MatPlotLib

MatPlotLib és una llibreria que permet crear gràfiques estàtiques, animades, o interactives [6]. A més a més, permet treballa amb els vectors i matrius de NumPy.

MatPlotLib s’ha fet servir per crear les gràfiques que mostra el simulador.

PySide6

PySide6 és la llibreria oficial del *framework* multi-plataforma Qt 6.0+ [7]. Serveix per crear programes amb interfícies gràfiques *(GUI, Graphical User Interface)*.

Les interfícies es poden programar manualment o dissenyar gràficament amb el programa Qt Designer, el qual ve inclòs amb mòdul. En el cas de fer servir Qt Designer, un cop dissenyades només cal carregar el fitxer *.ui* al script de python i programar les funcionalitats.

PySide6 s’ha fet servir per crear la GUI del programa. Tot i així, apart de tots els components que la llibreria ofereix, s’han creat dos widgets*[[7]](#footnote-8)* propis: un per poder integrar les gràfiques de MatPlotLib la GUI [8] [9], i l’altre per representar les dades de Pandas com a taula [10] [11].

## Implementació del simulador (programa)

Al inici, el simulador era un sol fitxer, el qual anava bé per començar a provar els conceptes i llibreries. Ara, degut a la mida que ha anat prenent el programa, s’ha segmentat el codi en diferents mòduls bàsics. L’estructura final és la següent:

main.py

AlgorithmsConfig.py

constants.py

Dataframe.py

Load.py

Optimize.py

PlotController.py

QMplWidgets.py

QPandasWidgets.py

Results.py

Simulator.py

programa

mòduls per GUI extres

mòduls de simulació

Figura 5.1. Simulador – Dependència entre mòduls

De tots els mòduls, només explicarem parcialment com s’ha implementat el *simulator.py*, ja que és on fa la simulació. Si expliquéssim la resta de mòduls ens hi podríem estar molta estona. Per veure com s’han implementat, es pot llegir el codi de cada un. El programa es troba dins la carpeta *./sim* i els mòduls dins la carpeta *./sim/lib.*

El que si que explicarem és què ofereix cada mòdul per contextualitzar la implementació del programa.

main.py

És el programa, conté la lògica de la GUI i fa les crides necessàries a la resta de mòduls.

AlgorithmsConfig.py

La classe AlgorithmConfig representa qualsevol configuració d’un algoritme. Les tres subclasses representen les configuracions de cada algoritme implementat: HysteresisConfig, MinOnTimeConfig, TimeToConsume.

En el cas que un paràmetre de la configuració sigui un rang d’opcions, i no un numero, s’ha representa amb una enumeració, és a dir, un objecte Enum.

constants.py

Conté les constants que es fan servir en múltiples mòduls.

Dataframe.py

La classe DataFrameIn representa un dataframede Pandas que es pot fer servir per simular, ja que té les columnes de dades necessàries. Aquest comprova que el *dataframe,* que se li passa al inicialitzar la classe, tingui les columnes necessàries. També ofereix alguns mètodes útils.

La subclasse DataFrameOut representa un dataframede Pandas que es pot fer servir per calcular resultats. És subclasse de DataFrameIn i també es fer servir per simular. És l’objecte que la simulació retorna.

Load.py

Emulen a una càrrega controlada pel sistema. S’inicialitzen indicant la seva potència de consum. Es poden apagar i engegar amb el mètode .set\_status(status:bool). El mètode .get\_power() retorna la potència que consumeix la càrrega, en funció de si està engegada o no.

Gràcies a aquesta classe el simulador no ha de gestionar l’estat i consum de les càrregues.

Optimize.py

La classe Optimize serveix per crear un dataframe on s’hi guarden alguns resultats de cada simulació. S’afegeix amb el mètode .add\_result(index:int, results:Results).

El fa servir l’apartat “d’optimitzar” paràmetres. Aquest simula múltiples cop l’algoritme canviant el valor d’un paràmetre de la configuració dins del rang indicat.

PlotController.py

La classe BarPlotController serveix per dibuixar una gràfica de barres en un widget QMplPlot a partir d’un dataframe Pandas i controlar-la. S’ha creat perquè hi ha múltiples gràfiques d’aquest estil. Així només cal crear els controladors necessaris, enllaçar els widgets amb els controladors i enllaçar els controls de la GUI amb els Controladors.

La subclasse EBPlotController serveix per dibuixar i controlar la gràfica de barres d’energies, ja les dades i con és representen són diferents a la classe BarPlotController.

QMplWidgets.py

La classe QMplPlot és el widget permet integrar les gràfiques de MatPlotLib a la GUI. A més a més, apart de les funcionalitats ja oferires per MatPlotLib, se n’ha afegit d’extres:

* Les línies és poden amagar clicant la línia corresponent de la llegenda.
* Si algun eix és del tipus hores, les coordenades del cursor que apareixen a dalt a la dreta es mostren formatjades correctament.
* Es pot fer zoom amb la rodeta. També és pot fer zoom en un sol eix, només cal clicar la tecla x o y menters es mou la rodeta.
* S’ha afegit una nova opció a la barra d’eines. Permet activar un punt de mira que es posa sobre la mostra més propera al cursor i de la línia seleccionada. Per canviar de línia seleccionada simplement s’ha de clicar sobre la nova línia.

La subclasse QMplTwinxPlot és el widget que permet tenir gràfiques amb dos eixos y. Apart de les funcionalitats anteriors, és podent alinear els dos eixos y en el 0.

La subclasse QMplPlotterWidget és el widget que permet tenir una gràfica editable. Per editar la gràfica, només cal seleccionar un dels punts i moure’l en l’eix y. Externament, al canviar algun valor amb la funció set\_value(), la gràfica s’actualitza sola.

La classe QMplMovableHVLine és un controlador que permet moure la línia vertical o horitzontal en l’eix x o y, respectivament.

QPandasWidgets.py

La classe QPandasModel és el widget model que enllaça el widget QTalbeView amb el dataframede Pandas. Permet triar el numero de decimals a mostrar, i si algun valor és del tipus data, la mostra adequadament.

La subclasse QPandasModelEdit afegeix la funcionalitat de poder editar les cel·les.

La classe WrapHeader és una re-implementació de la classe QHeaderView per tal que el text de les capçaleres tinguin salt de línia automàtic *(word wrap)*.

Results.py

La classe Results calcula els resultats de la simulació a partir d’un DataFrameOut i els emmagatzema en tres dataframesde Pandas, aquest són:

* .df\_hour, conté les energies, el balanç monetari, les eficiències, les commutacions de cada hora simulada.
* .df\_total, conte les mateixes dades que df\_hour però a nivell de simulació i a nivell diària.
* .df\_results, conte les mateixes dades que df\_total que no corresponen a energies, juntament amb les càrregues aproximades i el numero de mostres, hores i hores diàries que la càrrega ha estat engegada.

Bàsicament, si mirem el codi, veurem que la classe Results fa quasi tots els càlculs definits a l’apartat 3.2 i 3.3.

Simulator.py

Conté la funció simulate() que serveix per simular el sistema a partir de les dades d’entrada, la seva configuració de algoritme, la càrrega 1, la càrrega 2, i la càrrega base. Aquests paràmetres és corresponen a les classes DataFrameIn, AlgorithmConfig, Load, Load i float.

A més a més, si la càrrega base val None i les dades d’entrada són del tipus DataFrameOut, és fan servir les dades reals de al càrrega base, en comptes d’un valor constant.

## Implementació de la simulació (simulator.py)

Estructura

L’emulació del sistema i algoritme és pot resumir en els següent passos:

1. Obtenir els paràmetres i dades del sistema → *Són els arguments de la funció. El programa les obté de l’usuari a través de la GUI.*
2. Inicialitzar les variables que puguem necessitar.
3. Iterar sobre cada fila del dataframe, per simular cada instant de temps → *Emula el bucle infinit que faria el controlador real, sense tenir temps d’espera entre mostres*.
   1. Obtenir o calcular les potències i balanç.
   2. Aplicar l’algoritme, és a dir, engegar o apagar les càrregues.
   3. Guardar les dades simulades rellevants.
4. Guardar els resultats de la simulació en un nou dataframe.
5. Calcular la resta de dades de simulació que es poden calcular a partir de les columnes. Això és fa així perquè és més eficient que fer el càlcul en cada iteració.

Si mirem el codi, podem veure que els codi està agrupat per blocs[[8]](#footnote-9), cada un corresponent a un pas.

def simulate(df\_in:DataFrameIn, algorithm:AlgorithmConfig, load1: Load, load2:Load, baseload:float|int|None=0): # (1)

#region -> Init (2)

for row in df\_in.df.to\_dict('records'):

#region -> Calc/Get Powers/Energies (3.1)

#region -> Algorithm (3.2)

#region -> [Simulation] Calc/Save other energys for data avaluation (3.3)

#region -> [Simulation] Store data to pandas dataframe (4)

#region -> [Simulation] Calc extra sim data (5)

Potències i energies

Per poder prendre decisions, l’algoritme necessita el balanç energètic en aquell instant de temps (mostra). El càlcul discret el tenim en la formula 3.19 de l’apartat de definicions 3.2.

Primer obtenim les potències que hi ha en aquell moment.

* La potència de producció l’obtenim de dataframe de dades reals que l’usuari ha proporcionat.
* Si a la funció se li ha indicat un valor de càrrega base, la potència de consum base serà el valor. Si en comptes d’un valor s’ha passat None i el dataframe contenen la potència base, obtindrem el valor a través de les dades reals.
* Per calcular la potència consumida, sumem les potències de consumides de cada càrrega. La potència de cada càrrega controlada l’obtenim dels objectes Load.
* Finalment calculem la potència disponible que ens queda.

El següent pas és determinar si hi ha hagut canvi d’hora. Per fer-ho, obtenim l’instant de temps del dataframe i comparem si l’hora anterior és diferent.

Si hi ha hagut canvi d’hora haurem de posar a 0 el balanç i guardar la nova hora. Com que també volem obtenir l’energia produïda per analitzar la simulació, també l’haurem de posar a 0.

Finalment, calculem el balanç en aquell instant de temps[[9]](#footnote-10).

for row in df\_in.df.to\_dict('records'):

#region -> Calc/Get Powers/Energies

power\_lb = row['powerLB'] if use\_df\_bl else baseload # [Simulation]

timestamp = row['timestamp']

power\_p = row['powerP']

power\_c = power\_lb + load1.get\_power() + load2.get\_power()

power\_a = power\_p - power\_c

#region -> If hour has passed reset powers

if current\_hour != timestamp.hour:

energy\_b = energy\_p =  0 # [Simulation] energy\_p

current\_hour = timestamp.hour

next\_hour = timestamp.replace(second=0, microsecond=0, minute=0) + dt.timedelta(hours=1)

#endregion

# Calc Available Energy [Wh]

energy\_b = (power\_a) \* df\_in.Ts / 3600 + energy\_b

#endregion

...

Algoritmes

Un cop tenim l’instant de temps i el balanç actual calculat, apliquem l’algoritme per controlar les dues càrregues.

#region -> Algorithm

match algorithm.type:

case AlgorithmType.hysteresis:

...

case AlgorithmType.min\_on\_time:

...

case AlgorithmType.time\_to\_consume:

...

#endregion

Pel control a partir del temps mínim engegada s’ha hagut d’implementar l’apagada automàtica de la càrrega, ja que l’objecte Load no té aquesta funcionalitat. Per fer-ho, cada cop que s’engega una càrrega és calcula i guarda l’instant de temps en que s’apagarà.

Al final d’aplicar l’algoritme, si l’instant de temps que s’està simulant és igual o major al guardat, la càrrega s’apaga.

case AlgorithmType.min\_on\_time:

if two\_load\_system:

...

else:

time\_to\_use = energy\_b/load1.value \* 3600

if time\_to\_use >= algorithm.time\_limit:

off\_timestamps[0] = timestamp + dt.timedelta(seconds=algorithm.time\_limit)

on\_offL1 = load1.set\_status(True)

# Simulaicó del timer de la càrrega

if timestamp >= off\_timestamps[0]:

on\_offL1 = load1.set\_status(False)

if timestamp >= off\_timestamps[1]:

on\_offL2 = load2.set\_status(False)

Optimitzacions

És molt important implementar el simulador de forma que sigui el màxim de ràpid. Un punt del codi que pot afectar considerablement a nivell d’eficiència és com iterem sobre les files del dataframe.

Si busquem per internet com fer-ho, podem pensar que la funció *builtin* de Pandas .iterrows() és suficient. Però no ho és, ja que encapsula les dades en una objecte Series de Pandas [12], afegint temps de càlcul que augmenta amb el numero de mostres del dataframe [13].

Si l’únic que volem és iterar, hem de fer servir la funció .to\_dict(‘records’). Gràcies a aquest canvi s’ha reduït el temps d’execució per un factor d’entre 5 i 10.

Un altre canvi que també ha reduït el temps d’execució per un factor de 2 ha estat passar el dataframe amb només les columnes necessàries (timestamp, powerC i powerLB), ja que sinó, al transformar-ho en diccionari, estaríem processant dades que no faríem servir.

# Implementació real

L’objectiu principal del projecte ha estat portar al món real els sistemes de control dissenyats. El codirector del TFG Jordi Bonet ha ofert la seva instal·lació solar d’autoconsum de casa i ha aportat tots els recursos necessaris per muntar-hi el sistema.

## Descripció de la instal·lació

Font d’energia renovable

La producció d’energia renovable prové d’un conjuntament plaques solars que, en les millors condicions, produeix entre 2 i 2,5 kW de potència.

Càrregues i control remot

Les càrregues que es volen controlar són dues bombes de calor, una de 700 W i l’altre 1 kW. Per poder engegar, apagar i monitoritzar la potència de les càrregues remotament s’han instal·lat dos interruptors remotes Shelly, una per cada alimentació.

L’interruptor Shelly és un relé intel·ligent el qual es pot controlar i monitoritzar per WiFi. El model utilitzat és el Shelly1L, Figura 6.1, que ofereix diferents formes d’interacció [14]:

* Amb un botó físic.
* Atacant a l’API rest, fent peticions HTTP a través de la xarxa local.
* Enviant la instrucció a través del núvol o MQTT.
* Amb un horari setmanal configurat.
* A partir d’un esdeveniment generat per la sortida/la posta de sol.
* Una sobrecàrrega de potència apagarà el relé.

Mesurador de producció i consum

El sistema té instal·lat el Enphase Envoy-S Metered, Figura 6.2. És un mesurador de producció i consum que, apart d’altres coses, permet obtenir la informació a través de peticions HTTP.

Control - Servidor

Per poder controlar tot el sistema, s’ha fet servir una Raspberry Pi[[10]](#footnote-11), Figura 6.3, connectada a la xarxa local, que és el canal que tots els dispositius fan servir per comunicar-se.

Per poder programar la Raspberry remotament s’ha fet servir la VPN (Virtual Private Network) OpenVPN.

Base de dades i visualització

Per poder veure el comportament del sistema a temps real i comprovar que tot funciona, totes les dades monitoritzades s’han guardat a una base de dades de InfluxDB [15], amb la qual és pot interaccionar remotament a través d’una API Rest [16].

Per visualitzar les dades guardades a temps real, s’ha fet servir l’aplicació web Grafana [17]. El *dashboard* l’ha creat el Jordi Bonet, Figura 6.4.

|  |  |
| --- | --- |
| Shelly1L  Figura 6.1. Shelly1L | Enphase plug-and-play solar energy storage system to begin pilot program  Figura 6.2. Enphase Envoy-S Metered |
| Raspberry Pi - Viquipèdia, l'enciclopèdia lliure  Figura 6.3. Raspberry Pi Model B | |



Figura 6.4. Dashboard creat en Grafana que mostra les dades del sistema a temps real

## Implementació del software

A l’inici, s’havia implementat la solució amb un sol codi i era específica per la instal·lació que teníem. Però tenia un problema, no era portable entre instal·lacions. Així dons, un s’ha aconseguit que funciones per el cas específic, s’ha refet tot el codi per fer-lo portable i senzill d’utilitzar. Aquest es troba en la carpeta .*/src* i s’ha fet enPython 3.7.

El primer pas ha estat definir l’estàndard dels controladors[[11]](#footnote-12) *(drivers)* de cada tipus perifèric diferent, així l’únic que cal canviar entre instal·lacions són les configuracions dels dispositius i els divers dels dispositius diferents. Els perifèrics són:

* La base de dades.
* Mesurador de producció i consum.
* Els dispositius per controlar la càrregues.

L’altre canvi que s’ha fet ha estat separar el programa en dos: *managing.py* i *monitoring.py*, ja que les dues funcionalitats que implementava, gestió i monitorització, són independents.

Com que el sistema operatiu de la Raspberry és una distribució Linux, s’ha fet servir *systemd* per crear els serveis, coneguts com a *daemons*, per tenir els dos programes funcionant contínuament i de fons. Aquest es troben a la carpeta *./src/daemons*

Drivers

La programació dels drivers s’ha fet orientada a objectes, és a dir, s’han creat una classe per cada driver. D’aquesta manera podem controlar dos perifèrics igual amb dues instàncies de la mateixa classe.

Per tal d’estandarditzar els drivers, s’han definit uns drivers base sense cap funcionalitat. Simplement són una classe abstracta que defineix els mètodes del driver.

Si és vol veure l’estàndard definit per a cada driver o com s’han implementat els drivers per els dispositius d’aquesta instal·lació, els drivers es troben en la carpeta *./src/drivers*, on cada subcarpeta correspon a un perifèric. Cada subcarpeta conté el driver base i les seves implementacions específiques.

managin.py

PowerMeter.py

DataBaseBase.py

LoadBase.py

InfluxDB.py

EnvoyS.py

Shelly.py

Programes

managin.py

Drivers específics

Drivers base

Figura 6.5. Relació programa-driver

Per implementar el driver d’InfluxDB, simplement s’han embolcallat certes funcionalitats de la llibreria oficial d’InfluxDB [18]. La implementació el driver Shelly, també ha consistit en embolcallat certes funcionalitats de la llibreria ShellyPy [19]. En canvi, per implementar el driver EnvoyS no s’ha trobat cap llibreria, per tant s’ha fet les peticions HTTP amb el mòdul requests.

Monitorització (monitoring.py)

El passos per monitoritzar són:

1. Inicialitzar els drivers*.*
2. Inicialitzar les variables necessàries.
3. De forma indefinida:
   1. Obtenir o calcular les potències i balanç.
   2. Obtenir la informació de les càrregues.
   3. Guardar les dades a la base de dades.
   4. Esperar . On és el període de mostreig.

La raó perquè el temps d’espera és , i no , és perquè executar el codi tarda un segons a executar-se. Si no ho factoritzéssim a l’espera, estaríem mostrejant cada .

Apart dels passos anteriors, s’han implementat certes funcionalitats extres:

* En la inicialització de variables, s’ha fet que el programa recuperi els valors de l’ultima mostra si aquesta està dins de l’hora en que es troba, si no els valors comencen des de 0. Això permet reiniciar el monitoratge en qualsevol moment i no perdre l’estat anterior el balanç i energia produïda fins el moment i l’estat de les càrregues).

Gestió (managing.py)

El passos per controlar les càrregues són:

1. Inicialitzar els drivers*.*
2. Inicialitzar les variables necessàries i posar les càrregues a 0.
3. De forma indefinida:
   1. Obtenir o calcular les potències i balanç.
   2. Aplicar l’algoritme, és a dir, engegar o apagar les càrregues, només si estem en hores de treball.
   3. Esperar . On és el període de mostreig.

També hem d’esperar per la mateixa raó que en la monitorització.

Apart dels passos anteriors, s’han implementat certes funcionalitats extres:

* En la inicialització de variables, s’ha fet que el programa recuperi el balanç de l’ultima mostra de la base de dades de monitoratge dins de l’hora en que es troba. Si no hi ha valors, comença a 0. Això permet reiniciar el gestor sense perdre el balanç que s’havia calculat fins el moment.
* El gestor només controla les càrregues l’hora d’inici i l’hora final indicada. Així, durant la nit, per exemple, podem engegar les càrregues si fes falta (ja que el gestor les apagaria al instant).
* Quan s’engega una càrrega, se l’indica que ho faci durant 3630 segons, per evitar que si els control peta o queda *offline*, la càrrega s’apagui i no consumeixi indefinidament. Ara bé, els algoritmes que no indiquen el temps que ha d’estar engegada esperarien que mai s’apagués. Fent que s’apagui al cap de 3630 segons permet refrescar l’estat durant una hora més al fer canvi d’hora.

## Instal·lació i ús

A l’hora d’instal·lar aquesta implementació en qualsevol instal·lació elèctrica similar s’han de seguir els següents passos:

1. Copiar la capeta *./src* al servidor.
2. Dins de la carpeta *./src* executar `pip install -r requirements.txt`.
3. Copiar els *daemons* a la carpeta */etc/systemd/system.*
4. Crear la base de dades.
5. Crear drivers necessaris.
6. Configurar els dos programes. La configuració es troba en la secció de *Settings.*
7. Indicar els *daemons.*

# Anàlisi

A partir de les dades reals del sistema a implementat obtingudes a través del monitoratge, les qual es poden trobar a la carpeta *./sim/db*, s’ha pogut simular els diversos algoritmes en les mateixes condicions.

## Precisió del simulador

## Histèresi vs Temps mínim engegada

Histèresi és millor 100%

# Conclusions

La implementació real s’ha fet durant la primavera i estiu 🡪 Porvar-ho al hivern

# Bibliografia

|  |  |
| --- | --- |
| [1] | A. E. B. O. d. Estado, «Real Decreto 244/2019,» Agencia Estatal Boletín Oficial del Estado, 6 4 2019. [En línia]. Available: https://www.boe.es/diario\_boe/txt.php?id=BOE-A-2019-5089. [Últim accés: 23 06 2022]. |
| [2] | S. Energia, «Autoproducció: què és el balanç net horari?,» Som Energia, [En línia]. Available: https://ca.support.somenergia.coop/article/849-autoproduccio-que-es-el-balanc-net-horari. [Últim accés: 23 6 2022]. |
| [3] | «Hysteresis,» Wikipedia, 16 04 2022. [En línia]. Available: https://en.wikipedia.org/wiki/Hysteresis. [Últim accés: 28 06 2022]. |
| [4] | p. d. team, «Pandas Documentation,» pandas development team, 23 06 2022. [En línia]. Available: https://pandas.pydata.org/docs/reference/index.html. [Últim accés: 30 06 2022]. |
| [5] | N. Developer, «NumPy documentation,» NumPy Developer, [En línia]. Available: https://numpy.org/doc/stable/. [Últim accés: 30 06 2022]. |
| [6] | T. M. d. team, «Matplotlib documentation,» The Matplotlib development team, [En línia]. Available: https://matplotlib.org/3.5.2/index.html. [Últim accés: 30 06 2022]. |
| [7] | «Qt for Python,» 2021. [En línia]. Available: https://doc.qt.io/qtforpython/. [Últim accés: 19 12 2021]. |
| [8] | M. Fitzpatrick, «Matplotlib plots in PySide, embedding charts in your GUI applications,» Python GUIS, 30 05 2022. [En línia]. Available: https://www.pythonguis.com/tutorials/pyside-plotting-matplotlib/. [Últim accés: 30 06 2022]. |
| [9] | Q. f. Python, «Matplotlib Widget Gaussian Example,» Qt for Python, [En línia]. Available: https://doc.qt.io/qtforpython/examples/example\_external\_matplotlib\_widget\_gaussian.html?highlight=matplotlib. [Últim accés: 30 06 2022]. |
| [10] | M. Fitzpatrick, «Display numpy and pandas tables in PySide2 QTableView,» Python GUIs, 2 9 2021. [En línia]. Available: https://www.pythonguis.com/tutorials/pyside-qtableview-modelviews-numpy-pandas/. [Últim accés: 30 06 2022]. |
| [11] | Q. f. Python, «Pandas Simple Example,» Qt for Python, [En línia]. Available: https://doc.qt.io/qtforpython/examples/example\_external\_\_pandas.html. [Últim accés: 30 06 2022]. |
| [12] | Jeff, «Does pandas iterrows have performance issues?,» Stack Overflow, 09 04 2019. [En línia]. Available: https://stackoverflow.com/questions/24870953/does-pandas-iterrows-have-performance-issues. [Últim accés: 01 07 2022]. |
| [13] | S. Kumar, «Here’s the most efficient way to iterate through your Pandas Dataframe,» towardsdatascience, 19 06 2021. [En línia]. Available: https://towardsdatascience.com/heres-the-most-efficient-way-to-iterate-through-your-pandas-dataframe-4dad88ac92ee. [Últim accés: 1 07 2022]. |
| [14] | «Shelly1L - API Reference,» Shelly, [En línia]. Available: https://shelly-api-docs.shelly.cloud/gen1/#shelly1l. [Últim accés: 03 07 2022]. |
| [15] | «InfluxDB: Open Source Time Series Database,» Influxdata, [En línia]. Available: https://www.influxdata.com/. [Últim accés: 03 07 2022]. |
| [16] | «InfluxDB v2 API,» InfluxDB, [En línia]. Available: https://docs.influxdata.com/influxdb/v2.3/reference/api/. [Últim accés: 03 07 2022]. |
| [17] | «Grafana,» Grafana Labs, [En línia]. Available: https://grafana.com/. [Últim accés: 03 07 2022]. |
| [18] | S. Energia, «Les diferents modalitats d'autoproducció renovable domèstica,» Som Energia, [En línia]. Available: https://ca.support.somenergia.coop/article/803-les-diferents-modalitats-dautoproduccio-renovable-domestica. [Últim accés: 23 6 2022]. |
| [19] | S. Energia, «Com funciona la compensació simplificada d'excedents?,» Som Energia, [En línia]. Available: https://ca.support.somenergia.coop/article/783-com-funciona-la-compensacio-simplificada-dexcedents. [Últim accés: 23 6 2022]. |

1. Torbareu la informació a la pàgina 9 del Reial Decret 244/2019 [1]. [↑](#footnote-ref-2)
2. Per una explicació més completa podeu anar al centre d’ajuda de Som Energia [15] [↑](#footnote-ref-3)
3. Torbareu la informació a la pàgina 17 del Reial Decret 244/2019 [1]. [↑](#footnote-ref-4)
4. Més informació al centre d’ajuda de Som Energia [16] [↑](#footnote-ref-5)
5. Les línies de la llegenda semitransparents signifiquen que estan amagades. [↑](#footnote-ref-6)
6. Això no treu que simplement mirant energies i potències puguem comparar algoritmes. [↑](#footnote-ref-7)
7. Un *widget* és un bloc de construcció bàsic d’una aplicació en termes de Qt. [↑](#footnote-ref-8)
8. VSCode permet delimitar blocs de codi python, aquests és poden plegar o desplegar per ajudar a simplificar la lectura del codi. És només una definició a nivell d’eina, no modifica el llenguatge en cap aspecte. Una bloc de codi és aquell que està entre els comentaris #region i #endregion. [↑](#footnote-ref-9)
9. Recordar que el balanç el calculem de la segona forma definida a l’apartat 3.2I.3.2. Per això primer posem a 0 el balanç en el canvi d’hora i després calculem el balanç. Si ho féssim després de calcular el balanç estaríem fent la primera opció. [↑](#footnote-ref-10)
10. Raspberry Pi 3 Model B Plus Rev 1.3 [↑](#footnote-ref-11)
11. Un driver és un mòdul que permet al programa interaccionar amb el perifèric, fent una abstracció d’aquest i proporcionant una interfície (normalment estandarditzada) per fer-lo servir. Actua com a traductor entre maquinari i aplicació (o driver no estandarditzat i aplicació). [↑](#footnote-ref-12)