Jornadas de Automática

Red neuronal imitando un EMS basado en optimización: prueba de concepto en un PLC.

Borja Monsalvez Pozo, Francesc Xavier Blasco Ferragud[a,*], Alberto Pajares[a], Javier Sanchis[a]

[a] *Instituto de Automática e Informática Industrial, Universitat Politècnica de València*

To cite this article: Monsalvez Pozo, B., Blasco Ferragud, F.X., Pajares, A. 2026. Neural network imitating an optimisation-based EMS: proof of concept on a PLC. Jornadas de Automática, 47. https://doi.org/10.17979/ja-cea.2026.47.13768

Resumen

Este trabajo presenta una prueba de concepto implementada en un PLC de un sistema de gestión de energía (EMS) basado en inteligencia artificial para redes eléctricas inteligentes con almacenamiento en batería. Se entrena una red neuronal multicapa (MLP) para aproximar las decisiones de control óptimas de un EMS convencional basado en control predictivo con modelos (MPC), que resuelve un problema de programación lineal entera mixta (MILP). El modelo de IA, entrenado con 267.894 pares entrada-salida generados por el optimizador de referencia, replica la política de control óptima con un error cuadrático medio (RMSE) de 0,073 y un tiempo de inferencia unas 21 veces inferior al del optimizador. La red se implementa íntegramente en CODESYS sobre un PLC simulado, demostrando su operatividad en un controlador industrial estándar conforme a IEC 61131-3. Los resultados validan la viabilidad de sustituir un optimizador MILP por una red neuronal para la gestión energética en tiempo real con recursos computacionales reducidos.

Palabras clave: EMS, redes inteligentes, MPC, red neuronal, control predictivo, CODESYS, PLC, almacenamiento energético.

Neural network imitating an optimisation-based EMS: proof of concept on a PLC

Abstract

This paper presents a proof-of-concept of an artificial-intelligence-based energy management system (EMS) for smart grids with battery storage. A multilayer perceptron (MLP) is trained to approximate the optimal control decisions of a model predictive control (MPC) EMS that solves a mixed-integer linear programming (MILP) problem at each time step. The AI model, trained on 267,894 input-output pairs generated by the reference optimiser, replicates the optimal control policy with an RMSE of 0.073 and an inference time approximately 21 times lower than the optimiser. The network is fully implemented in CODESYS on a simulated PLC, demonstrating its operation on a standard IEC 61131-3 industrial controller. Results validate the feasibility of replacing a MILP optimiser with a neural network for real-time energy management under limited computational resources.

Keywords: EMS, smart grids, MPC, neural network, predictive control, CODESYS, PLC, energy storage.

1. Introducción

La transición energética ha impulsado el despliegue de fuentes renovables y sistemas de almacenamiento en redes eléctricas inteligentes (smart grids, Hossain et al., 2016). Los sistemas de gestión de energía (EMS) coordinan los flujos de generación, almacenamiento y consumo con el objetivo de minimizar el coste operativo y garantizar el equilibrio entre oferta y demanda (Bordons et al., 2020).

Actualmente, el enfoque más preciso es el control predictivo basado en modelos (MPC, Garcia-Torres & Bordons, 2015), que formula la gestión como un problema de programación lineal entera mixta (MILP) a resolver en cada instante de control. Aunque proporciona soluciones de alta

*Autor para correspondencia: xblasco@upv.es

calidad, su coste computacional crece de manera exponencial con el aumento de las variables, dificultando la implementación directa en controladores industriales de recursos limitados (Pajares et al., 2023; Vivas et al., 2025).

Las técnicas de *deep learning* ofrecen una vía alternativa: entrenar una red neuronal para imitar el comportamiento del optimizador, realizando la inferencia en tiempo real sin resolver el problema de optimización en cada ciclo (Liu et al., 2025; Matrone et al., 2024). Este paradigma, denominado *approximate* MPC, ha demostrado su eficacia en publicaciones recientes, aunque la validación en hardware industrial sigue siendo escasa.

Este trabajo constituye una prueba de concepto completa que cubre el ciclo íntegro: generación de datos con el optimizador MPC, diseño y entrenamiento de una MLP, comparación dinámica con el optimizador y validación en un PLC simulado mediante CODESYS. La contribución principal es demostrar que una red neuronal puede sustituir al EMS basado en MILP manteniendo un rendimiento satisfactorio, con una reducción del coste computacional de un orden de magnitud, y siendo directamente desplegable en plataformas industriales IEC 61131-3 (Werner, 2009).

2. Descripción del sistema

El sistema modelado corresponde a una microrred residencial que integra: conexión bidireccional a la red eléctrica (potencias de compra $Pgrid_c$ y venta $Pgrid_v$), potencias de generación fotovoltaica (Ppv), almacenamiento en batería (potencias de carga $Pbat_c$ y descarga $Pbat_d$) y demanda eléctrica (Pload). La Figura 1 muestra el esquema del sistema y la Tabla 1 los parámetros de diseño.

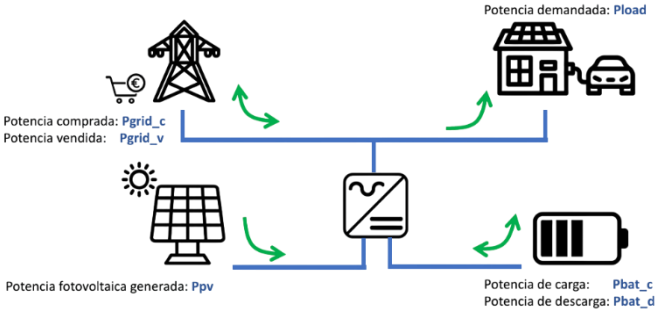

Figura 1: Esquema del sistema eléctrico usado.

Tabla 1: Parámetros de la instalación y del EMS.

Parámetro	Valor	Unidad.
Pot. máx inversor.	±3	kW
Pot. máx. carga/descarga bat.	3/3	kW
Pot. máx. compra/venta red	10/3	kW
SOC mínimo/máximo	10/90	%
Capacidad batería	7	kWh
Rendimiento carga/descarga	0,89/0,92	-
Horizonte de predicción $[Horizonte_{temporal}]$	24	h
Periodo de muestreo	60	min

El EMS de referencia resuelve un problema MILP en un horizonte de 24 horas (periodo de muestreo: 1 h), calculando en cada instante las potencias que minimizan el coste económico de operación (Pajares et al. (2023)):

$$min\ J$$

donde:

$$J = \sum_{k=1}^{PH} \sum_{i=MEF}^{|MEF|} C_i^{var}(k) \cdot P_i(k) \cdot T_s + C_i^{fix}(k) \cdot WT_i(k) \cdot T_s + C_i^{start}(k) \cdot Start(k) \quad (1)$$

sujeto a:

$$\underline{P_i} \cdot WT_i(k) \le P_i(k) \le \overline{P_i} \cdot WT_i(k) \quad \forall\, i \in MEF_i$$

$$WT_{ch_j}(k) + WT_{dis_j}(k) \le 1 \;\forall\, \mathrm{j} \in ESS_j$$

$$WT_{gridin}(k) + WT_{gridout}(k) \le 1$$

$$\underline{x_l} \le x_l(k) \le \overline{x_l} \;\forall\, x_l(k)$$

$$P_{nMEF+} - P_{nMEF-} - Loss(k) + P_{grid}(k) + \sum_{j=1}^{n} P_j(k) = 0$$

El problema de optimización mixta-entera lineal (MILP) formulado busca minimizar un índice de coste económico de operación (J) dentro de un horizonte de predicción (PH). La función objetivo se compone de tres términos principales escalados por el periodo de muestreo del sistema (T_s) expresado en horas. El primer término integra los costes variables (C_i^{var}) asociados al flujo de energía manipulable (MEF) i (en €/Wh) multiplicados por su respectiva potencia (P_i) en el instante de tiempo k. El segundo término añade el coste fijo de operación de dicho flujo manipulable (C_i^{fix} en €/h), condicionado por la variable binaria de funcionamiento (WT), la cual toma el valor 1 si el flujo está activo y 0 en caso contrario. El tercer término incorpora el coste económico de arranque (C_i^{start} en €), activado mediante la variable binaria de inicio ($Start(k)$), que toma el valor 1 exclusivamente en el instante en que el dispositivo o flujo manipulable se pone en marcha. Este sumatorio se extiende a lo largo de todo el horizonte de predicción para el conjunto completo de flujos de energía manipulables, donde el término |MEF| denota el cardinal o número total de dichos elementos controladores del balance de la microrred.

Por otra parte, la viabilidad operativa del sistema se rige por un conjunto de restricciones técnicas y balances de potencia. La ecuación de balance energético garantiza que en todo intervalo la suma de la generación local más los recursos externos cubran la demanda instantánea del sistema, asegurando que el balance neto se equilibre a cero en cada instante. En dicha expresión, P_{nMEF+} y P_{nMEF-} representan los flujos de energía no manipulables, donde el signo positivo define la potencia total inyectada —que en este sistema residencial equivale a la potencia fotovoltaica generada (P_{pv})— y el signo negativo define la potencia total demandada o consumida, representada por la carga de la instalación (P_{load}). A este balance se le restan las pérdidas totales del sistema ($Loss$), asociadas al rendimiento y disipación de los equipos, cuyas ecuaciones detalladas de cálculo pueden consultarse en Pajares et al. (2023). El balance se completa

con la interacción con la red eléctrica principal mediante la potencia de compra (P_{gridc}) y la potencia de venta (P_{gridv}), junto con las potencias procedentes del sistema de almacenamiento de energía (ESS), el cual se identifica genéricamente con el subíndice j para describir los flujos de potencia de carga (P_{batc}) y de descarga (P_{batd}) de la batería.

Finalizando el modelo, se imponen límites físicos y restricciones lógicas para garantizar la seguridad y coherencia comercial de la instalación. Los límites de potencia operativos de cada flujo manipulable se acotan en cada instante multiplicando sus capacidades límites físicas individuales ($\underline{P_i}$ y $\overline{P_i}$) por la variable binaria de estado de funcionamiento (WT_i). Para evitar la degradación del sistema y asegurar la coherencia física, las variables binarias de acción para la carga (WT_{ch_j}) y descarga (WT_{dis_j}) de los sistemas de almacenamiento están restringidas de manera que su suma sea menor o igual a 1, lo que impide matemáticamente que un dispositivo se cargue e inyecte energía en el mismo intervalo temporal. Una lógica análoga se aplica a la interacción con la red externa mediante las variables de compra (WT_{gridc}) y venta (WT_{gridv}) para evitar la simultaneidad de ambas operaciones comerciales. Por último, el comportamiento dinámico del sistema queda confinado por las variables de estado x_l, que representan niveles de almacenamiento físico y que en este trabajo se centran en el estado de carga de la batería (SOC), el cual evoluciona dinámicamente según los rendimientos de carga y descarga de la batería, debiendo mantenerse estrictamente entre sus límites operativos mínimos ($\underline{x_l}$) y máximos ($\overline{x_l}$) autorizados.

Para unificar la nomenclatura matemática, cabe aclarar que las variables genéricas de potencia manipulable P_i indexadas se corresponden de forma unívoca con las consignas físicas del balance del sistema, cumpliéndose que $P_i \in \{P_{gridc}, P_{gridv}, P_{batc}, P_{batd}\}$. En este trabajo se considera como sistema de almacenamiento de energía el SOC de la batería se modela mediante la siguiente ecuación:

$$SOC_{k+1} = SOC_k + \eta_c \cdot Pbat_c - \frac{1}{\eta_D} \cdot Pbat_d \qquad (2)$$

donde η_c = 0,89 y η_D = 0,92 representan los rendimientos de carga y descarga, respectivamente. Asimismo, el balance energético debe satisfacerse en cada intervalo mediante la siguiente restricción de igualdad:

$$P_{grid_{k+i}} + P_{bat_{k+i}} + Ppv_{k+i} = Pload_{k+i} \quad \forall i \in [0 - Horizonte_{temporal} - 1] \qquad (3)$$

Donde $P_{grid_{k+i}} = P_{gridc_{k+i}} - P_{gridv_{k+i}}$ y $P_{bat_{k+i}} = P_{batd_{k+i}} - P_{batc_{k+i}}$. Esta expresión garantiza que en todo instante la suma de la generación local más los recursos externos cubran la demanda instantánea del sistema.

3. Metodología

3.1. Generación del conjunto de datos

Los datos de entrenamiento se obtienen ejecutando el EMS-MILP sobre un amplio conjunto de escenarios generados variando de forma combinada cinco variables: SOC inicial, perfil de demanda ($Pload$), potencia fotovoltaica prevista (Ppv_{pred}), precio de compra y precio de venta de la energía.

El proceso iterativo de generación de datos sintéticos culminó en un tercer ciclo de discretización fina que produjo 267.894 pares entrada-salida. Para mapear el espacio de posibles estados de entrada, se variaron de forma anidada todas las variables mencionadas: el SOC se discretizó en intervalos de 2 unidades desde el 10% hasta el 90%; las cargas de la vivienda se ajustaron mediante un factor multiplicador entre 0,2 y 1,4 con un incremento de 0,2; y, finalmente, las variables de potencia predicha y precios de compra/venta se escalaron mediante un factor entre 0,5 y 3, variando en pasos de 0,25. En la figura 2 se puede apreciar el proceso iterativo previamente mencionado.

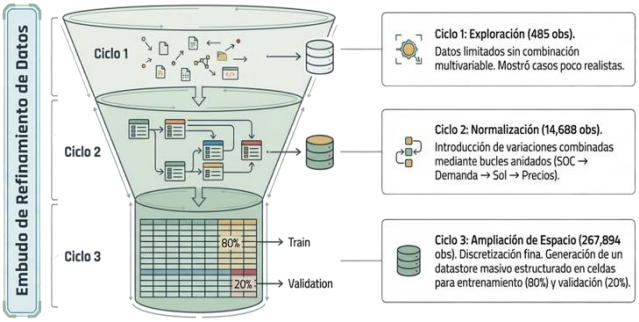

Figura 2: Preparación del *dataset*.

En cada escenario, el MPC calcula el vector de acciones para el horizonte completo de 24 horas, pero solo se registra la primera acción de cada variable de decisión, siguiendo la filosofía del horizonte deslizante.

Cada muestra de entrada tiene 97 características: 24 valores por cada variable temporal ($Pload$, Ppv_{pred}, $Precio_c$, $Precio_v$) y el valor actual del SOC. Todas las variables se normalizaron (media cero, desviación unitaria) para mejorar la estabilidad del entrenamiento.

3.2. Arquitectura de la red neuronal

Para la aproximación del EMS se ha diseñado una arquitectura de red neuronal Multi-Layer Perceptron (MLP) totalmente conectada, cuyo esquema estructural se detalla de forma gráfica en la Figura 3. Esta tipología ofrece un equilibrio idóneo entre precisión y eficiencia para tareas de regresión multivariable con datos tabulares, garantizando una inferencia determinista y un bajo coste computacional. Como se aprecia en la ilustración, el modelo procesa en paralelo los bloques temporales de predicción y el estado de la batería para generar de manera simultánea las consignas óptimas de operación, condiciones críticas para su posterior despliegue en un controlador lógico programable (PLC).

En este contexto se han descartado, por un lado, las redes recurrentes (RNN) al no existir dependencias temporales que el modelo deba extraer de forma interna, dado que los 97 valores de entrada son conocidos explícitamente en cada instante de cálculo. Por otro lado, se descartaron las redes

convolucionales (CNN) debido a la ausencia de dependencias espaciales o correlaciones locales en el conjunto de datos.

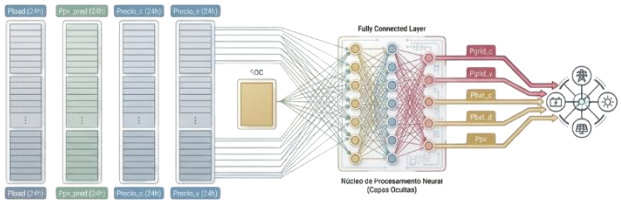

Figura 3: Arquitectura de la red neuronal.

El uso de estas tipologías añadiría una complejidad estructural injustificada. Asimismo, su implementación resultaría significativamente más compleja al traducirlas al estándar de programación Texto Estructurado (ST, IEC 61131-3), lo cual complicaría tanto el mantenimiento del código como la eficiencia de ejecución en el hardware industrial objetivo.

El entrenamiento se realizó con Deep Network Designer de MATLAB, usando el optimizador Adam (α = 0,001), mini-batch de 64 muestras, 30 épocas máximas y división de datos 80%/20% entrenamiento/validación. Las capas ocultas usan activación ReLU; la capa de salida es lineal con pérdida MSE.

Se evaluaron 9 arquitecturas con distintas profundidades y anchos. La Tabla 2 recoge los resultados de la media del RMSE de 50 escenarios diferentes para validar la calidad de la representación.

Tabla 2: Principales arquitecturas evaluadas (E = entrada, S = salida).

Red	Arquitectura	Nº obs	RMSE
R4	E-128-64-32-16-8-S	14.688	0,1915
R5	E-256-128-64-32-16-8-S	14.688	0,1619
R6	E-64-32-16-8-S	267.894	0,0725
R8	E-128-64-32-16-S	267.894	0,1352
R9	E-10-64-32-16-8 16-32-64-10-64 32-16-8-S	267.894	0,1804

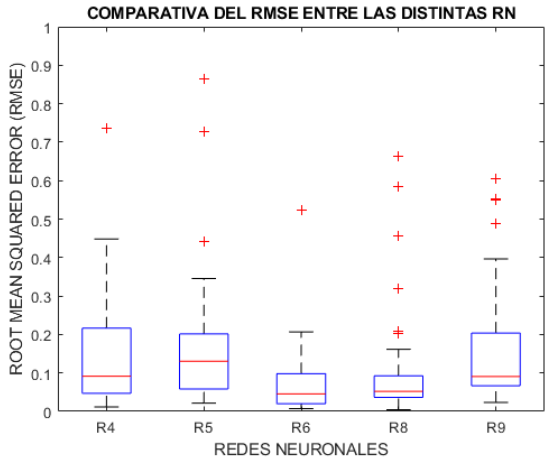

Figura 4: Gráfica comparativa del RMSE de las distintas RN.

Se establecieron dos etapas de entrenamiento diferenciadas por el volumen de datos: una fase inicial con 14.688 observaciones, donde se exploraron diversas tipologías de 'embudo' (reducción progresiva de neuronas, ej. 64-32-16-8), y una fase posterior con 267.894 observaciones. En esta segunda etapa, se experimentó tanto configuraciones de embudo simple como estructuras más complejas de tipo decreciente-creciente-decreciente. Estas tipologías fueron seleccionadas por ser las arquitecturas estándar ampliamente utilizadas.

La arquitectura R6 (E-64-32-16-8-S), entrenada con el *dataset* ampliado, obtiene el menor RMSE (0,073) a pesar de ser menos profunda que R8 y R9. Esto evidencia que para el presente trabajo la cantidad y diversidad de los datos es más determinante que la complejidad arquitectónica.

A través de los diagramas de caja de la Figura 4 se puede realizar un análisis más detallado de la dispersión del error en los 50 escenarios de test. Al comparar las dos arquitecturas con mejor rendimiento sobre el dataset ampliado, R6 y R8, se observa que ambas son similares en media. Sin embargo, aunque R6 presenta una variabilidad sutilmente mayor en su rango intercuartílico, la red R8 acumula una cantidad notablemente superior de valores atípicos (outliers) con errores elevados. Por tanto, de cara a asegurar la calidad y estabilidad de la aproximación ante escenarios imprevistos, R6 se postula como la opción preferible. De este modo, la arquitectura R6 (E-64-32-16-8-S) obtiene el menor RMSE (0,073) a pesar de ser menos profunda que R8 y R9. Esto evidencia que para el presente trabajo la cantidad y diversidad de los datos es más determinante que la complejidad arquitectónica.

3.3. Implementación en CODESYS

Para la fase de implementación, se seleccionó la arquitectura R9. La elección de este modelo, a pesar de presentar un error de validación ligeramente superior al de R6, responde al objetivo de analizar la viabilidad y los límites de computación de redes neuronales de mayor complejidad en entornos industriales. Metodológicamente, validar la propuesta bajo la configuración más pesada y con mayor número de parámetros (R9) permite establecer una cota superior del coste computacional, garantizando que, si R9 es viable en el hardware objetivo, cualquier arquitectura optimizada de menor tamaño (como R6) lo será con holgura.

La red se implementó en un PLC simulado (CODESYS, IEC 61131-3, Werner, 2009) siguiendo el esquema de bloques funcionales e intercambio de variables que se ilustra en la Figura 5. El proceso se realizó siguiendo tres pasos: (1) exportación automática de pesos y sesgos desde MATLAB mediante un script capaz de traducir código Matlab a ST; (2) traducción al lenguaje Texto Estructurado de las operaciones capa a capa, incluyendo la función de activación ReLU (mediante el mismo script previamente mencionado); y (3) encapsulado como bloque funcional con 97 entradas y 5 salidas.

Como se puede apreciar en el flujo de la Figura 5, el script desarrollado para hacer esta tarea permite traducir arquitecturas MLP desde MATLAB a código ST de manera automatizada. Permitiendo hacer una integración prácticamente automática una vez se ha desarrollado el modelo en MATLAB.

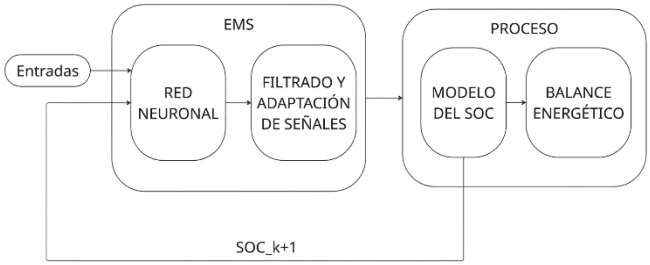

Figura 5: Esquema de la implementación en CODESYS.

Junto a la programación de la red se implementaron tres bloques adicionales. Dentro del bloque de simulación del EMS, se programó un filtrado y adaptación de señales que se encarga de forzar a 0 los valores residuales del orden de 10^{-6} y asegurar que no se compre y venda en el mismo instante y que tampoco se cargue y descargue la batería a la misma vez.

Para las pruebas experimentales se ha desarrollado en el propio PLC virtual un modelo dinámico que simula el proceso. Dentro de la simulación, se calculaba la dinámica del sistema (SOC siguiente) validando el límite físico del sistema, y por otro lado el bloque del balance energético para obtener la potencia neta que se compra o se vende a la red.

4. Resultados y discusión

4.1. RN vs. MPC: validación de la aproximación

La Figura 6 y la Figura 7 se muestra un ejemplo particular para comparar la evolución del SOC y la potencia de carga de batería respectivamente durante un ciclo de 24 horas para el MPC de referencia (modo dinámico) y la red neuronal R9, ambos ejecutados en MATLAB sobre un escenario de validación con condiciones distintas a las del entrenamiento.

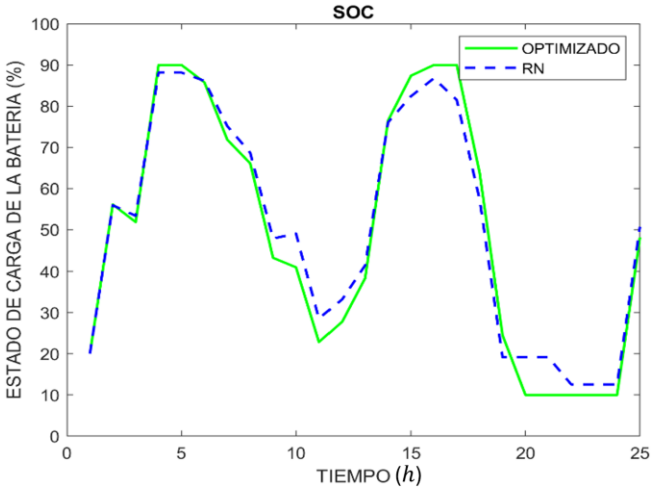

Figura 6: Comparativa del SOC entre la RN y optimizado.

Ambas curvas presentan una tendencia prácticamente idéntica en todas las variables de decisión (SOC, P_{batc}, P_{batd}, Ppv, P_{gridc}, P_{gridv}). Las pequeñas desviaciones puntuales observadas son consecuencia del error de aproximación de la red y no se acumulan significativamente a lo largo del horizonte, lo que confirma la estabilidad dinámica del sistema controlado por la MLP. Esta ausencia de deriva o error acumulativo en el SOC a largo plazo demuestra la robustez del sistema en bucle cerrado, manteniendo la estabilidad operativa durante todo el ciclo de funcionamiento. Asimismo, cabe destacar que la pérdida intrínseca de garantías analíticas sobre el cumplimiento de restricciones físicas (característica propia al sustituir un optimizador MPC por una aproximación estadística) queda mitigada con éxito gracias al bloque de filtrado de seguridad programado en el PLC. Este enfoque híbrido absorbe las desviaciones de la red y asegura la viabilidad operativa estricta frente a restricciones críticas.

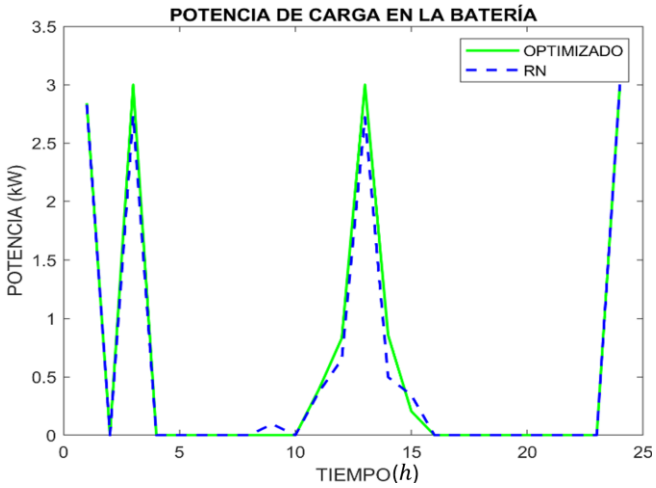

Figura 7: Comparativa de la potencia de carga entre la RN y optimizado.

4.2. CODESYS vs MATLAB: validación de la implementación

La Figura 8 y la Figura 9 se muestra un ejemplo particular para comparar los resultados de la red neuronal ejecutada en CODESYS con los obtenidos en MATLAB para la evolución del SOC y la potencia de compra a la red respectivamente con las mismas entradas. Las curvas son completamente idénticas, lo que confirma que: (a) la exportación de pesos y sesgos se realizó sin pérdida de precisión numérica; (b) la traducción a ST es matemáticamente fiel; y (c) no se introducen errores adicionales en el entorno del PLC.

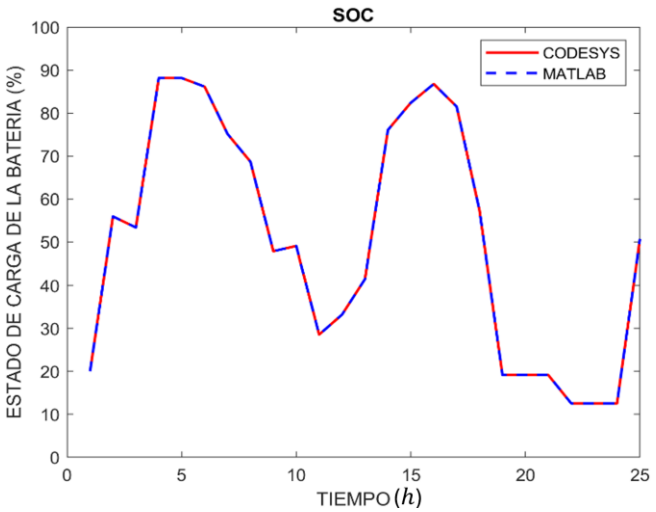

Figura 8: Comparativa del SOC entre la CODESYS y MATLAB.

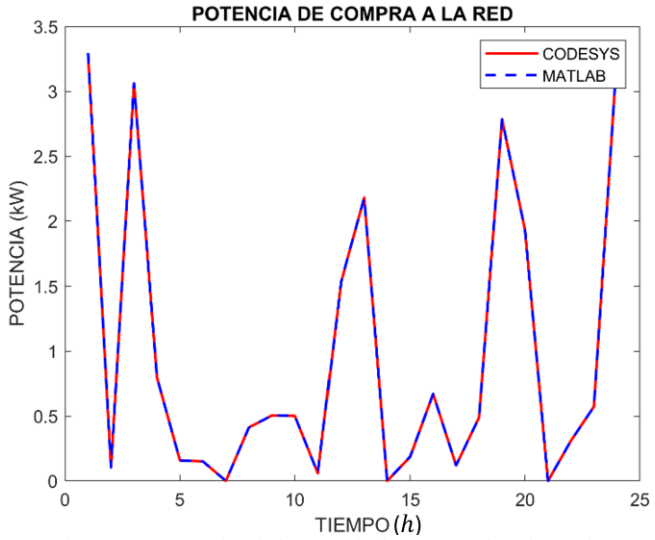

Figura 9: Comparativa de la potencia de compra a la red entre la CODESYS y MATLAB.

4.3. Coste computacional

La Tabla 3 compara los tiempos medios de ejecución por instante de control. La red neuronal presenta tiempos de inferencia de ~3 ms frente a los ~80 ms del optimizador MILP, lo que representa una reducción media de un factor 21, con valores superiores a 30 en algunos instantes.

Tabla 3: Tiempos de ejecución medios.

Método	T. medio (ms)	Factor.
MPC-MILP (MATLAB)	80,7	1x
Red neuronal R9 (MATLAB)	3,0	~21x

Más allá del tiempo de ejecución, donde el optimizador MILP requiere 80,7 ms frente a los 3ms de la red neuronal, la principal diferencia radica en la complejidad de implementación.

Mientras que la red neuronal se reduce a operaciones matriciales sencillas y robustas en el estándar IEC 61131-3, el EMS-MILP exige integrar librerías de optimización externas, lo que incrementa significativamente la carga de desarrollo. Además, ante escenarios de mayor complejidad o mayores horizontes de predicción, el crecimiento exponencial en el tiempo de computación del MILP compromete la viabilidad del control en tiempo real, validando la red neuronal como una solución más escalable y eficiente para entornos industriales.

5. Conclusiones

Este trabajo demuestra la viabilidad técnica de sustituir un EMS basado en optimización MILP por una red neuronal MLP para la gestión energética en tiempo real en redes inteligentes con almacenamiento. Las principales conclusiones son:

- La red neuronal R6 (E-64-32-16-8-S), entrenada con 267.894 observaciones, aproxima fielmente las decisiones del MPC con RMSE = 0,073, sin degradación observable del comportamiento dinámico.
- La cantidad y representatividad del conjunto de datos para este proyecto tiene mayor impacto en el rendimiento que la profundidad de la arquitectura.
- La implementación en CODESYS, mediante un script de exportación automática a Texto Estructurado, produce resultados idénticos a MATLAB, confirmando la correcta translación del modelo al PLC.
- El tiempo de inferencia de la red neuronal es ~21 veces inferior al del optimizador, abriendo la puerta al despliegue en controladores industriales de recursos limitados.

Como trabajos futuros se plantea: explorar arquitecturas recurrentes (LSTM, GRU) para capturar dependencias temporales; implementar modelos híbridos MPC/RN; integrar modelos de predicción de demanda y generación; validar el sistema en un PLC físico conectado a una planta real; y evaluar el impacto en esta metodología cuando se escala a instalaciones energéticas más complejas.

Agradecimientos

Este trabajo ha sido financiado parcialmente por el proyecto PID2024-158941OB-I00 financiado por MICIU/AEI/10.13039/501100011033 y por FEDER, UE. Se ha realizado en el marco del Trabajo Fin de Máster del Máster Universitario en Ingeniería Industrial de la Universitat Politècnica de València, con el apoyo del Grupo de Control Predictivo y Optimización Heurística (CPOH-UPV).

Referencias

Bordons, C., Garcia-Torres, F., & Ridao, M. A. (2020). *Model Predictive Control of Microgrids*. https://doi.org/10.1007/978-3-030-24570-2

Garcia-Torres, F., & Bordons, C. (2015). Optimal Economical Schedule of Hydrogen-Based Microgrids With Hybrid Storage Using Model Predictive Control. *IEEE Transactions on Industrial Electronics*, *62*(8), 5195–5207. https://doi.org/10.1109/TIE.2015.2412524

Hossain, M. S., Madlool, N. A., Rahim, N. A., Selvaraj, J., Pandey, A. K., & Khan, A. F. (2016). Role of smart grid in renewable energy: An overview. In *Renewable and Sustainable Energy Reviews* (Vol. 60, pp. 1168–1184). Elsevier Ltd. https://doi.org/10.1016/j.rser.2015.09.098

Liu, C., Shi, S., Alan, A., Venayagamoorthy, G. K., & De Schutter, B. (2025). *Approximate Model Predictive Control for Microgrid Energy Management via Imitation Learning*. https://doi.org/10.48550/arXiv.2510.20040

Matrone, S., Pozzi, A., Ogliari, E., & Leva, S. (2024). *Deep Learning-Based Predictive Control for Optimal Battery Management in Microgrids*. https://doi.org/10.1109/ACCESS.2024.3458435

Pajares, A., Vivas, F. J., Blasco, X., Herrero, J. M., Segura, F., & Andújar, J. M. (2023). Methodology for energy management strategies design based on predictive control techniques for smart grids. *Applied Energy*, *351*. https://doi.org/10.1016/j.apenergy.2023.121809

Vivas, F. J., Pajares, A., Blasco, X., Herrero, J. M., Segura, F., & Andújar, J. M. (2025). A novel energy management system based on two-level hierarchical economic model predictive control for use in microgrid control. *Energy Conversion and Management: X*, *26*. https://doi.org/10.1016/j.ecmx.2025.101027

Werner, B. (2009). Object-oriented extensions for IEC 61131-3. *IEEE Industrial Electronics Magazine*, *3*(4), 36–39. https://doi.org/10.1109/MIE.2009.934795