
Gestor energètic

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

8 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 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 consumida de la xarxa es compensa amb l'energia produïda injectada, fent que la reducció del preu de factura de la llum sigui més significativa en comparació a no tenir balanç net horari. També permet a l'usuari tenir cert joc per consumir de la xarxa, i retornar-ho en un altre moment.

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 consumed from the grid can be offset by the energy produced injected, making the reduction of the energy bill price more substantial, compared to a not having “*Balanç net horari*”. This also allows the user to have some wiggle room to consume from the grid and return the consumption later.

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
Abstract	v
I. Memòria	1
 1. Introducció	3
1.1. Punt de partida	3
1.2. Objectiu/s	4
1.3. Codi font	5
 2. Model del sistema	7
2.1. Model general i model ideal	7
2.2. Model simplificat	7
 3. Terminologia.....	9
3.1. Potències	9
3.1.1 Definicions	9
3.1.2 Traduccions	10
3.1.3 Segmentació visual	10
3.1.4 Exemples.....	11
3.2. Energies	14
3.2.1 Definicions	14
3.2.2 Traduccions	16
3.2.3 Segmentació visual	17
3.2.4 Exemples.....	17
3.3. Per comparació	21
3.3.1 Definicions	21
3.4. Traduccions	22
 4. Algoritmes de control	23
4.1. Histèresi	23
4.1.1 Una càrrega	23
4.1.2 Dues càrregues	24
4.1.3 Observacions	24
4.2. Temps mínim engegada	27
4.2.1 Una càrrega	27
4.2.2 Dues càrregues	27
4.2.3 Observacions	28
4.3. Predictiu (Temps de consum)	32
4.3.1 Com predir el balanç final.....	32
4.3.2 Una càrrega	33

4.3.3 Dues càrregues	35
5. Simulador	37
5.1. Llenguatge i llibreries	37
5.2. Implementació del simulador (programa)	38
5.3. Implementació de la simulació (simulator.py)	41
5.4. Instal·lació i ús	43
6. Implementació real.....	45
6.1. Descripció de la instal·lació	45
6.2. Implementació del software.....	47
6.3. Instal·lació i ús	49
7. Anàlisi (sistema real)	51
7.1. Precisió del simulador	51
7.2. Sistema real	51
7.2.1 Dia assolellat.....	52
7.2.2 Dia de núvols/pluja	63
7.2.3 Conclusions	65
8. Conclusions.....	67
Bibliografia	69
II. Annex	71
A. Realitat vs Simulació	73
A.1. Histèresis	73
A.2. Tems mínim engegada.....	75
A.3. Predictiu	76
B. Dia assolellat.....	79
B.1. Predictiu – càrrega 1 > càrrega 2	79

Figures

Figura 1.1. Càlcul de la factura de la llum	3
Figura 2.1. Model energètic (vàlid tant per potència o energia)	8
Figura 3.1. Segmentació visual de la potència del sistema	10
Figura 3.2. Exemple – Potència produïda d'un dia assolellat	11
Figura 3.3. Exemple - Potència produïda d'un dia ennuvolat	11
Figura 3.4. Exemple - Potència produïda d'un dia plujós	12
Figura 3.5. Exemple - Potència consumida i de cada càrrega	12
Figura 3.6. Exemple – Potència disponible i de la xarxa – Vista completa	13
Figura 3.7. Exemple - Potència del consum màxim (ideal)	13
Figura 3.8. Formes de calcular el balanç	15
Figura 3.9. Segmentació visual de l'energia del sistema	17
Figura 3.10. Exemple – Consum màxim d'energia a partir d'àrees	17
Figura 3.11. Exemple - Energia perduda a partir d'àrees	18
Figura 3.12. Exemple – Energia disponible del balanç i deguda a la xarxa – Vista comportament	18
Figura 3.13. Exemple – Resultats per hores de les energies (vista model)	19
Figura 3.14. Exemple – Resultats per hores de les energies (1a segmentació)	19
Figura 3.15. Exemple – Resultats per hores de les energies (2a segmentació)	20
Figura 3.16. Exemple – Model energètic per hores (2a segmentació)	20
Figura 4.1. Corba d'hystèresi del control d'una càrrega	23
Figura 4.2. Histèresi – Màquina d'estats per controlar una càrrega	24
Figura 4.3. Histèresi – 1a càrrega amb preferència d'engegada respecta la 2a	24
Figura 4.4. Histèresi – 2 càrregues amb els mateixos llindars equivalen a 1	25
Figura 4.5. Histèresis – Llindar baix a 0 i càlcul de balanç projectant al passat	26
Figura 4.6. Histèresis – Llindar baix a 0 i càlcul de balanç projectant al futur	26
Figura 4.7. Temps mín. engegada – Solapament de comportament de les implementacions	28
Figura 4.8. Temps mín. engegada – Implementacions amb el mateix comportament quan $C1 < C2$	29
Figura 4.9. Temps mín. engegada – Implementacions amb el mateix comportament quan $C1 > C2$	30
Figura 4.10. Histèresi (segona part) amb mateixos llindars que Temps mínim engegat (primera part)	31
Figura 4.11. Predictiu – Predicció projectant la potència disponible	32
Figura 4.12. Predictiu – Exemplificació de l'àrea de $pDAvg1h$ equivalent a la integral $pD1h$	33
Figura 4.13. Predictiu – Predicció a partir de la potència mitjana disponible	33
Figura 4.14. Predictiu base – Situació ideal	33
Figura 4.15. Predictiu base – Situacions extremes no desitjades	34
Figura 4.16. Predictiu base – Solució al problema a) i nou problema que presenta	34
Figura 4.17. Predictiu base – Solució al problema b)	34
Figura 4.18. Predictiu – Màquina d'estats per controlar una càrrega	35
Figura 4.19. Predictiu – Màquina d'estats per controlar dues càrregues	35
Figura 5.1. Simulador – Dependència entre mòduls	38
Figura 6.1. Shelly1	46
Figura 6.2. Enphase Envoy-S Metered	46
Figura 6.3. Raspberry Pi Model B	46
Figura 6.4. Dashboard creat en Grafana que mostra les dades del sistema a temps real	46
Figura 6.5. Relació programa-driver	47
Figura 7.1. Anàlisi dia assolellat – Dades per fer la simular	52
Figura 7.2 Anàlisi dia assolellat – Forma exponencial de les commutacions a l'optimitzar el llindar alt	52
Figura 7.3. Anàlisi dia assolellat – Histèresis amb una càrrega – Optimització llindar alt	53
Figura 7.4. Anàlisi dia assolellat – Histèresis amb una càrrega – Optimització llindar baix	54

<i>Figura 7.5. Anàlisi dia assolellat – Histèresis amb una càrrega – Optimització llindar alt 2.....</i>	55
<i>Figura 7.6. Anàlisi dia assolellat – Histèresis amb una càrrega – Optimització llindar baix.....</i>	56
<i>Figura 7.7. Anàlisi dia assolellat – Temps mínim engegada – $C1 < C2$, versions Simple i Completa A ..</i>	57
<i>Figura 7.8. Anàlisi dia assolellat – Temps mínim engegada – $C1 < C2$, versió Completa B</i>	58
<i>Figura 7.9. Anàlisi dia assolellat – Temps mínim engegada – $C1 > C2$, versió Simple.....</i>	59
<i>Figura 7.10. Anàlisi dia assolellat – Temps mínim engegada – $C1 > C2$, versions Completa A i B</i>	60
<i>Figura 7.11. Anàlisi dia assolellat – Configuració óptima - Histèresis</i>	62
<i>Figura 7.12. Anàlisi dia de núvols/pluja – Dades per fer la simular.....</i>	63
<i>Figura 7.13. Anàlisi dia assolellat – Configuració óptima - Histèresis</i>	64
<i>Figura A.1. Histèresis – Comportament i resultats del sistema real.....</i>	73
<i>Figura A.2. Histèresis – Comportament i resultats simulats (sense dades reals del consum base).....</i>	74
<i>Figura A.3. Histèresis – Comportament i resultats simulats (amb dades reals del consum base).....</i>	75
<i>Figura A.4. Predictiu – Comportament i resultats del sistema real</i>	76
<i>Figura A.5. Predictiu – Comportament i resultats simulats (sense dades reals del consum base)</i>	77
<i>Figura B.6. Anàlisi dia assolellat – Predictiu– $C1 < C2$, sense predicción, pas 1</i>	79
<i>Figura B.7. Anàlisi dia assolellat – Predictiu – $C1 < C2$, sense predicción, pas 2</i>	80
<i>Figura B.8. Anàlisi dia assolellat – Predictiu – $C1 < C2$, sense predicción, pas 3</i>	81
<i>Figura B.9. Anàlisi dia assolellat – Predictiu – $C1 < C2$, potència mitjana disponible, pas 1</i>	82
<i>Figura B.10. Anàlisi dia assolellat – Predictiu – $C1 < C2$, potència mitjana disponible, pas 2</i>	83
<i>Figura B.11. Anàlisi dia assolellat – Predictiu – $C1 < C2$, potència mitjana disponible, pas 3</i>	84
<i>Figura B.12. Anàlisi dia assolellat – Predictiu – $C1 < C2$, projecció potència disponible, pas 1</i>	85
<i>Figura B.13. Anàlisi dia assolellat – Predictiu – $C1 < C2$, projecció potència disponible, pas 2</i>	86
<i>Figura B.14. Anàlisi dia assolellat – Predictiu – $C1 < C2$, projecció potència disponible, pas 3</i>	87

Taules

<i>Taula 1.1. Exemple – Càlcul del balanç net horari</i>	4
<i>Taula 1.2. Exemple – Càlcul de la factura amb balanç net horari.....</i>	4
<i>Taula 1.3. Exemple – Càlcul de la factura sense balanç net horari</i>	4
<i>Taula 3.1. Traduccions i nomenclatures – Potències</i>	10
<i>Taula 3.2. Traduccions i nomenclatures – Energies.....</i>	16
<i>Taula 3.3. Traduccions i nomenclatures – Comparació.....</i>	22
<i>Taula 4.1. Temps mín. engegada – Implementacions amb el mateix comportament quan $C1 < C2$.....</i>	28
<i>Taula 4.2. Temps mín. engegada – Implementacions amb el mateix comportament quan $C1 > C2$.....</i>	28
<i>Taula 4.3. Temps mín. engegada – Implementacions amb el mateix comportament quan $C1 = C2$.....</i>	28
<i>Taula 5.1. Columnes de DataFrameIn</i>	39
<i>Taula 5.2. Columnes de DataFrameOut</i>	39
<i>Taula 6.1. Monitorització - Dades que es guarden a la base de dades.....</i>	48
<i>Taula 7.1. Anàlisi dia assolellat – Resultats del sistema en funció de l'algoritme</i>	62
<i>Taula 7.2. Anàlisi dia assolellat – Configuració òptima - Temps mínim engegada.....</i>	62
<i>Taula 7.3. Anàlisi dia assolellat – Configuració òptima - Predictiu</i>	62
<i>Taula 7.4. Anàlisi dia de núvols/pluja – Resultats del sistema en funció de l'algoritme</i>	64
<i>Taula 7.5. Anàlisi dia assolellat – Configuració òptima - Temps mínim engegada.....</i>	64
<i>Taula 7.6. Anàlisi dia assolellat – Configuració òptima - Predictiu</i>	64

I. Memòria

1. Introducció

1.1. 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 acollir¹. En resum hi ha dues modalitats en funció de l'energia produïda²:

- 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.
- Amb excedents: L'energia produïda no consumida és injectada a la xarxa de transport i distribució. Aquests excedents poden ser:
 - Sense compensació econòmica
 - 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, 4}:

- 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 arribar a 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).

$$\text{Balanç net horari}_i = \text{energiaX} - \text{energiaI} \quad (1.1)$$

Des del punt de vista d'un sistema equivalent, un resultat positiu significa que només s'ha consumit de la xarxa. Si és negatiu, que només s'ha injectat.

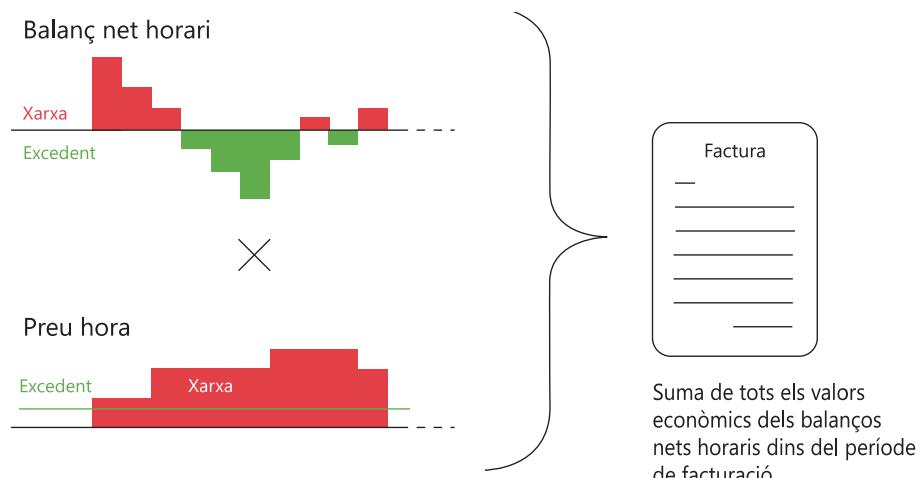


Figura 1.1. Càlcul de la factura de la llum

¹ Torbareu la informació a la pàgina 9 del Reial Decret 244/2019 [1].

² Per una explicació més completa podeu anar al centre d'ajuda de Som Energia [15]

³ Torbareu la informació a la pàgina 17 del Reial Decret 244/2019 [1].

⁴ Més informació al centre d'ajuda de Som Energia [16]

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

El balanç net horari és 0,6 kWh. Això equival a un sistema que només ha injectat 0,6 kWh i no ha consumit res de la xarxa.

$$\text{Balanç net horari}_{13h} = 0,2 - 0,8 = -0,6 \quad (1.2)$$

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:

Hora	Energia [kWh]			Preu [€/kWh]			Factura [€]
	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	
							0,4055

Taula 1.2. Exemple – Càlcul de la factura amb balanç net horari

1.2. 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:

Hora	Energia [kWh]		Preu [€/kWh]			[€]		
	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.
- Minimitzar les commutacions de les càrregues.
- Tenir el menor impacte ambiental.

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.

1.3. Codi font

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

2. 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.

2.1. 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; ...

2.2. 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 produïda 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:

$$Sistema = Produïda + Xarxa = Consumida + Produïda \text{ no consumida} \quad (2.1)$$

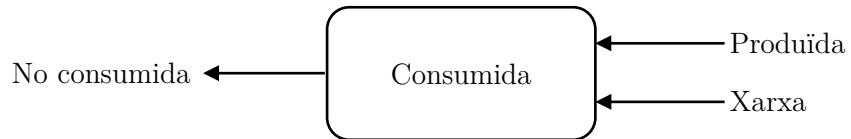


Figura 2.1. Model energètic (vàlid tant per potència o energia)

3. 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 interície 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.

3.1. Potències

3.1.1 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, és la suma dels consums de totes les càrregues.

$$pC(t) = pCB(t) + pC1(t) + pC2(t) = \sum pCi(t) \quad (3.1)$$

- 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).

$$pCM(t) = \begin{cases} pP(t) & \text{si } pP(t) < carregaMax \\ carregaMax & \text{altrament} \end{cases} \quad (3.2)$$

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

$$carregaMax = \sum carrega_i \quad (3.3)$$

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

$$carregaMax = \sum carregaAprox_i \quad (3.4)$$

$$carregaAprox_i = \frac{\sum pCi_i}{\text{count}(pCi_i)} \quad \text{on } pCi_i > 0 \quad (3.5)$$

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 suprir l'energia consumida quan la produïda no cobreix la demanda.

$$pX(t) = \begin{cases} pC(t) - pP(t) & \text{si } pX(t) > 0 \\ 0 & \text{altrament} \end{cases} \quad (3.6)$$

- 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 $pP - pC$.

$$pPnC(t) = \begin{cases} pP(t) - pC(t) & \text{si } pPnC(t) > 0 \\ 0 & \text{altrament} \end{cases} \quad (3.7)$$

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

$$pXP(t) = \begin{cases} pX(t) & \text{si } pP(t) > 0 \\ 0 & \text{altrament} \end{cases} \quad (3.8)$$

3.1.2 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

3.1.3 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.

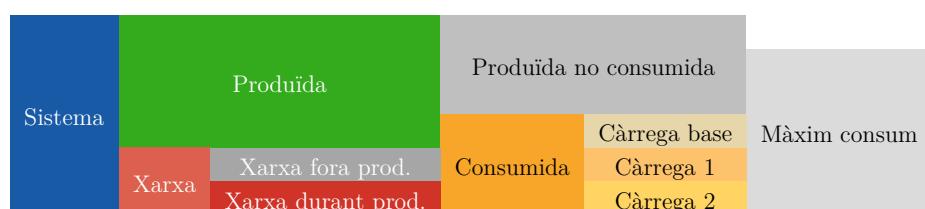


Figura 3.1. Segmentació visual de la potència del sistema

3.1.4 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 simulador⁵ 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.

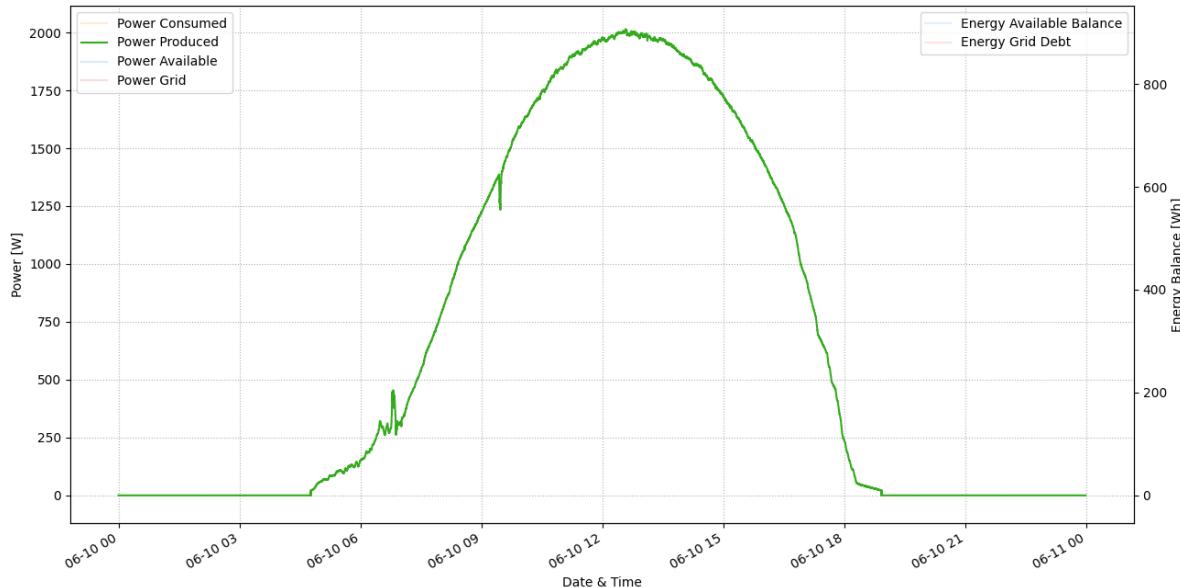


Figura 3.2. Exemple – Potència produïda d'un dia assolellat

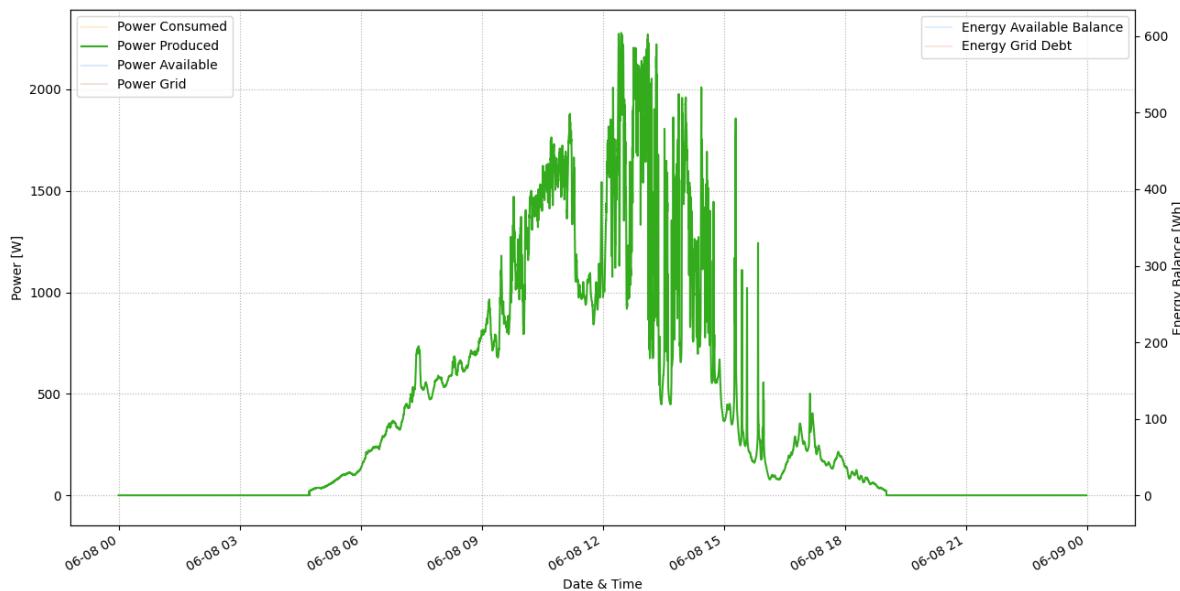


Figura 3.3. Exemple - Potència produïda d'un dia ennuvolat

⁵ Les línies de la llegenda semitransparentes signifiquen que estan amagades.

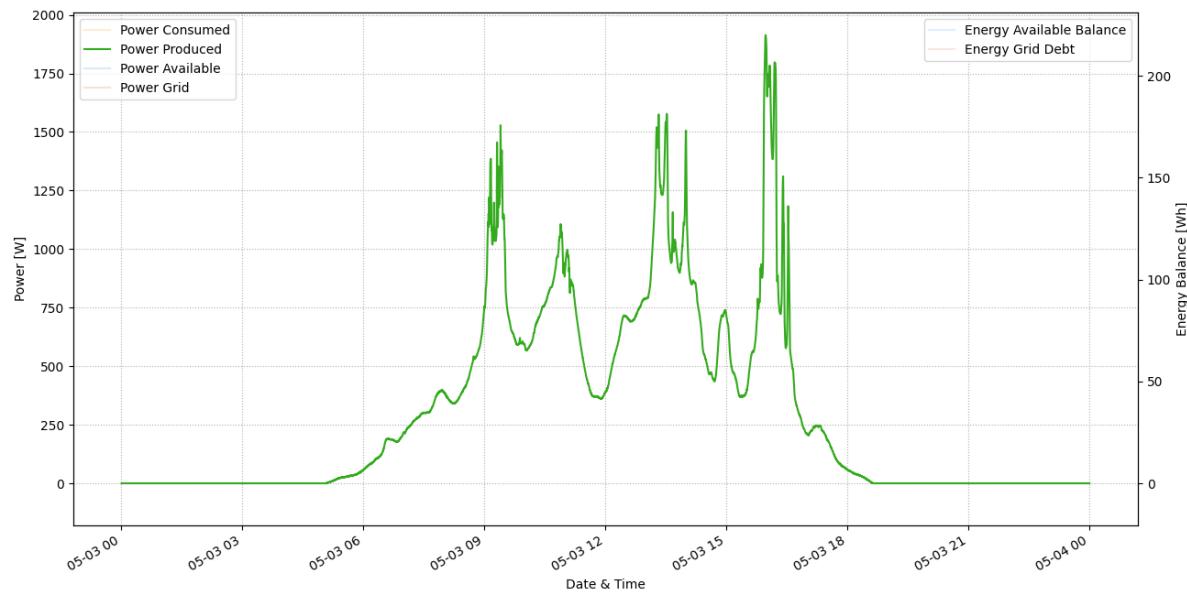


Figura 3.4. Exemple - Potència produïda d'un dia plujós

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).

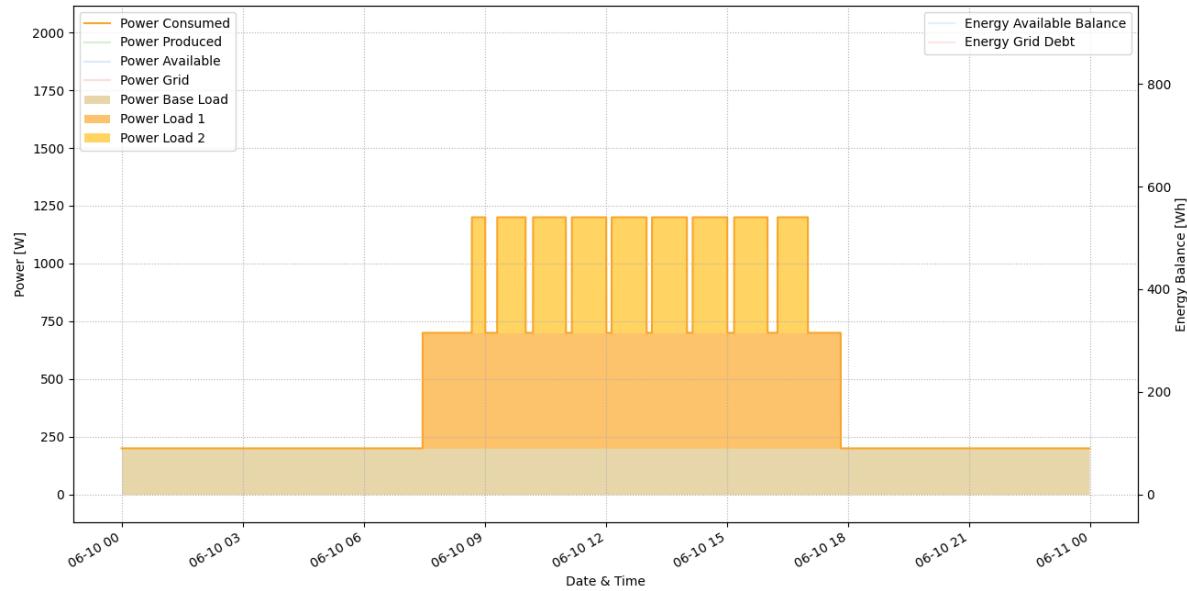


Figura 3.5. Exemple - Potència consumida i de cada càrrega

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 línia de la producció està dins de l'àrea de la potència consumida és quan consumim de la xarxa. En canvi, quan està per fora, és quan en tenim excedent.

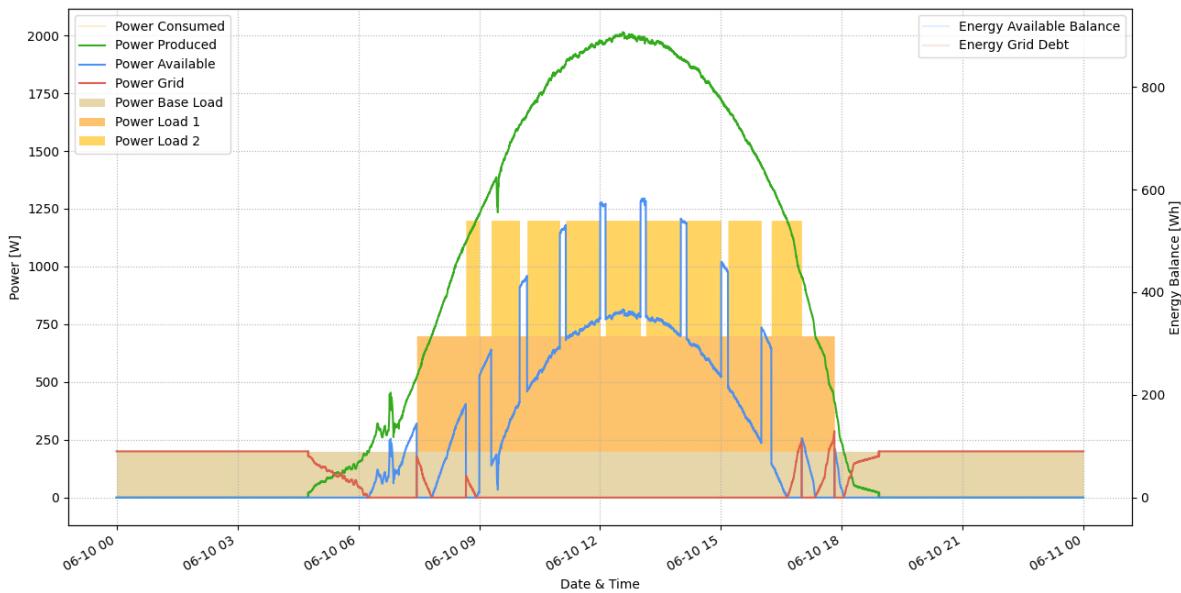


Figura 3.6. Exemple – Potència disponible i de la xarxa – Vista completa

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 càrregues donades.

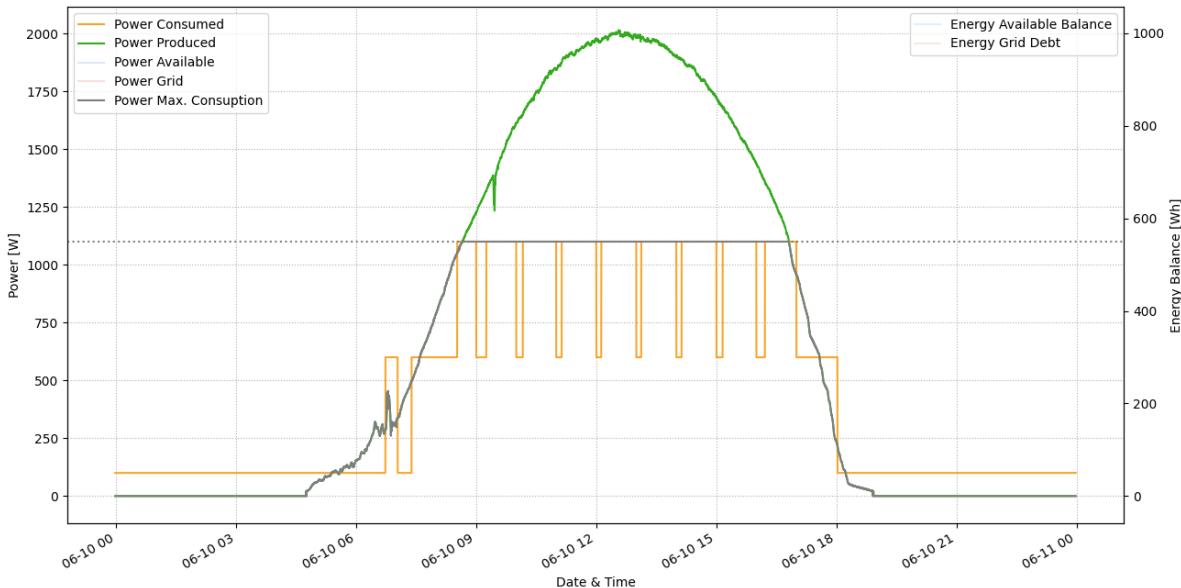


Figura 3.7. Exemple - Potència del consum màxim (ideal)

3.2. Energies

3.2.1 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.

$$eSI = eP + eX = eC + ePnC \quad (3.9)$$

- Produïda: energia produïda per les fonts d'energia renovable.

$$eP = \int_{t_0}^{t_1} pP(t)dt \quad (3.10)$$

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

$$eB = \int_{t_0}^{t_1} pB(t)dt \quad (3.11)$$

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

$$eCi = \int_{t_0}^{t_1} pCi(t)dt \quad (3.12)$$

- Consumida: energia consumida per la instal·lació.

$$eC = \int_{t_0}^{t_1} pC(t)dt = \int_{t_0}^{t_1} \sum pCi(t)dt = dt = \sum \int_{t_0}^{t_1} pCi(t)dt \quad (3.13)$$

- 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.

$$eX = \int_{t_0}^{t_1} pX(t)dt \quad (3.14)$$

- Xarxa durant prod.: energia consumida de la xarxa quan s'ha estat produint energia.

$$eXP = \int_{t_0}^{t_1} pXP(t)dt \quad (3.15)$$

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

$$eXnP = eX - eXP \quad (3.16)$$

- 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.

$$ePC = eC - eX \quad (3.17)$$

- 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.

$$ePnC = \int_{t_0}^{t_1} pPnC(t)dt = eP - ePC = eP - eC + eX \quad (3.18)$$

- Balanç: balanç net horari.

$$eB_h = \int_0^{3600} pD_h(t)dt = eB(i_h) \quad (3.19)$$

L'algoritme el fa servir per prendre les decisions, per tant, necessitarem calcular-lo a temps real. La 1a mostra al canviar d'hora l'anomenarem i_h . Hi ha dues formes de fer els càlculs.

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.

$$eB(i) = eB(i-1) + pD \cdot \Delta t \quad \text{on } eB(i_h) = 0 \quad (3.20)$$

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 $pD \cdot \Delta t$.

$$eB(i) = eB(i-1) + pD \cdot \Delta t \quad \text{on } eB(i_h) = pD \cdot \Delta t \quad (3.21)$$

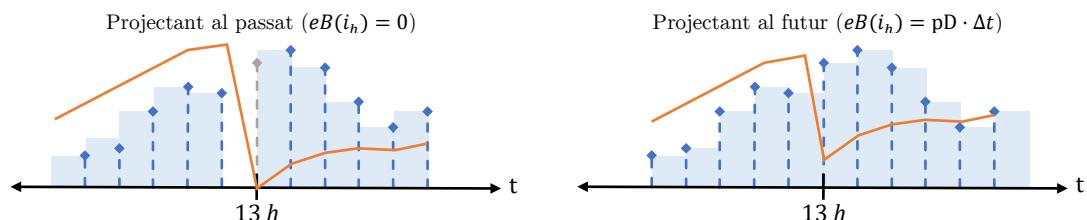


Figura 3.8. Formes de calcular el balanç

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.

$$eXD(i) = \begin{cases} eB(i) & \text{si } eB(i) < 0 \\ 0 & \text{altrament} \end{cases} \quad (3.22)$$

$$eXD = eXD(i_h) \quad (3.23)$$

- 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.

$$eBD(i) = \begin{cases} eB(i) & \text{si } eB(i) > 0 \\ 0 & \text{altrament} \end{cases} \quad (3.24)$$

$$eBD = eBD(i_h) \quad (3.25)$$

- Retornada a la xarxa: mai serà més gran que l'energia consumida de la xarxa.

$$eXR = eX - eXD \quad (3.26)$$

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

$$eCM = \int_{t_0}^{t_1} pCM(t)dt \quad (3.27)$$

- 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.

$$eS = eP - eCM \quad (3.28)$$

- No aprofitada (perduda): energia del balanç disponible que s'hauria pogut consumir. És a dir, energia que ha faltat per arribar al consum màxim.

$$ePe = \begin{cases} eBD - eS & \text{si } ePe > 0 \\ 0 & \text{altrament} \end{cases} \quad (3.29)$$

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:

$$eX = \int_{t_0}^{t_1} pX(t)dt \rightarrow eX = \sum pX_i \cdot \Delta t \quad (3.30)$$

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

Taula 3.2. Traduccions i nomenclatures – Energies

3.2.3 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.

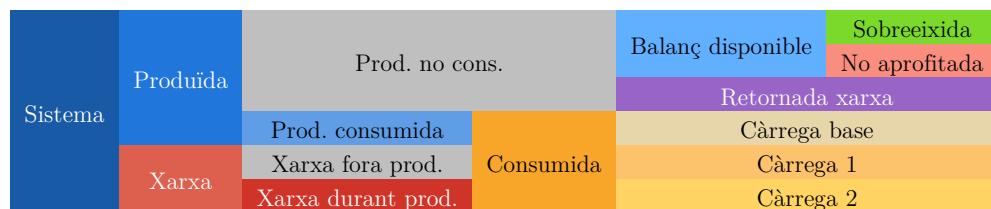


Figura 3.9. Segmentació visual de l'energia del sistema

3.2.4 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.

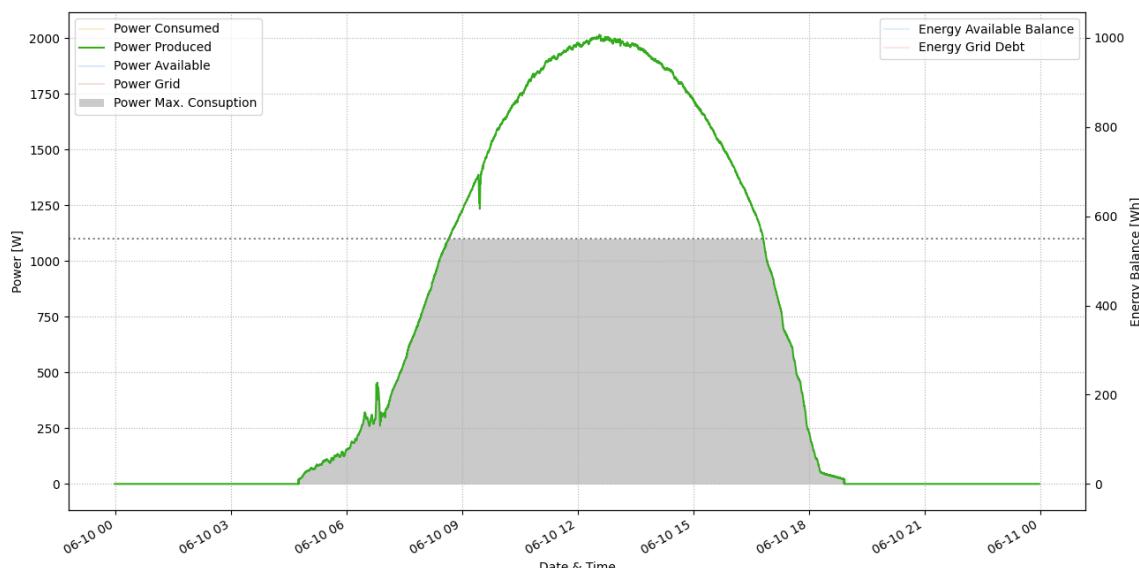


Figura 3.10. Exemple – Consum màxim d'energia a partir d'àrees

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 no aprofitada (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.

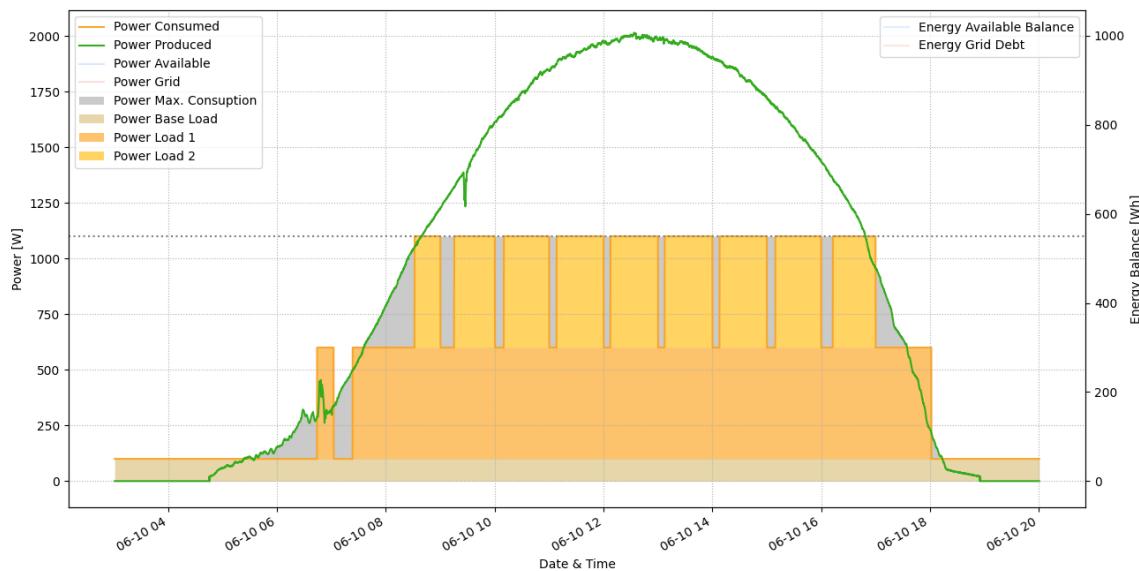


Figura 3.11. Exemple - Energia perduda a partir d'àrees

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.

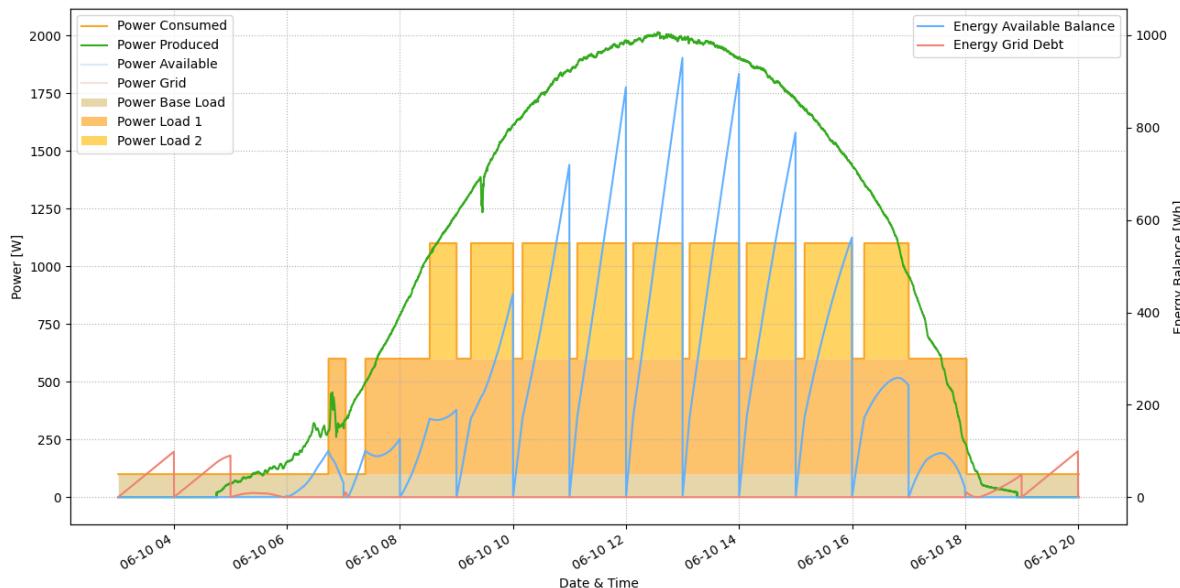


Figura 3.12. Exemple – Energia disponible del balanç i deguda a la xarxa – Vista comportament

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.

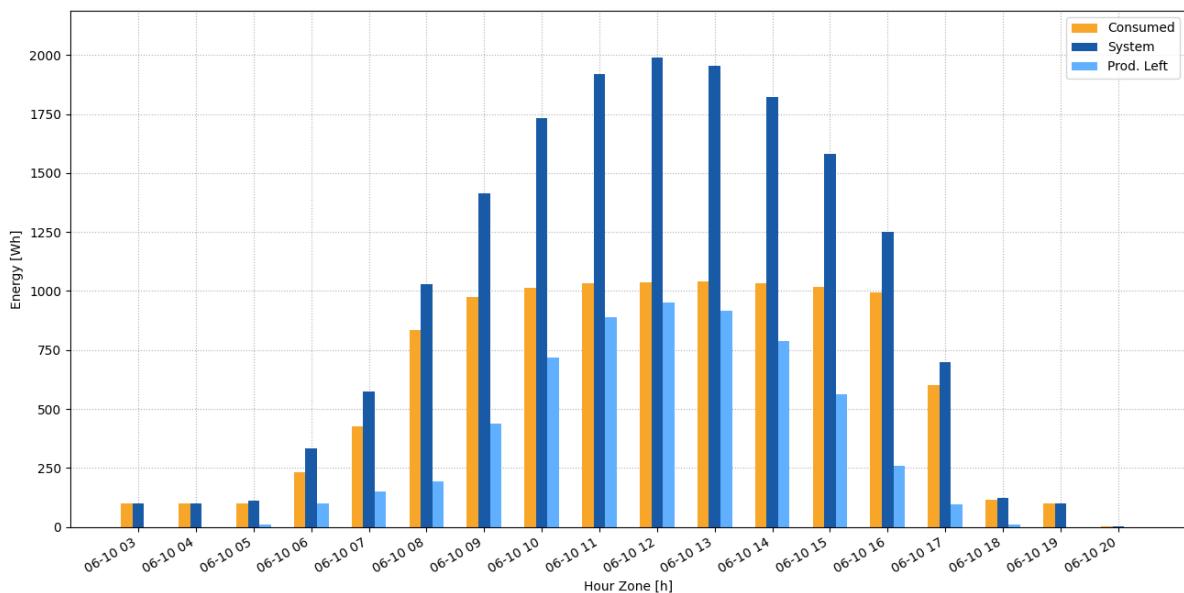


Figura 3.13. Exemple – Resultats per hores de les energies (vista model)

Amb la primera segmentació podem començar a veure coses interessants:

- Quanta energia ens consumeix cada càrrega.
- Si l'energia ha estat 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.

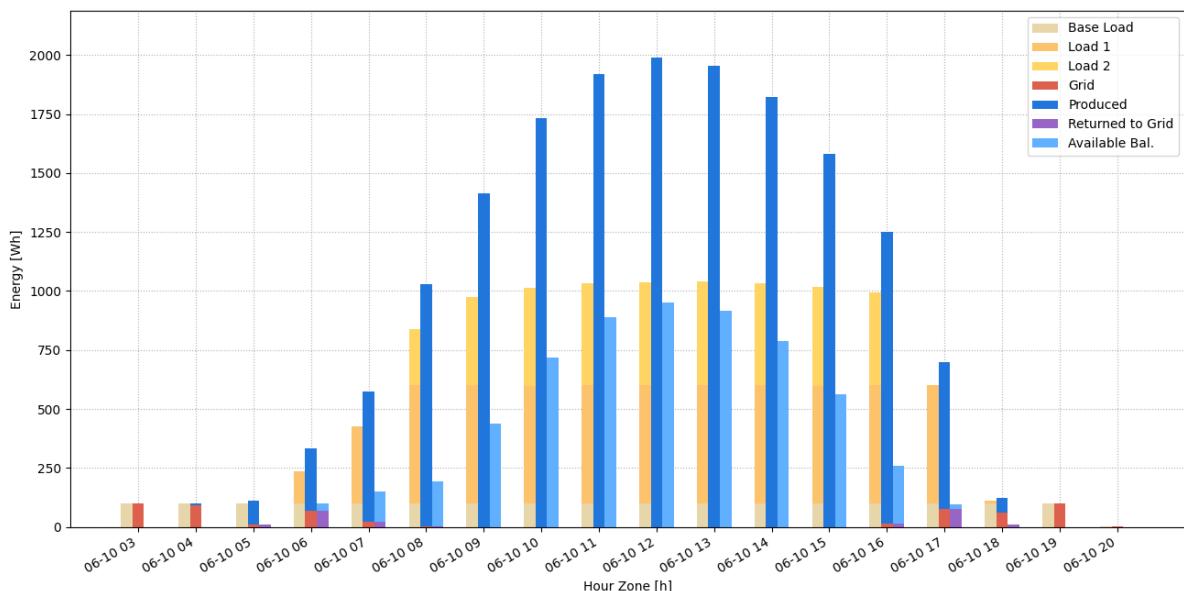


Figura 3.14. Exemple – Resultats per hores de les energies (1a segmentació)

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.

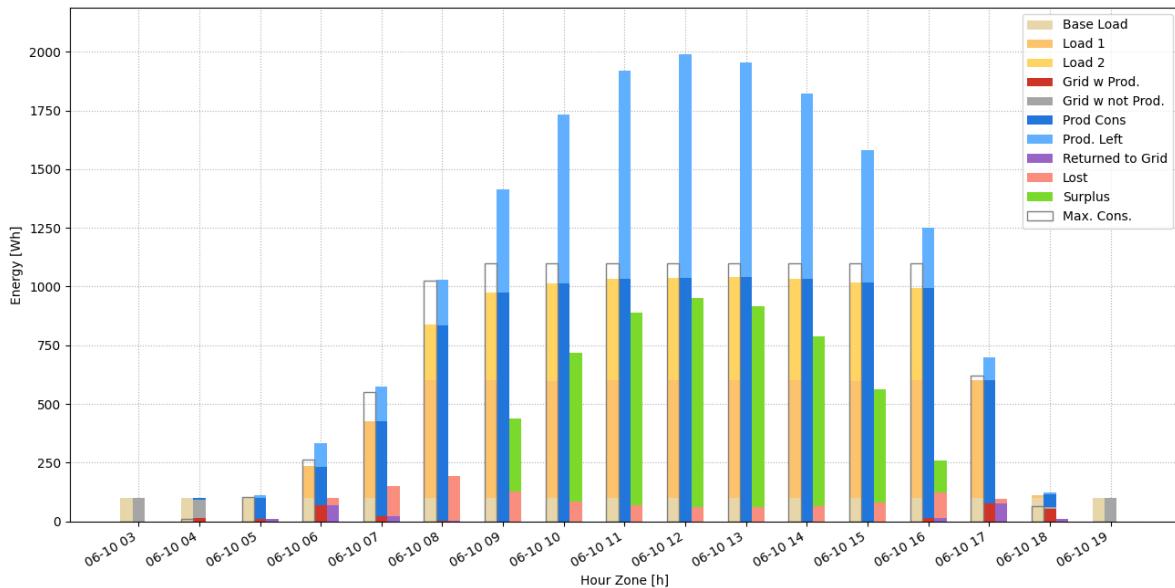


Figura 3.15. Exemple – Resultats per hores de les energies (2a segmentació)

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.

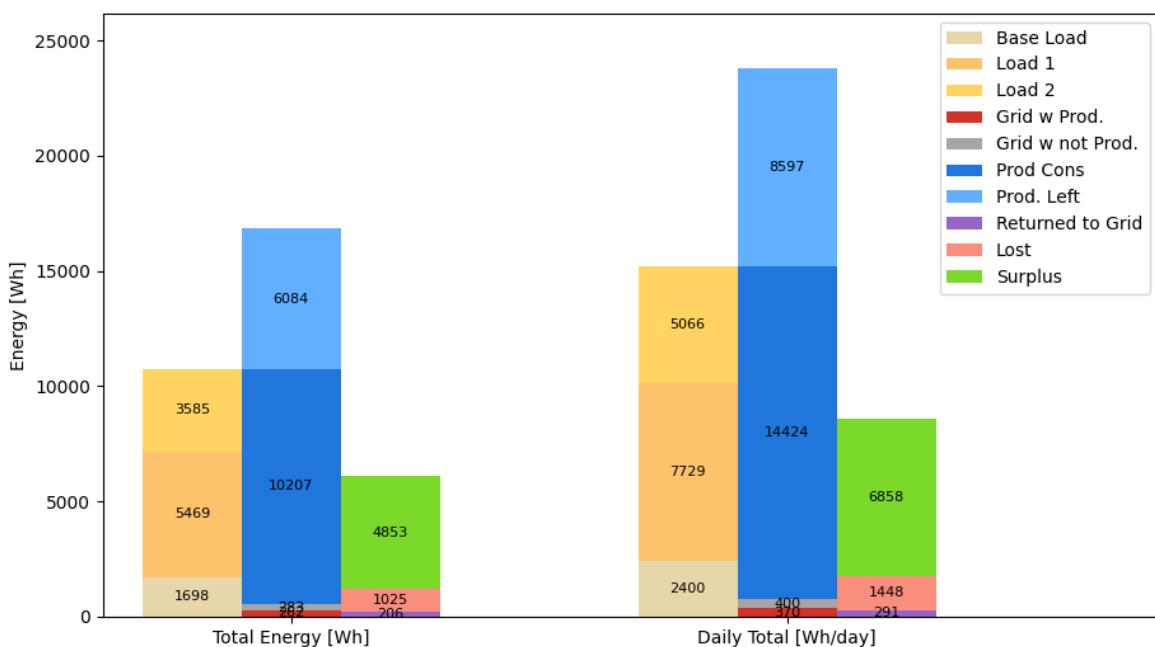


Figura 3.16. Exemple – Model energètic per hores (2a segmentació)

3.3. Per comparació

3.3.1 Definicions

Per poder comparar els diversos algoritmes a partir dels objectius marcats, s'han definit els següents termes representatius⁶:

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

$$efC = \left(1 - \frac{eL}{eCM}\right) \cdot 100 = \frac{eCP + eGR}{eCM} \cdot 100 \quad (3.31)$$

- Eficiència del retorn (a la xarxa): percentatge d'energia retornada a la xarxa respecte a la consumida de la xarxa durant la producció. La calculem perquè el consum a fora de producció només és degut a la càrrega base que el sistema no controla.

$$efXR = \frac{eXR}{eXP} \cdot 100 \quad (3.32)$$

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

$$balanç = \sum eAB_h \cdot preuVenta - \sum eGD_h \cdot preuCompra_h \quad (3.33)$$

- Commutacions: número de vegades que la càrrega s'ha engegat i apagat.

$$commut = n^o \text{ engegat} + n^o \text{ apagat} \quad (3.34)$$

- Mostres o hores (amb la càrrega X) engegada

Si a algun dels termes s'hiafegeix la paraula diàries, significa que representa un interval de 24h. Si no s'indica que la dada és diària, l'intervall 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.

$$ef_{total} = \frac{ef_1 + \dots + ef_i}{i} \quad (3.35)$$

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

$$ef_i = \frac{a_i}{b_i} \quad (3.36)$$

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

$$ef_{total} = \frac{a_{total}}{b_{total}} = \frac{\sum a_i}{\sum b_i} = \frac{a_1 + \dots + a_i}{b_1 + \dots + b_i} \neq \frac{ef_1 + \dots + ef_i}{i} = \frac{\frac{a_1}{b_1} + \dots + \frac{a_i}{b_i}}{i} \quad (3.37)$$

⁶ Això no treu que simplement mirant energies i potències puguem comparar algoritmes.

3.4. 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	<i>efC</i>	<i>efficC</i>
Eficiència del retorn a la xarxa	Grid return efficiency	<i>efG</i>	<i>efficG</i>
Balanç monetari	Balance	<i>balanç</i>	<i>balance</i>
Commutacions	Commutations	<i>commut</i>	<i>commut</i>
Hores engegada	Hours on	<i>horesOn</i>	<i>hoursOn</i>
Commutacions diàries	Daily Commutations	<i>commutD</i>	<i>commutD</i>
Hores engegada diàries	Daily hours on	<i>horesonOnD</i>	<i>hoursOnD</i>
Mostres engegada	Samples On	<i>mostresOn</i>	<i>samplesOn</i>

Taula 3.3. Traduccions i nomenclatures – Comparació

4. Algoritmes de control

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

- Histèresi
- Temps mínim engegada
- 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).

4.1. Histèresi

Ha estat el primer algoritme plantejat ja que és molt senzill, és un concepte conegut en el món tècnic i científic [4], i és utilitzat en sistemes de control, com pot ser en un termòstat.

El terme histèresi descriu el comportament d'un sistema que 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.

4.1.1 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:

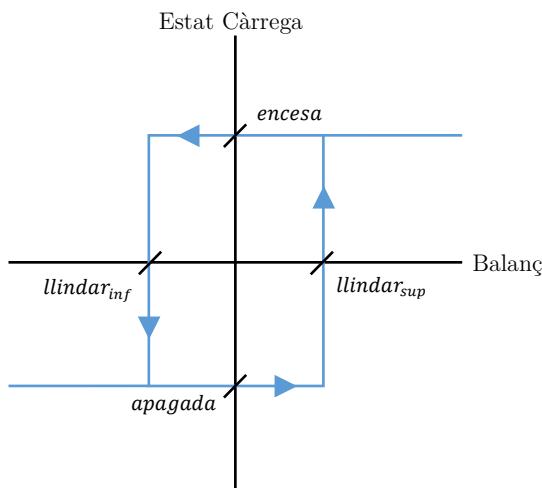


Figura 4.1. Corba d'histèresi del control d'una càrrega

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

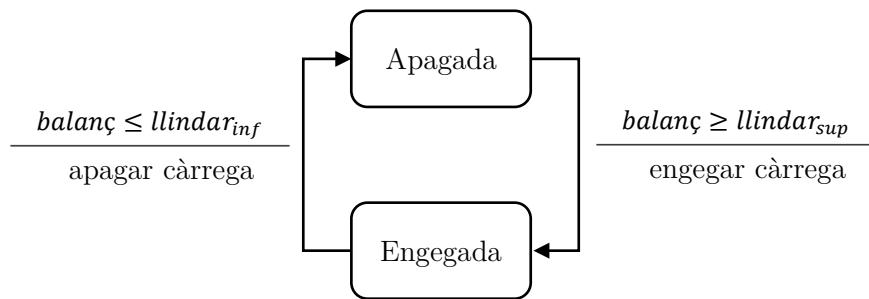


Figura 4.2. Histèresi – Màquina d'estats per controlar una càrrega

4.1.2 Dues càrregues

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

4.1.3 Observacions

Si volem que una càrrega 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.

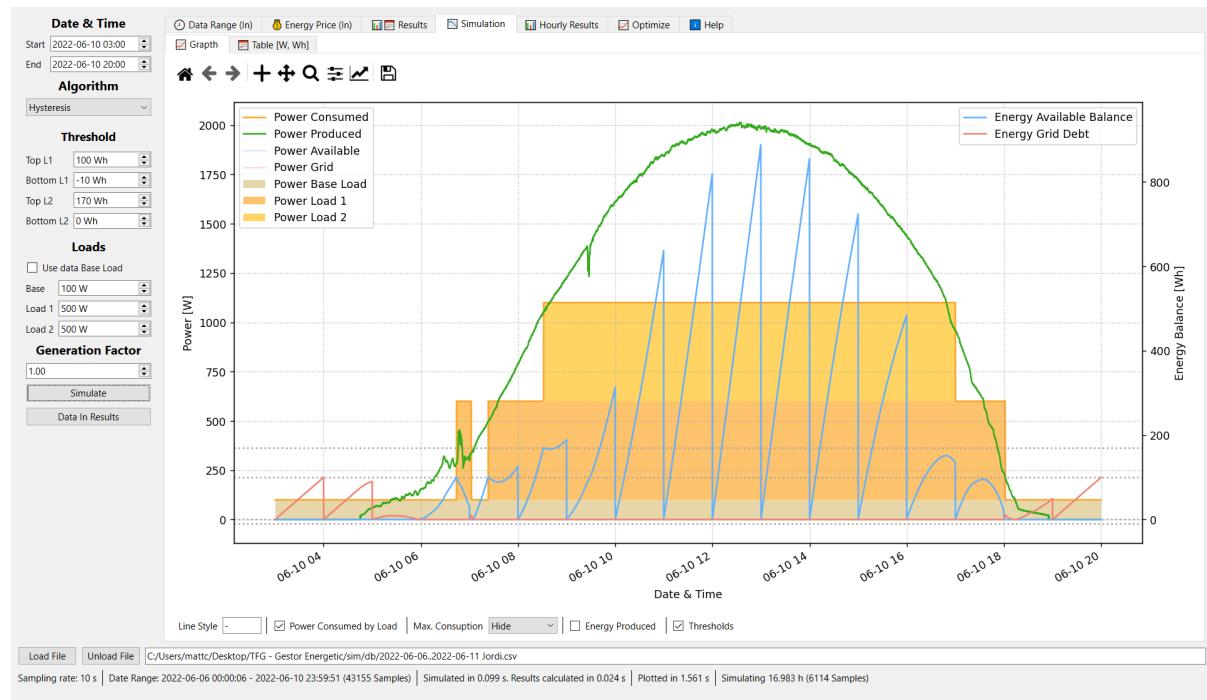


Figura 4.3. Histèresi – 1a càrrega amb preferència d'engegada respecta la 2a

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

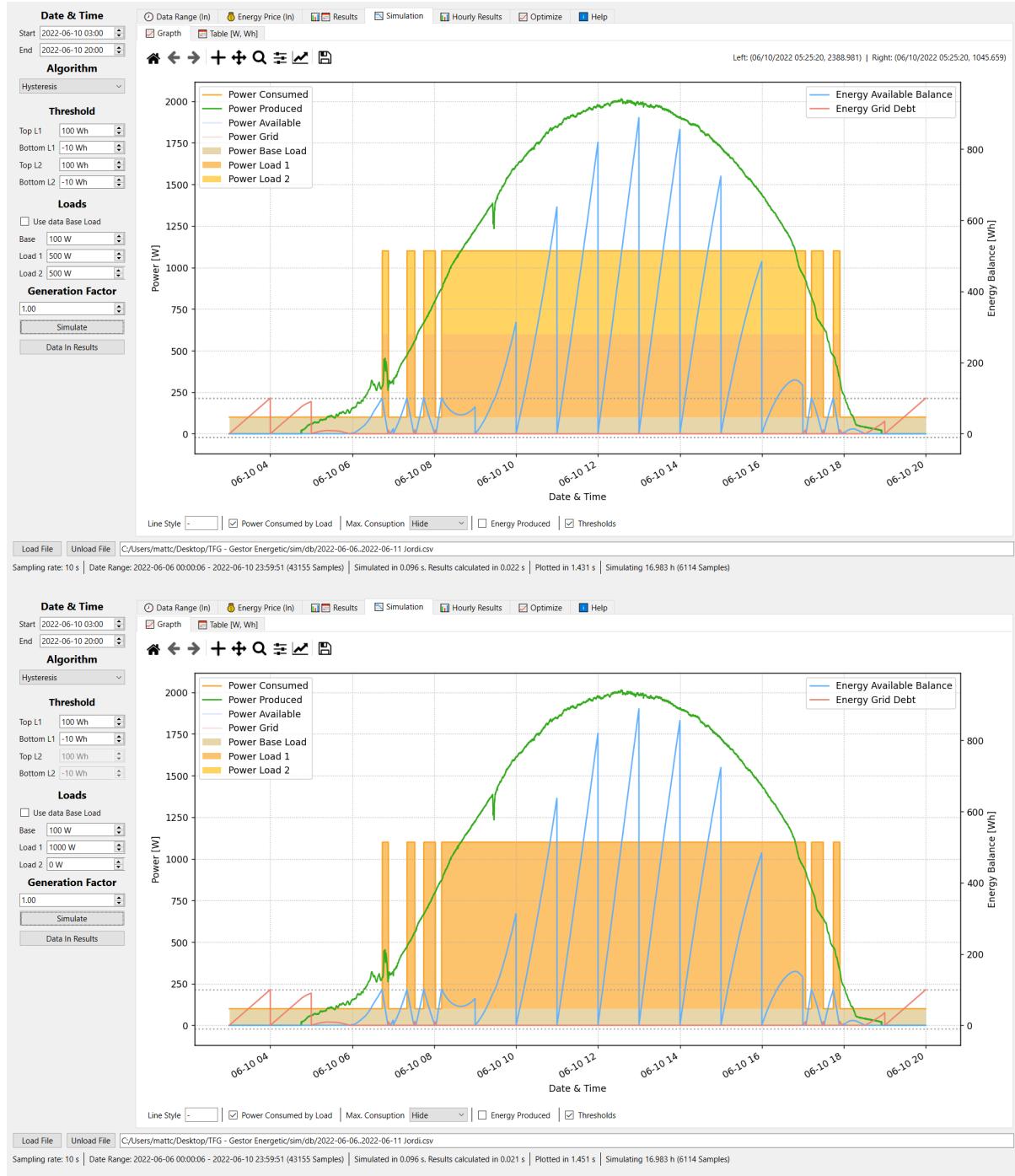


Figura 4.4. Histèresi – 2 càrregues amb els mateixos llindars equivalen a 1

A l'apartat 3.2 s'han definit dues formes de calcular el balanç a temps real i s'ha dit 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 al fet que el càlcul donaria 0.

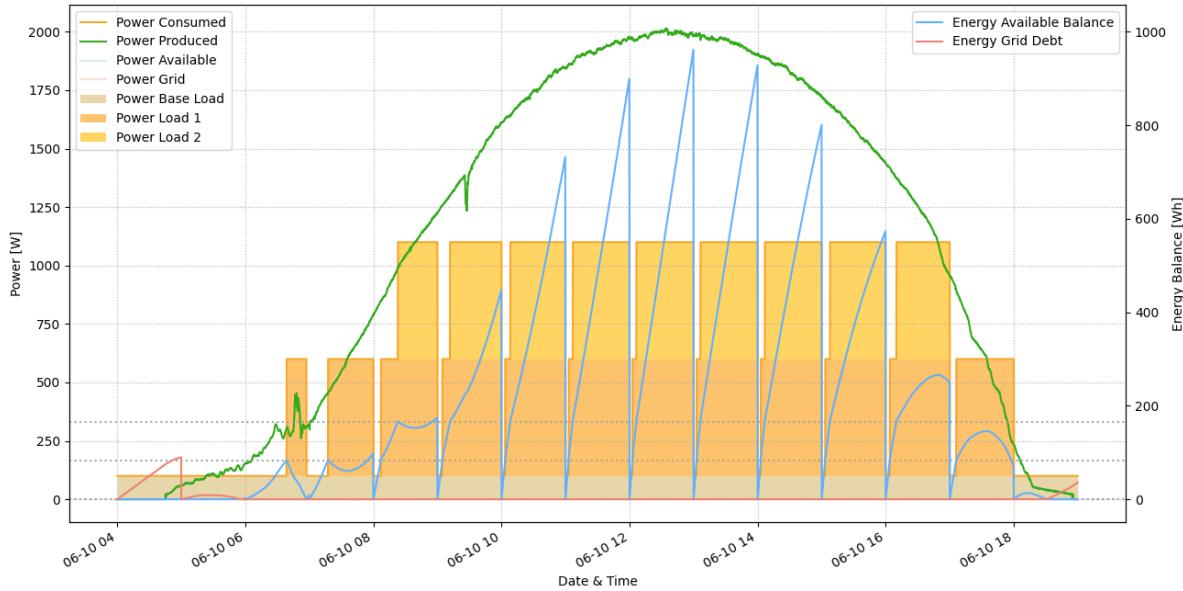


Figura 4.5. Histèresis – Llindar baix a 0 i càcul de balanç projectant al passat

En canvi, de l'altra forma només tindrem commutacions en els canvis d'hora si preveiem que el balanç en $t + \Delta t$ serà 0 o negatiu.

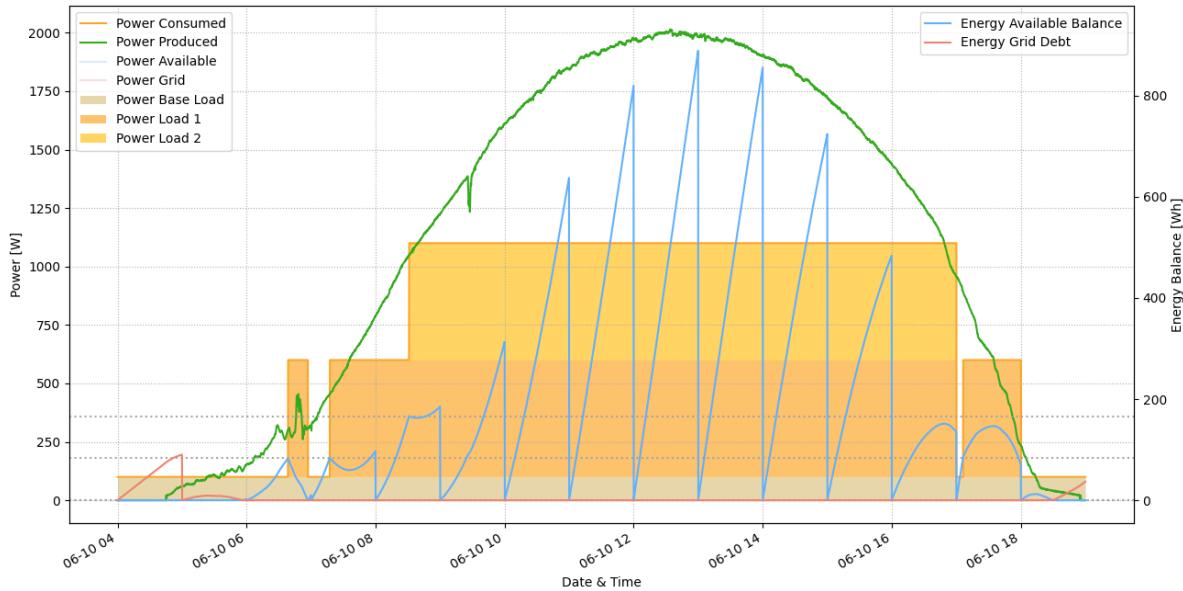


Figura 4.6. Histèresis – Llindar baix a 0 i càcul de balanç projectant al futur

4.2. Temps mínim engegada

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

Partim de la idea que es pot indicar a les càrregues que s'engeguin durant un cert temps, per exemple, engega't durant 2 minuts. Si la càrrega ja està engegada, repetir aquesta instrucció únicament posarà el temporitzador al nou valor indicat.

En el nostre cas, el sistema real fa servir relés Shelly que ja tenen la funcionalitat. Però en el cas que la càrrega no tingui la funcionalitat de temporització, l'algoritme haurà d'implementar-la. Es el que s'ha hagut de fer per simular-ho.

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 és igual o més alt al temps mínim. El temps en que es tardarà a consumir el balanç disponible dependrà de les càrregues que es vulguin engegar:

$$temp_{cons} = \frac{eB}{\sum c\ddot{a}rregaa_i} \quad (4.1)$$

4.2.1 Una càrrega

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

4.2.2 Dues càrregues

En canvi, si controllem 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 és:

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

Aquesta implementació l'anomenarem simplificada.

Però si no ens importa la preferència, una altra solució seria considerar totes les situacions possibles 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 sense engegar l'altra, necessitarem fixar un mètode per discernir quina de les dues engegar, podem fer:
 - a) Triar la càrrega seguint un ordre.
 - b) Triar la càrrega 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.

Aquesta implementació l'anomenarem completa. En funció del mètode per discernir triat, serà la variant A o B.

4.2.3 Observacions

Les tres possibles implementacions es 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:

Situació	Càrrega a engegar	Implementació	
		Completa A	Simplificada
1a	1a i 2a	-	-
2a	1a	degot a l'ordre	la 2a no es pot engegar
3a	1a	és l'única que pot consumir-ho en el temps mínim	si la 1a no ho està

Taula 4.1. Temps mín. engegada – Implementacions amb el mateix comportament 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:

Situació	Càrrega a engegar	Implementació	
		Completa A	Completa B
1a	1a i 2a	-	-
2a	1a	degot a l'ordre	consumeix més ràpid
3a	1a	és l'única que pot consumir-ho en el temps mínim	

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

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

Situació	Càrrega a engegar	Implementació		
		Completa A	Completa B	Simplificada
1a	1a i 2a	-	-	-
2a	1a	degot a l'ordre	degot a l'ordre	la 2a no es pot engegar
3a	1a	degot a l'ordre	degot a l'ordre	si la 1a no ho està

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

Si representem visualment el solapament queda de la següent manera:

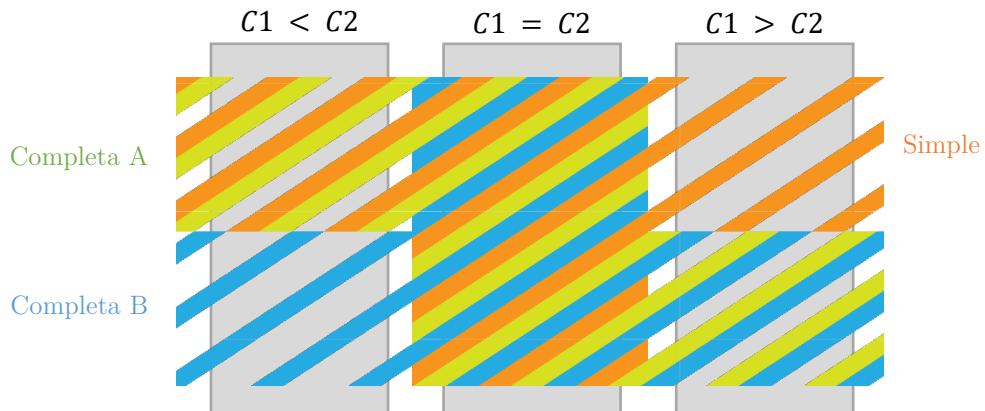


Figura 4.7. Temps mín. engegada – Solapament de comportament de les implementacions

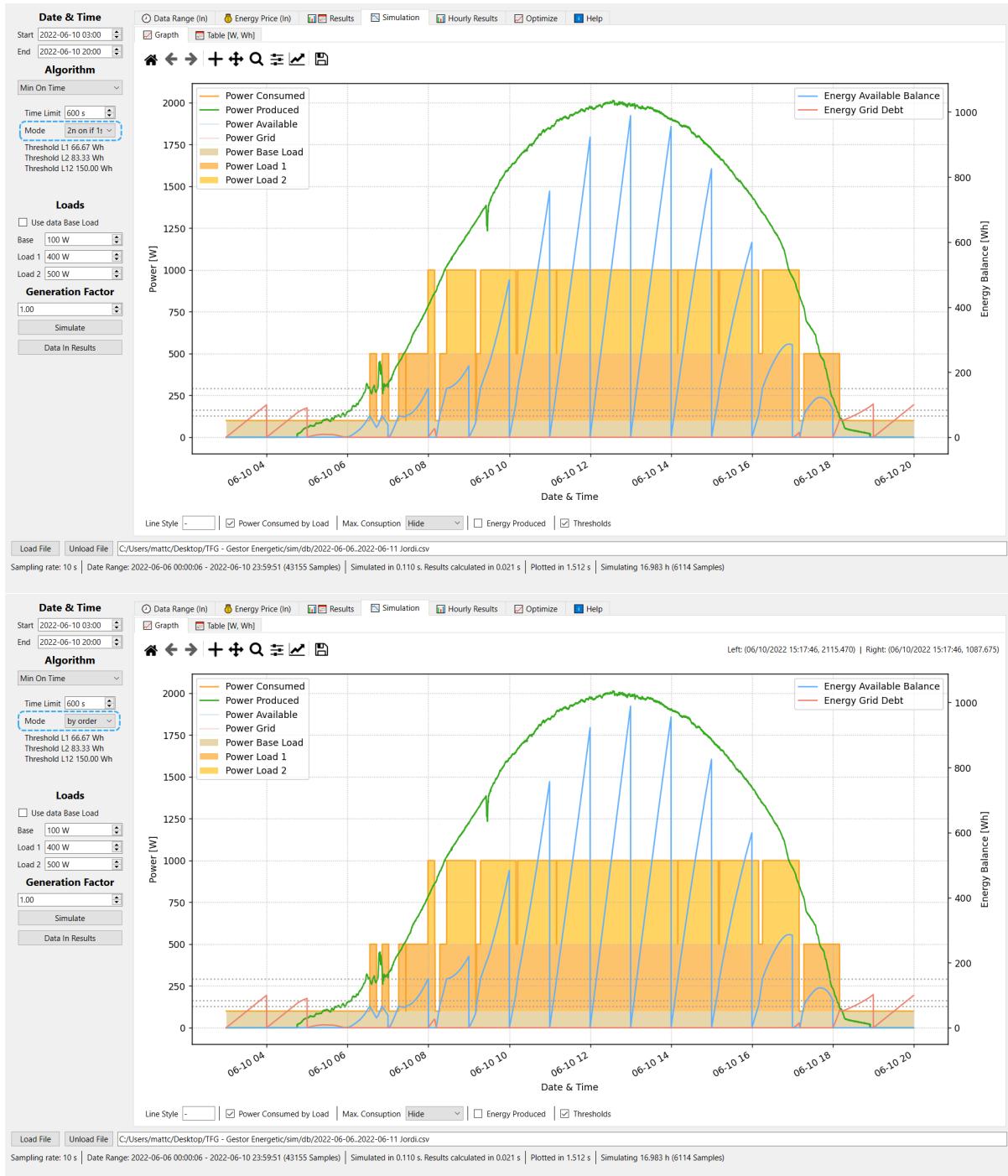


Figura 4.8. Temps mín. engegada – Implementacions amb el mateix comportament quan $C_1 < C_2$

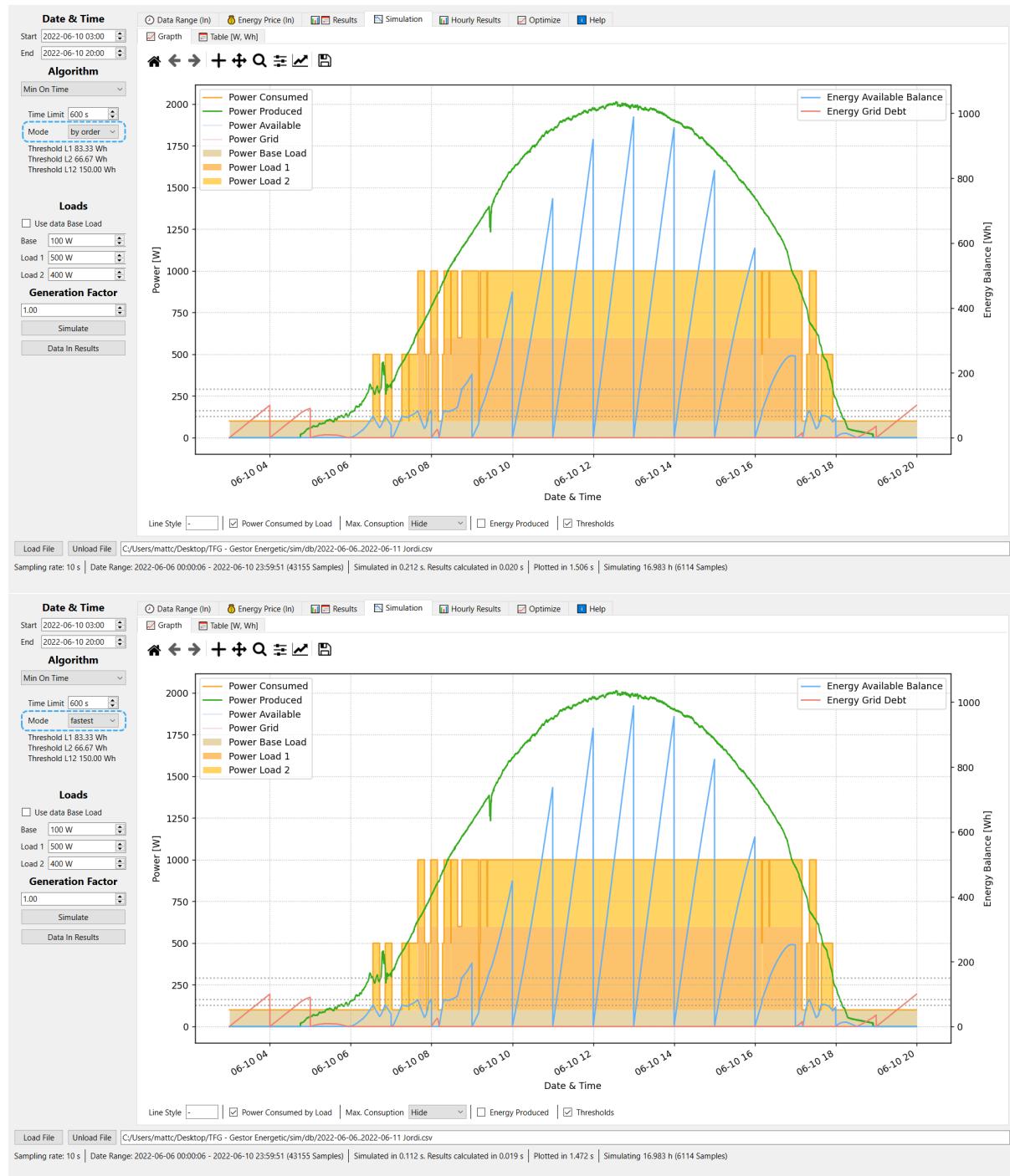


Figura 4.9. Temps mín. engegada – Implementacions amb el mateix comportament quan $C1 > C2$

Una altra característica és que el temps mínim es 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, quan el balanç esdevé menor que llindar inferior, la càrrega es manté engegada el temps mínim, evitant així múltiples commutacions.

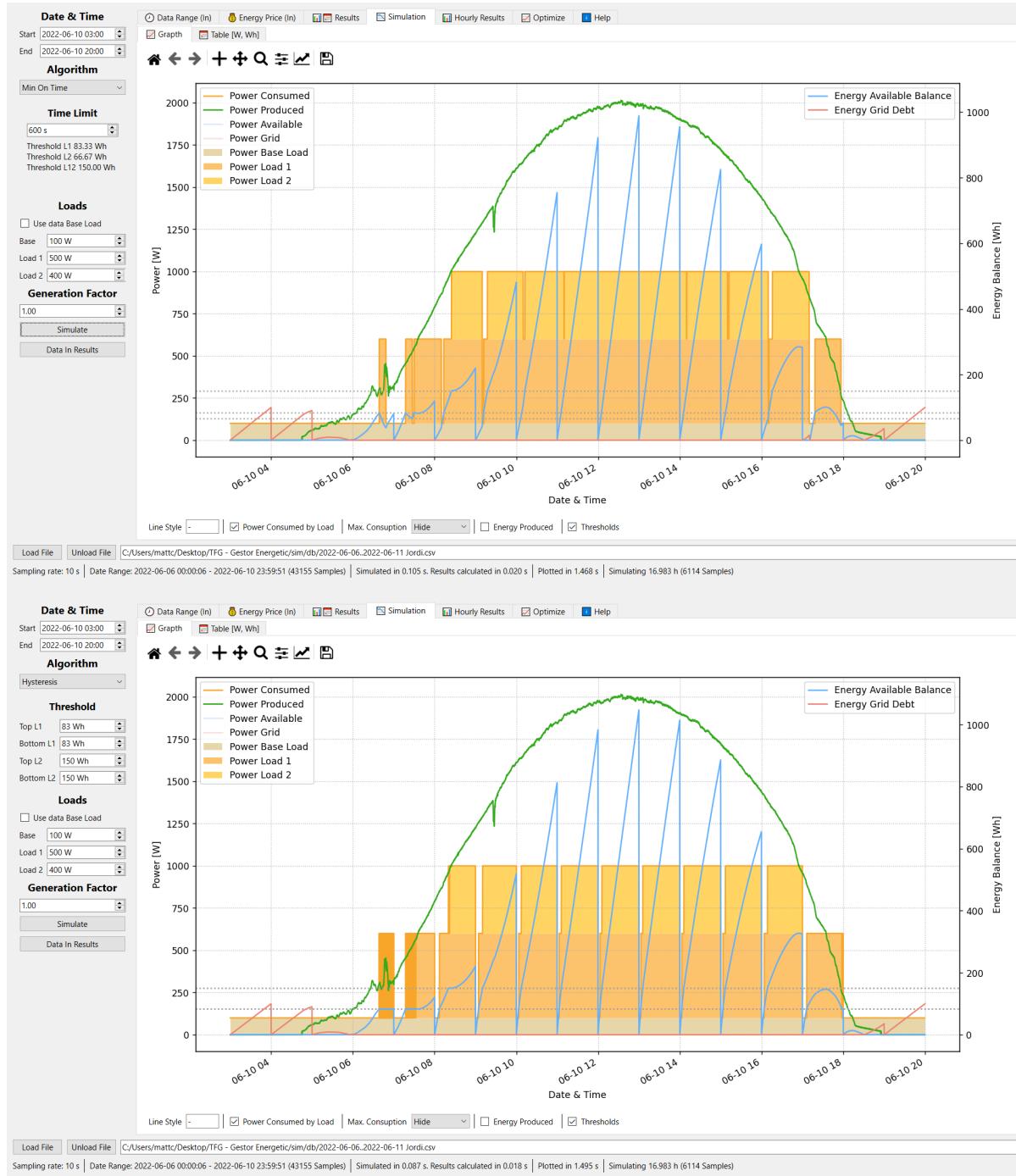


Figura 4.10. Histèresi (segona part) amb mateixos llindars que Temps mínim engegat (primera part)

4.3. Predictiu (Temps de consum)

Ha estat l'últim algoritme dissenyat i implementat ja que primer es volien implementar els 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 queda per d'acabar l'hora.

4.3.1 Com predir el balanç final

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

- Considerar que tindrem 0 producció i consum en el futur.
- Projectant la potència disponible en el que queda d'hora.
- A partir de la potència mitjana disponible.

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.

$$eB1h_{off} = eB1h + Ci \cdot temps_{restant} \quad (4.2)$$

0 producció i consum futur

Equivalent a no fer cap predicció i considerar que el balanç actual serà el balanç final horari.

$$eB1h = eB \quad (4.3)$$

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. Si ho pensem en termes d'àrea, estem afegint un rectangle.

$$eB1h = eB + temps_{restant} \cdot pPnC(t) \quad (4.4)$$

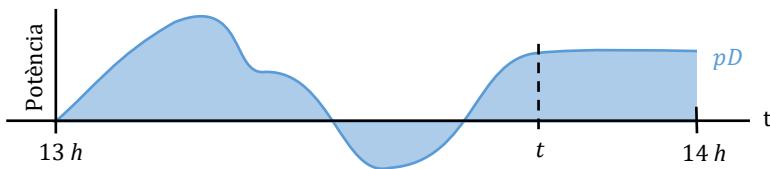


Figura 4.11. Predictiu – Predicció projectant la potència disponible

A partir de la potència mitjana disponible

Una integral es pot pensar com 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:

$$mitjana = \frac{1}{b-a} \int_a^b f(x)dx \quad (4.5)$$

$$Area = (b-a) \cdot mitjana = \frac{b-a}{b-a} \int_a^b f(x)dx = \int_a^b f(x)dx \quad (4.6)$$

Com que el balanç net horari és la integral de la potència disponible en aquella hora, podem calcular-lo a partir de la potència mitjana disponible.

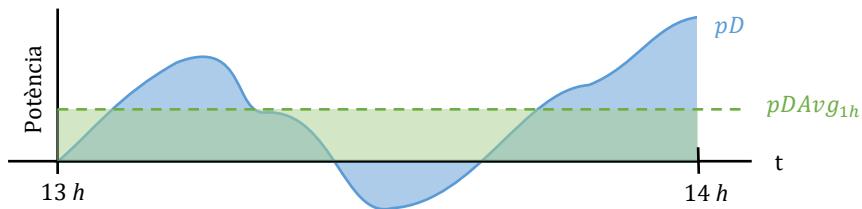


Figura 4.12. Predictiu – Exemplificació de l'àrea de $pDAvg_{1h}$ equivalent a la integral pD_{1h}

Ara bé, la potència mitjana disponible d'aquella hora no la sabrem fins que acabem l'hora. Tot i així, la podem aproximar a partir de la potència mitjana disponible que tenim en aquell moment i dins de l'interval horari.

$$pDAvg_{1h} \approx pDAvg_{h..h+t} \quad (4.7)$$

I la potència mitjana disponible fins al moment la podem calcular a partir del balanç actual, ja que, com hem dit, el balanç net horari és la integral de la potència disponible en aquella hora i, la integral d'aquesta potència, n'és la mitjana.

$$pDAvg_{h..h+t} = \frac{eB(t)}{3600 - temps_{restant}} \quad (4.8)$$

$$eB1h = pDAvg \cdot 3600 \quad (4.9)$$

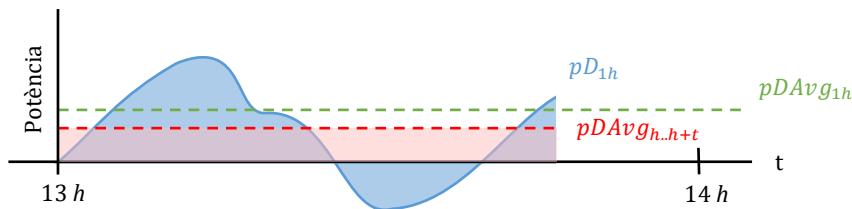


Figura 4.13. Predictiu – Predicció a partir de la potència mitjana disponible

4.3.2 Una càrrega

4.3.2.1 Plantejament

Per començar 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:

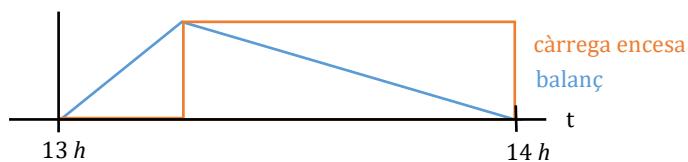


Figura 4.14. Predictiu base – Situació ideal

Ara bé, aquest primer plantejament presenta dos situacions extremes no desitjades:

- Si la producció no supleix el consum, farem moltes commutacions. Ja que, sí que fem servir la xarxa com una bateria, però no la fem servir per demanar “deute”.
- Si la producció acaba sent més de la predida, pot ser que estiguem desaprofitant energia.

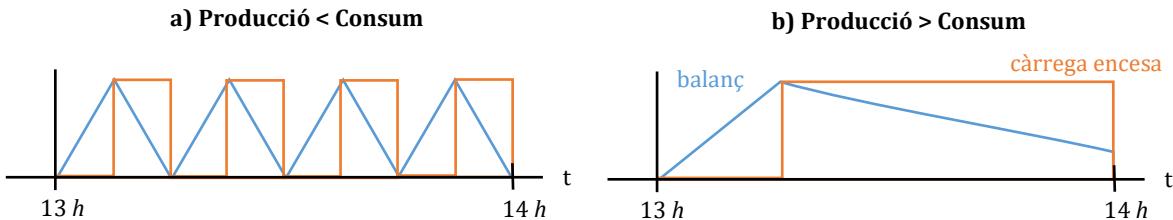


Figura 4.15. Predictiu base – Situacions extremes no desitjades

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 sigui igual o menor a 0, comptant que apaguéssim la càrrega.

Podem pensar-ho com si estiguéssim aglutinant els múltiples consums en una sola commutació. Però a la realitat és una mica “arriscat”, ja que, si a final d'hora tenim algun consum espontani o la predicció és imprecisa, podem acabar amb deute, és a dir, balanç net horari negatiu.

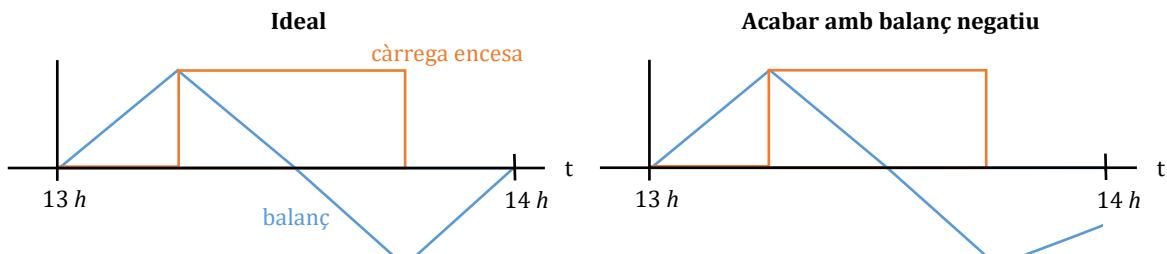


Figura 4.16. Predictiu base – Solució al problema a) i nou problema que presenta

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

A l'afegir 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'enengregar la càrrega quan el temps de consum sigui igual o major al temps restant, l'enengarem 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.

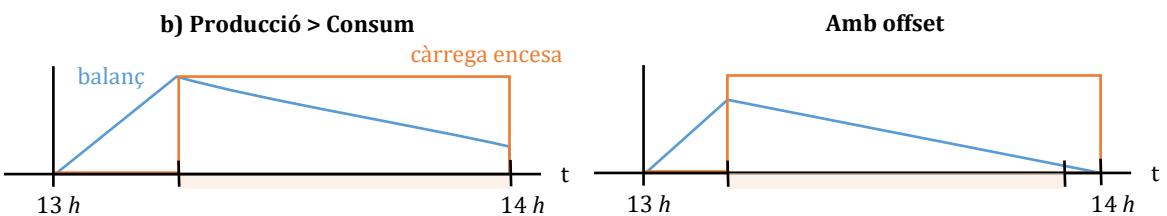


Figura 4.17. Predictiu base – Solució al problema b)

4.3.2.2 Versió final

Afegint tots els 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 és igual o menor a un llindar mínim, comptant que apaguéssim la càrrega, s'apaga.

La màquina d'estats és la següent:

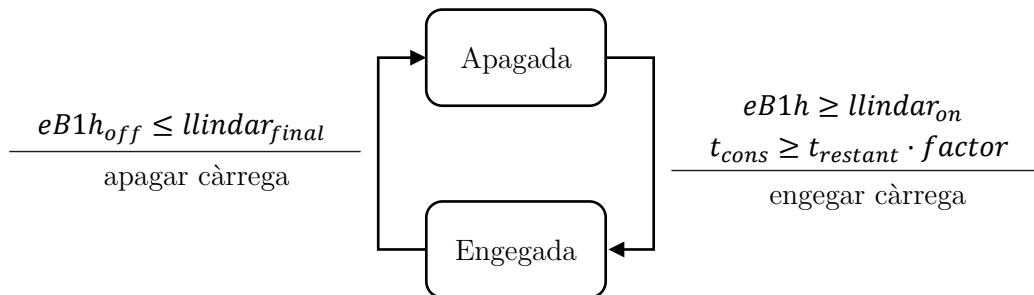


Figura 4.18. Predictiu – Màquina d'estats per controlar una càrrega

4.3.3 Dues càrregues

Per controlar dues càrregues, podríem pensar que es 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:

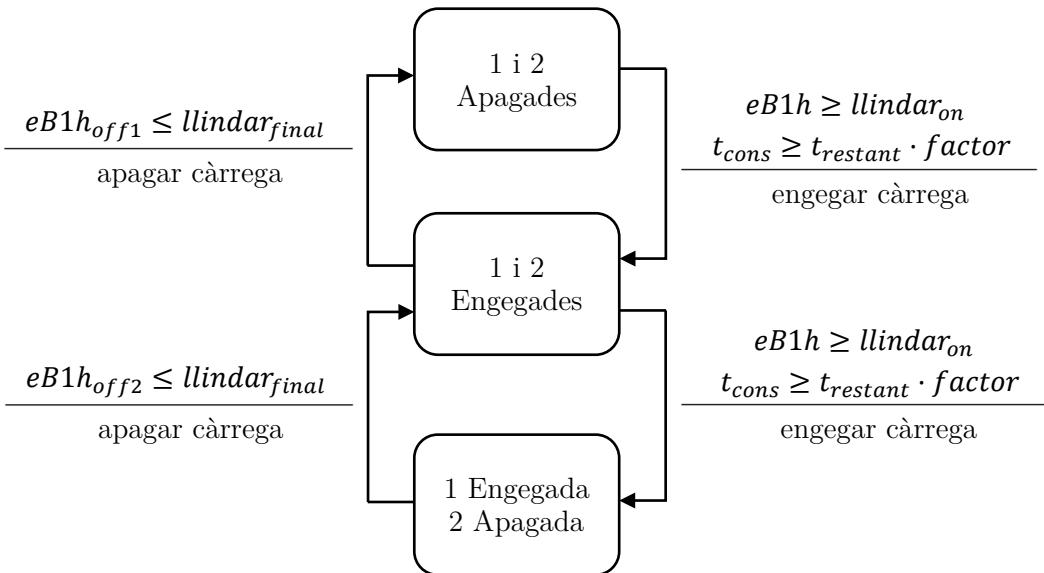


Figura 4.19. Predictiu – Màquina d'estats per controlar dues càrregues

5. Simulador

Abans d'implementar els algoritmes al món real, s'ha creat un programa per simular el sistema amb els diferents algoritmes i extreure'n resultats, amb l'objectiu de poder veure els comportaments del sistema 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 anteriorment.
- S'ha fet genèric, per poder afegir i emular qualsevol algoritme en un sol programa.
- La gran quantitat de dades i paràmetres que havia d'incloure.
- Els càlculs s'han optimitzat al màxim, per minimitzar el temps de simulació.
- S'ha fet que es puguin fer múltiples simulacions on només varia un paràmetre de la configuració, això es poden optimitzar els paràmetres dels algoritmes per a cada sistema real específic.
- S'ha fet el màxim d'accessible a l'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.

A part de simular el comportament, extreure'n resultats i generar gràfiques, també s'ha fet que els resultats i gràfiques es puguin obtenir a partir de dades reals, si aquestes tenen tota la informació que la simulació generaria, permetent així comparar la simulació amb el comportament del sistema real.

5.1. 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 [5]. Pandas està creada sobre la llibreria NumPy, que 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 fa 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 [6]. Python no té fama de ser un llenguatge molt eficient, però gràcies al nucli de la llibreria, que està escrit i optimitzat en C, la llibreria és molt eficient. És tant potent que actualment 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 [7]. A més a més, permet treballar 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+ [8]. 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, que també s'instal·la a l'instal·lar la llibreria. Una vegada dissenyades les interfícies amb Qt Designer, 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, a part de tots els components que la llibreria ofereix, s'han creat dos widgets⁷ propis: un per poder integrar les gràfiques de MatPlotLib la GUI [9] [10] i l'altre per representar les dades de Pandas com a taula [11] [12].

5.2. Implementació del simulador (programa)

A l'inici, el simulador era un sol fitxer. Això era pràctic 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:

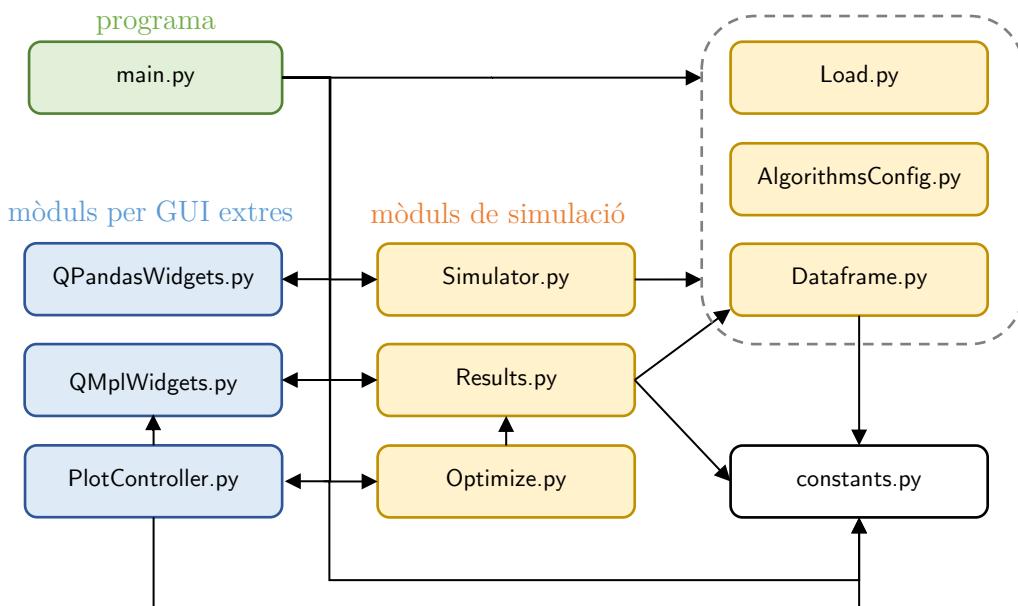


Figura 5.1. Simulador – Dependència entre mòduls

De tots els mòduls, només entrarem en el detall d'implementació del *simulator.py*, ja que és on es fa la simulació. Per la resta de mòduls explicarem què ofereixen per contextualitzar la implementació del programa.

L'avantatge d'estar programat en Python, és que és un llenguatge de molt alt nivell, així que si es vol veure com s'han implementat els mòduls, es pot llegir el codi de cada un. El programa es troba dins la carpeta `./sim` i els mòduls dins la carpeta `./sim/lib`.

⁷ Un *widget* és un bloc de construcció bàsic d'una aplicació en termes de Qt.

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 `HysteresisConfig`, `MinOnTimeConfig`, `TimeToConsume` representen les configuracions de cada algoritme implementat.

En el cas que un paràmetre de la configuració sigui un rang d'opcions, i no un número, s'ha representat 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 dataframe de Pandas que es pot fer servir per simular. Aquest comprova que el dataframe que se li passa a l'inicialitzar la classe tingui les columnes necessàries. També ofereix alguns mètodes útils.

Nom	Tipus
<code>timestamp</code>	<code>datetime.date</code>
<code>powerP</code>	<code>float, int</code>

Taula 5.1. Columnes de `DataFrameIn`

La classe `DataFrameOut` representa un dataframe de Pandas que es pot fer servir per calcular resultats. Aquest comprova que el dataframe que se li passa a l'inicialitzar la classe tingui, com a mínim, les columnes per poder calcular les columnes que faltarien per poder calcular els resultats. A més a més, al ser classe de `DataFrameIn` significa que també serveix per fer simulacions.

Nom	Tipus	Opcionals	Funció que la calcula
<code>timestamp</code>	<code>datetime.date</code>		
<code>powerP</code>			
<code>powerC</code>			
<code>powerL1</code>		<i>Si</i>	-
<code>powerL2</code>			
<code>on_offL1</code>			
<code>on_offL2</code>			
<code>powerA</code>	<code>float, int</code>		<code>calc_powerAG</code>
<code>powerG</code>			
<code>powerCM</code>			<code>calc_powerCM</code>
<code>energyAB</code>		<i>Si</i>	<code>split_energyB o</code>
<code>energyGD</code>			<code>fill_missing_energy</code>
<code>energyB</code>			
<code>energyP</code>			<code>fill_missing_energy</code>

Taula 5.2. Columnes de `DataFrameOut`

Quan s'importen les dades en el simulador, a partir de un `.csv`, es mira si és poden guardar en un objecte `DataFrameOut` o com a mínim en un `DataFrameIn`, per validar que és poden fer servir per simular i calcular resultats.

Load.py

Emulen una càrrega controlada pel sistema. S'inicialitzen indicant la seva potència de consum. Es poden engegar i apagar 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 vegades 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 que, les dades i com aquestes es representen són diferents a la classe `BarPlotController`.

QMplWidgets.py

La classe `QMplPlot` és un widget que permet integrar les gràfiques de MatPlotLib a la GUI. A més a més, a part de les funcionalitats ja ofertes 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 formatejades 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 mentre 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. A part de les funcionalitats anteriors, es poden 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 un widget model que enllaça el widget `QTableview` amb el dataframe de Pandas. Permet triar el número 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 dataframes de Pandas, aquest són:

- `.df_hour`, conté les energies, el balanç monetari, les eficiències i les commutacions de cada hora simulada.
- `.df_total`, conté les mateixes dades que `df_hour` però a nivell de simulació i a nivell diari.
- `.df_results`, conte les mateixes dades que `df_total` que no corresponen a energies, juntament amb les càrregues aproximades i el número 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 configuració de l'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`, es fan servir les dades reals de la càrrega base, en comptes d'un valor constant.

5.3. Implementació de la simulació (simulator.py)

Estructura

L'emulació del sistema es pot resumir en els següents passos:

1. Obtenir els paràmetres i dades del sistema → *Són els arguments de la funció. El programa els 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 el 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 es fa així perquè és més eficient que fer el càlcul en cada iteració.

Si mirem el codi, podem veure que està agrupat per blocs⁸, 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 evaluation (3.3)
        #region -> [Simulation] Store data to pandas dataframe (4)
        #region -> [Simulation] Calc extra sim data (5)
```

⁸ VSCode permet delimitar blocs de codi python, aquests es 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. Un bloc de codi és aquell que està entre els comentaris `#region` i `#endregion`.

Un cop simulat el sistema, el programa obté els resultats a nivell horari i total a través de la classe `Results`.

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 conté la potència base, obtindrem el valor a través de les dades reals.
- Per calcular la potència consumida, sumem les potències 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 el balanç a 0 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 temps⁹.

```
for row in df_in.df.to_dict('records'):
    #region -> Calc/Get Powers/Energies (3.1)
    power_lb = row['powerLB'] if use_df_lb 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 (3.2)
match algorithm.type:
    case AlgorithmType.hysteresis:
        ...
    case AlgorithmType.min_on_time:
        ...
    case AlgorithmType.time_to_consume:
        ...
#endregion
```

⁹ Recordar que el balanç el calculem de la segona forma definida a l'apartat 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.

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 es calcula i guarda l'instant de temps en que s'apagará.

Una vegada aplicat 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 [13], afegint temps de càcul que augmenta amb el número de mostres del dataframe [14].

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.

5.4. Instal·lació i ús

Per fer servir el simulador només cal instal·lar les llibreries utilitzades. 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 i veure totes les funcionalitats. També es 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.

6. 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. La UPC ha proporcionat la base de dades.

6.1. Descripció de la instal·lació

Font d'energia renovable

La producció d'energia renovable prové d'un conjuntament de 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'altra 1 kW. Per poder engegar, apagar i monitoritzar la potència de les càrregues remotament s'han instal·lat dos interruptors remots Shelly, un per cada alimentació.

L'interruptor Shelly és un relé intel·ligent que es pot controlar i monitoritzar per WiFi. El model utilitzat és el Shelly1, Figura 6.1, que ofereix diferents formes d'interacció [15]:

- Amb un botó físic.
- Atacant a l'API rest, fent peticions HTTP a través de la xarxa local.
- Enviant la instrucció a través del nívol o MQTT.
- Amb un horari setmanal configurat.
- A partir d'un esdeveniment generat per la sortida/la posta de sol.
- Una sobrecàrrega de potència apagará 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, a part 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 Pi¹⁰, 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 d'InfluxDB [16], amb la qual es pot interaccionar remotament a través d'una API Rest [17].

Per visualitzar les dades guardades a temps real, s'ha fet servir l'aplicació web Grafana [18], que es pot connectar a InfluxDB. El *dashboard* l'ha creat el Jordi Bonet, Figura 6.4.

¹⁰ Raspberry Pi 3 Model B Plus Rev 1.3



Figura 6.1. Shelly1



Figura 6.2. Enphase Envoy-S Metered



Figura 6.3. Raspberry Pi Model B



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

6.2. Implementació del software

A l'inici, s'havia implementat la solució en un sol codi, específic per la instal·lació i que permetia triar i configurar l'algorisme que es volia fer servir. Però tenia un problema, no era portable entre instal·lacions. Així doncs, una vegada s'ha aconseguit que funcione per el cas específic, s'ha refet tot el codi per fer-lo portable i senzill d'utilitzar. Aquest s'ha programat en Python 3.7 i es troba a la carpeta `./src`.

El primer pas ha estat definir l'estàndard dels controladors¹¹ (*drivers*) de cada tipus perifèric diferent, així l'únic que cal canviar entre instal·lacions són les configuracions dels dispositius i els drivers dels diferents dispositius. Els perifèrics són:

- La base de dades.
- El mesurador de producció i consum.
- Els dispositius per controlar les 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, la gestió i la monitorització, són independents.

Com que el sistema operatiu de la Raspberry és una distribució Linux, s'ha fet servir *systemd* per crear dos serveis, coneguts com a *daemons*. Així podem 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'ha creat una classe per cada driver. D'aquesta manera podem controlar dos perifèrics iguals 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.

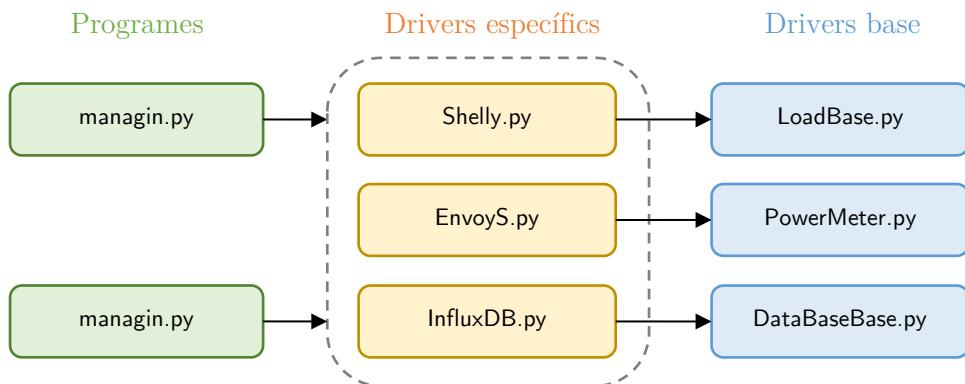


Figura 6.5. Relació programa-driver

Si és vol veure l'estàndard definit per a cada driver, o com s'han programat els drivers de cada dispositiu 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.

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).

Per implementar el driver d'InfluxDB, simplement s'han embolcallat certes funcionalitats de la llibreria oficial d'InfluxDB [19]. La implementació del driver Shelly, també ha consistit en embolcallar certes funcionalitats de la llibreria ShellyPy [20]. 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 essencials per monitoritzar són:

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

A part dels passos essencials, s'ha implementat una funcionalitat extra:

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

Igual que en el simulador, aquest passos corresponen a les diverses regions del codi per fer-lo més fàcil d'entendre.

```
# ===== Settings =====
... # (1)
# ===== MAIN =====
if __name__ == '__main__':
    #region -> Init (2)
    while True: # (3)
        #region -> Get Power/Calc Energy (3.1)
        #region -> Get Loads (on/off & power) (3.2)
        #region -> Store data (3.3)
        #region -> Wait (3.4)
```

Les dades que guarda la monitorització en cada instant de temps són:

Nom	Tipus
<i>powerP</i>	
<i>powerC</i>	
<i>energyB</i>	
<i>energyP</i>	float
<i>powerL1</i>	
<i>powerL2</i>	
<i>on_offL1</i>	1: has turned on
<i>on_offL2</i>	0: state didn't change -1: state changeed

Taula 6.1. Monitorització - Dades que es guarden a la base de dades

Es guarden només aquestes dades ja que, a partir d'elles, es poden calcular tots els termes definits en l'apartat 3 (potències, energies i valors de comparació).

Gestió (managing.py)

El passos per essencials 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 el balanç.
 2. Aplicar l'algoritme, és a dir, engegar o apagar les càrregues, només si estem en hores de treball.
 3. Esperar $Ts - temps_{execució}$. On Ts és el període de mostreig¹².

A part dels passos essencials, s'han implementat certes funcionalitats extres:

- En la inicialització de variables, s'ha fet que el programa recuperi el balanç de l'última 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 al moment.
- El gestor només controla les càrregues entre 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 a l'instant).
- Quan s'engega una càrrega, se li indica que ho faci durant 3630 segons, per evitar que si el 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 refreshar l'estat durant una hora més, just al fer el canvi d'hora.

Igual que en el simulador, aquest passos corresponen a les diverses regions del codi per fer-lo més fàcil d'entendre.

```
# ===== Settings =====
... # (1)
# ===== MAIN =====
if __name__ == '__main__':
    #region -> Init (2)
    while True: # (3)
        #region -> Get Power/Calc Energy (3.1)
        #region -> Algorithm (3.2)
        #region -> Wait (3.3)
```

6.3. 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. Configurar els *daemons* adequadament (usuari, direcció on es troba el script, ...)
5. Crear la base de dades.
6. Crear drivers necessaris.
7. Configurar els dos programes. La configuració es troba en la secció de *Settings*.
8. Indicar els *daemons*.

¹² La raó perquè ens esperem $Ts - temps_{execució}$, i no Ts , és perquè el codi tarda un temps a executar-se. Si no ho factoritzéssim, estaríem mostrejant cada $Ts + temps_{execució}$.

7. Anàlisi (sistema real)

A partir de les dades reals del sistema físic implementat i obtingudes a través del monitoratge, s'ha pogut simular els diversos algoritmes en les mateixes condicions. Les dades es troben a la carpeta `./sim/db`.

7.1. Precisió del simulador

Gran part de l'anàlisi i conclusions d'aquest projecte s'han fet a partir de simulacions. Per poder dir que són resultats fiables, hem de verificar que el simulador crea simulacions molt pròximes al comportament real. Per fer-ho hem de comparar el comportament real del sistema amb les simulacions fetes amb dades reals.

Això ho podem fer gràcies a que s'ha implementat un sistema real, s'han obtingut dades del seu comportament i el simulador permet calcular-ne resultats, a part de simular el sistema.

Com podem veure en les comparacions de l'annex A, la simulació és quasi idèntica al comportament real. Per tant, podem dir que el simulador és altament precís, sobretot si tenim el consum base real.

Si ens hi parem a pensar, té sentit. Per anar bé, el codi de la simulació (*simulator.py*) i el de la implementació real (*managing.py*), haurien de ser quasi idèntics, i en el nostre cas ho són.

7.2. Sistema real

En aquest apartat analitzarem quina és la millor configuració de cada algoritme pel sistema real. Partint dels objectius plantejats en l'apartat 1.2, la millor configuració serà aquella que (per ordre d'importància):

- Maximitzi l'eficiència de consum màxima.
- Minimitzi les commutacions diàries mínimes.
- Minimitzi el cost de la factura (pròxim 0€).
- Minimitzi l'impacte ambiental.

Com podrem veure al llarg de l'apartat, les dues primeres condicions són oposades. Ja que per consumir tota la potència possible, voldríem commutar el màxim de vegades.

Tot i així, el número de commutacions en funció de la variació d'un paràmetre acostuma a tenir una forma exponencial, per tant, acostuma a haver-hi una regió en la que les commutacions no varien gaire però si l'eficiència. Un exemple clar és la Figura 7.2.

En canvi, les dues últimes condicions estan relacionades. Si minimitzem el cost de la factura vol dir que, o no hem consumit de la xarxa o hem pogut compensar tot el consum de la xarxa, que gran part prové d'energies fòssils.

Com que la producció pot variar molt entre un dia assolat i un dia de núvols/pluja, s'han analitzat per separat, per veure si realment un sol algoritme serveix per a qualsevol situació. A més a més, per obtenir resultats mitjans, s'ha simulat més d'un dia consecutiu.

Un cop tinguem les millors configuracions podrem determinar quin és el millor algoritme pel sistema.

7.2.1 Dia assolellat

Les dades que s'han fet servir per simular el dia assolellat són les del 09/06/2022 i 10/06/2022.

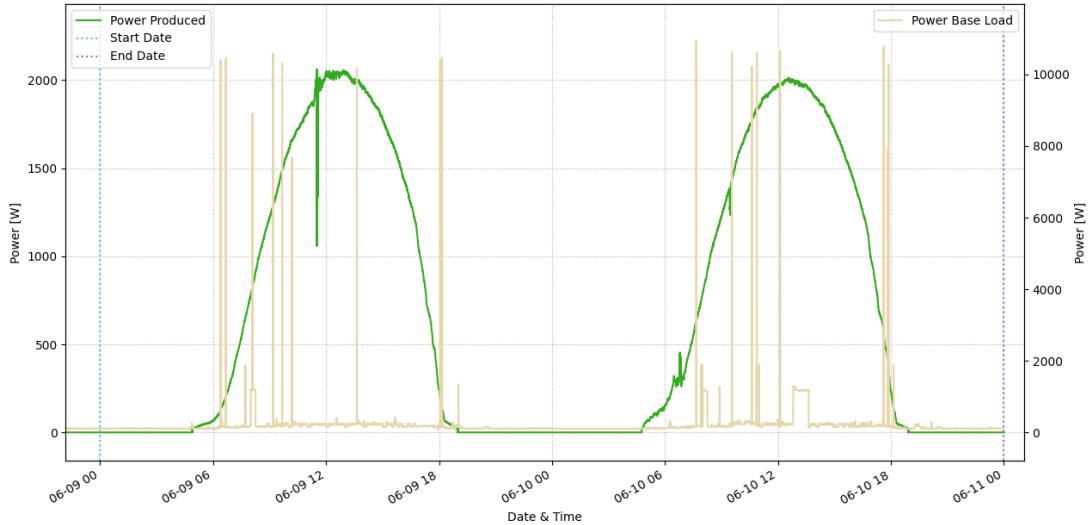


Figura 7.1. Anàlisi dia assolellat – Dades per fer la simular

7.2.1.1 Histèresis

Una sola càrrega

Per fer-ho fàcil, començarem amb una sola càrrega. El primer que necessitem saber és quin és el mínim de commutacions que podríem assolir variant el llindar alt. Així tenim una referència per saber si ens aprotem o no al límit.

En aquest cas el límit són dues commutacions, una per encendre i una per apagar. Però això és degut a que tenim energia de sobres. Si fos núvol, el límit seria més gran.

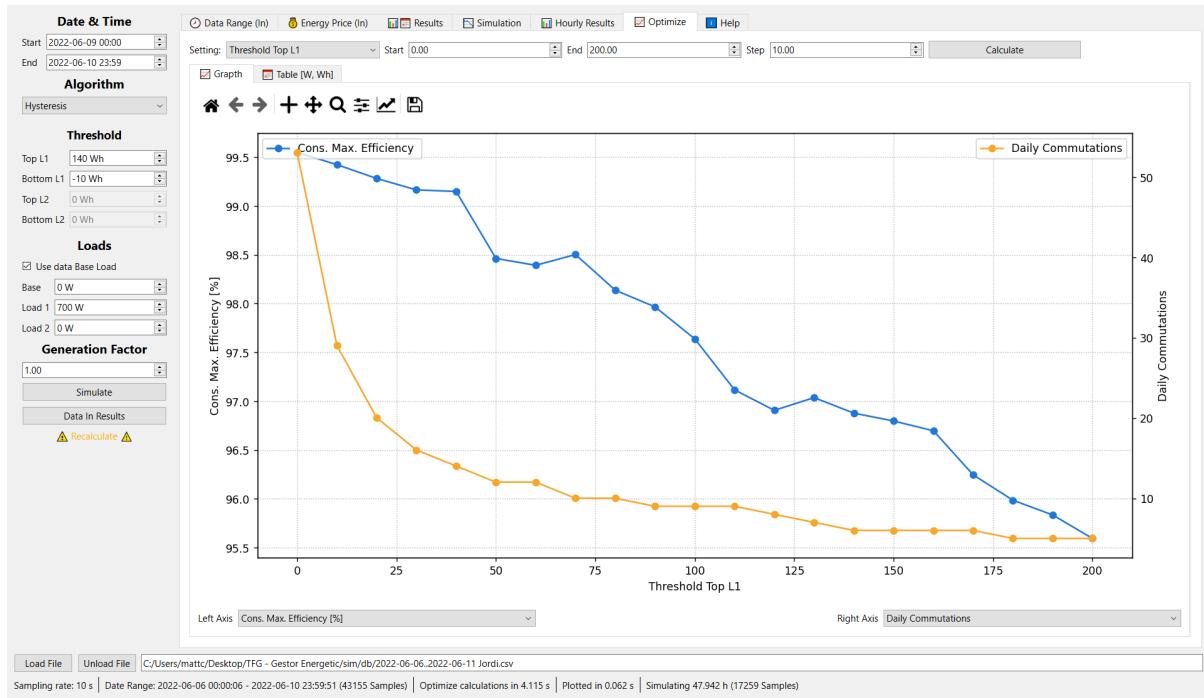


Figura 7.2 Anàlisi dia assolellat – Forma exponencial de les commutacions a l'optimitzar el llindar alt

Un cop sabem el mínim de commutacions que podem assolir, començarem optimitzant el llindar alt. Per fer-ho mantindrem el llindar baix a 0 i variarem el llindar alt entre 50 i 150 Wh en increments de 5.

Com podem veure, l'eficiència i les commutacions no varien gaire i sobra tanta energia que podem cobrir els consums de la nit amb el balanç net horari.

Hem considerat que fer 10 commutacions al dia, amb una eficiència del consum 98,15% i una eficiència de la xarxa del 72,11%, és acceptable, per tant, el llinda alt serà 70 Wh.

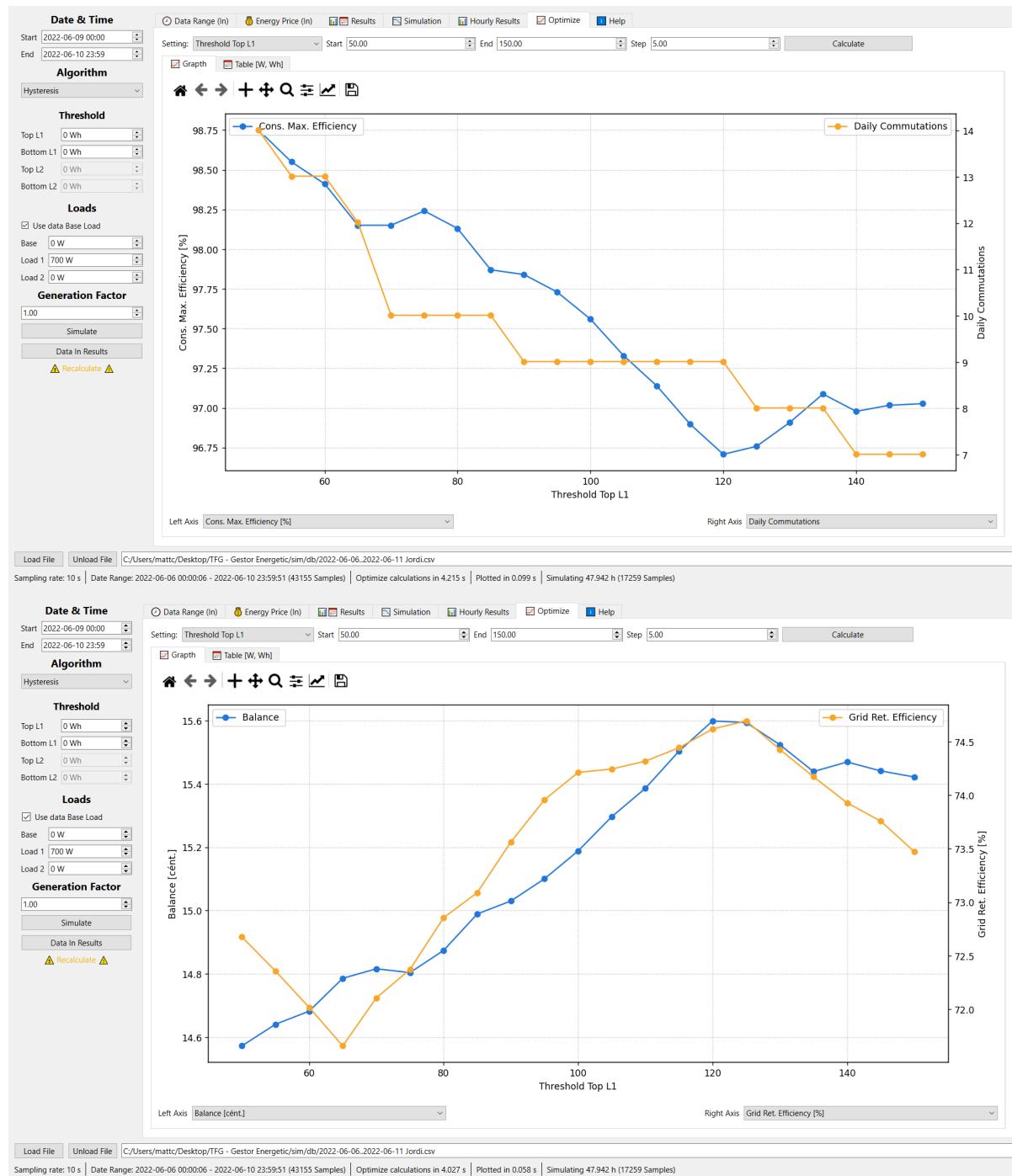


Figura 7.3. Anàlisi dia assolellat – Histèresis amb una càrrega – Optimització llindar alt

El següent pas és determinar el llindar baix. L'hem fet variar entre -50 i 60 Wh de 5 en 5.

Com que les commutacions fan un gran salt a partir de 0, hem triat un llindar baix de -10 Wh. El sistema farà 10 commutacions diàries, tindrà una eficiència del consum 98,51%, una eficiència de la xarxa del 73,62% i no haurem de pagar la factura de la llum.

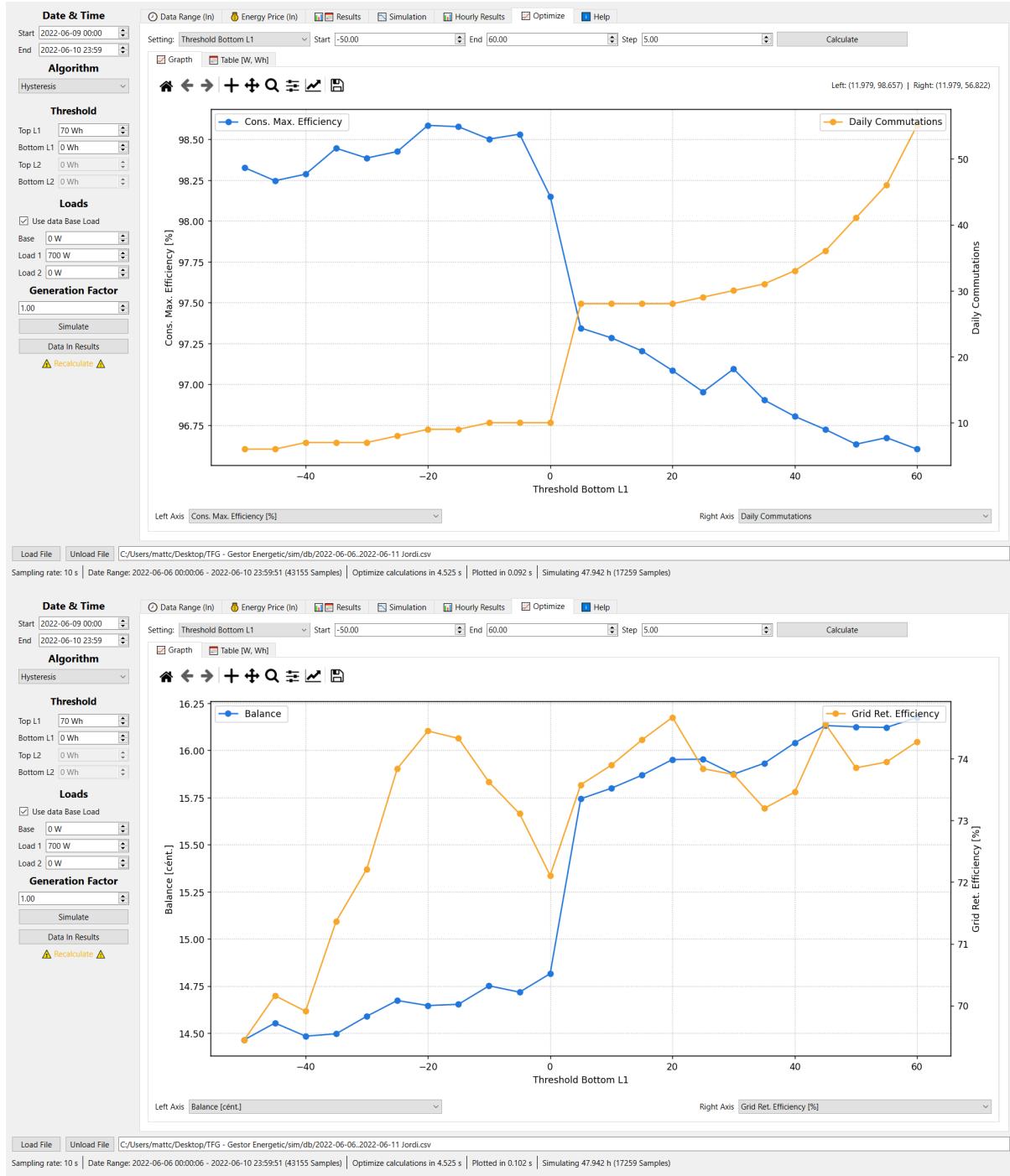


Figura 7.4. Anàlisi dia assolellat – Histèresis amb una càrrega – Optimització llindar baix

Un cop obtinguts els dos llindars, podríem tornar a mirar d'optimitzar el llindar alt i després el baix de forma cíclica, per trobar els millors valors. Però si ho provem una altra vegada, podem veure que la diferència és molt petita.

Dues càrregues

Partint dels paràmetres anteriors, hem afegit la segona càrrega i repetim els mateixos passos fets anteriorment, però ara amb els llindars de la segona càrrega.

Com podem veure, el mínim de commutacions que pot fer el sistema, si variem el segon llindar alt, s'estabilitza a 26. Si volem mantenir l'eficiència alta, haurem de triar moltes commutacions (que bàsicament les fa la segona càrrega). Com que no volem desgastar els relés massa ràpid, posarem el llindar a 200 Wh i tindrem 27 commutacions diàries, una eficiència del 93,93%, una eficiència de retorn a la xarxa del 85,92% i haurem de pagar uns 1,8 cèntims/dia de factura.

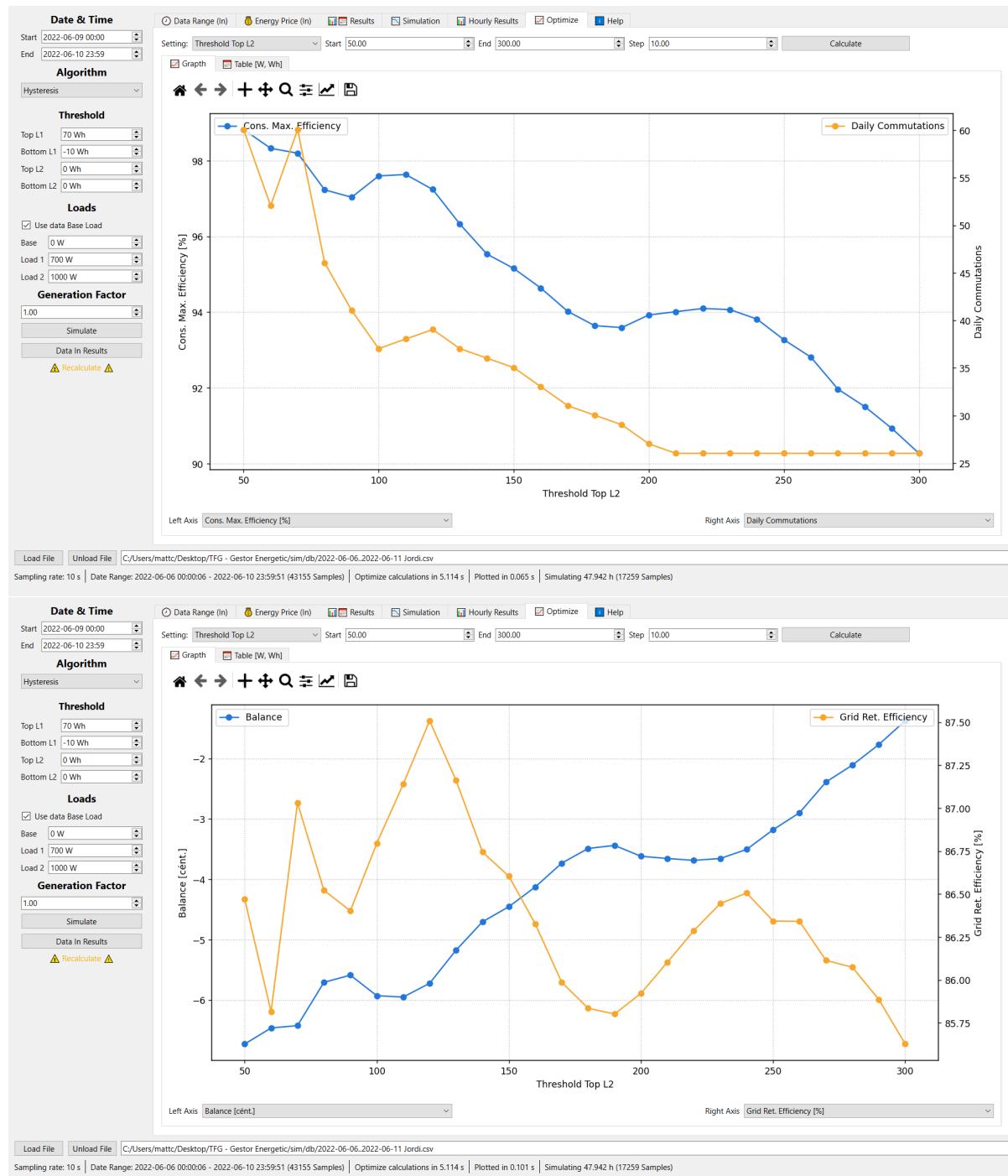


Figura 7.5. Anàlisi dia assolellat – Histèresis amb una càrrega – Optimització llindar alt 2

Finalment, optimitzem el llindar baix. En aquest cas hi ha un clar guanyador, 0Wh, que és el que ja teníem. És el llindar que té menys commutacions, quasi la millor eficiència de retorn a la xarxa, el balanç és pròxim als millors i l'eficiència està just al mig.

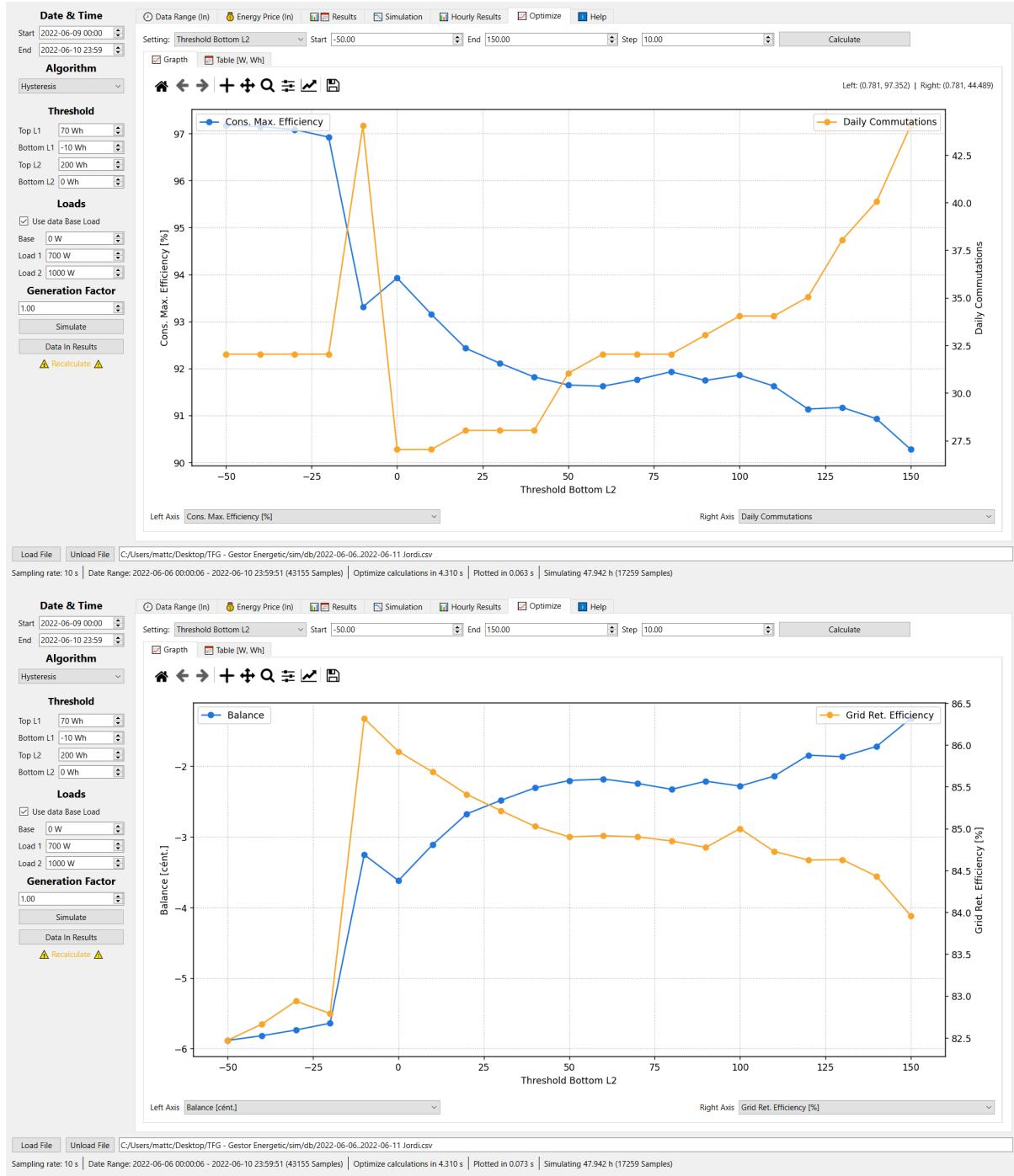


Figura 7.6. Anàlisi dia assolellat – Histèresis amb una càrrega – Optimització llindar baix

7.2.1.2 Temps mínim engegada

Com s'ha explicat en l'apartat 4.2, l'algoritme té 3 formes d'implementar-se. Com que les càrregues es poden ordenar de 2 formes diferents, podríem pensar que hem de simular el sistema 6 vegades (3×2). Però com s'ha explicat en les observacions 4.2.3, cada variant de l'algoritme té comportaments solapats amb altres variants en funció de les càrregues. Això redueix les simulacions a 4.

Càrrega 1 < Càrrega 2

Solapament versió Simple i Complexa A: Buscant el compromís entre eficiència i commutacions, una opció acceptable és un temps mínim de 600 segons. El sistema faria 50 commutacions al dia, tindria una eficiència del 85,54%, una eficiència de retorn a la xarxa del 85,28% i no caldría pagar la factura.

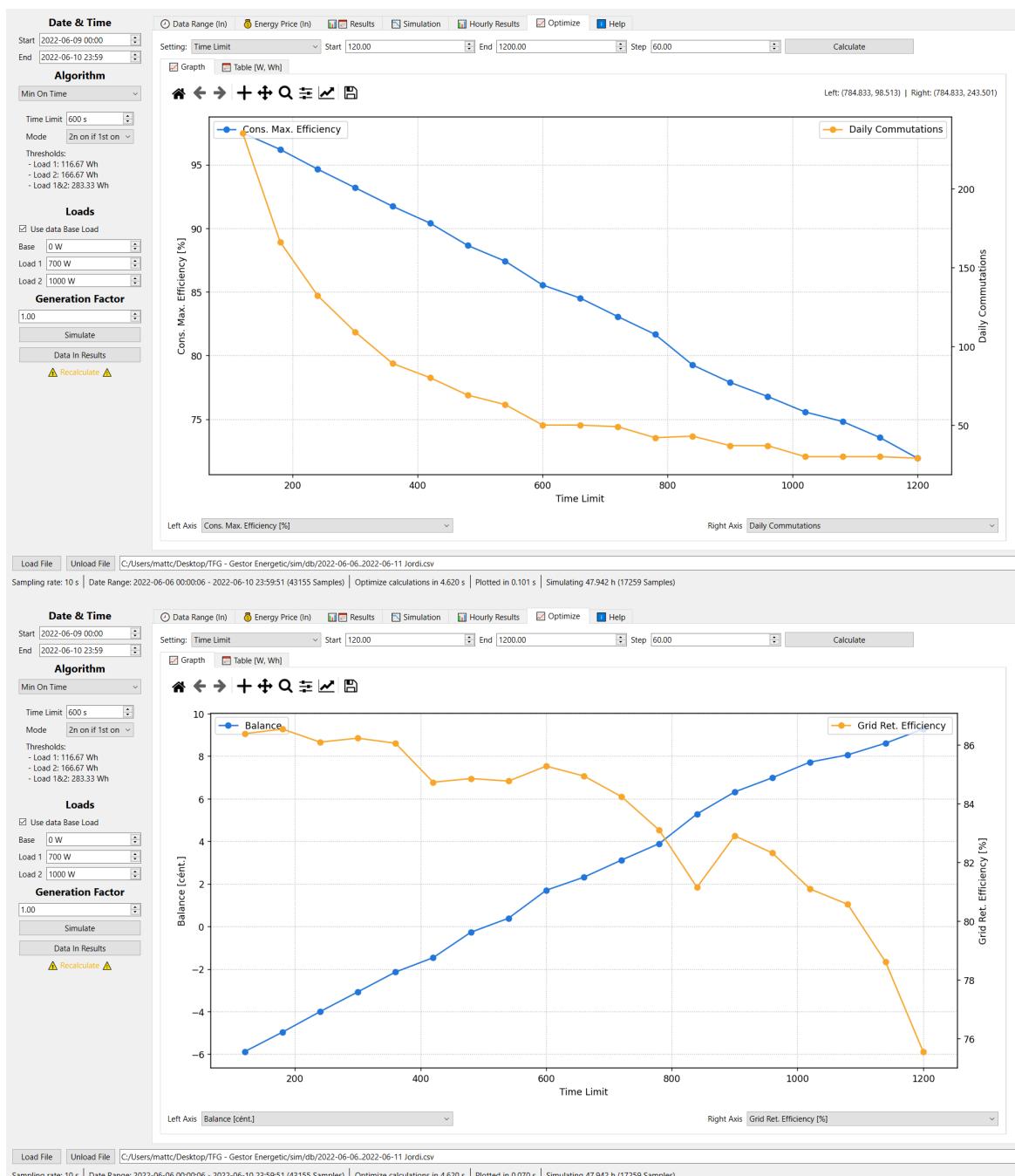


Figura 7.7. Anàlisi dia assolellat – Temps mínim engegada – C1 < C2, versions Simple i Completa A

Versió Complexa B: Buscant el compromís entre eficiència i commutacions, una opció acceptable és un temps mínim de 1020 segons. El sistema faria 50 commutacions al dia, tindria una eficiència del 84,88%, una eficiència de retorn a la xarxa del 84,69% i no caldria pagar la factura.

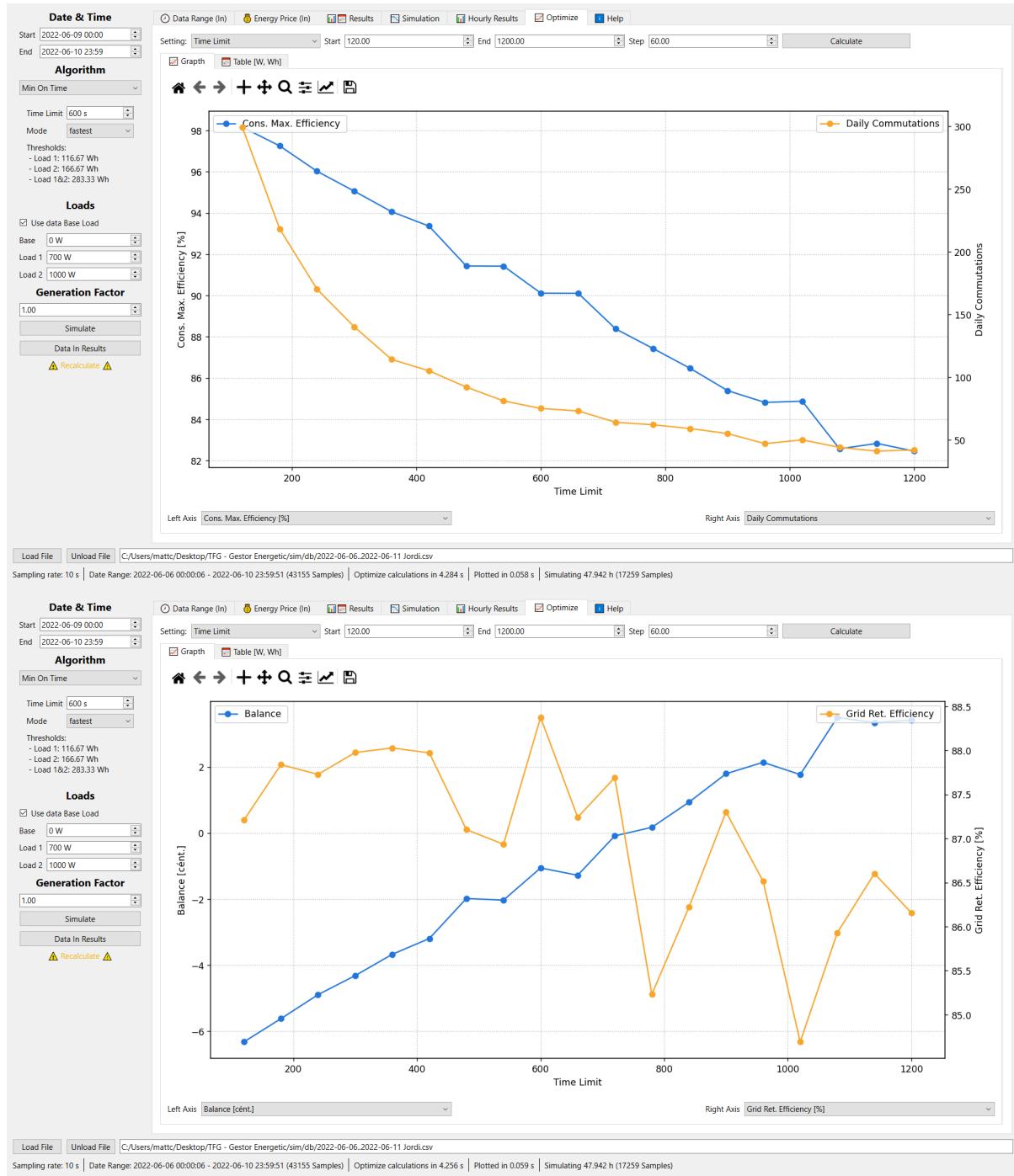


Figura 7.8. Anàlisi dia assolellat – Temps mínim engegada – $C1 < C2$, versió Completa B

Càrrega 1 > Càrrega 2

Versió Simple: Buscant el compromís entre eficiència i commutacions, una opció acceptable és un temps mínim de 600 segons. El sistema faria 50 commutacions al dia, tindria una eficiència del 86,07%, una eficiència de retorn a la xarxa del 84,11% i no caluria pagar la factura.

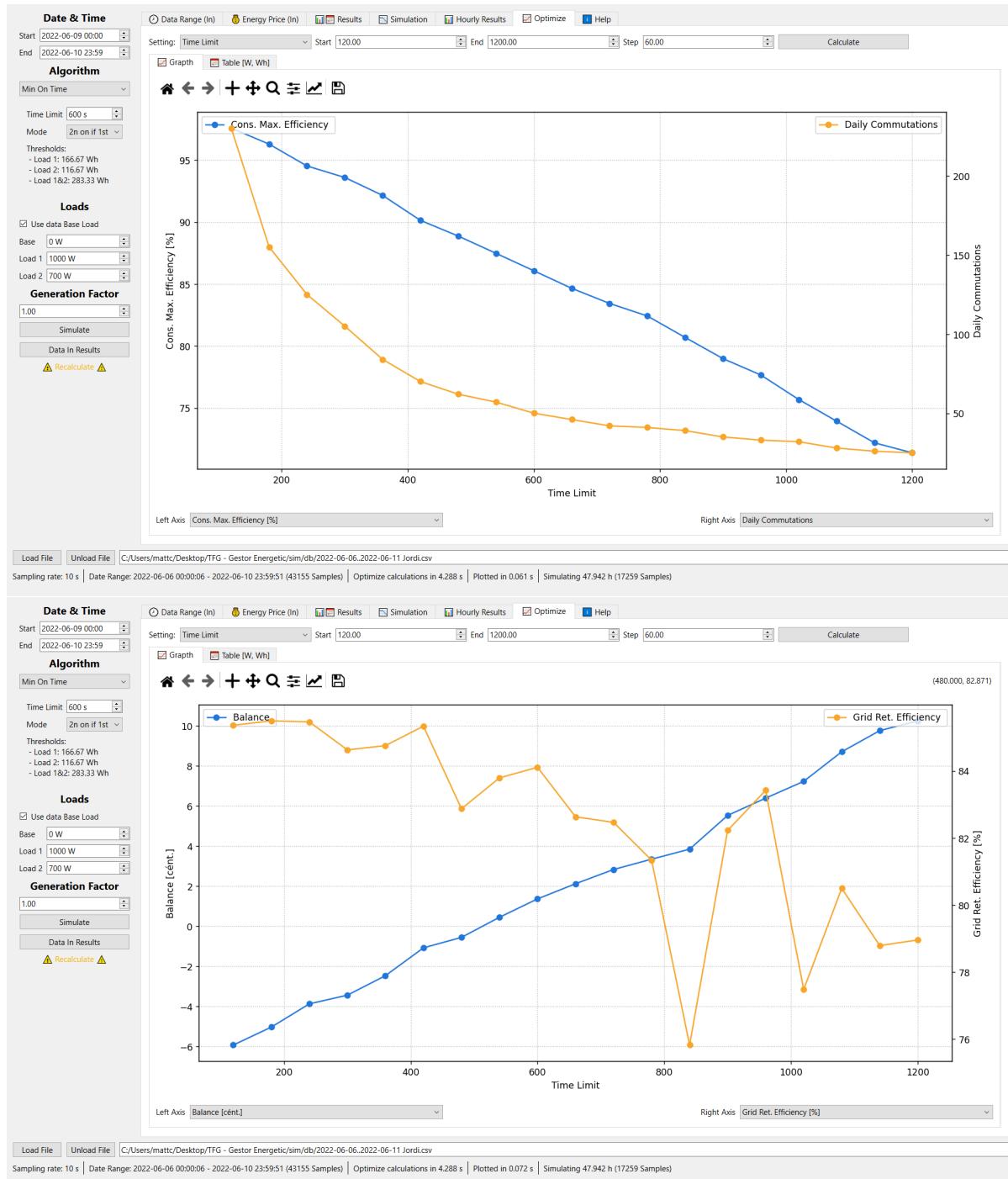


Figura 7.9. Anàlisi dia assolellat – Temps mínim engegada – C1 > C2, versió Simple

Versió Complexa A i B: Buscant el compromís entre eficiència i commutacions, una opció acceptable és un temps mínim de 600 segons. El sistema faria 50 commutacions al dia, tindria una eficiència del 84,87%, una eficiència de retorn a la xarxa del 84,69% i no caldria pagar la factura.

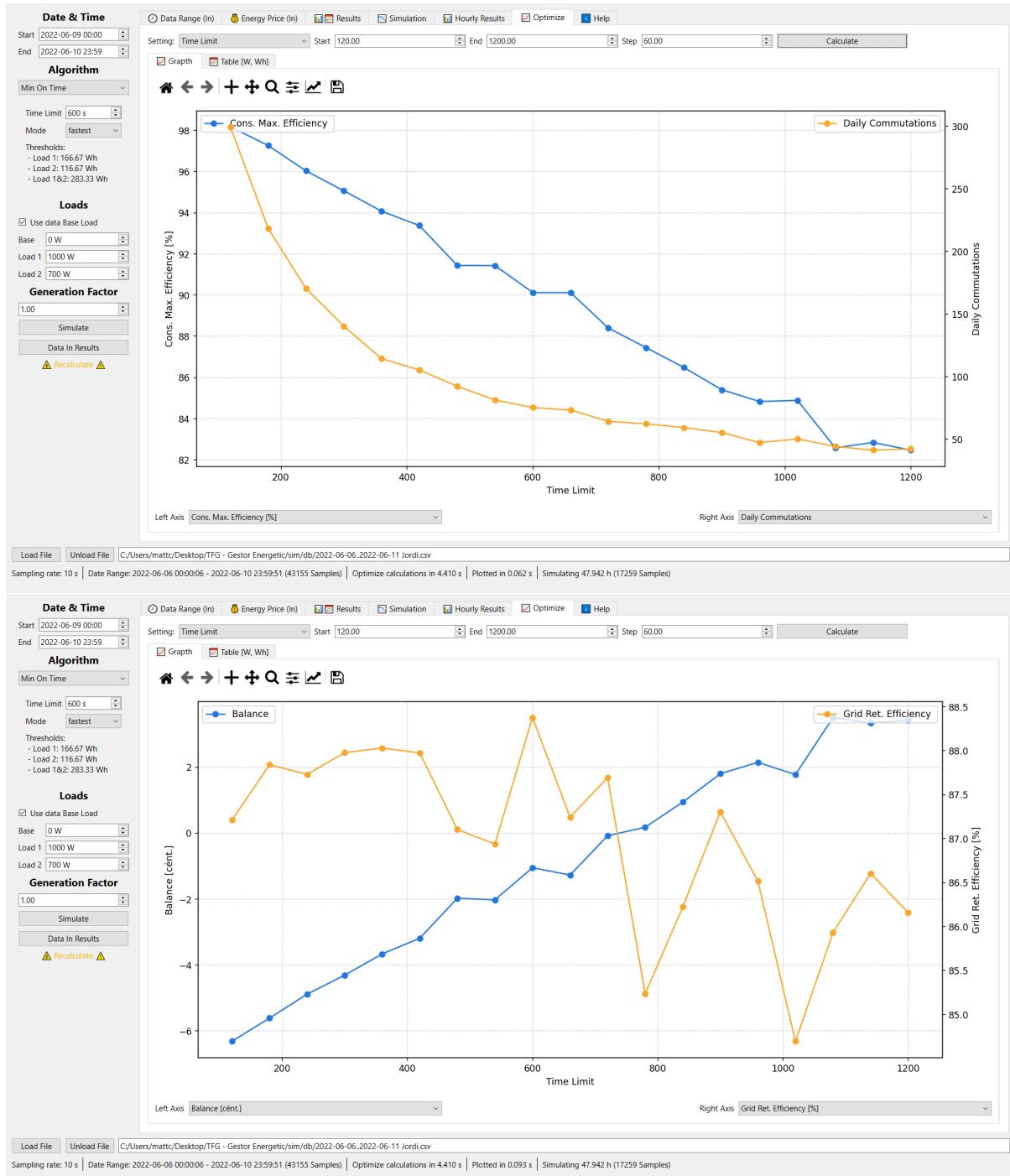


Figura 7.10. Anàlisi dia assolellat – Temps mínim engegada – C1 > C2, versions Completa A i B

7.2.1.3 Predictiu

Com s'ha explicat en l'apartat 4.3, l'algoritme té 3 tres variants en funció de com prediem el balanç final d'hora. Com que les càrregues s'engeguen per ordre, tenim 6 combinacions a simular, tres per quan $C_1 > C_2$ i tres quan $C_2 < C_1$.

A diferència dels apartats anteriors, on hem explicat el procés de buscar els paràmetres en paral·lel als resultats obtinguts, degut a que són moltes imatges, només es mostren els resultats de cada pas per quan $C_1 > C_2$, les imatges es troben a l'annexa B. El procés per buscar els paràmetres és:

1. Optimitzarem l'energia mínima d'engegada, mantenint el llindar final a 0 Wh i el factor del temps a 1.
2. Optimitzar el llindar final, mantenint l'energia mínim d'engegada al valor triat al pas 1 i el factor de temps a 1.
3. Optimitzar el factor de temps, mantenint els valors triats en el pas 1 i 2.

Càrrega 1 < Càrrega 2

No fer predicció:

1. Energia mínima d'engegada a 75 Wh. L'eficiència és del 91,98%, l'eficiència de retorn del 89,87%, les commutacions diàries són 34 i la factura surt a 1,11 cèntims/dia.
2. Llindar final a 0 Wh. Els resultats són els mateixos que el pas 1.
3. Podem fer un factor de temps de 0, així aconseguim una eficiència més gran, del 98,52%, però impactarem negativament en l'eficiència de retorn, 84,95%, les commutacions diàries, 39 i la factura surt a 3,5 cèntims/dia.

El fet de tenir un factor de 0, significa que és comporta com una histèresis: si el balanç és més de 75 Wh la càrrega s'engega, si el balanç és 0 o menys, s'apaga.

Potència disponible mitjana:

1. Energia mínima d'engegada a 75 Wh. L'eficiència és del 97,73%, l'eficiència de retorn del 84,87%, les commutacions diàries són 35 i la factura surt a 3,22 cèntims /dia.
2. Llindar final a 0 Wh. Els resultats són els mateixos que el pas 1.
3. Factor de temps a 1. Els resultats són iguals que el pas 1. Podríem haver fet igual que el cas anterior i fer-lo 0, però l'impacte que té en els altres resultats no és desitjat.

Projecció potència disponible:

1. Energia mínima d'engegada a 75 Wh. L'eficiència és del 99,32%, l'eficiència de retorn del 82,65%, les commutacions diàries són 34 i la factura surt a 3,7 cèntims/dia.
2. Llindar final a 0 Wh. Els resultats són els mateixos que el pas 1.
3. Factor de temps a 1. Els resultats són els mateixos que el pas 1.

Càrrega 1 > Càrrega 2

No fer predicció:

- La configuració és: energia mínima d'engegada a 150 Wh, llindar final a 0 i factor de temps a 1.
- L'eficiència és del 90,59, l'eficiència de retorn del 84,62%, les commutacions diàries són 30 i la factura surt a 1,1 cèntims/dia.

Potència disponible mitjana:

- La configuració és: energia mínima d'engegada a 600 Wh, llindar final a 0 i factor de temps a 1.
- L'eficiència és del 85,76%, l'eficiència de retorn del 76,31%, les commutacions diàries són 32 i la factura surt a 0,25 cèntims/dia.

Projecció potència disponible:

- La configuració és: energia mínima d'engegada a 110 Wh, llindar final a 0 i factor de temps a 1.
- L'eficiència és del 98,25%, l'eficiència de retorn del 80,03%, les commutacions diàries són 26 i la factura surt a 3,8 cèntims/dia.

7.2.1.4 Conclusions

El resum dels resultats de cada configuració optimitzada són els següents:

	Algoritme	Eficiència de consum [%]	Commutacions	Eficiència de retorn [%]	Cost Factura [cèntims/dia]
Temps mínim engegada	<i>Histèresis</i>	93,93	27	85,92	1,8
	<i>Versió Simple i A</i>	85,54	50	85,29	0
	<i>Versió B</i>	84,88	50	84,69	0
	<i>Versió Simple</i>	86,07	50	84,11	0
	<i>Versió A i B</i>	84,88	50	84,69	0
	<i>No predir</i>	98,52	39	84,95	3,5
Predictiu	<i>Potència disponible mitjana</i>	97,73	35	84,87	3,2
	<i>Projecció potència disponible</i>	99,32	34	82,65	3,7
	<i>No predir</i>	90,59	30	84,62	1,1
	<i>Potència disponible mitjana</i>	85,76	32	76,31	0,25
	<i>Projecció potència disponible</i>	98,25	26	80,03	3,8

Taula 7.1. Anàlisi dia assolellat – Resultats del sistema en funció de l'algoritme

	[Wh]			
	Llindar alt 1	Llindar baix 1	Llindar alt 2	Llindar baix 2
<i>Histèresis</i>	70	-10	200	0

Figura 7.11. Anàlisi dia assolellat – Configuració òptima - Histèresis

	Situació	Versió	Temps mínim [s]
Temps mínim engegada	$C1 < C2$	<i>Simple i A</i>	600
		<i>B</i>	1020
	$C1 > C2$	<i>Simple</i>	600
		<i>A i B</i>	600

Taula 7.2. Anàlisi dia assolellat – Configuració òptima - Temps mínim engegada

	Situació	Variant	Mín. engegar [Wh]	Llindar final [Wh]	Factor de temps
$C1 < C2$		<i>No predir</i>	75	0	0
		<i>Potència disponible mitjana</i>	75	0	1
		<i>Projecció potència disponible</i>	75	0	1
Predictiu		<i>No predir</i>	50	0	1
		<i>Potència disponible mitjana</i>	600	0	1
		<i>Projecció potència disponible</i>	110	0	1

Taula 7.3. Anàlisi dia assolellat – Configuració òptima - Predictiu

D'aquests resultats podem conoure que tots els algoritmes es comporten excel·lentment si sempre hi ha sol i s'optimitza la configuració d'aquests al màxim:

- Tots els algoritmes optimitzen bé l'ús d'energia del sistema, l'eficiència més baixa és del 84,87%.
- El deute que es genera al consumir de la xarxa és retorna quasi tot, l'eficiència de retorn a la xarxa més baix és del 76,31% i la segona més baixa del 82,62%. Això implica un impacte ambiental baix.
- Les commutacions són acceptables, la majoria ronden entre les 30 i 50. A canvi d'eficiència podríem disminuir el nombre de commutacions, en la majoria dels casos.
- El cost diari de la factura és nul o quasi nul.

Si comparem els algoritmes entre ells i els ordenem de millor a pitjor obtenim que:

1. L'algoritme predictiu és el millor de tots. Té les eficiències més altes, aconsegueix mantenir les commutacions acceptables, pot retornar gran part de l'energia consumida de la xarxa i té un cost diari baix, tot i ser el més alt de tots els algoritmes.
 - a. La millor versió de l'algoritme és quan la càrrega 1 és més gran que la 2 i el balanç net horari es prediu projectant la potència disponible. La raó és que té l'eficiència més alta de totes les versions i les commutacions més baixes.
2. La histèresis és el segon millor, per ser un sistema molt simple, té molt bons resultat. En general les commutacions, l'eficiència de la xarxa i el cost diari de la factura són millors que l'algoritme anterior. L'únic que el fa estar segon és l'eficiència.
3. L'algoritme de Temps mínim engegada és el pitjor, tot i ser molt bo. La raó és que totes les variants de l'algoritme tenen les eficiències més baixes i les commutacions més altes. Tot i així, és l'únic que aconsegueix fer arribar el cost diari de la factura a 0.

Cal recordar que els resultats són respecte un dia assolellat. Que un algoritme sigui millor aquí, no vol dir que ho sigui en general.

7.2.2 Dia de núvols/pluja

Les dades que s'han fet servir per simular el dia de núvols/pluja són les del 03/05/2022 i 04/05/2022.

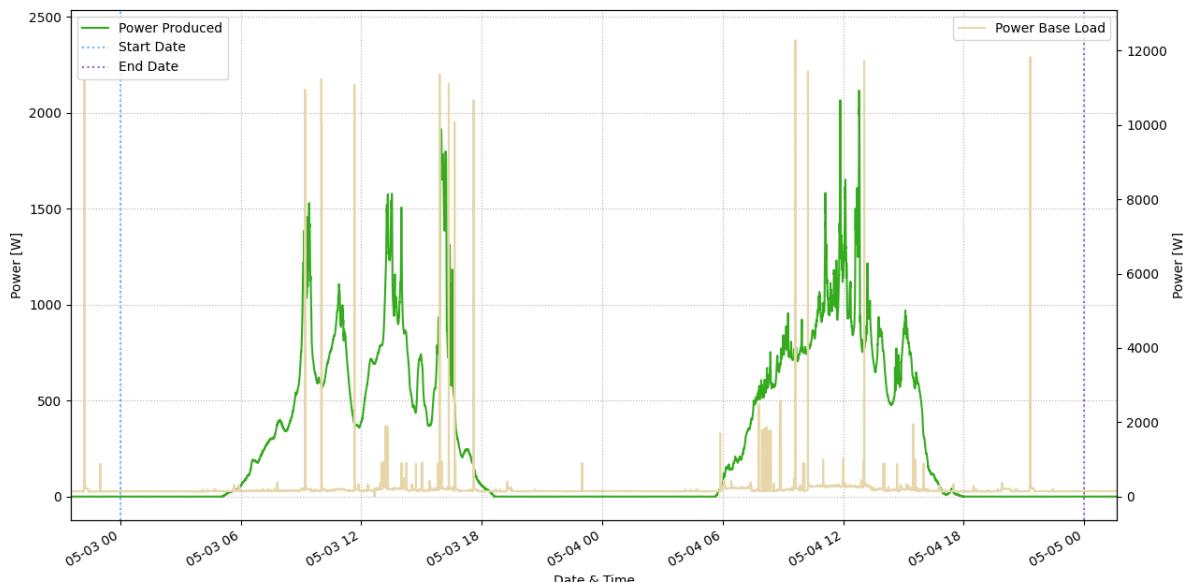


Figura 7.12. Anàlisi dia de núvols/pluja – Dades per fer la simular

Com que en l'apartat anterior ja hem vist com optimitzar els valors, en aquest apartat donarem directament els resultats i els analitzarem:

	Algoritme	Eficiència de consum [%]	Commutacions	Eficiència de retorn [%]	Cost Factura [cèntims/dia]
	<i>Histèresis</i>	94,96	27	80,48	5,26
<i>Temps mínim engegada</i>	<i>Versió Simple i A</i>	85,81	37	79,93	3,91
	<i>Versió B</i>	88,11	36	81,67	4,25
	<i>Versió Simple</i>	83,16	30	83,80	3,56
	<i>Versió A i B</i>	88,11	36	81,67	4,25
<i>Predictiu</i>	<i>No predir</i>	96,45	36	85,01	5,61
	<i>Potència disponible mitjana</i>	97,33	38	81,42	5,77
	<i>Projecció potència disponible</i>	97,73	32	75,45	5,95
	<i>No predir</i>	95	34	85,136	5,62
<i>C1 > C2</i>	<i>Potència disponible mitjana</i>	85,42	37	78,59	4,35
	<i>Projecció potència disponible</i>	98,67	26	78,66	6,25

Taula 7.4. Anàlisi dia de núvols/pluja – Resultats del sistema en funció de l'algoritme

	[Wh]			
	Llindar alt 1	Llindar baix 1	Llindar alt 2	Llindar baix 2
<i>Histèresis</i>	70	-10	200	0

Figura 7.13. Anàlisi dia assolellat – Configuració òptima - Histèresis

	Situació	Versió	Temps mínim [s]
<i>Temps mínim engegada</i>	<i>C1 < C2</i>	<i>Simple i A</i>	600
		<i>B</i>	680
	<i>C1 > C2</i>	<i>Simple</i>	600
		<i>A i B</i>	680

Taula 7.5. Anàlisi dia assolellat – Configuració òptima - Temps mínim engegada

	Situació	Variant	Mín. engegar [Wh]	Llindar final [Wh]	Factor de temps
<i>Predictiu</i>	<i>C1 < C2</i>	<i>No predir</i>	75	0	1
		<i>Potència disponible mitjana</i>	75	0	1
	<i>C1 > C2</i>	<i>Projecció potència disponible</i>	75	0	1
		<i>No predir</i>	125	0	1
<i>C1 > C2</i>	<i>Potència disponible mitjana</i>	380	0	1	
	<i>Projecció potència disponible</i>	40	0	1	

Taula 7.6. Anàlisi dia assolellat – Configuració òptima - Predictiu

7.2.3 Conclusions

Com podem veure, les configuracions dels algoritmes i els resultats de les simulacions per un dia assolellat i un dia de núvols/pluja són quasi iguals. Això significa que les configuracions per els dies assolellats també són les més òptimes per dies de poca producció, i el “rànking” d’algoritmes també és el mateix, recordem-lo:

1. L’algoritme predictiu és el millor de tots, on la millor versió és quan la càrrega 1 és més gran que la 2 i el balanç net horari es prediu projectant la potència disponible.
2. La histèresis és el segon millor. Té molt bons resultats tot i ser l’algoritme més simple.
3. L’algoritme de Temps mínim engegada és el pitjor, tot i ser molt bo. La raó és que totes les variants de l’algoritme tenen les eficiències més baixes i les commutacions més altes. Tot i així, és l’únic que aconsegueix fer arribar el cost diari de la factura a 0.

En conclusió, l’algoritme que hauríem de fer servir per aquest sistema real és el predictiu, predir el balanç a partir de la projecció de la potència disponible i col·locant la càrrega més gran a la primera posició. Els llindar mínim d’engegada seria entre 40 i 110 Wh, el factor de temps seria 1 i el llindar final seria de 0Wh.

Amb aquesta configuració aconseguiríem una eficiència de consum al voltant 98%, unes 26 commutacions diàries entre les dues càrregues, una eficiència de retorn a la xarxa del 80% i un cost diari de la factura d’entre 4 i 6 cèntims.

El resultat és que, amb aquest algoritme, el sistema pot cobrir les necessitats de l’habitatge i controlar dues bombes de calor (climatització) perquè estiguin el màxim de temps engegades, tot reduint la factura mensual a un cost d’entre 1,2€ i 1,8€.

8. Conclusions

En aquesta tesi, hem explicat al lector què ha motivat la creació del projecte i quins objectius ens hem marcat. Després, hem contextualitzat el projecte dins la legalitat d'Espanya i hem definit com modelaríem el sistema d'autoconsum a gestionar. També hem definit els diversos termes que s'han fet servir al llarg del projecte per tal de parlar el mateix llenguatge amb el lector.

Posteriorment, hem explicat en detall els tres algoritmes que s'ha dissenyat per controlar el sistema. Un cop definits, s'ha explicat tant la implementació del simulador com la del sistema real, sense entrar en com s'ha programat.

Finalment, s'ha mostrat com optimitzar la configuració de cada algoritme per al sistema real implementat.

Al ser el primer treball (o un dels primers) a tractar aquest tema, encara queda molt camí per assolir el sistema ideal plantejat. Alguns dels següents passos que es podrien fer són:

- Comprovar el funcionament del sistema en altre estacions, ja que la implementació real s'ha fet durant la primavera i estiu, que és quan les condicions són les més ideals.
- Fer un model de sistema més complex. Per exemple, es podria fer que la calefacció només s'engegués quan fos necessari, fins i tot que simulés/predis la temperatura per ajustar-se al màxim. Això permetria afegir altres càrregues a controlar sense afectar la climatització del habitatge.
- Afegir prediccions meteorològiques com a entrades del sistema.
- Fer que el sistema sigui capaç de predir el consum base. Això es podria fer a partir de:
 - Calcular la mitjana del consum diari, setmanal o anual a partir dels últims X dies.
 - A través d'un IA (Intel·ligència Artificial).
- ...

Tot i ser el primer treball sobre el tema, els fruits d'aquest ja són aplicables al món real i s'han fet accessibles a tot el món perquè la gent se'n beneficiï. D'aquesta forma, el simulador i sistema de control també poden ser millorats per els usuaris.

En un to més personal, estic molt satisfet amb el treball, ja que s'han assolit tots els objectius marcats, els quals podien semblar ambiciosos a l'inici del projecte.

En conclusió, el treball ha assolit tots els objectius marcats. quest treball és el primer pas cap a un sistema de control de càrregues intel·ligent eficient i respectuós amb el medi. Tot i quedar molt camí, ja s'ha pogut crear una primera implementació al món real, la qual tothom utilitzar i millorar.

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] «mattcarter11/TFG---Gestor-Energetic,» Github, [En línia]. Available: <https://github.com/mattcarter11/TFG---Gestor-Energetic>. [Últim accés: 03 07 2022].
- [4] «Hysteresis,» Wikipedia, 16 04 2022. [En línia]. Available: <https://en.wikipedia.org/wiki/Hysteresis>. [Últim accés: 28 06 2022].
- [5] 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].
- [6] N. Developer, «NumPy documentation,» NumPy Developer, [En línia]. Available: <https://numpy.org/doc/stable/>. [Últim accés: 30 06 2022].
- [7] 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].
- [8] «Qt for Python,» 2021. [En línia]. Available: <https://doc.qt.io/qtforpython/>. [Últim accés: 19 12 2021].
- [9] 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].
- [10] 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].
- [11] 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].
- [12] 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].
- [13] 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].
- [14] 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-10a2f3a2a2>.

- most-efficient-way-to-iterate-through-your-pandas-dataframe-4dad88ac92ee. [Últim accés: 1 07 2022].
- [15] «Shelly1L - API Reference,» Shelly, [En línia]. Available: <https://shelly-api-docs.shelly.cloud/gen1/#shelly1l>. [Últim accés: 03 07 2022].
- [16] «InfluxDB: Open Source Time Series Database,» Influxdata, [En línia]. Available: <https://www.influxdata.com/>. [Últim accés: 03 07 2022].
- [17] «InfluxDB v2 API,» InfluxDB, [En línia]. Available: <https://docs.influxdata.com/influxdb/v2.3/reference/api/>. [Últim accés: 03 07 2022].
- [18] «Grafana,» Grafana Labs, [En línia]. Available: <https://grafana.com/>. [Últim accés: 03 07 2022].
- [19] «Python client library,» Influxdata, [En línia]. Available: <https://docs.influxdata.com/influxdb/cloud/api-guide/client-libraries/python/>. [Últim accés: 03 07 2022].
- [20] «ShellyPy,» PyPi, 12 03 2022. [En línia]. Available: <https://pypi.org/project/ShellyPy/>. [Últim accés: 03 07 2022].
- [21] 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-renewable-domestica>. [Últim accés: 23 6 2022].
- [22] 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].

II. Annex

A. Realitat vs Simulació

Per indicar la configuració del sistema real en les figures corresponents al comportament real, es fa els camps configurables de la simulació, tot i no afectar en cap aspecte.

Per poder fer la simulació correctament, la potència de les càrregues s'han obtingut de l'aproximació de potència donada pels resultats reals.

A.1. Histèresis

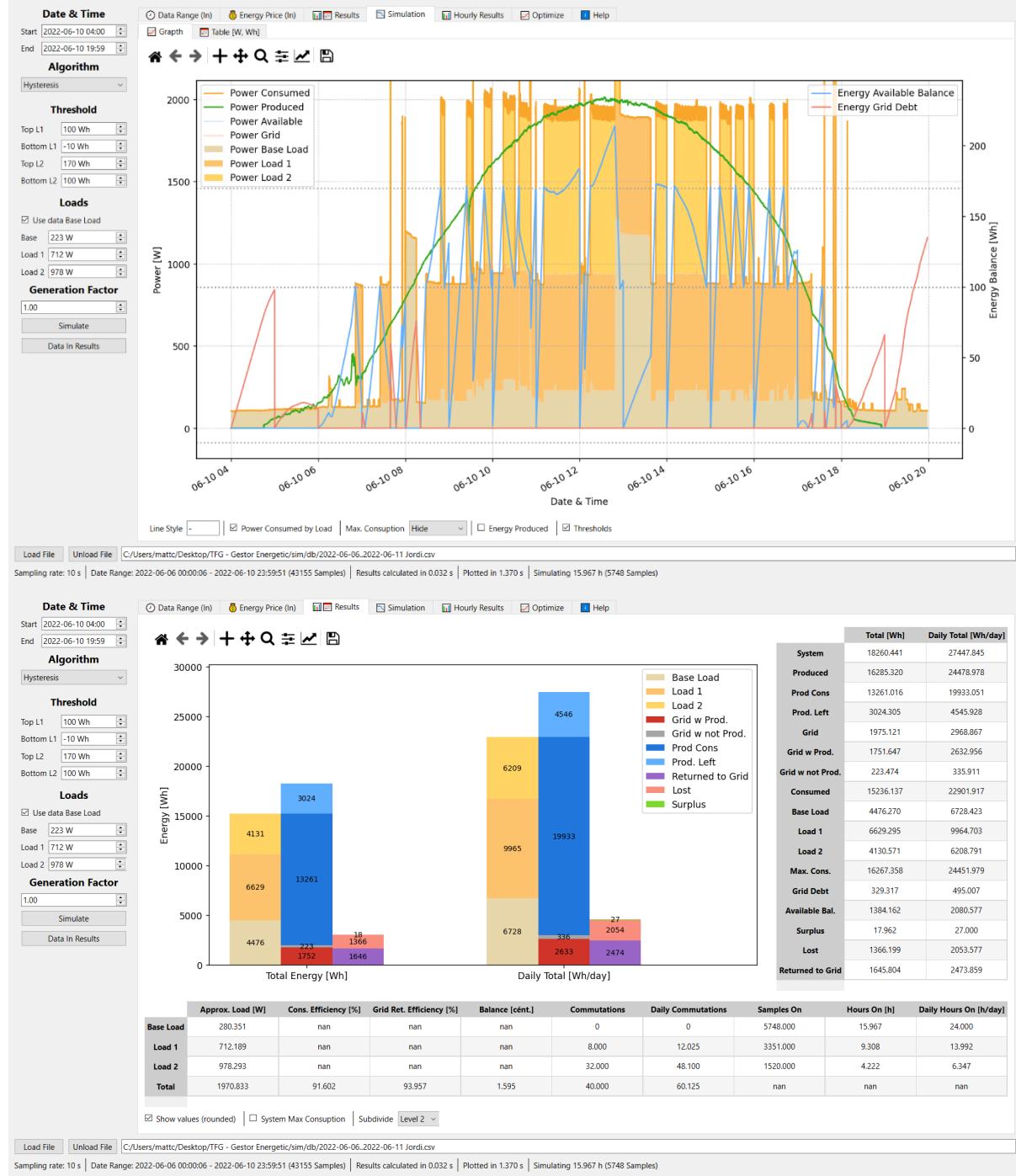


Figura A.1. Histèresis – Comportament i resultats del sistema real

A. Realitat vs Simulació



Figura A.2. Histèresis – Comportament i resultats simulats (sense dades reals del consum base)

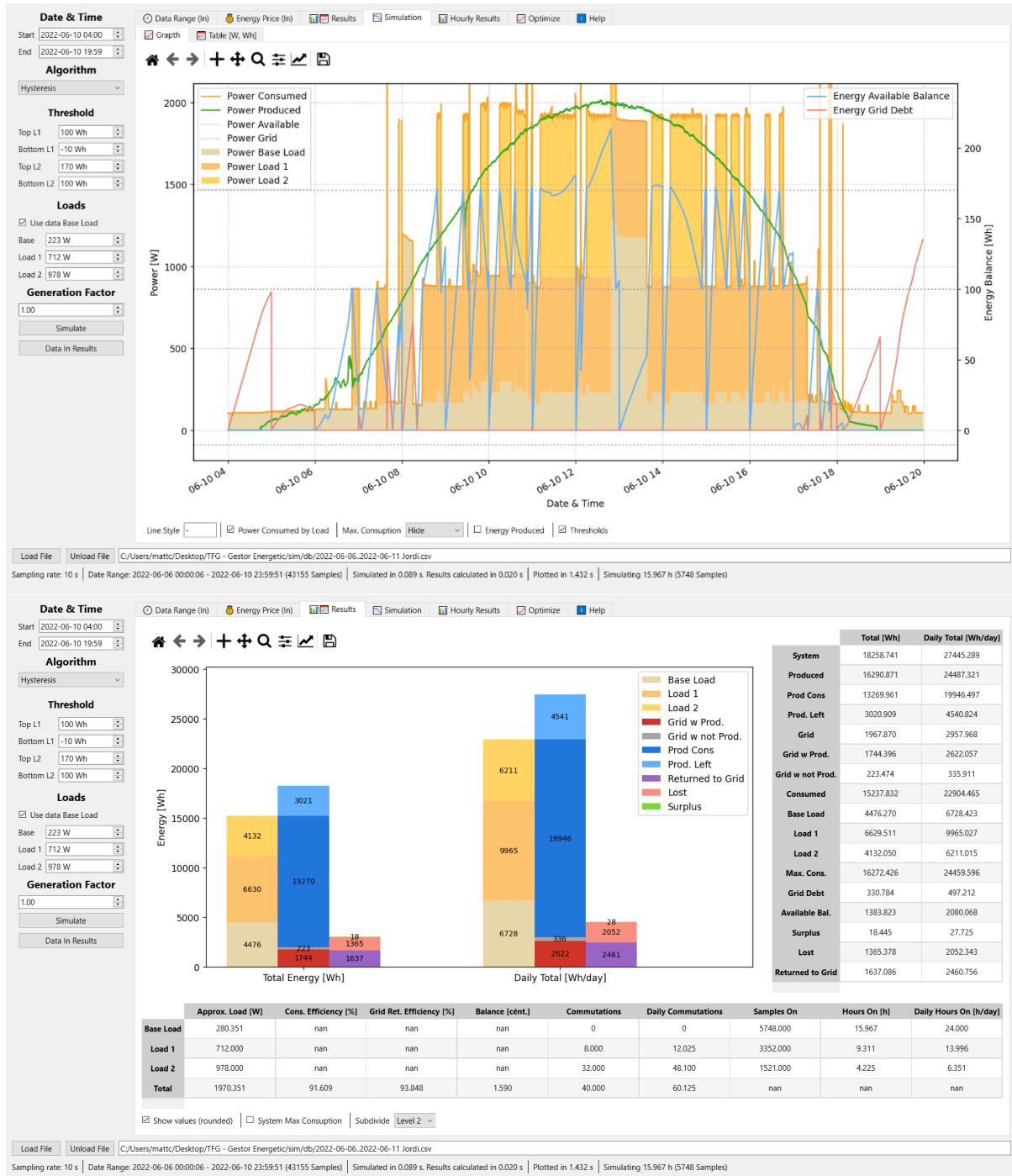


Figura A.3. Histèresis – Comportament i resultats simulats (amb dades reals del consum base)

A.2. Tems mínim engegada

Tot i haver programant l'algoritme en el sistema real, no s'ha pogut verificar el seu funcionament, ja que, veient la seva eficiència en la simulació i que les proves no van ser exitoses, es va decidir provar el predictiu, que era el que faltava. És l'únic cas que no podrem assegurar al 100% que la simulació és precisa.

A.3. Predictiu

A l'implementar l'algoritme predictiu en el sistema real es va fer un error en programar una de les variables, això feia que el comportament fos similar al predictiu. Aquest fet, juntament amb el d'haver-ho implementat l'últim, ha fet que només puguem tenir dades bones del comportaments de mig dia, ja que fins que no hem comparat el comportament real amb el simulador no ens n'hem adonat del error.

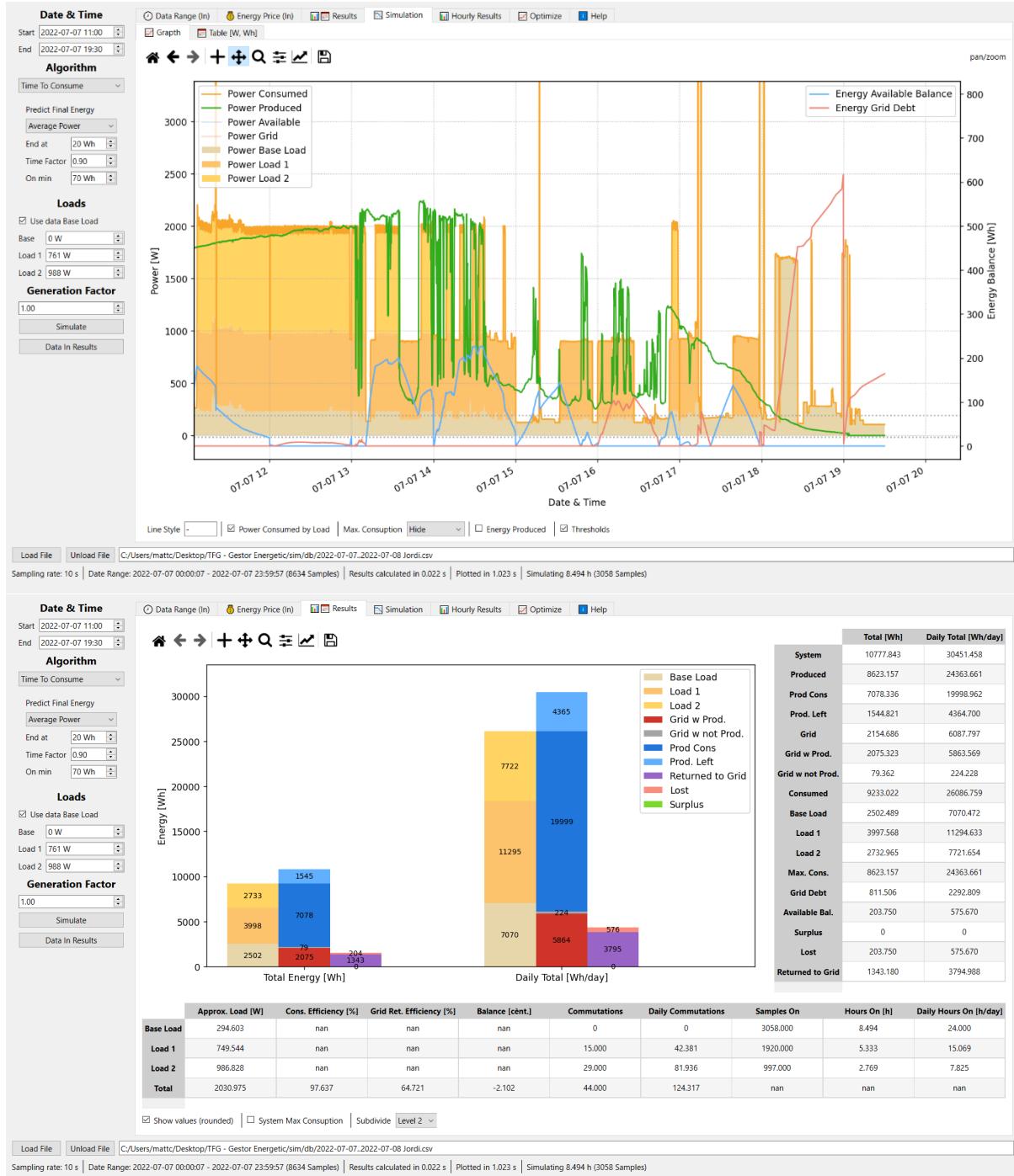


Figura A.4. Predictiu – Comportament i resultats del sistema real

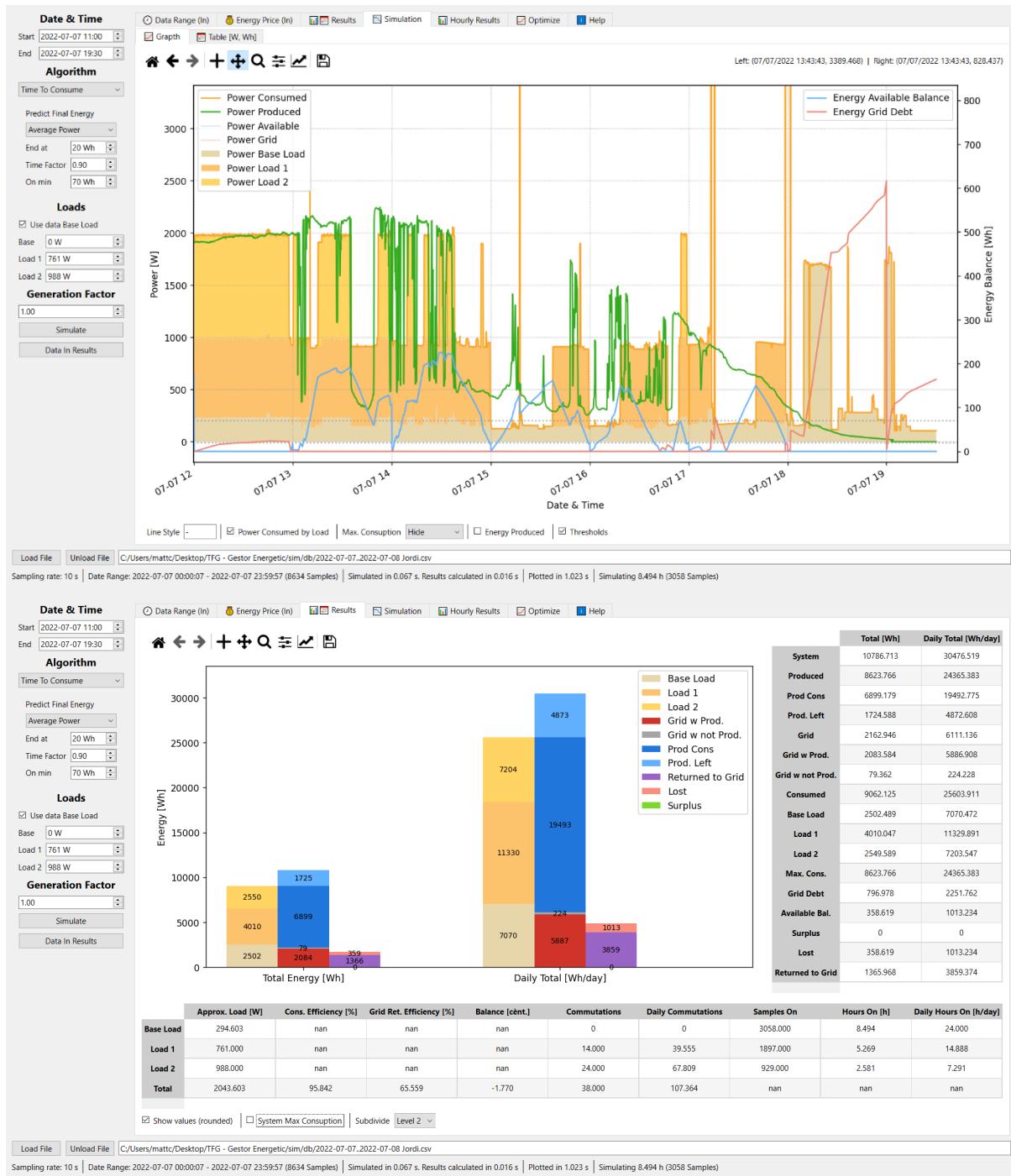


Figura A.5. Predictiu – Comportament i resultats simulats (sense dades reals del consum base)

B. Dia assolellat

B.1. Predictiu – càrrega 1 > càrrega 2

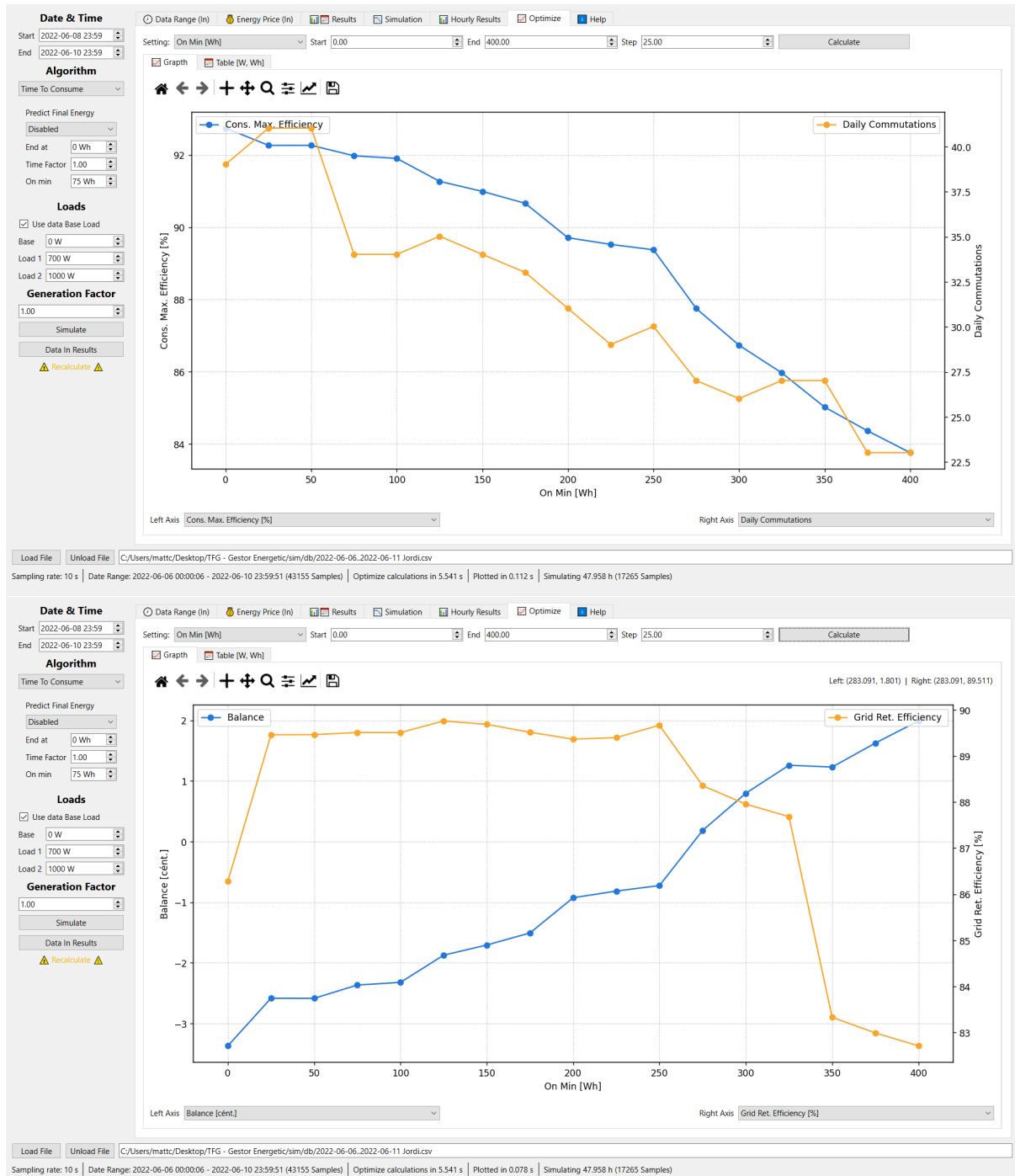


Figura B.6. Anàlisi dia assolellat – Predictiu– C1 < C2, sense prediccó, pas 1



Figura B.7. Anàlisi dia assolellat – Predictiu – C1 < C2, sense prediccó, pas 2

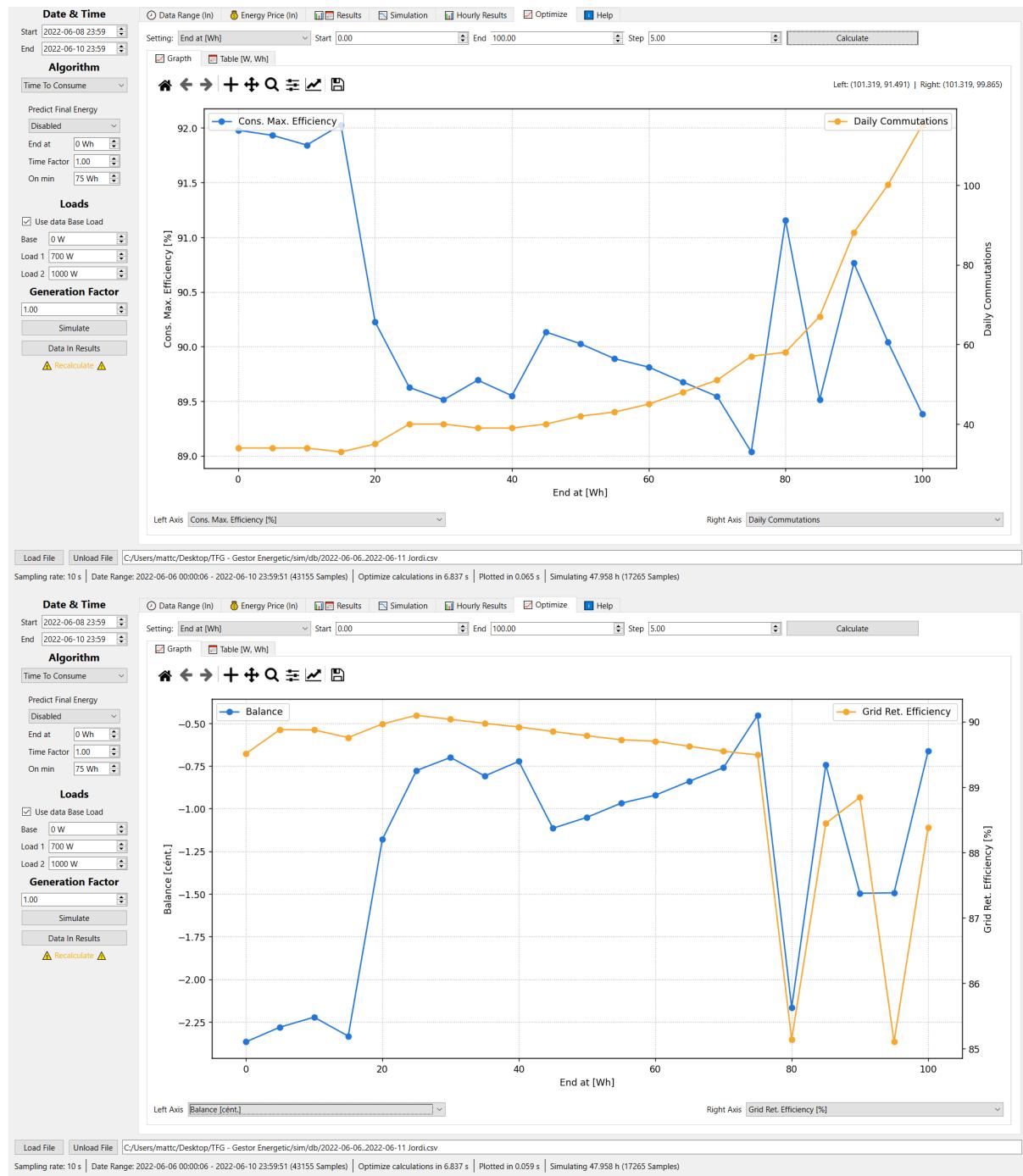


Figura B.8. Anàlisi dia assolellat – Predictiu – C1 < C2, sense prediccó, pas 3

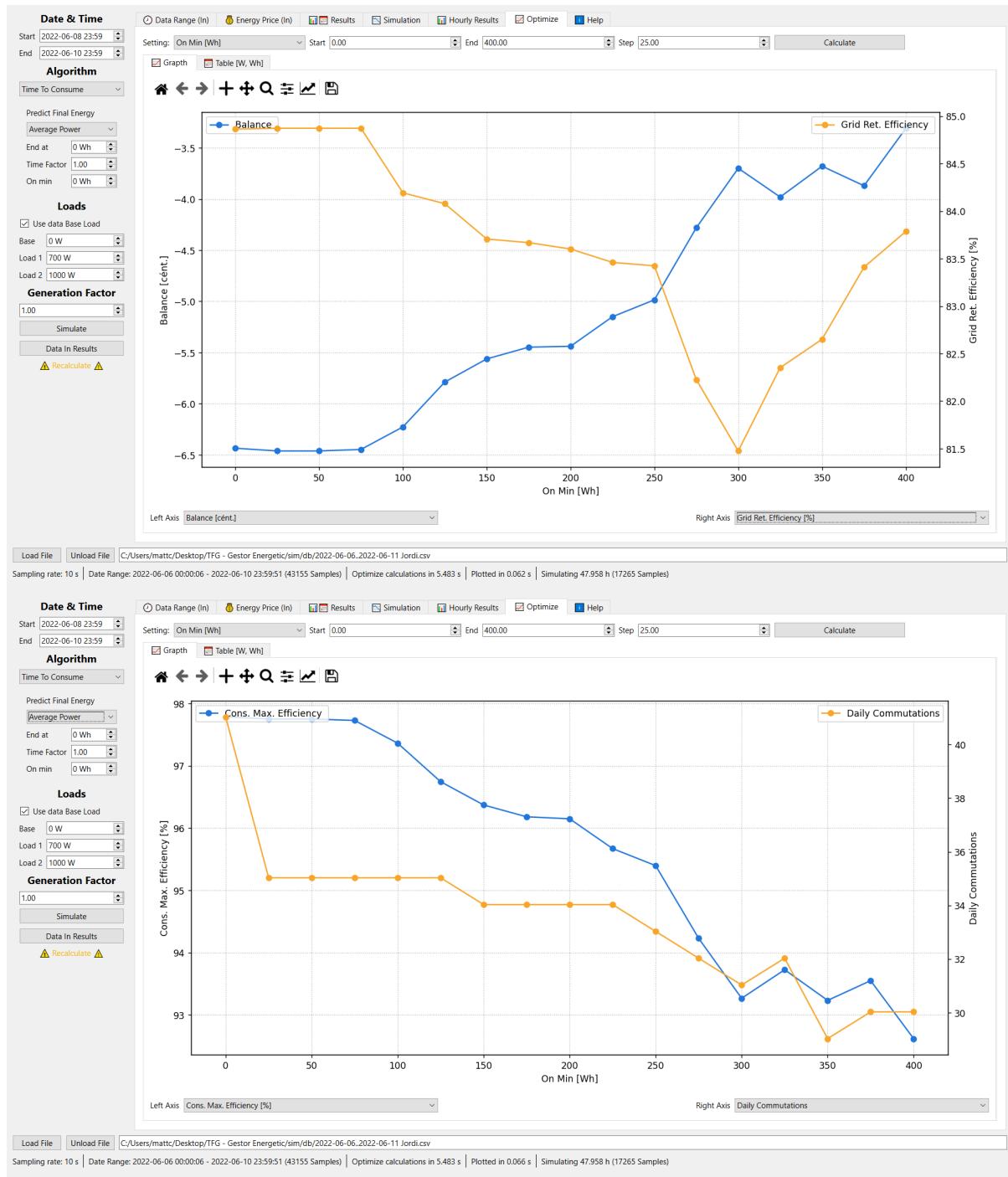


Figura B.9. Anàlisi dia assolellat – Predictiu – C1 < C2, potència mitjana disponible, pas 1

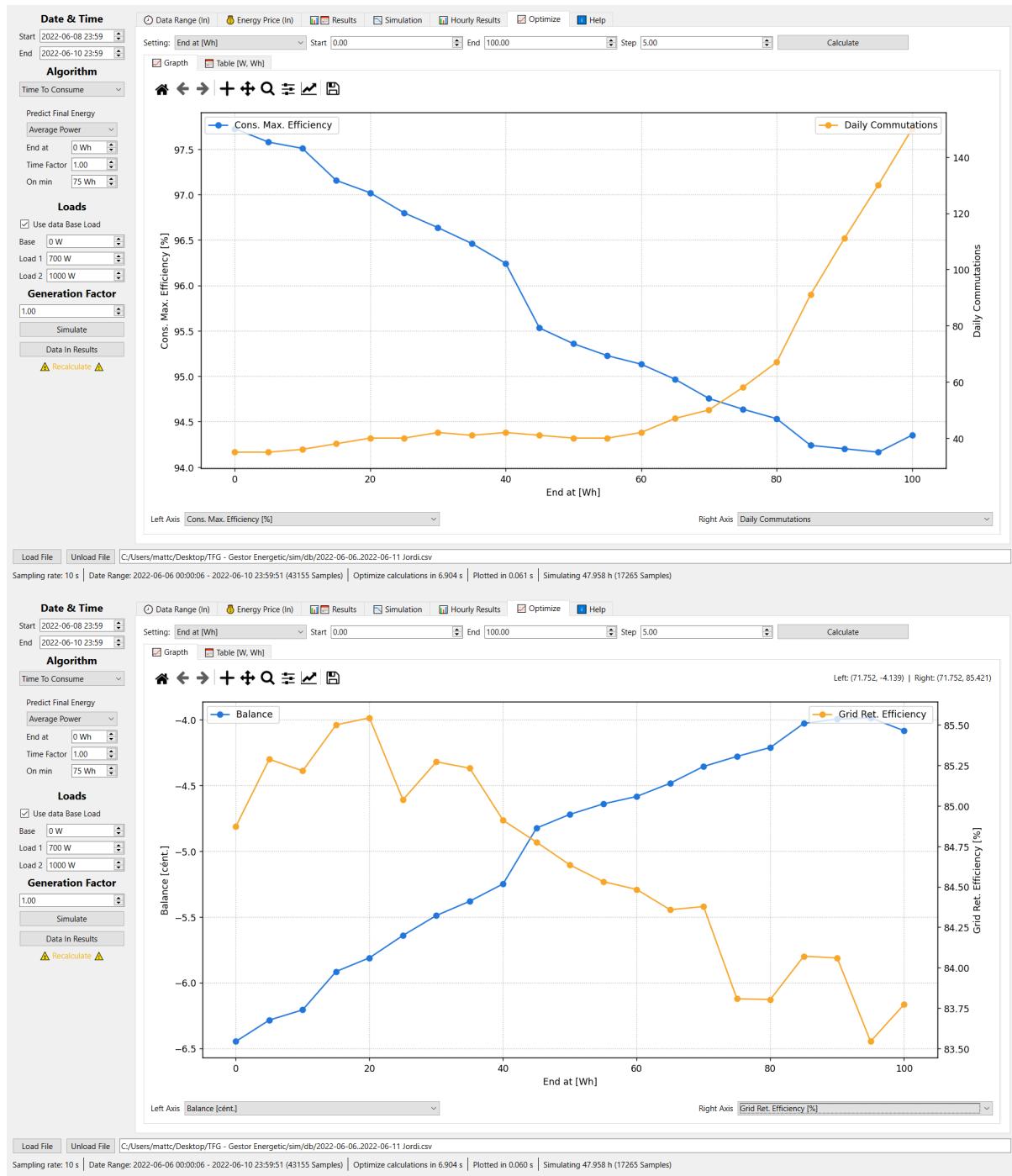


Figura B.10. Anàlisi dia assolellat – Predictiu – C1 < C2, potència mitjana disponible, pas 2

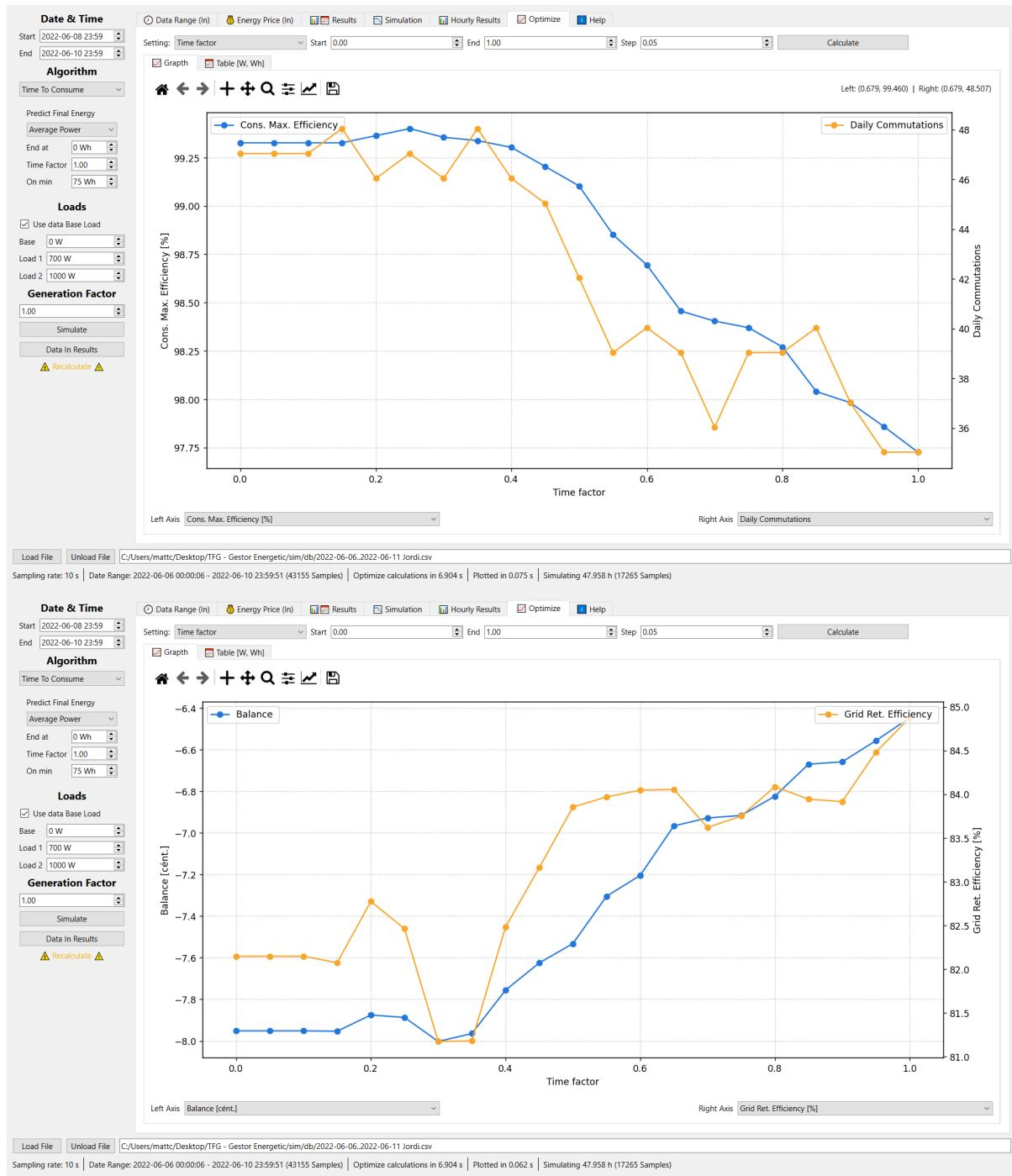


Figura B.11. Anàlisi dia assolellat – Predictiu – C1 < C2, potència mitjana disponible, pas 3

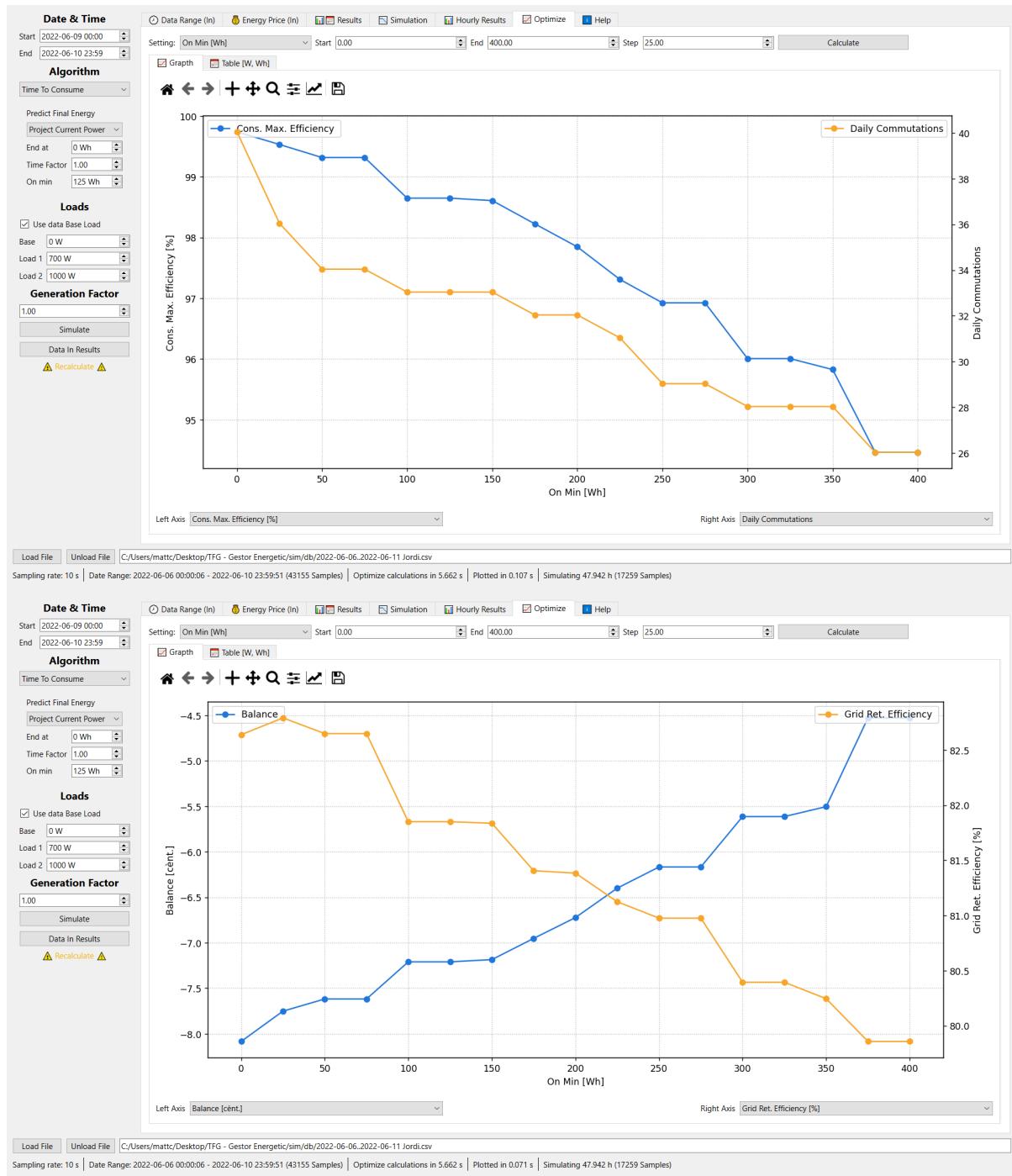


Figura B.12. Anàlisi dia assolellat – Predictiu – C1 < C2, projecció potència disponible, pas 1

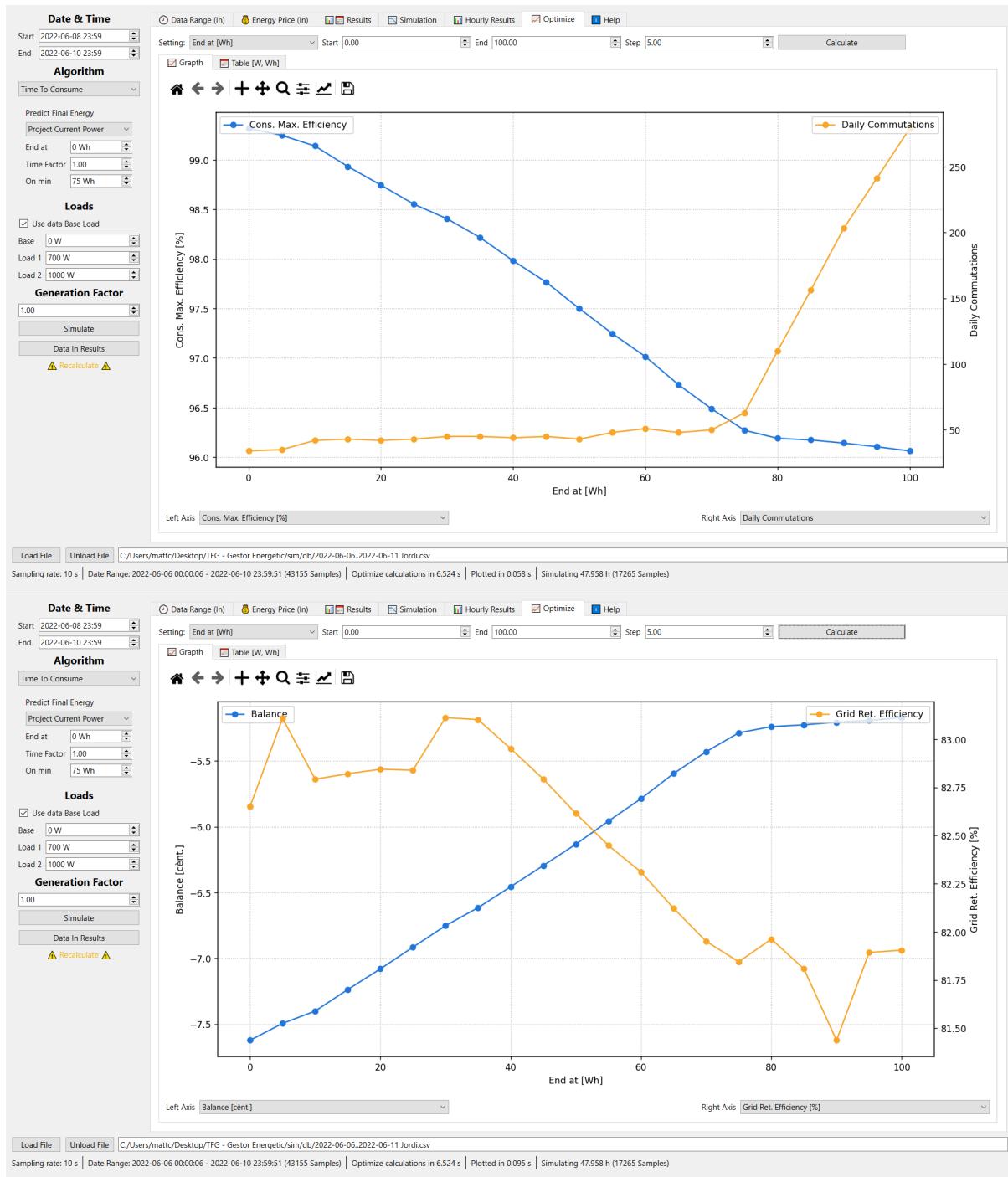


Figura B.13. Anàlisi dia assolellat – Predictiu – C1 < C2, projecció potència disponible, pas 2



Figura B.14. Anàlisi dia assolellat – Predictiu – C1 < C2, projecció potència disponible, pas 3