

Trabajo Fin de Grado

Grado en Ingeniería Aeroespacial

Control electrónico basado en algoritmos de optimización multiobjetivo para la microrred de una aeronave

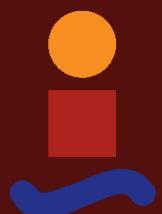
Autor: Mario Vázquez Acosta

Tutores: Eduardo Galván Díez

Carlos García Santacruz

Dpto. Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2022



Trabajo Fin de Grado

Grado en Ingeniería Aeroespacial

Control electrónico basado en algoritmos de optimización multiobjetivo para la microrred de una aeronave

Autor:

Mario Vázquez Acosta

Tutores:

Eduardo Galván Díez

Catedrático de Universidad

Carlos García Santacruz

Personal Técnico de Apoyo

Dpto. Ingeniería Electrónica

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2022

Trabajo Fin de Grado: Control electrónico basado en algoritmos de optimización multi-objetivo para la microrred de una aeronave

Autor: Mario Vázquez Acosta
Tutores: Eduardo Galván Díez
Carlos García Santacruz

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

Con este trabajo se dan por concluidos mis estudios en el Grado en Ingeniería Aeroespacial de la Universidad de Sevilla, no sin antes agradecer a todas las personas que me han acompañado durante estos años y la confianza depositada en mí.

Agradecer a mi familia por no dudar de mis decisiones y apoyarme en todo momento, en especial a mi madre, por hacerlo todo siempre más fácil.

Dar gracias a mis amigos, por tanto momentos buenos y malos, y por haber hecho de la carrera, un camino.

Por último, reconocer el trabajo de mis tutores, Eduardo y Carlos, por haberme ayudado y darme la oportunidad de descubrir un campo nuevo gracias a este trabajo.

Resumen

El objetivo principal de este trabajo es el modelado de un programa de gestión para optimizar los flujos de potencia entre los diferentes generadores, sistemas de almacenamiento y cargas de una aeronave comercial *More Electric Aircraft* (MEA) para conseguir mayor autonomía, fiabilidad y robustez.

Para ello, se ha realizado una adaptación de un trabajo previo descrito en el artículo [1], en el cual se plantea una estrategia de gestión centralizada a nivel de sistema basada en Control Predictivo Basado en Modelo (MPC) para que un MEA sea capaz de programar los sistemas de baterías y explote la flexibilidad en la demanda de las cargas mientras satisface los requisitos operativos variantes en el tiempo.

La estrategia de control propuesta tiene como objetivo mantener el almacenamiento de energía en un rango aceptable para evitar la descarga y prolongar el ciclo de vida de la batería, al tiempo que minimiza las desconexiones de cargas, tratando de evitar constantes cambios de estado para incrementar la vida útil de los dispositivos. Para abordar la resolución del problema, éste se modela mediante un problema de programación lineal con variable binaria (MILP), proponiéndose diferentes funciones objetivo para optimizar el control, mediante la definición de diversos requisitos operacionales, así como la definición de restricciones que modelan los comportamientos del sistema.

Además, para asentar las bases y comprender en qué consisten estas nuevas generaciones de aeronaves MEA y los sistemas eléctricos de potencia, se explicará en qué consiste este concepto y se ofrecerá una revisión general de los avances tecnológicos empleados para lograr diversos objetivos como reducir las emisiones de gases de efecto invernadero y disminuir el consumo de combustibles fósiles.

Abstract

The main objective of this work is the modeling of a management system to optimize the power flows between the different generators, storage systems and loads of a commercial aircraft MEA to achieve greater autonomy, reliability and robustness.

To this end, an adaptation of a previous work described in the paper [1], in which a centralized system-level management strategy based on model predictive control (MPC) is proposed for an MEA to be able to schedule battery systems and exploit flexibility in load demand while satisfying time-varying operational requirements.

The proposed control strategy aims to maintain energy storage and prolong the battery life cycle, while minimizing load disconnections, trying to avoid constant state changes to increase the lifetime of the devices. Using a mixed-integer linear programming (MILP) formulation, different objective functions are proposed to realize the control objectives, with soft constraints that improve the feasibility of the model.

In addition, to lay the groundwork and understand what these new generations of More Electric Aircraft and electric power systems consist of, the concept will be explained and a general review of the technological advances used to achieve various objectives such as reducing greenhouse gas emissions and reducing fossil fuel consumption will be provided.

Índice Abreviado

<i>Resumen</i>	III
<i>Abstract</i>	V
<i>Índice Abreviado</i>	VII
<i>Notación</i>	XI
1 Introducción	1
1.1 Reseña histórica	3
1.2 El concepto More Electric Aircraft	4
1.3 Desafíos del sistema eléctrico de potencia del avión	6
2 Sistema de potencia eléctrica	9
2.1 Sistemas de potencia a bordo	10
2.2 Arquitecturas de EPS	16
3 Técnicas de control y optimización de microrredes	21
3.1 Autómata finito	21
3.2 Algoritmos heurísticos y metaheurísticos	23
3.3 Redes neuronales	24
3.4 Control Predictivo Basado en Modelo	26
4 Modelado de la estrategia de optimización	33
4.1 EPS del modelo	33
4.2 Descripción de MILP-MPC	34
4.3 Modelado del sistema	37

4.4	Implementación del modelo	41
4.5	Implementación en Simulink	43
5	Resultados	47
5.1	Parámetros del modelo	47
5.2	Índices de evaluación	48
5.3	Simulación del sistema	54
5.4	Casos particulares	56
5.5	Simulación en Simulink	59
6	Conclusiones	65
Apéndice A	Códigos de Python	67
<i>Índice de Figuras</i>		89
<i>Índice de Tablas</i>		91
<i>Índice de Códigos</i>		93
<i>Bibliografía</i>		95

Índice

<i>Resumen</i>	III
<i>Abstract</i>	V
<i>Índice Abreviado</i>	VII
<i>Notación</i>	XI
1 Introducción	1
1.1 Reseña histórica	3
1.2 El concepto More Electric Aircraft	4
1.3 Desafíos del sistema eléctrico de potencia del avión	6
2 Sistema de potencia eléctrica	9
2.1 Sistemas de potencia a bordo	10
2.1.1 Generación de potencia en MEA EPS	10
2.1.1.1 Generadores principales	10
2.1.1.2 Unidad de Potencia Auxiliar	12
2.1.1.3 Ram Air Turbine	12
2.1.2 Consumo de energía a bordo del MEA	13
2.1.2.1 Sistema de protección contra el hielo	14
2.1.2.2 Sistema de control ambiental	14
2.1.2.3 Actuadores	15
2.2 Arquitecturas de EPS	16
2.2.1 Arquitectura de EPS de aeronaves convencionales	17
2.2.2 Arquitecturas principales de MEA EPS	17
2.2.2.1 Arquitectura híbrida CA/CC	17
2.2.2.2 Arquitectura de bus único	19
2.2.3 Diferencia distribución flexible/centralizada	20
3 Técnicas de control y optimización de microrredes	21
3.1 Autómata finito	21
3.2 Algoritmos heurísticos y metaheurísticos	23
3.3 Redes neuronales	24
3.3.1 Combinación de redes neuronales con algoritmos heurísticos	26
3.4 Control Predictivo Basado en Modelo	26
3.4.1 Principios del MPC	26
3.4.2 Aplicaciones del MPC	29

4 Modelado de la estrategia de optimización	33
4.1 EPS del modelo	33
4.2 Descripción de MILP-MPC	34
4.3 Modelado del sistema	37
4.3.1 Definición funciones objetivos	37
4.3.1.1 Función objetivo 1	37
4.3.1.2 Función objetivo 2	38
4.3.2 Restricciones del modelo	38
4.3.3 Linealización de funciones no lineales con valor absoluto	40
4.4 Implementación del modelo	41
4.5 Implementación en Simulink	43
5 Resultados	47
5.1 Parámetros del modelo	47
5.2 Índices de evaluación	48
5.2.1 Índice de desconexión de cargas	49
5.2.2 Índice de cambio de estado de contactores	49
5.2.3 Índice de nivel de carga de la batería	51
5.2.4 Índice de cambio de potencia de la batería	51
5.2.5 Índice multiobjetivo	52
5.3 Simulación del sistema	54
5.4 Casos particulares	56
5.4.1 MHE frente a MPC simple	56
5.4.2 Condición inicial del SOC	58
5.4.3 Desconexión parcial de las cargas	59
5.5 Simulación en Simulink	59
6 Conclusiones	65
Apéndice A Códigos de Python	67
<i>Índice de Figuras</i>	89
<i>Índice de Tablas</i>	91
<i>Índice de Códigos</i>	93
<i>Bibliografía</i>	95

Notación

APU	Auxiliary Power Unit
CB	Circuit Breaker
EPDS	Electrical Power Distribution System
EPS	Electrical Power System
ESS	Energy Storage System
FC	Fuel Cell
FSM	Finite State Machine control
GHG	Greenhouse Gas
IHM	Isolated Hybrid Microgrid
IDG	Integrated Drive Generator
LVDC	Low Voltage Direct Current
MEA	More Electric Aircraft
MES	Main Engine Start
MHE	Moving Horizon Estimation
MPC	Model Predictive Control
NN	Neural Network
PE	Power Electronics
PEC	Power Electronics Converter
RAT	Ram Air Turbine
SOC	State of Charge
LRU	Line Replaceable Unit
VSCF	Variable Speed Constant Frequency
WIPS	Wing Ice Protection System

1 Introducción

Las previsiones del gobierno estadounidense y de las agencias globales han pronosticado un crecimiento del transporte aéreo del 4 – 5 % anual. Esta tasa de crecimiento conduce a una duplicación aproximada de la demanda mundial de vuelos cada 15 años, lo que coincide con los datos históricos anteriores, como se muestra en la Figura 1.1. Este crecimiento conduce a un aumento del combustible quemado y a un fuerte impacto de la aviación en el medio ambiente [2].

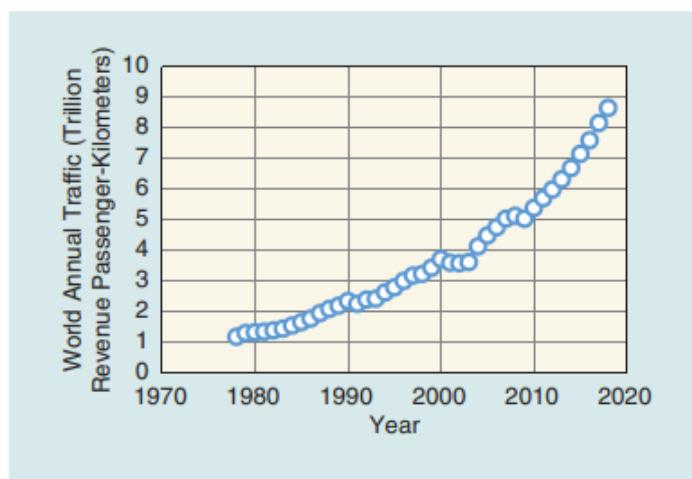


Figura 1.1 Crecimiento histórico del tráfico aéreo comercial mundial. Los datos proceden de la Organización de Aviación Civil Internacional y de Airbus [3].

Las emisiones de gases de efecto invernadero de la aviación comercial están aumentando rápidamente. Por ejemplo, si el sector de la aviación mundial se tratara como una nación, hubiera sido la sexta fuente de emisiones de dióxido de carbono (CO₂) por consumo de energía según datos del año 2015, emitiendo más que Alemania. La Organización Internacional de Aviación Civil (ICAO), que representa a la organización de las Naciones Unidas con autoridad sobre la aviación mundial, prevé que las emisiones de CO₂ de la aviación internacional se tripliquen aproximadamente en 2050 si se mantienen las tendencias actuales. Si otros sectores se descarbonizan de acuerdo con las ambiciones climáticas del Acuerdo de París [4], la aviación podría representar una cuarta parte del presupuesto mundial de carbono a mediados de siglo. Es por esto que se están intentando tomar medidas para reducir ese impacto aproximadamente un 50 % para alrededor del año 2050.

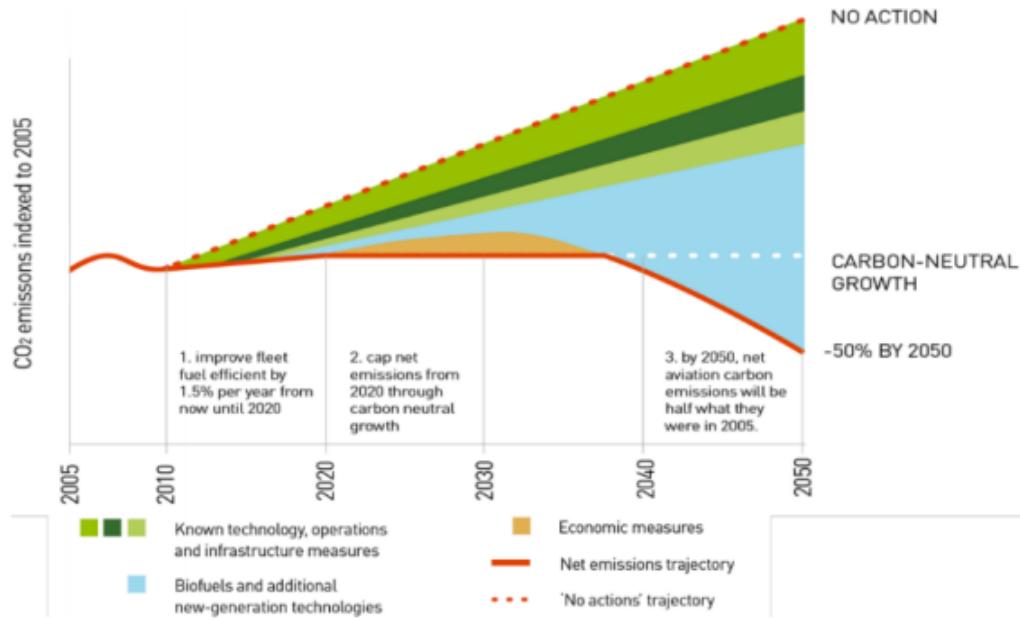


Figura 1.2 Esquema del camino hacia la reducción del CO₂ hasta 2050 [3].

En [5], se analizaron casi 39 millones de vuelos de 2018, y 38 millones de ellos fueron realizados por aviones de pasajeros. Los aviones de fuselaje estrecho y de fuselaje ancho son responsables de más del 75 % de las emisiones de gases de efecto invernadero (GHG) de la aviación, como por ejemplo el Boeing 737 y los aviones como el Boeing 787 y el Airbus 380, que son responsables de alrededor del 43 % y el 33 % de las emisiones. El total de las emisiones de CO₂ de todas las operaciones comerciales, incluyendo el transporte de pasajeros, ascendió a 918 millones de toneladas métricas (MMT) en 2018. Esto supone el 2,4 % de las emisiones mundiales de CO₂ procedentes del uso de combustibles fósiles y un aumento del 32 % en los últimos cinco años.

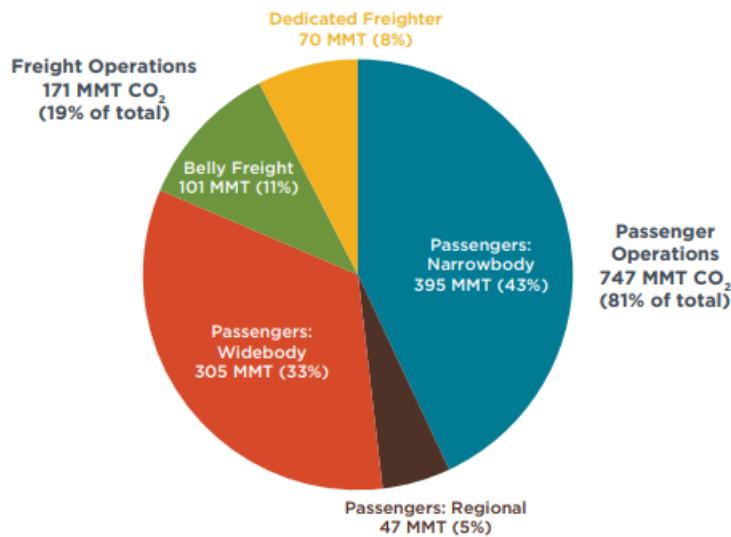


Figura 1.3 Emisiones de CO₂ en 2018 por operaciones y clase de avión [5].

Con las nuevas normativas y planes gubernamentales, en todos los sectores de transporte, se

apuesta por una electrificación de los vehículos para el fomento de la transición energética y, por supuesto, también en el sector aeronáutico está siendo una tendencia clave en los fabricantes de aeronaves en los últimos años. Al igual que los esfuerzos por avanzar hacia los vehículos eléctricos, muchas investigaciones se han centrado en la idea de un avión más eléctrico (MEA). Las motivaciones de esta investigación son similares a las de los vehículos e incluyen los objetivos de reducir las emisiones y disminuir el consumo de combustible.

En comparación con las aeronaves convencionales, las aeronaves más eléctricas tienen una menor dependencia de los combustibles basados en el carbono, menos emisiones, mayor eficiencia, más fiabilidad y menos ruido. Esto último incluso podría acabar con las prohibiciones de los vuelos nocturnos en muchos aeropuertos [6]. Las aeronaves totalmente eléctricas (AEA) proporcionan beneficios adicionales a los MEA, como cero emisiones e incluso un coste de los viajes más barato como resultado directo de la eliminación de la necesidad de combustible, ya que el éste es el principal coste de los viajes aéreos. Sin embargo, a menos que se asuma la generación de electricidad por medio de energías renovable, la reducción/eliminación del consumo de combustible en los MEA/AEA no conlleva necesariamente una reducción del consumo total de energía o de las emisiones de carbono.

1.1 Reseña histórica

El concepto de *More/All Electric Aircraft*, en el que la mayoría o todas las necesidades de energía secundaria del avión se suministran de forma eléctrica, no es ni mucho menos nuevo. En la época de la Segunda Guerra Mundial, aeronaves de ambos bandos ya utilizaban energía eléctrica. Sin embargo, otros aviones de la misma época utilizaban la hidráulica y la neumática para las mismas funciones, y así comenzó el debate sobre las ventajas y desventajas relativas de las diferentes fuentes de energía.

Después de la Segunda Guerra Mundial, tanto la velocidad como el tamaño de los aviones aumentaron, lo que llevó a la necesidad de potenciar funciones que antes podían ser operadas por el piloto sin ayuda, por ejemplo, los controles de vuelo motorizados. Por ello, se produjo una progresión general en todo el mundo hacia los tres tipos de energía secundaria que encontramos como estándar en los aviones actuales, tanto civiles como militares: la energía hidráulica se utiliza para la mayoría de las funciones de accionamiento; la neumática, para el acondicionamiento/presurización y la protección contra el hielo; y la eléctrica, para la aviónica y las funciones de utilidad.

Hasta finales de la década de 1970, la energía eléctrica se limitaba en general a estas funciones electrónicas y de servicios, y sólo se utilizaba ocasionalmente para otras funciones en casos especiales. Sin embargo, hacia finales de la década de 1970, la idea sobre el uso de un único tipo de energía secundaria, empezó a ganar terreno. Basado en los avances en materiales magnéticos permanentes y dispositivos eléctricos de potencia, se podía intuir que el concepto de un avión totalmente eléctrico, sin ningún sistema hidráulico o neumático, podía ofrecer muchas ventajas.

En la década de 1980, la NASA patrocinó los estudios Integrated Digital/Electric Aircraft (IDEA), aplicados a las aeronaves civiles, sobre las ventajas del concepto AEA combinado con la tecnología de control digital [7]. También se iniciaron varios estudios y demostraciones de equipos patrocinados por el ejército, y el interés a este lado del Atlántico aumentó con el inicio de los primeros estudios en Cranfield en 1984.

Las investigaciones realizadas a principios y mediados de la década de 1980 mostraron que, aunque muchos de los equipos eran técnicamente viables y se podían obtener algunos beneficios del concepto AEA, los riesgos que implicaba este cambio total en la tecnología de los equipos eran demasiado grandes, en relación con las ganancias, para la conservadora industria aeroespacial.



Figura 1.4 El Vickers Valiant usaba energía eléctrica para los controles de vuelo primario y otras funciones.

Desde entonces, la industria aeroespacial ha preferido considerar el enfoque gradual en los MEA, con la adopción de la energía eléctrica para las funciones que muestran una ventaja particular.

1.2 El concepto More Electric Aircraft

En las aeronaves convencionales, la potencia generada por un motor se convierte en cuatro tipos principales de energía: eléctrica, mecánica, hidráulica y neumática. Los sistemas y subsistemas de la aeronave pueden utilizar un tipo o una combinación de estos tipos de energía para alguna de sus funciones u operaciones. Sin embargo, todos los tipos de energía tienen diferentes inconvenientes, como el sacrificio de la eficiencia total del motor en el proceso de aprovechamiento de una energía concreta, como ocurre con los sistemas hidráulicos y neumáticos: utilizando cajas de cambios, la energía del motor se convierte en energía mecánica, y posteriormente se transforma en energía eléctrica e hidráulica; utilizando compresores, la energía (como aire del sangrado) se convierte en energía neumática. Algunos de los consumidores de energía más vitales en un avión son: los sistemas de aviónica (eléctricos), las bombas (mecánicas o hidráulicas), los sistemas de actuación (hidráulicos), el sistema de control ambiental (ECS) (neumático) y el sistema de protección contra el hielo del ala (WIPS) (neumático).

En las últimas décadas, se han producido enormes avances para lograr aviones más eléctricos, y muchos subsistemas que antes utilizaban energía hidráulica, mecánica y neumática han sido sustituidos total o parcialmente por sistemas eléctricos. El concepto de MEA hace referencia a la evolución hacia una generación de aviones más eficientes a través de la reducción e incluso la eliminación de los sistemas de accionamiento tradicionales para que el sistema eléctrico sea el de mayor proporción o el único de la aeronave. Un MEA puede conseguirse simplemente sustituyendo un subsistema mecánico, hidráulico o neumático por una alternativa eléctrica, de forma que penalicemos menos el rendimiento del motor, ya que una menor parte de la potencia generada en él tenga que ser transformada en otros tipos de energía.

En la Figura 1.6, se representa un esquema de la generación y la distribución de energía en un

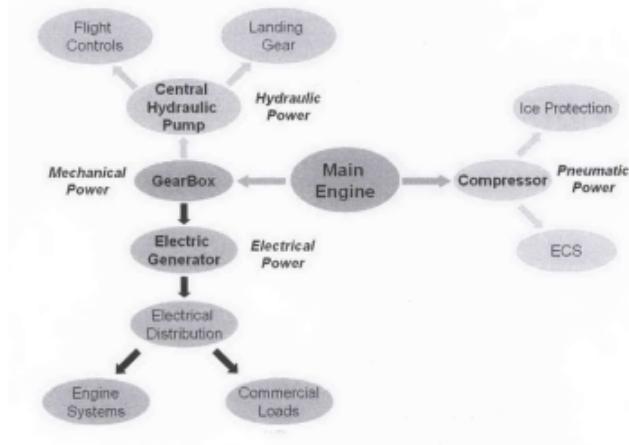


Figura 1.5 Esquema de la distribución de energía convencional [8].

MEA con aire sangrado. En él, se puede apreciar como la potencia generada en los motores de propulsión principales, se convierte en potencia neumática a través de los compresores y el aire sangrado, y en potencia eléctrica por medio de un generador. Posteriormente, cada tipo de potencia se distribuye a los sistemas correspondientes para su funcionamiento.

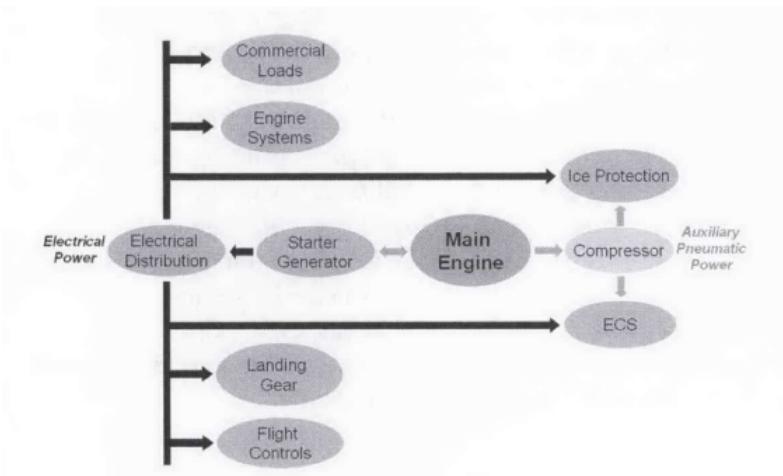


Figura 1.6 Esquema de la distribución de energía en MEA (Con sangrado de aire) [8].

En el B787, un MEA lanzado por primera vez en 2009, se implementó una arquitectura de sistema sin sangrado que sustituye al sistema neumático tradicional. El colector de sangrado es sustituido por un sistema eléctrico de alta potencia con mayor capacidad para transformar potencia eléctrica a partir de los motores, que además de las funciones del sistema eléctrico, soporta la mayoría de las funciones del avión que tradicionalmente se realizaban a través del aire de purga. También, el WIPS se sustituyó completamente por un sistema electrotérmico de antihielo/deshielo (Figura 1.7).

Continuando con la idea, en un AEA no sólo se sustituyen todos los sistemas reemplazables por sus alternativas eléctricas. La electrificación de la propulsión constituye el siguiente reto, exigiendo un aumento de la densidad de potencia y empujando a la tecnología a sus límites, haciendo uso de unidades de energía electroquímica como baterías y pilas de combustible. La propulsión híbrida ya se está investigando y se han presentado algunos prototipos como el desarrollado en [10], una

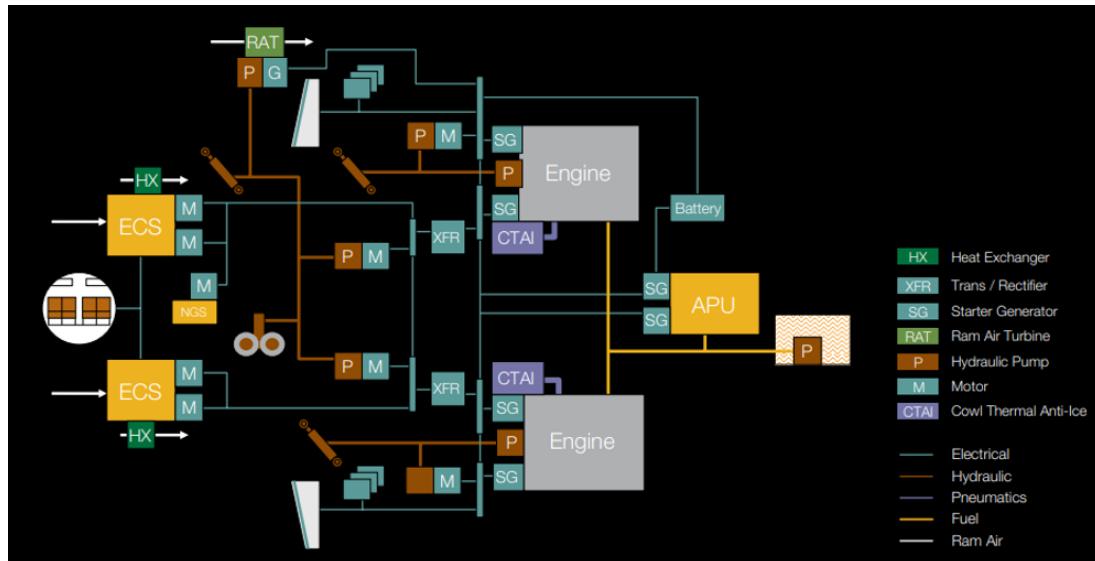


Figura 1.7 Arquitectura del B787 sin sangrado de aire [9].

aeronave con un motor híbrido paralelo. La idea es muy similar a la de los coches híbridos: que el motor funcione en el punto de máxima eficiencia y que los motores eléctricos alimenten el motor. Un AEA de cuerpo estrecho y pasillo único para 150 pasajeros que opere adecuadamente en misiones comerciales típicas puede considerarse un objetivo alcanzable en los próximos 20-30 años.

Los beneficios principales de un MEA están relacionados con una mayor eficiencia:

- Los sistemas de generación se simplifican hacia una única fuente de potencia.
- Aumenta la seguridad y la fiabilidad al haber menos probabilidad de fallos mecánicos y de incendios o explosiones.
- Incrementa la eficiencia energética reciclando el aire del sangrado de los motores
- Reduce el peso de la aeronave, traduciéndose en un ahorro de combustible.

Por último, otras consideraciones para la electrificación del avión son los costes de mantenimiento asociados a los diferentes sistemas y el menor riesgo de fuga de fluidos. Esto es importante porque cualquier tiempo inesperado de "inmovilización" conlleva un aumento significativo de los costes para las compañías aéreas. Por lo tanto, incluso en el caso de que la electrificación no mejore el peso, el volumen y el coste inicial del avión, el ahorro potencial a través del aumento de la fiabilidad del despacho y la reducción del mantenimiento puede tener sentido desde el punto de vista financiero.

1.3 Desafíos del sistema eléctrico de potencia del avión

El reemplazo de los sistemas neumáticos e hidráulicos tradicionales por sistemas eléctricos en las aeronaves para alcanzar una mayor eficiencia y un menor uso de combustible, causa una mayor demanda de energía y un incremento en el número de problemas de seguridad relacionados con la electrificación.

La electrificación supone incrementar la potencia eléctrica, por ejemplo, siendo necesarias decenas de kVA sólo por la sustitución del WIPS, que debe ser proporcionada por el sistema de potencia eléctrica del avión (EPS). La potencia eléctrica ha pasado de unos 100kVA en la primera generación del B737 lanzada en 1965, a 1MVA en el B787. Esta potencia eléctrica de 1MVA no

incluye la generación de la unidad de potencia auxiliar (APU), que en sí misma puede ser de hasta 450kVA.

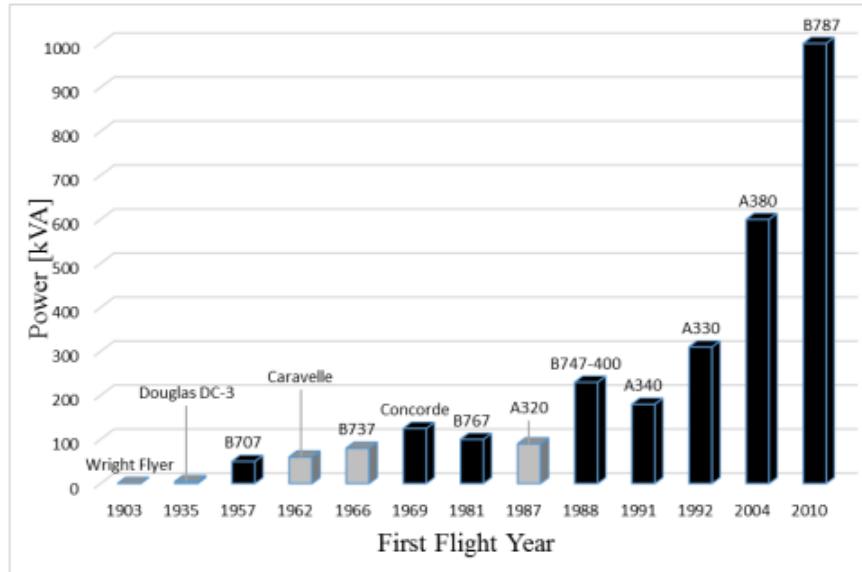


Figura 1.8 Evolución de la necesidad de energía eléctrica (gris: aviones de corto a medio alcance. negro: aviones de medio a largo alcance) [11].

El EPS de la aeronave es una microrred, sistemas que agregan recursos energéticos distribuidos, cargas y dispositivos electrónicos de potencia de forma estable y equilibrada. Se basan en sistemas de gestión de la energía para programar de forma óptima los recursos energéticos distribuidos. Aparte de los elementos mostrados en la Figura 1.9, un EPS está formado por diferentes componentes, como los convertidores de electrónica de potencia (PE), las máquinas eléctricas, las unidades de energía electroquímica (baterías), los disyuntores (CBs), diferentes cargas, etc. Estos están conectados a los buses apropiados y son necesario tenerlos en cuenta al analizar un EPS.

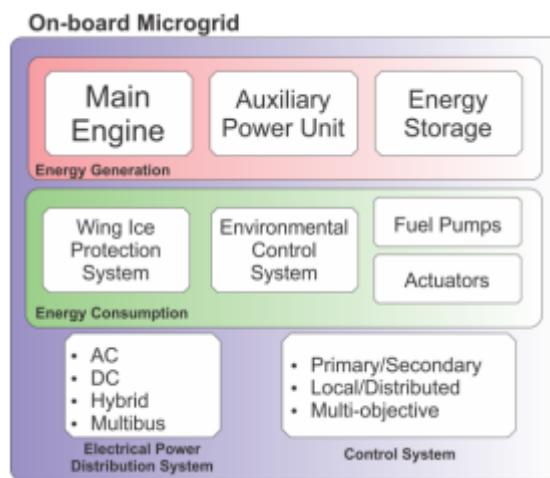


Figura 1.9 Elementos de la microrred a bordo [12].

Las cargas críticas, responsables de la seguridad del vuelo, como los actuadores de la superficie

de control de vuelo y los sistemas de control ambiental, deben recibir energía en todos los escenarios de vuelo, independientemente de los picos de alta potencia o de momentos de emergencias que provoquen fallos. El sistema de almacenamiento de energía (ESS) tiene un papel importante en la MEA, no sólo para apoyar las tensiones de los buses, sino también como respaldo para hacer frente a la escasez de potencia durante los fallos que se producen en los sistemas de generación o transmisión, o los picos de alta potencia. Además, la desconexión de cargas no críticas (por ejemplo, la iluminación de la cabina de pasajeros) también se utiliza para reducir el uso de potencia en los picos, ayudando a cubrir la potencia para las cargas críticas. Combinar inteligentemente la conexión/desconexión de las cargas con las capacidades del ESS, es esencial para el funcionamiento seguro del sistema de a bordo.

Por otro lado, aunque existen similitudes entre la EPS de aeronave y una microrred en tierra, la EPS de aeronave es una microrred aislada a bordo, diferente de las microrredes en tierra [12]. A continuación, se describen algunas diferencias:

- Se necesita un nivel muy alto de fiabilidad y seguridad en las aeronaves, lo cual requiere métodos como aislamientos o separación de los buses.
- La densidad de potencia es uno de los parámetros prioritarios, ya que cuanto más peso haya por parte del sistema de distribución de energía eléctrica (EPDS), más perjudicará el consumo de combustible.
- En los aviones, la prioridad de la carga cambia durante una misión.
- Los EPS de las aeronaves permiten una mejor reconfiguración.

Muchos modelos y enfoques que se han propuesto específicamente para las microrredes terrestres pueden aplicarse y adoptarse, si es necesario, para ser utilizados en MEA/AEA. Además, el MEA EPS es intrínsecamente híbrido, ya que está compuesto de buses de CA y CC, por lo que proponer/adoptar métodos que traten adecuadamente una microrred híbrida aislada es una necesidad.

Surge de los desafíos ambientales, energéticos y operacionales expuestos a lo largo de esta sección, la necesidad obtener técnicas y estrategias de control y optimización avanzadas de los EPS de las aeronaves para cumplir con los requisitos que hemos expuesto, como la reducción del consumo de combustible o el suministro ante cualquier fallo o emergencia. Por ello, una gran cantidad de estudios se han publicado buscando desarrollar modelos y algoritmos adecuados para abordar la complejidad del funcionamiento óptimo de los MEA. En secciones posteriores, se debatirá sobre distintas técnicas de gestión y se expondrá las razones por las cuales se adopta el MPC como estrategia de control para un MEA EPS.

2 Sistema de potencia eléctrica

El objetivo de esta sección es presentar una revisión de los MEA EPS. Aunque se centre en los EPS, se trata de ofrecer un debate general y una revisión tecnológica sobre otras cuestiones relacionadas con los EPS y las microrredes.

Todos los aviones utilizan un EPS aislado, en el que la seguridad del suministro y la densidad de potencia representan los principales requisitos. Coexisten diferentes sistemas de distribución (CA y CC) y niveles de tensión, donde los convertidores de potencia juegan un papel fundamental en conectarlos con una alta fiabilidad. Garantizar la seguridad del suministro con una redundancia limitada es uno de los objetivos del diseño del sistema, ya que permite aumentar la densidad de potencia. Este es el principal reto suele abordarse con una gestión adecuada de la carga y estrategias avanzadas de control.

La capacidad de potencia los generadores conectados a los motores principales dependen del tipo de motor, del tipo de generador y de las condiciones reales de funcionamiento, y esta potencia se transfiere a las cargas a través del sistema de distribución. Teniendo en cuenta que los convertidores de electrónica de potencia interconectan la mayoría de las cargas, las interacciones entre el sistema de control hacen que la estabilidad de la microrred sea un reto, ya que la estabilidad de la tensión y la frecuencia no puede ser ayudada por la presencia de grandes generadores síncronos como la red tradicional.

A parte de la estabilidad, se encuentran otras dificultades. Dos de los requisitos fundamentales en un MEA EPS son la densidad de potencia y la resistencia a fallos. Desafortunadamente, ambos requisitos son contradictorios, debido a que una forma directa de lograr resistencia a un fallo en el EPDS es implementar la redundancia a gran escala, la cual aumenta el peso total de la aeronave, disminuyendo la densidad de potencia. Mientras que, para incrementar la densidad de potencia, debemos reducir el peso total de los dispositivos electrónicos, dejando sin lugar a la redundancia. El reto consiste en minimizar la potencia total instalada, manteniendo los niveles de seguridad en la aeronave a través de los múltiples sistemas redundantes.

Un gran problema en el EPS de una aeronave son los picos de potencia, para los cuales este debe ser capaz de proporcionar la potencia necesaria. El EPDS debe ser dimensionado para la potencia máxima en el peor de los casos. Por ello, la solución óptima se encuentra cuando se minimiza la diferencia entre la potencia máxima y la media.

2.1 Sistemas de potencia a bordo

Con el fin de proporcionar una demanda de energía muy superior, los sistemas eléctricos de potencia a bordo han experimentado cambios significativos, cumpliendo al mismo tiempo requisitos extremadamente estrictos en cuanto a peso y volumen, seguridad y fiabilidad, calidad de la energía eléctrica, disponibilidad, etc. Los cambios afectan tanto a las arquitecturas de los EPS como a los subsistemas individuales responsables de la generación, distribución, conversión, utilización y almacenamiento de energía.

2.1.1 Generación de potencia en MEA EPS

En las aeronaves se emplea una generación de potencia eléctrica multinivel. La mayor parte de la potencia proviene de las fuentes de potencia primarias, es decir, de los generadores principales conectados a los motores de propulsión, normalmente de corriente alterna. A su vez, encontramos fuentes secundarias normalmente usadas en tierra como las unidades de potencia auxiliar (APU), y fuentes terciarias como la *Ram Air Turbine* (RAT) para casos de emergencia.

2.1.1.1 Generadores principales

Con el fin de suprir este gran aumento de la demanda de potencia eléctrica en las aeronaves, el concepto MEA representa una de las mayores motivaciones y un gran impulsor en la investigación y desarrollo de nuevas tecnologías de generación de energía eléctrica. A continuación, se describen cuatro tipos de generadores usados en aviones:

1. El *Integrated Drive Generator* (IDG) o *constant frequency* CF, (Figuras 2.1 y 2.2, es una de las opciones más comunes. Dispone de una caja de cambios para adaptar la velocidad del motor a una velocidad constante, por lo que encarece su precio y la hace menos fiable.

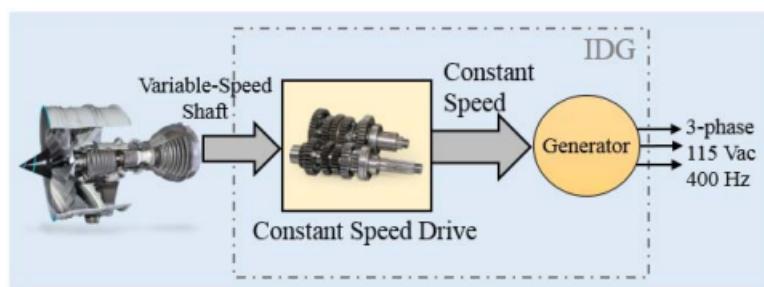


Figura 2.1 sistema CF IDG [13].

2. El *Variable Speed Constant Frequency* (VSCF) de la Figura 2.3 es el generador preferido para la mayoría de las aplicaciones militares. Este se caracteriza por una generación de potencia eléctrica en CC por medio de un *DC link*, que incorpora un rectificador y un inversor. Destacan su simplicidad y fiabilidad
3. Los cicloconvertidores VSCF convierten directamente la potencia de entrada de CA de frecuencia variable en potencia de CA con frecuencia y amplitud fijas
4. El generador de frecuencia variable (VF) es el más reciente. Las características más prometedoras de la VF son su pequeño tamaño, peso, volumen y coste en comparación con otras opciones de generación de energía eléctrica para aviones. Sin embargo, la FV puede suponer un riesgo significativo a niveles de potencia más elevados, especialmente con cargas de motor de alta potencia. Figura 2.5

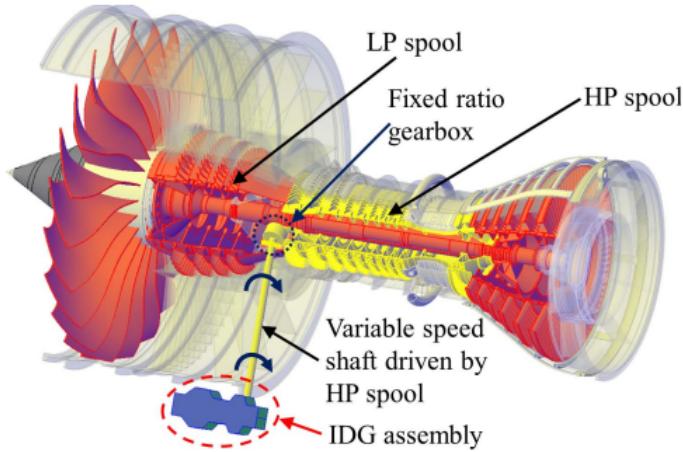


Figura 2.2 Motor turbofán de doble eje [13].

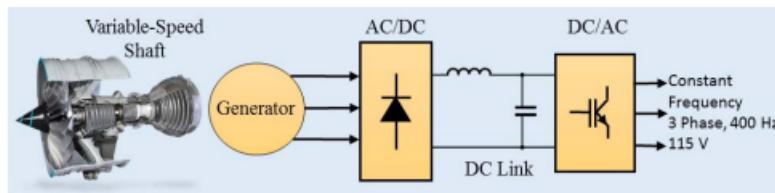


Figura 2.3 sistema VSCF con DC link [13].

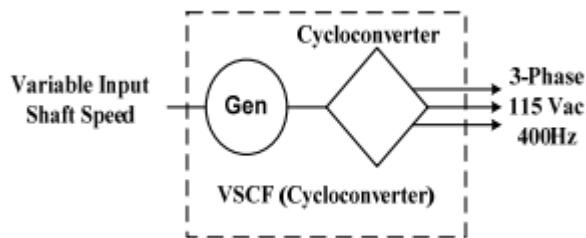


Figura 2.4 Cicloconvertidor VSCF [14].

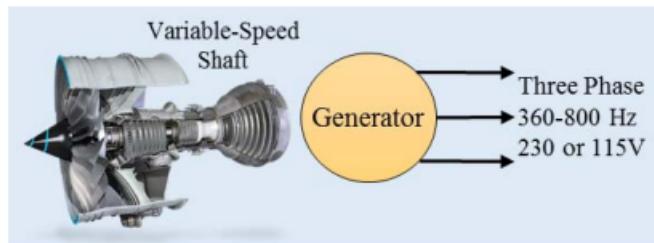


Figura 2.5 sistema del generador VF [13].

Como fuente de potencia principal en la aplicación de MEA, el motor síncrono de tres fases es

la tecnología más empleada en la mayoría de aviones comerciales y militares actualmente. Este generador es altamente fiable e intrínsecamente seguro, ya que la excitación de campo puede eliminarse. A lo largo de los años, se ha conseguido aumentar su potencia generada, sin embargo, el aumento previsto de los requisitos de generación de energía eléctrica en el MEA sugiere que los generadores de alta potencia se acoplen directamente al motor, montados en el eje del motor y utilizados también para su arranque. Se ha investigado la aplicación en los MEA de los siguientes tipos de generadores:

El generador de inducción requiere una electrónica de potencia compleja y se considera poco probable que tenga la densidad de potencia de los otros generadores.

La máquina de reluctancia conmutada tiene una estructura robusta muy sencilla y puede funcionar en un amplio rango de velocidades. La electrónica de potencia es comparativamente sencilla y el generador es intrínsecamente tolerante a los fallos.

El generador de imanes permanentes se considera una de las opciones más atractivas para el MEA debido a su fiabilidad, robustez y facilidad de refrigeración. Tiene una alta densidad de potencia y una buena eficiencia en un amplio rango de velocidades.

2.1.1.2 Unidad de Potencia Auxiliar

La fuente secundaria más común es la unidad de energía auxiliar (APU), la cual generalmente proporciona energía cuando la aeronave está en tierra para el arranque del motor principal (MES). Las APUs también pueden entregar potencia mientras están en el aire bajo ciertas condiciones de operación, especialmente durante emergencias. Sin embargo, la capacidad de energía de una APU está limitada a altitudes muy altas debido a la reducida densidad del aire. Existen varias baterías en la aeronave para poner en marcha la APU y proporcionar energía de reserva para los equipos críticos de la cabina, así como para otras funciones importantes como la iluminación de emergencia para los pasillos.

Las APUs tradicionales proveían también potencia hidráulica de respaldo en caso de avería del principal sistema hidráulico de la aeronave. Con el encendido eléctrico de los motores y de los sistemas antihielo, se podrían eliminar la alternativa hidráulica y neumática que presentaban las APUs, teniendo por tanto APUs totalmente eléctricas.

Una de las principales diferencias entre las APU convencionales y las eléctricas es que la APU eléctrica requiere un generador muy grande y, como resultado, las consideraciones de diseño del sistema global pueden requerir que se utilicen dos APU por la potencia total necesaria y/o por razones de redundancia.

Las nuevas tecnologías para las fuentes de energía secundarias consideran principalmente la sustitución de la APU por pilas de combustible (FC) que ofrecen una eficiencia mucho mayor y no producen emisiones. Sin embargo, la inclusión de FC a bordo de la MEA requiere otra pieza de electrónica de potencia, el convertidor para interconectar la FC con el EPS de a bordo. También hay informes sobre el desarrollo de las fuentes secundarias basadas en la combinación de baterías de iones de litio con supercondensadores.

2.1.1.3 Ram Air Turbine

Por último, se incluyen fuentes terciarias como la *Ram Air Turbine* (RAT). Es un dispositivo formado por un mecanismo de extensión, un mástil, una bomba hidráulica y una hélice. Inicialmente, está sujetada mediante un gancho que en caso de emergencia se suelta haciendo que el aire mueva la hélice alimentando así a un generador eléctrico. También hay otra de igual funcionamiento pero que mueve una bomba hidráulica.

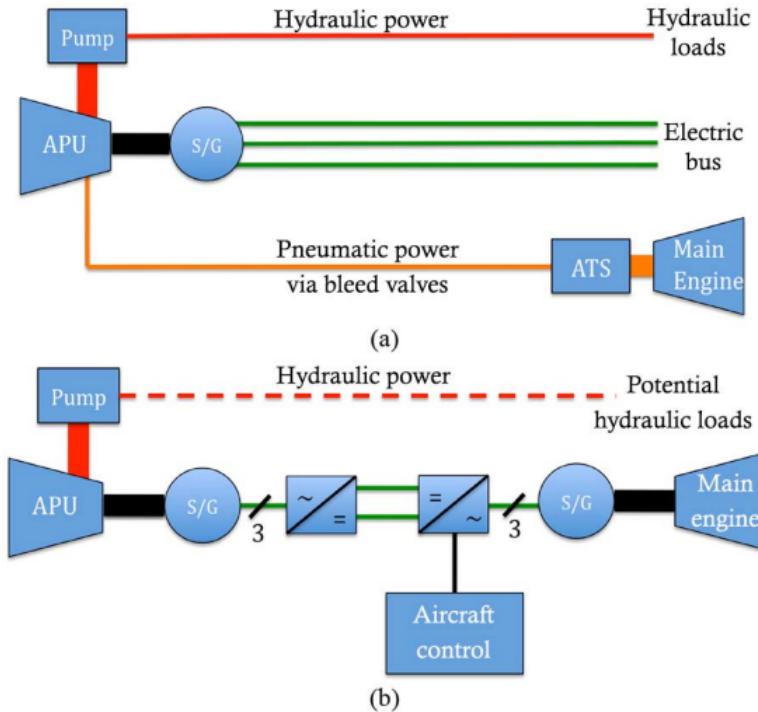


Figura 2.6 (a) APU tradicional con energía eléctrica, neumática e hidráulica. (b) APU eléctrica que ilustra el MES [15].

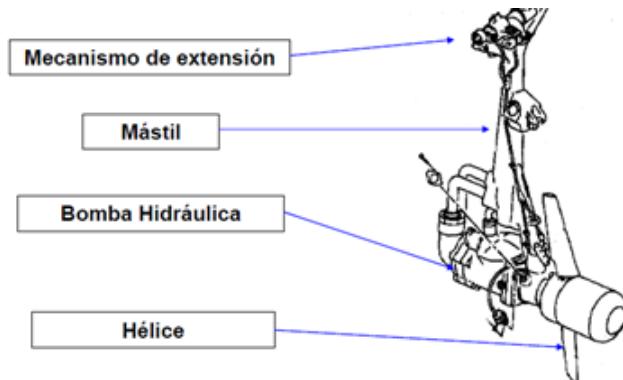


Figura 2.7 RAT.

2.1.2 Consumo de energía a bordo del MEA

Haciendo referencia al objetivo principal del MEA de sustituir los actuadores hidráulicos y neumáticos de las aeronaves tradicionales, se hará una revisión de las nuevas tecnologías de accionamiento eléctrico que sustituyen a los anteriores sistemas. Se ofrecerá una visión general de los mismos y de cuáles son sus diferencias respecto a los tradicionales. Cabe destacar que, aunque no estén recogidas el resto de cargas de la aeronave, como luces o servicios de cocina, estas siguen estando presentes y consumen energía de la microrred. En la Figura 2.8 se ilustra la ubicación de las tecnologías más eléctricas en un avión moderno.

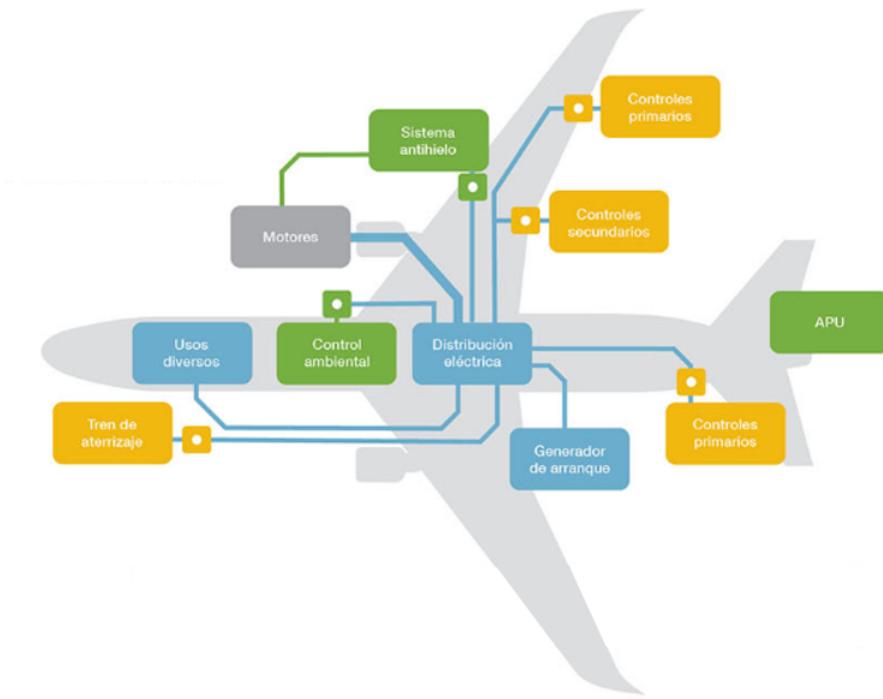


Figura 2.8 Subsistemas eléctricos en MEA [16].

2.1.2.1 Sistema de protección contra el hielo

En lugar de usar aire caliente extraído del motor como en los aviones convencionales, en un MEA se incorporan alfombras térmicas resistivas a través de las cuales se aporta el calor necesario en las alas, superficies de mando, sensores, etc. para evitar los graves problemas como consecuencia de la aparición de hielo. Gracias a la electrónica de potencia controla eficientemente la potencia suministrada y la temperatura de la superficie indicada. En el caso de los aviones de tamaño medio, esta carga puede requerir entre 40 y 60 kW en el modo de deshielo y hasta 200 kW en el modo antihielo [17].

2.1.2.2 Sistema de control ambiental

Se trata de un sistema necesario para garantizar tanto la habitabilidad en cabina (control de temperatura y presión) como la protección de equipos (ventilación). Debe cumplir condiciones de altura, presión, temperatura, humedad, etc. En el MEA, se emplea un accionamiento eléctrico para comprimir el aire ambiente y controlar los parámetros del aire para proporcionar comodidad a los pasajeros. Para los aviones de tamaño medio, se necesitan varios ECS, con una potencia típica de 70 kW cada uno [8]. Una comparación entre los ECS convencionales y el de un MEA sin sangrado de aire se muestra a continuación en la Figura 2.9

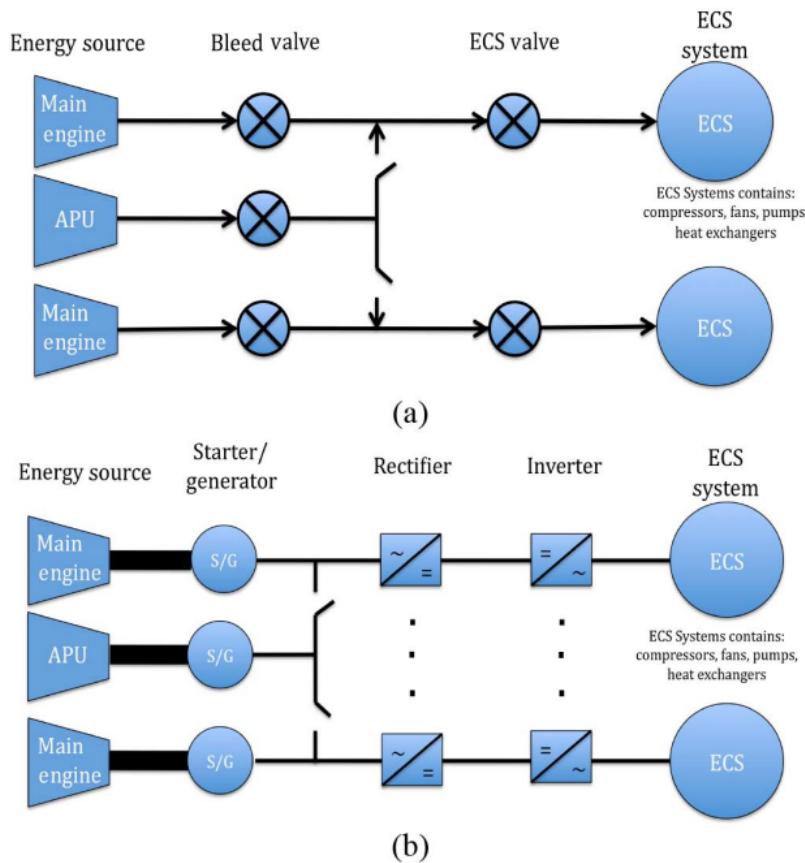


Figura 2.9 (a) ECS con sangrado de aire. (b) MEA ECS [15].

2.1.2.3 Actuadores

En las aeronaves tradicionales, los sistemas hidráulicos se utilizan para el control de las superficies primarias y secundarias, el frenado, el tren de aterrizaje y muchas otras funciones importantes. Estos sistemas hidráulicos dependen de actuadores de accionamiento mecánico, pero la tendencia actual es sustituirlos por actuadores electrohidráulicos (EHA) o electromecánicos (EMA).

Tanto los EMA como los EHA requieren un motor eléctrico y un inversor. Los EHA incluyen una bomba hidráulica reversible, un cilindro y un depósito de fluido hidráulico. Estos actuadores resultan atractivos para los aviones del futuro ya que eliminan la fuente hidráulica externa y los sistemas de tuberías. Por lo tanto, los EHA se consideran ventajosos por su peso, volumen, fiabilidad de envío y ventajas de coste. Por el contrario, los EMA no utilizan ningún tipo de energía hidráulica, sino que emplean una caja de cambios y un sistema mecánico para transformar el movimiento rotatorio en movimiento lineal. Esto permite que los motores EMA hagan funcionar una bomba hidráulica reversible.

Como resultado, los EMA son más eficientes que los EHA y son una mejor alternativa para un funcionamiento sin fugas y con mayor fiabilidad. Sin embargo, uno de los principales inconvenientes de los EMA es su potencial de atasco mecánico. Este es un reto importante que hay que resolver para que los EMA se conviertan en una opción viable para aplicaciones de seguridad crítica, como las superficies primarias y el despliegue del tren de aterrizaje. Dependiendo de las superficies de vuelo, la potencia nominal de los EMA puede variar de 2 a 40 kW.

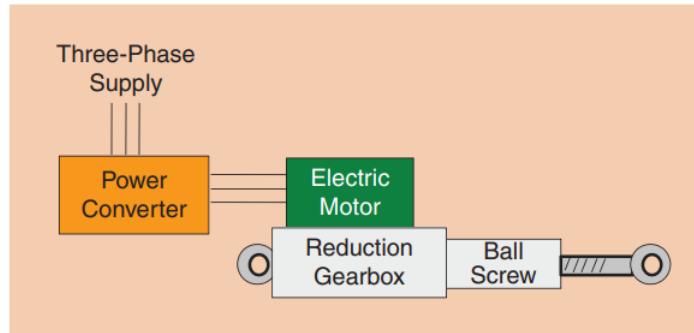


Figura 2.10 Diagrama del sistema de un EMA [18].

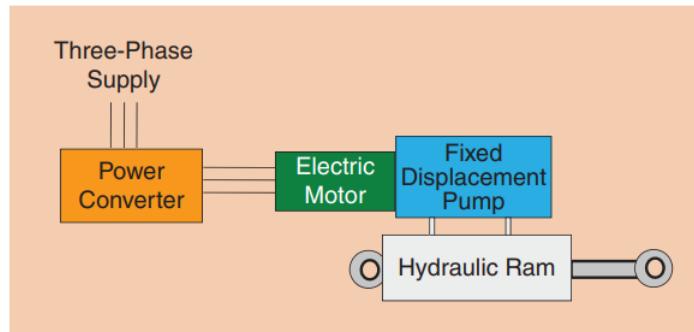


Figura 2.11 Diagrama del sistema de un EHA [18].

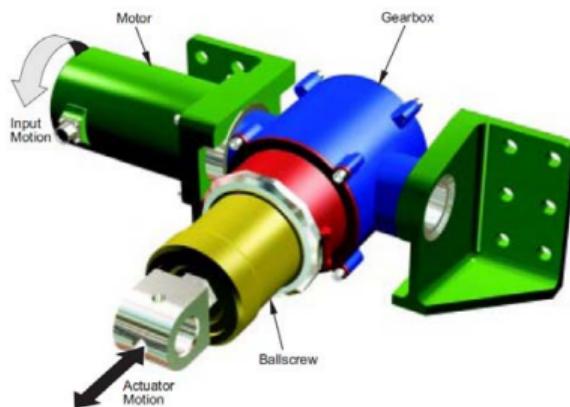


Figura 2.12 Actuador electromecánico de alta potencia [18].

2.2 Arquitecturas de EPS

Se comienza dando una visión general de como son los sistemas eléctricos de potencia en las aeronaves antiguas, para resaltar los cambios introducidos en los MEA. Las arquitecturas MEA EPS que se presentarán a continuación son los desarrollos tecnológicos en microrredes más avanzados en términos de EPS de aeronaves. Se analizarán de una forma general para explicar su funcionamiento, para posteriormente analizar las dificultades de su gestión y optimización.

2.2.1 Arquitectura de EPS de aeronaves convencionales

El sistema eléctrico de las aeronaves de transporte comercial convencionales suele utilizar una tensión de 115V con una frecuencia constante de 400Hz. En esta arquitectura, el generador está conectado al motor principal a través de un accionamiento mecánico, que mantiene la velocidad y frecuencia eléctrica constantes en el bus eléctrico del avión. Sin embargo, la generación de energía eléctrica necesaria por motor es menor que la de un MEA más reciente, debido a que muchas funciones importantes de este avión no se alimentan de energía eléctrica, como pueden ser el arranque del motor principal, los ECS, el deshielo o el sistema hidráulico.

En la arquitectura tradicional encontramos dos buses de tensiones diferentes. El primero, caracterizado por la tensión y frecuencia constantes indicados en el párrafo anterior, y un segundo bus de baja tensión de 28VDC, obtenidos convirtiendo la tensión de 115VAC y 400Hz, mediante unidades rectificadoras y transformadores. Por último, se realiza una La reducción adicional del voltaje de corriente continua a 5V y 3.3V para alimentar los circuitos integrados, los microporcesadores y la electrónica de nivel de señal.

Además, existen varios buses en la aeronave para acomodar las redundancias necesarias para las emergencias. Se utiliza un disyuntor para unir los buses según sea necesario y se utilizan un gran número interruptores para desconectar los generadores, las cargas y los buses del sistema eléctrico de la aeronave.

2.2.2 Arquitecturas principales de MEA EPS

Las diferentes arquitecturas de EPS para aviones dan lugar a diferentes pesos netos, eficiencia y fiabilidad. Según el estado del arte, el diseño de las microrredes considera dos formas principales de distribución de potencia: una distribución primaria de corriente alterna de alta tensión 230V con frecuencia variable 360-900Hz; o una distribución primaria de alta tensión en corriente continua 270, ±270 ó 540V.

2.2.2.1 Arquitectura híbrida CA/CC

Una de las arquitecturas del MEA EPS más populares y recogidas en artículos, es la arquitectura con un bus principal de CA, que se encuentra en los nuevos aviones como el Boeing 787 y los Airbus A380 y A350. En esta arquitectura, los buses correspondientes a cada generador suelen presentar una estructura aislada, teniendo cada generador por separado sus propias cargas y capas de distribución. Únicamente en situaciones de fallo, algunos de los contactores de interconexión pueden transferir la carga de fuente primaria sana a un bus con déficit energético.

El bus convencional de tensión y frecuencia constantes se sustituye por el bus de tensión constante y frecuencia variable. En este escenario, mientras la tensión se regula a 115 o 230VAC, la frecuencia del bus cambia proporcionalmente a la velocidad del motor, y también dependiendo del motor y del tipo de avión. En la figura 2.13 se representa conceptualmente un MEA EPS similar al de un B787.

Se pueden observar cuatro buses de potencia distintos: 230VAC en el bus AC primario, 115VAC en el bus AC secundario, 270VDC en el bus HVDC, y 28VDC para alimentar la aviónica (LVDC).

Debido a su naturaleza híbrida, esta arquitectura es rica en electrónica de potencia debido a que depende en gran medida de las conversiones electrónicas de potencia. No obstante, dado que muchas de las cargas a bordo sólo se requieren durante un período relativamente corto durante la misión de vuelo, la tasa de utilización de la electrónica de potencia dentro de este tipo de EPS es baja. La mejora del número de dispositivos de conversión de potencia permitirá reducir significativamente el peso y el coste total de la EPS.

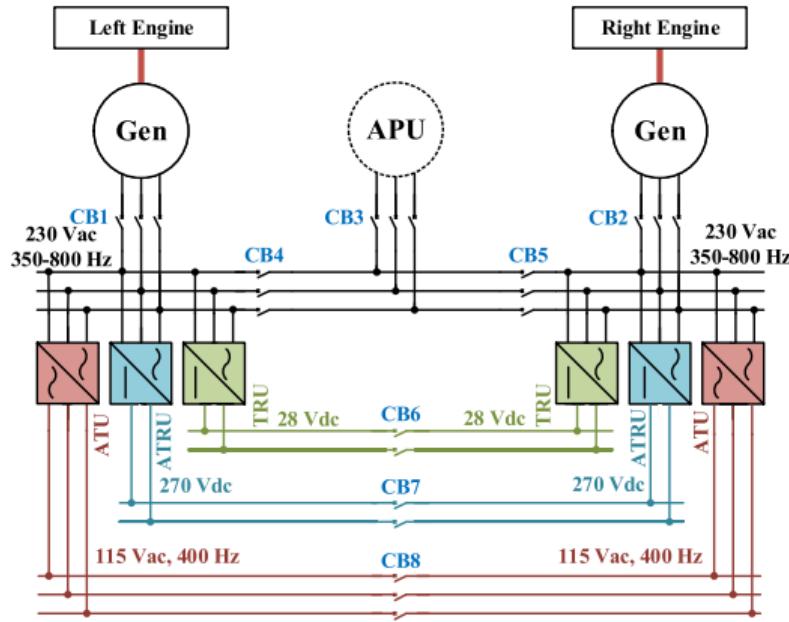


Figura 2.13 EPS con bus AC primario de 230V con frecuencia variable 358-800Hz. Cargas conectadas a los buses correspondientes [19].

Otro ejemplo de arquitectura MEA EPS híbrida de CA/CC es la estudiada en el proyecto MOET de la Unión Europea [20], ilustrada en la Figura 2.14.

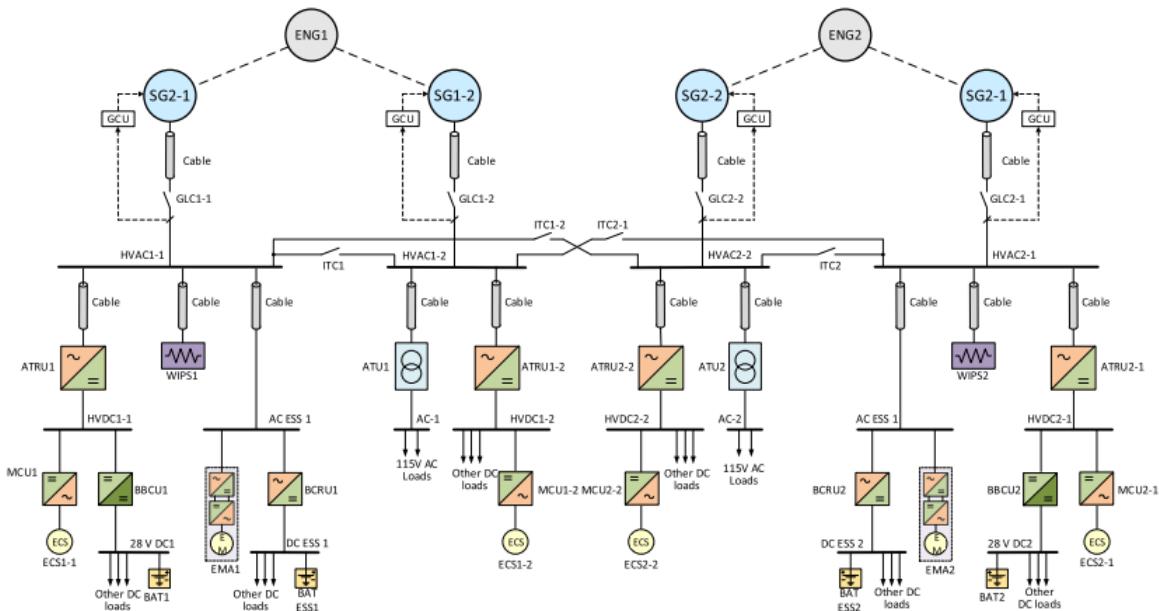


Figura 2.14 Arquitectura MOET MEA EPS [12].

2.2.2.2 Arquitectura de bus único

En la literatura, la distribución de energía primaria en los aviones se ha basado tradicionalmente en el paradigma de un solo generador por bus, con una distribución conmutada que proporciona la conectividad y la integridad del sistema. En cambio, el concepto propuesto de "bus único" utiliza el enfoque de microrred en el que todos los generadores y cargas están conectados a un único bus de distribución, como se ilustra en la Figura 2.15. En comparación con el sistema de CA generalmente adoptado en las aeronaves modernas, el sistema HVDC de 270V es superior debido a su alta eficiencia, la conveniencia de poner en paralelo múltiples buses de CC, y la no necesidad de compensación de potencia reactiva. Por lo tanto, el HVDC es la mejor opción para la estructura EPS.

Esta configuración de bus único se ha utilizado ampliamente en otras aplicaciones, como las microrredes residenciales. Este sistema tiene el potencial de reducir considerablemente el peso de la EPS, ya que la masa del bus se reduce y la función de aislamiento de fallos de la carga y del generador puede integrarse en los convertidores de potencia. Además, el reparto controlado de energía entre los generadores tiene el potencial de reducir la capacidad del generador y operar a niveles de máxima eficiencia.

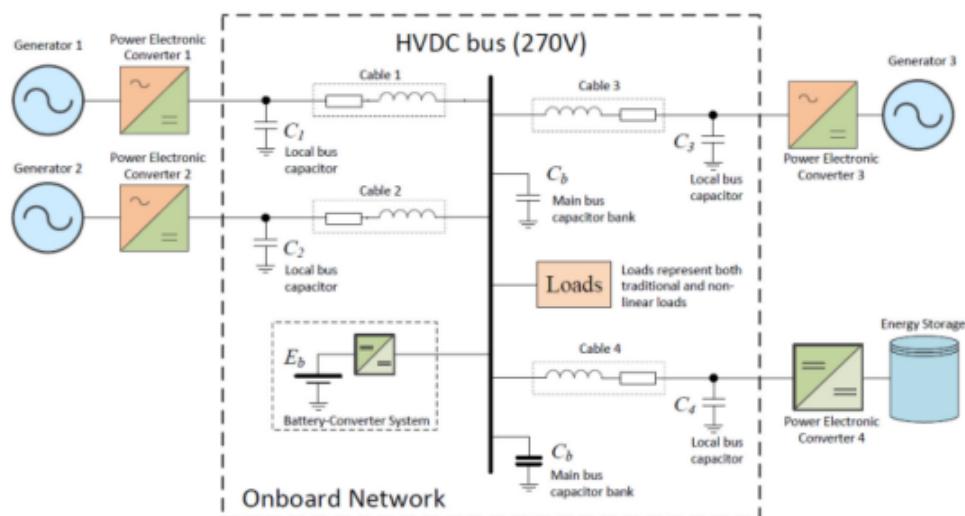


Figura 2.15 Arquitectura MEA EPS de bus único [12].

Esta arquitectura es una de las más prometedoras para el futuro del MEA, y en consonancia con adecuadas estrategias de control, la microrred podría reducir el valor de potencia total necesaria. De esta forma, no haría falta una generación de potencia tan elevada y se podrían disminuir los valores de diseño de las fuentes principales, conllevando una reducción sustancial del peso.

Mientras que los principios de control de esta topología para garantizar el cumplimiento de los requisitos de calidad de la energía se continúan investigando y poniendo en práctica, la protección contra fallos es uno de los retos que aún deben abordarse ya que solo se dispone de un bus. Cabe destacar, que esta arquitectura es con la que se trabajará más adelante en este estudio.

2.2.3 Diferencia distribución flexible/centralizada

Una ventaja adicional que aporta el paso al MEA, es la flexibilidad para generar y distribuir la potencia de forma eficiente cerca de donde se consume. El EPS de la aeronave puede ser centralizado o distribuido remotamente. Una tendencia importante en el desarrollo de los EPS MEA tiene que ver con la introducción de arquitecturas distribuidas en lugar de la centralización de los aviones tradicionales.

En el modelo de avión tradicional, toda la energía se genera en las alas, cerca de los motores principales y en la popa, cerca de la APU, antes de dirigirse a la parte delantera del avión para su protección y control. Esta arquitectura tradicional se muestra en la Figura 2.16 y se denomina distribución centralizada del sistema de energía.

Las arquitecturas distribuidas permiten lograr importantes beneficios. Este enfoque permite a las unidades de distribución, como los convertidores de PE, ser colocadas más cerca de las cargas. En la imagen (b) de la Figura 2.16, se ilustra un ejemplo de esta moderna configuración de red perteneciente al Boeing 787.

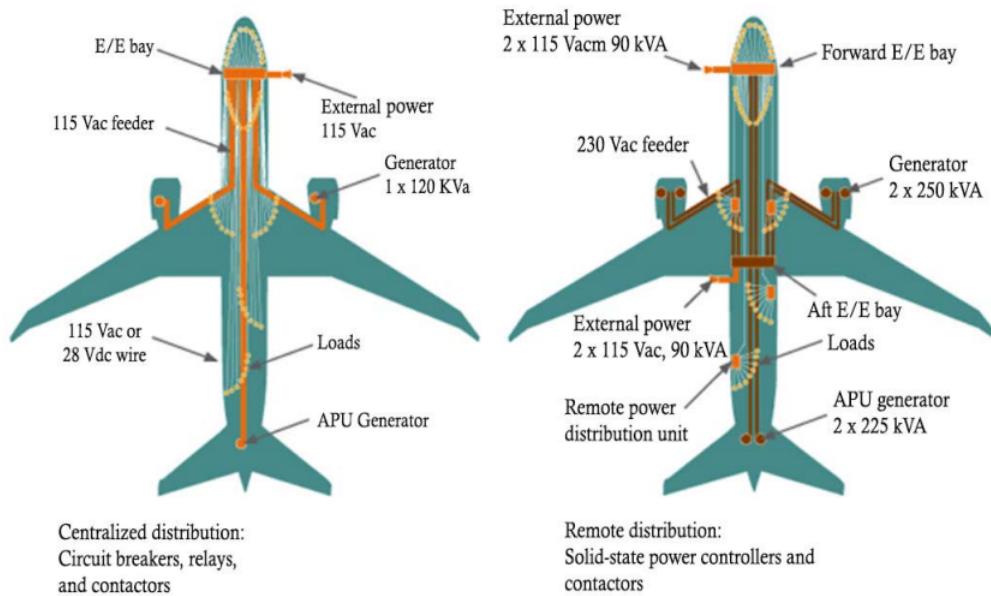


Figura 2.16 (a) Distribución centralizada. (b) Distribución remota [15].

El resultado es un aumento de la eficiencia del sistema de distribución de energía, ya que las pérdidas en la línea se reducen debido a las distancias más cortas entre la generación y el consumo. Además, se puede conseguir un importante ahorro de peso y volumen, mediante la optimización del cableado y la reducción de la potencia nominal de los conductores principales.

Estos ahorros de peso y reducciones de potencia contribuyen tanto a mejorar la eficiencia del combustible de la aeronave como a reducir el coste total, ya que el equipo de menor potencia puede proporcionarse a un precio reducido. También, la configuración del sistema de energía distribuida permite ahorrar costes de mantenimiento.

3 Técnicas de control y optimización de microrredes

En esta sección, se estudiarán diferentes técnicas de control y optimización aplicadas a microrredes aptas para su aplicación en tiempo real, poniendo en relieve sus ventajas y desventajas.

En la gestión de microrredes en aeronaves, el sistema de control debe garantizar el cumplimiento de dos tareas prioritarias para garantizar las operaciones vitales del EPS:

- De acuerdo con las reglas del sistema de gestión, se lleva a cabo una estrategia de reconfiguración utilizada por el sistema de control, la cual puede incluir normas de seguridad y calidad de la energía. Gracias a esta, se distribuye eficientemente la energía a lo largo del EPS cambiando la configuración eléctrica del sistema mediante el uso de interruptores y fuentes de alimentación disponibles.
- Encontrar la mejor solución a través de la asignación de carga en los buses, respetando la potencia disponible y los niveles de prioridad de cada carga.

3.1 Autómata finito

Un autómata finito o máquina de estado finito, conocido como *Finite-State Machine control* (FSM), es un modelo de computación que puede implementarse con hardware o software y puede utilizarse para simular la lógica secuencial. Los FSM pueden utilizarse para modelar problemas en muchos campos, como las matemáticas y la inteligencia artificial. En un FSM, el comportamiento del sistema puede modelarse como un conjunto de estados y transiciones entre estados, estos sistemas suelen conocerse como un sistema reactivo. De forma matemática, el FSM puede verse como:

$$f(\Sigma, S, s_0, \delta, F) \quad (3.1)$$

Donde Σ representa un conjunto finito de símbolos, S es un conjunto finito de estados, s_0 es el estado inicial, de forma que $s_0 \in S$, y δ es una función de transición de estados.

$$\delta : S \times \Sigma \rightarrow S \quad (3.2)$$

Además, F es el conjunto de estados finales. Un ejemplo de la formulación de un FSM se recoge en la Figura 3.1.

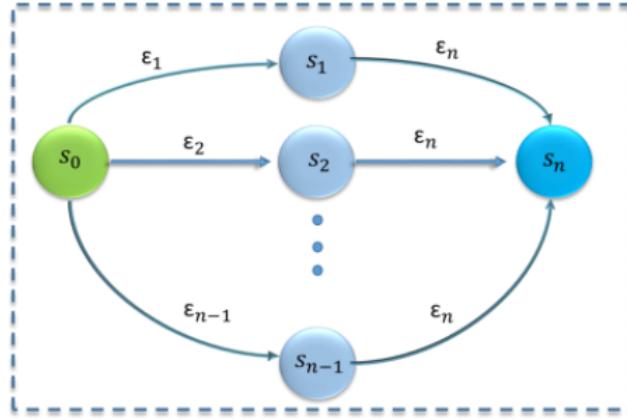


Figura 3.1 Ejemplo de sistema reactivo [21].

En el artículo [21], se adoptan máquinas de estado finito para modelar los EPS de CC de los MEA como un conjunto de estados y transiciones. Se desarrolla un controlador de supervisión basado en un autómata finito que aplica un conjunto de reglas predefinidas de gestión de potencia y de seguridad a un sistema eléctrico representativo de una aeronave que incluye una smartgrid de distribución de energía. En estos estudios, se definen varios conjuntos de reglas de seguridad para hacer frente a diferentes casos de fallo y garantizar el suministro ininterrumpido de energía a las cargas de alta prioridad. En concreto, los autores definen 3 estados y 3 sets de condiciones para modelar el controlador, el cual actúa sobre los convertidores y actuadores. Este modelo es aplicado a 3 casos de estudios a pesar de que en ninguno de ellos se simula una situación de falta de potencia

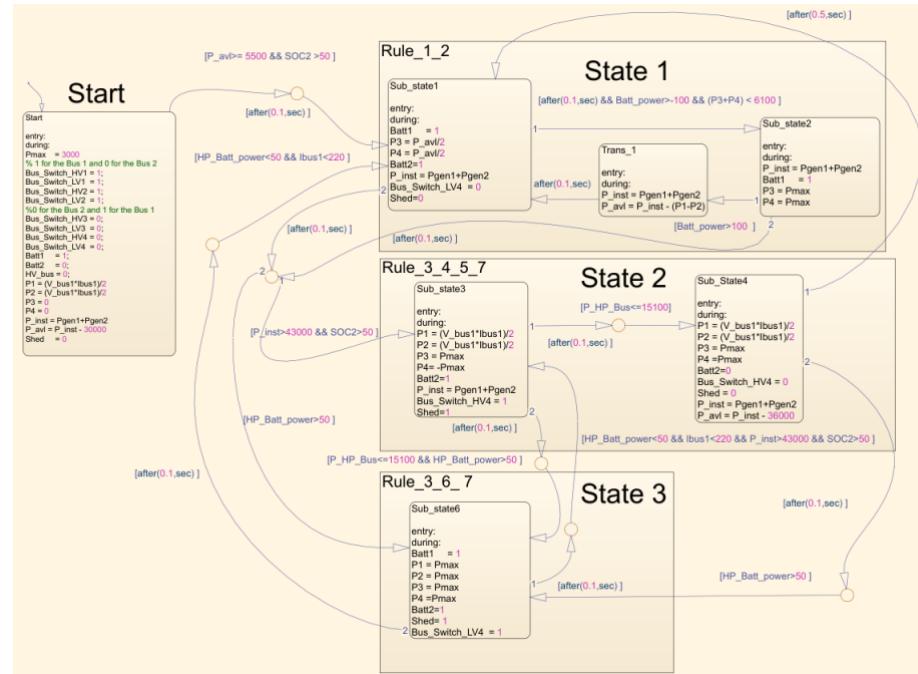


Figura 3.2 Modelo del FMS usado para programar la lógica del controlador de supervisión en [21].

Se demuestra que el controlador activa un estado particular del sistema en función de los eventos que pueden ocurrir en el sistema eléctrico captados a través de conjuntos predefinidos de condiciones, de manera que se respeten un conjunto de reglas acordadas para el sistema eléctrico.

Sin embargo, como conclusión, este método se basa en mediciones precisas para cambiar el estado de funcionamiento y es difícil integrar múltiples objetivos de funcionamiento. Además, el establecimiento de reglas óptimas puede ser difícil para sistemas complejos debido al gran número de estados del sistema.

3.2 Algoritmos heurísticos y metaheurísticos

Un algoritmo heurístico es un algoritmo diseñado para resolver un problema de una manera más rápida y eficiente que los métodos tradicionales, sacrificando la optimización, la exactitud, la precisión o la exhaustividad por la velocidad. Los algoritmos heurísticos se utilizan a menudo para resolver problemas NP-completos, una clase de problemas de decisión. En estos problemas, no se conoce una forma eficiente de encontrar una solución de forma rápida y precisa, aunque las soluciones pueden verificarse cuando se dan. La heurística puede producir una solución o utilizarse para proporcionar una buena base y complementarla con algoritmos de optimización. Los algoritmos heurísticos se emplean con mayor frecuencia cuando las soluciones aproximadas son suficientes y las soluciones exactas son necesariamente costosas desde el punto de vista computacional.

Los algoritmos metaheurísticos son algoritmos aproximados de optimización y búsqueda de propósito general. Son procedimientos iterativos que guían una heurística subordinada combinando de forma inteligente distintos conceptos para explorar y explotar adecuadamente el espacio de búsqueda.

Para obtener buenas soluciones, cualquier algoritmo de búsqueda debe establecer un balance adecuado entre dos características contrapuestas del proceso:

- Intensificación: cantidad de esfuerzo empleado en la búsqueda en la región actual (explotación del espacio).
- Diversificación: cantidad de esfuerzo empleado en la búsqueda en regiones distantes del espacio (exploración).

Diversos tipos de algoritmos heurísticos y metaheurísticos se utilizan ampliamente para resolver problemas de gestión de potencia y cargas en microrredes. Se analiza un ejemplo:

El artículo [22] presenta el diseño de un controlador de lógica difusa de baja complejidad de sólo 25 reglas para ser integrado en un sistema de gestión de potencia residencial conectada a la red que incluye fuentes de energía renovable y sistemas de almacenamiento. El sistema asume que ni la generación renovable ni la demanda de carga son controlables. El objetivo principal del diseño es minimizar las fluctuaciones del perfil de potencia de la red mientras se mantiene el estado de carga de la batería dentro de unos límites seguros, objetivos muy similares a los buscados en los MEA EPS del presente trabajo.

El diseño del controlador de lógica difusa se ha basado en dos pasos. El primero ha sido la elección de la tasa de cambio de potencia de la microrred y el SOC de la batería como entradas del controlador difuso para controlar la potencia entregada/absorbida por la red principal. El segundo paso ha sido un proceso de optimización offline basado en un conjunto de criterios de calidad de la evaluación para derivar todos los parámetros, es decir, las funciones de pertenencia y la base de reglas.

Este diseño permite que el EPS reaccione rápidamente ante los cambios de energía de la microrred para fijar el SOC de la batería en el 75 % de la capacidad nominal de la misma. En consecuencia, el rango dinámico disponible del SOC de la batería puede compensar las fluctuaciones de la potencia neta de la microrred y manteniendo el SOC dentro de unos límites seguros.

Sin embargo, el estudio realiza una optimización offline, la cual no es de utilidad para los objetivos de nuestro trabajo. Estos enfoques no son adecuados para la aplicación en tiempo real, ya que pueden caer en la trampa de las soluciones subóptimas y pueden seguir siendo computacionalmente intensivos.

3.3 Redes neuronales

En los últimos años, el desarrollo de la inteligencia artificial (IA) ha sido notable. Su importancia radica en la capacidad de representar o emular el conocimiento humano para memorizar, adquirir conocimientos, percibir y tomar decisiones inteligentes. Dentro de este campo, podemos encontrar la aplicación de las redes neuronales (NNs) a la identificación de sistemas, control de procesos, predicción, diagnóstico, etc. En el área de la electrónica de potencia, la aplicación de las NN ha logrado grandes mejoras, y el futuro parece prometedor.

Una red neuronal es un modelo simplificado que emula el modo en que el cerebro humano procesa la información: Funciona simultaneando un número elevado de unidades de procesamiento interconectadas que parecen versiones abstractas de neuronas.

Las unidades de procesamiento se organizan en capas. Hay tres partes normalmente en una red neuronal: una capa de entrada, con unidades que representan los campos de entrada; una o varias capas ocultas; y una capa de salida, con una o varias variables que representan los objetivos. Las unidades se conectan con fuerzas de conexión variables (pesos). Los datos de entrada se presentan en la primera capa, y los valores se propagan desde cada neurona hasta cada neurona de la capa siguiente. al final, se envía un resultado desde la capa de salida.

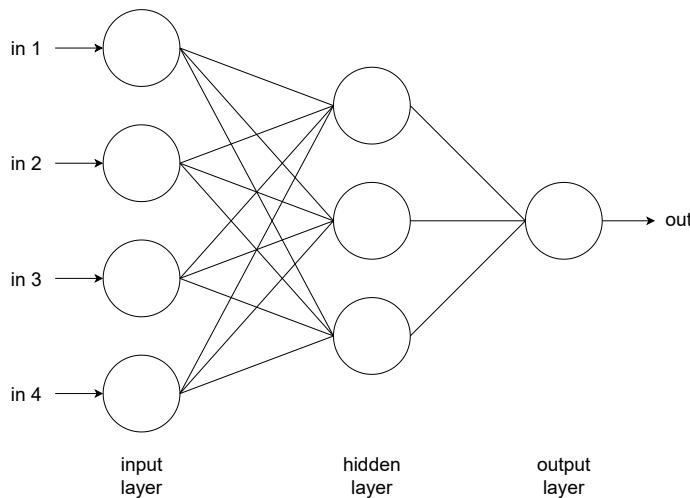


Figura 3.3 Arquitectura de una red neuronal simple [23].

La red aprende examinando los registros individuales, generando una predicción para cada registro y realizando ajustes a las ponderaciones cuando realiza una predicción incorrecta. Este proceso se repite muchas veces y la red sigue mejorando sus predicciones hasta haber alcanzado uno o varios criterios de parada.

Al principio, todas las ponderaciones son aleatorias y las respuestas que resultan de la red son, posiblemente, disparatadas. La red aprende a través del entrenamiento. Continuamente se presentan a la red ejemplos para los que se conoce el resultado, y las respuestas que proporciona se comparan con los resultados conocidos. La información procedente de esta comparación se pasa hacia atrás a través

de la red, cambiando las ponderaciones gradualmente. A medida que progresó el entrenamiento, la red se va haciendo cada vez más precisa en la replicación de resultados conocidos. Una vez entrenada, la red se puede aplicar a casos futuros en los que se desconoce el resultado.

En [24], se ha desarrollado y probado un sistema de gestión de energía muy eficiente para vehículos eléctricos híbridos (HEV), que utiliza redes neuronales. El sistema minimiza la necesidad de potencia del vehículo y puede funcionar con diferentes fuentes de energía primaria como pilas de combustible, microturbinas, baterías de zinc-aire u otras fuentes de energía con poca capacidad de recuperación de energía por medio de frenado regenerativo, o con escasa capacidad de potencia para aceleraciones rápidas.

El objetivo es obtener una implementación de la estrategia de control desarrollada por los autores en tiempo real, respaldada por las redes neuronales, cuya función es la de optimizar la función objetivo y mejorar el modelo en busca de una mayor eficiencia.

Como primer paso, se resolvió la optimización numéricamente del modelo del sistema para 30 conjuntos de datos de conducción urbana, obteniendo datos de entrada y salida.

A continuación, para definir la arquitectura ideal de la red neuronal, se utilizaron algoritmos de "prunning" o poda. La poda es una técnica de compresión de datos en aprendizaje automático y algoritmos de búsqueda que reduce el tamaño de los árboles de decisión al eliminar secciones del árbol que no son críticas y redundantes para clasificar instancias.

Por último, para validar la red, se analizó el error cuadrático medio. De 30 conjuntos de datos, 10 de ellos se utilizaron como datos de entrenamiento y 20 como datos de validación. La Figura 3.4 muestra la arquitectura de la red neuronal obtenida tras el algoritmo de poda.

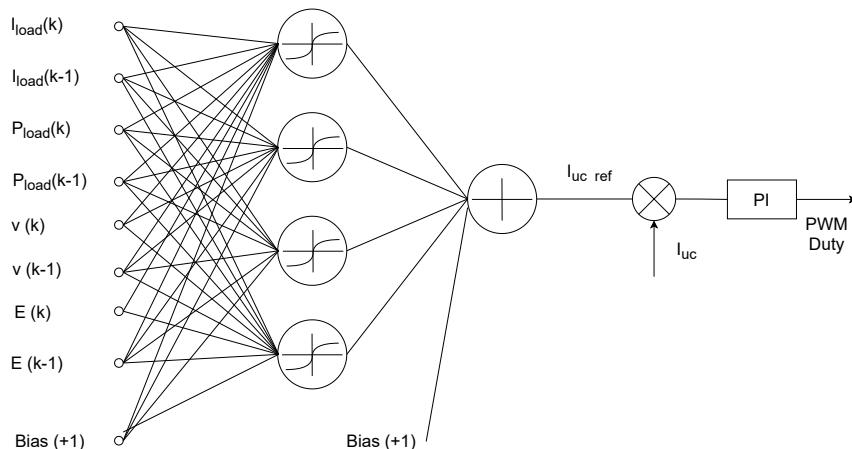


Figura 3.4 Red neuronal entrenada [24].

Cuando se instalaron ultracondensadores en el vehículo y se utilizó el control óptimo con NN, el aumento de la autonomía fue de hasta el 8,9 %. Se ha podido observar que las redes neuronales son lo suficientemente rápidas para la aplicación en tiempo real como se demuestra en diversos estudios, entre ellos el que acabamos de analizar. Sin embargo, necesitan entrenamiento, por lo que son menos capaces de hacer frente a situaciones inesperadas, y puede ser difícil verificar su fiabilidad para el uso en aviones. De hecho, poca literatura se ha encontrado acerca de la aplicación directa de redes neuronales al EPS de una aeronave, aunque si a tareas más concretas o prevención de fallos.

3.3.1 Combinación de redes neuronales con algoritmos heurísticos

El proceso de optimización necesita cierta predicción de los dispositivos no controlados. El uso de redes neuronales para la predicción del modelo y algoritmos heurísticos para la optimización del problema, es una de las técnicas más empleadas en el estado del arte para la gestión y optimización de microrredes. En [25], se presenta un método basado en dichas técnicas para optimizar el sistema de almacenamiento de energía en una microrred con cargas y generación fotovoltaica.

En primer lugar, se utilizan redes neuronales artificiales para obtener una predicción de las cargas y de la generación de potencia al comienzo de cada día. Estas reciben información según el día del calendario que sea para realizar la predicción y prever la demanda de potencia y la generación de la instalación fotovoltaica. Las redes neuronales están estructuradas en 3 capas, con 3 entradas para la información del calendario y 96 salidas para los valores de potencia predichos cada cuarto de hora del día. Para su entrenamiento y evaluación, se emplearon datos históricos de un año.

El algoritmo seleccionado para la optimización tras la predicción inicial es el algoritmo evolutivo PSO, o optimización por enjambre de partículas. Se trata de un algoritmo metaheurístico, ya que asume pocas o ninguna hipótesis sobre el problema a optimizar y puede aplicarse en grandes espacios de soluciones candidatas. El número máximo de iteraciones es de 200, el paso mínimo de las partículas es de 0,0001 y el cambio mínimo de la función objetivo es de 0,0001€/mes. De este modo, una vez que las iteraciones no tienen un impacto real en el coste anual de la electricidad, el método se detiene.

Tras la optimización, los valores obtenidos de las variables se implementan en un ciclo de decisión donde se comparan los valores de las mediciones con los deseados, modificando el comportamiento del sistema y llevando a cabo el control en tiempo real.

3.4 Control Predictivo Basado en Modelo

En este trabajo se desarrolla un controlador para la gestión óptima del flujo de potencia del sistema eléctrico de una aeronave por medio de la técnica Control Predictivo Basado en Modelo, o comúnmente conocido como *Model Predictive Control* (MPC). El MPC proporciona una estrategia de funcionamiento óptima gracias a la predicción de las trayectorias o estados futuros del sistema, prescindiendo de esta forma del análisis de los peores casos y evitando altos costes computacionales.

En la aplicación en un MEA, el controlador es capaz de lidiar con los estados futuros del sistema. Asimismo, las restricciones operativas y de acoplamiento temporal pueden integrarse en el modelo MPC. En la gestión de microrredes en aeronaves, el sistema de control debe garantizar el cumplimiento de dos tareas prioritarias para garantizar las operaciones vitales del EPS:

- De acuerdo con las reglas del sistema de gestión, se lleva a cabo una estrategia de reconfiguración utilizada por el sistema de control, la cual puede incluir normas de seguridad y calidad de la energía. Gracias a esta, se distribuye eficientemente la energía a lo largo del EPS cambiando la configuración eléctrica del sistema mediante el uso de interruptores y fuentes de alimentación disponibles.
- Encontrar la mejor solución a través de la asignación de carga en los buses, respetando la potencia disponible y los niveles de prioridad de cada carga.

3.4.1 Principios del MPC

El Control Predictivo Basado en Modelo (MPC), es una estrategia de optimización donde teniendo un modelo del sistema, se ejecutan un conjunto de predicciones futuras en un corto periodo de

tiempo del comportamiento del sistema para diferentes estrategias de actuación minimizando una función objetivo, optimizándose así las actuaciones de control u para alcanzar una señal de salida y deseada. De esta forma, se determina la próxima acción de control inmediata en función de esa optimización.

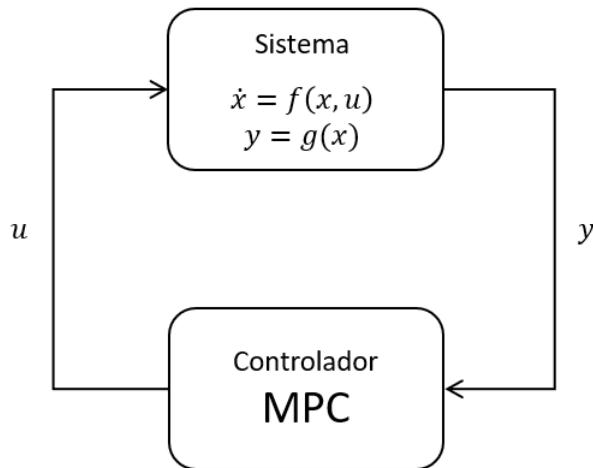


Figura 3.5 Diagrama de control retroalimentado.

Una vez aplicada en el sistema esa acción de control inmediata, se vuelve a iniciar de nuevo el proceso de optimización desplazándose la ventana del horizonte de control para calcular la siguiente acción del controlador. Este mecanismo, llamado *Moving Horizon Estimation* (MHE), se aplica en cada paso de tiempo, avanzando así en el tiempo. Gracias a ello, el controlador es dotado de una gran robustez frente a la incertidumbre.

Encontrándose el sistema en un estado y_k en un tiempo igual a k , se ejecuta el proceso de optimización en una pequeña ventana de tiempo, denominada *horizonte*, en la cual se optimiza las señales de entrada del sistema para alcanzar el objetivo deseado. Tras haber llevado a cabo una ejecución de la optimización en un horizonte concreto, se obtendrá el valor de las variables de entrada u , según la estrategia de optimización definida. A continuación, el controlador solo toma en consideración el primer valor de las acciones de control y aplica dicha entrada al sistema, obteniendo una respuesta del sistema y . Una vez que el controlador ha aplicado dicha acción de control, mueve el horizonte hacia delante desplazándose del paso k a $k+1$ (como se ejemplifica en la Figura 3.6), iniciándose de nuevo la optimización, siendo posible que el perfil de las variables de control u cambie en el tiempo. Este mecanismo es el que se conoce como MHE.

La estrategia de control puede expresarse como:

$$K(y_k) = u_{k+1}(y_k) \quad (3.3)$$

La cual expresa que la señal del controlador K en el instante y_k será igual a cual sea la acción de control u_{k+1} en el paso de tiempo y_k . En concreto, $u_{k+1}(y_k)$ básicamente representa la estrategia de control óptimo en un corto intervalo de tiempo empezando en y_k , para la cual solo se implementa en el controlador K la primera acción de control en el siguiente paso de tiempo $k+1$.

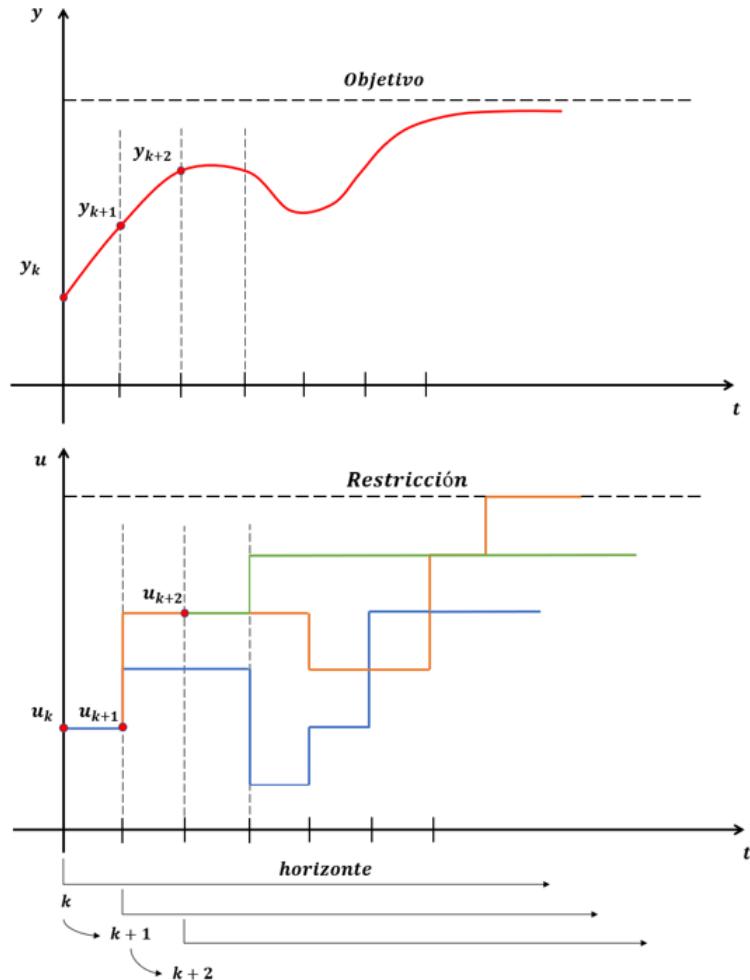


Figura 3.6 Esquema Model Predictive control (MPC).

Por tanto, la idea básica del MPC es optimizar las acciones de control para un número de pasos de tiempo, pero solo implementar la primera señal de entrada, $u_{k+1}(y_t)$, obteniendo como resultado $k(y_k)$, la acción del controlador en el paso de tiempo k . Después, el sistema cambia su estado a y_{k+1} , se inicia nuevamente la optimización para un horizonte completo de tiempo desde $k+1$ y se vuelve a implementar la acción de control en el primer paso de tiempo, y así sucesivamente, para obtener las señales de entradas más adecuadas según los nuevos estados del sistema.

El MPC es un conjunto de métodos que se diferencian entre sí por el tipo de modelo, la función de costes y el método de resolución. Dado que el almacenamiento es un componente importante de las microrredes, los modelos dinámicos de las microrredes suelen formularse como ecuaciones de espacio de estados en las que la variable de estado $x(t)$ coincide con el SOC de las unidades de almacenamiento de energía. Además, esta formulación puede tratar fácilmente con sistemas multivariados, que es el caso común en las microrredes. Las siguientes ecuaciones se utilizan en el caso lineal para capturar la dinámica del sistema.

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{3.4}$$

Una vez que se dispone de un modelo dinámico, se puede incluir en la función de costes y proceder a su minimización. Normalmente, el objetivo principal es que la salida futura $y(t)$ siga una determinada señal de referencia $w(t)$ a lo largo del horizonte, penalizando el esfuerzo de control $u(t)$ necesario para ello. La expresión general de la función objetivo es:

$$J(N_p, N_c) = \sum_{k=1}^{N_p} \delta u(k)[\hat{y}(t+k) - w(t+k)]^2 + \sum_{k=1}^{N_p} \lambda(k)[\Delta(t+k-1)]^2 \quad (3.5)$$

Donde \hat{y} es la predicción de la salida, N_p es el número total de pasos de tiempo que serán analizados en el problema, y N_c son los pasos de tiempo que constituyen la ventana del horizonte de control.

La característica que dota al MPC de una gran robustez, es la posibilidad de establecer límites o restricciones tanto en el estado del sistema, para no sobrepasar ciertos umbrales, como en las acciones de control. Estas limitaciones son muy importantes para evitar respuestas irreales del controlador, pudiéndose imponer restricciones duras y suaves.

$$u_{min} \leq u(t) \leq u_{max} \quad \forall t \quad (3.6)$$

$$u_{min} \leq u(t) - u(t-1) \leq \Delta u_{max} \quad \forall t \quad (3.7)$$

$$y_{min} \leq y(t) \leq y_{max} \quad \forall t \quad (3.8)$$

Este método, es mucho más flexible que otros métodos de control lineales ya que si el sistema empieza a moverse, desviarse o a hacer en general aquello que no consideramos adecuado, la optimización en cada paso de tiempo vuelve a calcular la solución óptima.

En cuanto a las desventajas que podemos encontrar, una de ellas es la dependencia de ordenadores y hardware potentes, debido al hecho de realizar esta optimización en bucle para cada paso de tiempo. En otros controladores, se lleva a cabo la optimización offline y después se toman esos resultados para introducirlos en un sistema de control online. En el MPC se necesita gran poder computacional para ejecutar y resolver estas optimizaciones en poco tiempo y aplicar el primer resultado de la optimización en el horizonte inmediatamente en el sistema, y adecuar las respuestas del controlador si el sistema comienza a desviarse de los valores adecuados o las dinámicas varían.

Habitualmente en el MPC, se aplica un sistema linealizado de ecuaciones de movimiento, debido a que el tiempo de cálculo y resolución es menor. Sin embargo, si se tiene un sistema no lineal, es posible modelar la linealización de las dinámicas del sistema en todos los puntos para introducirlos en un optimizador lineal. Por otro lado, gracias a que cada vez los ordenadores son más rápidos, también es factible optimizar directamente un sistema de ecuaciones no lineales. Además, aunque con estas técnicas la formulación matemática resulte más complicada, resultan mucho más rápidas que otras estrategias de control, como la descrita en [25], la cual usa redes neuronales y algoritmos evolutivos.

3.4.2 Aplicaciones del MPC

Recientemente, el Control Predictivo Basado en Modelo, ha llamado la atención de la comunidad de sistemas de energía para el control y la gestión de la operación de las microrredes.

La utilización de esta técnica está ampliamente extendida y pueden encontrarse en numerosos proyectos tanto para sistemas lineales como no lineales. Su aplicación está contemplada en sistemas de diversa índole:

- En [26], se discute el diseño de un controlador MPC para un reactor de tanque agitado continuo (CSTR), sistema químico utilizado en una amplia variedad de aplicaciones industriales que necesitan la adición y eliminación de reactivos y productos de forma continua.
- Generar diversos comportamientos con un robot humanoide requiere una mezcla de supervisión humana y control automático. Lo ideal es que la aportación del usuario se limite a instrucciones y orientaciones de alto nivel, y que el controlador sea lo suficientemente inteligente como para realizar las tareas de forma autónoma. En [27], se describe un sistema integrado que logra este objetivo por medio de un controlador automático basado MPC en tiempo real aplicado a la dinámica completa del robot.

En este apartado se analizarán diversos estudios en los cuales se ha empleado MPC para abordar alguna cuestión de control y optimización en microrredes. Se comienza revisando artículos enfocados en microrredes en tierra y coches eléctricos.

El artículo [28] estudia la minimización del consumo de combustible de los vehículos eléctricos híbridos (HEV) utilizando MPC. El controlador que se presenta calcula una secuencia óptima de entradas de control en la planta del vehículo a lo largo del horizonte de predicción basado en la demanda de par estimada a partir de la velocidad deseada del vehículo y el estado de carga (SOC) deseado de la batería. Para validar el controlador MPC presentado, se desarrolla el modelo dinámico lineal del Toyota Prius 2004, un vehículo eléctrico híbrido. Los resultados de la simulación muestran la viabilidad del uso del controlador MPC para mejorar el rendimiento del vehículo y minimizar el consumo de combustible.

Un novedoso entorno de trabajo MPC para microrredes interconectadas para distritos urbanos que comparten un recurso energético distribuido se presenta en [29]. El funcionamiento de las microrredes, junto con el recurso compartido, se coordina de forma que se consiga un objetivo común. El marco propuesto es flexible y puede manejar microrredes con diferentes capacidades de generación local y requisitos de energía, características técnicas y operativas. Se señala que, aunque el marco propuesto es escalable y puede coordinar un número arbitrario de microrredes, el tiempo computacional depende del tiempo de resolución de los problemas MILP individuales.

Debido al uso ampliamente extendido del MPC como estrategia de control para la gestión de microrredes, los científicos lo trasladaron a los sistemas eléctricos de potencia de las aeronaves MEA. Los requisitos operativos y de suministro en cualquier situación que ocurra en el avión pueden ser implementados en el controlador MPC, y gracias a su habilidad para corregir y anticiparse a eventos futuros por medio del horizonte móvil, puede adaptarse, por ejemplo, a momentos de picos de potencia.

En el trabajo desarrollado en [30], se ha propuesto un MPC no lineal para minimizar el consumo de combustible, el consumo de potencia y las emisiones durante una misión de vuelo. El EMS óptimo se obtiene en base a la relación propulsión/aerodinámica, utilizando un MPC no lineal junto al método de entropía cruzada (CEM). Para lograr un adecuado balance de potencias se establece una restricción en la que la potencia a bordo sea igual a la demanda de potencia de la misión. Así, evita la penalización por arrastre causada por una cantidad excesiva de potencia a bordo.

Una estrategia de optimización para el control integrado y coordinado de dos generadores eléctricos conectados a un mismo motor de turbina de gas en un avión se propone en [8]. La arquitectura de control combina un mapa de reparto de potencia entre los generadores, basado en la minimización del consumo de combustible y un controlador MPC, encargado de ajustar la relación combustible/aire de entrada al motor y los requisitos de potencia de los dos generadores. El controlador responde a las peticiones de empuje y potencia eléctrica, manejando las restricciones de estado y control para proteger los componentes del sistema. Los resultados obtenidos demuestran

la capacidad del controlador para cumplir con las restricciones del modelo y alcanzar el objetivo definido. Esta configuración puede ser ventajosa para los futuros MEA y AEA, que tienen que satisfacer mayores requerimientos de empuje y potencia eléctrica en estado estacionario y transitorio.

En [31], aplica un entorno de trabajo de modelado basado en gráficos y control predictivo jerárquico para el sistema de propulsión de un UAV eléctrico híbrido. Queda reflejada la capacidad de planificación y coordinación simultánea de dinámicas rápidas y lentas del sistema, como la corriente de la batería y el SOC. Los principales resultados mostraron que los diseños de control avanzados permitieron reducir el consumo de combustible en aproximadamente un 10 %, mejorando el rendimiento y la fiabilidad del sistema, contribuyendo de esta forma a una reducción de la huella de carbono en las aeronaves.

En los trabajos (o mayoría de trabajos) presentes en el estado del arte no se pone demasiado énfasis a una de las ventajas más significativas del MPC, como es la capacidad de cumplir con varios objetivos de control al mismo tiempo para hacer el controlador más completo y con objetivos reales. Se considera que es posible explotar aún más las capacidades del MPC en este sentido y se tratará de ponerlo en relieve en la estrategia de control desarrollada en la sección 4.

Una de las líneas de investigación principales o mejoras es investigar en la variación de los parámetros como la longitud del horizonte o los pesos de las funciones de coste para priorizar ciertos objetivos de funcionamiento. Más adelante en este trabajo, se estudiará cómo varían las respuestas del sistema según la longitud del horizonte de predicción. Investigar cómo estos grados de libertad pueden afectar al rendimiento del sistema de control ayudará a decidir mejor los ajustes del MPC para satisfacer los requisitos del sistema.

4 Modelado de la estrategia de optimización

En este trabajo, se diseñará la estrategia de optimización que sería implementada en el controlador para resolver el problema de gestión óptima de flujo de potencia multiobjetivo en un MEA EPS. Basado en un modelo MPC, el objetivo que se plantea es la gestión inteligente de los sistemas de almacenamiento de energía, y la explotación de la flexibilidad en el lado de la demanda, al tiempo que se cumplen los requisitos variables en el tiempo de potencia consumida por las cargas.

El controlador MPC debe procurar que siempre haya energía en el ESS para utilizarla cuando fuera necesario o conveniente, con la menor cantidad de variaciones entre los estados carga y descarga, para reducir los transitorios y prolongar el ciclo de vida de la batería. Por otro lado, también es responsable de minimizar las desconexiones de las cargas con el menor número de cambios de estados de los contactores entre abiertos y cerrados, para proteger nuevamente los equipos. Estos criterios de operación están presentes en la formulación de la función objetivo, por lo que es de especial importancia prestar atención a su correcto planteamiento. Sin embargo, esta misma flexibilidad en la formulación de la función objetivo, aumenta la complejidad del diseño debido a que hay que modelar de manera correcta los objetivos o restricciones necesarios a cumplir

Para la resolución de la optimización, el modelado del problema se aborda mediante la definición de un problema MILP debido a las características de dicho modelo. Esta formulación del modelo incluye tanto las funciones objetivo como las restricciones del sistema relacionadas con el flujo de potencia del sistema, la conversión de energía y las restricciones de conexión.

A continuación, se describirá la arquitectura del EPS sobre el que se trabaja y enumerando los parámetros presentes en el problema.

4.1 EPS del modelo

Normalmente en una aeronave, la distribución del EPS es simétrica respecto a los lados izquierdo y derecho del avión. Por este motivo, para simplificar el modelado y el estudio de la microrred analizaremos solo la arquitectura de un lado de la aeronave. Este modelo solo se centra en la gestión del sistema de energía y de las cargas en el bus de baja tensión. En la figura 4.1, se muestra la arquitectura del EPS bajo estudio.

Se asume una arquitectura de bus único, explicada en 2.2.2.2. Como se observa en la Figura 4.1, se muestra más detalladamente la configuración del bus LVDC. La fuente principal, compuesta por un solo generador conectado al bus HVDC, alimenta al bus de baja tensión donde están conectadas

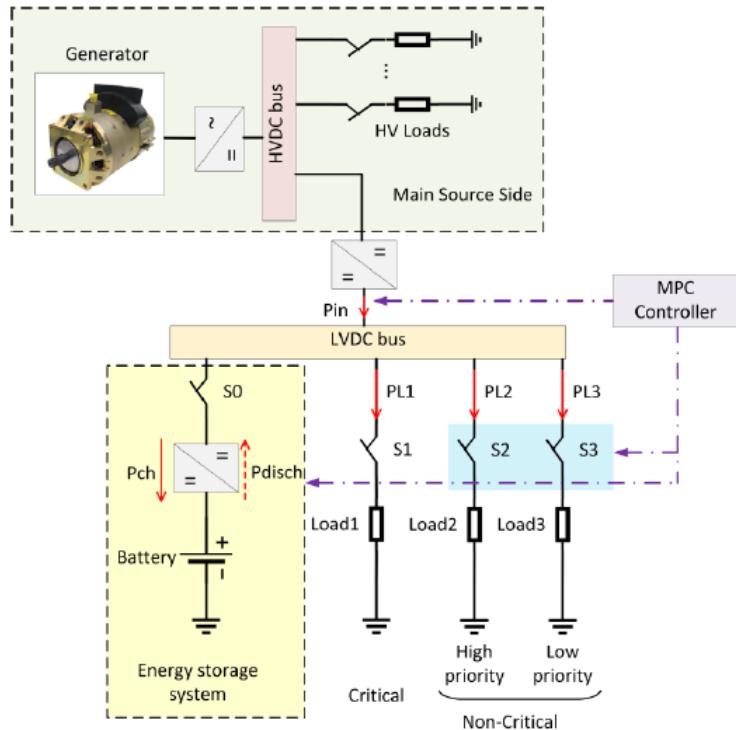


Figura 4.1 Arquitectura del EPS bajo estudio [1].

las cargas a través de sus correspondientes contactores y el ESS. Por un lado, el EES está formado por una batería, un convertidor bidireccional CC/CC y un contactor. Por otro lado, se consideran tres tipos de cargas clasificadas como:

- Load1 (críticas), que deben ser alimentadas bajo cualquier circunstancia.
- Load2 (alta prioridad), las cuales se deberían de tratar alimentar siempre, pero pueden ser desconectadas si es necesario.
- Load3 (baja prioridad), que se desconectarán para salvaguardar las de alta prioridad.

En las tablas 4.1, 4.2, 4.3, se muestra la nomenclatura de los diferentes parámetros y variables utilizados para modelar el problema.

Por último, se encuentra el controlador MPC, responsable de calcular las variables de control P_{in} , P_{ch} , P_{disch} y decidir la conexión o desconexión de las cargas no críticas, S_{Li} , para optimizar los objetivos impuestos en el sistema y garantizar el balance de energía.

Cabe destacar que el método propuesto para la gestión óptima de la microrred puede ser aplicado a otras arquitecturas de EPS que incluyan, por ejemplo, el lado de alta tensión o partes con corriente alterna, también muy comunes en aeronaves.

4.2 Descripción de MILP-MPC

El controlador trabaja a nivel de sistema y se conecta al EPS para recibir datos sobre el estado del sistema y dar órdenes como datos de salida. Debido a la naturaleza del problema, que se trata de simular el vuelo real de una aeronave, se establece una escala de tiempo lenta de orden de 1 minuto para optimizar el rendimiento a largo plazo del sistema. Como puede verse en la Figura 4.2, el controlador recibe la información actualizada del sistema al principio de cada ciclo de control

Tabla 4.1 Parámetros.

Parámetros		Parámetros	
k	Intervalos de tiempo	H	Horizonte de predicción (s)
T_S	Tiempo de sampleo (s)	T	Tiempo total de simulación (s)
η_{ch}/η_{disch}	Eficiencia de carga/descarga de la batería	B_{cap}	Capacidad de la batería (kWh)
$p_{ch}^{max}/p_{disch}^{max}$	Potencia máxima de carga/descarga (kW)	LO/HI	Límites superior/inferior del SOC de la batería
p_{in}^{max}	Potencia máxima de entrada (kW)	p_{Li}^{shed}	Potencia de la carga no crítica i
γ_{Li}	Prioridad de la carga no crítica i (kW)	N_{Li}	Número de cargas no críticas
$p_{Li}^{nonshed}(k)$	Potencia de la carga crítica (kW)		

Tabla 4.2 Variables continuas.

Variables continuas	Variables continuas		
$p_{in}(k)$	Potencia de entrada (kW)	$SOC(k)$	Estado de carga de la batería
$p_{ch}(k)$	Potencia de carga (kW)	$p_{disch}(k)$	Potencia de descarga (kW)

Tabla 4.3 Variables Binarias.

Variables binarias	
$S_{Li}(k)$	Estado del interruptor de la carga no crítica i
$\zeta_{ch}(k)$	Indicador de carga de la batería
$\zeta_{disch}(k)$	Indicador de descarga de la batería

definido por el tiempo de muestreo deseado (1 minuto). Esta información incluye las restricciones, el estado de las baterías y la predicción de las demandas de carga y las prioridades asociadas.

El problema se modela como un *Mixed-Integer Linear Programming*. En un MILP, la función objetivo, las restricciones y los límites de las variables deben estar compuestas por expresiones lineales. Además, estas variables pueden ser tanto continuas como discretas, haciendo que algunas de ellas solo adopten valores enteros. En este caso, se declara la variable binaria $S_{Li}(k)$ para representar el estado de los interruptores, de forma que, si la variable es igual a 1, significa que interruptor está encendido y la carga correspondiente conectada, mientras que si es igual a 0, el interruptor está apagado y la carga desconectada. El resto de variables son continuas, de ahí proviene el término *mixed-integer*.

El MILP representa un enfoque de modelización matemática eficaz para resolver tareas de optimización complejas e identificar las posibles compensaciones entre los objetivos en conflicto, lo que puede proporcionar una mejor comprensión de los sistemas y apoyar a los responsables de la toma de decisiones en la elaboración de las vías sostenibles hacia los objetivos.

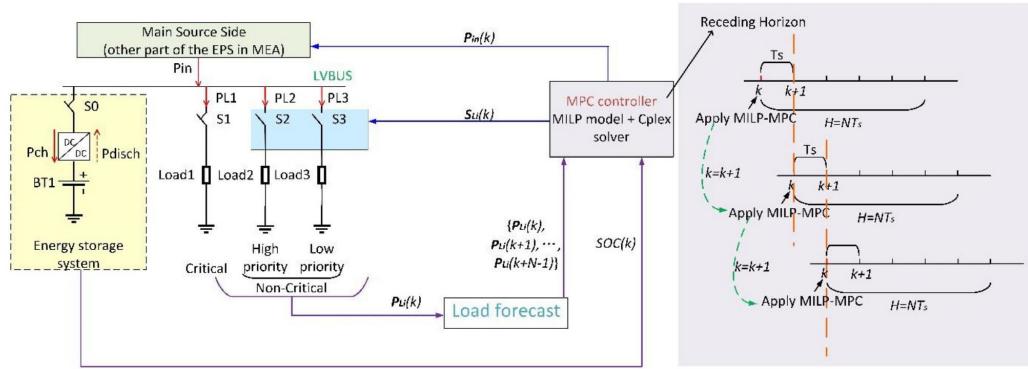


Figura 4.2 Optimización online MILP-MPC con horizonte móvil [1].

El núcleo de la estrategia MPC es un modelo del sistema bajo control para predecir la evolución de la salida del sistema en el horizonte deseado. Un problema individual de optimización online se resuelve para cada paso de tiempo k , $k = 0, 1, \dots, N - 1$, sobre el horizonte de predicción finito H , el cual indica el tiempo de anticipación en el futuro para evaluar la función objetivo. Puede calcularse como $H = NT_s$.

A parte del estado en un preciso instante, el controlador MPC trabaja tomando como datos las predicciones futuras del sistema y aportando una respuesta para toda la ventana de tiempo en base al esquema del problema de un horizonte móvil representado en la Figura 4.2. Sin embargo, solo la acción de control del primer muestreo del horizonte se aplica al sistema, y el problema de control se vuelve a lanzar para el paso de tiempo $k + 1$. La integración de los objetivos y las restricciones del modelo se plantearán en la próxima sección.

Por tanto, se requiere una predicción demanda de potencia de las cargas a lo largo del horizonte para actualizar el modelo según los datos más recientes del sistema. En este trabajo, la predicción se realiza observando un perfil de cargas muestreado, sin tener en cuenta los cambios de demanda en vuelo en tiempo real, ni las desviaciones entre la carga predicha y los valores reales con el objetivo de simplificar el análisis. Dicha demanda podría ser precedida con métodos de Inteligencia Artificial, como Machine Learning, que obtengan una predicción de la potencia requerida en base a las etapas de vuelo, la altura y los datos históricos.

A continuación, se explica detalladamente el algoritmo MILP-MPC:

1. Actualiza el sistema al paso k
 - a) Inicializa el modelo con las variables iniciales.
 - b) Predice en el horizonte H la demanda de potencia de las cargas para cada sampleo.
2. Se lleva a cabo la optimización del modelo y el controlador calcula las acciones de control para todo el horizonte, basándose en el estado del sistema actual y las predicciones futuras. Las acciones de control incluyen la conexión o desconexión de las cargas, la potencia introducida en el LVDC bus, P_{in} y la carga o descarga de la batería.
3. Aplica el resultado de la optimización solo en el primer paso del sampleo k .
4. El sistema avanza un paso, estableciendo $k = k + 1$, y se repite el proceso desde el nuevo estado.

4.3 Modelado del sistema

Para el diseño de la estrategia de optimización, es necesario realizar un modelo matemático del sistema. Se debe expresar mediante ecuaciones, cómo funciona e interactúa el sistema, para establecer el comportamiento, los objetivos operacionales y las restricciones de este.

4.3.1 Definición funciones objetivos

La función objetivo será igual a la suma total de distintas funciones de coste. Cada una de estas funciones de coste representa uno de los objetivos de control que se desean minimizar en el problema, y por tanto tendrán significados y expresiones diferentes.

Aprovechando, la flexibilidad de la expresión de la función objetivo, se formulan dos funciones objetivos distintas para cumplir con los criterios de control y analizar la presencia de diferentes funciones de coste en el rendimiento del sistema.

4.3.1.1 Función objetivo 1

La primera función objetivo propuesta, $Obj1$, se centra en la conexión de las cargas, los estados de los conectores y la potencia cargada/descargada de la batería. Esta se compone de la suma de las funciones de coste (4.2)-(4.5) ponderadas por sus correspondientes pesos $\omega_S, \omega_\delta, \omega_{P_{ch}}, \omega_{P_{disch}}$. La selección del valor de los pesos se detallará en la próxima sección.

$$\text{Minimize} \quad Obj1 = \omega_S J_{S_{Li}} + \omega_\delta J_{\delta_{Li}} + \omega_{P_{ch}} J_{P_{ch}} + \omega_{P_{disch}} J_{P_{disch}} \quad (4.1)$$

donde

$$J_{S_{Li}} = \sum_{k=0}^{N-1} \sum_{i=1}^{N_{Li}} \gamma_{Li}(1 - S_{Li}(k)) / N \sum_{i=1}^{N_{Li}} \gamma_{Li} \quad (4.2)$$

$$J_{\delta_{Li}} = \sum_{k=0}^{N-1} \sum_{i=1}^{N_{Li}} |S_{Li}(k+1) - S_{Li}(k)| / N \cdot N_{Li} \quad (4.3)$$

$$J_{P_{ch}} = \sum_{k=0}^{N-1} (P_{ch}^{max} - P_{ch}(k)) / N P_{ch}^{max} \quad (4.4)$$

$$J_{P_{disch}} = \sum_{k=0}^{N-1} P_{disch}(k) / N P_{disch}^{max} \quad (4.5)$$

La ecuación (4.2) representa la penalización de la desconexión de las cargas, tratando de conseguir el máximo tiempo de vuelo con los tres grupos de cargas conectados. γ_{Li} es la prioridad de la carga no crítica i , S_{Li} es el estado del interruptor de la carga no crítica i , y N es la longitud del horizonte.

Mientras que la ecuación (4.3) representa la minimización los cambios de estado de los contactores para cada intervalo de tiempo, evitando una repetida conexión y desconexión de las cargas en periodos breves de tiempo. N_{Li} es el número de cargas no críticas. Esta función incluye un valor absoluto, el cual es no lineal, por lo que se debe linealizar para conseguir que el modelo sea un MILP, cuya resolución es más sencilla. Las linealizaciones correspondientes se explicarán en el apartado 4.3.3.

La preferencia de la carga de la batería se modela en la función (4.4). Esta minimiza la diferencia entre la potencia de carga máxima, P_{ch}^{max} y la potencia de carga en cada instante, $P_{ch}(k)$.

Por último, en la ecuación (4.5) se penaliza la descarga de la batería, minimizando el valor de la potencia de descarga en cada instante, $P_{disch}(k)$.

Variando los valores de los pesos de las funciones de coste, se pueden obtener desempeños diferentes del sistema: si $\omega_S > \omega_{Pch}$, el sistema priorizará mantener más tiempo las cargas conectadas, cargando lentamente la batería o incluso descargándola más a menudo; en cambio si $\omega_{Pch} > \omega_S$, sucederá al contrario y las cargas se desconectarán más frecuentemente para priorizar la carga de la batería.

En modelado del problema se ha optado por fijar los valores máximos y mínimos del SOC, sin dar opción al sistema a vulnerar mínimamente estos límites, a diferencia del modelado en [1].

4.3.1.2 Función objetivo 2

La segunda función objetivo propuesta, $Obj2$, es la misma $Obj1$ a la que se le añaden dos nuevas funciones de coste J_{SOC} , J_{Pbatt} multiplicadas por sus correspondientes pesos ω_{SOC} , ω_{Pbatt} :

$$\text{Minimize } Obj2 = \omega_S J_{S_{Li}} + \omega_\delta J_{\delta_{Li}} + \omega_{Pch} J_{P_{ch}} + \omega_{Pdisch} J_{P_{disch}} + \omega_{SOC} J_{SOC} + \omega_{Pbatt} J_{P_{batt}} \quad (4.6)$$

La primera función de coste añadida, J_{SOC} , representa al SOC acumulado de la batería:

$$J_{SOC} = \sum_{k=0}^{N-1} |HI - SOC(k+1)| / (N \cdot HI) \quad (4.7)$$

Fomenta que el SOC de la batería esté lo más cargado posible para estar preparado en caso de necesidad. Cuanto más grande sea el horizonte elegido, más se acelerará la carga, ya que el SOC es una variable acumulativa en el tiempo.

La segunda función de coste añadida, J_{Pbatt} , representa los cambios de estado de la batería entre carga y descarga:

$$J_{Pbatt} = \sum_{k=0}^{N-1} |(P_{ch}(k+1) - P_{disch}(k+1)) - (P_{ch}(k) - P_{disch}(k))| / N(P_{ch}^{max} + P_{disch}^{max}) \quad (4.8)$$

Esta función de coste se concibe para prolongar la vida útil de la batería, reduciendo las variaciones entre carga y descarga, e incluso la variación de la propia potencia de carga o de descarga. Esta expresión también necesita ser linealizada.

4.3.2 Restricciones del modelo

En este apartado se describen las restricciones asociadas al modelo. Como se ha comentado anteriormente, las dinámicas del sistema se modelan como restricciones para asegurar el correcto funcionamiento y no obtener soluciones irreales o fuera de las capacidades del sistema.

1. Balance de potencia

Atendiendo a la ley de Kirchhoff, la suma de la potencia generada más la consumida en el bus LVDC tiene que ser igual a 0. En el modelado de esta restricción, el efecto de las pérdidas no

se contempla de forma directa o específica. En el sistema bajo estudio de la Figura 4.1, los únicos generadores de potencia se corresponden con la fuente principal y la batería cuando se encuentre en descarga. Por tanto, la restricción puede ser formulada como:

$$P_{in}(k) + P_{disch} = P_{ch} + \sum_i S_{Li}(k)P_{Li}^{shed}(k) + \sum_j P_{Lj}^{nonshed}(k) \quad (4.9)$$

2. Actualización del estado de carga de la batería

El SOC de la batería puede ser calculado a partir de la cantidad de potencia cargada/descargada en el tiempo:

$$SOC(k+1) = SOC(k) + T_S \cdot \eta_{ch} \cdot \frac{P_{ch}(k)}{B_{cap}} - T_S \cdot \frac{P_{disch}(k)}{\eta_{disch} \cdot B_{cap}} \quad (4.10)$$

3. Límites máximos y mínimos del SOC

El SOC puede tomar valores entre 0 y 1 ($SOC \in [0,1]$), donde $SOC = 1$ indica que la batería está totalmente cargada y $SOC = 0$ corresponde a la batería totalmente agotada. A pesar de ello, en una aeronave nunca es buena idea tener la batería totalmente descargada, por si es necesario un aporte extra de emergencia en alguna situación de emergencia. También es conveniente dejar un pequeño margen antes de la capacidad total para no evitar problemas de sobrecarga. Por ello, se definen límites superiores e inferiores [$LO = 0.3, HI = 0.9$] para los valores del SOC.

$$LO \leq SOC(k) \leq HI \quad (4.11)$$

4. Modelado de carga y descarga

Se declaran dos variables binarias para controlar el estado de la batería y evitar la carga y la descarga del sistema al mismo tiempo.

$$\zeta_{ch}(k) + \zeta_{disch}(k) \leq 1 \quad (4.12)$$

$$0 \leq P_{ch}(k) \leq \zeta_{ch}(k) \cdot P_{ch}^{max} \quad (4.13)$$

$$0 \leq P_{disch}(k) \leq \zeta_{disch}(k) \cdot P_{disch}^{max} \quad (4.14)$$

La ecuación 4.12 representa la restricción asociada a que no se puede producir de forma simultánea la carga y descarga. De esta forma, para que se cumpla la restricción, una de las dos variables tiene que ser igual a 0 en cada paso de tiempo k . Mientras que las ecuaciones 4.13 y 4.14 representan los valores máximos de carga y descarga respectivamente.

5. Límites de la potencia de entrada desde la fuente principal

La potencia de entrada, P_{in} , está limitada por la máxima potencia que puede ser convertida por los convertidores CC/CC:

$$0 \leq P_{in}(k) \leq P_{in}^{max} \quad (4.15)$$

4.3.3 Linealización de funciones no lineales con valor absoluto

Para la linealización de las funciones 4.3 y 4.8, se ha llevado a cabo el método propuesto en [32]. En la formulación de las funciones objetivo, se encuentran dos funciones de coste con valores absolutos. Si una función de valor absoluto contiene una función no lineal, como es son estos casos, entonces el valor absoluto se puede reformular en dos expresiones lineales.

Si se tiene la siguiente función objetivo:

$$\text{Min} \quad |f(x)| \quad (4.16)$$

se puede reformular como:

$$\begin{aligned} \text{Min} \quad & z \\ \text{s.t.} \quad & z \geq f(x) \\ & z \geq -f(x) \end{aligned} \quad (4.17)$$

De este modo, las linealizaciones correspondientes a las funciones de costes indicadas se muestran a continuación:

(4.3) $J_{\delta_{Li}}$:

$$\text{Min} \quad \sum_{k=0}^{N-1} \sum_{i=1}^{N_{Li}} |S_{Li}(k) - S_{Li}(k-1)| / N \cdot N_{Li} \quad (4.18)$$

$$\begin{aligned} \text{Min} \quad & \sum_{k=0}^{N-1} \sum_{i=1}^{N_{Li}} J_{\delta_{LIN}}(k) / N \cdot N_{Li} \\ \text{s.t.} \quad & J_{\delta_{LIN}}(k) \geq S_{Li}(k) - S_{Li}(k-1) \\ & J_{\delta_{LIN}}(k) \geq S_{Li}(k-1) - S_{Li}(k) \end{aligned} \quad (4.19)$$

(4.8) $J_{P_{batt}}$:

$$\text{Min} \quad \sum_{k=0}^{N-1} |(P_{ch}(k+1) - P_{disch}(k+1)) - (P_{ch}(k) - P_{disch}(k))| / N(P_{ch}^{\max} + P_{disch}^{\max}) \quad (4.20)$$

$$\begin{aligned} \text{Min} \quad & \sum_{k=0}^{N-1} J_{P_{battLIN}}(k) / N(P_{ch}^{\max} + P_{disch}^{\max}) \\ \text{s.t.} \quad & J_{P_{battLIN}}(k) \geq (P_{ch}(k+1) - P_{disch}(k+1)) - (P_{ch}(k) - P_{disch}(k)) \\ & J_{P_{battLIN}}(k) \geq (P_{ch}(k) - P_{disch}(k)) - (P_{ch}(k+1) - P_{disch}(k+1)) \end{aligned} \quad (4.21)$$

4.4 Implementación del modelo

Para la implementación del algoritmo, se ha utilizado Pyomo, un paquete de software de código abierto basado en Python, usado para resolver modelos de optimización en programación lineal. Una de las capacidades principales de Pyomo es el modelado de aplicaciones de optimización estructuradas. Puede utilizarse para definir problemas simbólicos generales, crear instancias de problemas específicos y resolver estas instancias utilizando solvers comerciales y de código abierto. Los objetos de modelado de Pyomo están descritos dentro de un lenguaje de programación de alto nivel con todas las funciones que proporciona un rico conjunto de bibliotecas de apoyo, lo que distingue a Pyomo de otros lenguajes de modelado algebraico como AMPL, AIMMS y GAMS. El solver empleado para resolver el problema en cada paso de tiempo se trata de CPLEX.

A continuación, se muestran fragmentos de código para ilustrar la programación de las funciones objetivo con sus funciones de costes, además de las restricciones, las cuales incluyen las linealizaciones necesarias.

La primera función objetivo se define de la siguiente forma:

Código 4.1 Primera función objetivo.

```

1 def obj1_expression(mpc):
2
3     J_S_L = sum(mpc.gamma_L[i] * (1 - mpc.S_L[i,k]) / (mpc.N * (mpc.
4         gamma_L[0]+mpc.gamma_L[1])) for k in mpc.k for i in mpc.i)
5
6     J_delta_L = sum( mpc.lineal[i,k] / (mpc.N * mpc.N_L) for k in mpc.k
7         for i in mpc.i)
8
9     J_P_ch = sum((mpc.P_ch_max() - mpc.P_ch[k]) / (mpc.N() * mpc.
10        P_ch_max()) for k in mpc.k)
11
12    J_P_disch = sum(mpc.P_disch[k] / (mpc.N * mpc.P_disch_max) for k in
13        mpc.k)
14
15    return (mpc.omega_S*J_S_L + mpc.omega_delta*J_delta_L + mpc.
16        omega_P_ch*J_P_ch + mpc.omega_P_disch*J_P_disch)
17
18 mpc.obj1 = pyo.Objective(expr = obj1_expression)

```

En un modelo diferente, se define la segunda función objetivo:

Código 4.2 Segunda función objetivo.

```

1 def obj2_expression(mpc):
2
3     J_S_L      = sum((mpc.gamma_L[i] * (1 - mpc.S_L[i,k])) / (mpc.N *
4         mpc.gamma_L[0]+mpc.gamma_L[1])) for k in mpc.k for i in mpc.i)
5
6     J_delta_L = sum(mpc.lineal_J_S[i,k] / (mpc.N * mpc.N_L) for k in
7         mpc.k for i in mpc.i)

```

```

7         J_P_ch      = sum((mpc.P_ch_max() - mpc.P_ch[k]) / (mpc.N() * mpc.
8             P_ch_max()) for k in mpc.k)
9
9         J_P_disch = sum(mpc.P_disch[k] / (mpc.N * mpc.P_disch_max) for k
10            in mpc.k)
11
11        J_SOC      = sum((mpc.HI - mpc.SOC[k]) / (mpc.N * mpc.HI) for k
12            in mpc.k)
13
13        J_P_batt   = sum(mpc.lineal_J_P_batt[k] / (mpc.N * (mpc.P_ch_max
14            + mpc.P_disch_max)) for k in mpc.k)
15
15        return (mpc.omega_S*J_S_L + mpc.omega_delta*J_delta_L + mpc.
16            omega_P_ch*J_P_ch + mpc.omega_P_disch*J_P_disch + mpc.omega_SOC*
17            J_SOC + mpc.omega_P_batt*J_P_batt)
18
18    mpc.obj2 = pyo.Objective(expr = obj2_expression)

```

Las restricciones de ambos modelos, exceptuando la linealización correspondiente al modelo de la segunda función objetivo, son las siguientes:

Código 4.3 Restricciones.

```

1     def power_balance(mpc, k):
2         return mpc.P_in[k] - mpc.P_ch[k] + mpc.P_disch[k] - mpc.S_L[0,k
3             ]*P_L_HP[k] - mpc.S_L[1,k]*P_L_LP[k] - P_L_crit[k] == 0
4         mpc.power_balance_constraint = pyo.Constraint(mpc.k, rule=
5             power_balance)
6
7     def SOC(mpc, k):
8         if(k==0):
9             return SOC_0 + mpc.T_s*mpc.nu_ch*mpc.P_ch[k]/mpc.B_cap - mpc
10                .T_s*mpc.P_disch[k]/(mpc.nu_disch*mpc.B_cap) == mpc.SOC[k]
11         else:
12             return mpc.SOC[k-1] + mpc.T_s*mpc.nu_ch*mpc.P_ch[k]/mpc.
13                 B_cap - mpc.T_s*mpc.P_disch[k]/(mpc.nu_disch*mpc.B_cap) == mpc.SOC[k
14             ]
15         mpc.SOC_constraint = pyo.Constraint(mpc.k, rule=SOC)
16
17     def battery_mode(mpc, k):
18         return mpc.dseta_ch[k] + mpc.dseta_disch[k] == 1
19         mpc.battery_mode_constraint = pyo.Constraint(mpc.k, rule=
20             battery_mode)
21
22     def charging_mode(mpc, k):
23         return mpc.P_ch[k] <= mpc.dseta_ch[k]*mpc.P_ch_max
24         mpc.charging_mode_constraint = pyo.Constraint(mpc.k, rule=
25             charging_mode)
26
27     def discharging_mode(mpc, k):

```

```

21         return mpc.P_disch[k] <= mpc.dseta_disch[k]*mpc.P_disch_max
22 mpc.discharging_mode_constraint = pyo.Constraint(mpc.k, rule=
23     discharging_mode)
24
25 def lineal_J_S_pos (mpc, i, k):
26     if(k==0):
27         return mpc.S_L[i,k] - S_L_0[i] <= mpc.lineal_J_S[i,k]
28     else:
29         return mpc.S_L[i,k] - mpc.S_L[i,k-1] <= mpc.lineal_J_S[i,k]
30 mpc.lineal_J_S_pos_constraint = pyo.Constraint(mpc.i,mpc.k, rule=
31     lineal_J_S_pos)
32
33 def lineal_J_S_neg (mpc, i, k):
34     if(k==0):
35         return S_L_0[i] - mpc.S_L[i,k] <= mpc.lineal_J_S[i,k]
36     else:
37         return mpc.S_L[i,k-1] - mpc.S_L[i,k] <= mpc.lineal_J_S[i,k]
38 mpc.lineal_J_S_neg_constraint = pyo.Constraint(mpc.i,mpc.k, rule=
39     lineal_J_S_neg)
40
41 def lineal_J_P_batt_pos (mpc, k):
42     if(k==0):
43         return mpc.P_ch[k] - mpc.P_disch[k] - (P_ch_0 - P_disch_0)
44     <= mpc.lineal_J_P_batt[k]
45     else:
46         return mpc.P_ch[k] - mpc.P_disch[k] - (mpc.P_ch[k-1] - mpc.
47         P_disch[k-1]) <= mpc.lineal_J_P_batt[k]
48 mpc.lineal_J_P_batt_pos_constraint = pyo.Constraint(mpc.k, rule=
49     lineal_J_P_batt_pos)
50
51 def lineal_J_P_batt_neg (mpc, k):
52     if(k==0):
53         return P_ch_0 - P_disch_0 - (mpc.P_ch[k] - mpc.P_disch[k])
54     <= mpc.lineal_J_P_batt[k]
55     else:
56         return mpc.P_ch[k-1] - mpc.P_disch[k-1] - (mpc.P_ch[k] - mpc.
57         P_disch[k]) <= mpc.lineal_J_P_batt[k]
58 mpc.lineal_J_P_batt_neg_constraint = pyo.Constraint(mpc.k, rule=
59     lineal_J_P_batt_neg)

```

4.5 Implementación en Simulink

Con el objetivo de llevar a la práctica la estrategia de optimización y simular cómo se comportaría el sistema, se ha diseñado una simplificación del modelo del sistema de potencia eléctrica en Simulink. En este primer diseño, se sustituyen muchos de los elementos presentes en el modelo original de la EPS de la aeronave con la que se trabaja (Figura 4.1), simplemente para validar el modelo y comprobar que los resultados concuerdan con la optimización en Python.

Cabe destacar que en todo momento las optimizaciones realizadas son offline y los resultados

de estas, son introducidas a través del bloque *Signal Builder* en Simulink para la simulación del circuito. En las Figuras 4.3 y 4.4 se muestra cómo se ha implementado el modelo.

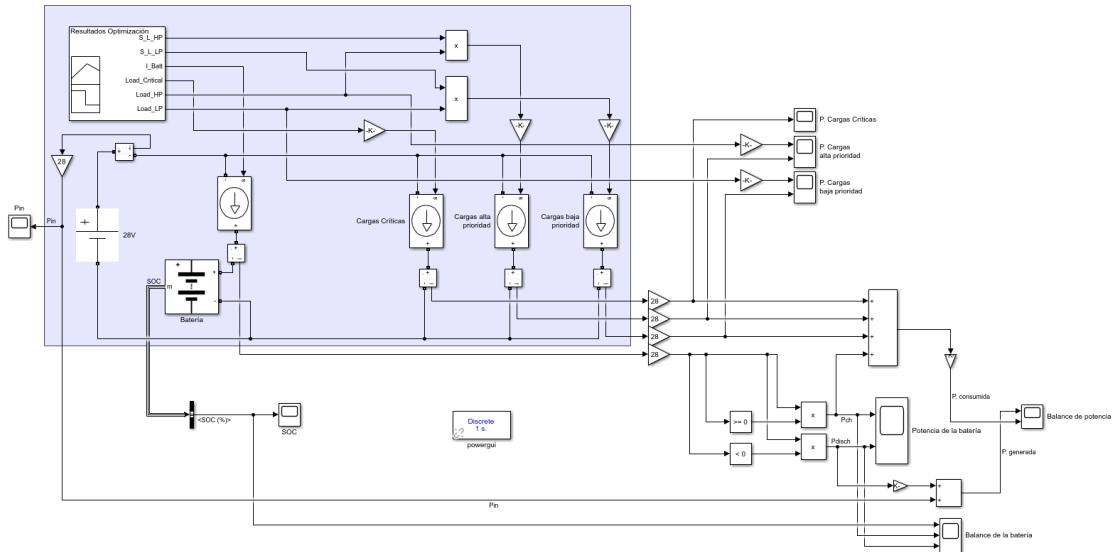


Figura 4.3 Diseño en Simulink.

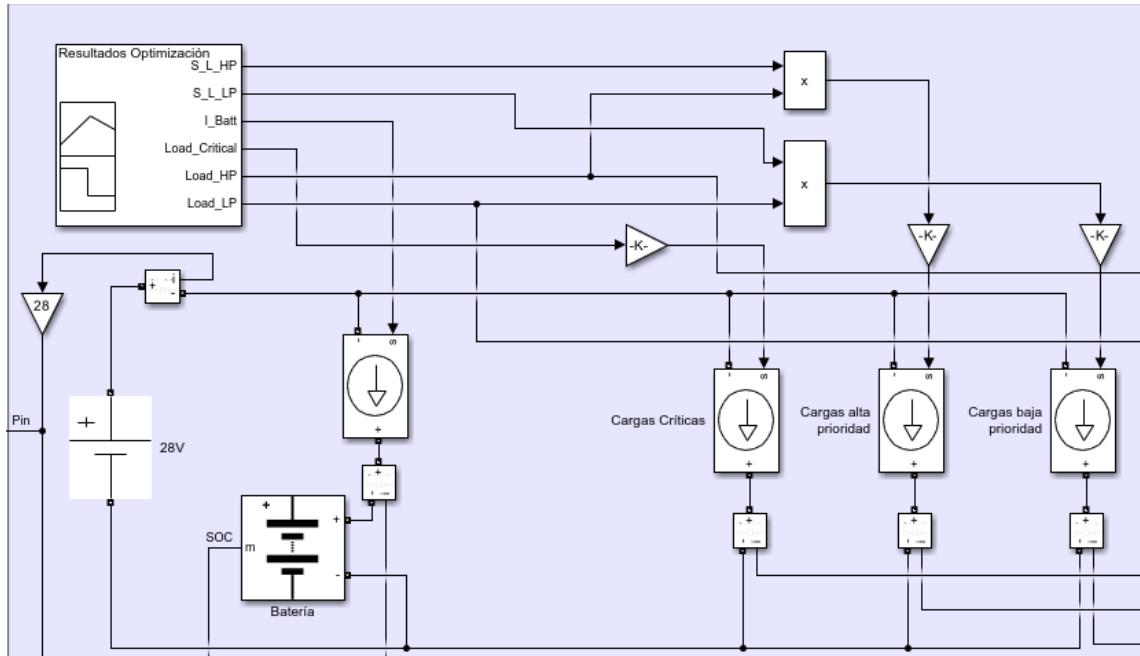


Figura 4.4 Arquitectura de la EPS simplificada del modelo en Simulink.

Las simplificaciones llevadas a cabo respecto a la microrred de la Figura 4.1 son las siguientes:

- El generador junto al convertidor AC/CC y el bus de alta tensión no se modelan debido a que no tienen repercusión en el funcionamiento de nuestro modelo, ya que el único parámetro de interés es la potencia de entrada, P_{in} .

- El convertidor CC/CC a la salida del bus de alta tensión, que a priori regula la potencia de entrada necesaria, se modela como una fuente de tensión constante de 28V.
- El convertidor CC/CC de la batería se sustituye por una fuente de intensidad controlable. Esta recibe como dato de entrada el resultado de la optimización de la intensidad que fluye en la batería en cada paso de tiempo. De este modo, se consigue un funcionamiento similar de la batería en comparación con la batería del modelo en Python. Los parámetros de la batería introducidos se corresponden con un voltaje nominal de 24V, una capacidad nominal de 142.85Ah y un estado inicial de carga del 30%.
- Las distintos grupos de cargas junto a sus respectivos contactores se modelan también como fuentes de intensidad controlables en paralelo. Como datos de entrada a las fuentes de intensidad se les proporciona el producto de los resultados de la optimización de los estados de los contactores y el perfil de demanda de las cargas, dividida por la tensión de la fuente principal, obteniéndose así la intensidad que atravesaría las cargas. De esta forma, la potencia que consume cada fuente de intensidad controlable es igual a la demanda de potencia de cada grupo de cargas del sistema real, ya que al estar conectadas en paralelo, a cada fuente de intensidad le corresponde una tensión igual a la de la fuente de tensión principal, y el producto de tensión por intensidad igualaría la potencia correspondiente. El hecho de multiplicar el perfil de demanda de las cargas por el estado de los contactores en cada paso de tiempo, se realiza para simular la conexión/desconexión de cargas.
- El controlador MPC no es necesario modelarlo ya que se trata de una optimización offline y los datos son introducidos manualmente.

5 Resultados

Para llevar a cabo el análisis de los resultados, se propone un sistema de índices con el cual estudiar el rendimiento de cada objetivo y, en consecuencia, de cada estrategia de optimización, definidos por las ecuaciones establecidas en la sección anterior. Aunque la aplicación en tiempo real del controlador MILP-MPC queda fuera del objeto de estudio de este trabajo, debería cumplirse que el tiempo de computación de la estrategia de optimización sea menor que el tiempo de sampleo $T_S = 1 \text{ min}$, ya indicado en el apartado 4.2. El tiempo total del vuelo será $T = M \cdot T_S$, donde M es el número total de intervalos, el cual en este trabajo se establece igual a 150. En las próximas secciones, se comenzará mostrando los valores adoptados por los parámetros del sistema y se explicará el entorno de evaluación por índices para analizar los resultados.

5.1 Parámetros del modelo

A continuación, se consideran los parámetros de funcionamiento de los diferentes elementos del sistema. han sido sintonizados de manera manual, viendo la influencia que tenían en los diferentes resultados. Otra forma de sintonizarlos, es usando técnicas como redes neuronales [33] las cuales se han usado en el artículo en el que se basa este trabajo. . Se ha iterado numerosas veces con diferentes pesos, respetando los objetivos operacionales del sistema, hasta obtener resultados consistentes con los del resto de artículos. Los valores se recogen en las Tablas 5.1 y 5.2.

Tabla 5.1 Valores de los parámetros del EPS.

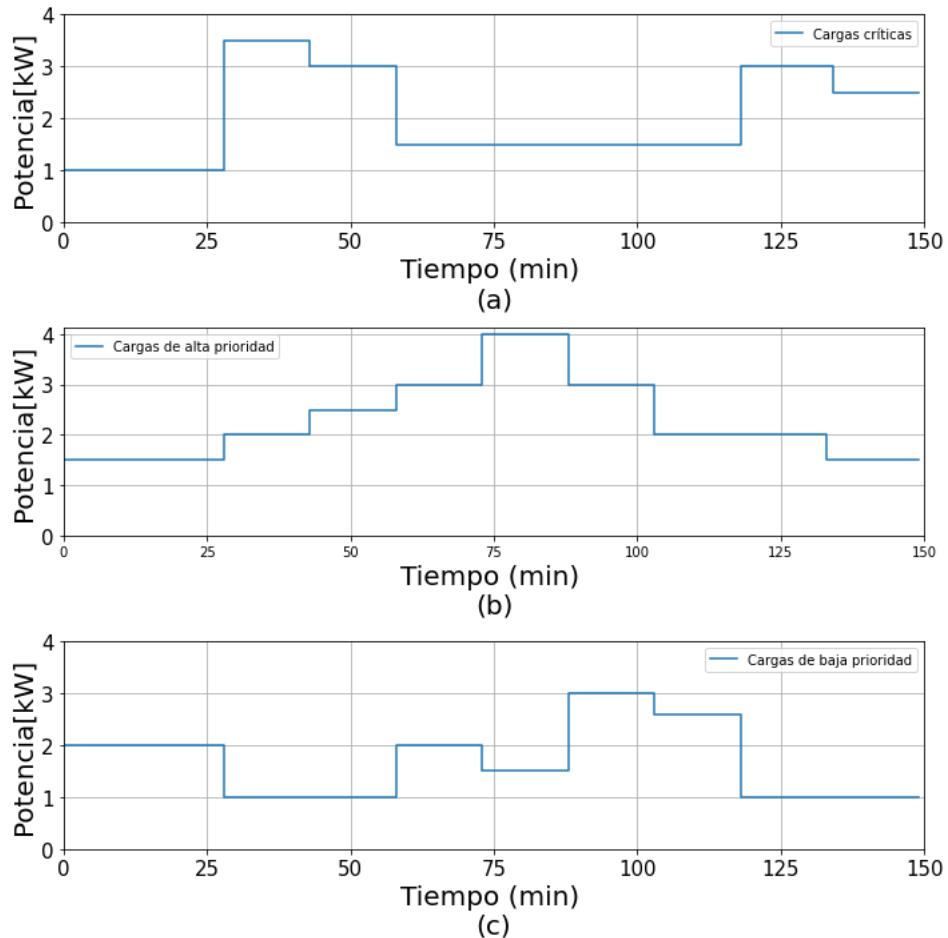
B_{cap}	4kWh	$p_{ch}^{max}, p_{disch}^{max}$	4kWh
p_{in}^{max}	4kWh	η_{ch}, η_{disch}	90 %
p_{Load}^{max}	4kWh	$SOC(0)$	0.3
γ_{HP}	2	γ_{LP}	1

El perfil de cargas que simula el consumo de una aeronave se muestra en la Figura 5.1. Las etapas del vuelo incluyen operaciones en tierra, *taxiing-out*, despegue, subida, crucero, descenso, aterrizaje y *taxiing-in*:

Para cada etapa del vuelo, cada grupo de cargas (críticas, alta prioridad y baja prioridad) requiere una potencia determinada. Por ejemplo, en el despegue y aterrizaje hay una mayor demanda de potencia de las cargas críticas por parte de los sistemas de control de vuelo y el tren de aterrizaje,

Tabla 5.2 Valores de los pesos del modelo.

ω_S	5	ω_δ	1
ω_{SOC}	9	ω_{Pbatt}	0.75
ω_{Pch} 3 in <i>Obj1</i> , 2.3 in <i>Obj2</i>			
ω_{Pdisch} 3			

**Figura 5.1** Perfil de cargas: (a) críticas, (b) alta prioridad y (c) baja prioridad.

mientras que, durante la etapa de vuelo en crucero, la cabina demanda la mayor parte de la potencia debido a los servicios y el confort de los pasajeros.

5.2 Índices de evaluación

Se proponen 5 índices para analizar los resultados en base al escenario presentado por la demanda del perfil de cargas en la Figura 5.1. En estos índices, se toman en consideración los resultados de la optimización de las funciones objetivo *Obj1* y *Obj2* para un rango de horizontes de predicción de $1T_S$ a $30T_S$.

Todos los índices están normalizados y evalúan los diferentes objetivos de control para todo el tiempo de vuelo. Un mayor rendimiento se obtiene cuanto más bajos sean los valores obtenidos en los índices.

5.2.1 Índice de desconexión de cargas

El índice normalizado de desconexión de cargas cuantifica el ratio de cargas desconectadas a lo largo de todo el tiempo de vuelo:

$$G_{S_{Li}} = \sum_{k=0}^{M-1} \sum_{i=1}^{N_{Li}} \gamma_{Li}(1 - S_{Li}(k)) / (M \cdot \sum_{i=1}^{N_{Li}} \gamma_{Li}) \quad (5.1)$$

En la ecuación 5.1, el valor del índice aumenta cada vez que en una iteración las cargas están desconectadas, ya que el valor de S_{Li} sería 0. El valor del parámetro M hace referencia al total de pasos de tiempo del vuelo, es decir, 150.

El resultado de este índice para los horizontes analizados se muestra en la figura 5.2. Se puede observar como para *Obj1* tiende a disminuir ligeramente conforme aumenta el horizonte de predicción, aunque podemos considerar que los valores son prácticamente constantes ya que las pequeñas variaciones entre ellos se deben a escasos y cortos intervalos de tiempo de desconexión de las cargas diferentes a lo largo de toda la simulación entre los horizontes evaluados, haciendo los balances de potencias entre ellos muy parecidos. Esta pequeña tendencia a priorizar la conexión de las cargas se debe a los valores de los pesos aplicados en nuestro estudio, ya que $\omega_S > \omega_{P_{ch}}$.

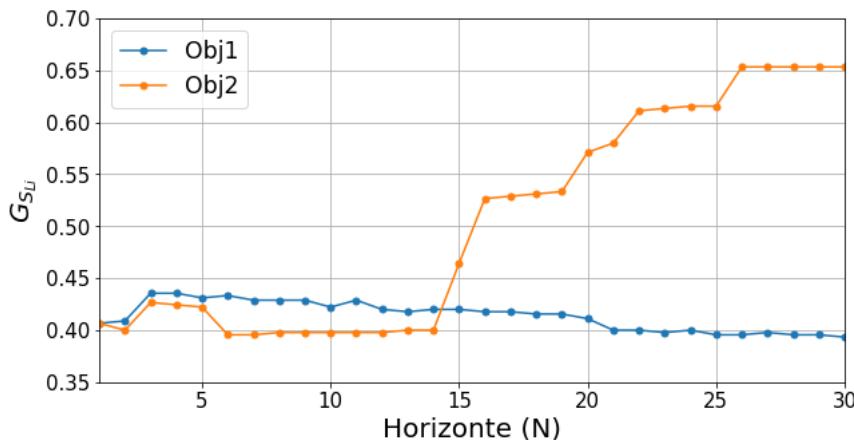


Figura 5.2 Comparación del índice normalizado de desconexión de cargas para diferentes funciones objetivo y horizontes de predicción.

En *Obj2*, para valores pequeños del horizonte, se tiene un comportamiento parecido los resultados obtenidos con la función *Obj1* e incluso mejor entre $N = 5$ y 14 . Sin embargo, llegados a cierto valor del horizonte fijado, en este caso $N = 14$, se encuentra un punto de inflexión y se produce un descenso en el rendimiento del sistema donde este comienza a desconectar las cargas durante periodos de tiempo más largos. En concreto este aumento se debe a la desconexión de las cargas de alta prioridad, las cuales al tener un valor más alto en la preferencia, tienen más peso en la función de $G_{S_{Li}}$. Esto se debe principalmente a la presencia de la función de coste acumulativa en el tiempo J_{SOC} en la función objetivo, la cual prioriza la carga de la batería, para lo cual incentiva que mayor parte de la potencia generada, P_{in} , sea transformada en potencia de carga P_{ch} .

5.2.2 Índice de cambio de estado de contactores

El número medio de cambios en el estado de cada contactor en todo el tiempo de vuelo se evalúa en el índice de cambio de estado de los contactores:

$$G_{\delta Li} = \sum_{k=0}^{M-1} \sum_{i=1}^{N_{Li}} |S_{Li}(k+1) - S_{Li}(k)| / (M \cdot N_{Li}) \quad (5.2)$$

En la ecuación 5.2, el valor del índice aumenta cada vez que en una iteración cambia el estado de los contactores.

En la figura 5.3 se comparan los valores para dicho índice. Para ambos casos, a partir de $N = 3$, el comportamiento mejora rápidamente y la tendencia es reducir el cambio de estado de los contactores conforme aumenta la longitud del horizonte de predicción, con unos resultados similares en ambas funciones objetivo definidas. En *Obj1* los resultados son un poco más altos e inestables. En contraposición, los valores obtenidos para *Obj2* son más estables para todos los horizontes de predicción. Se puede observar entre $N = 15$ y $N = 25$, justo después del punto de inflexión descrito en el anterior índice, un descenso de los cambios de estado de los contactores para *Obj2*, debido a la mayor capacidad de programación de la carga de la batería de dicha función objetivo por la inclusión de J_{SOC} , haciendo que se conecten y desconecten con menos frecuencia las cargas. Aun así, los valores del índice son muy parejos y pequeños en comparación del resto de índices, por lo que contribuirán en menor medida en el rendimiento general del sistema.

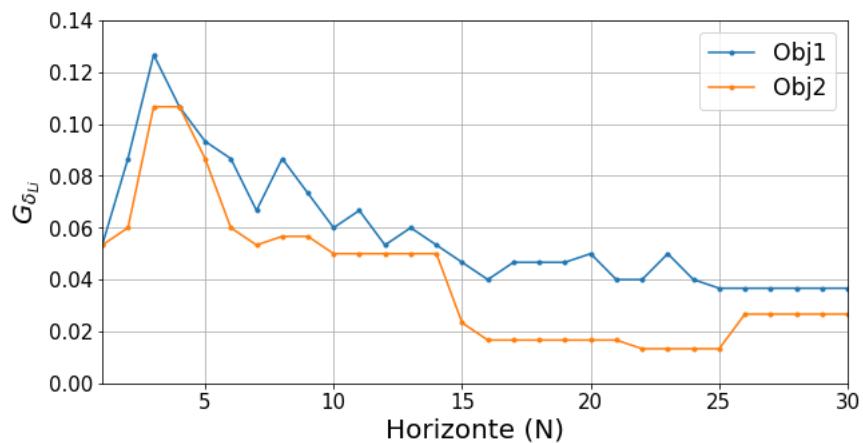


Figura 5.3 Comparación del índice normalizado de cambio de estado de contactores para diferentes funciones objetivo y horizontes de predicción.

Se puede observar como para *Obj2* se obtienen resultados peores para el índice de desconexión de cargas de la Figura 5.2, mientras que en este índice son mejores. Esto se debe a que los objetivos de desconexión de cargas y almacenamiento de energía en las baterías son contrarios.

Antes de continuar, es necesaria una aclaración respecto a los valores obtenidos cuando el horizonte es igual a 1. Cuando los índices son evaluados para $N = 1$, el sistema simplemente prioriza el requisito operacional que más le convenga en ese paso según los pesos de las funciones de coste y la situación en la que se encuentre. Por ello, no necesita cambiar continuamente los estados de los contactores ya que si, por ejemplo, se encuentran las cargas de alta prioridad conectadas y no necesita una excesiva P_{disch} para cumplir la demanda, simplemente permanecerá en ese estado hasta que la batería se descargue demasiado o la potencia demandada sea demasiado alta. Un razonamiento análogo puede realizarse para el índice del apartado 5.2.4

5.2.3 Índice de nivel de carga de la batería

El índice G_{SOC} , permite evaluar el nivel medio de energía almacenada en la batería durante el vuelo de la aeronave. Es preferible mantener la batería cargada, para que en caso de que ocurra algún fallo o emergencia pueda suministrarse esta potencia auxiliar.

$$G_{SOC} = \sum_{k=0}^{M-1} |0.9 - SOC(k)| / (M \cdot 0.9) \quad (5.3)$$

En la ecuación 5.3, cuanto mayor es la diferencia entre el SOC y el valor máximo del mismo para cada paso de tiempo, más aumenta el valor del índice.

Los resultados obtenidos en la Figura 5.4 reflejan la diferencia respecto a la priorización de la carga de la batería en las dos funciones objetivos. En ambas funciones objetivos, es una tendencia que la carga de la batería aumenta conforme el horizonte de predicción es mayor. En *Obj1*, la carga de la batería aumenta ligeramente según el horizonte de predicción es mayor, aunque esta tendencia no es muy acusada ya que para la primera función objetivo, $\omega_{P_{ch}}$ es menor que ω_S , no siendo la mayor prioridad el aumento del SOC. Pero debido a que con el aumento del horizonte el sistema adquiere una mayor capacidad predictiva y es capaz de visionar las demandas futuras de potencia, gestiona mejor las variaciones de potencia dando lugar a que también se cargue la batería. Además, en ocasiones esta se acerca a el límite inferior, por lo que el sistema debe derivar potencia a su carga.

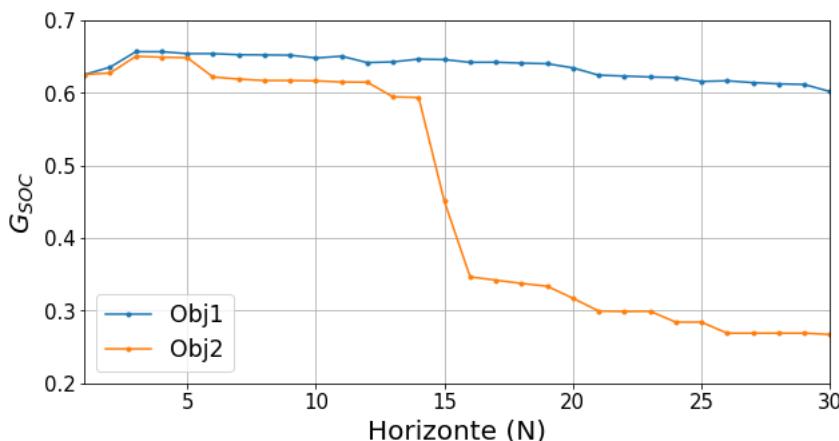


Figura 5.4 Comparación del índice normalizado de nivel de carga de la batería para diferentes funciones objetivo y horizontes de predicción.

Por otro lado, la presencia de la función J_{SOC} y su elevado peso, ω_{SOC} , en la función objetivo *Obj2*, hacen que el valor de los resultados descienda en gran medida para dicha función objetivo cuando supera el punto de inflexión $N > 14$.

Claramente, *Obj2* indica un mejor comportamiento para el índice de carga de la batería. El horizonte de predicción tampoco debe ser seleccionado demasiado grande, ya que para valores medios se obtiene un rendimiento mucho mejor que *Obj1* y muy aceptable.

5.2.4 Índice de cambio de potencia de la batería

La vida de la batería se ve reducida cuando se somete continuamente a cambios constantes en la carga y descarga. El índice de cambio de potencia de la batería calcula los cambios de potencia de la batería para todo el tiempo de vuelo:

$$G_{P_{batt}} = \sum_{k=0}^{N-1} |(P_{ch}(k+1) - P_{disch}(k+1)) - (P_{ch}(k) - P_{disch}(k))| / N(P_{ch}^{max} + P_{disch}^{max}) \quad (5.4)$$

En la ecuación 5.4, cuanto mayor es la diferencia entre la potencia que carga o descarga la batería entre pasos de tiempo consecutivos, más aumenta el valor del índice.

El comportamiento entre las funciones objetivo es semejante cuando se evalúa el índice de cambio de potencia de la batería analizado en la Figura 5.5. Al igual que en $G_{\delta_{Li}}$, a partir del punto de inflexión en $N = 3$, el rendimiento de ambas funciones mejora rápidamente. Para todos los horizontes definidos $Obj2$ tiene un mejor rendimiento comparado con $Obj1$, debido a que $Obj1$ carece de la función de coste J_{Pbatt} . Aunque también hay que destacar la presencia de J_{SOC} en la segunda función objetivo.

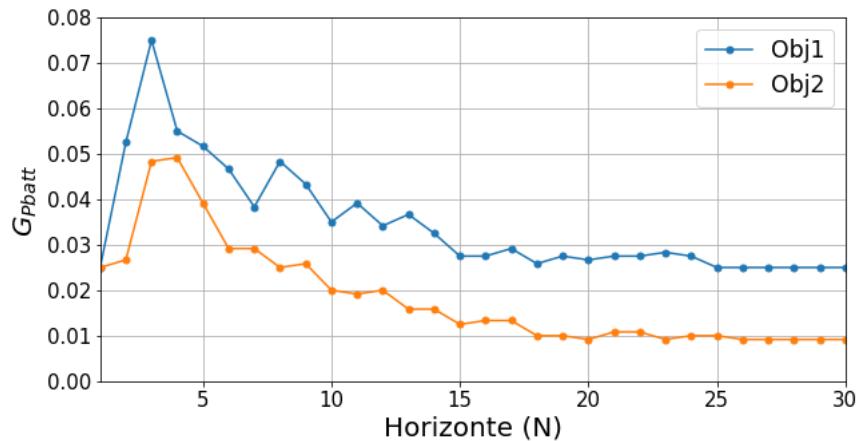


Figura 5.5 Comparación del índice normalizado de cambio de potencia de la batería para diferentes funciones objetivo y horizontes de predicción.

En ambas funciones objetivos los cambios de potencia de la batería tienden a disminuir conforme aumenta el horizonte. Este resultado se debe a que como se ha explicado anteriormente, el sistema tiene más capacidad de programación de la demanda y como se puede comprobar, los cambios de estado de los contactores también disminuyen, ayudando que las variaciones de potencia de la batería sean más suaves. Cabe destacar que J_{Pbatt} y J_{SOC} influyen indirectamente en los resultados de las desconexiones de cargas y los cambios de estado de los contactores, siendo este otro de los motivos por los cuales es difícil ajustar el valor de los pesos de las funciones objetivos.

En resumen, $Obj2$ alcanza mejores resultados como era de esperar gracias a incluir el coste J_{Pbatt} . Esta función objetivo es más adecuada cuando la vida de la batería es importante.

5.2.5 Índice multiobjetivo

Se establece este último índice con el que se pretende analizar el comportamiento general del sistema para el total del vuelo.

$$G_{MO} = v_S G_{S_{Li}} + v_\delta G_{\delta_{Li}} + v_{SOC} G_{SOC} + v_{P_{batt}} G_{P_{batt}} \quad (5.5)$$

En la ecuación 5.5 se suman todos los índices anteriores ponderados con sus pesos correspondientes.

En un problema multiobjetivo con funciones de coste que representan objetivos contrarios, se producen compensaciones entre los índices de evaluación. Por ello, se incluyen los pesos para cada índice recogidos en la Tabla 5.3, con el objetivo de representar los objetivos de control en el rendimiento general del sistema.

Tabla 5.3 Valores de los pesos del índice multiobjetivo.

v_S	2.8	v_δ	0.4
v_{SOC}	1	v_{Pbatt}	0.4

La evaluación general del sistema por medio del índice multiobjetivo se muestra en la Figura 5.6. Es posible apreciar que, aunque ambas funciones objetivos consigan resultados similares en sus horizontes óptimos, la función objetivo *Obj2* lo hace para un horizonte mucho menor, lo cual beneficia al controlador debido al menor tiempo de computación que requiere. Realmente, estos resultados dependen totalmente de cómo se hayan elegido los pesos de las funciones de coste, según los objetivos de control que quiera priorizar el usuario, dotando de una gran flexibilidad al sistema. Por tanto, estos resultados ejemplifican el funcionamiento de un controlador donde se pretende dar una importancia similar a la conexión de las cargas y a la carga de la batería, es decir, un balance intermedio del uso de la potencia. Sin embargo, al añadir las funciones de costes centradas en el estado de la batería y sus fluctuaciones en la función objetivo *Obj2*, el cuidado de la vida y los ciclos de la batería pasa a ser el principal requisito operacional.

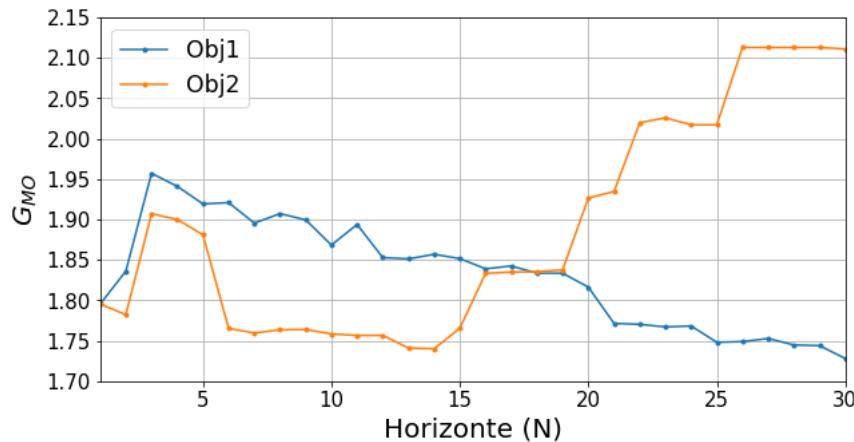


Figura 5.6 Comparación del índice multiobjetivo para diferentes funciones objetivo y horizontes de predicción.

Por otro lado, aunque para valores del horizonte 1 o 2 se obtengan buenos resultados, se debe a lo explicado en el apartado 5.2.2 acerca de cuando se opera con horizonte 1. En estos casos, aunque los resultados sean buenos, no siempre se trata de la solución óptima o pretendida sino la más cómoda para el sistema.

5.3 Simulación del sistema

En este apartado, se ofrece una comparación de las simulaciones del sistema para cada función objetivo. En primer lugar, se realiza una comparativa general para mostrar el correcto funcionamiento del sistema y como varía el flujo de potencia en él. Para ello, se recoge en las Figuras 5.7, 5.8, 5.9, la simulación de ambos modelos para un horizonte de predicción $N = 10$. Se ha tomado esta longitud de horizonte porque aporta resultados más diferenciables a simple vista, manteniendo la veracidad de una aplicación real.

Cabe destacar que la representación de la potencia de entrada, P_{in} , se omite debido a que siempre es igual a 4. Este resultado del sistema se debe a que no existe ninguna penalización en la generación de potencia de entrada, por lo que lo más óptimo en todo momento es generar la mayor P_{in} posible para alimentar más tiempo las cargas o cargar la batería.

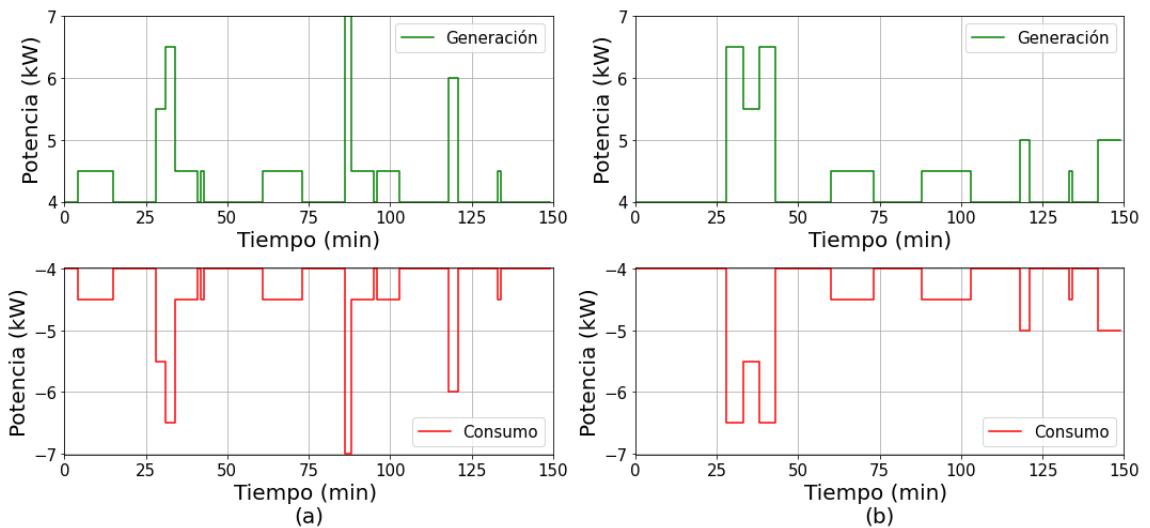


Figura 5.7 Balance de potencia con $N = 10$ para: (a) *Obj1*, (b) *Obj2*.

Puede apreciarse como la potencia generada siempre es igual a la consumida. Por otro lado, el sistema carga o descarga la batería, pero nunca ambas cosas a la vez, cumpliendo esta otra restricción. Además, se observa que cuando el sistema carga la batería, el balance de potencia siempre es igual a 4, ya que P_{disch} tiene que ser igual a 0 y no existe otro método de generación d potencia. Por último, se observa que el SOC responde a las variaciones de la potencia de la batería aumentando o disminuyendo su valor según se cargue o descargue.

A continuación, se eligen diferentes horizontes para que las diferencias entre la respuesta del sistema según modelo u otro sean más notables.

En primer lugar, se muestra el esquema de conexión/desconexión de cargas de alta y baja prioridad a lo largo del tiempo de vuelo para las funciones objetivo *Obj1* y para *Obj2* en las Figuras 5.10 y 5.11. El análisis se realiza para $N = 8$ y $N = 20$, de forma que podamos observar cómo cambia la actuación del sistema. Podemos observar cómo para *Obj1* los estados del contactor varían en mayor medida que los de *Obj2* para ambos horizontes. Para $N = 8$, aunque *Obj1* varíe más el estado de los contactores, el tiempo de desconexión de las cargas es mayor que en *Obj2* como se sabe de los resultados obtenidos por el índice G_S . Además, para $N = 20$ el tiempo de desconexión de las cargas es mucho mayor para *Obj2*, como se predecía de los valores de los índices, y prioriza la carga de la batería. Como veremos más adelante.

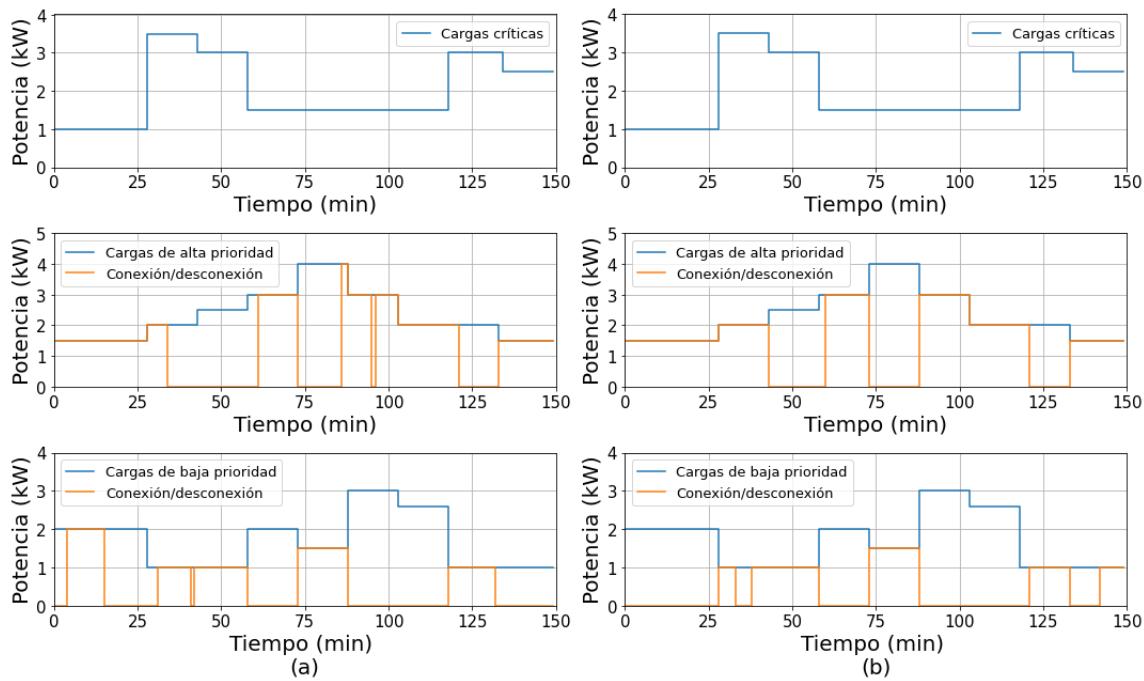


Figura 5.8 Conexión/desconexión de cargas con $N = 10$ para: (a) *Obj1*, (b) *Obj2*.

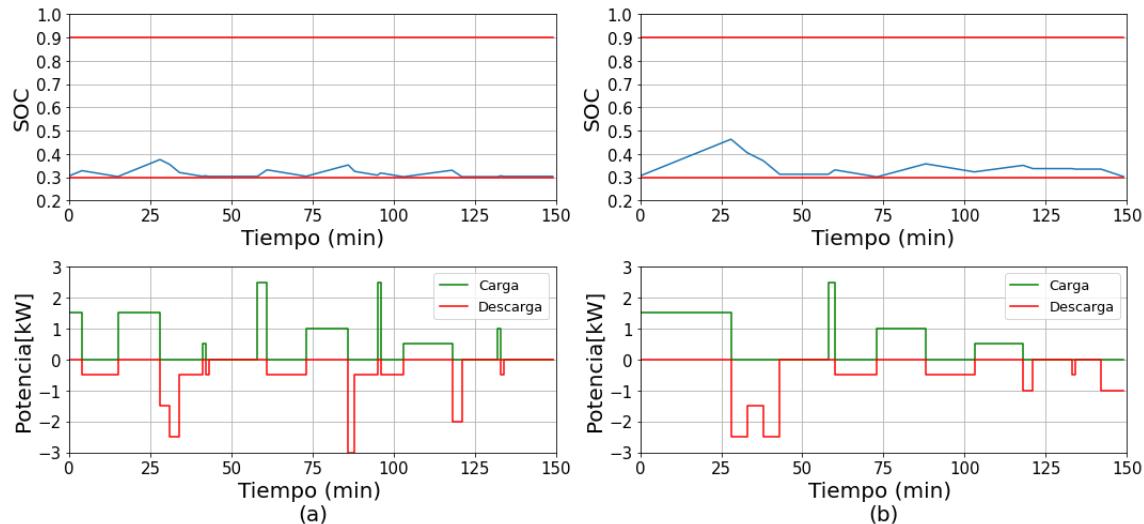


Figura 5.9 Evolución de la batería con $N = 10$ para: (a) *Obj1*, (b) *Obj2*.

En segundo lugar, se ofrece una comparación entre el estado del SOC y los cambios de potencia de la batería para las funciones objetivo *Obj1* y para *Obj2* en la Figura 5.12. Dicha comparación se realiza únicamente para $N = 15$, ya que para horizontes anteriores el sistema tiene un comportamiento similar y para horizontes mayores, la priorización de la carga de la batería en *Obj2* es demasiado acusada.

Se comprueba como la evolución positiva o negativa del SOC se corresponde con instantes en los que el sistema genera potencia de carga o de descarga. Además, como era obvio dado los resultados de los índices, para *Obj2* el SOC es mucho más elevado, manteniéndose entre 0.4 y 0.7 casi todo el tiempo, en comparación con *Obj1*, donde permanece siempre cercano al límite inferior. *Obj1* tiene, por tanto, mucha menor capacidad para lidiar con escenarios de emergencia, picos de potencia o fallos. También se puede corroborar como las variaciones de potencia de la batería son más suaves

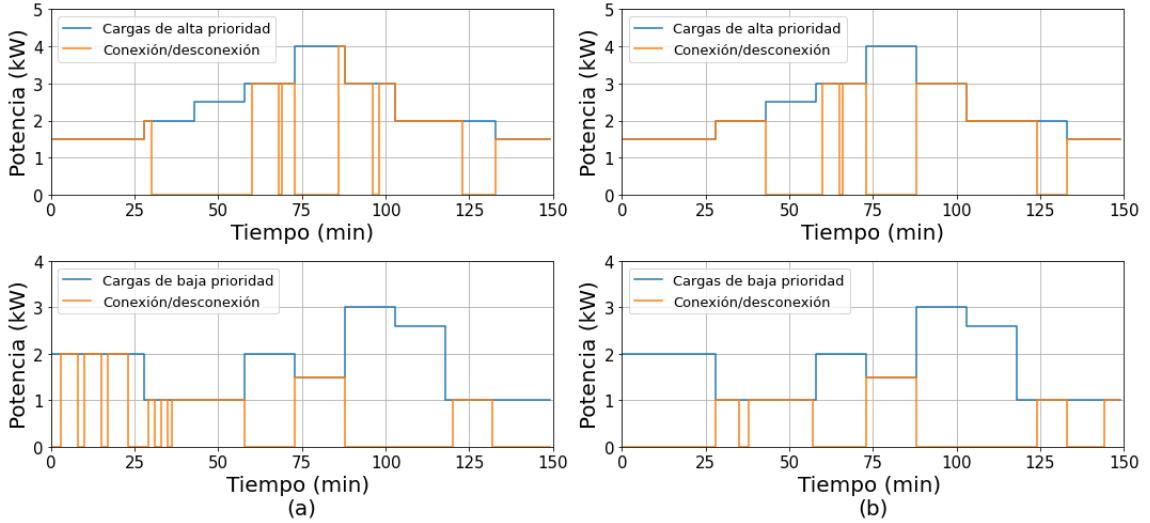


Figura 5.10 Conexión/desconexión de cargas con $N = 8$ para: (a) *Obj1*, (b) *Obj2*.

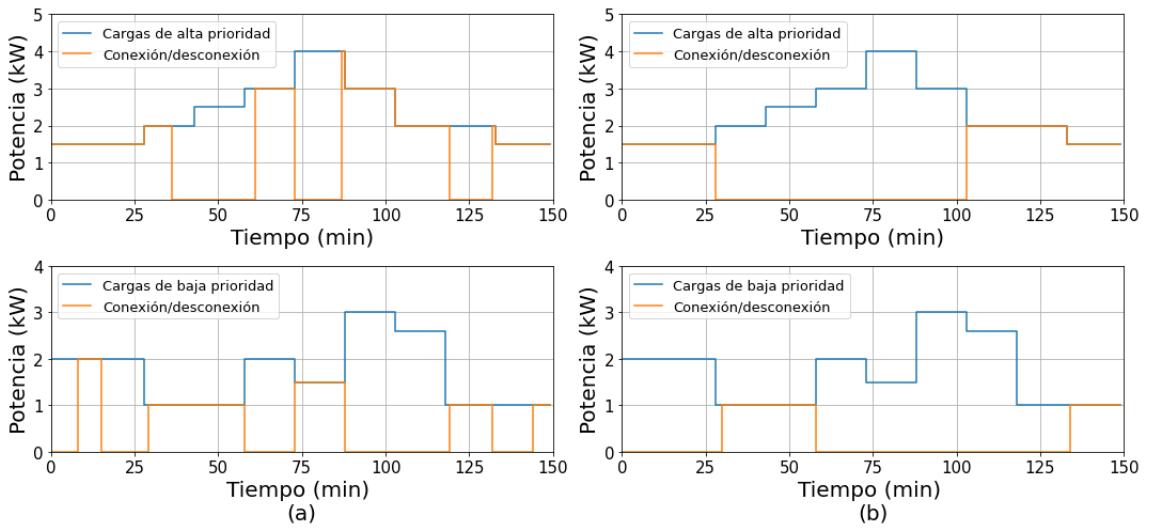


Figura 5.11 Conexión/desconexión de cargas con $N = 20$ para: (a) *Obj1*, (b) *Obj2*.

en *Obj2* gracias a J_{Pbatt} .

5.4 Casos particulares

En este apartado, se ensayarán algunos casos para corroborar la validez del modelo y analizar cómo responde ante variaciones de las condiciones iniciales o del modelo.

5.4.1 MHE frente a MPC simple

En este apartado se analizarán como varían los resultados entre el MPC-MHE con $N = 30$ y el mismo problema optimizado con un controlador MPC sin MHE, es decir, una optimización simple del sistema pero con un horizonte de predicción igual al total del vuelo de la aeronave ($N = 150$). Se empleará la función objetivo *Obj1*.

Se obtiene $G_{S_{MPC}} = 0.37$, muy cercano al valor de $G_{S_{Li}}$ del controlador MPC-MHE, por lo que se confirma que el tiempo de desconexión de cargas es muy parecido, como se puede observar en

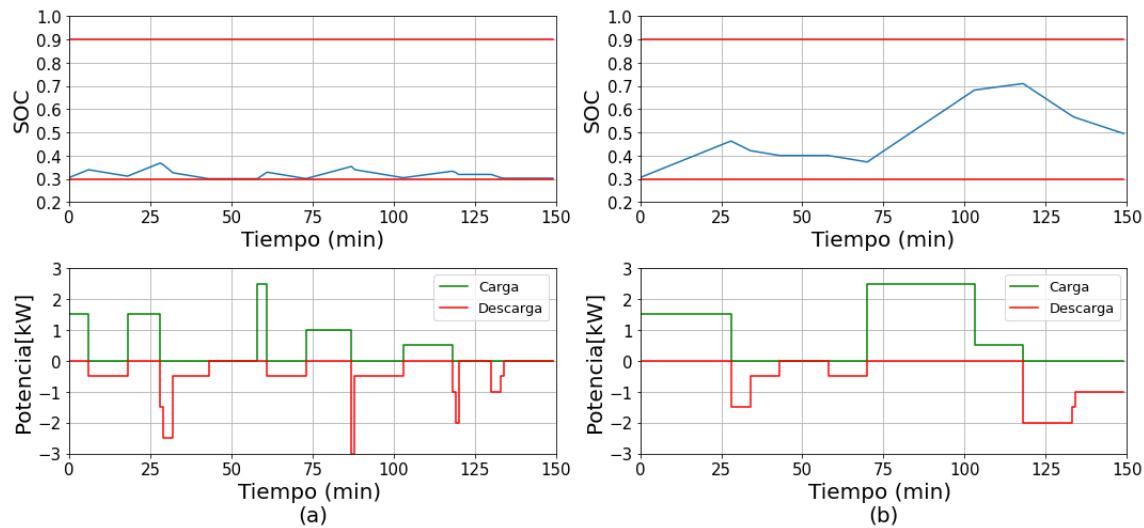


Figura 5.12 Evolución del SOC y de la potencia de carga y descarga de la batería con $N = 15$ para:
(a) *Obj1*, (b) *Obj2*.

la Figura 5.13. Sin embargo, para el controlador MPC simple los contactores cambian menos de estado.

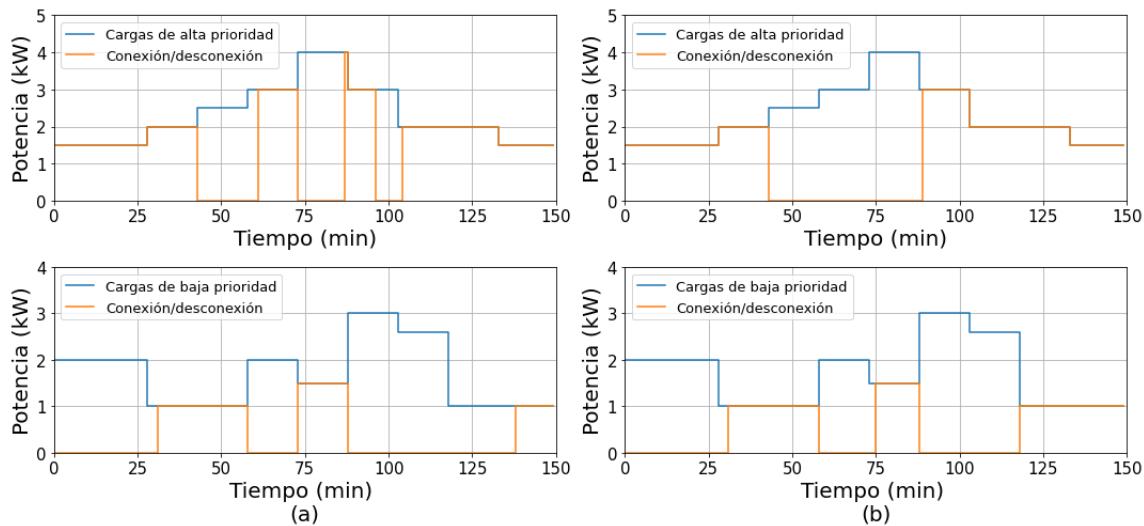


Figura 5.13 Desconexión de cargas para: (a) controlador MPC-MHE con $N = 30$, (b) optimización simple con $N = 150$.

Como se muestra en la Figura 5.14, El controlador MPC simple también preserva mejor la carga de la batería debido a que cuenta con un horizonte mucho mayor, de forma que es capaz de programar mejor la demanda. Por último, como cabía esperar, también sufre muchas menos variaciones de potencia en la batería.

Estos resultados demuestran que *Obj1* es más eficiente conforme mayor sea el horizonte de predicción y que, efectivamente, el modelo y las simulaciones son correctos. Sin embargo, que el controlador MPC simple haya obtenido mejores resultados que el controlador MPC-MHE, no quiere decir que este sea mejor. El controlador sin la herramienta predictiva y de actualización del

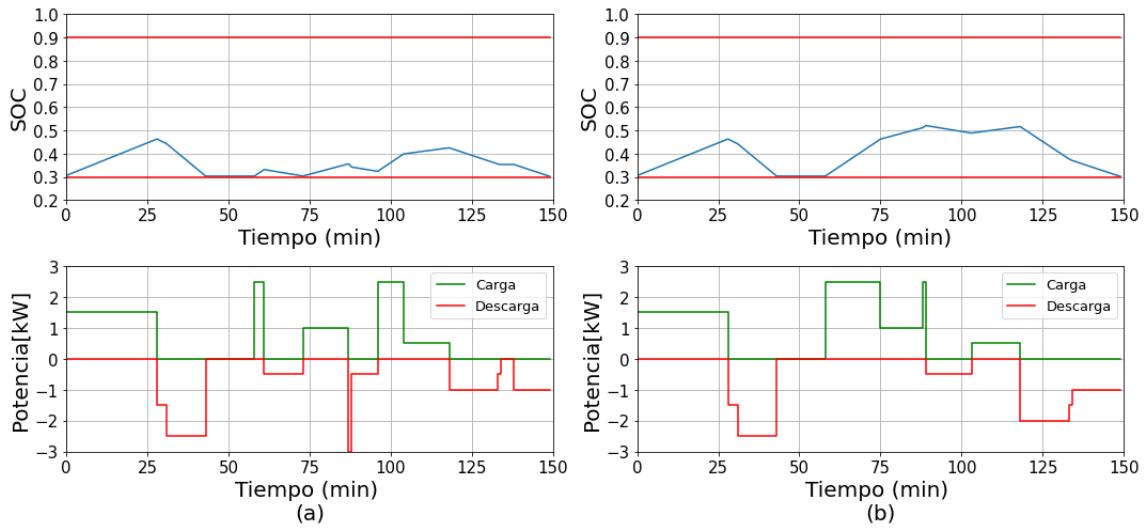


Figura 5.14 Evolución de la batería para: (a) controlador MPC-MHE con $N = 30$, (b) optimización simple con $N = 150$.

sistema carece de flexibilidad y capacidad de reacción ante variaciones en el sistema, haciéndolo inviable para la aplicación en una aeronave.

5.4.2 Condición inicial del SOC

En este apartado, se cambia el valor inicial del SOC de 0.3 a 0.6, para analizar cómo responde el sistema teniendo más potencia disponible en la batería. También se emplea *Obj1* con un horizonte $N = 30$.

Efectivamente, el sistema responde conectando tanto las cargas de alta como de baja prioridad debido a que tiene energía suficiente disponible y según los pesos de las funciones de coste, esta sería la operación más rentable. Modificando dichos pesos o usando la función *Obj2*, observaríamos como seguramente el valor del SOC inicial no descendería tan rápido, se mantendría constante, o incluso tendería a aumentar.

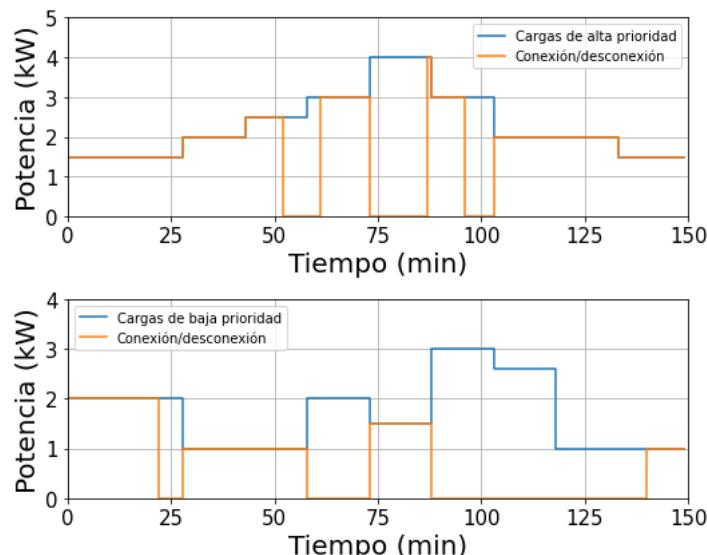


Figura 5.15 Desconexión de cargas para un estado de carga inicial igual a $SOC_0 = 0.6$.

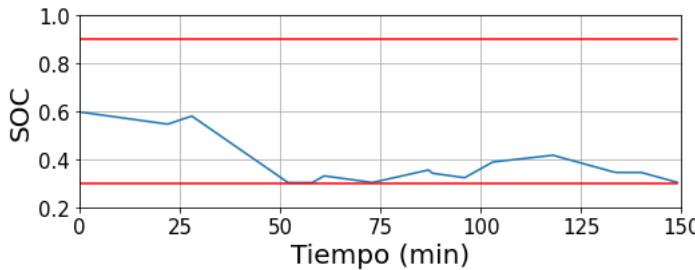


Figura 5.16 Evolución del estado de carga a lo largo del tiempo partiendo de un estado de carga inicial $SOC_0 = 0.6$.

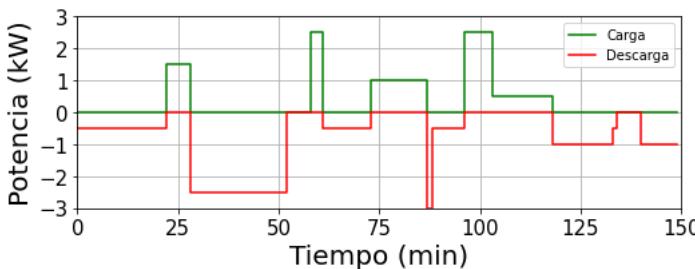


Figura 5.17 Potencia de carga y descarga para un estado inicial de carga de la batería $SOC_0 = 0.6$.

5.4.3 Desconexión parcial de las cargas

La posibilidad de desconexión parcial de las cargas podría ser incluido en futuros trabajos ya que su análisis es más complejo y ofrece más posibilidades. Para ofrecer una idea de cómo trabajaría el sistema con esta modificación se presentan los siguientes análisis. En las Figuras 5.18, 5.19, 5.20 se muestran los resultados.

En (a) para $Obj1$ y $N = 30$, se observa como debido a la posibilidad de desconectar un porcentaje de las cargas, el sistema mantiene las cargas conectadas más tiempo y la batería no es cargada tan a menudo como en el caso sin desconexión parcial

En (b) para $Obj2$ y $N = 14$ (antes del punto de inflexión analizado en los índices de evaluación), las cargas permanecen conectadas más tiempo y además, el sistema es capaz de cargar más la batería.

Podemos observar en ambos casos como desconecta o conecta progresivamente las cargas para beneficiar la carga de la batería o reducir la descarga, con el mismo ritmo con el que desconecta/conecta las cargas.

5.5 Simulación en Simulink

Para terminar, se exponen los resultados obtenido en Simulink del modelo. La simulación se ha realizado para los 150 minutos del vuelo y se han introducido los resultados de la optimización empleando la segunda función objetivo, $Obj2$, con un horizonte igual a 10. Es posible analizar los resultados en las Figuras 5.21, 5.22, 5.23 y 5.24

Los resultados obtenidos son prácticamente iguales que los de la optimización en Python ya que como se ha explicado, los valores de las variables introducidos en Simulink son extraídas de dichos resultados de la optimización offline y por tanto, simplemente se limita a simular el sistema. Sin

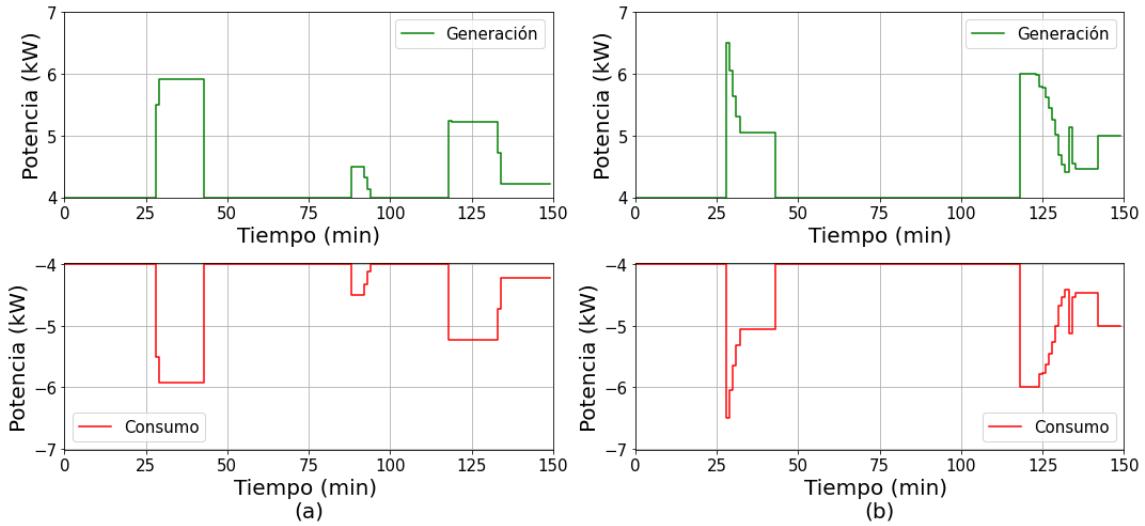


Figura 5.18 Demanda de potencia con desconexión parcial de cargas para: (a) *Obj1* con $N=30$, (b) *Obj2* con $N=14$.

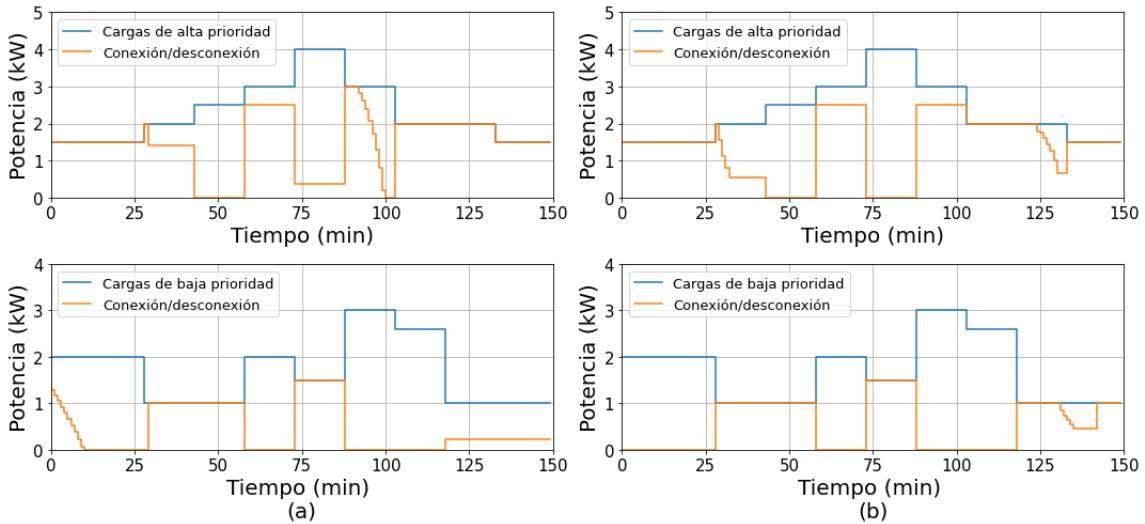


Figura 5.19 Desconexión parcial de cargas para: (a) *Obj1* con $N=30$, (b) *Obj2* con $N=14$.

embargo, puede observarse en la Figura 5.21 como se producen en P_{in} unos picos de potencia que no aparecen en la optimización del modelo. Estos picos de potencia coinciden con los intervalos de tiempo en los que se conecta un grupo de cargas y se desconecta el otro, debido a que el transitorio entre los estados de los conectores es lineal durante los 60 segundos del intervalo de tiempo. Por tanto, se da lugar a estos aumentos puntuales de P_{in} para compensar el progresivo aumento de potencia demandada por el grupo de cargas que se conecta y la disminución de demanda del grupo que se desconecta.

Por otro lado, los resultados del SOC no son exactamente iguales a los de la optimización debido a las diferencias en el modelado entre la batería en Simulink y la batería modelada en Python, ya que no se disponen de los parámetros característicos de la batería del artículo en el que se basa este trabajo. Sin embargo, se puede observar como el comportamiento sí es el mismo.

Como conclusión, se justifica el por qué es conveniente realizar las simulaciones en este tipo de entornos para detectar las posibles transiciones y diferencias que podrían existir en la práctica, y

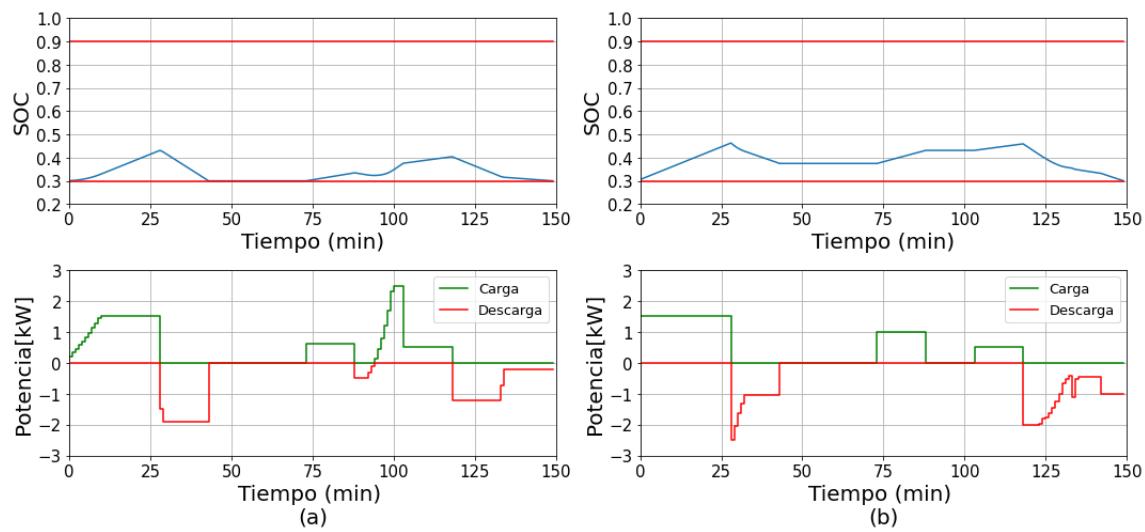


Figura 5.20 Evolución de la batería con desconexión parcial de cargas para: (a) *Obj1* con $N=30$, (b) *Obj2* con $N=14$.

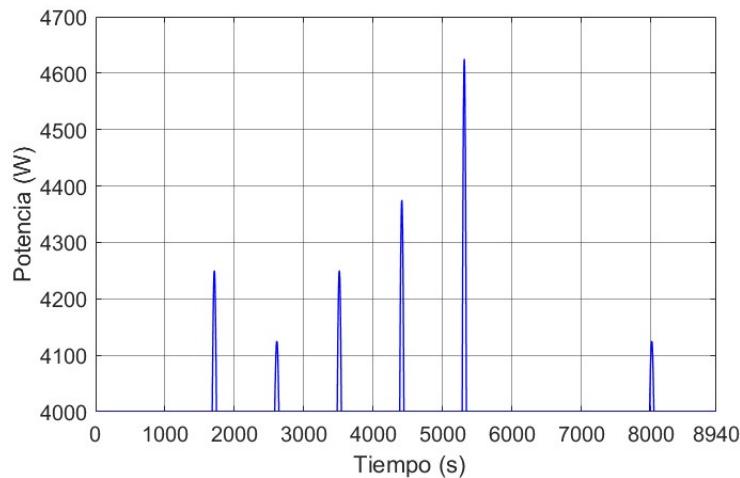


Figura 5.21 Potencia de entrada P_{in} en Simulink.

analizar convenientemente dichos casos.

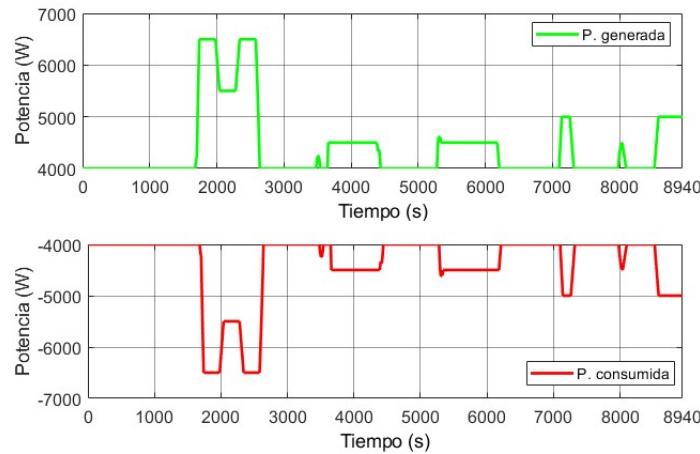


Figura 5.22 Balance de potencia.

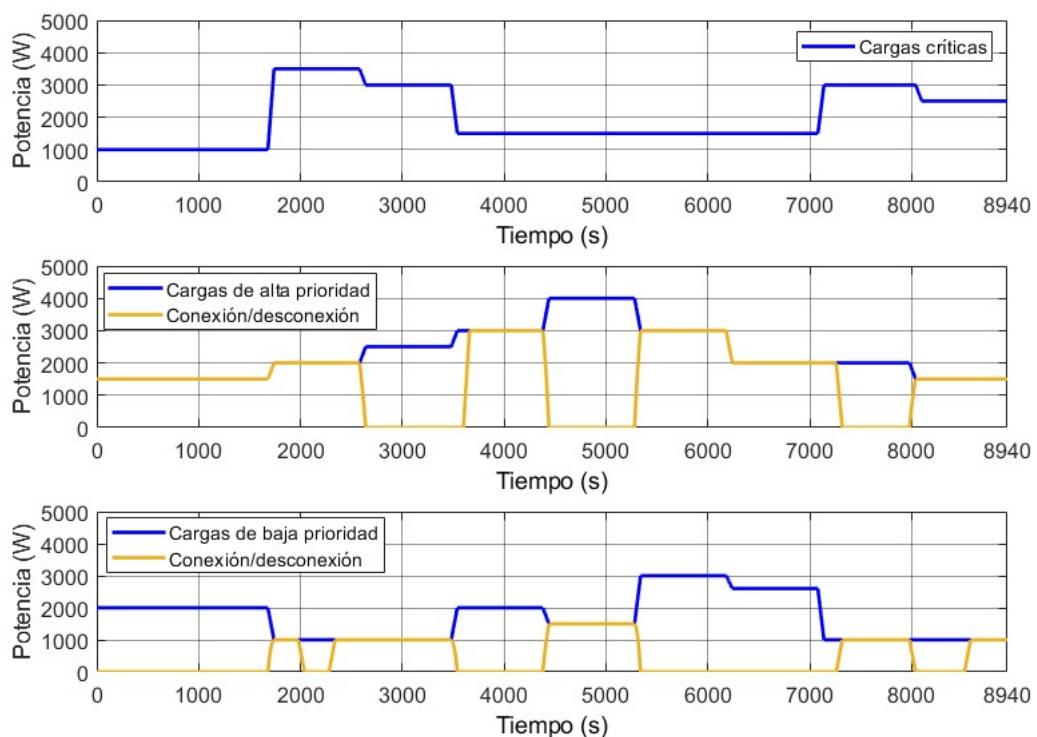


Figura 5.23 Conexión/desconexión de cargas.

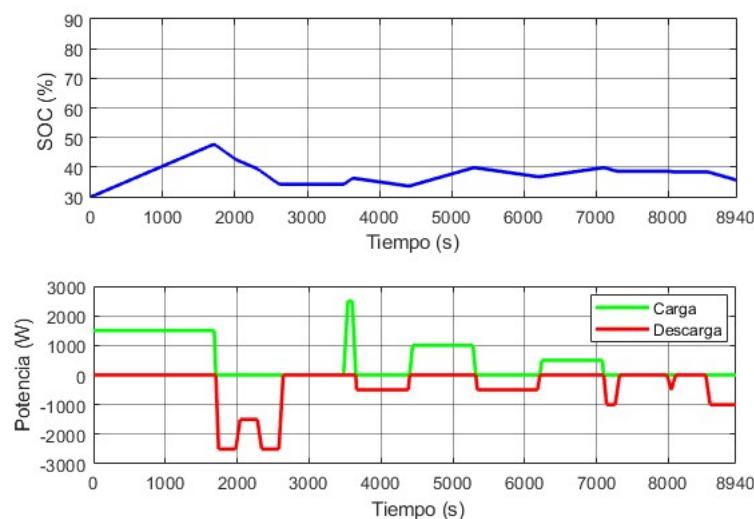


Figura 5.24 Evolución de la batería.

6 Conclusiones

En este trabajo se discute acerca del concepto MEA de las nuevas generaciones de aeronaves y presentándose una revisión general de la tecnología empleada. La llegada de las aeronaves MEA ha cambiado la forma en la que entendemos el funcionamiento de las aeronaves y constituye tan solo el primer paso de los grandes avances que están por llegar dentro del mundo de la aviación. El desarrollo de los MEA aporta soluciones al problema del cambio climático que se verá agravado en la próxima década, marcando el camino de la industria aeronáutica hacia la reducción del uso de combustibles fósiles y la emisión de gases de efecto invernadero. Además, abre las puertas a conceptos más avanzados como los AEA e incrementa la seguridad de los vuelos.

Por otro lado, se propone un entorno de trabajo MPC a nivel de sistema para optimizar el control y la gestión del almacenamiento de energía y la flexibilidad en el lado de la carga para un sistema MEA. El MPC conforma una estrategia de optimización muy útil para los sistemas eléctricos de potencia de las aeronaves que es capaz de adaptarse a la mayoría de exigencias impuestas por los requisitos de la gestión energética durante el vuelo de una aeronave. La investigación de estas técnicas es muy importante para la evolución simultánea de los MEA, debido a que una mayor presencia de dispositivos eléctricos requieren estrategias de control más avanzadas y complejas. El MPC brinda numerosas oportunidades de modelado y la posibilidad de ser complementado con técnicas de Inteligencia Artificial para obtener resultados más precisos. Es importante para este tipo de aplicaciones ya que permite la implementación de una optimización en tiempo real, con una capacidad de resolución más rápida que muchas otras técnicas de optimización, haciéndolo idóneo para el vuelo de una aeronave debido a los cortos períodos de tiempo y de muestreo en los que trabaja la microrred en comparación a otras aplicaciones.

Siguiendo los objetivos de control, se presentaron dos funciones objetivo diferentes, identificándose la importancia de las diferentes funciones objetivo y los horizontes de predicción mediante el análisis de los resultados de la simulación a través de los índices de evaluación propuestos. Respecto a los resultados, se pueden observar diferencias entre las funciones objetivos definidas, así como los índices para valorar los resultados:

1. Índice de desconexión de cargas

En general, se puede decir que la función $Obj1$ (4.1) tiene un mejor rendimiento manteniendo las cargas conectadas, dado que a partir de $N > 14$ la función $Obj2$ (4.6) reduce su rendimiento rápidamente después del punto de inflexión en $N = 14$. Sin embargo, para horizontes cortos $Obj2$ tiene un mejor rendimiento.

Por tanto, para horizontes largos es preferible el uso de $Obj1$, mientras que para horizontes más cortos, $N < 14$, $Obj2$ se obtienen ligeramente mejores resultados y es preferible su empleo.

En conclusión, la opción óptima para este índice sería usar $Obj2$ para $N < 14$, debido a que se obtienen mejores resultados y se ahorra tiempo de computación.

2. Índice de cambios de estado de los contactores.

En conclusión, para este índice se puede afirmar que $Obj2$ tiene un mejor rendimiento para reducir el cambio de estado de los contactores. La opción óptima sería tomar horizontes largos $15 \leq N \leq 25$ (después del punto de inflexión).

3. Índice de carga de la batería

Para este índice se demuestra que el rendimiento es mayor cuanto más largo es el horizonte de predicción. En general, $Obj2$ obtiene siempre mejores resultados, pero concretamente, a partir de $N > 14$, donde se produce el punto de inflexión, el rendimiento aumenta mucho y se prioriza el estado del SOC y carga la batería en mayor medida. Como solución final, se toma el $Obj2$ y horizontes largos, $N > 14$.

4. Índice de cambios de potencia de la batería

A partir del punto de inflexión en $N = 3$, el rendimiento de ambas funciones mejora, pero $Obj2$ obtiene mejores resultados gracias a J_{Pbatt} . Además, para horizontes largos los valores se estabilizan alrededor del mejor resultado, por lo que se concluye que la solución óptima es dicha función objetivo para horizontes largos.

5. Índice multiobjetivo

Para horizontes cortos entre $N > 3$ y el punto de inflexión alrededor de $N = 14$, $Obj2$ tiene un mejor rendimiento y un menor coste de computación. Para horizontes mayores, $Obj1$ obtiene mejores resultados. La elección de una función objetivo u otra dependerá de que objetivos se quieran priorizar, pero en general, se puede afirmar que $Obj2$ con un horizonte de predicción $N = 14$, es la solución óptima del problema, ya que aún no sobrepasa el punto de inflexión y no prioriza la batería desmesuradamente, obteniendo mejores resultados para todos los índices y constando de un tiempo de procesamiento menor.

Tras haber detallado el impacto que tienen los diferentes horizontes de predicción sobre cada uno de los índices de evaluación, ha quedado demostrada la aplicabilidad de dicha estrategia de control al sistema eléctrico de potencia de una aeronave MEA.

El abanico de posibilidades que abren los resultados del estudio es muy amplio. En primer lugar, se podría extender la configuración del sistema a todo el EPS de una aeronave, teniendo en cuenta los buses de alta tensión, de corriente alterna y fuentes de generación de energía secundarias. Además, se podría configurar el controlador MPC para hacer frente a más restricciones y escenarios de operación, como por ejemplo relaciones entre la potencia generada con el gasto de combustible, lo cual dotaría al modelo de una visión más realista. Otra ampliación posible podría ser la consideración de los efectos de la temperatura en el sistema o la reducción de la capacidad de la batería con el uso.

Por último, se podría continuar trabajando en el diseño del modelo en Simulink e implementar la estrategia de optimización programada en Python en un controlador con el objetivo de realizar la optimización en tiempo real y emular la EPS de una aeronave. Añadido a esto, algoritmos de inteligencia artificial podrían añadirse al controlador para la predicción de la demanda de potencia de las cargas según las condiciones de vuelo por medio de aprendizaje automático y redes neuronales, en vez de introducir un perfil de cargas fijo. Estas dos perspectivas podrían dar lugar a una continuación de este trabajo en la que seguir trabajando con los que llegar a obtener un controlador totalmente implementable en la microrred de una aeronave.

Apéndice A

Códigos de Python

Código A.1 Principal.py.

```
1 import pyomo.environ as pyo
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
5 import time
6 import random
7
8 from Modelos import build_model_1, build_model_2
9 from Funciones import prepare_loads_data, prepare_loads_disturbed,
    extract_horizon_solution, MHE_solution, MHE_disturbed,
    horizons_analysis, MPC_fun
10 from Gráficas import plot_power_balance, plot_demand_shedding, plot_SOC
    , plot_charge_discharge, plot_G_S_L_h, plot_G_delta_L_h,
    plot_G_SOC_h, plot_G_P_batt_h, plot_G_M0_h, plot_power_profile
11
12
13 start = time.time()
14
15 (P_L_crit_dic, P_L_HP_dic, P_L_LP_dic) = prepare_loads_data(0, 150)
16
17 P_L_crit = list(P_L_crit_dic.values())
18 P_L_HP    = list(P_L_HP_dic.values())
19 P_L_LP    = list(P_L_LP_dic.values())
20
21 ##### DISTURBED DEMAND DATA #####
22
23 (P_L_crit_dic, P_L_HP_dic, P_L_LP_dic) = prepare_loads_disturbed(0, 150)
24
25 P_L_crit = list(P_L_crit_dic.values())
26 P_L_HP    = list(P_L_HP_dic.values())
27 P_L_LP    = list(P_L_LP_dic.values())
```

```
28
29
30 x = np.arange(0, 150)
31
32 plot_power_profile(x, P_L_HP, P_L_LP, P_L_crit)
33
34 T = 150
35 cont = 0
36
37 ##### FOR ONE HORIZON
38 SOC_0, S_L_0, P_ch_0, P_disch_0 = 0.3, [1,0], 0, 0
39 horizon = 14
40
41 (model, P_in, S_L, SOC, P_ch, P_disch) = MHE_solution (T, cont, horizon,
        build_model_1, SOC_0, S_L_0, P_ch_0, P_disch_0)
42 obj = 1
43
44 plot_power_balance (obj, x, horizon, P_L_HP, P_L_LP, P_L_crit, P_in,
        S_L, P_ch, P_disch)
45 plot_demand_shedding (obj, x, horizon, P_L_HP, P_L_LP, P_L_crit, P_in,
        S_L, P_ch, P_disch)
46 plot_SOC (obj, x, horizon, SOC)
47 plot_charge_discharge (obj, x, horizon, P_ch, P_disch)
48
49 (model, P_in, S_L, SOC, P_ch, P_disch) = MHE_solution (T, cont, horizon,
        build_model_2, SOC_0, S_L_0, P_ch_0, P_disch_0)
50 obj = 2
51 plot_power_balance (obj, x, horizon, P_L_HP, P_L_LP, P_L_crit, P_in,
        S_L, P_ch, P_disch)
52 plot_demand_shedding (obj, x, horizon, P_L_HP, P_L_LP, P_L_crit, P_in,
        S_L, P_ch, P_disch)
53 plot_SOC (obj, x, horizon, SOC)
54 plot_charge_discharge (obj, x, horizon, P_ch, P_disch)
55
56 obj = 1
57 (model, P_in, S_L, SOC, P_ch, P_disch, P_L_crit, P_L_HP, P_L_LP,
        P_L_crit_dic, P_L_HP_dic, P_L_LP_dic) = MHE_disturbed(T, cont,
        horizon, build_model_1, SOC_0, S_L_0, P_ch_0, P_disch_0)
58
59 plot_power_profile(x, P_L_HP, P_L_LP, P_L_crit)
60 plot_power_balance (obj, x, horizon, P_L_HP, P_L_LP, P_L_crit, P_in,
        S_L, P_ch, P_disch)
61 plot_demand_shedding (obj, x, horizon, P_L_HP, P_L_LP, P_L_crit, P_in,
        S_L, P_ch, P_disch)
62 plot_SOC (obj, x, horizon, SOC)
63 plot_charge_discharge (obj, x, horizon, P_ch, P_disch)
64
65 ##### len(horizons) HORIZONS
#####
```

```

66
67 horizons = np.arange(1, 31, 1)
68
69 (G_S_L_h_obj1, G_delta_L_h_obj1, G_SOC_h_obj1, G_P_batt_h_obj1, G_MO_h_obj1)
    = horizons_analysis(x, horizons, T, cont, build_model_1,
    P_L_crit_dic, P_L_HP_dic, P_L_LP_dic, P_L_HP, P_L_LP, P_L_crit)
70 obj = 1
71
72 print('G_S_L_h      =', G_S_L_h_obj1)
73 print('G_delta_L_h =', G_delta_L_h_obj1)
74 print('G_SOC_h      =', G_SOC_h_obj1)
75 print('G_P_batt_h   =', G_P_batt_h_obj1)
76 print('G_MO_h       =', G_MO_h_obj1)
77
78 plot_G_S_L_h      (obj, horizons, G_S_L_h_obj1)
79 plot_G_delta_L_h (obj, horizons, G_delta_L_h_obj1)
80 plot_G_SOC_h      (obj, horizons, G_SOC_h_obj1)
81 plot_G_P_batt_h  (obj, horizons, G_P_batt_h_obj1)
82 plot_G_MO_h      (obj, horizons, G_MO_h_obj1)
83
84 (G_S_L_h_obj2, G_delta_L_h_obj2, G_SOC_h_obj2, G_P_batt_h_obj2,
    G_MO_h_obj2) = horizons_analysis(x, horizons, T, cont, build_model_2,
    P_L_crit_dic, P_L_HP_dic, P_L_LP_dic, P_L_HP, P_L_LP, P_L_crit)
85 obj = 2
86 plot_G_S_L_h      (obj, horizons, G_S_L_h_obj2)
87 plot_G_delta_L_h (obj, horizons, G_delta_L_h_obj2)
88 plot_G_SOC_h      (obj, horizons, G_SOC_h_obj2)
89 plot_G_P_batt_h  (obj, horizons, G_P_batt_h_obj2)
90 plot_G_MO_h      (obj, horizons, G_MO_h_obj2)
91
92 ##### MPC #####
93 SOC_0, S_L_0, P_ch_0, P_disch_0 = 0.3, [1,0], 0, 0
94 horizon = 150
95
96 obj = 1
97 MPC_fun(T, obj, build_model_1, horizon, SOC_0, S_L_0, P_ch_0, P_disch_0,
    P_L_crit_dic, P_L_HP_dic, P_L_LP_dic, x, P_L_HP, P_L_LP, P_L_crit)
98
99 obj = 2
100 MPC_fun(T, obj, build_model_3, horizon, SOC_0, S_L_0, P_ch_0, P_disch_0,
    P_L_crit_dic, P_L_HP_dic, P_L_LP_dic, x, P_L_HP, P_L_LP, P_L_crit)
101
102
103
104
105 end = time.time()
106 print('computation time =', end - start)

```

```
1 import pyomo.environ as pyo
2
3 def build_model_1(horizon, SOC_0, S_L_0, P_L_crit, P_L_HP, P_L_LP):
4
5     mpc = pyo.ConcreteModel()
6
7     ##### SACALAR PARAMETERS #####
8
9     mpc.N = pyo.Param(initialize=horizon)
10    mpc.T_s = pyo.Param(initialize=1) # 1 minuto
11
12    mpc.P_in_max = pyo.Param(initialize=4)
13
14    mpc.B_cap = pyo.Param(initialize=240)
15    mpc.L0 = pyo.Param(initialize=0.3)
16    mpc.HI = pyo.Param(initialize=0.9)
17
18    mpc.P_ch_max = pyo.Param(initialize=4)
19    mpc.P_disch_max = pyo.Param(initialize=4)
20    mpc.nu_ch = pyo.Param(initialize=0.9)
21    mpc.nu_disch = pyo.Param(initialize=0.9)
22
23    mpc.N_L = pyo.Param(initialize=2)
24
25    mpc.omega_S = pyo.Param(initialize=5)
26    mpc.omega_delta = pyo.Param(initialize=1)
27    mpc.omega_P_disch = pyo.Param(initialize=3)
28    mpc.omega_P_ch = pyo.Param(initialize=3)
29
30
31     ##### SETS #####
32
33    mpc.k = pyo.Set(initialize = range(mpc.N()))
34    mpc.i = pyo.Set(initialize = range(mpc.N_L()))
35
36     ##### INDEXED PARAMETERS #####
37
38    mpc.gamma_L = pyo.Param(mpc.i, initialize=[2,1])
39
40     ##### VARIABLES #####
41
42    mpc.P_in = pyo.Var(mpc.k, bounds=(0,4))
43
44    mpc.S_L = pyo.Var(mpc.i,mpc.k, domain=pyo.Binary)
45    # mpc.S_L = pyo.Var(mpc.i,mpc.k, bounds=(0,1))
46
47    mpc.lineal = pyo.Var(mpc.i,mpc.k, within=pyo.NonNegativeReals)
48
49    mpc.SOC = pyo.Var(mpc.k, bounds=(mpc.L0,mpc.HI))
```

```

50
51     mpc.P_ch      = pyo.Var(mpc.k, bounds=(0,4))
52     mpc.P_disch = pyo.Var(mpc.k, bounds=(0,4))
53
54     mpc.dseta_disch = pyo.Var(mpc.k, domain=pyo.Binary)
55     mpc.dseta_ch      = pyo.Var(mpc.k, domain=pyo.Binary)
56
57 ##### OBJECTIVE FUNCTION #####
58
59 def obj1_expression(mpc):
60
61     J_S_L = sum(mpc.gamma_L[i] * (1 - mpc.S_L[i,k]) / (mpc.N * (mpc.
62 gamma_L[0]+mpc.gamma_L[1])) for k in mpc.k for i in mpc.i)
63
64     J_delta_L = sum( mpc.lineal[i,k] / (mpc.N * mpc.N_L) for k in
65 mpc.k for i in mpc.i)
66
67     J_P_ch = sum((mpc.P_ch_max() - mpc.P_ch[k]) / (mpc.N() * mpc.
68 P_ch_max()) for k in mpc.k)
69
70     J_P_disch = sum(mpc.P_disch[k] / (mpc.N * mpc.P_disch_max) for k
71 in mpc.k)
72
73     return (mpc.omega_S*J_S_L + mpc.omega_delta*J_delta_L + mpc.
74 omega_P_ch*J_P_ch + mpc.omega_P_disch*J_P_disch)
75
76 mpc.obj1 = pyo.Objective(expr = obj1_expression)
77
78 ##### CONSTRAINTS #####
79
80 def power_balance(mpc, k):
81     return mpc.P_in[k] - mpc.P_ch[k] + mpc.P_disch[k] - mpc.S_L[0,k]
82     ]*P_L_HP[k] - mpc.S_L[1,k]*P_L_LP[k] - P_L_crit[k] == 0
83     mpc.power_balance_constraint = pyo.Constraint(mpc.k, rule=
power_balance)
84
85 def SOC(mpc, k):
86     if(k==0):
87         return SOC_0 + mpc.T_s*mpc.nu_ch*mpc.P_ch[k]/mpc.B_cap - mpc
88 .T_s*mpc.P_disch[k]/(mpc.nu_disch*mpc.B_cap) == mpc.SOC[k]
89     else:
90         return mpc.SOC[k-1] + mpc.T_s*mpc.nu_ch*mpc.P_ch[k]/mpc.
91 B_cap - mpc.T_s*mpc.P_disch[k]/(mpc.nu_disch*mpc.B_cap) == mpc.SOC[k]
92
93 mpc.SOC_constraint = pyo.Constraint(mpc.k, rule=SOC)
94
95 def battery_mode(mpc, k):
96     return mpc.dseta_ch[k] + mpc.dseta_disch[k] == 1
97     mpc.battery_mode_constraint = pyo.Constraint(mpc.k, rule=
battery_mode)

```

```

89
90     def charging_mode(mpc, k):
91         return mpc.P_ch[k] <= mpc.dseta_ch[k]*mpc.P_ch_max
92     mpc.charging_mode_constraint = pyo.Constraint(mpc.k, rule=
93         charging_mode)
94
95     def discharging_mode(mpc, k):
96         return mpc.P_disch[k] <= mpc.dseta_disch[k]*mpc.P_disch_max
97     mpc.discharging_mode_constraint = pyo.Constraint(mpc.k, rule=
98         discharging_mode)
99
100    def lineal_pos (mpc, i, k):
101        if(k==0):
102            return mpc.S_L[i,k] - S_L_0[i] <= mpc.lineal[i,k]
103        else:
104            return mpc.S_L[i,k] - mpc.S_L[i,k-1] <= mpc.lineal[i,k]
105    mpc.lineal_pos_constraint = pyo.Constraint(mpc.i,mpc.k, rule=
106        lineal_pos)
107
108    def lineal_neg (mpc, i, k):
109        if(k==0):
110            return S_L_0[i] - mpc.S_L[i,k] <= mpc.lineal[i,k]
111        else:
112            return mpc.S_L[i,k-1] - mpc.S_L[i,k] <= mpc.lineal[i,k]
113    mpc.lineal_neg_constraint = pyo.Constraint(mpc.i,mpc.k, rule=
114        lineal_neg)
115
116 ##### SOLVER #####
117
118 opt = pyo.SolverFactory('cplex')
119 results = opt.solve(mpc)
120
121 # mpc.pprint()
122 # print('Solver Status: ', results.solver.termination_condition)
123
124 def build_model_2(horizon, SOC_0, S_L_0, P_ch_0, P_disch_0, P_L_crit,
125     P_L_HP, P_L_LP):
126
127     mpc = pyo.ConcreteModel()
128
129 ##### SACALAR PARAMETERS #####
130
131     mpc.N = pyo.Param(initialize=horizon)
132     mpc.T_s = pyo.Param(initialize=1) # 1 minuto
133
134     mpc.P_in_max = pyo.Param(initialize=4)

```

```

134
135 mpc.B_cap = pyo.Param(initialize=240)
136 mpc.L0 = pyo.Param(initialize=0.3)
137 mpc.HI = pyo.Param(initialize=0.9)
138
139 mpc.P_ch_max      = pyo.Param(initialize=4)
140 mpc.P_disch_max = pyo.Param(initialize=4)
141 mpc.nu_ch      = pyo.Param(initialize=0.9)
142 mpc.nu_disch = pyo.Param(initialize=0.9)
143
144 mpc.N_L = pyo.Param(initialize=2)
145
146 mpc.omega_S      = pyo.Param(initialize=5)
147 mpc.omega_delta   = pyo.Param(initialize=1)
148 mpc.omega_P_disch = pyo.Param(initialize=3)
149 mpc.omega_P_ch    = pyo.Param(initialize=2.3)
150
151 mpc.omega_SOC     = pyo.Param(initialize=9)
152 mpc.omega_P_batt  = pyo.Param(initialize=0.75)
153
154 ##### SETS #####
155
156 mpc.k = pyo.Set(initialize = range(mpc.N()))
157 mpc.i = pyo.Set(initialize = range(mpc.N_L()))
158
159 ##### INDEXED PARAMETERS #####
160
161 mpc.gamma_L = pyo.Param(mpc.i, initialize=[2,1])
162
163 ##### VARIABLES #####
164
165 mpc.P_in      = pyo.Var(mpc.k, bounds=(0,4))
166 mpc.P_ch      = pyo.Var(mpc.k, bounds=(0,4))
167 mpc.P_disch   = pyo.Var(mpc.k, bounds=(0,4))
168 mpc.S_L       = pyo.Var(mpc.i,mpc.k, domain=pyo.Binary)
169 # mpc.S_L      = pyo.Var(mpc.i,mpc.k, bounds=(0,1)))
170 mpc.SOC       = pyo.Var(mpc.k, bounds=(mpc.L0,mpc.HI))
171
172 mpc.lineal_J_S      = pyo.Var(mpc.i,mpc.k, within=pyo.
173 NonNegativeReals)
174 mpc.lineal_J_P_batt = pyo.Var(mpc.k, within=pyo.NonNegativeReals)
175
176 mpc.dseta_disch = pyo.Var(mpc.k, domain=pyo.Binary)
177 mpc.dseta_ch    = pyo.Var(mpc.k, domain=pyo.Binary)
178
179 ##### OBJECTIVE FUNCTION #####
180
181 def obj2_expression(mpc):

```

```

182         J_S_L      = sum((mpc.gamma_L[i] * (1 - mpc.S_L[i,k])) / (mpc.N *
183                         (mpc.gamma_L[0]+mpc.gamma_L[1])) for k in mpc.k for i in mpc.i)
184
185         J_delta_L = sum(mpc.lineal_J_S[i,k] / (mpc.N * mpc.N_L) for k in
186                         mpc.k for i in mpc.i)
187
188         J_P_ch     = sum((mpc.P_ch_max() - mpc.P_ch[k]) / (mpc.N() * mpc.
189                         P_ch_max()) for k in mpc.k)
190
191         J_P_disch = sum(mpc.P_disch[k] / (mpc.N * mpc.P_disch_max) for k
192                         in mpc.k)
193
194         J_SOC      = sum((mpc.HI - mpc.SOC[k]) / (mpc.N * mpc.HI) for k
195                         in mpc.k)
196
197         J_P_batt   = sum(mpc.lineal_J_P_batt[k] / (mpc.N * (mpc.P_ch_max
198                         + mpc.P_disch_max)) for k in mpc.k)
199
200
201         return (mpc.omega_S*J_S_L + mpc.omega_delta*J_delta_L + mpc.
202             omega_P_ch*J_P_ch + mpc.omega_P_disch*J_P_disch + mpc.omega_SOC*
203             J_SOC + mpc.omega_P_batt*J_P_batt)
204
205         mpc.obj2 = pyo.Objective(expr = obj2_expression)
206
207
208 ###### CONSTRAINTS #####
209
210     def power_balance(mpc, k):
211         return mpc.P_in[k] - mpc.P_ch[k] + mpc.P_disch[k] - mpc.S_L[0,k]
212             ]*P_L_HP[k] - mpc.S_L[1,k]*P_L_LP[k] - P_L_crit[k] == 0
213         mpc.power_balance_constraint = pyo.Constraint(mpc.k, rule=
214             power_balance)
215
216     def SOC(mpc, k):
217         if(k==0):
218             return SOC_0 + mpc.T_s*mpc.nu_ch*mpc.P_ch[k]/mpc.B_cap - mpc
219             .T_s*mpc.P_disch[k]/(mpc.nu_disch*mpc.B_cap) == mpc.SOC[k]
220         else:
221             return mpc.SOC[k-1] + mpc.T_s*mpc.nu_ch*mpc.P_ch[k]/mpc.
222                 B_cap - mpc.T_s*mpc.P_disch[k]/(mpc.nu_disch*mpc.B_cap) == mpc.SOC[k
223                 ]
224         mpc.SOC_constraint = pyo.Constraint(mpc.k, rule=SOC)
225
226     def battery_mode(mpc, k):
227         return mpc.dseta_ch[k] + mpc.dseta_disch[k] == 1
228         mpc.battery_mode_constraint = pyo.Constraint(mpc.k, rule=
229             battery_mode)
230
231     def charging_mode(mpc, k):
232         return mpc.P_ch[k] <= mpc.dseta_ch[k]*mpc.P_ch_max

```

```

217     mpc.charging_mode_constraint = pyo.Constraint(mpc.k, rule=
218         charging_mode)
219
219     def discharging_mode(mpc, k):
220         return mpc.P_disch[k] <= mpc.dseta_disch[k]*mpc.P_disch_max
221     mpc.discharging_mode_constraint = pyo.Constraint(mpc.k, rule=
222         discharging_mode)
223
223     def lineal_J_S_pos (mpc, i, k):
224         if(k==0):
225             return mpc.S_L[i,k] - S_L_0[i] <= mpc.lineal_J_S[i,k]
226         else:
227             return mpc.S_L[i,k] - mpc.S_L[i,k-1] <= mpc.lineal_J_S[i,k]
228     mpc.lineal_J_S_pos_constraint = pyo.Constraint(mpc.i,mpc.k, rule=
229         lineal_J_S_pos)
230
230     def lineal_J_S_neg (mpc, i, k):
231         if(k==0):
232             return S_L_0[i] - mpc.S_L[i,k] <= mpc.lineal_J_S[i,k]
233         else:
234             return mpc.S_L[i,k-1] - mpc.S_L[i,k] <= mpc.lineal_J_S[i,k]
235     mpc.lineal_J_S_neg_constraint = pyo.Constraint(mpc.i,mpc.k, rule=
236         lineal_J_S_neg)
237
237     def lineal_J_P_batt_pos (mpc, k):
238         if(k==0):
239             return mpc.P_ch[k] - mpc.P_disch[k] - (P_ch_0 - P_disch_0)
240             <= mpc.lineal_J_P_batt[k]
241         else:
242             return mpc.P_ch[k] - mpc.P_disch[k] - (mpc.P_ch[k-1] - mpc.
243             P_disch[k-1]) <= mpc.lineal_J_P_batt[k]
244     mpc.lineal_J_P_batt_pos_constraint = pyo.Constraint(mpc.k, rule=
245         lineal_J_P_batt_pos)
246
246     def lineal_J_P_batt_neg (mpc, k):
247         if(k==0):
248             return P_ch_0 - P_disch_0 - (mpc.P_ch[k] - mpc.P_disch[k])
249             <= mpc.lineal_J_P_batt[k]
250         else:
251             return mpc.P_ch[k-1] - mpc.P_disch[k-1] - (mpc.P_ch[k] - mpc.
252             P_disch[k]) <= mpc.lineal_J_P_batt[k]
253     mpc.lineal_J_P_batt_neg_constraint = pyo.Constraint(mpc.k, rule=
254         lineal_J_P_batt_neg)
255
255 ##### SOLVER #####
256
256     opt = pyo.SolverFactory('cplex')
257     results = opt.solve(mpc)
258

```

```

257     # mpc pprint()
258     # print('Solver Status: ', results.solver.termination_condition)
259
260     return mpc

```

Código A.3 Funciones.py.

```

1 import pyomo.environ as pyo
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
5 import time
6 import random
7
8 from Modelos import build_model_1, build_model_3
9 from Gráficas import plot_power_balance, plot_demand_shedding, plot_SOC
10 , plot_charge_discharge, plot_G_S_L_h
11
12 def prepare_loads_data(cont, horizon): #EXTRACT LOADS DEMAND FOR
13     SELECTED HORIZON TO DICTIONARY
14
15     # READ CSV LOADS DEMAND
16     critical_loads_dic = pd.read_csv ('D:\\Critical_loads.csv', names =
17     ['critical loads'])
18     HP_loads_dic      = pd.read_csv ('D:\\HP_loads.csv', names = ['HP
19     loads'])
20     LP_loads_dic      = pd.read_csv ('D:\\LP_loads.csv', names = ['LP
21     loads'])
22
22     P_L_crit_dic = {}
23     P_L_HP_dic = {}
24     P_L_LP_dic = {}
25
26     P_L_crit_np_array = critical_loads_dic["critical loads"] [cont:(cont+
27     horizon)].to_numpy()
28     P_L_HP_np_array   = HP_loads_dic["HP loads"] [cont:(cont+horizon)].
29     to_numpy()
30     P_L_LP_np_array   = LP_loads_dic["LP loads"] [cont:(cont+horizon)].
31     to_numpy()
32
32     for k in range(horizon):
33         P_L_crit_dic[k] = P_L_crit_np_array[k]
34         P_L_HP_dic[k]   = P_L_HP_np_array[k]
35         P_L_LP_dic[k]   = P_L_LP_np_array[k]
36
36     return P_L_crit_dic, P_L_HP_dic, P_L_LP_dic
37
38 def prepare_loads_disturbed(cont, horizon):

```

```

35
36 critical_loads_dic = pd.read_csv ('D:\\Critical_loads.csv', names =
37 ['critical loads'])
38 HP_loads_dic      = pd.read_csv ('D:\\HP_loads.csv', names = ['HP
39 loads'])
40 LP_loads_dic      = pd.read_csv ('D:\\LP_loads.csv', names = ['LP
41 loads'])

42 P_L_crit_dic = {}
43 P_L_HP_dic = {}
44 P_L_LP_dic = {}

45 P_L_crit_np_array = critical_loads_dic["critical loads"] [cont:(cont+
46 horizon)].to_numpy()
47 P_L_HP_np_array   = HP_loads_dic["HP loads"] [cont:(cont+horizon)].
48 to_numpy()
49 P_L_LP_np_array   = LP_loads_dic["LP loads"] [cont:(cont+horizon)].
50 to_numpy()

51 for k in range(horizon):
52     P_L_crit_dic[k] = P_L_crit_np_array[k]*random.uniform(0.5, 1)
53     P_L_HP_dic[k]   = P_L_HP_np_array[k]*random.uniform(0.5, 1)
54     P_L_LP_dic[k]   = P_L_LP_np_array[k]*random.uniform(0.5, 1)

55
56 def extract_horizon_solution(model): # for N steps in predefined horizon
57
58 t      = []
59 P_in_aux  = []
60 S_L_aux  = [ [], [] ]
61 SOC_aux  = []
62 P_ch_aux = []
63 P_disch_aux= []

64 for k in model.k:
65     t.append(pyo.value(k))
66     P_in_aux.append(pyo.value(model.P_in[k]))
67     S_L_aux[0].append(pyo.value(model.S_L[0,k]))
68     S_L_aux[1].append(pyo.value(model.S_L[1,k]))
69     SOC_aux.append(pyo.value(model.SOC[k]))
70     P_ch_aux.append(pyo.value(model.P_ch[k]))
71     P_disch_aux.append(pyo.value(model.P_disch[k]))

72
73 return t, P_in_aux, S_L_aux, SOC_aux, P_ch_aux, P_disch_aux

74
75
76
77 def MHE_solution(T, cont, horizon, build_model, SOC_0, S_L_0, P_ch_0,
78 P_disch_0): #for T steps (complete optimization)

```

```
78
79     P_in, SOC, S_L, P_ch, P_disch = [], [], [[], []], [], []
80
81     while cont < T:
82
83         (P_L_crit_dic, P_L_HP_dic, P_L_LP_dic) = prepare_loads_data(cont,
84             horizon)
85
86         if build_model == build_model_1:
87             model = build_model(horizon, SOC_0, S_L_0, P_L_crit_dic,
88             P_L_HP_dic, P_L_LP_dic)
89         else:
90             model = build_model(horizon, SOC_0, S_L_0, P_ch_0, P_disch_0
91             , P_L_crit_dic, P_L_HP_dic, P_L_LP_dic)
92
93         (t, P_in_aux, S_L_aux, SOC_aux, P_ch_aux, P_disch_aux) =
94             extract_horizon_solution(model)
95
96         P_in.append(P_in_aux[0])
97         S_L[0].append(S_L_aux[0][0])
98         S_L[1].append(S_L_aux[1][0])
99         SOC.append(SOC_aux[0])
100        P_ch.append(P_ch_aux[0])
101        P_disch.append(P_disch_aux[0])
102
103        SOC_0 = SOC_aux[0]
104        S_L_0[0] = S_L_aux[0][0]
105        S_L_0[1] = S_L_aux[1][0]
106
107        cont += 1;
108
109
110
111 def MHE_disturbed(T, cont, horizon, build_model, SOC_0, S_L_0, P_ch_0,
112     P_disch_0): #for T steps (complete optimization)
113
114     P_in, SOC, S_L, P_ch, P_disch, P_L_crit, P_L_HP, P_L_LP = [], [],
115     [[], []], [], [], [], []
116     P_L_crit_dic, P_L_HP_dic, P_L_LP_dic = {}, {}, {}
117
118     while cont < T:
119
120         (P_L_crit_dic_aux, P_L_HP_dic_aux, P_L_LP_dic_aux) =
121             prepare_loads_disturbed(cont, horizon)
122
123         if build_model == build_model_1:
```

```

121         model = build_model(horizon, SOC_0, S_L_0, P_L_crit_dic_aux,
122 P_L_HP_dic_aux, P_L_LP_dic_aux)
123     else:
124         model = build_model(horizon, SOC_0, S_L_0, P_ch_0, P_disch_0
125 , P_L_crit_dic, P_L_HP_dic, P_L_LP_dic)
126
127     (t, P_in_aux, S_L_aux, SOC_aux, P_ch_aux, P_disch_aux) =
128 extract_horizon_solution(model)
129
130     P_in.append(P_in_aux[0])
131     S_L[0].append(S_L_aux[0][0])
132     S_L[1].append(S_L_aux[1][0])
133     SOC.append(SOC_aux[0])
134     P_ch.append(P_ch_aux[0])
135     P_disch.append(P_disch_aux[0])
136
137     SOC_0 = SOC_aux[0]
138     S_L_0[0] = S_L_aux[0][0]
139     S_L_0[1] = S_L_aux[1][0]
140
141     P_ch_0      = P_ch_aux[0]
142     P_disch_0   = P_disch_aux[0]
143
144     P_L_crit_aux = list(P_L_crit_dic_aux.values())
145     P_L_HP_aux   = list(P_L_HP_dic_aux.values())
146     P_L_LP_aux   = list(P_L_LP_dic_aux.values())
147
148     P_L_crit_dic[cont] = P_L_crit_aux[0]
149     P_L_HP_dic[cont]   = P_L_HP_aux[0]
150     P_L_LP_dic[cont]   = P_L_LP_aux[0]
151
152     P_L_crit.append(P_L_crit_aux[0])
153     P_L_HP.append(P_L_HP_aux[0])
154     P_L_LP.append(P_L_LP_aux[0])
155
156     cont += 1;
157
158 return model, P_in, S_L, SOC, P_ch, P_disch, P_L_crit, P_L_HP,
159 P_L_LP, P_L_crit_dic, P_L_HP_dic, P_L_LP_dic
160
161 def horizons_analysis(x, horizons, T, cont, build_model, P_L_crit_dic,
162 P_L_HP_dic, P_L_LP_dic, P_L_HP, P_L_LP, P_L_crit):
163
164     M = T
165     G_S_L_h, G_delta_L_h, G_SOC_h, G_P_batt_h, G_MO_h = [], [], [], [],
166     []
167
168     for h in horizons:
169
170         model = build_model(horizon, SOC_0, S_L_0, P_ch_0, P_disch_0
171 , P_L_crit_dic, P_L_HP_dic, P_L_LP_dic)
172
173         (t, P_in_aux, S_L_aux, SOC_aux, P_ch_aux, P_disch_aux) =
174 extract_horizon_solution(model)
175
176         P_in.append(P_in_aux[0])
177         S_L[0].append(S_L_aux[0][0])
178         S_L[1].append(S_L_aux[1][0])
179         SOC.append(SOC_aux[0])
180         P_ch.append(P_ch_aux[0])
181         P_disch.append(P_disch_aux[0])
182
183         SOC_0 = SOC_aux[0]
184         S_L_0[0] = S_L_aux[0][0]
185         S_L_0[1] = S_L_aux[1][0]
186
187         P_ch_0      = P_ch_aux[0]
188         P_disch_0   = P_disch_aux[0]
189
190         P_L_crit_aux = list(P_L_crit_dic_aux.values())
191         P_L_HP_aux   = list(P_L_HP_dic_aux.values())
192         P_L_LP_aux   = list(P_L_LP_dic_aux.values())
193
194         P_L_crit_dic[cont] = P_L_crit_aux[0]
195         P_L_HP_dic[cont]   = P_L_HP_aux[0]
196         P_L_LP_dic[cont]   = P_L_LP_aux[0]
197
198         P_L_crit.append(P_L_crit_aux[0])
199         P_L_HP.append(P_L_HP_aux[0])
200         P_L_LP.append(P_L_LP_aux[0])
201
202         cont += 1;
203
204     return M, G_S_L_h, G_delta_L_h, G_SOC_h, G_P_batt_h, G_MO_h

```

```

165     SOC_0, S_L_0, P_ch_0, P_disch_0 = 0.3, [1,0], 0, 0
166     G_S_L, G_delta_L, G_SOC, G_P_batt = 0, 0, 0, 0
167
168     (model, P_in, S_L, SOC, P_ch, P_disch) = MHE_solution (T, cont,
169     h, build_model, SOC_0, S_L_0, P_ch_0, P_disch_0)
170
171     # obj = 1
172     # plot_power_balance (obj, x, h, P_L_HP, P_L_LP, P_L_crit,
173     P_in, S_L, P_ch, P_disch)
174     # plot_demand_shedding (obj, x, h, P_L_HP, P_L_LP, P_L_crit,
175     P_in, S_L, P_ch, P_disch)
176     # plot_SOC (obj, x, h, SOC)
177     # plot_charge_discharge (obj, x, h, P_ch, P_disch)
178
179
180     SOC_0, S_L_0, P_ch_0, P_disch_0 = 0.3, [1,0], 0, 0
181     for k in range(T):
182
183         G_SOC = G_SOC + abs(0.9 - SOC[k]) / (M * 0.9)
184
185         if k > 0:
186             G_P_batt = G_P_batt + abs( (P_ch[k] - P_disch[k]) - (
187             P_ch[k-1] - P_disch[k-1]) ) / (M * (4 + 4))
188         else:
189             G_P_batt = G_P_batt + abs( (P_ch[k] - P_disch[k]) - (
190             P_ch_0 - P_disch_0) ) / (M * (4 + 4))
191
192         for i in range(len(S_L)):
193
194             G_S_L = G_S_L + ( (model.gamma_L[i] * (1 - S_L[i][k])) /(
195             M * (model.gamma_L[0]+model.gamma_L[1])))
196
197             if k==0:
198                 G_delta_L = G_delta_L + abs(S_L[i][k] - S_L_0[i]) /
199                 (M*model.N_L)
200             else:
201                 G_delta_L = G_delta_L + abs(S_L[i][k] - S_L[i][k-1])
202                 / (M*model.N_L)
203
204             G_S_L_h .append(G_S_L)
205             G_delta_L_h.append(G_delta_L)
206             G_SOC_h .append(G_SOC)
207             G_P_batt_h .append(G_P_batt)
208
209             nu_S      = 2.8
210             nu_delta  = 0.4
211             nu_SOC    = 1
212             nu_P_batt = 0.4

```

```

206         G_MO_h.append( nu_S*G_S_L + nu_delta*G_delta_L + nu_SOC*G_SOC +
207             nu_P_batt*G_P_batt )
208
209     return G_S_L_h, G_delta_L_h, G_SOC_h, G_P_batt_h, G_MO_h
210
211 def MPC_fun(T, obj, build_model, horizon, SOC_0, S_L_0, P_ch_0,
212     P_disch_0, P_L_crit_dic, P_L_HP_dic, P_L_LP_dic, x, P_L_HP, P_L_LP,
213     P_L_crit):
214
215     if build_model == build_model_1:
216         model = build_model(horizon, SOC_0, S_L_0, P_L_crit_dic,
217             P_L_HP_dic, P_L_LP_dic)
218     else:
219         model = build_model(horizon, SOC_0, S_L_0, P_ch_0, P_disch_0,
220             P_L_crit_dic, P_L_HP_dic, P_L_LP_dic)
221
222     (t, P_in, S_L, SOC, P_ch, P_disch) = extract_horizon_solution(model)
223
224     G_S_L, G_delta_L, G_SOC, G_P_batt, G_MO = 0, 0, 0, 0, 0
225     M=T
226
227     for k in range(T):
228
229         G_SOC = G_SOC + abs(0.9 - SOC[k]) / (M * 0.9)
230
231         if k > 0:
232             G_P_batt = G_P_batt + abs( (P_ch[k] - P_disch[k]) - (P_ch[k-1] - P_disch[k-1]) ) / (M * (4 + 4))
233         else:
234             G_P_batt = G_P_batt + abs( (P_ch[k] - P_disch[k]) - (P_ch_0 - P_disch_0) ) / (M * (4 + 4))
235
236         for i in range(len(S_L)):
237
238             G_S_L = G_S_L + ( (model.gamma_L[i] * (1 - S_L[i][k]))/(M * (model.gamma_L[0]+model.gamma_L[1])))
239
240             if k==0:
241                 G_delta_L = G_delta_L + abs(S_L[i][k] - S_L_0[i]) / (M*model.N_L)
242             else:
243                 G_delta_L = G_delta_L + abs(S_L[i][k] - S_L[i][k-1]) / (M*model.N_L)
244
245             nu_S      = 2.8
246             nu_delta  = 0.4
247             nu_SOC    = 1
248             nu_P_batt = 0.4

```

```

246
247     G_MO = nu_S*G_S_L + nu_delta*G_delta_L + nu_SOC*G_SOC +
248     nu_P_batt*G_P_batt
249
250     plot_power_balance    (obj, x, horizon, P_L_HP, P_L_LP, P_L_crit,
251     P_in, S_L, P_ch, P_disch)
252     plot_demand_shedding (obj, x, horizon, P_L_HP, P_L_LP, P_L_crit,
253     P_in, S_L, P_ch, P_disch)
254     plot_SOC              (obj, x, horizon, SOC)
255     plot_charge_discharge (obj, x, horizon, P_ch, P_disch)
256
257     print(G_S_L)
258     print(G_delta_L)
259     print(G_SOC)
260     print(G_P_batt)
261     print(G_MO)

```

Código A.4 Gráficas.py.

```

1 import pyomo.environ as pyo
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
5 import matplotlib
6 import time
7
8 def plot_power_balance(obj, x, horizon, P_L_HP, P_L_LP, P_L_crit, P_in,
9     S_L, P_ch, P_disch):
10
11     plt.figure(11, figsize=(10, 5)) # POWER BALANCE
12
13     P_generated = []
14     for i in range(150): P_generated.append( P_in[i] + P_disch[i] )
15     plt.subplot(211)
16     plt.step(x, P_generated, color='green')
17     plt.xticks(np.arange(0, 160, 25))
18     plt.ylabel('P generada [kW]', fontsize=15)
19     plt.suptitle("Balance de portencia (Obj{0}, horizonte = {1})" .
format(obj, horizon), fontsize=20)
20     # plt.suptitle("Balance de portencia (Obj{0})" .format(obj),
21     # fontsize=20)
22     plt.grid(True)
23     plt.tight_layout()
24
25
26     P_consumed = []
27     for i in range(150): P_consumed.append((-1)*(P_ch[i] + np.multiply(
28         S_L[0], P_L_HP)[i] + np.multiply(S_L[1], P_L_LP)[i] + P_L_crit[i]))

```

```

27 plt.subplot(212)
28 plt.step(x, P_consumed, color='red')
29 # plt.xlabel('Tiempo [min] \n(b)', fontsize=20)
30 plt.xlabel('Tiempo [min] \n(b)', fontsize=20)
31 plt.xticks(np.arange(0, 160, 25))
32 plt.ylabel('P consumida [kW]', fontsize=15)
33 plt.grid(True)
34 plt.tight_layout()
35
36 plt.show()
37
38 return
39
40 def plot_demand_shedding(obj, x, horizon, P_L_HP, P_L_LP, P_L_crit, P_in,
41 , S_L, P_ch, P_disch):
42
43     plt.figure(figsize=(10, 7.5)) # POWER DEMAND / SHEDDING
44
45     plt.subplot(311)
46     plt.step(x, P_L_crit, label='Cargas críticas')
47     plt.xticks(np.arange(0, 160, 25))
48     plt.xlim(0)
49     plt.ylabel('Potencia[kW]', fontsize=15)
50     plt.yticks(np.arange(0, 4.1, 0.5))
51     plt.legend(fontsize='large')
52     plt.grid(True)
53     plt.suptitle("Desconexión de cargas (Obj{0}, N = {1})".format(obj,
54 horizon), fontsize=20)
55     # plt.suptitle("Desconexión de cargas (Obj{0})".format(obj),
56     # fontsize=20)
57
58     plt.subplot(312)
59     plt.step(x, P_L_HP, label='Cargas de alta prioridad')
60     plt.step(x, np.multiply(P_L_HP, S_L[0]), label='Con desconexión')
61     plt.xticks(np.arange(0, 160, 25))
62     plt.xlim(0)
63     plt.ylabel('Potencia[kW]', fontsize=15)
64     # plt.xlabel('\n(b)')
65     plt.yticks(np.arange(0, 4.1, 0.5))
66     plt.legend(loc = 'upper left', fontsize='large')
67     plt.grid(True)
68     plt.suptitle("Desconexión de cargas (Obj{0}, N = {1})".format(obj,
69 horizon), fontsize=20)
70
71     plt.subplot(313)
72     plt.step(x, P_L_LP, label='Cargas de baja prioridad')
73     plt.step(x, np.multiply(P_L_LP, S_L[1]), label='Con desconexión')
74     plt.xlabel('Tiempo (min) \n(b)', fontsize=15)
75     # plt.xlabel('Tiempo (min)', fontsize=15)
76     plt.xticks(np.arange(0, 160, 25))

```

```
73     plt.xlim(0)
74     plt.ylabel('Potencia[kW]', fontsize=15)
75     plt.yticks(np.arange(0, 4.1, 0.5))
76     plt.legend(fontsize='large')
77     plt.grid(True)
78     plt.tight_layout()
79
80     plt.show()
81
82     return
83
84 def plot_demand_shedding(obj, x, horizon, P_L_HP, P_L_LP, P_L_crit, P_in
85 , S_L, P_ch, P_disch):
86
87     plt.figure(figsize=(10, 5.5)) # POWER DEMAND / SHEDDING
88
89     plt.subplot(211)
90     plt.step(x, P_L_HP, label='Cargas de alta prioridad')
91     plt.step(x, np.multiply(P_L_HP, S_L[0]), label='Con desconexión')
92     plt.xticks(np.arange(0, 160, 25))
93     plt.xlim(0)
94     plt.ylabel('Potencia[kW]', fontsize=15)
95     # plt.xlabel('\n(b)')
96     plt.yticks(np.arange(0, 4.1, 0.5))
97     plt.legend(loc = 'upper left', fontsize='large')
98     plt.grid(True)
99     plt.title("Desconexión de cargas (Obj{0}, N = {1})".format(obj,
100 horizon), fontsize=20)
101
102     plt.subplot(212)
103     plt.step(x, P_L_LP, label='Cargas de baja prioridad')
104     plt.step(x, np.multiply(P_L_LP, S_L[1]), label='Con desconexión')
105     # plt.xlabel('Tiempo (min) \n(b)', fontsize=15)
106     plt.xlabel('Tiempo (min) \n(b)', fontsize=15)
107     # plt.xlabel('Tiempo (min)', fontsize=15)
108     plt.xticks(np.arange(0, 160, 25))
109     plt.xlim(0)
110     plt.ylabel('Potencia[kW]', fontsize=15)
111     plt.yticks(np.arange(0, 4.1, 0.5))
112     plt.legend(fontsize='large')
113     plt.grid(True)
114     plt.tight_layout()
115
116     plt.show()
117
118     return
119
120 def plot_SOC(obj, x, horizon, SOC):
```

```

121 plt.figure(figsize=(10,5)) # SOC
122 plt.plot(x, SOC)
123 # plt.xlabel('Tiempo [min] \n(b)', fontsize=20)
124 plt.xlabel('Tiempo [min] \n(b)', fontsize=20)
125 # plt.xlabel('Tiempo [min]', fontsize=20)
126
127 plt.xticks(np.arange(0, 160, 25))
128 plt.xlim(0,150)
129 plt.ylabel('SOC', fontsize=15)
130 plt.ylim(0.2, 1)
131 plt.grid(True)
132 plt.suptitle("SOC (Obj{0}, horizonte = {1})".format(obj, horizon),
133               fontsize=20)
134 # plt.suptitle("SOC (Obj{0})".format(obj), fontsize=20)
135 # plt.suptitle("SOC", fontsize=20)
136
137 LO = []
138 for i in range(len(x)): LO.append(0.3)
139 HI = []
140 for i in range(len(x)): HI.append(0.9)
141 plt.plot(x, HI, color='red')
142 plt.plot(x, LO, color='red')
143 plt.tight_layout()
144
145 plt.show()
146
147
148
149 def plot_charge_discharge(obj, x, horizon, P_ch, P_disch):
150
151 plt.figure(figsize=(10,5)) # CHARGE/DISCHARGE
152
153 plt.step(x, P_ch, color='green', label='Potencia de carga')
154 plt.xticks(np.arange(0, 160, 25))
155 plt.xlim(0,150)
156 plt.grid(True)
157
158
159 P_disch_neg = np.multiply(P_disch, -1)
160 plt.step(x, P_disch_neg, color='red', label='Potencia de descarga')
161 plt.xticks(np.arange(0, 160, 25))
162 plt.xlim(0,150)
163 # plt.xlabel('Tiempo [min] \n(b)', fontsize=20)
164 plt.xlabel('Tiempo [min] \n(b)', fontsize=20)
165 # plt.xlabel('Tiempo [min]', fontsize=20)
166 plt.ylabel('Potencia[kW]', fontsize=15)
167 plt.legend(loc='upper right', fontsize='large')
168 plt.suptitle("Potencia de la batería (Obj{0}, horizonte = {1})".
169               format(obj, horizon), fontsize=20)

```

```
169     # plt.suptitle("Potencia de la batería (Obj{0})".format(obj),  
170     # fontsize=20)  
171     # plt.suptitle("Potencia de la batería", fontsize=20)  
172     plt.tight_layout()  
173  
174     plt.show()  
175  
176 def plot_G_S_L_h(obj, horizons, G_S_L_h):  
177  
178     plt.figure(1, figsize=(10,5))  
179  
180     plt.plot(horizons, G_S_L_h, marker='.', markersize=10, label="Obj{0}  
181     ".format(obj))  
182     plt.xlabel('Horizonte (N)')  
183     plt.xticks(np.arange(0, 31, 5))  
184     plt.ylabel(r'$G_{S_{Li}}$', fontsize=20)  
185     plt.yticks(np.arange(0.35, 0.75, 0.05))  
186     plt.grid(True)  
187     plt.title('Índice de desconexión de cargas', fontsize=20)  
188     plt.legend(fontsize='xx-large')  
189  
190     # plt.show()  
191  
192     return  
193  
194 def plot_G_delta_L_h(obj, horizons, G_delta_L_h):  
195  
196     plt.figure(2, figsize=(10,5))  
197  
198     plt.plot(horizons, G_delta_L_h, marker='.', label="Obj{0}".format(  
199     obj))  
200     plt.xlabel('Horizonte (N)')  
201     plt.xticks(np.arange(0, 31, 5))  
202     plt.ylabel(r'$G_{\delta Li}$', fontsize=20)  
203     plt.grid(True)  
204     plt.legend(fontsize='xx-large')  
205     plt.title('Índice de cambio de estado de los contactores', fontsize  
206     =20)  
207  
208     # plt.show()  
209  
210     return  
211  
212  
213 def plot_G_SOC_h(obj, horizons, G_SOC_h):  
214  
215     plt.figure(4, figsize=(10,5))  
216  
217     plt.plot(horizons, G_SOC_h, marker='.', label="Obj{0}".format(obj))  
218     plt.xlabel('Horizonte (N)')
```

```

215 plt.xticks(np.arange(0, 31, 5))
216 plt.ylabel(r'$G_{SOC}$', fontsize=15)
217 plt.yticks(np.arange(0.2, 0.71, 0.1))
218 plt.grid(True)
219 plt.title('Índice de carga de la batería', fontsize=20)
220 plt.legend(fontsize='xx-large')
221
222 # plt.show()
223
224 return
225
226 def plot_G_P_batt_h(obj, horizons, G_P_batt_h):
227
228 plt.figure(3, figsize=(10,5))
229
230 plt.plot(horizons, G_P_batt_h, marker='.', markersize=10, label="Obj{0}".format(obj))
231 plt.xlabel('Horizonte (N)')
232 plt.xticks(np.arange(0, 31, 5))
233 plt.ylabel(r'$G_{Pbatt}$', fontsize=15)
234 plt.yticks(np.arange(0.01, 0.08, 0.01))
235 plt.grid(True)
236 plt.title('Índice de cambio de potencia de la batería', fontsize=20)
237 plt.legend(fontsize='xx-large')
238
239 # plt.show()
240
241 return
242
243 def plot_G_MO_h(obj, horizons, G_MO_h):
244
245 plt.figure(5, figsize=(10,5))
246
247 plt.plot(horizons, G_MO_h, marker='.', label="Obj{0}".format(obj))
248 plt.xlabel('Horizonte (N)')
249 plt.xticks(np.arange(0, 31, 5))
250 plt.ylabel(r'$G_{MO}$', fontsize=15)
251 plt.grid(True)
252 plt.title('Índice multiobjetivo', fontsize=20)
253 plt.legend(fontsize='xx-large')
254
255 # plt.show()
256
257 return
258
259
260 def plot_power_profile(x, P_L_HP, P_L_LP, P_L_crit):
261
262 plt.figure(figsize=(10, 7.5)) # POWER DEMAND / SHEDDING
263

```

```
264     plt.subplot(311)
265     plt.step(x, P_L_crit, label='Cargas críticas')
266     plt.xticks(np.arange(0, 160, 25))
267     plt.xlabel('Tiempo (min) \n(a)')
268     plt.ylabel('Potencia[kW]')
269     plt.yticks(np.arange(0, 4.1, 0.5))
270     plt.legend()
271     plt.grid(True)
272
273     plt.subplot(312)
274     plt.step(x, P_L_HP, label='Cargas de alta prioridad')
275     plt.xticks(np.arange(0, 160, 25))
276     plt.xlim(0)
277     plt.ylabel('Potencia[kW]')
278     plt.xlabel('Tiempo (min) \n(b)')
279     plt.yticks(np.arange(0, 4.1, 0.5))
280     plt.legend(loc = 'upper left')
281     plt.grid(True)
282
283     plt.subplot(313)
284     plt.step(x, P_L_LP, label='Cargas de baja prioridad')
285     plt.xlabel('Tiempo (min) \n(c)')
286     plt.xticks(np.arange(0, 160, 25))
287     plt.xlim(0)
288     plt.ylabel('Potencia[kW]')
289     plt.yticks(np.arange(0, 4.1, 0.5))
290     plt.legend()
291     plt.grid(True)
292     plt.tight_layout()
293
294     plt.show()
295
296     return
```

Índice de Figuras

1.1	Crecimiento histórico del tráfico aéreo comercial mundial. Los datos proceden de la Organización de Aviación Civil Internacional y de Airbus [3]	1
1.2	Esquema del camino hacia la reducción del CO ₂ hasta 2050 [3]	2
1.3	Emisiones de CO ₂ en 2018 por operaciones y clase de avión [5]	2
1.4	El Vickers Valiant usaba energía eléctrica para los controles de vuelo primario y otras funciones	4
1.5	Esquema de la distribución de energía convencional [8]	5
1.6	Esquema de la distribución de energía en MEA (Con sangrado de aire) [8]	5
1.7	Arquitectura del B787 sin sangrado de aire [9]	6
1.8	Evolución de la necesidad de energía eléctrica (gris: aviones de corto a medio alcance. negro: aviones de medio a largo alcance) [11]	7
1.9	Elementos de la microrred a bordo [12]	7
2.1	sistema CF IDG [13]	10
2.2	Motor turbofán de doble eje [13]	11
2.3	sistema VSCF con DC link [13]	11
2.4	Cicloconvertidor VSCF [14]	11
2.5	sistema del generador VF [13]	11
2.6	(a) APU tradicional con energía eléctrica, neumática e hidráulica. (b) APU eléctrica que ilustra el MES [15]	13
2.7	RAT	13
2.8	Subsistemas eléctricos en MEA [16]	14
2.9	(a) ECS con sangrado de aire. (b) MEA ECS [15]	15
2.10	Diagrama del sistema de un EMA [18]	16
2.11	Diagrama del sistema de un EHA [18]	16
2.12	Actuador electromecánico de alta potencia [18]	16
2.13	EPS con bus AC primario de 230 V con frecuencia variable 358-800 Hz. Cargas conectadas a los buses correspondientes [19]	18
2.14	Arquitectura MOET MEA EPS [12]	18
2.15	Arquitectura MEA EPS de bus único [12]	19
2.16	(a) Distribución centralizada. (b) Distribución remota [15]	20
3.1	Ejemplo de sistema reactivo [21]	22
3.2	Modelo del FMS usado para programar la lógica del controlador de supervisión en [21]	22
3.3	Arquitectura de una red neuronal simple [23]	24
3.4	Red neuronal entrenada [24]	25
3.5	Diagrama de control retroalimentado	27

3.6	Esquema Model Predictive control (MPC)	28
4.1	Arquitectura del EPS bajo estudio [1]	34
4.2	Optimización online MILP-MPC con horizonte móvil [1]	36
4.3	Diseño en Simulink	44
4.4	Arquitectura de la EPS simplificada del modelo en Simulink	44
5.1	Perfil de cargas: (a) críticas, (b) alta prioridad y (c) baja prioridad	48
5.2	Comparación del índice normalizado de desconexión de cargas para diferentes funciones objetivo y horizontes de predicción	49
5.3	Comparación del índice normalizado de cambio de estado de contactores para diferentes funciones objetivo y horizontes de predicción	50
5.4	Comparación del índice normalizado de nivel de carga de la batería para diferentes funciones objetivo y horizontes de predicción	51
5.5	Comparación del índice normalizado de cambio de potencia de la batería para diferentes funciones objetivo y horizontes de predicción	52
5.6	Comparación del índice multiobjetivo para diferentes funciones objetivo y horizontes de predicción	53
5.7	Balance de potencia con $N = 10$ para: (a) Obj_1 , (b) Obj_2	54
5.8	Conexión/desconexión de cargas con $N = 10$ para: (a) Obj_1 , (b) Obj_2	55
5.9	Evolución de la batería con $N = 10$ para: (a) Obj_1 , (b) Obj_2	55
5.10	Conexión/desconexión de cargas con $N = 8$ para: (a) Obj_1 , (b) Obj_2	56
5.11	Conexión/desconexión de cargas con $N = 20$ para: (a) Obj_1 , (b) Obj_2	56
5.12	Evolución del SOC y de la potencia de carga y descarga de la batería con $N = 15$ para: (a) Obj_1 , (b) Obj_2	57
5.13	Desconexión de cargas para: (a) controlador MPC-MHE con $N = 30$, (b) optimización simple con $N = 150$	57
5.14	Evolución de la batería para: (a) controlador MPC-MHE con $N = 30$, (b) optimización simple con $N = 150$	58
5.15	Desconexión de cargas para un estado de carga inicial igual a $SOC_0 = 0.6$	58
5.16	Evolución del estado de carga a lo largo del tiempo partiendo de un estado de carga inicial $SOC_0 = 0.6$	59
5.17	Potencia de carga y descarga para un estado inicial de carga de la batería $SOC_0 = 0.6$	59
5.18	Demandas de potencia con desconexión parcial de cargas para: (a) Obj_1 con $N=30$, (b) Obj_2 con $N=14$	60
5.19	Desconexión parcial de cargas para: (a) Obj_1 con $N=30$, (b) Obj_2 con $N=14$	60
5.20	Evolución de la batería con desconexión parcial de cargas para: (a) Obj_1 con $N=30$, (b) Obj_2 con $N=14$	61
5.21	Potencia de entrada P_{in} en Simulink	61
5.22	Balance de potencia	62
5.23	Conexión/desconexión de cargas	62
5.24	Evolución de la batería	63

Índice de Tablas

4.1	Parámetros	35
4.2	Variables continuas	35
4.3	Variables Binarias	35
5.1	Valores de los parámetros del EPS	47
5.2	Valores de los pesos del modelo	48
5.3	Valores de los pesos del índice multiobjetivo	53

Índice de Códigos

4.1	Primera función objetivo	41
4.2	Segunda función objetivo	41
4.3	Restricciones	42
A.1	Principal.py	67
A.2	Modelos.py	69
A.3	Funciones.py	76
A.4	Gráficas.py	82

Bibliografía

- [1] X. Wang, J. Atkin, N. Bazmohammadi, S. Bozhko, and J. M. Guerrero, “Optimal Load and Energy Management of Aircraft Microgrids Using Multi-Objective Model Predictive Control,” *Sustainability*, vol. 13, no. 24, 2021. [Online]. Available: <https://www.mdpi.com/2071-1050/13/24/13907>
- [2] P. J. Ansell and K. S. Haran, “Electrified Airplanes: A Path to Zero-Emission Air Travel,” *IEEE Electrification Magazine*, vol. 8, no. 2, pp. 18–26, 2020.
- [3] “Aircraft Technology Roadmap to 2050,” <https://www.iata.org/contentassets/8d19e716636a47c184e7221c77563c93/Technology-roadmap-2050.pdf>.
- [4] U. N. C. Change, “El acuerdo de parís,” <https://unfccc.int/es/process-and-meetings/the-paris-agreement/el-acuerdo-de-paris>.
- [5] B. Graver, K. Zhang, and D. Rutherford, “CO₂ emissions from commercial aviation, 2018,” *The International Council on Clean Transportation*, Sep. 2019. [Online]. Available: <https://theicct.org/publication/co2-emissions-from-commercial-aviation-2018/#:~:text=CO2%20emissions%20from%20all,over%20the%20past%20five%20years>.
- [6] H. Schefer, L. Fauth, T. H. Kopp, R. Mallwitz, J. Fribe, and M. Kurrat, “Discussion on Electric Power Supply Systems for All Electric Aircraft,” *IEEE Access*, vol. 8, pp. 84 188–84 216, 2020.
- [7] G. E. Tagge and L. A. Irish, “Systems Study for an Integrated Digital/Electric Aircraft (IDEA),” *BOEING COMMERCIAL AIRPLANE CO SEATTLE WA*, Jan. 1985. [Online]. Available: <https://apps.dtic.mil/sti/citations/ADA304177>
- [8] J. Seok, I. Kolmanovsky, and A. Girard, “Integrated/coordinated control of aircraft gas turbine engine and electrical power system: Towards large electrical load handling,” in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 3183–3189.
- [9] M. Sinnett, “787 No-Bleed Systems: Saving Fuel and Enhancing Operational Efficiencies,” *Boeing*. [Online]. Available: https://www.boeing.com/commercial/aeromagazine/articles/qtr_4_07/AERO_Q407_article2.pdf
- [10] U. of Cambridge, “Watts up - aeroplanes go hybrid-electric,” Dec. 2014, <https://www.cam.ac.uk/research/news/watts-up-aeroplanes-go-hybrid-electric>.
- [11] V. Madonna, P. Giangrande, and M. Galea, “Electrical Power Generation in Aircraft: Review, Challenges, and Opportunities,” *IEEE Transactions on Transportation Electrification*, vol. 4, no. 3, pp. 646–659, 2018.

- [12] G. Buticchi, S. Bozhko, M. Liserre, P. Wheeler, and K. Al-Haddad, “On-Board Microgrids for the More Electric Aircraft—Technology Review,” *IEEE Transactions on Industrial Electronics*, vol. 66, no. 7, pp. 5588–5599, 2019.
- [13] V. Madonna, P. Giangrande, and M. Galea, “Electrical power generation in aircraft: Review, challenges, and opportunities,” *IEEE Transactions on Transportation Electrification*, vol. 4, no. 3, pp. 646–659, 2018.
- [14] A. AbdElhafez and A. Forsyth, “A Review of More-Electric Aircraft,” *International Conference on Aerospace Sciences and Aviation Technology*, vol. 13, no. AEROSPACE SCIENCES & AVIATION TECHNOLOGY, ASAT- 13, May 26 – 28, 2009, pp. 1–13, 2009, publisher: The Military Technical College _eprint: https://asat.journals.ekb.eg/article_23726_d651e3d27a968a3f2f5dbc6112c742c2.pdf. [Online]. Available: https://asat.journals.ekb.eg/article_23726.html
- [15] B. Sarlioglu and C. T. Morris, “More Electric Aircraft: Review, Challenges, and Opportunities for Commercial Transport Aircraft,” *IEEE Transactions on Transportation Electrification*, vol. 1, no. 1, pp. 54–64, 2015.
- [16] “Infografía / El Avión Más Eléctrico,” <https://aertecsolutions.com/multimedia/infografia-meavion-mas-electrico/>.
- [17] I. Munuswamy and P. W. Wheeler, “Electric DC WIPS: All electric aircraft,” in *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)*, 2017, pp. 1–5.
- [18] A. Boglietti, A. Cavagnino, A. Tenconi, S. Vaschetto, and P. di Torino, “The safety critical electric machines and drives in the more electric aircraft: A survey,” in *2009 35th Annual Conference of IEEE Industrial Electronics*, 2009, pp. 2587–2594.
- [19] A. Barzkar and M. Ghassemi, “Electric Power Systems in More and All Electric Aircraft: A Review,” *IEEE Access*, vol. 8, pp. 169 314–169 332, 2020.
- [20] “More Open Electrical Technologies,” <https://cordis.europa.eu/project/id/30861>.
- [21] C. Spagnolo, S. Sumsurooah, and S. Bozcko, “Advanced smart grid power distribution system for More Electric Aircraft application,” in *2019 International Conference on Electrotechnical Complexes and Systems (ICOECS)*, 2019, pp. 1–6.
- [22] D. Arcos-Aviles, J. Pascual, L. Marroyo, P. Sanchis, and F. Guinjoan, “Fuzzy Logic-Based Energy Management System Design for Residential Grid-Connected Microgrids,” *IEEE Transactions on Smart Grid*, vol. 9, no. 2, pp. 530–543, 2018.
- [23] P. Sharma and A. Singh, “Era of deep neural networks: A review,” in *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 2017, pp. 1–5.
- [24] J. Moreno, M. Ortuzar, and J. Dixon, “Energy-management system for a hybrid electric vehicle, using ultracapacitors and neural networks,” *IEEE Transactions on Industrial Electronics*, vol. 53, no. 2, pp. 614–623, 2006.
- [25] L. Galván, J. M. Navarro, E. Galván, J. M. Carrasco, and A. Alcántara, “Optimal Scheduling of Energy Storage Using A New Priority-Based Smart Grid Control Method,” *Energies*, vol. 12, no. 4, 2019. [Online]. Available: <https://www.mdpi.com/1996-1073/12/4/579>
- [26] V. Thamminaidu and S. K V, “Model Predictive Control (MPC) of System Identified Continuous Stirred Tank Reactor (CSTR) with Constraints,” in *2021 IEEE 6th International Conference on Computing, Communication and Automation (ICCCA)*, 2021, pp. 195–201.

- [27] T. Erez, K. Lowrey, Y. Tassa, V. Kumar, S. Kolev, and E. Todorov, “An integrated system for real-time model predictive control of humanoid robots,” in *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2013, pp. 292–299.
- [28] P. Poramapojana and B. Chen, “Minimizing HEV fuel consumption using model predictive control,” in *Proceedings of 2012 IEEE/ASME 8th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications*, 2012, pp. 148–153.
- [29] A. Parisio, C. Wiezorek, T. Kyntäjä, J. Elo, K. Strunz, and K. H. Johansson, “Cooperative MPC-Based Energy Management for Networked Microgrids,” *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 3066–3074, 2017.
- [30] J. Zhang, I. Roumeliotis, and A. Zolotas, “Nonlinear Model Predictive Control-Based Optimal Energy Management for Hybrid Electric Aircraft Considering Aerodynamics-Propulsion Coupling Effects,” *IEEE Transactions on Transportation Electrification*, vol. 8, no. 2, pp. 2640–2653, 2022.
- [31] C. T. Aksland and A. G. Alleyne, “Hierarchical model-based predictive controller for a hybrid UAV powertrain,” *Control Engineering Practice*, vol. 115, p. 104883, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S096706612100160X>
- [32] M. Asghari, A. M. Fathollahi-Fard, S. M. J. Mirzapour Al-e hashem, and M. A. Dulebenets, “Transformation and linearization techniques in optimization: A state-of-the-art survey,” *Mathematics*, vol. 10, no. 2, 2022. [Online]. Available: <https://www.mdpi.com/2227-7390/10/2/283>
- [33] X. Wang, Y. Gao, J. Atkin, and S. Bozhko, “Neural network based weighting factor selection of mpc for optimal battery and load management in mea,” in *2020 23rd International Conference on Electrical Machines and Systems (ICEMS)*, 2020, pp. 1763–1768.