



Universidad
LOYOLA

UNIVERSIDAD LOYOLA ANDALUCÍA

Máster en Ingeniería Industrial

TRABAJO FIN DE MÁSTER

**Gemelo digital basado en redes neuronales y elementos finitos para
la optimización de sistemas almacenamiento de energía térmica
basados en cemento con aditivos**

Miguel de los Santos González
Tutor: Francisco de Paula Montero Chacón

10 de julio de 2026

RESUMEN

El objetivo del siguiente documento es el estudio de la viabilidad de gemelos digitales para sistemas de almacenamiento térmico basados en cemento. Para mejorar estos sistemas, han de utilizarse aditivos que puedan mejorar las propiedades térmicas y físicas del cemento. Adicionalmente, se pretende conocer la respuesta termomecánica del sistema en diversas condiciones para poder replicar y predecir el comportamiento con un modelo subrogado de bajo coste computacional.

La metodología comienza con una revisión del estado del arte de las tecnologías de almacenamiento existentes y utilizarla para comprender el sistema que se desea implantar. Una vez conocido, se desarrolla un modelo de elementos finitos (FEM) que se ajuste al comportamiento real del equipo y evalúa cuál es el efecto de los aditivos en el sistema final. Ese modelado será el que se utilice para entrenar una red neuronal que pueda predecir el comportamiento.

A continuación, se especifican los requisitos para la implantación del sistema completo. Se consideran las posibles implantaciones y los componentes auxiliares necesarios para su adecuado funcionamiento.

Por último, se realiza una evaluación de la viabilidad económica del sistema. La evaluación cuantifica los costes de capital y operación (CAPEX y OPEX) para posteriormente obtener el coste nivelado de la energía almacenada (LCOS). Comparar ese LCOS con el coste de la energía (LCOE) en el mercado actual y con el ahorro en huella de carbono serán los principales indicadores de la viabilidad, además de los habituales como son VAN, TIR o *Payback*

ABSTRACT

The objective of this document is to evaluate the feasibility of digital twins applied to cement-based thermal energy storage (TES) systems. To optimize these systems, additives must be incorporated to enhance the thermophysical properties of the cement matrix. Additionally, this study aims to characterize the thermomechanical response of the system under variable operating conditions in order to replicate and predict its behavior using a low-cost computational surrogate model.

The methodology begins with a comprehensive review of the state-of-the-art in existing storage technologies to ground the proposed design. Subsequently, a Finite Element Method (FEM) model calibrated to the actual behavior of the equipment is developed to evaluate the impact of the additives on the global system. This numerical model serves as the foundation for training an artificial neural network dedicated to predicting thermal behavior.

Next, the engineering requirements for the full-scale system deployment are specified. This includes assessing optimal plant co-location scenarios and dimensioning the auxiliary components necessary to ensure proper operation.

Finally, a techno-economic feasibility study of the system is conducted. The evaluation quantifies both capital and operational expenditures (CAPEX and OPEX) to subsequently determine the Levelized Cost of Storage (LCOS). The project's profitability and viability are dictated by benchmarking the LCOS against the Levelized Cost of Energy (LCOE) of current market alternatives and the associated carbon footprint reduction, alongside standard financial metrics such as NPV, IRR, and Payback period.

Índice

1. Introducción	12
1.1. Motivación	12
1.2. Objetivo	12
2. Estado del arte	14
2.1. Almacenamiento térmico	14
2.2. Colectores Cilindro Parabólicos	16
2.3. Dopado del cemento	17
2.3.1. Cemento	17
2.3.2. Fibra de basalto	17
2.3.3. Ilmenita	17
2.4. Elementos finitos para almacenamiento de energía	20
2.5. Modelado mediante redes neuronales	20
3. Tecnología de almacenamiento	21
3.1. Datos disponibles del HTF	21
3.1.1. Cálculo de Coeficiente de película	22
3.2. Almacenamiento de energía térmica	24
3.2.1. Propiedades combinación hormigón e ilmenita	24
3.2.2. Propiedades combinación hormigón y fibra	25
3.3. Método elementos finitos	33
3.3.1. Formulación Térmica (Estado Estacionario)	33

3.3.2. Formulación Elástica (Mecánica)	33
3.3.3. Acoplamiento Termo-Mecánico	34
3.4. Modelo empleado	35
3.4.1. Condiciones de contorno	35
3.4.2. Cargas aplicadas	37
3.4.3. Configuración de cálculo	37
3.4.4. Simulaciones realizadas	39
4. Modelo subrogado	50
4.1. Redes Neuronales	52
4.1.1. Redes convolucionales	52
4.1.2. Redes recurrentes	55
4.1.3. Aplicación a problemas de elementos finitos	55
4.2. Preparación de datos	56
4.3. Red neuronal utilizada	57
4.3.1. Modelo completo frente a modelo específico	57
4.3.2. Modelo convolucional frente a modelo convolucional y recurrente	59
4.4. Resultados	60
4.4.1. Predicción autorregresión	60
4.4.2. Predicciones intervalos	72
5. Instalación industrial de sistema con almacenamiento energético	85
5.1. Esquemas CCP	86
5.1.1. Descripción General del Proceso	86
5.1.2. Designación de Equipos e Instrumentación (<i>Tags</i> ISA-5.1)	86
5.1.3. Lazos de control	88
5.1.4. Lógica de Interbloqueos (<i>Interlocks</i>)	88
5.2. Cálculo potencia CCP	90

5.3. Instalación residencial	93
5.3.1. Número de viviendas	93
5.3.2. Energía en CTES	94
6. Estudio de viabilidad económica	96
6.1. Estimación de costes	96
6.1.1. CAPEX	96
6.1.2. OPEX	98
6.2. Costes nivelados	99
6.2.1. Costo Nivelado de Energía (LCOE)	99
6.2.2. Costo Nivelado de Almacenamiento (LCOS)	99
6.3. Efecto medioambiental	101
6.3.1. Huella de carbono	101
6.4. Indicadores de rentabilidad	103
6.4.1. VAN	103
6.4.2. TIR	103
6.4.3. Payback Simple	103
7. Conclusiones	104
7.1. Tecnología	105
7.2. Instalación	106
7.3. Presupuesto	107
7.4. Consideraciones finales	108
8. Anexos	109
8.1. Código utilizado	109
8.1.1. Cálculo de propiedades	109
8.1.2. Cálculo automático en Abaqus	118
8.1.3. Scripts de la red neuronal	126

8.1.4. Resultados modelos	143
8.2. Planos	147
8.3. Presupuestos	148

Índice de figuras

3.1.1 Ecuaciones obtenidas tras aproximar con polinomios cada una de las variables de interés del HTF	23
3.2.1 Efecto en las características del material al añadir fibra	26
3.2.2 Efecto en las características del material al añadir ilmenita	27
3.2.3 Efecto en las características del material al añadir ilmenita y fibra en la concentración máxima teórica	29
3.2.4 Comportamiento durante 10 horas con HTF a 400°C y 2 m/s del CTES solo con cemento	30
3.2.5 Comportamiento durante 10 horas con HTF a 400°C y 2 m/s del CTES con el máximo teórico de aditivos	31
3.4.1 Sección que se revoluciona para obtener la figura 3D	35
3.4.2 Condiciones de contorno en tubería	36
3.4.3 Condiciones de contacto entre tubería y CTES	38
3.4.4 Gradiente de temperatura para la simulación inicial	39
3.4.5 Gradiente de temperatura para la simulación aislada	40
3.4.6 Gradiente de temperatura al inicio para la simulación a 400°C, velocidad HTF 2 m/s y diferencia de temperatura entre entrada y salida de 5°C . . .	41
3.4.7 Gradiente de temperatura al inicio para la simulación a 400°C, velocidad HTF 2 m/s y diferencia de temperatura entre entrada y salida de 5°C . . .	41
3.4.8 Temperatura de nodos en la simulación para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s	42
3.4.9 Temperatura de nodos en la simulación para 400°C de temperatura de HTF, diferencia de temperatura de 5°C entre entrada y salida y velocidad de HTF 2 m/s	43

3.4.1	Comportamiento de la tensión máxima en la simulación de cemento para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s	44
3.4.1	Comportamiento de la tensión máxima en la simulación de cemento para 400°C de temperatura de HTF, diferencia de temperatura de 5°C entre entrada y salida y velocidad de HTF 2 m/s	45
3.4.1	Comportamiento de la tensión mínima en la simulación de cemento para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s	45
3.4.1	Comportamiento de la tensión mínima en la simulación de cemento para 400°C de temperatura de HTF, diferencia de temperatura de 5°C entre entrada y salida y velocidad de HTF 2 m/s	46
3.4.1	Campo de temperatura final en la simulación de cemento para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s	47
3.4.1	Campo de temperatura final en la simulación de cemento con fibras para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s	47
3.4.1	Campo de temperatura final en la simulación de cemento con ilmenita para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s	48
3.4.1	Campo de temperatura final en la simulación de cemento con fibras e ilmenita para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s	48
3.4.1	Temperatura de nodos en la simulación con aditivos para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s	49
3.4.1	Temperatura de nodos en la simulación con aditivos para 400°C de temperatura de HTF, diferencia de temperatura de 5°C entre entrada y salida y velocidad de HTF 2 m/s	49
4.0.1	Alzado y perfil de la posición de los nodos utilizados para el estudio	51
4.1.1	Visualización de una <i>shallow neural network</i> , extraído de [10]	53
4.1.2	Visualización de una <i>convolutional neural network</i> haciendo un barrido en unos datos 2D con un filtro 3x3, extraído de [10]	54
4.3.1	Simulación hasta el paso 300 con redes recurrentes	58
4.3.2	Simulación únicamente de temperatura en mismas condiciones hasta el paso 300 con redes recurrentes	58

4.4.1 Comparación de temperatura calculada con RNN en los nodos centrales del CTES	60
4.4.2 Comparación de tensión máxima calculada con CNN en los nodos centrales del CTES	61
4.4.3 Comparación de tensión máxima calculada con RNN en los nodos centrales del CTES	62
4.4.4 Comparación de tensión mínima calculada con CNN en los nodos centrales del CTES	63
4.4.5 Comparación de tensión mínima calculada con RNN en los nodos centrales del CTES	64
4.4.6 Errores relativos de temperatura calculada con RNN en los nodos centrales del CTES	65
4.4.7 Errores relativos de tensión máxima calculada con CNN en los nodos centrales del CTES	66
4.4.8 Errores relativos de tensión máxima calculada con RNN en los nodos centrales del CTES	67
4.4.9 Errores relativos de tensión mínima calculada con CNN en los nodos centrales del CTES	68
4.4.10 Errores relativos de tensión mínima calculada con RNN en los nodos centrales del CTES	69
4.4.11 Comparación de temperatura calculada con CNN en los nodos centrales del CTES	72
4.4.12 Comparación de temperatura calculada con RNN en los nodos centrales del CTES	73
4.4.13 Comparación de tensión máxima calculada con CNN en los nodos centrales del CTES	74
4.4.14 Comparación de tensión máxima calculada con RNN en los nodos centrales del CTES	75
4.4.15 Comparación de tensión mínima calculada con CNN en los nodos centrales del CTES	76
4.4.16 Comparación de tensión mínima calculada con RNN en los nodos centrales del CTES	77
4.4.17 Errores relativos de temperatura calculada con CNN en los nodos centrales del CTES	78
4.4.18 Errores relativos de temperatura calculada con RNN en los nodos centrales del CTES	79

4.4.1 Errores relativos de tensión máxima calculada con CNN en los nodos centrales del CTES	80
4.4.2 Errores relativos de tensión máxima calculada con RNN en los nodos centrales del CTES	81
4.4.2 Errores relativos de tensión mínima calculada con CNN en los nodos centrales del CTES	82
4.4.2 Errores relativos de tensión mínima calculada con RNN en los nodos centrales del CTES	83
5.0.1 Muestra de posible instalación en azotea de edificio	85
5.1.1 Esquema de bloques funcional del diagrama de tuberías primario orientado a la generación de potencia.	89
5.3.1 Número de viviendas por tipo de edificio	93
8.2.1 Diagrama de elementos de sistema CCP con CTES	147
8.3.1 Plan de inversiones con amortizaciones y cuentas contables para un colector	148
8.3.2 Plan de inversiones con amortizaciones y cuentas contables para un lazo .	149

Índice de tablas

2.1.1Comparativa técnica de métodos de almacenamiento térmico.	15
2.3.1Propiedades mecánicas de los materiales empleados en el estudio	19
2.3.2Propiedades mecánicas de los materiales empleados en el estudio	19
3.1.1Propiedades termofísicas del HTF Dowtherm A en función de la tempe- ratura [5].	22
3.1.2Número de Reynolds (Re) en función de la velocidad y la temperatura.	22
3.1.3Número de Prandtl (Pr) calculado para cada rango de temperatura.	22
3.1.4Número de Nusselt (Nu) en función de la velocidad y la temperatura.	22
3.1.5Coeficientes de película (h) [W/m^2K] en función de la velocidad y la tem- peratura.	23
3.2.1Propiedades de cemento con aditivos	26
4.3.1Resultados de modelos	59
5.1.1Nomenclatura y especificación de componentes según norma ISA-5.1.	87
5.2.1Valores habituales para los cálculos	91
5.3.1Valores habituales para el diseño de lazos	93
6.4.1Flujos de caja iniciales	103

Capítulo 1

Introducción

1.1. MOTIVACIÓN

La mejora de eficiencias y el abaratamiento de costes de equipos termosolares permitiría la implementación de sistemas de almacenamiento de energía térmica. Hasta ahora se ha tratado con sistemas que almacenan agua caliente sanitaria, sin embargo esto tiene claras limitaciones de compacidad por no poder superar los 100°C.

El cemento es el material más utilizado en construcción y su uso está estandarizado y extendido, en su estado habitual presenta un comportamiento frágil frente a cambios bruscos de temperatura. Existen estudios sobre las características que podría obtener al doparse con otros materiales, permitiendo así resistir los ciclos de temperatura y ser una alternativa asequible para el almacenamiento de energía en forma de calor.

1.2. OBJETIVO

Con este proyecto se pretende estudiar un sistema auxiliar para almacenar energía térmica basado en el hormigón. Este sistema deberá ser capaz de obtener energía de otros sistemas para su posterior extracción ya sea para agua caliente sanitaria o para mantener un sistema de vapor.

El sistema está siendo investigado y se ha probado en elementos finitos, el siguiente punto es desarrollar un modelo que permita conocer el estado actual con fiabilidad y sin necesitar un gran proceso de cálculo. Para ello se estudiará el modelado mediante redes neuronales convolucionales adaptadas a un problema físico.

Posteriormente, se plantearán los recursos necesarios para conocer las necesidades que tendrá el sistema para su implantación en cualquier tipo de instalación. Se

tendrá que plantear cómo sería la sustitución de los sistemas habituales por esta propuesta. Además se podrán hacer variaciones de la composición del elemento almacenador según los requisitos de la instalación.

Por último, conocidas las posibilidades, se hará un estudio de viabilidad técnico-económica.

Capítulo 2

Estado del arte

La búsqueda de sistemas de almacenamiento o gestión de la energía es uno de los mayores retos de la actualidad, especialmente en países como España donde se ha invertido en energías renovables cuya producción no es constante. Es por ello que el almacenamiento de energía térmica (TES) es una importante línea de investigación y sus avances aportan a la descarbonización y mejora de la energías solar o para la recuperación en ciclos térmicos.

En este estudio, se planteará conocer el estado de la técnica a tres niveles: macroscópico, microscópico y de *software*. En primer lugar, se exploran las tecnologías actuales de almacenamiento térmico, evaluando sus rangos de operación y eficiencia. En segundo lugar, se analiza el uso de matrices cementicias como medio de almacenamiento sólido, haciendo especial hincapié en las modificaciones microestructurales y aditivos utilizados para mejorar su conductividad y capacidad calorífica. Finalmente, se examinan los fundamentos y metodologías de diseño de los gemelos digitales (*Digital Twins*), herramientas clave para la simulación en tiempo real y el análisis transitorio de estos activos industriales.

2.1. ALMACENAMIENTO TÉRMICO

Los diversos métodos actuales de almacenamiento de energía térmica.

Almacenamiento de Calor Sensible (SHS)

Es la tecnología de mayor madurez comercial. Se basa en la variación de la temperatura de un medio líquido o sólido. La cantidad de energía almacenada (Q) se rige por la ecuación:

$$Q = m \cdot c_p \cdot \Delta T \quad (2.1.1)$$

Donde m es la masa, c_p el calor específico y ΔT el gradiente de temperatura. El agua sigue siendo el medio predominante para aplicaciones de baja temperatura ($< 100^\circ\text{C}$),

mientras que las sales fundidas (mezclas binarias de nitratos) dominan las plantas de concentración solar (CSP) con rangos operativos de hasta 550°C [2].

Almacenamiento de Calor Latente (LHS)

Utiliza Materiales de Cambio de Fase (*Phase Change Materials*, PCM). El almacenamiento ocurre durante la transición de fase (generalmente sólido-líquido) a una temperatura casi constante. Esto presenta una considerable mejora en la densidad energética y control de la temperatura ya que durante el cambio de fase se mantiene la temperatura constante. Sin embargo, los PCMs orgánicos (parafinas) e inorgánicos (sales hidratadas) presentan una baja conductividad térmica. Actualmente, investigaciones como las de Xu et al. [1] se centran en la microencapsulación y el uso de matrices metálicas para mejorar la transferencia de calor.

Almacenamiento Termoquímico (TCS)

Se basa en reacciones químicas reversibles o procesos de sorción. Representa la frontera tecnológica (TRL 4-5) debido a su capacidad de almacenar energía de forma indefinida sin pérdidas térmicas significativas, lo que lo hace ideal para el almacenamiento estacional. A pesar de su densidad energética superior (hasta 10 veces mayor que el sensible), la complejidad del diseño de los reactores limita su implementación comercial en 2026, ver en F1000 Research [3].

Tabla 2.1.1: Comparativa técnica de métodos de almacenamiento térmico.

Tecnología	Densidad (MJ/m³)	Eficiencia (%)	Periodo	Madurez
Sensible (Agua/Sales)	50 – 150	50 – 90	Horas/Días	Comercial
Latente (PCM)	200 – 500	75 – 90	Días/Semanas	Demostración
Termoquímico (TCS)	500 – 1500	75 – 100	Estacional	Investigación

El estado del arte actual apunta hacia la hibridación. Los sistemas que combinan calor sensible en materiales granulares con PCMs integrados permiten optimizar la termoclina en tanques de almacenamiento. Asimismo, la digitalización de la gestión térmica mediante algoritmos de control predictivo es el estándar actual para maximizar el LCOS (*Levelized Cost of Storage*) en entornos industriales, AT Tafere [3].

En este documento se estudia principalmente la optimización y digitalización del calor sensible con perspectivas de utilizar PCM en investigaciones futuras.

2.2. COLECTORES CILINDRO PARABÓLICOS

El sistema de almacenamiento de energía térmica que se diseña sirve de sistema de apoyo a un tipo de generación de energía ya existente como es el sistema de colectores cilindro parabólicos. El procedimiento de generación de energía eléctrica, resumido, consiste en disponer concentradores de foco lineal con seguimiento en un eje y forma parabólica, de forma que toda la radiación incidente se concentra en el foco de la parábola donde se coloca un tubo por el que circula un fluido caloportante (*Heat Transfer Fluid*) habitualmente algún tipo de aceite. Posteriormente este aceite pasa por unos intercambiadores de calor para obtener vapor que se utilice en un bloque de potencia como es una turbina Rankine.

Estos sistemas pueden disponer de sistemas de almacenamiento térmico para producir energía incluso en momentos en los que no reciba radiación directa, pero los sistemas actuales son mediante tecnología de sales fundidas la cual requiere mucho espacio y equipos complejos. La investigación actual se centra en obtener sistemas de almacenamiento térmico más sencillos y baratos que puedan ser de apoyo en bloques de generación más pequeños y no necesariamente industriales, Barrera-Díaz et al. [4].

El funcionamiento y optimización de la obtención de la radiación no será objeto de estudio, directamente se asumirá la captación de la radiación. Lo que sí es importante comprender para las simulaciones es el comportamiento del HTE.

2.3. DOPADO DEL CEMENTO

El dopado del cemento con fibras y polvos representa una innovación clave en la ingeniería de materiales para la construcción, permitiendo mejorar las propiedades mecánicas y térmicas como ocupa a este estudio. Este enfoque, inspirado en técnicas de modificación nanométrica o adición de fases secundarias, busca optimizar la matriz cementicia mediante la incorporación controlada de refuerzos fibrosos o polvos minerales, reduciendo microfisuras y potenciando la resistencia a esfuerzos.

2.3.1. Cemento

El cemento Portland es un conglomerante hidráulico compuesto principalmente por el clínker, un material granular que contiene silicato tricálcico (C_3S), silicato dicálcico (C_2S), aluminato tricálcico (C_3A) y ferroaluminato tetracálcico (C_4AF), mezclado finalmente con una pequeña proporción de yeso para regular el fraguado, obtenido mediante la calcinación de calizas y arcillas. Es el material base de la construcción moderna, esencial para la producción de hormigón y morteros gracias a su capacidad de fraguar y endurecer al entrar en contacto con el agua.

Hay multitud de estudios referidos a estos, por citar algunos estudios recientes las propiedades térmicas de la pasta de cemento pueden obtenerse en función de la relación agua cemento en Baghban et al. [16] y las propiedades mecánicas pueden obtenerse en el libro de P.K. Mehta, P.J.M. Monteiro [17].

2.3.2. Fibra de basalto

Las fibras de basalto son filamentos continuos obtenidos mediante la fusión de roca basáltica de origen volcánico, caracterizadas por su alta resistencia mecánica y estabilidad térmica. Se consideran una alternativa sostenible y de alto rendimiento a las fibras de vidrio, siendo fundamentales en el refuerzo de materiales compuestos y en aplicaciones de aislamiento industrial. La elección de estas se debe a un mejor comportamiento térmico durante los ciclos de carga. En una matriz, las fibras actúan incrementando la tenacidad, ductilidad y control de agrietamiento. Estudios destacan que dosis del 0.5-2% en volumen mejoran la resistencia a flexión hasta un 50% y la durabilidad frente a ciclos.

Las propiedades térmicas se pueden estimar según Khalid et al. 2011 [14].

Las propiedades mecánicas se pueden extraer de Jianxun Liu et al. [15], en este estudio se observa que no solo mejora térmicamente respecto a la fibra de vidrio sino que también lo hace ligeramente en términos mecánicos lo que lo hace mejor candidato para el estudio actual.

2.3.3. Ilmenita

La ilmenita ($FeTiO_3$) es un mineral óxido de hierro y titanio, de color negro o gris, siendo la principal mena de titanio. Es un material fundamental en la industria para

obtener dióxido de titanio y titanio metálico.

Para obtener sus propiedades mecánicas se usa una aproximación basada en la velocidad de propagación de ondas, la cual da una medida de la rigidez del material:

Con densidad y la velocidad de propagación de ondas obtengo K y G (bulk and shear moduli) con esos obtengo módulo de Young y coeficiente de Poisson. Si bien este tipo de inclusiones no afectarán a la resistencia del material tanto como las fibras, el objetivo es obtener la mejor aproximación posible del material:

Propagación de ondas tipo S : $V_s = 4,2 km/s$

Propagación de ondas tipo P: $V_p = 7,7 km/s$

$$G_{ilmenita} = \rho_{ilmenita} \cdot V_s \quad (2.3.1)$$

$$K_{ilmenita} = \rho_{ilmenita} \cdot (V_s^2 - \frac{4}{3} \cdot V_s^2) \quad (2.3.2)$$

Por otra parte, el calor específico se calcula con los calores de FeO y TiO₂ y su proporción en el material. El coeficiente de expansión se obtiene con una relación a partir del coeficiente de Grüneisen, la densidad y el calor obtenido.

$$Cp_{ilmenita} = Cp_{FeO} \cdot Fe_{Masa} + cp_{TiO_2} \cdot (1 - Fe_{Masa}) \quad (2.3.3)$$

$$\alpha_{ilmenita} = C_{Gruinessen} \cdot Cp_{ilmenita} \cdot \rho_{ilmenita} \quad (2.3.4)$$

Propiedades mecánicas	Densidad (t/m³)	Módulos de Young (GPa)	Resistencia a tracción (MPa)	Coefficiente de Poisson (1/1)
Pasta de cemento	2.5-3	30	25	0.2
Fibra de basalto	2.75	95-102	2443-3818	0.22
Ilmenita	4.79	84.495	N/A	0.288

Tabla 2.3.1: Propiedades mecánicas de los materiales empleados en el estudio

Propiedades térmicas	Calor específico (MJ/kgK)	Conductividad (W/mK)	Coefficiente de expansión (1/K)
Pasta de cemento	0.88	0.28	5e-6
Fibra de basalto	0.86	0.035	5.5e-6
Ilmenita	0.681	3	7.7e-15

Tabla 2.3.2: Propiedades mecánicas de los materiales empleados en el estudio

2.4. ELEMENTOS FINITOS PARA ALMACENAMIENTO DE ENERGÍA

El objetivo del Método de Elementos Finitos (FEM) en este estudio es actuar como base fiable para la generación de datos sintéticos que soportarán el desarrollo del gemelo digital. Más allá de la formulación matemática detallada en el Capítulo de Tecnología de Almacenamiento, el modelado numérico permite resolver de forma acoplada problemas termomecánicos no lineales complejos, como son los fenómenos de transferencia de calor en régimen transitorio, resistencia térmica de contacto y tensiones térmicas inducidas por gradientes severos de temperatura [8].

En el contexto de matrices de cemento modificadas con aditivos, la literatura científica demuestra que el FEM reproduce con precisión la respuesta transitoria global, siempre que se implemente una correcta calibración y validación experimental [7].

Para este trabajo, el modelo FEM se formulará bajo un enfoque de propiedades termofísicas efectivas homogeneizadas, capturando el impacto de los aditivos en la conductividad térmica y la capacidad calorífica volumétrica del sistema. El espacio muestral obtenido será para entrenar la red neuronal subrogada, asegurando su capacidad de generalización con un coste computacional mucho menor en fase de explotación.

2.5. MODELADO MEDIANTE REDES NEURONALES

El último elemento necesario para este proyecto es una herramienta capaz de realizar predicciones rápidas y capaz de adaptarse a patrones no lineales con multitud de datos de entrada. Los modelos de redes neuronales cumplen ambos requisitos captando dinámicas y efectos dependientes del estado inicial [?], con la contrapartida de necesitar un entrenamiento costoso y depender tanto de los valores de entrenamiento como de los hiperparámetros de la red.

Capítulo 3

Tecnología de almacenamiento

En este capítulo se explica el sistema simulado y estudiado en profundidad. El objetivo es conocer el comportamiento del almacenamiento de energía térmica por calor sensible con cemento dopado con algunos elementos. Lo primero que se define es cómo se obtiene la energía para el sistema, en este caso se hace uso de colectores cilindro parabólicos solares, los cuales usan un fluido calo-portante para el transferir el calor y se examina cómo simular su comportamiento. A continuación, se introducen los materiales que se utilizan para el almacenamiento de calor, importante señalar que las adiciones de material requieren ecuaciones concretas para las inclusiones. En ese apartado, conocida la forma de caracterizar el material, se pasa a simular mediante elementos finitos el comportamiento, estas simulaciones serán el *groundtruth* o datos de entrenamiento.

3.1. DATOS DISPONIBLES DEL HTF

Se considera el caso de uso de un aceite calo-portante comercial conocido como *Downtherm*, [5], utilizado en CCP. La importancia que tiene este fluido es la capacidad que tiene de aportar el calor que obtuvo al pasar por los colectores al sistema de almacenamiento basado en cemento

Dadas las propiedades de los HTF como las indicadas en la tabla 3.1.1 se obtienen unos valores del número de Reynolds 3.1.2 y de Prandtl 3.1.3. Con esa información y la correlación de Nusselt 3.1.1 se obtienen los coeficientes de película para el interior de la tubería 3.1.5.

Tabla 3.1.1: Propiedades termofísicas del HTF Dowtherm A en función de la temperatura [5].

Propiedad	Unidad	20 °C	100 °C	200 °C	300 °C	400 °C
Densidad (ρ)	kg/m^3	1064	999	913	817	694
Visc. Cinemática (ν)	m^2/s	$4,03 \cdot 10^{-6}$	$9,56 \cdot 10^{-7}$	$4,32 \cdot 10^{-7}$	$2,71 \cdot 10^{-7}$	$2,11 \cdot 10^{-7}$
Visc. Dinámica (μ)	$Pa \cdot s$	0.004288	0.000955	0.000394	0.000221	0.000146
Calor Específico (c_p)	kJ/kgK	1.546	1.775	2.048	2.314	2.628
Difusividad (α)	m^2/s	$9,12 \cdot 10^{-5}$	$7,90 \cdot 10^{-5}$	$6,95 \cdot 10^{-5}$	$6,35 \cdot 10^{-5}$	$5,48 \cdot 10^{-5}$
Conductividad (k)	W/mK	0.15	0.14	0.13	0.12	0.10

Tabla 3.1.2: Número de Reynolds (Re) en función de la velocidad y la temperatura.

V [m/s]	20 °C	100 °C	200 °C	300 °C	400 °C
0.2	12 406.95	52 301.26	115 740.74	184 501.85	236 966.82
0.5	31 017.37	130 753.14	289 351.85	461 254.61	592 417.06
0.8	49 627.79	209 205.02	462 962.96	738 007.38	947 867.30
1.0	62 034.74	261 506.28	578 703.70	922 509.23	1 184 834.12
1.2	74 441.69	313 807.53	694 444.44	1 107 011.07	1 421 800.95

Tabla 3.1.3: Número de Prandtl (Pr) calculado para cada rango de temperatura.

Temperatura	20 °C	100 °C	200 °C	300 °C	400 °C
Pr	0.04419	0.01211	0.00621	0.00427	0.00385

3.1.1. Cálculo de Coeficiente de película

Se realizan simulaciones considerando los coeficientes de temperatura obtenidos a partir de la fórmula de Hausen para conductos tubulares con flujo interno.

$$Nu = 0,23 \cdot Re^{0,8} \cdot Pr^{0,4} \quad (3.1.1)$$

Tabla 3.1.4: Número de Nusselt (Nu) en función de la velocidad y la temperatura.

V [m/s]	20 °C	100 °C	200 °C	300 °C	400 °C
0.2	124.3958135	234.3014663	338.7297111	423.3287743	496.117778
0.5	258.9153337	487.6710932	705.0262686	881.1093222	1032.611118
0.8	377.0975905	710.2692281	1026.83647	1283.292873	1503.94787
1	450.7977792	849.0846898	1227.522031	1534.100433	1797.880381
1,2	521.5870074	982.4173118	1420.281048	1775.001766	2080.203344
1.5	623.526298	1174.421565	1697.861664	2121.90922	2486.759585
2	784.8845213	1478.34231	2137.23999	2671.023993	3130.291557

Tabla 3.1.5: Coeficientes de película (h) [W/m^2K] en función de la velocidad y la temperatura.

V [m/s]	20 °C	100 °C	200 °C	300 °C	400 °C
0.2	74.63748808	131.2088211	176.1394498	203.1978117	198.4471112
0.5	155.3492002	273.0958122	366.6136597	422.9324746	413.0444473
0.8	226.2585543	397.7507678	533.9549641	615.9805789	601.579148
1	270.4786675	475.4874263	638.3114559	736.3682081	719.1521525
1.2	312.9522045	550.1536946	738.5461452	852.0008477	832.0813377
1.5	374.1157788	657.6760764	882.8880652	1018.516426	994.7038341
2	470.9307128	827.8716936	1111.364795	1282.091517	1252.116623

Predicciones

Dado que el objetivo último del estudio sería poder caracterizar los cambios que se producen en el equipo, es necesario tener una aproximación con mayor precisión. Para ello se ha formulado una regresión lineal polinomial para interpolar el rango 200°C a 400°C que es el de mayor interés para el estudio, Figura 3.1.1. Puesto que se dispone de tres valores en ese rango, [5], basta con un polinomio de tercer grado.

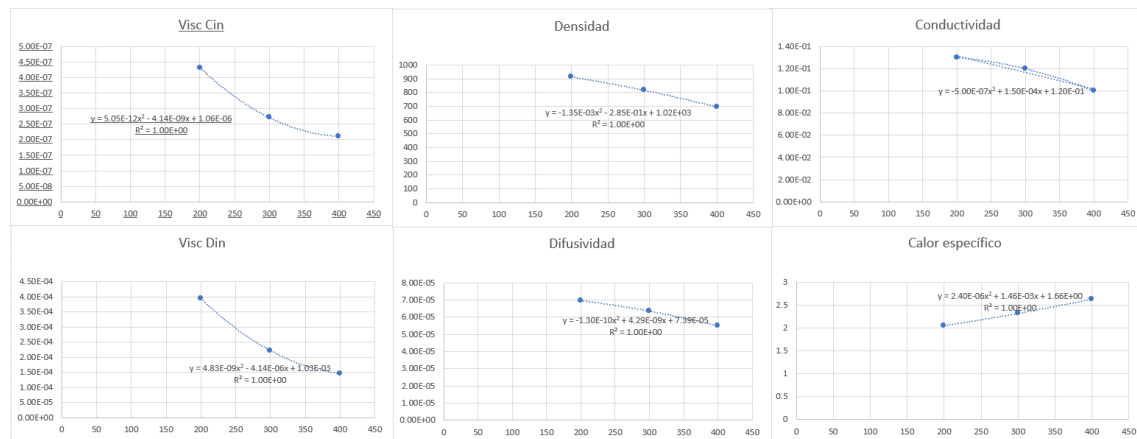


Figura 3.1.1: Ecuaciones obtenidas tras aproximar con polinomios cada una de las variables de interés del HTF

3.2. ALMACENAMIENTO DE ENERGÍA TÉRMICA

El elemento individual que se va a estudiar es un cilindro de cemento con espacio en el centro del eje para una tubería de acero por la que circula el HTE. El comportamiento del acero inoxidable de la tubería se considera conocido, se estudia por tanto las propiedades del cemento y sus modificaciones al añadir aditivos.

3.2.1. Propiedades combinación hormigón e ilmenita

El primero de los aditivos empleados es la ilmenita. Material mineral con alta conductividad. Se plantea puesto que es utilizado en la obtención de titanio y a menudo forma parte de los materiales de desecho en ese proceso.

Se hace uso de ecuación Mori-Tanaka para inclusiones esferoidales, Ghatage et al. [18], donde se especifican las ecuaciones para el cálculo de las propiedades E, nu:

$$f_1 = (G_m \cdot (9 \cdot K_m + 8 \cdot G_m)) / (6 \cdot (K_m + 2 \cdot G_m)) \quad (3.2.1)$$

$$K_{ef} = K_m + \frac{V_f \cdot (K_f - K_m)}{1 + (1 - V_f) \cdot \frac{K_f - K_m}{K_m + (4/3) \cdot G_m}} \quad (3.2.2)$$

$$G_{ef} = G_m + \frac{V_f \cdot (G_f - G_m)}{1 + (1 - V_f) \cdot \left(\frac{G_f - G_m}{G_m + f_1} \right)} \quad (3.2.3)$$

$$E_{ef} = (9 \cdot K_{efectivo} \cdot G_{ef}) / (3 \cdot K_{ef} + G_{ef}) \quad (3.2.4)$$

$$nu_{ef} = (3 \cdot K_{ef} - 2 \cdot G_{ef}) / (2 \cdot (3 \cdot K_{ef} + G_{ef})) \quad (3.2.5)$$

La conductividad se obtiene con otra adaptación de Mori-Tanaka/Hashin-Shtrikman para inclusiones esferoidales [18].

$$A = V_p \cdot (k_p - k_m) \quad (3.2.6)$$

$$B = 1 + V_m \cdot ((k_p - k_m) / (3 \cdot k_m)) \quad (3.2.7)$$

$$k_{efectiva} = k_m + A/B \quad (3.2.8)$$

El coeficiente de expansión se obtiene con una a partir de los valores de K (*shear moduli strength*) de matriz e inclusiones.

$$\alpha_c = \frac{(\alpha_p - \frac{(1/K_p - 1/K_c)}{(1/K_m - 1/K_c)} \cdot \alpha_m)}{(1 - \frac{(1/K_p - 1/K_c)}{(1/K_m - 1/K_c)})} \quad (3.2.9)$$

El calor específico se obtiene con homogeneización paralela igual que en las combinaciones de óxidos.

$$Cp_{efectiva} = Cp_m \cdot M_m + Cp_p \cdot (1 - M_p) \quad (3.2.10)$$

3.2.2. Propiedades combinación hormigón y fibra

Uso de la ecuación de Mori-Tanaka para fibras en una dirección: Se definen los tensores de rigidez de la matriz C_m y de las fibras C_f , dependientes de sus correspondientes módulos de Young y coeficiente de Poisson. Además se ha de definir el tensor de Eshelby que depende de del coeficiente de Poisson de la matriz y de la relación entre longitud y diámetro de la fibra:

$$S_{11} = S_{12} = S_{13} = 0 \quad (3.2.11)$$

$$S_{22} = S_{33} = \frac{5 - 4\nu_m}{8(1 - \nu_m)} \quad (3.2.12)$$

$$S_{21} = S_{31} = \frac{\nu_m}{2(1 - \nu_m)} \quad (3.2.13)$$

$$S_{23} = S_{32} = \frac{4\nu_m - 1}{8(1 - \nu_m)} \quad (3.2.14)$$

$$S_{44} = \frac{1 - 2\nu_m}{4(1 - \nu_m)} \quad (3.2.15)$$

$$S_{55} = S_{66} = \frac{1}{4} \quad (3.2.16)$$

Se obtienen los tensores de concentración diluido 3.2.17 y tensor de concentración de deformación de Mori-Tanaka 3.2.18 para obtener finalmente el tensor de rigidez efectiva 3.2.19.

$$\mathbf{A}_{dil} = [\mathbf{I} + \mathbf{S} \mathbf{C}_m^{-1} (\mathbf{C}_f - \mathbf{C}_m)]^{-1} \quad (3.2.17)$$

$$\mathbf{A}_{MT} = \mathbf{A}_{dil} [(1 - V_f) \mathbf{I} + V_f \mathbf{A}_{dil}]^{-1} \quad (3.2.18)$$

$$\mathbf{C}_{eff} = \mathbf{C}_m + V_f (\mathbf{C}_f - \mathbf{C}_m) \mathbf{A}_{MT} \quad (3.2.19)$$

Posteriormente se aplica una homogeneización de las propiedades de forma que el material vuelva a ser isotrópico:

$$K = (C_{11} + C_{22} + C_{33} + 2 \cdot (C_{12} + C_{23} + C_{13}))/9 \quad (3.2.20)$$

$$G = (C_{11} + C_{22} + C_{33} - C_{12} - C_{23} - C_{13} + 3 \cdot (C_{44} + C_{55} + C_{66}))/15 \quad (3.2.21)$$

$$E = \frac{(9 \cdot K \cdot G)}{3 \cdot K + G} \quad (3.2.22)$$

$$\nu = \frac{(3 \cdot K - 2 \cdot G)}{2 \cdot (3 \cdot K + G)} \quad (3.2.23)$$

Tabla 3.2.1: Propiedades de cemento con aditivos

Propiedad	Fibra 0.1 %	Ilmenita 5 %	Fibra 0.1 % + Ilmenita 5 %
Mecánicas			
Módulo de Young, E (MPa)	30064.98	31771.91	31836.96
Coeficiente de Poisson, ν	0.1999	0.2043	0.2043
Térmicas			
Densidad, ρ (t/mm ³)	$2,5004 \times 10^{-9}$	$2,6374 \times 10^{-9}$	$2,6376 \times 10^{-9}$
Conductividad Térmica, K_c (W/m · K)	0.2795	0.3204	0.3197
Calor Específico (mJ/t · K)	$8,7997 \times 10^8$	$8,2720 \times 10^8$	$8,2725 \times 10^8$
Coef. Expansión Térmica, α_c (1/K)	$5,0000 \times 10^{-6}$	$4,6156 \times 10^{-6}$	$4,6166 \times 10^{-6}$

Efecto en características

Determinada la forma de caracterizar el material, el efecto de las inclusiones es el que se muestra en las Figuras 3.2.1 y 3.2.2.

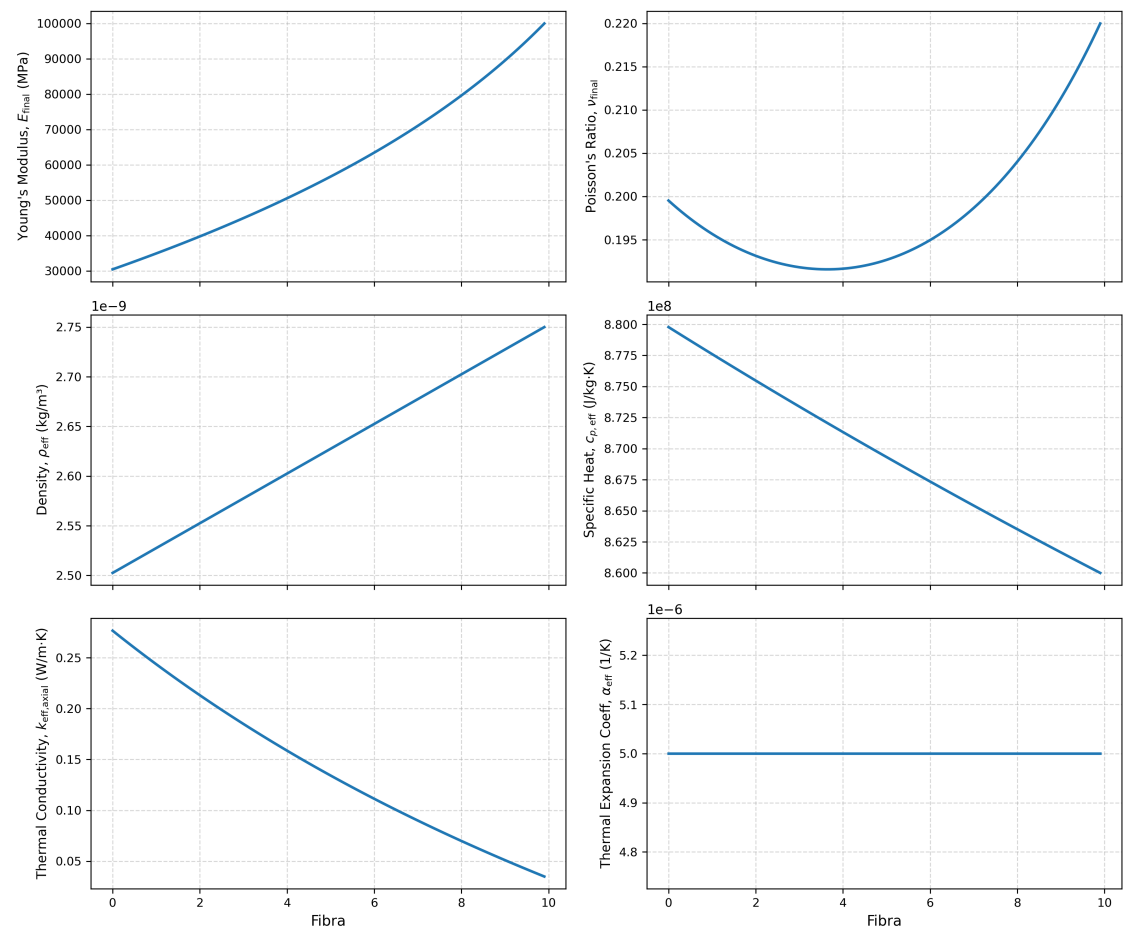


Figura 3.2.1: Efecto en las características del material al añadir fibra

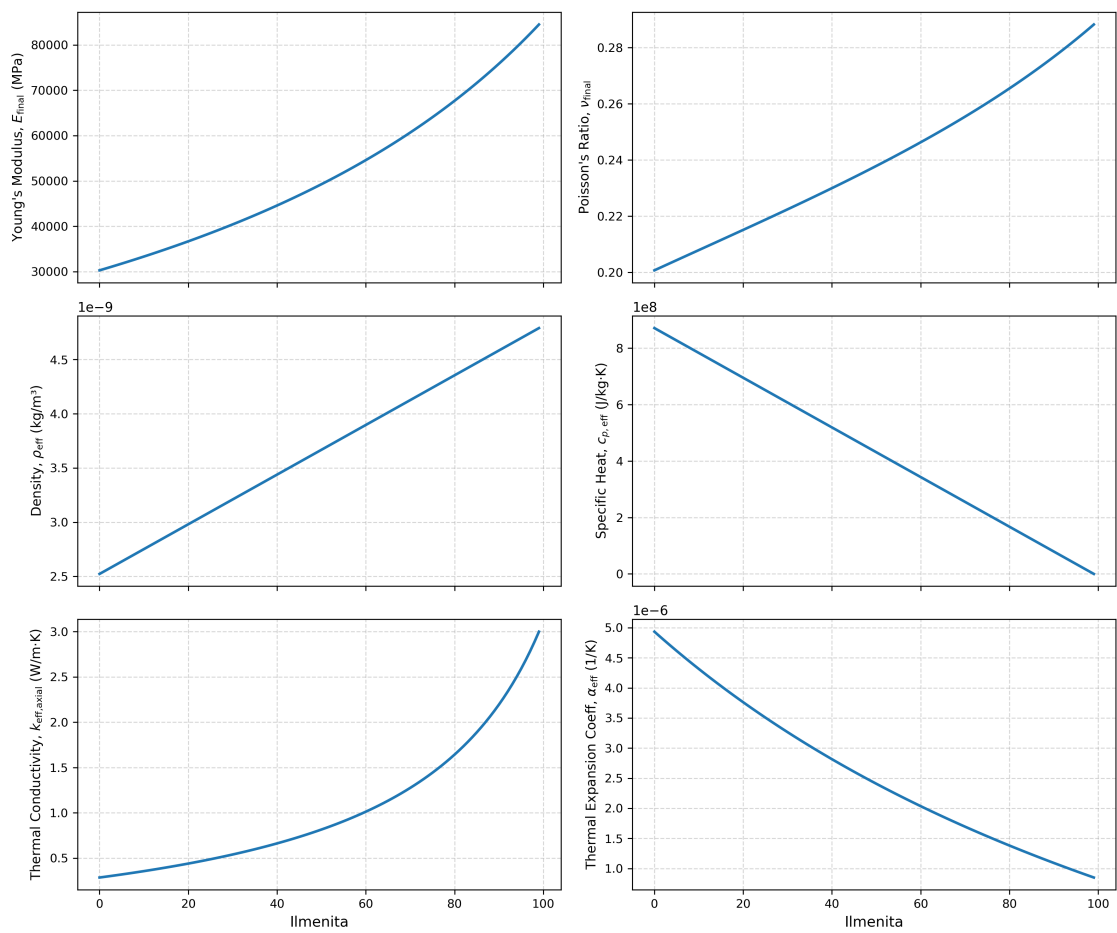


Figura 3.2.2: Efecto en las características del material al añadir ilmenita

El método propuesto para el cálculo de la combinación es hacer las combinaciones en serie, de forma que las características de la combinación de ilmenita y cemento sean la base para el cálculo de la combinación ilmenita, fibra y cemento. Se han de hacer varias suposiciones para que se cumpla esta suposición:

- Las partículas son nanométricas o polvo, muy inferiores en tamaño a las fibras.
- El volumen total de inclusiones es tal que, estadísticamente, no hay contacto entre fibras. En general se considera que el volumen total es inferior al 30 % para garantizarlo.

Por tanto para conocer el porcentaje máximo de cada elemento:

$$(\%_{Ilm} + \%_{Cem} + \%_{Fib}) = 1 \quad (3.2.24)$$

Con $\%_{Ilm}$ se hace referencia al porcentaje final de la mezcla, de forma que el porcentaje que se tiene que utilizar en la primera mezcla sea $\frac{\%_{Ilm}}{(1 - \%_{Fib})}$

$$\%_{Ilm} + \%_{Fib} = 0,3 \quad (3.2.25)$$

Se tienen dos ecuaciones, para tener una tercera que permita conocer los rangos aceptables de aditivos se añade:

$$\%_{Ilm} = \%_{Fib} \cdot 10 \quad (3.2.26)$$

Por las siguientes razones:

1. Se reduce la posibilidad de contacto entre fibras.
2. La ilmenita puede proceder del procesamiento del titanio como recurso de deshecho, siendo así una segunda vida para el recurso.

- $\%_{Ilm} = 0,273$
- $\%_{Fib} = 0,027$
- $\%_{Cem} = 0,7$

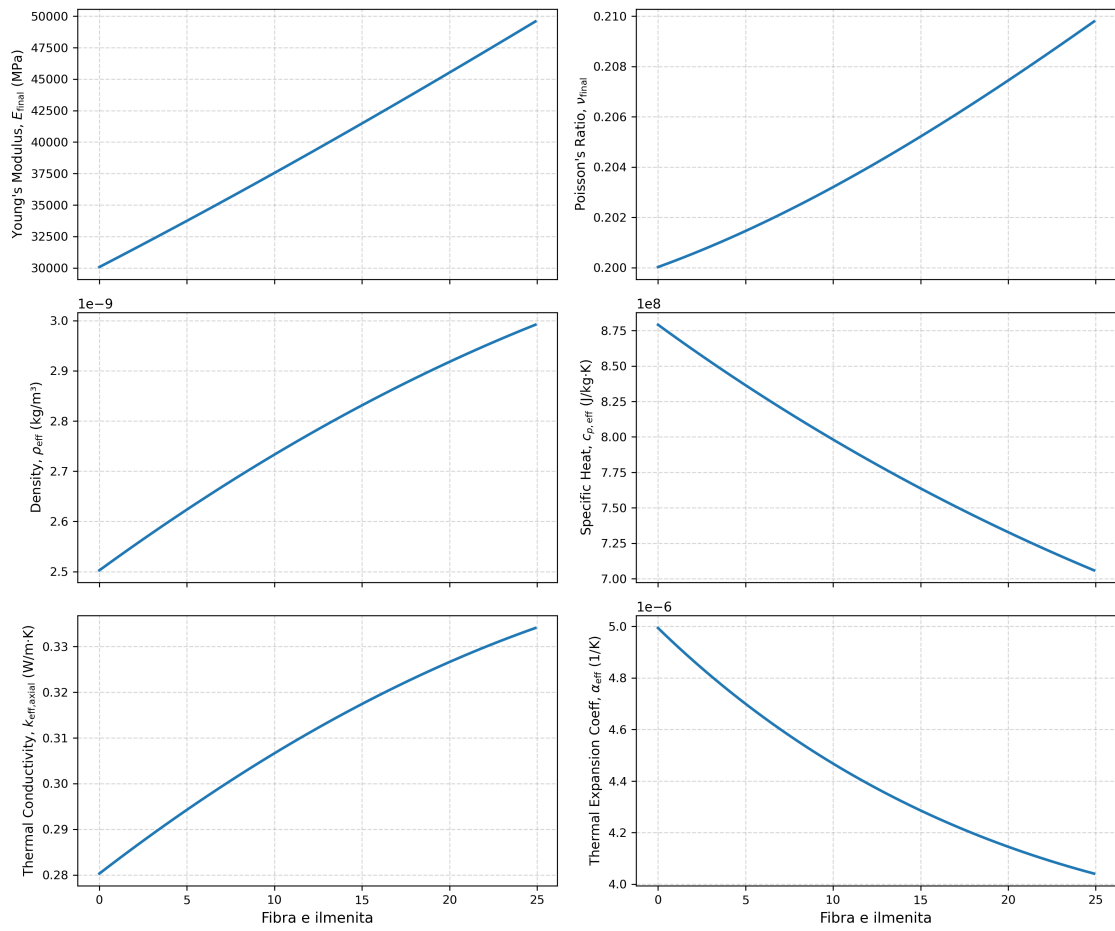


Figura 3.2.3: Efecto en las características del material al añadir ilmenita y fibra en la concentración máxima teórica

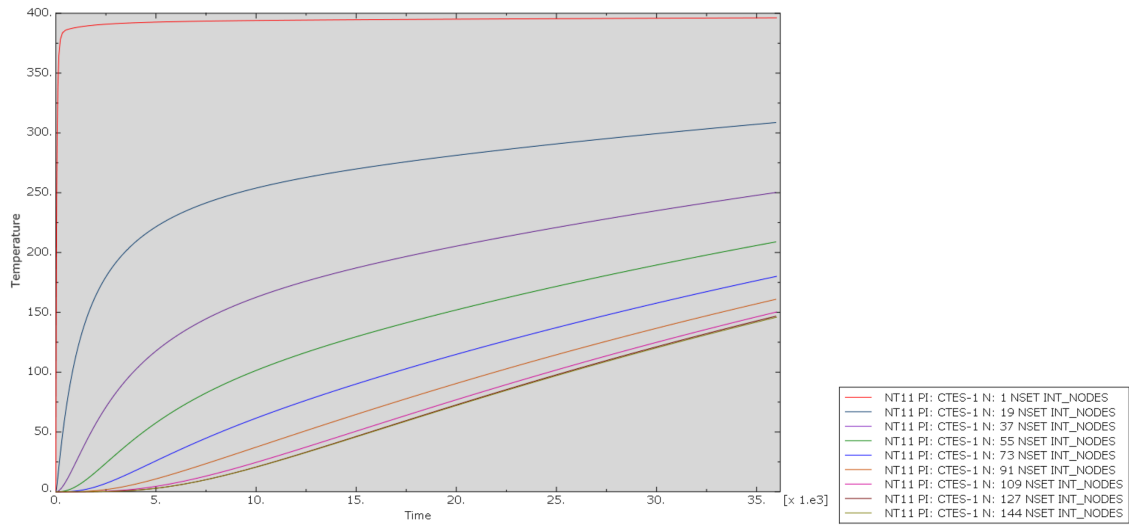


Figura 3.2.4: Comportamiento durante 10 horas con HTF a 400°C y 2 m/s del CTES solo con cemento

Finalmente, como comprobación de este efecto, se simula con el modelo que se verá en la siguiente sección el comportamiento en el tiempo de una serie de puntos equidistantes y alineados en la cara de entrada del sistema sin aditivos y otro del sistema con este máximo calculado:

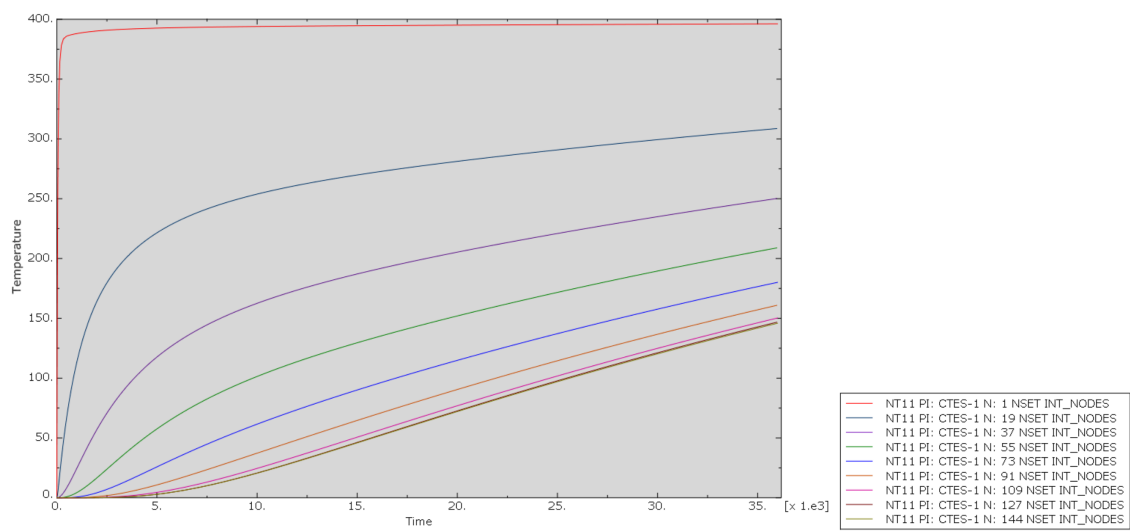


Figura 3.2.5: Comportamiento durante 10 horas con HTF a 400°C y 2 m/s del CTES con el máximo teórico de aditivos

Se observa una mayor temperatura alcanzada así como mayor rapidez para captar calor desde el inicio, por lo que también tendrá mayor capacidad de ceder calor cuando se requiera.

3.3. MÉTODO ELEMENTOS FINITOS

El Método de los Elementos Finitos (MEF) es una técnica numérica para hallar soluciones aproximadas a problemas de contorno descritos por ecuaciones diferenciales parciales. El dominio se discretiza en subdominios denominados elementos, interconectados por nodos. La aproximación de la variable de campo (temperatura o desplazamiento) dentro de cada elemento se define mediante funciones de forma $[N]$.

3.3.1. Formulación Térmica (Estado Estacionario)

En un análisis térmico, el objetivo es resolver la ecuación de conducción de calor. A nivel elemental, la interrelación térmica se rige por la matriz de conductividad y el vector de flujo:

$$[K_t]T = Q \quad (3.3.1)$$

Donde:

- $[K_t]$ es la matriz de conductividad térmica del elemento, definida como:

$$[K_t] = \int_V [B]^T [D] [B] dV \quad (3.3.2)$$

siendo $[B]$ la matriz de gradientes y $[D]$ la matriz de conductividad del material.

- $\{T\}$ es el vector de temperaturas nodales.
- $\{Q\}$ es el vector de cargas térmicas (flujos, generación interna o convección).

3.3.2. Formulación Elástica (Mecánica)

Para el análisis estructural, se busca el equilibrio estático basado en la rigidez del material. La ecuación fundamental entre fuerzas y desplazamientos es:

$$[K_m]\{u\} = \{F_{ext}\} + \{F_{th}\} \quad (3.3.3)$$

Donde:

- $[K_m]$ es la matriz de rigidez mecánica:

$$[K_m] = \int_V [B]^T [C] [B] dV \quad (3.3.4)$$

siendo $[C]$ el tensor de elasticidad o matriz constitutiva.

- $\{u\}$ es el vector de desplazamientos nodales.
- $\{F_{ext}\}$ representa las fuerzas externas aplicadas.

- $\{F_{th}\}$ son las fuerzas térmicas equivalentes que representan la expansión o contracción.

3.3.3. Acoplamiento Termo-Mecánico

La interrelación reside en la forma en la que el gradiente de temperatura genera deformaciones mecánicas. La deformación total ϵ se descompone en una parte elástica y una térmica:

$$\epsilon = \epsilon_e + \epsilon_{th} \quad (3.3.5)$$

La deformación térmica se modela según el coeficiente de expansión térmica α :

$$\epsilon_{th} = \alpha \Delta T \quad (3.3.6)$$

Donde ΔT es el diferencial de temperatura obtenido del análisis térmico previo. Esta relación permite que el vector de carga térmica $\{F_{th}\}$ en la ecuación elástica se calcule como:

$$\{F_{th}\} = \int_V [B]^T [C] \{\epsilon_{th}\} dV \quad (3.3.7)$$

|

3.4. MODELO EMPLEADO

El elemento individual que se va a estudiar es un cilindro de cemento de 250mm de alto, 250mm de diámetro, con espacio en el centro del eje para una tubería de acero de 1 pulgada por la que circula el HTE.

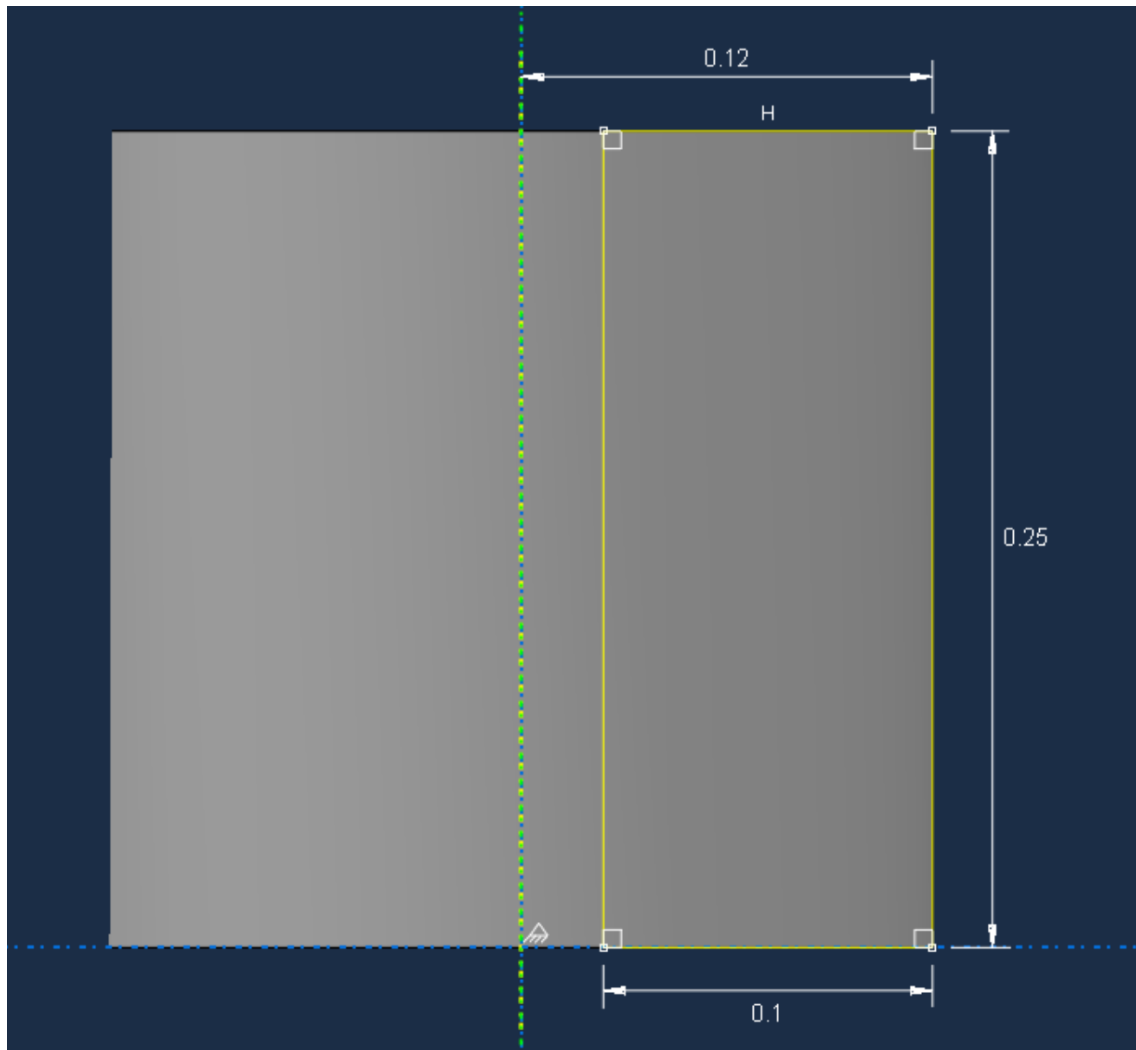


Figura 3.4.1: Sección que se revoluciona para obtener la figura 3D

3.4.1. Condiciones de contorno

Para simular un cilindro de cemento que tenga permitida la expansión libre, se simula dicho cilindro con una tubería de acero que es a la que constriñe los desplazamientos. Para garantizar que no haya rotación pero se mantenga la expansión radial,

se limitan todos los desplazamientos de uno de los puntos de la tubería y en el extremo diametralmente opuesto se limitan solo los desplazamientos tangenciales. Esto evita rotaciones manteniendo las tensiones del material del mismo orden que si solo se hubiera empotrado uno de los puntos 3.4.2.

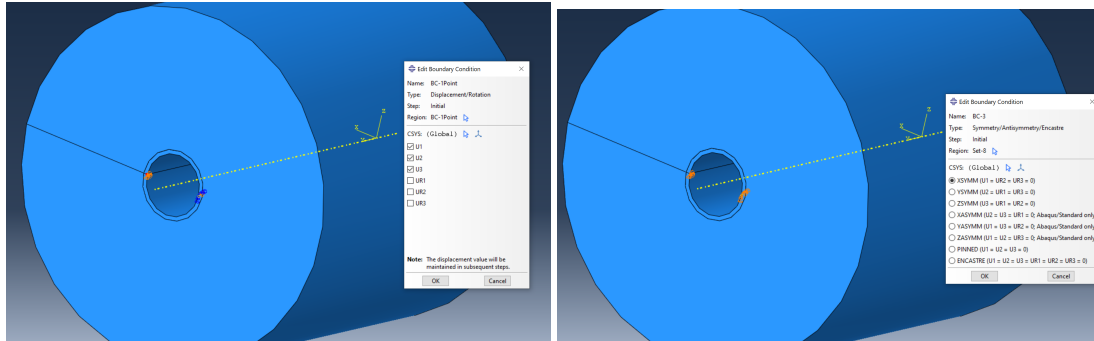


Figura 3.4.2: Condiciones de contorno en tubería

3.4.2. Cargas aplicadas

El método para aplicar estas cargas debidas al calor, de forma parecida a lo que ocurre con las condiciones de contorno, requiere algunas consideraciones y explicaciones. Lo primero es la necesidad de un cálculo termomecánico acoplado, dando lugar a una resolución no lineal con elementos que tengan grados de libertad espaciales y térmicos durante la definición de la malla.

En el interior de la tubería de acero se simula el líquido a alta temperatura aportando calor al conjunto. Las tensiones introducidas por la expansión del acero al calentarse y tener un coeficiente de expansión diferente al cemento y, posteriormente, la expansión del cemento al recibir ese calor, serán todas las cargas mecánicas que sufrirá el sistema.

Para simular ese proceso, se hace uso de una subrutina para indicar de forma precisa el calor que se está inyectando en el sistema de almacenamiento, ver Anexo Cálculo automático en Abaqus. La subrutina calcula lo siguiente:

- Dada una temperatura de entrada y una temperatura de salida, obtiene el coeficiente de película a aplicar para cada temperatura.
- Realiza un regresión lineal entre los coeficientes de película para simular la caída de temperatura entre extremos. En cada elemento aporta calor en función de la temperatura obtenida en la regresión.
- Realiza un bucle con los materiales deseados para realizar cada cálculo.
- Extrae la información de unos puntos seleccionados previamente en un .csv y elimina los archivos generados por el programa de elementos finitos (salvo que se indique lo contrario para un estudio fino de algún caso particular) para no exceder memoria de almacenamiento.

La obtención del modelo final requiere primero un estudio del funcionamiento y los resultados que se obtienen con, en este caso, Abaqus como programa de elementos finitos.

3.4.3. Configuración de cálculo

El cálculo está configurado como un análisis de transferencia de calor en régimen transitorio (figura 1) con un tiempo total de simulación de 36000 s (10 horas de operación, un ciclo típico para sistemas de almacenamiento térmico). Método de Resolución (Equation Solver):

- Método Directo. Para problemas de transferencia de calor acoplados o con aditivos no lineales, el solver directo proporciona una estabilidad numérica robusta y exacta en la resolución del sistema lineal de ecuaciones $K \cdot \Delta T = R$, a costa de un mayor uso de memoria RAM en comparación con los métodos iterativos.

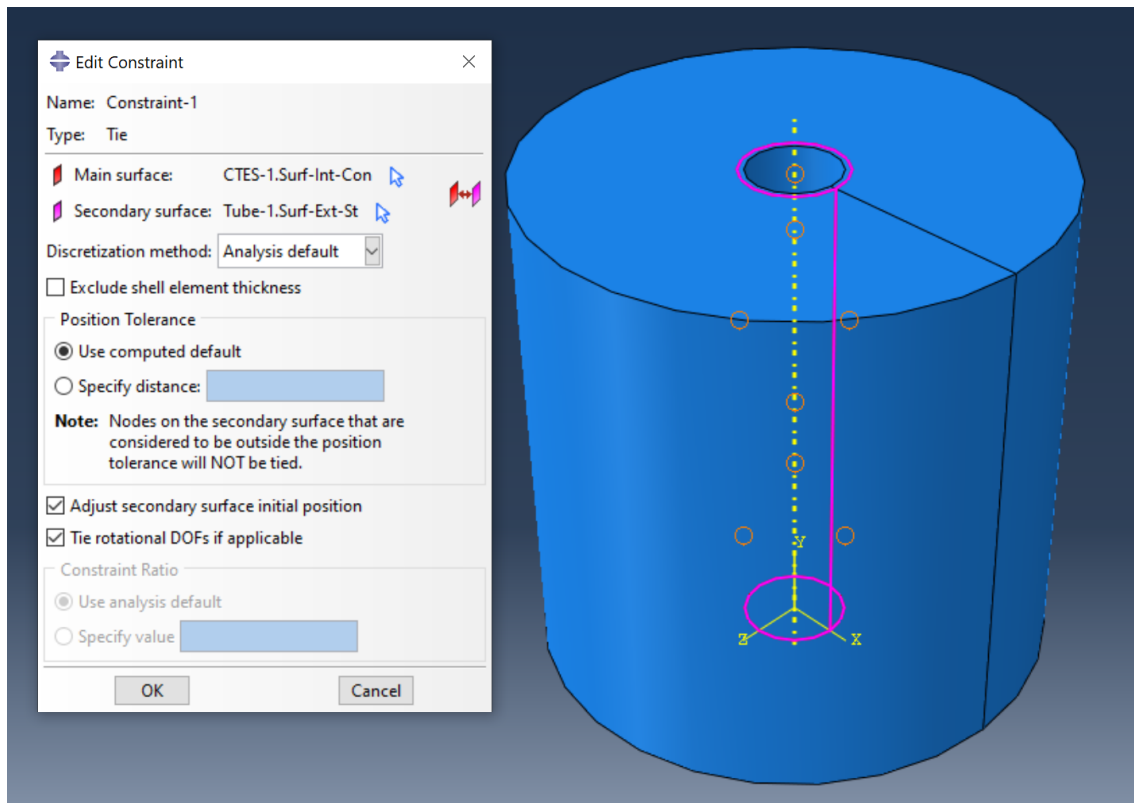


Figura 3.4.3: Condiciones de contacto entre tubería y CTES

- Full Newton (Newton-Raphson completo). Esto implica que la matriz de rigidez térmica (o matriz de conductividad/capacidad calórica elemental) se reformula y actualiza en cada iteración interna de cada incremento. Es la opción más fiable para capturar no linealidades severas, como la variación de propiedades térmicas con grandes cambios de temperatura (especialmente al inicio).

El algoritmo de control de incrementos está configurado en modo Automático (figura 2), delegando en Abaqus la gestión del tamaño del paso temporal (Δt) para asegurar la convergencia bajo los siguientes límites:

- Restricciones de Tiempo:
 - Incremento inicial: 10 s. Incremento mínimo permitido: 0.36 s (si el corte de incrementos baja de este valor por problemas de convergencia, el análisis se detiene).
 - Incremento máximo permitido: 36 s (acota el paso para evitar perder resolución transitoria frente a gradientes térmicos rápidos). Número máximo de incrementos: 10000.
 - Criterio de Parada por Estado Estacionario: Se activa la opción de finalizar el step si el cambio de temperatura global por segundo es inferior a 0.0001 °C/s, lo que optimiza el coste computacional si el bloque de almacenamiento alcanza el equilibrio antes de las 10 horas.

Por último, mencionar el mallado sencillo en 3689 (aproximadamente 0.03 m de dimensión característica) elementos tipo C3D8T hexaédricos con 4 nodos.

3.4.4. Simulaciones realizadas

Simulación inicial

Primero, es necesario conocer el comportamiento esperable de este sistema. Para ello, se simula lo siguiente:

- Tiempo simulado: 36000 segundos, por ser un sistema que depende de la radiación solar es poco probable más de 10 horas de irradiancia suficiente.
- Material: Cemento con comportamiento homogéneo
- Convección externa: Se realiza con temperatura constante 20°C y un valor típico de coeficiente de película fluido sólido $20\text{W}/\text{m}^2$.
- Radiación externa: Se realiza con temperatura constante 20°C .
- Convección interna: Aporte de calor con un fluido a temperatura constante de 400°C y un valor de coeficiente de película de $100\text{W}/\text{m}^2$.

Se obtiene el siguiente estado para el final de la simulación 3.4.4. Por las pérdidas asociadas a convección y radiación parece necesario un aislamiento con el entorno.

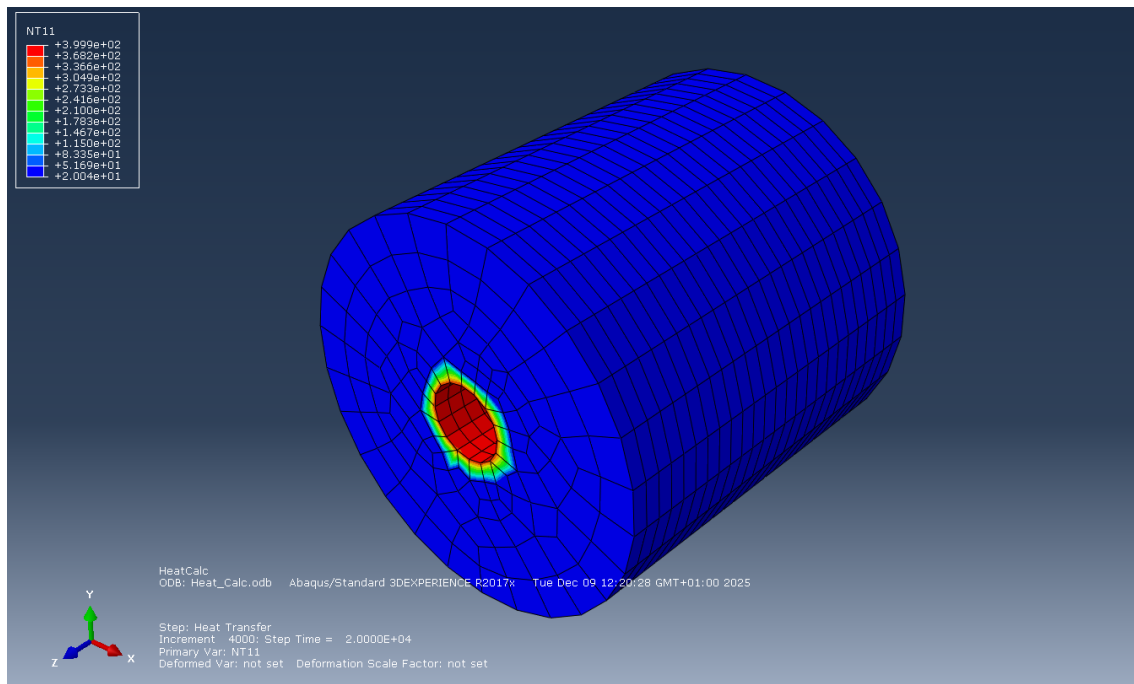


Figura 3.4.4: Gradiente de temperatura para la simulación inicial

Simulación aislada

Primero, es necesario conocer el comportamiento esperable de este sistema. Para ello, se simula lo siguiente:

- Tiempo simulado: 36000 segundos, por ser un sistema que depende de la radiación solar es poco probable más de 10 horas de irradiancia suficiente.
- Material: Cemento con comportamiento homogéneo
- Aislado del exterior.
- Convección interna: Aporte de calor con un fluido a temperatura constante de 400°C y un valor de coeficiente de película de 100W/m^2 .

Se obtiene el siguiente estado para el final de la simulación 3.4.5. Cabe destacar que la temperatura del exterior (329°C) es mucho más cercana a la del interior que sin aislar lo cual era el objetivo. Se mantendrá la solución aislada.

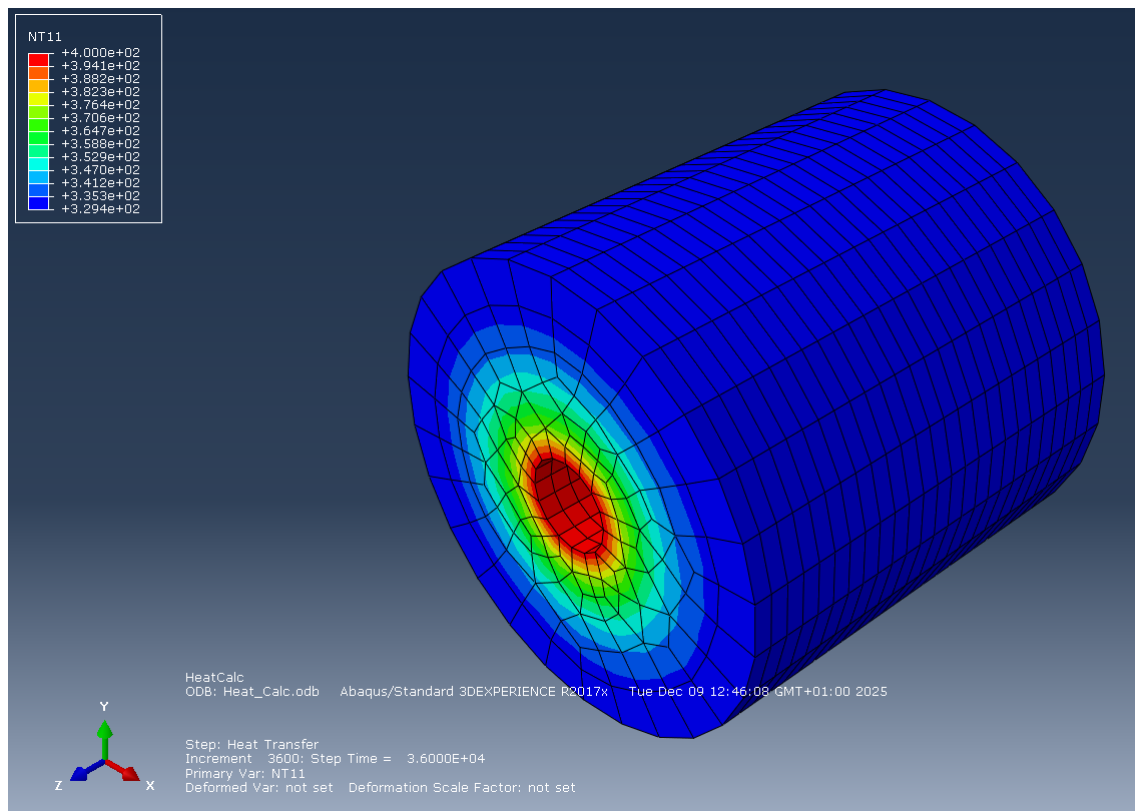


Figura 3.4.5: Gradiente de temperatura para la simulación aislada

Simulación calculando la adición de calor del fluido

Para que la red pueda adaptarse a posibles escenarios, se plantea que el calor aportado por el HTF pueda ser diferente para las mismas condiciones de entrada. El calor aportado se conoce por la variación de temperatura del HTF a la salida y es el parámetro que se tiene para modificar en la subrutina de Abaqus. Se puede apreciar en las siguientes figuras 3.4.6, 3.4.7 que al aumentar la diferencia de temperaturas cambia el campo de temperaturas, concretamente aportándose más calor al que ha tenido mayor diferencia:

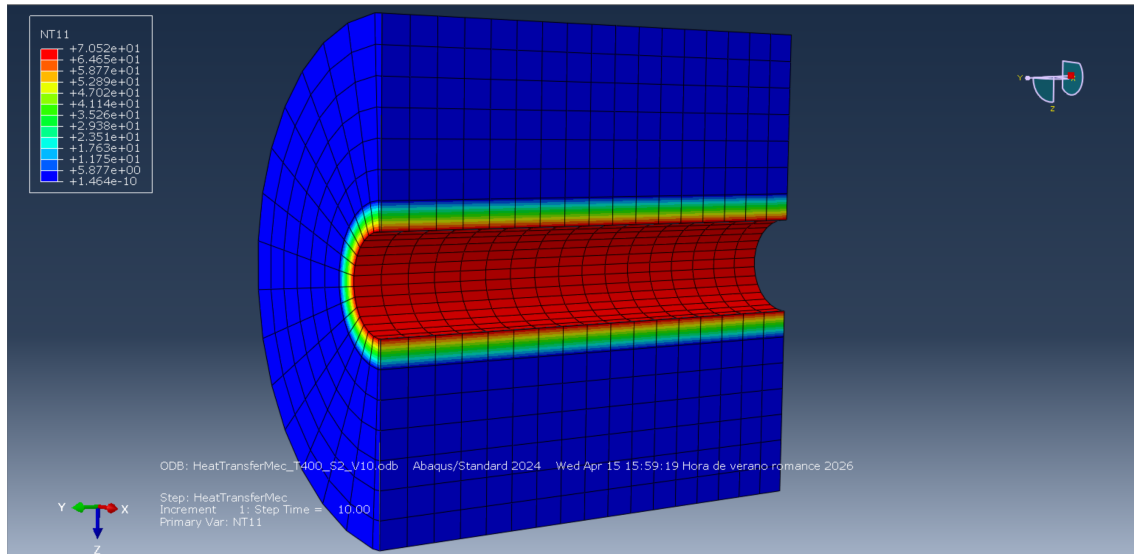


Figura 3.4.6: Gradiente de temperatura al inicio para la simulación a 400°C, velocidad HTF 2 m/s y diferencia de temperatura entre entrada y salida de 5°C

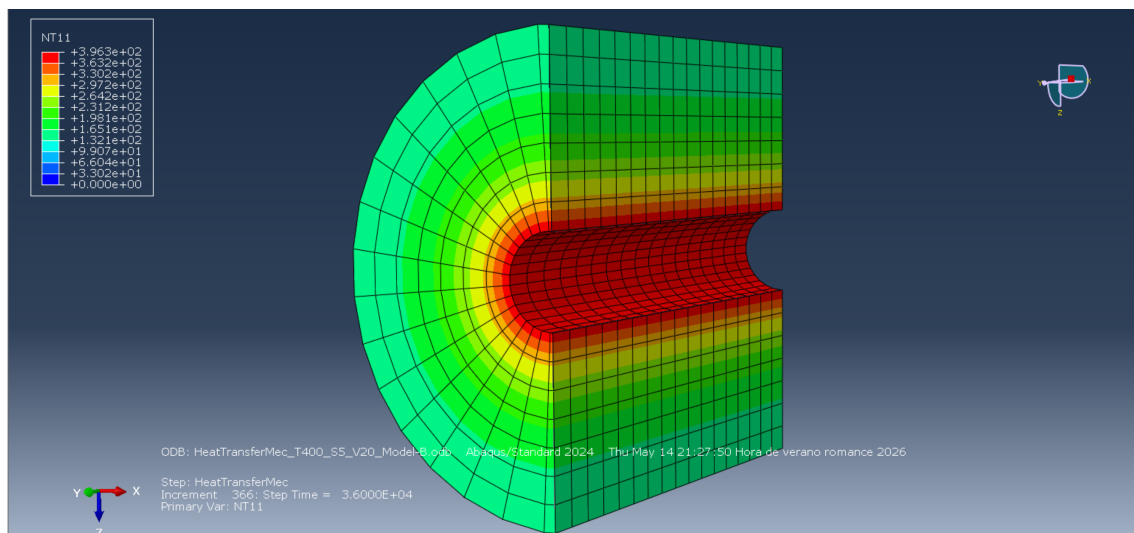


Figura 3.4.7: Gradiente de temperatura al inicio para la simulación a 400°C, velocidad HTF 2 m/s y diferencia de temperatura entre entrada y salida de 5°C

Además, es necesario comprobar la convergencia de las temperaturas. Para ello, se grafica el comportamiento de unos nodos del CTES. Para garantizar que sean representativos en las siguientes figuras se muestra la evolución de la temperatura en diversos puntos relevantes del CTES, estos son seleccionados para cumplir 3 nodos en la zona en contacto con el acero, 3 nodos en una zona intermedia y 3 nodos en la cara exterior:

Están distribuidos en el eje Z de la siguiente forma:

- Nodos 1-9-18: nodos en la superficie en contacto con el acero de la tubería.
- Nodos 37-45-54: nodos en el interior del CTES.
- Nodos 127-135-144: nodos en la superficie exterior del CTES.

Están distribuidos en el eje Y de la siguiente forma:

- Nodos 1-37-127: nodos en el inicio del CTES, zona donde entra el HTF.
- Nodos 9-45-135: nodos en el interior del CTES.
- Nodos 18-54-144: nodos en la superficie exterior del CTES.

Cumplen por tanto las siguientes condiciones:

- Nodos en el mismo plano YZ.
- Nodos solo en la mitad superior para evitar duplicidad por simetría radial.

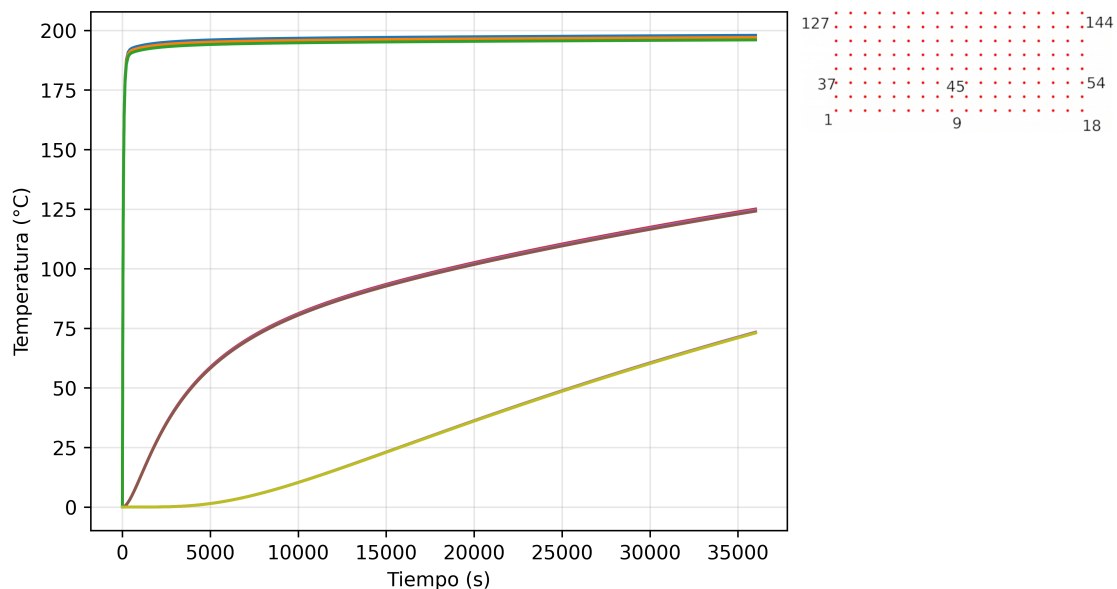


Figura 3.4.8: Temperatura de nodos en la simulación para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s

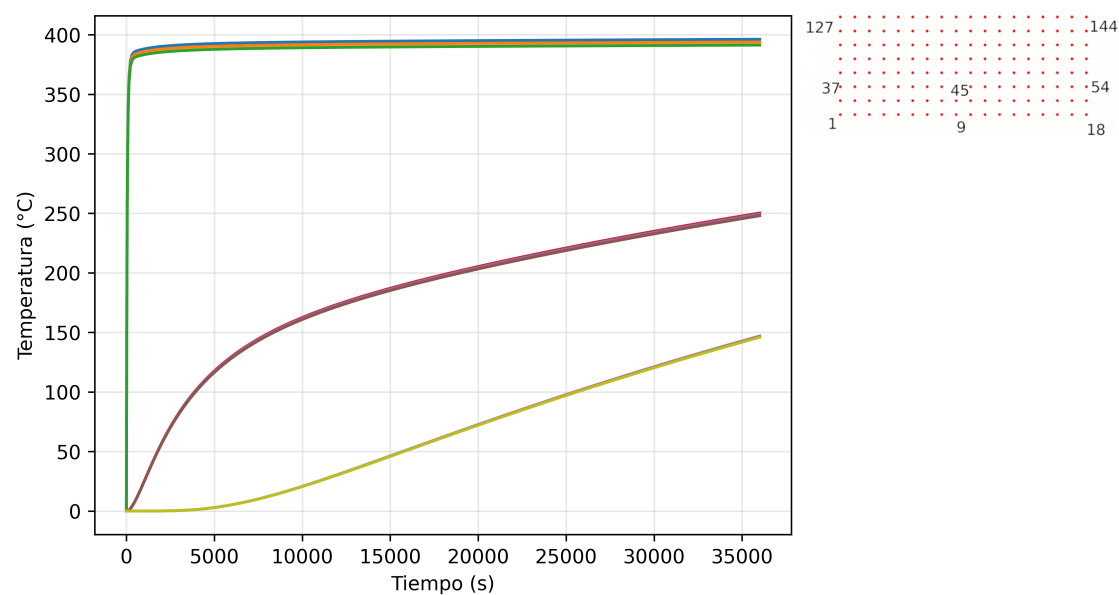


Figura 3.4.9: Temperatura de nodos en la simulación para 400°C de temperatura de HTE, diferencia de temperatura de 5°C entre entrada y salida y velocidad de HTF 2 m/s

Campo de tensiones

La temperatura es el parámetro que se quiere conocer para el aprovechamiento de la energía, sin embargo no es el único de interés. Es necesario conocer la tensión a la que está sometido el sistema a lo largo de los ciclos de carga para predecir la rotura. Al tratarse de un material frágil como el cemento, la resistencia máxima a tracción es crítica siendo del orden de un 20% de la resistencia a compresión. Para comprobar si están en el rango de funcionamiento, se extraen del modelo las tensiones máximas (correspondientes a la tracción) y mínimas (correspondientes a la compresión) del modelo.

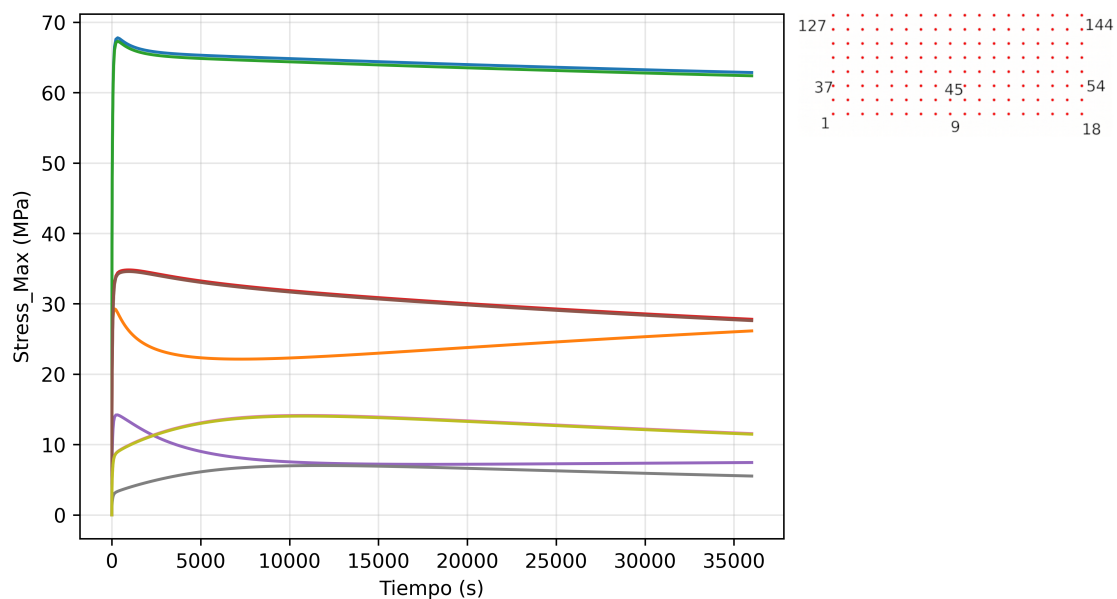


Figura 3.4.10: Comportamiento de la tensión máxima en la simulación de cemento para 200°C de temperatura de HTE, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTE 1 m/s

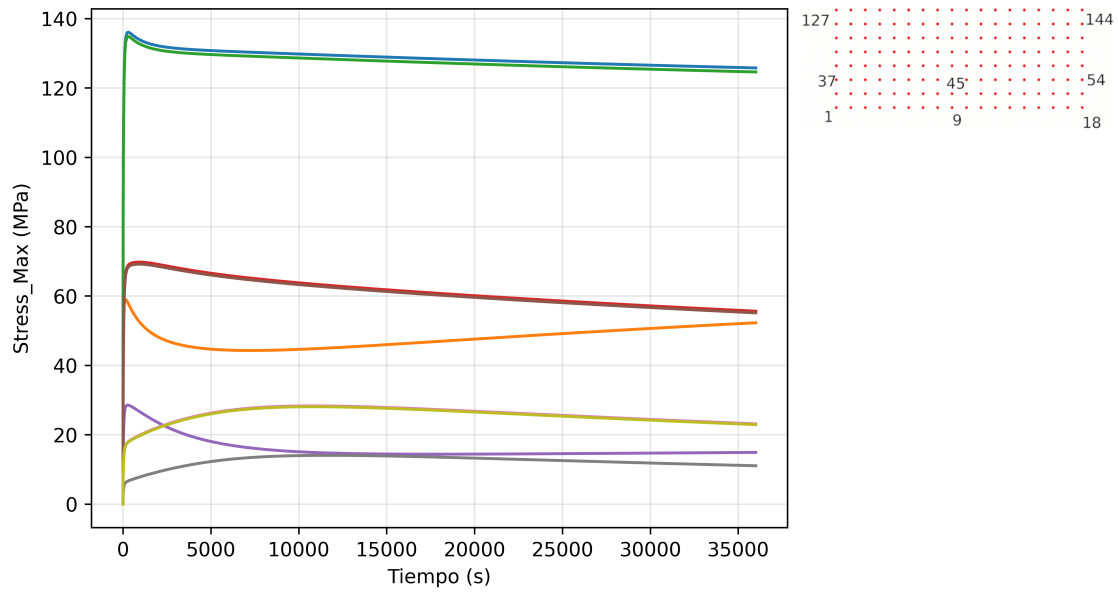


Figura 3.4.11: Comportamiento de la tensión máxima en la simulación de cemento para 400°C de temperatura de HTF, diferencia de temperatura de 5°C entre entrada y salida y velocidad de HTF 2 m/s

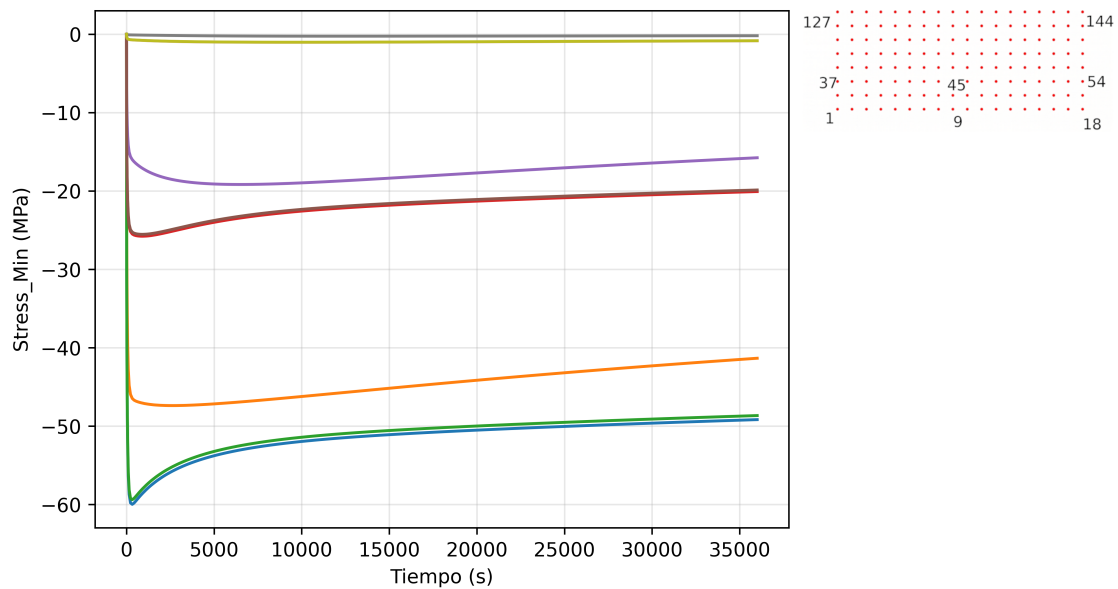


Figura 3.4.12: Comportamiento de la tensión mínima en la simulación de cemento para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s

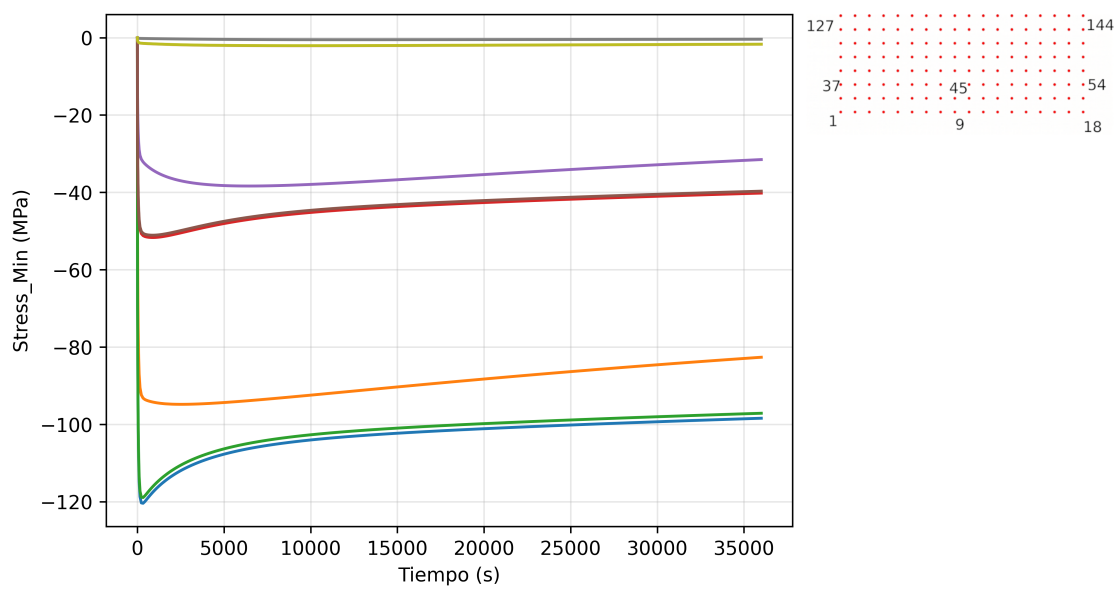


Figura 3.4.13: Comportamiento de la tensión mínima en la simulación de cemento para 400°C de temperatura de HTF, diferencia de temperatura de 5°C entre entrada y salida y velocidad de HTF 2 m/s

Simulación con aditivos

Resta añadir el nivel de detalle que se quiera alcanzar durante el estudio, se añade el efecto comentado de los aditivos que alteren la composición del material.

1. Material: Cemento.
2. Material: Cemento con fibra de basalto.
3. Material: Cemento con fibra de basalto e ilmenita.

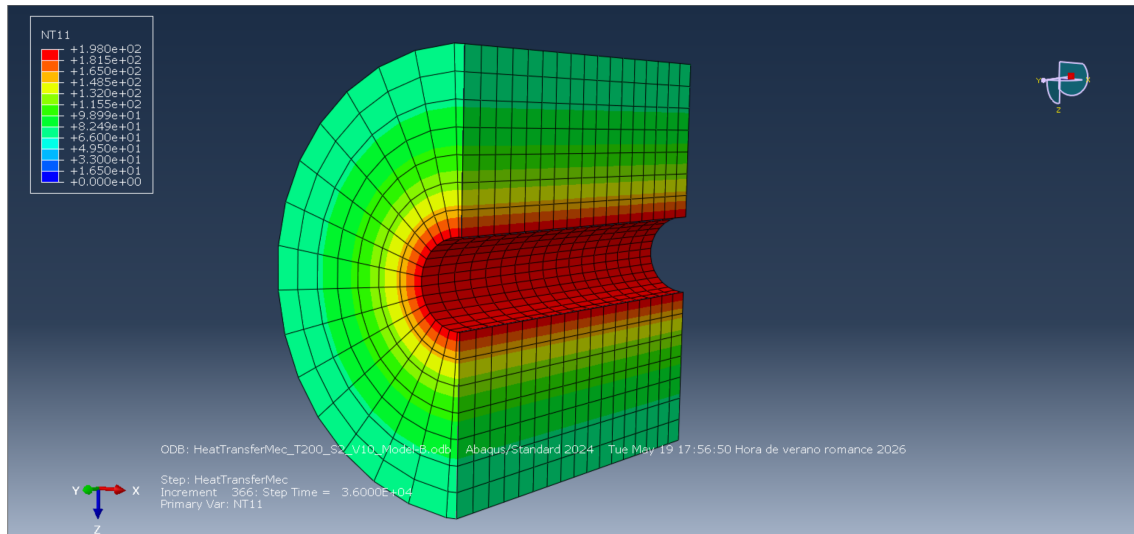


Figura 3.4.14: Campo de temperatura final en la simulación de cemento para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s

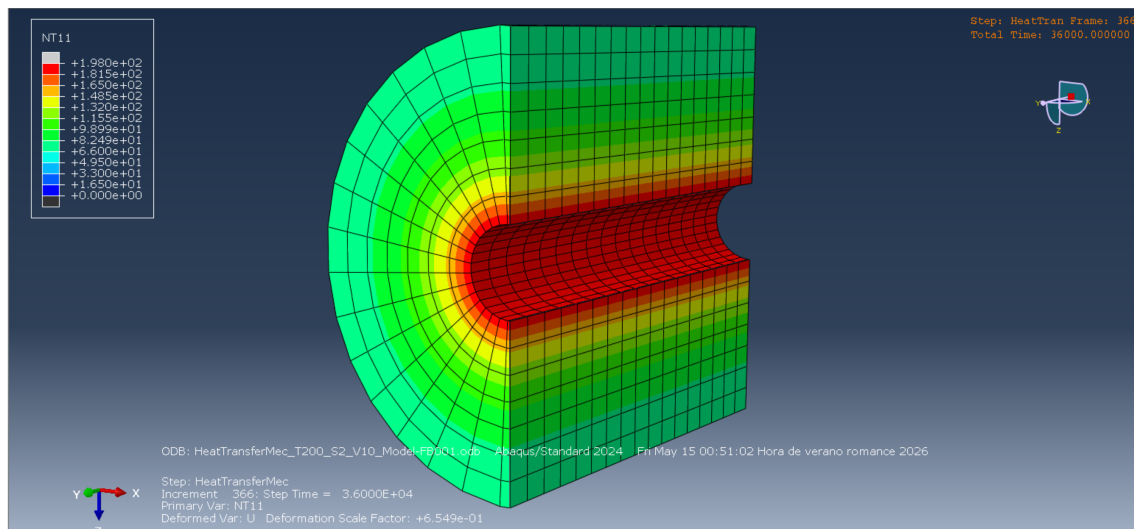


Figura 3.4.15: Campo de temperatura final en la simulación de cemento con fibras para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s

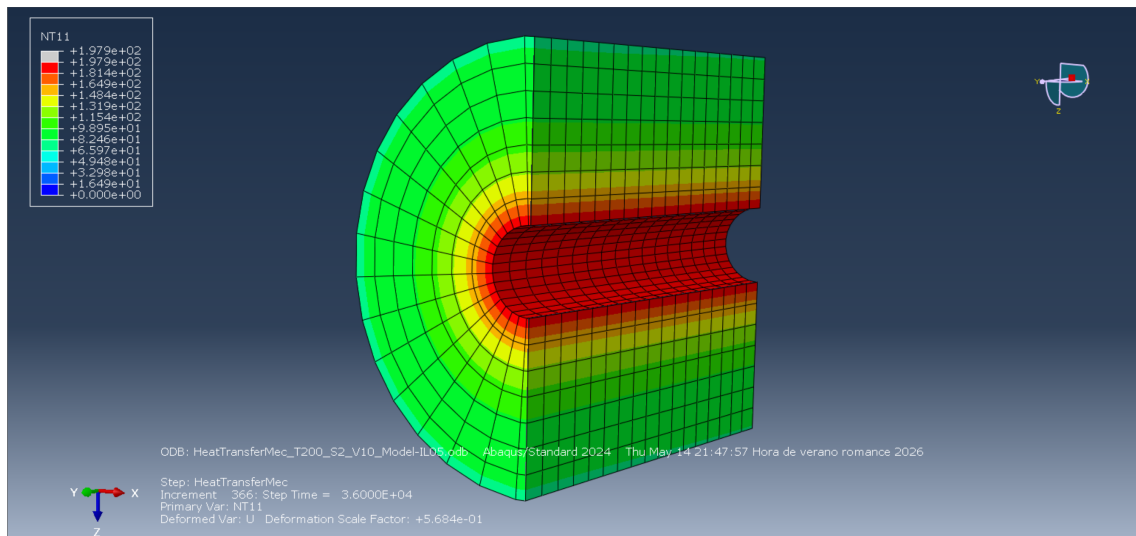


Figura 3.4.16: Campo de temperatura final en la simulación de cemento con ilmenita para 200°C de temperatura de HTE, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s

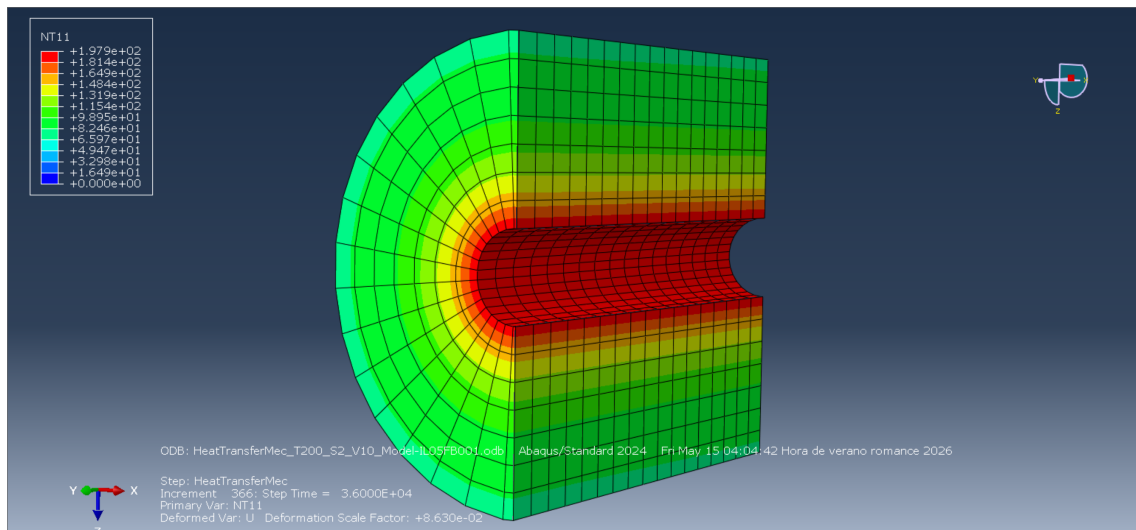


Figura 3.4.17: Campo de temperatura final en la simulación de cemento con fibras e ilmenita para 200°C de temperatura de HTE, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s

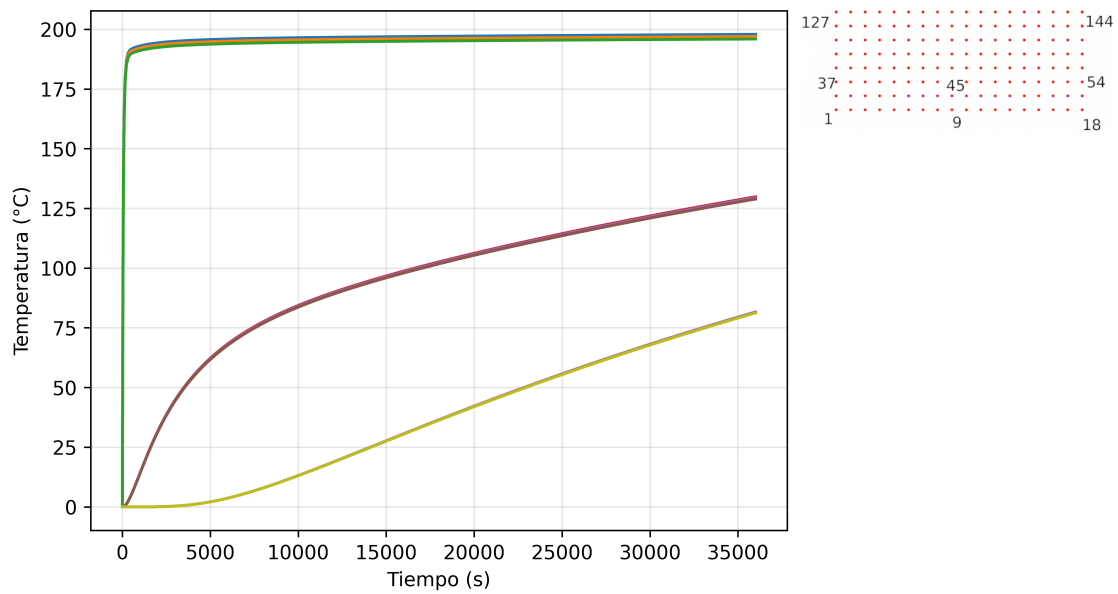


Figura 3.4.18: Temperatura de nodos en la simulación con aditivos para 200°C de temperatura de HTF, diferencia de temperatura de 2°C entre entrada y salida y velocidad de HTF 1 m/s

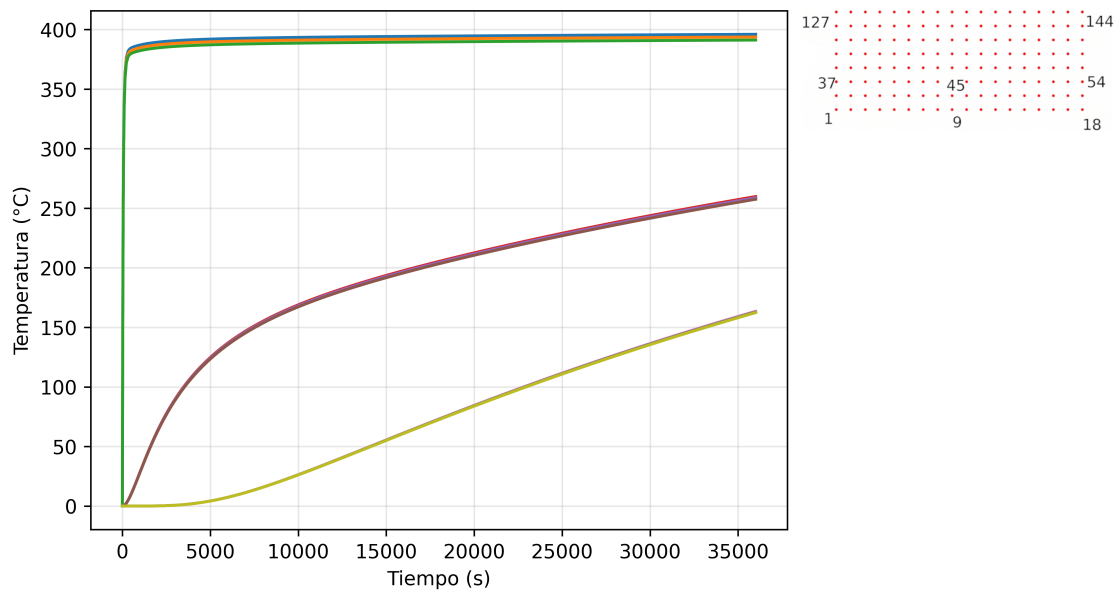


Figura 3.4.19: Temperatura de nodos en la simulación con aditivos para 400°C de temperatura de HTF, diferencia de temperatura de 5°C entre entrada y salida y velocidad de HTF 2 m/s

Comparar las Figuras 3.4.18 y 3.4.19 referidas al modelo con cemento con aditivos con las Figuras 3.4.8 y 3.4.9 referidas al modelo con cemento base, para comprobar la mejora de la capacidad del sistema para almacenar energía, es mayor la temperatura final alcanzada y mejor la pendiente de la curva de absorción en los nodos graficados.

Capítulo 4

Modelo subrogado

La solución obtenida previamente mediante elementos finitos es una solución correcta de los campos de temperatura y tensiones asociados al estado de cargas planteado, pero tiene el inconveniente de necesitar en torno a 20 minutos en hacer la resolución completa de todos los intervalos. Reduciendo el número de pasos para el intervalo o centrando el cálculo solo en un rango inferior a todo el ciclo de carga, se puede reducir el tiempo de cálculo aunque se seguiría manteniendo en el orden de minutos. El objetivo de este apartado es conocer la viabilidad de un gemelo digital que usando diversas estructuras de redes neuronales pueda reducir el tiempo de cálculo a segundos en lugar de minutos.

Las redes neuronales son una forma de calcular una predicción con funciones no lineales aplicadas a ecuaciones lineales de los datos de entrada. Esto es así para las redes *dense* donde todas las neuronas se interconectan entre sí. Los casos en los que la relación entre los datos de entrada del inicio y los del final son diferentes y casi carecen de relación resultarían en muchos valores nulos en la ecuación, por lo que se utiliza otra metodología en la que se conservan las ecuaciones, pero previamente se pasan los datos por unos filtros convolucionales para conocer la relación entre los puntos cercanos entre sí.

Este procedimiento es común en el tratamiento de imágenes y se utilizará el mismo procedimiento ya que la simetría axial del problema permite que modelando únicamente un corte del CTES se pueda conocer el estado del elemento concreto. Como los filtros convolucionales no se imponen sino que son los pesos que se aprenden durante el entrenamiento, serán los datos los que permitan que la red aprenda a distinguir los comportamientos. El corte en concreto se realiza de forma que se tienen 144 nodos en un rectángulo de 0.25,0 ; 0.125,0.125. Es el mismo set de nodos que se utilizó para graficar las temperaturas y las tensiones en 3.4.4

Por último destacar que de cada simulación realizada consta de intervalos temporales de 100 segundos hasta alcanzar los 36000 segundos de simulación y se almacenan los estados de los 144 nodos en cada uno de dichos pasos. Almacenando esos estados (posiciones, tiempo, temperatura y tensiones) en csv, se reduce a aproximadamente

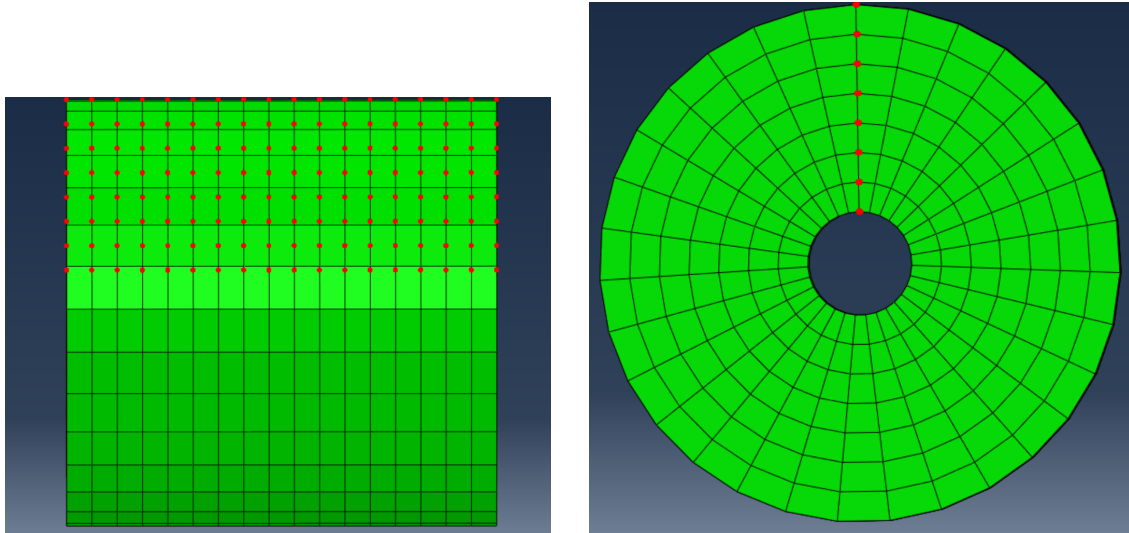


Figura 4.0.1: Alzado y perfil de la posición de los nodos utilizados para el estudio

6.5 MB cada simulación, a modo de comparación el .odb generado por Abaqus ocupa alrededor de 700 MB.

4.1. REDES NEURONALES

Para la comprensión del funcionamiento de las redes neuronales, lo primero es conocer el funcionamiento de los perceptrones (su unidad mínima), las *shallow neural networks* y finalmente las redes neuronales tal y como se utilizan actualmente:

Las *shallow neural networks* son funciones $y = f[x, \phi]$ con parámetros ϕ que mapean entradas multivariantes x a salidas multivariantes y . La idea principal puede expresarse con una red de ejemplo [10], $f[x, \phi]$ que mapea una entrada escalar x a una salida escalar y , y que cuenta con diez parámetros: $\phi = \{\phi_0, \phi_1, \phi_2, \phi_3, \theta_{10}, \theta_{11}, \theta_{20}, \theta_{21}, \theta_{30}, \theta_{31}\}$:

$$\begin{aligned} y = f[x, \phi] = \\ = \phi_0 + \phi_1 a[\theta_{10} + \theta_{11}x] + \phi_2 a[\theta_{20} + \theta_{21}x] + \phi_3 a[\theta_{30} + \theta_{31}x]. \end{aligned} \quad (4.1.1)$$

Podemos desglosar este cálculo en tres partes: primero, calculamos tres funciones lineales de los datos de entrada ($\theta_{10} + \theta_{11}x$, $\theta_{20} + \theta_{21}x$ y $\theta_{30} + \theta_{31}x$). Segundo, pasamos los tres resultados a través de una función de activación $a[\cdot]$. Finalmente, ponderamos las tres activaciones resultantes con ϕ_1 , ϕ_2 y ϕ_3 , las sumamos y añadimos un sesgo o desplazamiento (*offset*) ϕ_0 .

Para completar la descripción, debemos definir la función de activación $a[\cdot]$. Existen muchas posibilidades, pero la elección más común es la unidad lineal rectificada o ReLU (*Rectified Linear Unit*):

$$a[z] = \text{ReLU}[z] = \begin{cases} 0 & z < 0 \\ z & z \geq 0 \end{cases} \quad (4.1.2)$$

4.1.1. Redes convolucionales

La importancia de la proximidad en los problemas termomecánicos hace que sea necesario el uso de redes convolucionales, las cuales son habituales en el tratamiento de imágenes donde píxeles cercanos tienen relación estadística. Las redes vistas anteriormente tratan todas las relaciones entre neuronas de la misma manera por lo que no tienen en consideración la importancia de la cercanía. Además, la interpretación de la red debería ser la misma si se cambiase el sentido de flujo del HTF (lo que sería hacer una imagen especular) pero si se entrenasen las redes de forma habitual tendría que entrenarse con ambos casos puesto que la relación entre píxeles sería indiferente. Esta estructura mantiene la dependencia espacial de forma que los cambios por variaciones de temperatura o flujo serán más extrapolables.

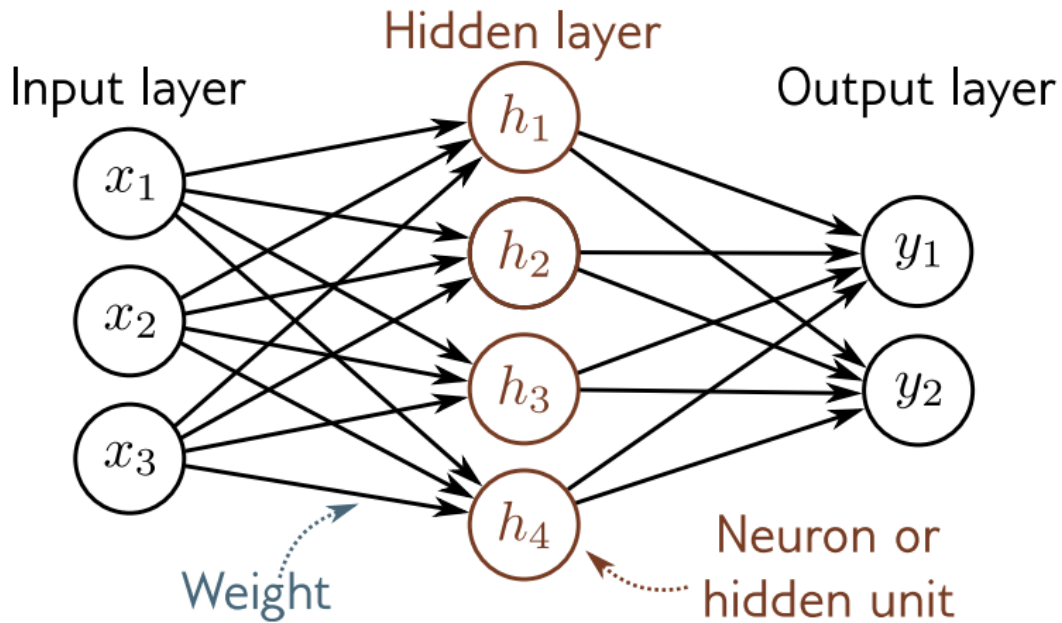


Figura 4.1.1: Visualización de una *shallow neural network*, extraído de [10]

Capas convolucionales

Las capas convolucionales son capas de red basadas en la operación de convolución. En una dimensión (1D), una convolución transforma un vector de entrada x en un vector de salida z , de modo que cada salida z_i es una suma ponderada de las entradas cercanas. Se utilizan los mismos pesos en cada posición y estos se denominan colectivamente núcleo de convolución (kernel) o filtro. El tamaño de la región sobre la cual se combinan las entradas se denomina tamaño del kernel. Para un tamaño de kernel de tres, tenemos:

$$z_i = \omega_1 x_{i-1} + \omega_2 x_i + \omega_3 x_{i+1} \quad (4.1.3)$$

donde $\omega = [\omega_1, \omega_2, \omega_3]$ es el kernel. Nótese que la operación de convolución es equivariante con respecto a la traslación. Si desplazamos la entrada x , la salida correspondiente z se desplazará de la misma manera.

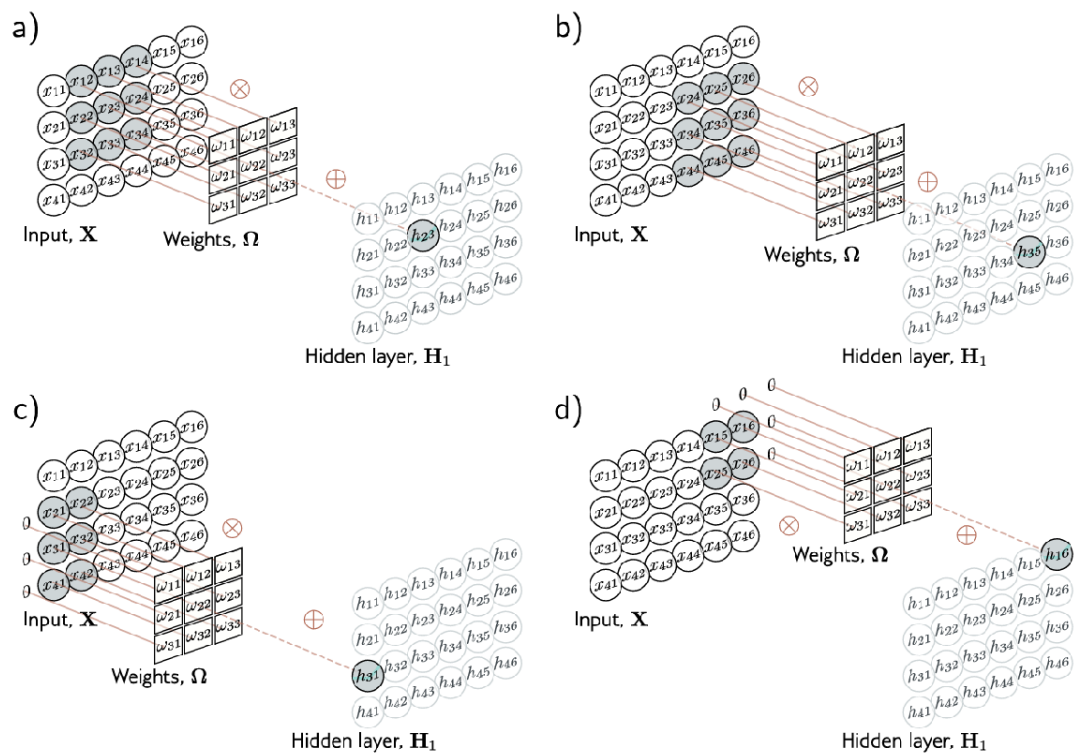


Figura 4.1.2: Visualización de una *convolutional neural network* haciendo un barrido en unos datos 2D con un filtro 3x3, extraído de [10]

4.1.2. Redes recurrentes

En esta estructura de redes son habituales en el tratamiento de series temporales donde datos secuenciales cercanos tienen relación estadística. Por ello permiten captar la importancia de la evolución temporal en los problemas termomecánicos. Las redes vistas anteriormente tratan todas las relaciones entre entradas de forma independiente por lo que no tienen en consideración la importancia de la secuencia. Con este procedimiento se mantiene la dependencia temporal de forma que los cambios por variaciones de temperatura o flujo a lo largo del tiempo serán más extrapolables.

Capas recurrentes

Las capas recurrentes son capas de red basadas en la operación de recurrencia mediante la actualización de un estado oculto. En una dimensión temporal, una recurrencia transforma un vector de entrada actual x_t y el estado oculto anterior h_{t-1} en un nuevo estado oculto h_t , de modo que cada salida es una combinación ponderada de la entrada actual y la memoria del pasado. Se utilizan los mismos pesos en cada paso de tiempo y estos se denominan colectivamente matrices de pesos del kernel recurrente. Para un paso de tiempo actual, tenemos:

$$h_t = \tanh(Wx_t + Uh_{t-1} + b) \quad (4.1.4)$$

donde W y U son las matrices de pesos y b es el sesgo. Nótese que la operación recurrente es invariante con respecto al tiempo. Si desplazamos la secuencia de entrada en el tiempo, la misma transición de estados se aplicará de la misma manera.

4.1.3. Aplicación a problemas de elementos finitos

Las redes convolucionales se han utilizado con suficiente precisión para la simulación de tensiones a partir de la geometría y las condiciones de contorno [11] [13]. Sin embargo, esto limita las simulaciones espaciales a formas rectangulares 2D [12] para poder predecir geometrías más complejas se hace uso de redes neuronales de grafos. El caso de estudio que se está tratando puede ser reducido a un modelo bidimensional considerando para la simulación únicamente un corte longitudinal del cilindro ya que presenta una simetría axial.

4.2. PREPARACIÓN DE DATOS

Las redes neuronales son sensibles a los datos utilizados para el entrenamiento, es decir, los datos introducidos originalmente de los que extrae patrones. Para la obtención de las redes se ha utilizado las siguientes variaciones en los datos de entrada:

1. Temperatura: Variación de 200°C a 400°C en saltos de 10°C.
2. Aporte de calor: caída de temperatura del HTF de 2°C o 5°C.
3. Velocidad del HTF: 1 m/s o 2 m/s.
4. Aditivos: base, con fibras de basalto al 0.1 %, con ilmenita 5 %, con ilmenita 5 % y fibra de basalto 0.1 %.

4.3. RED NEURONAL UTILIZADA

Los parámetros habitualmente fijos de una red son la profundidad de las capas densas (número de activaciones y pesos aprendibles) y el tamaño y número de filtros convolucionales, posteriormente se entrena varios hiperparámetros tales como el ratio de aprendizaje (*learning rate*), el número de épocas (*epochs*) y el tamaño de lote (*batch size*). Para optimizar al máximo la red obtenida, se hace uso de una librería de optimización de hiperparámetros, a la que además se añade que pueda entrenar con diversos tamaños de las capas de convolución, desde [8,16] a [32,64], ver Anexos con el código utilizado. Por parte de las capas densas, se podrían añadir a ese sobre entrenamiento, pero se decide no modificarlas en el entrenamiento porque estas lo único que hacen es incorporar a las características extraídas con la convolución la información que se tiene de los aditivos utilizados.

Para el entrenamiento de los modelos se usa el error medio cuadrático (*Mean Squared Error*) con respecto al paso calculado. Para comprobar la efectividad del sistema de forma similar a lo que haría en producción en un entorno real en el que se desee conocer el estado del sistema con antelación o hacer modificaciones en la composición y temperatura de circulación del HTF para optimizar, se simula de forma autoregresiva toda la simulación con la que se ha ensayado el sistema, las 10 horas de posible recepción de calor.

Con esa estructura, se plantean varios tipos de modelo subrogado:

4.3.1. Modelo completo frente a modelo específico

Inicialmente se considera la posibilidad de entrenar un modelo que pueda predecir conjuntamente la temperatura y las tensiones al estar estas relacionadas espacial y temporalmente. Sin embargo, los experimentos muestran que si bien existe la relación física, no es sencillo encontrar dicho patrón con este tipo de modelo.

Los modelos que solo entrenan una salida son más costosos de entrenar, especialmente considerando el entrenamiento de hiperparámetros que debe hacerse por cada red de forma independiente, pero consiguen resultados de mejor calidad.

Es posible observar en la Figura 4.3.1 que se llega a captar la forma de las cargas máxima y mínima con los máximos en los extremos y puntos cercanos a la tubería, pero no así su valor, mientras que en las simulaciones de la temperatura no aproxima correctamente. La simulación autorregresiva solo para la temperatura 4.3.2 es mucho más fidedigna a los datos que se usan de referencia.

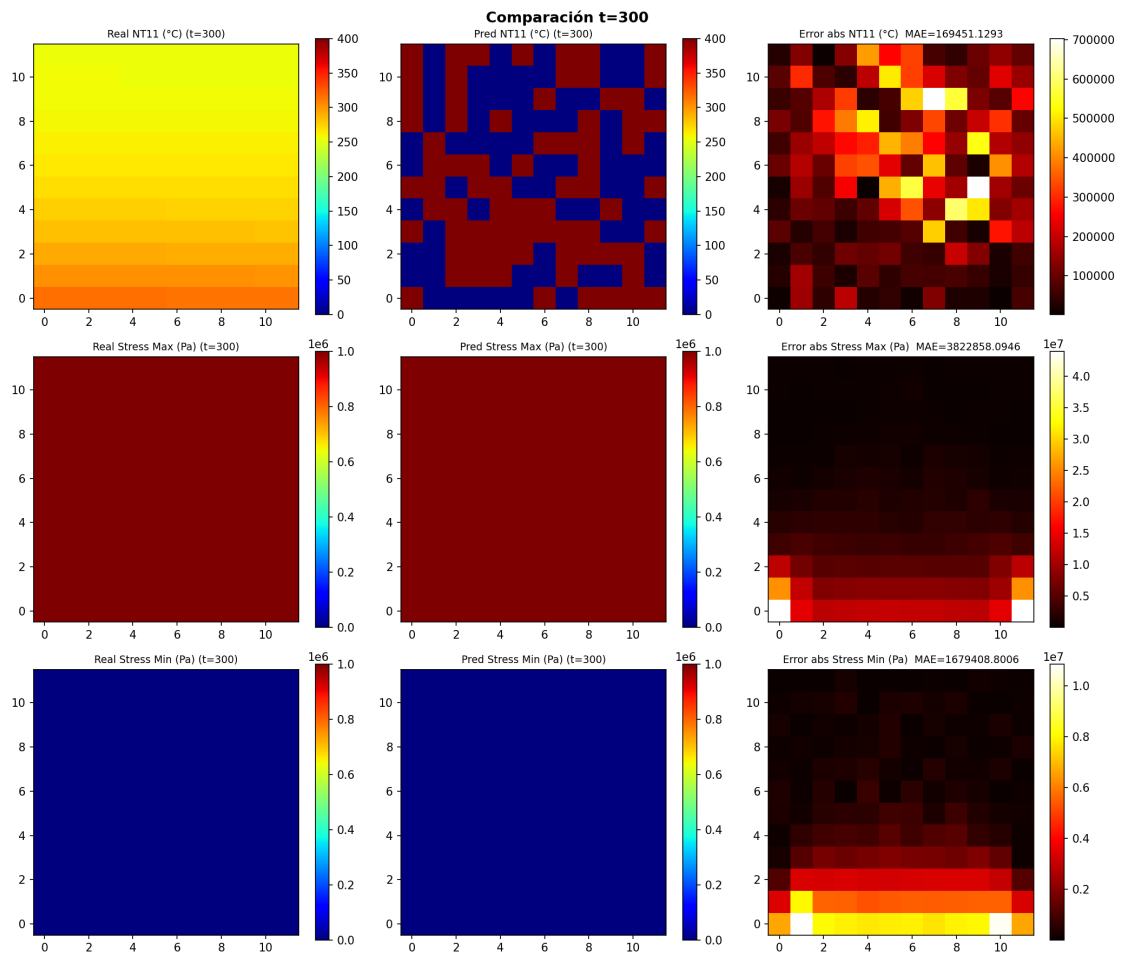


Figura 4.3.1: Simulación hasta el paso 300 con redes recurrentes

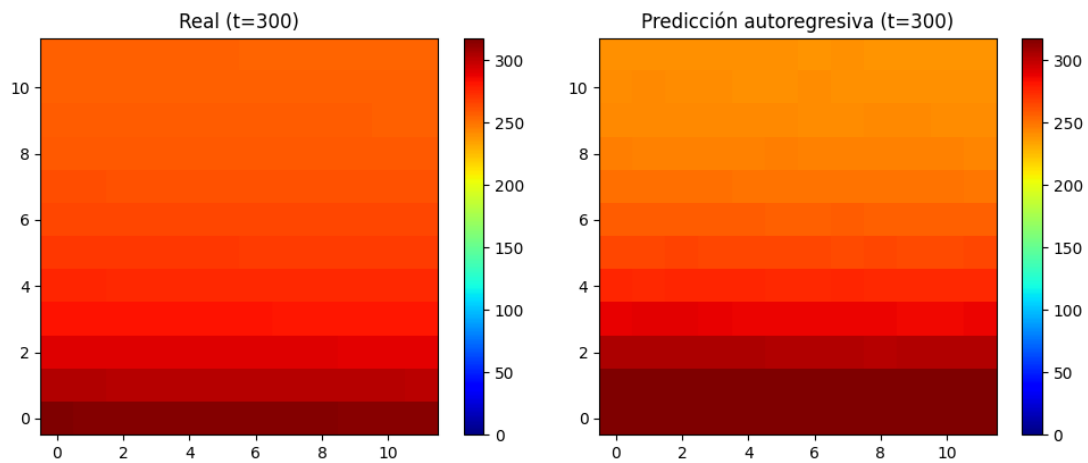


Figura 4.3.2: Simulación únicamente de temperatura en mismas condiciones hasta el paso 300 con redes recurrentes

4.3.2. Modelo convolucional frente a modelo convolucional y recurrente

Se plantea la misma estructura, pero a los modelos recurrentes se entrenan observando un paso y los diez anteriores para extrapolar la tendencia temporal y no solo la espacial. Este procedimiento da resultados más rápidos y mejores en el procesamiento de temperatura principalmente, en modelos de tensión hay una gran oscilación, ver Anexos para los MSE de cada paso recurrente.

Modelo	MSE	MAE	R2
CNN Temperatura	0.042543	0.066572	1.0000
RNN Temperatura	1.184578	0.555093	0.9999
CNN Tensión máxima	173772029952	81988.883	0.9997
RNN Tensión máxima	2040934825984	462787.687	0.9963
CNN Tensión mínima	90687832064	34570.211	0.9998
RNN Tensión mínima	1798563430400	443343.8123	0.9954

Tabla 4.3.1: Resultados de modelos

Estos son los errores medios obtenidos en los test con el paso siguiente en el caso de CNN y de los 10 pasos en el caso de RNN (por eso algo mayor en valor). Hay que tener en consideración que un fallo del orden de 10000 Pa en las tensiones que se espera que estén en el orden de MPa se considera aceptable.

4.4. RESULTADOS

Para comprobar si el modelo extrapola correctamente, se han realizado simulaciones en el rango de trabajo, pero con variables que no son las que se han utilizado para el estudio. Concretamente, se utilizan velocidad de 1.5 m/s del HTE, caída de temperatura de 4°C y temperatura de entrada 398°C, este último es valor habitual de funcionamiento de estos equipos.

4.4.1. Predicción autorregresión

La predicción se realiza de forma que se obtenga el valor de las características en el intervalo completo de carga comenzando por el estado inicial a temperatura 0°C. Se obtienen los valores que se muestran junto con los datos obtenidos por el sistema FEM y se calculan también los errores relativos para hacer comparaciones de precisión, ya que el valor absoluto del error depende de las unidades ($\approx$ MPa en tensiones hace que sean muy elevadas respecto a temperatura).

Comparaciones

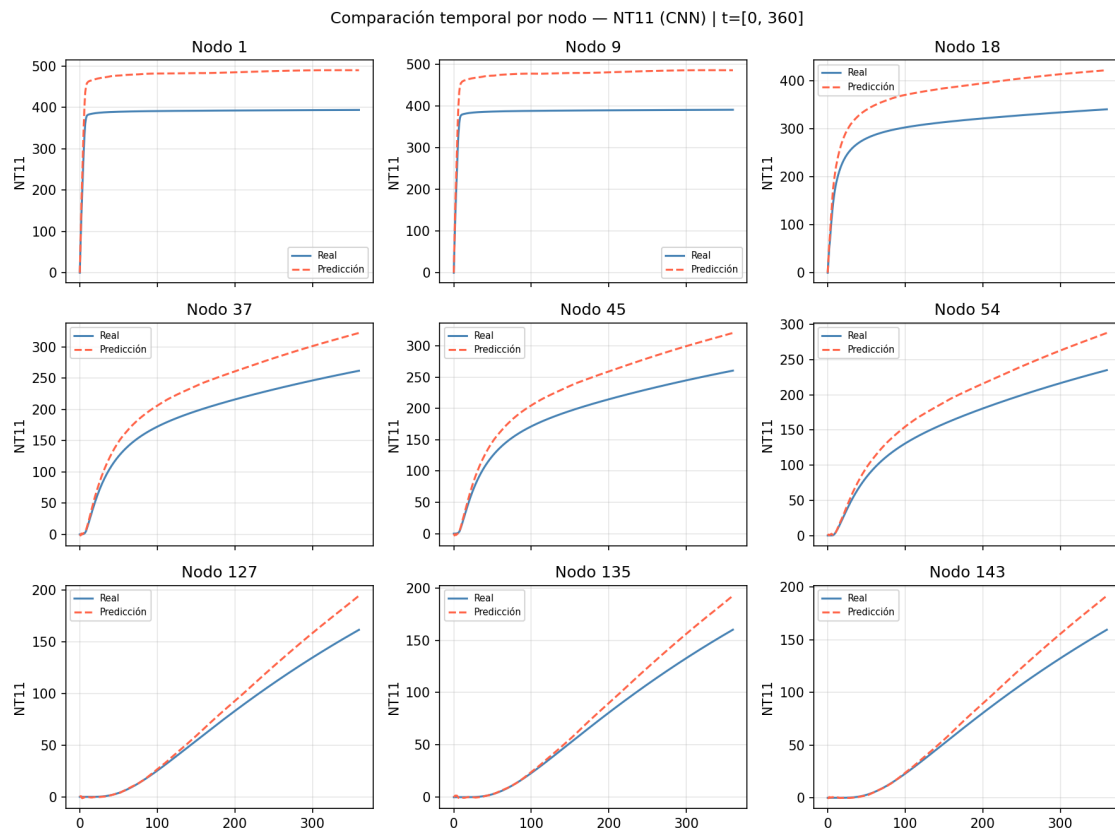


Figura 4.4.1: Comparación de temperatura calculada con RNN en los nodos centrales del CTES

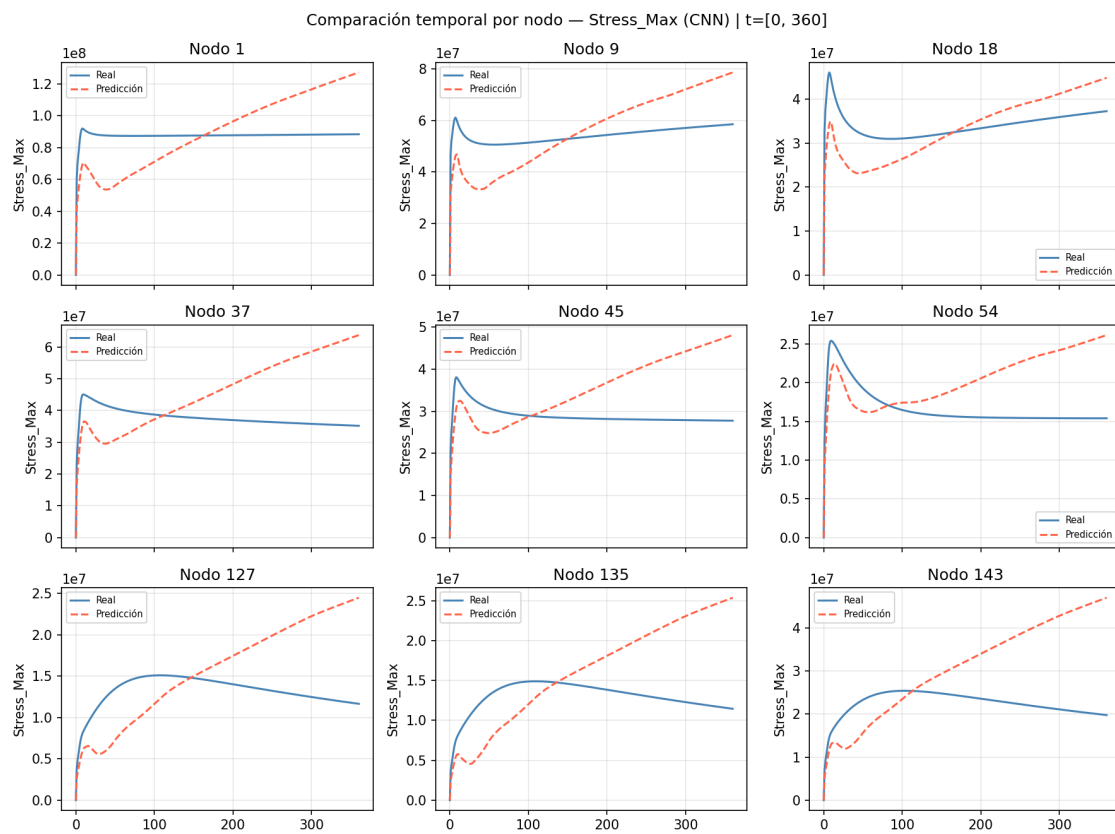


Figura 4.4.2: Comparación de tensión máxima calculada con CNN en los nodos centrales del CTES

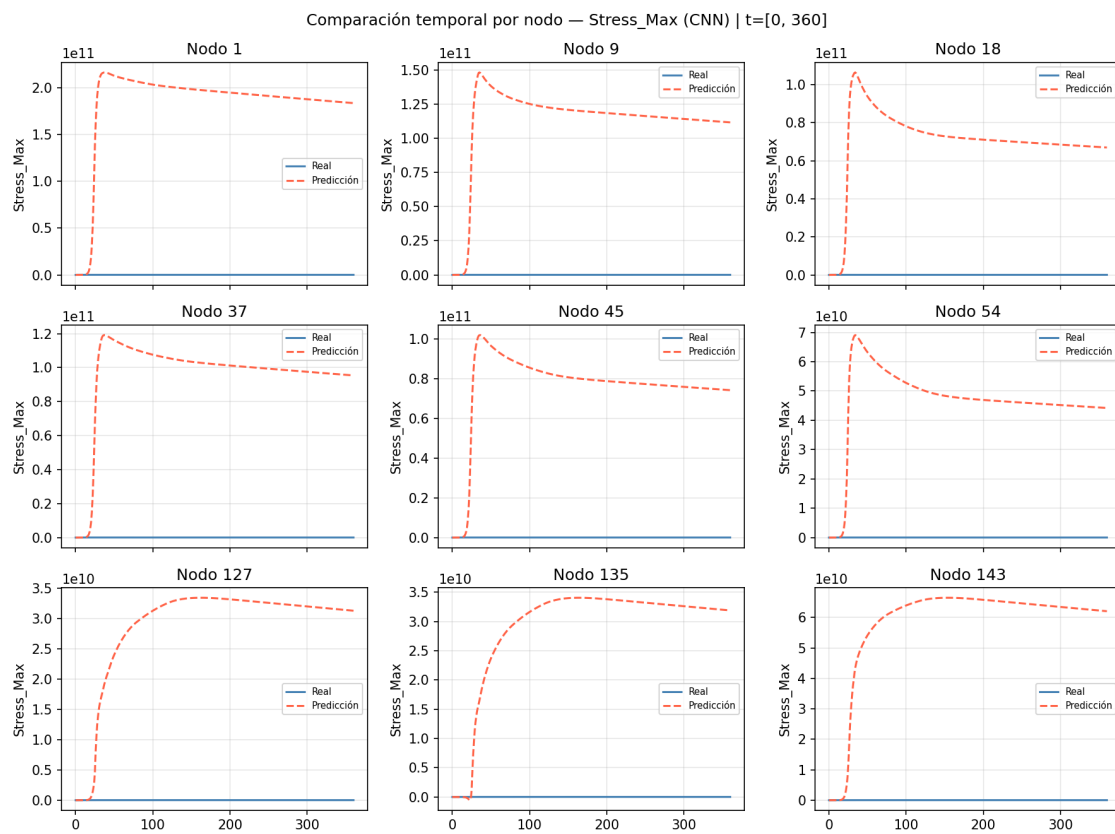


Figura 4.4.3: Comparación de tensión máxima calculada con RNN en los nodos centrales del CTES

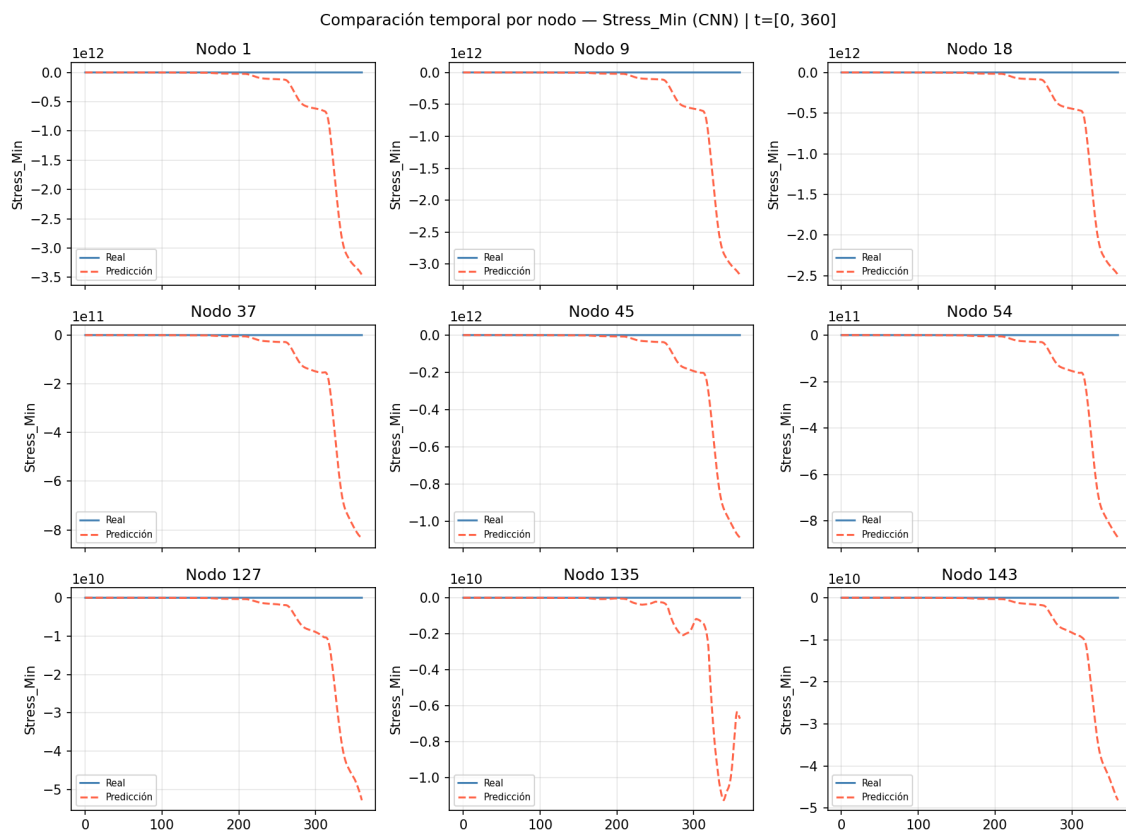


Figura 4.4.4: Comparación de tensión mínima calculada con CNN en los nodos centrales del CTES

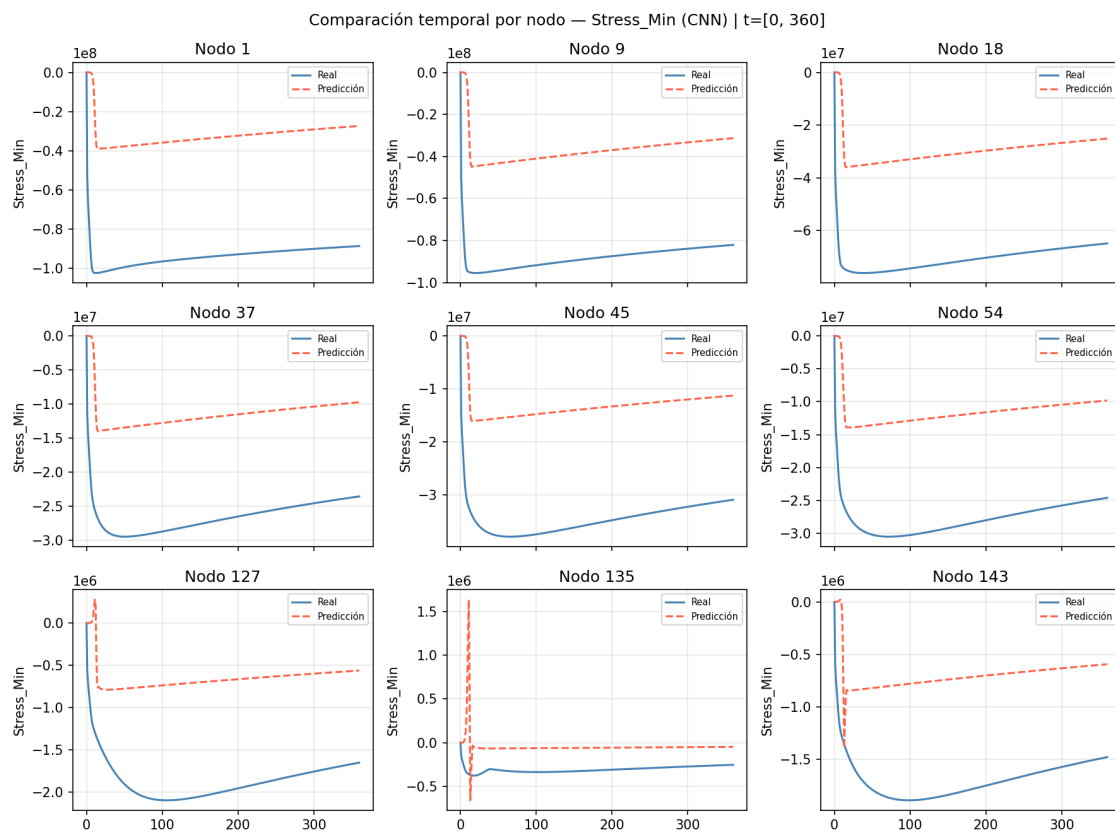


Figura 4.4.5: Comparación de tensión mínima calculada con RNN en los nodos centrales del CTES

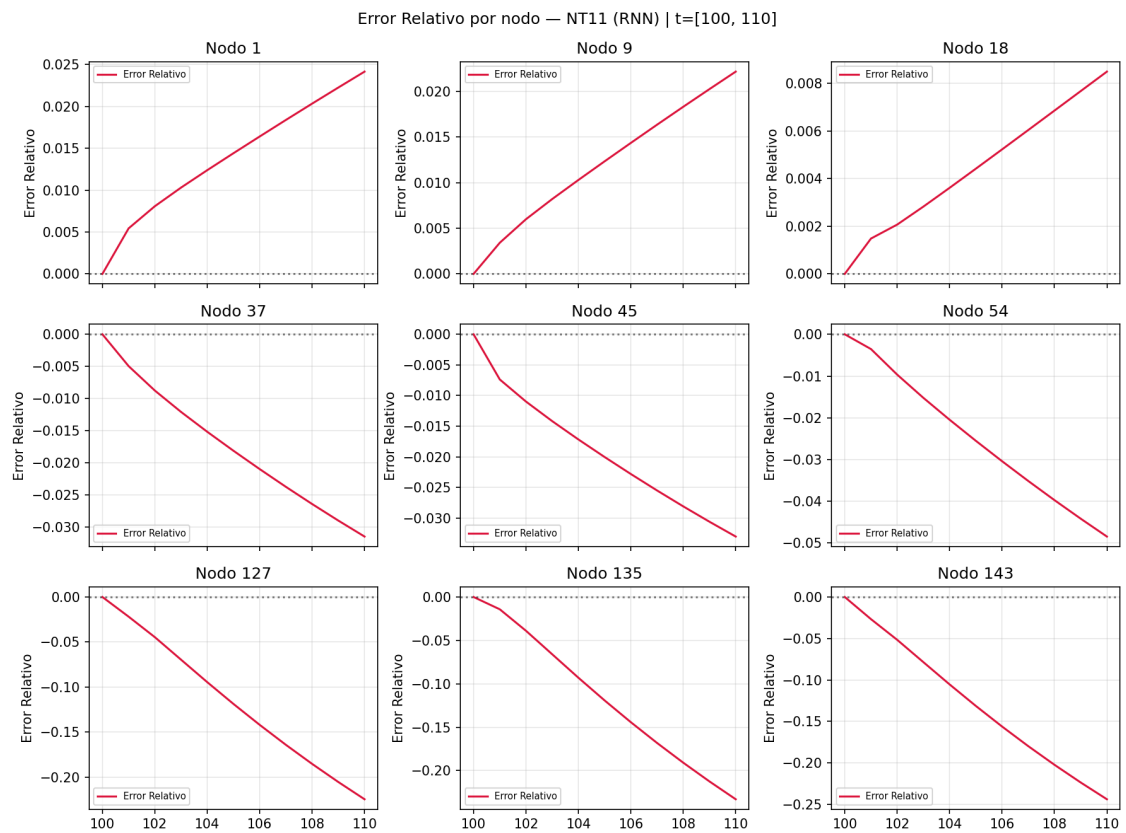
Errores

Figura 4.4.6: Errores relativos de temperatura calculada con RNN en los nodos centrales del CTES

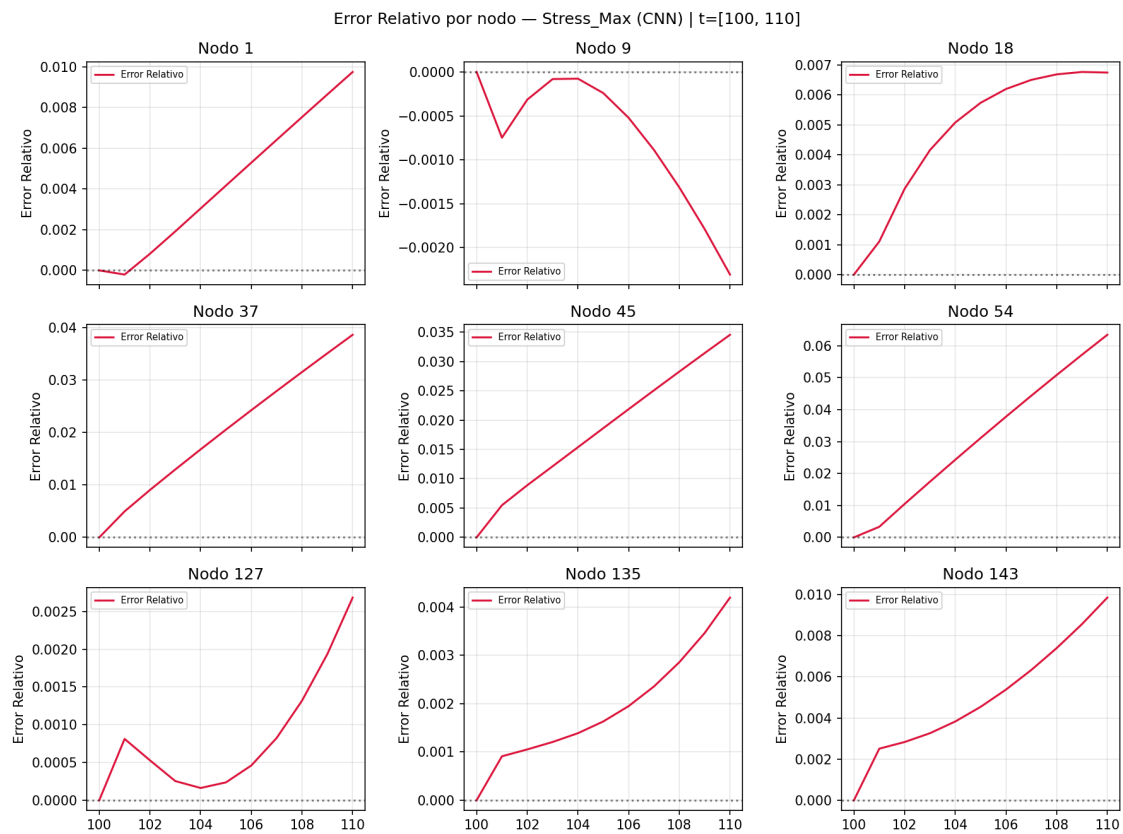


Figura 4.4.7: Errores relativos de tensión máxima calculada con CNN en los nodos centrales del CTES

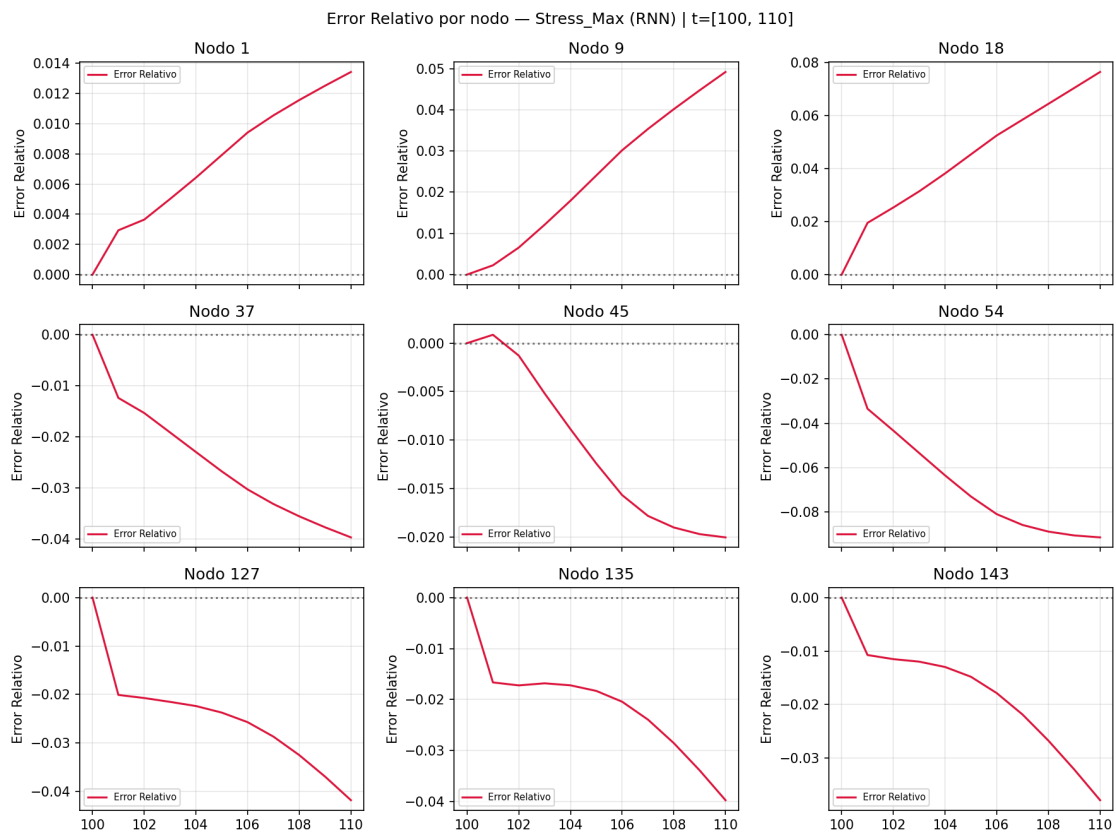


Figura 4.4.8: Errores relativos de tensión máxima calculada con RNN en los nodos centrales del CTES

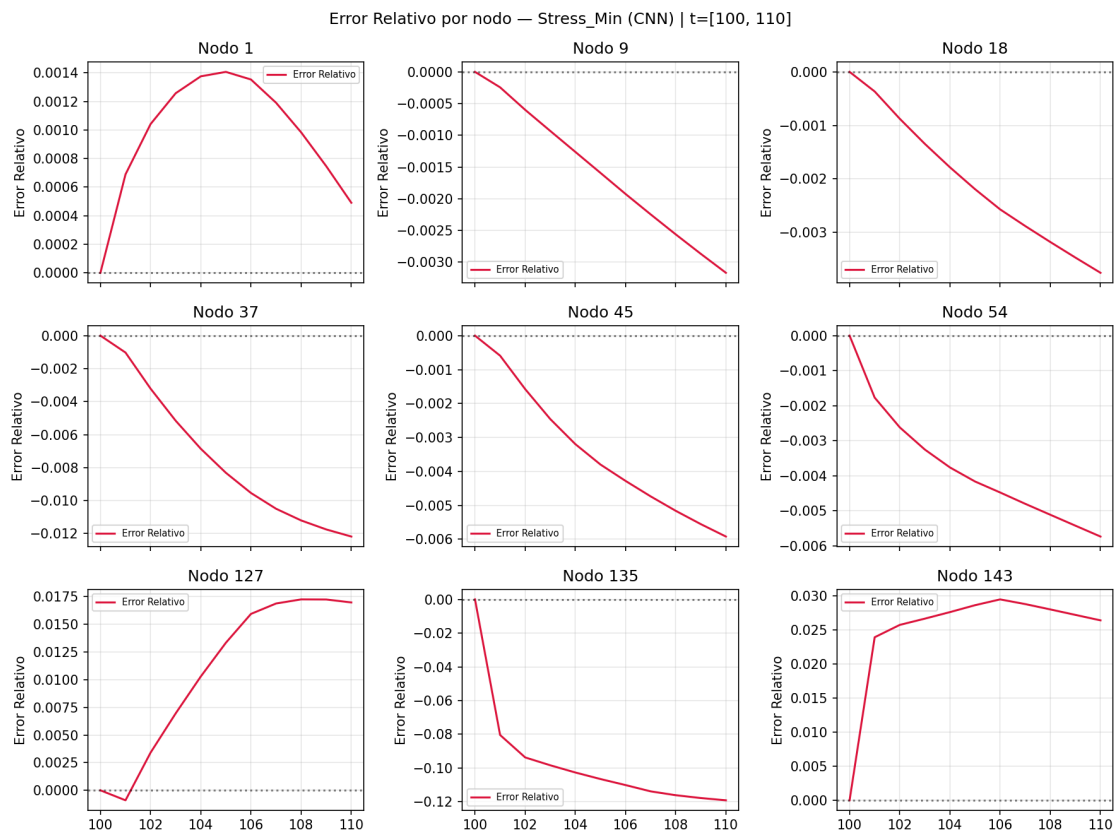


Figura 4.4.9: Errores relativos de tensión mínima calculada con CNN en los nodos centrales del CTES

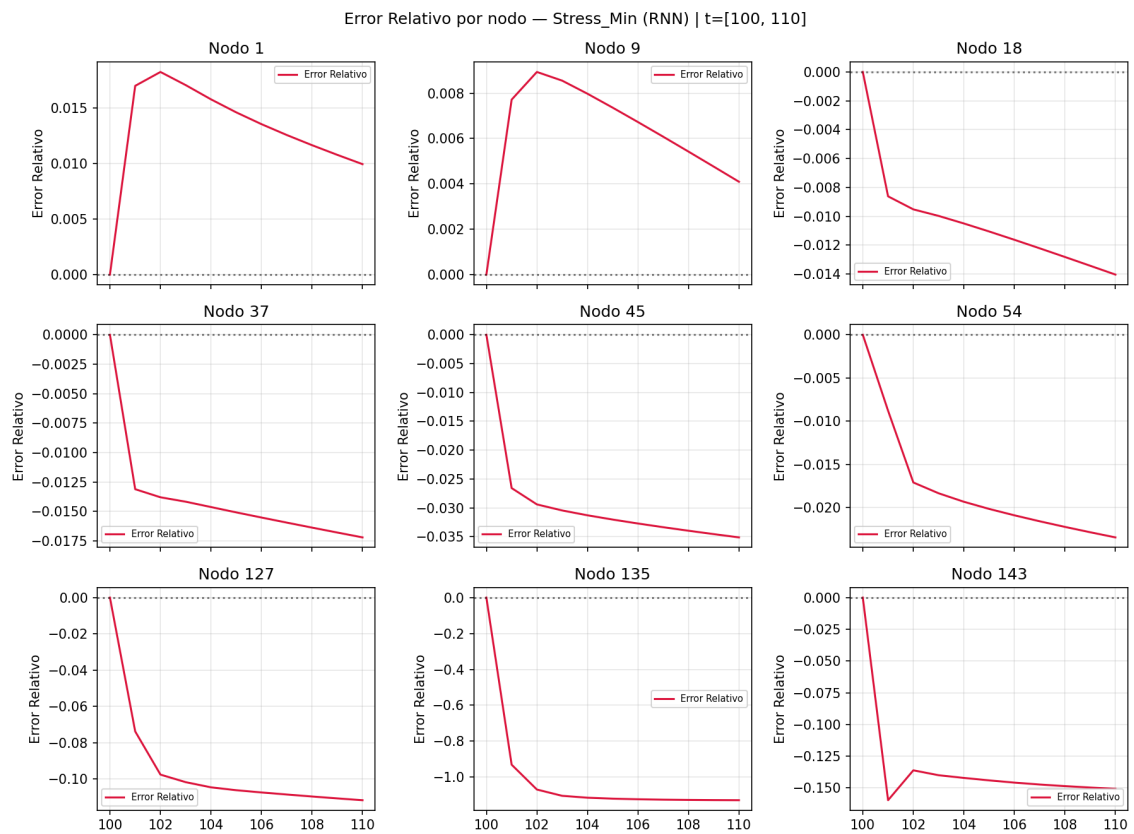


Figura 4.4.10: Errores relativos de tensión mínima calculada con RNN en los nodos centrales del CTES

Tiempo de inferencia

```

1      === TIEMPO DE PROCESADO DE LA EVALUACIÓN ===
2      Variable: NT11
3      Rango de pasos: t=[0, 360]
4      Modelo: /home/mdlsg/Documents/TFM/Data/Models/
OPTUNA_modelo_termico_E30_B128_LR0.00068_VNT11.pth
5      Tiempo total de inferencia de la red: 0.2422
segundos
6      Tiempo medio por frame: 0.000673 segundos/frame
7
8      === TIEMPO DE PROCESADO DE LA EVALUACIÓN ===
9      Variable: Stress_Max
10     Rango de pasos: t=[0, 360]
11     Modelo: /home/mdlsg/Documents/TFM/Data/Models/
OPTUNA_modelo_termico_E24_B64_LR0.00077_VStress_Max.pth
12     Tiempo total de inferencia de la red: 0.2151
segundos
13     Tiempo medio por frame: 0.000598 segundos/frame
14
15     === TIEMPO DE PROCESADO DE LA EVALUACIÓN ===
16     Variable: Stress_Min
17     Rango de pasos: t=[0, 360]
18     Modelo: /home/mdlsg/Documents/TFM/Data/Models/
OPTUNA_modelo_termico_E22_B32_LR0.00080_VStress_Min.pth
19     Tiempo total de inferencia de la red: 0.2223
segundos
20     Tiempo medio por frame: 0.000618 segundos/frame
21
22

```

Discusión

En el caso de la temperatura, se excede la temperatura del HTF por la acumulación de errores en ambos modelos lo cual es incoherente.

En el caso de las tensiones, hay comportamientos dispares, CNN aproxima razonablemente bien la tensión máxima, pero aumenta mucho en la tensión mínima en una zona en la que simplemente debería mantenerse. Aunque capte el comportamiento en la tensión máxima, su valor es inferior por lo que para predecir roturas o agrietamiento es insuficiente. Por otra parte, con RNN se capta la tendencia con la tensión mínima, mientras que la tensión máxima es varios órdenes de magnitud superior a lo que debería. En este caso, se produce un retraso entre el aumento de la variable predicha y la variable real, luego tampoco es útil para la predicción.

La inferencia de los resultados sí que cumple con lo previsto, obteniendo la simulación completa en períodos inferiores a un segundo.

A la vista de los resultados, no es viable tratar de predecir el comportamiento durante períodos tan largos.

4.4.2. Predicciones intervalos

Por otra parte, si las predicciones se restringen a un intervalo, se pierde captar la tendencia en CNN (puesto que no tiene conocimiento de lo que sucede anteriormente), pero se mantienen en un rango razonable.

Comparaciones

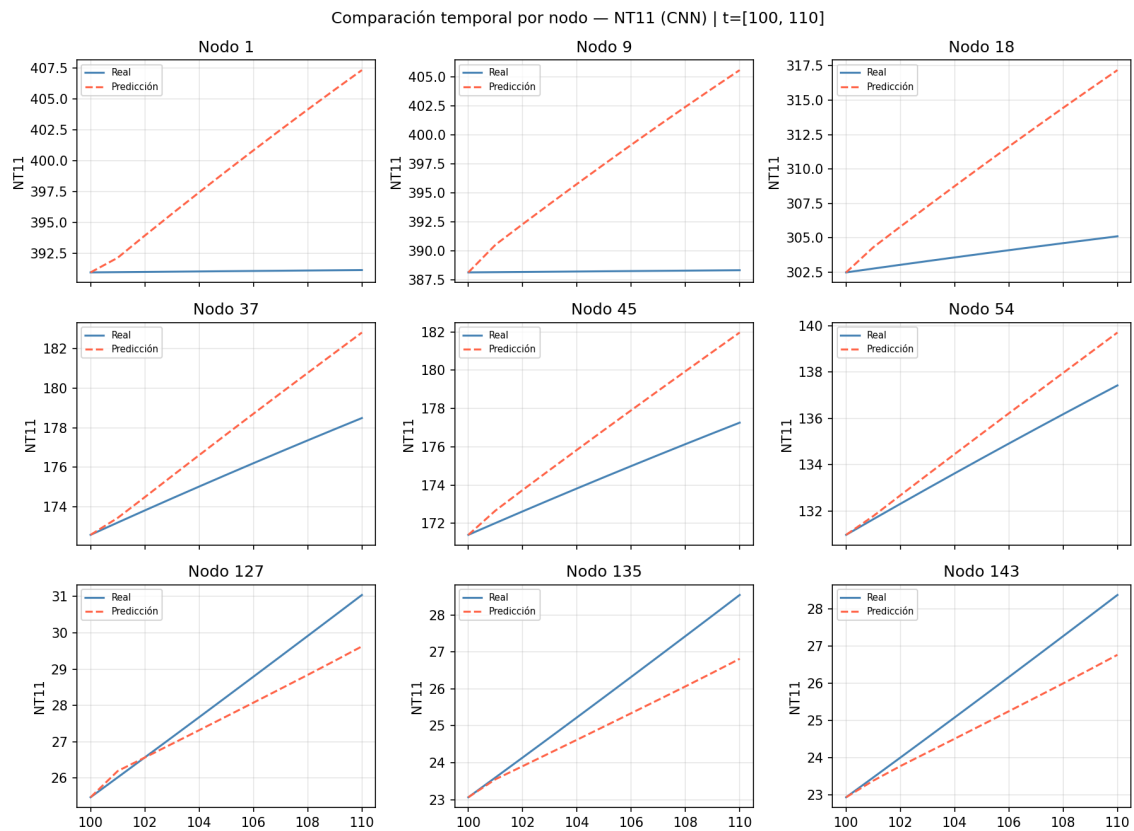


Figura 4.4.11: Comparación de temperatura calculada con CNN en los nodos centrales del CTES

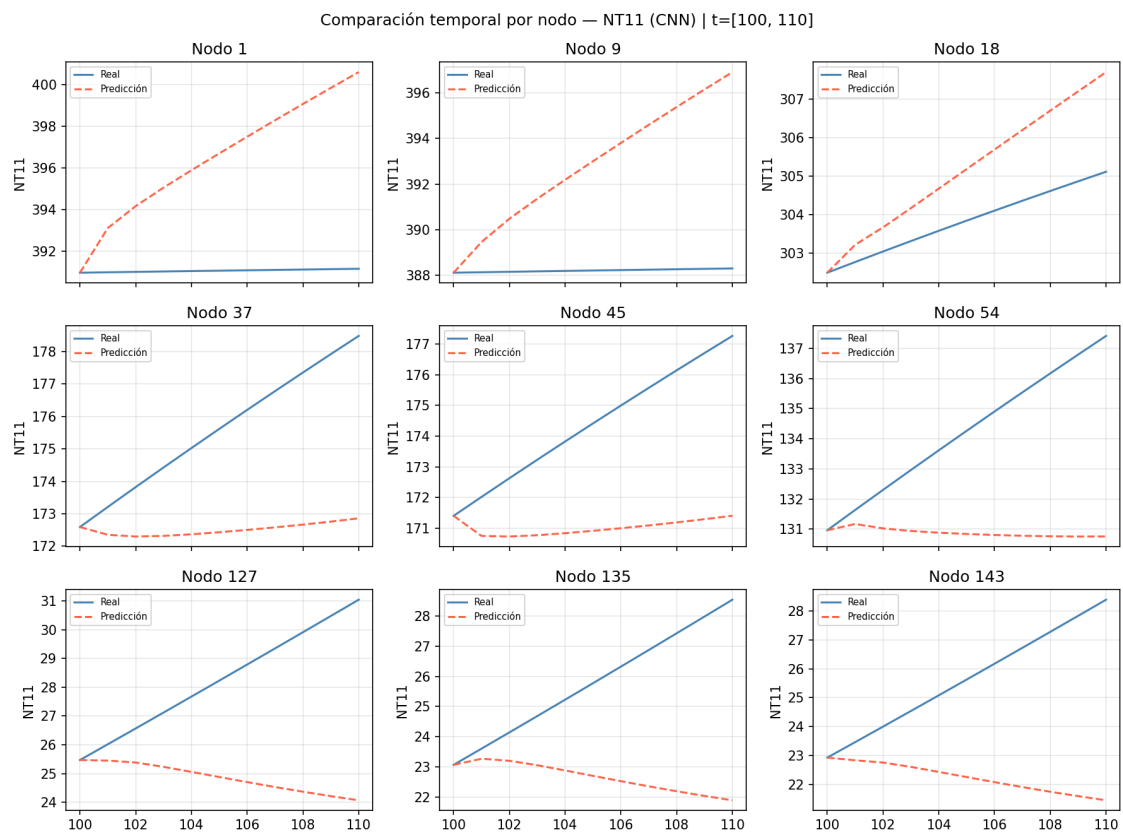


Figura 4.4.12: Comparación de temperatura calculada con RNN en los nodos centrales del CTES

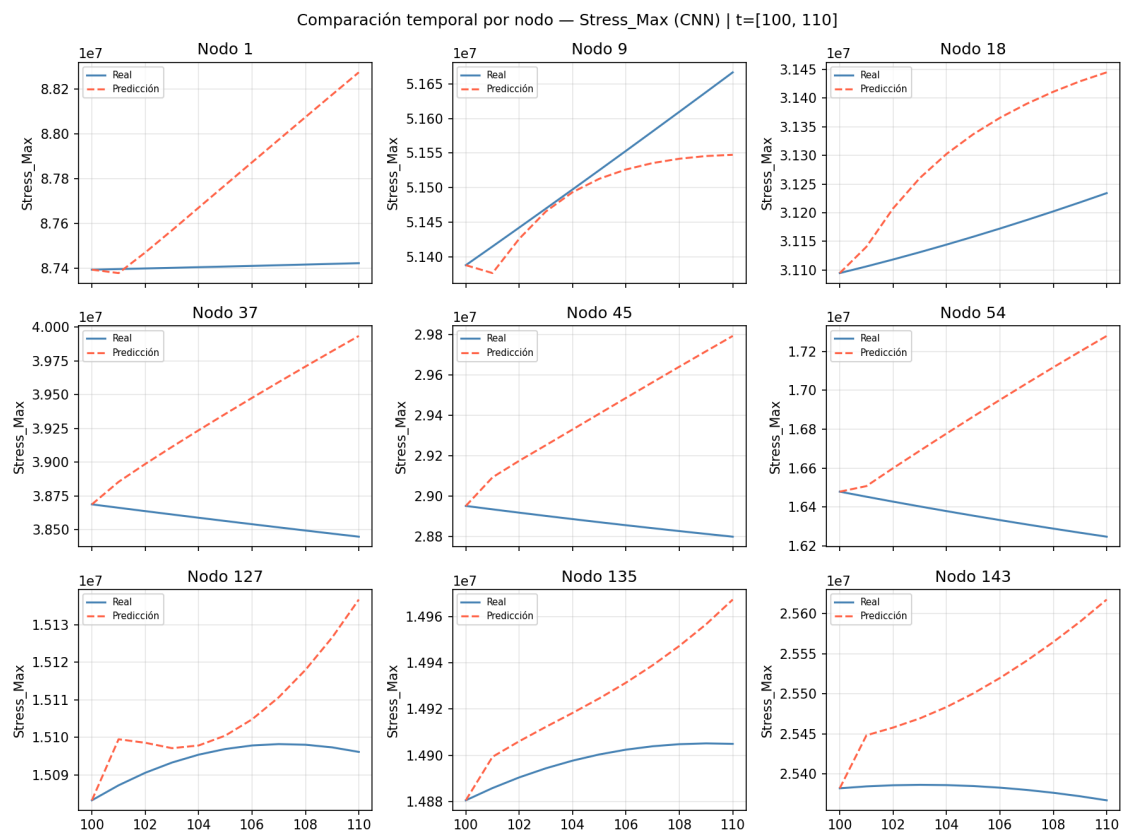


Figura 4.4.13: Comparación de tensión máxima calculada con CNN en los nodos centrales del CTES

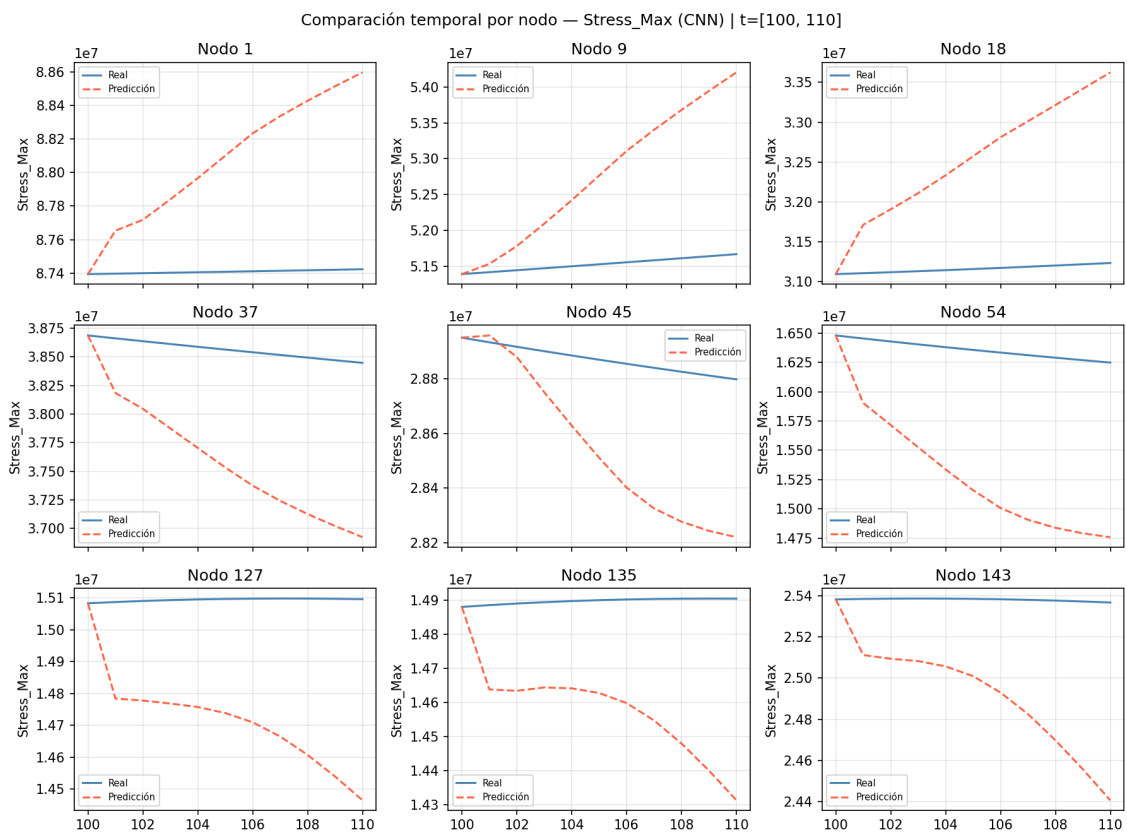


Figura 4.4.14: Comparación de tensión máxima calculada con RNN en los nodos centrales del CTES

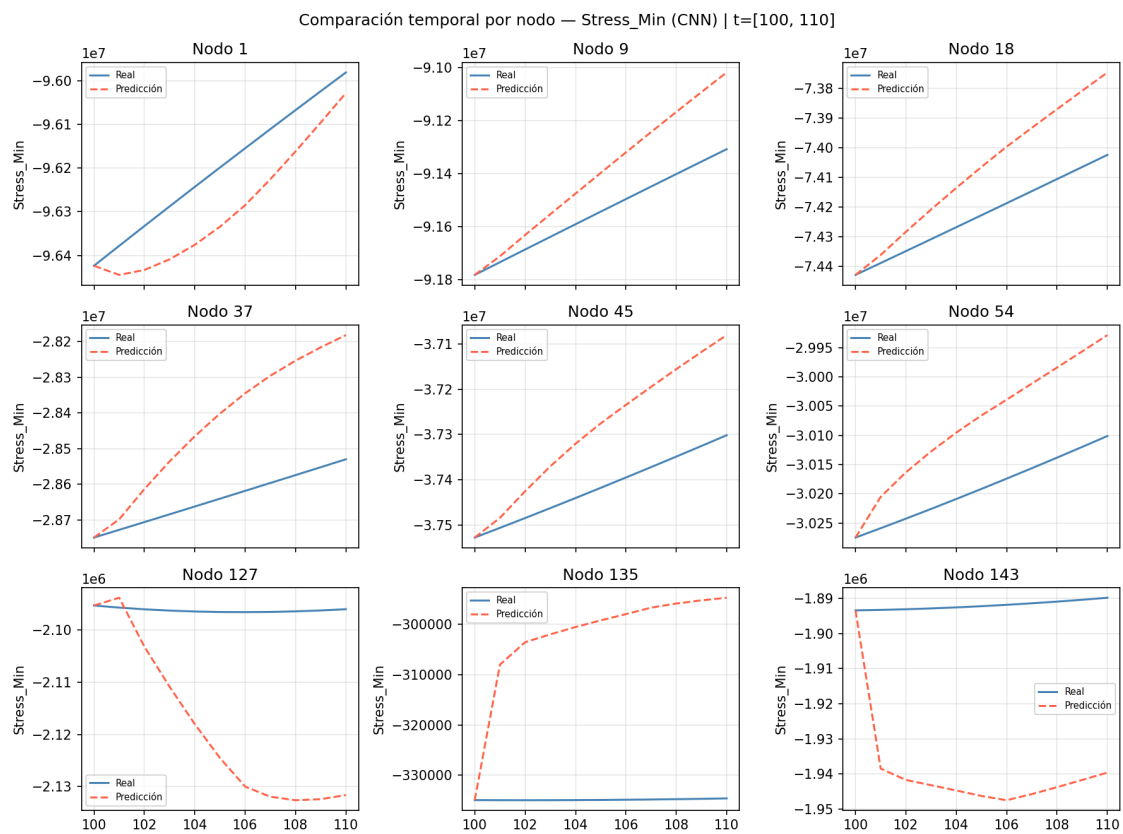


Figura 4.4.15: Comparación de tensión mínima calculada con CNN en los nodos centrales del CTES

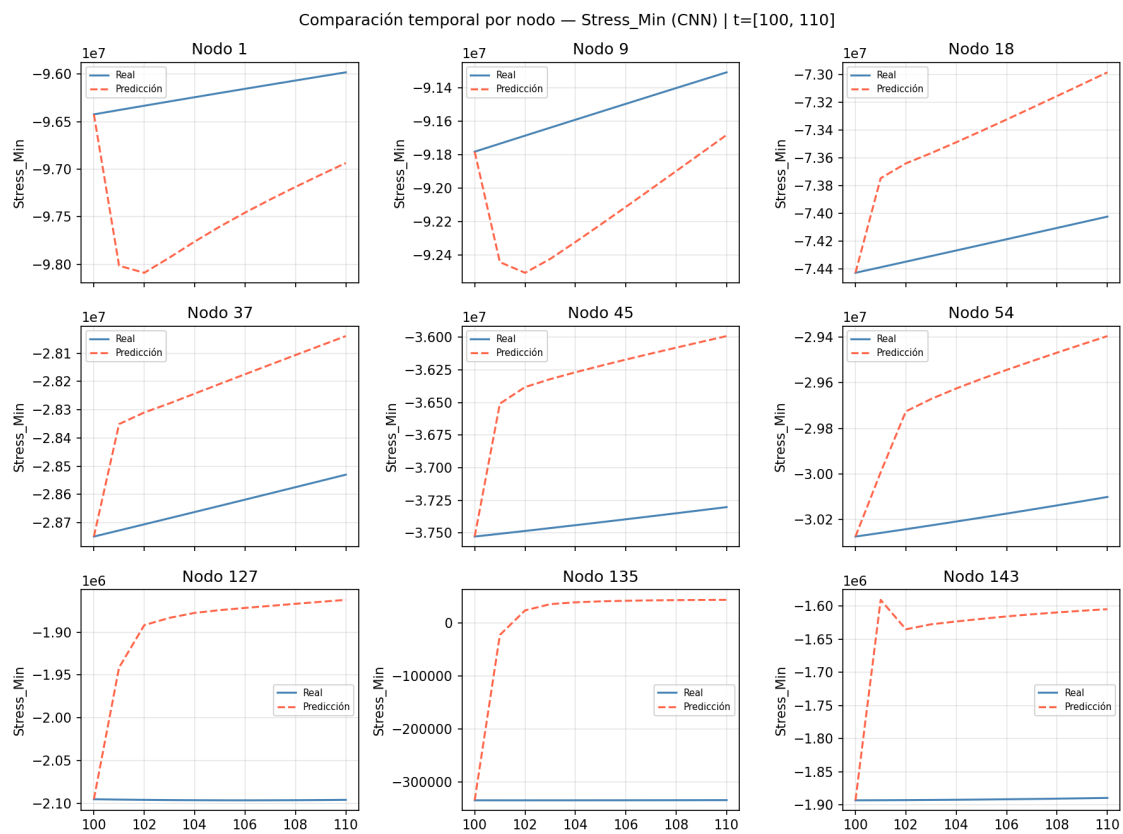


Figura 4.4.16: Comparación de tensión mínima calculada con RNN en los nodos centrales del CTES

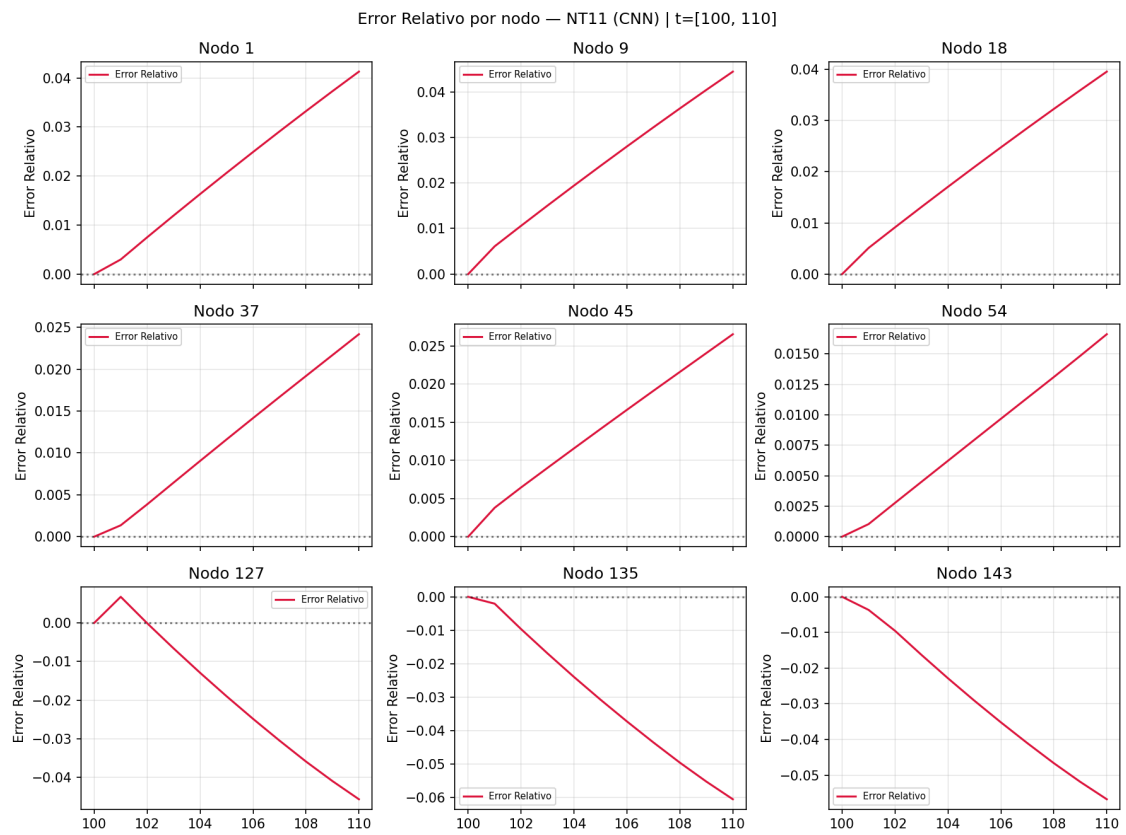
Errores

Figura 4.4.17: Errores relativos de temperatura calculada con CNN en los nodos centrales del CTES

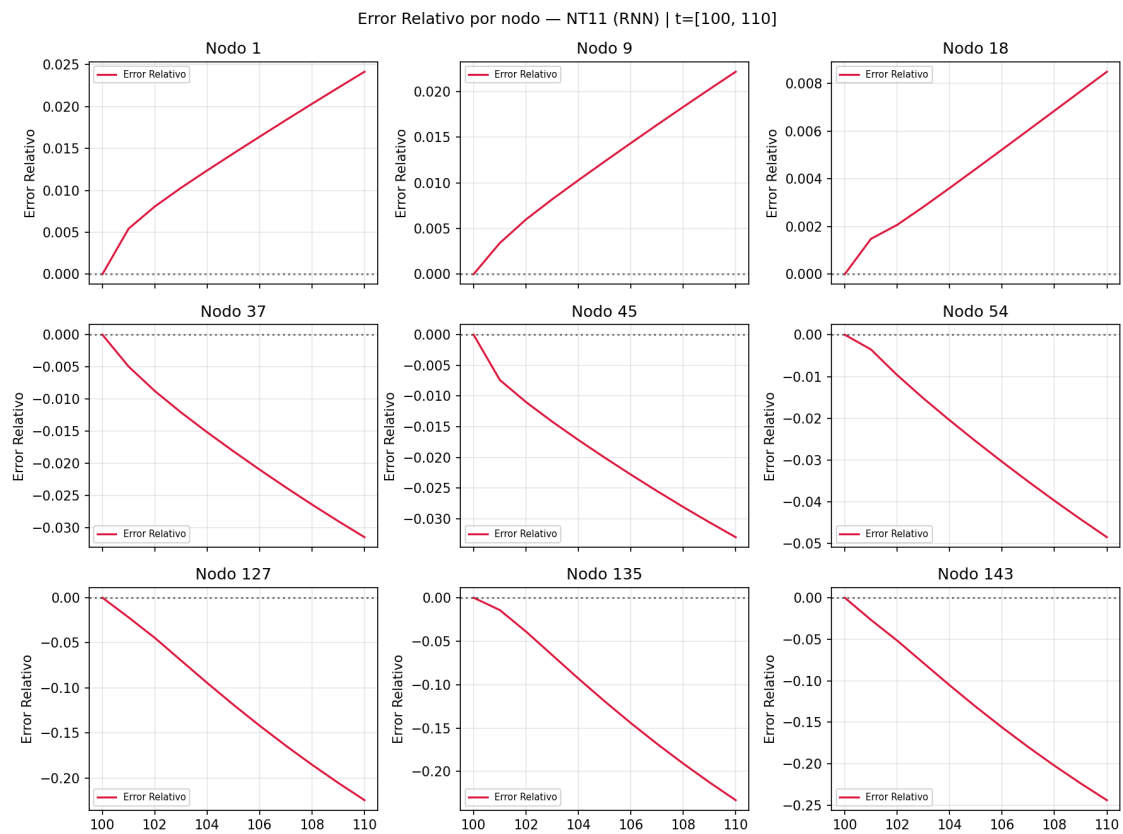


Figura 4.4.18: Errores relativos de temperatura calculada con RNN en los nodos centrales del CTES

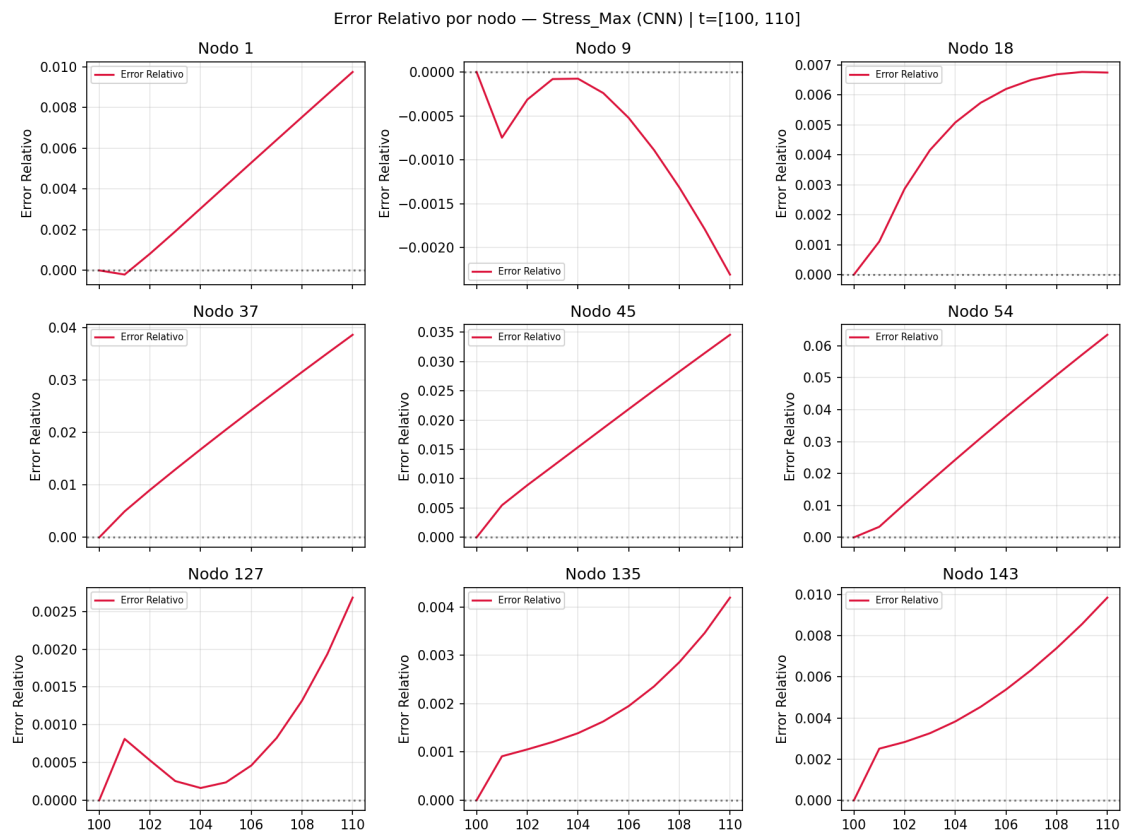


Figura 4.4.19: Errores relativos de tensión máxima calculada con CNN en los nodos centrales del CTES

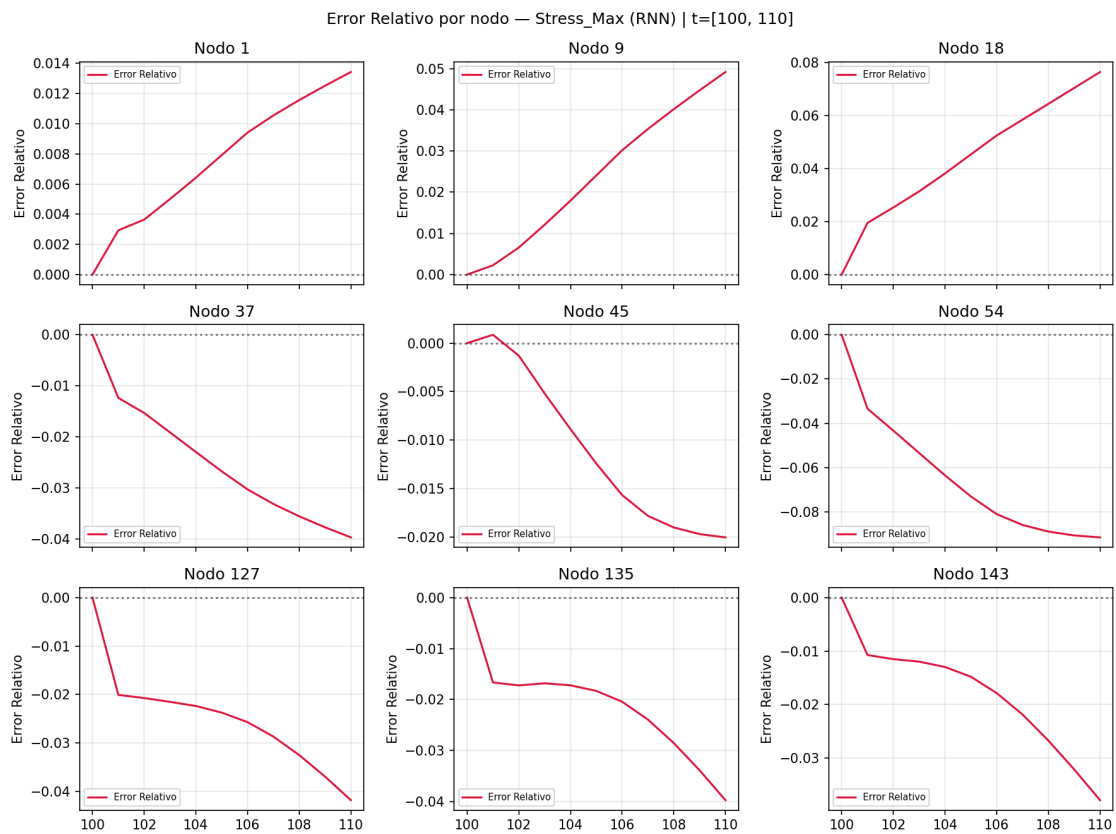


Figura 4.4.20: Errores relativos de tensión máxima calculada con RNN en los nodos centrales del CTES

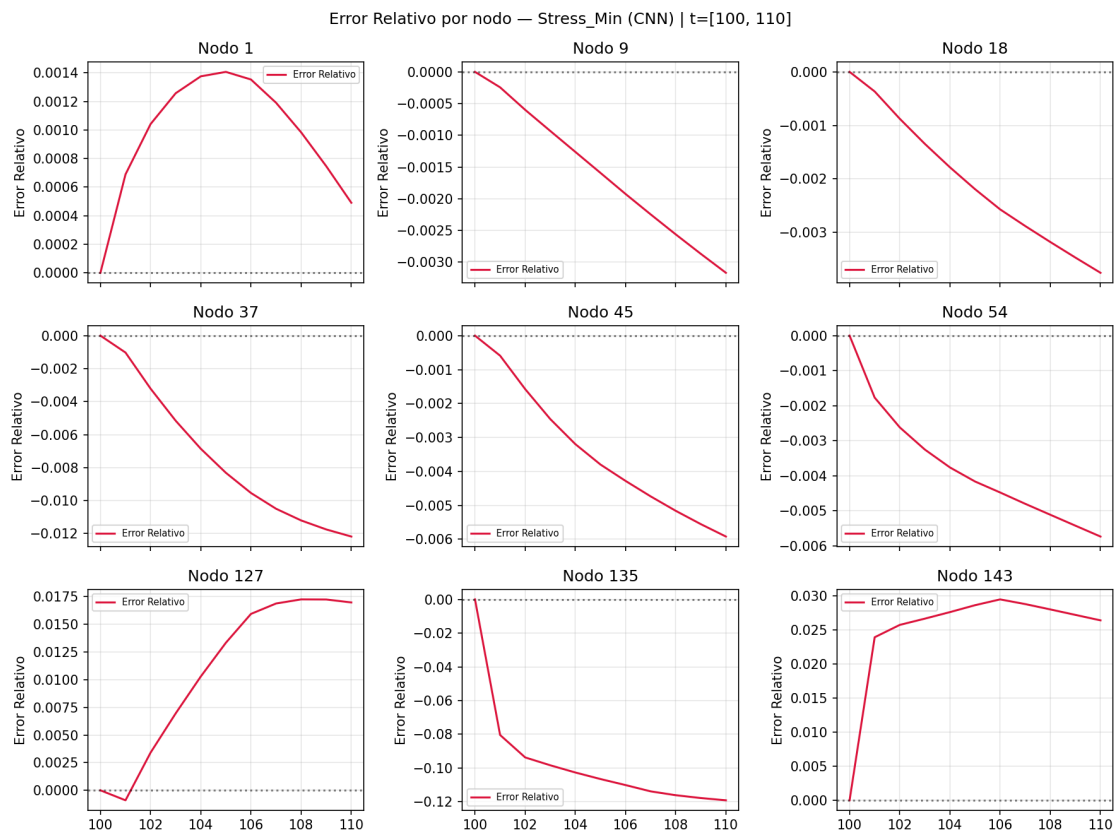


Figura 4.4.21: Errores relativos de tensión mínima calculada con CNN en los nodos centrales del CTES

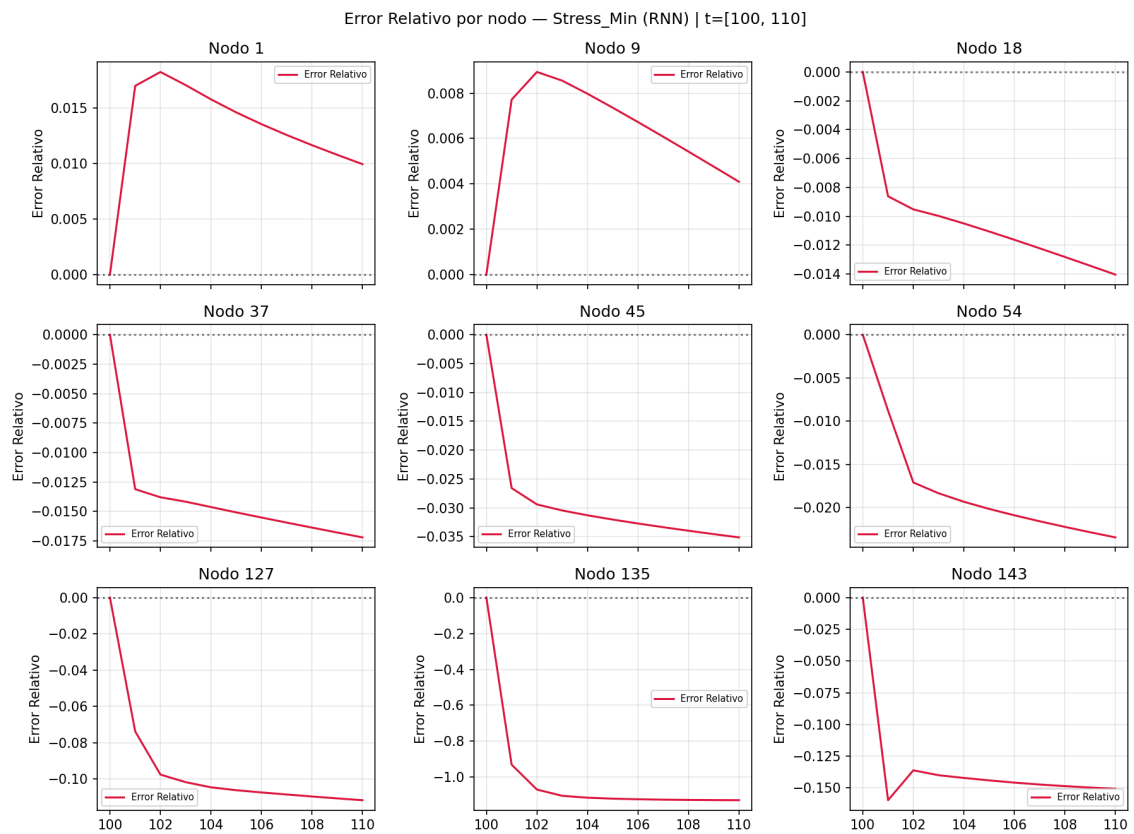


Figura 4.4.22: Errores relativos de tensión mínima calculada con RNN en los nodos centrales del CTES

Tiempo de inferencia

```
23      === TIEMPO DE PROCESADO DE LA EVALUACIÓN ===
24      Variable: NT11
25      Rango de pasos: t=[100, 110]
26      Modelo: /home/mdlsg/Documents/TFM/Data/Models/
OPTUNA_modelo_termico_E30_B128_LR0.00068_VNT11.pth
27      Tiempo total de inferencia de la red: 0.0066
segundos
28      Tiempo medio por frame: 0.000661 segundos/frame
29
30      === TIEMPO DE PROCESADO DE LA EVALUACIÓN ===
31      Variable: Stress_Max
32      Rango de pasos: t=[100, 110]
33      Modelo: /home/mdlsg/Documents/TFM/Data/Models/
OPTUNA_modelo_termico_E24_B64_LR0.00077_VStress_Max.pth
34      Tiempo total de inferencia de la red: 0.0054
segundos
35      Tiempo medio por frame: 0.000543 segundos/frame
36
37      === TIEMPO DE PROCESADO DE LA EVALUACIÓN ===
38      Variable: Stress_Min
39      Rango de pasos: t=[100, 110]
40      Modelo: /home/mdlsg/Documents/TFM/Data/Models/
OPTUNA_modelo_termico_E22_B32_LR0.00080_VStress_Min.pth
41      Tiempo total de inferencia de la red: 0.0049
segundos
42      Tiempo medio por frame: 0.000494 segundos/frame
43
44
45
```

Discusión

Dado estos resultados para el intervalo, sí es razonable utilizar estos datos para el lazo de control. El objetivo último es poder conocer la tensión a partir de la temperatura y que los saltos de tiempo son de 100 segundos, conocer la tensión del modelo con un error relativo del 5% con 1000 segundos de margen es un buen indicador para poder actuar en el lazo de control. Para optimizar los resultados se puede hacer uso de diferentes modelos CNN y RNN según lo que se desee calcular.

Capítulo 5

Instalación industrial de sistema con almacenamiento energético

Este capítulo aborda la instalación industrial de un sistema con almacenamiento térmico, detallando los componentes y la infraestructura requeridos para su integración. Con este propósito, se desarrolla un diagrama de tuberías e instrumentación o P&ID (*Piping and Instrumentation Diagram*), el cual describe el acoplamiento del sistema de almacenamiento térmico a una planta de cilindros parabólicos (CCP). En dicho diagrama se especifican y etiquetan de forma unívoca tanto los equipos mecánicos como los lazos de control asociados. Finalmente, una vez definida la arquitectura del sistema, se procede a su dimensionamiento técnico combinando los parámetros nominales de diseño de los componentes con datos estadísticos de demanda y consumo energético.



Figura 5.0.1: Muestra de posible instalación en azotea de edificio

5.1. ESQUEMAS CCP

5.1.1. Descripción General del Proceso

El sistema consta de tres subsistemas acoplados térmicamente: el circuito primario formado por CCP, HTF e intercambiadores de calor, el bloque de almacenamiento térmico basado en cemento (*CTES*) y el lazo secundario de potencia, diseñado para la generación de energía eléctrica mediante un ciclo de turbina de vapor.

El fluido caloportador (*HTF*, por sus siglas en inglés), compuesto por un aceite térmico, circula por el lazo primario absorbiendo la radiación solar concentrada en los colectores cilindro-parabólicos. Dependiendo del estado de carga del sistema y de la temperatura del fluido calo-portante, una válvula de control de tres vías (*TCV-101*) redirige el flujo hacia el bloque de hormigón *CTES* para su almacenamiento por calor sensible, o hacia el generador de vapor para la producción de electricidad con la turbina de vapor.

5.1.2. Designación de Equipos e Instrumentación (*Tags* ISA-5.1)

A continuación se detalla el inventario de equipos, válvulas y elementos de instrumentación necesarios para la construcción del diagrama de tuberías e instrumentación, ver plano de *P&ID* 8.2.1 en Anexos.

Tabla 5.1.1: Nomenclatura y especificación de componentes según norma ISA-5.1.

<i>Tag</i> de Equipo	Descripción Técnica	Función en el Proceso
PTC-100	Colector Cilindro-Parabólico	Captación y concentración solar.
CTES-200	<i>Concrete Thermal Energy Storage</i>	Sistema almacenamiento térmico.
ECON-300	Economizador	Precalentamiento de agua.
EVAP-300	Evaporador	Generación de vapor saturado.
SPH-300	Sobrecalentador	Calentamiento de vapor.
G-300	Generador Eléctrico Alternador	Conversión de potencia mecánica a eléctrica.
COND-300	Condensador de Superficie	Condensación del vapor de escape de la turbina.
P-101	Bomba de Circulación Lazo Primario	Impulsión del <i>HTF</i> con <i>VFD</i> .
P-401	Bomba de Alimentación de Agua	Impulsión de agua a alta presión hacia el ECON-300.
<i>Tag</i> Instrumento	Tipo de Instrumento	Lazo Asociado
TCV-101	Válv. Control Temperatura (3 vías)	Modulación de flujo hacia Carga o <i>CTES</i> .
TCV-102	Válv. Control Temperatura (3 vías)	Coordinación para llevar <i>HTF</i> a bombeo
TT-101	Transmisor de Temp.	Temperatura de entrada a SPH.
TT-102	Transmisor de Temp.	Temperatura de entrada a EVAP.
TT-103	Transmisor de Temp.	Temperatura de entrada a ECON.
TT-104	Transmisor de Temp.	Temperatura de entrada a CTES.
TT-105	Transmisor de Temp.	Temperatura de salida de CTES.
TCV-201	Válv. Control Paso	Temperatura de salida de CTES
TCV-202	Válv. Control Temperatura (3 vías)	Modulación Generación o ACS.
TCV-203	Válv. Control Temperatura (3 vías)	Coordinación para llevar vapor a generación.
PSV-201	Válv. Control Presión	Alivio en caso de sobrepresión.
TT-301	Transmisor de Temp.	Temperatura de salida de ECON.
TT-302	Transmisor de Temp.	Temperatura de salida de EVAP.
PT-301	Transmisor de Presión	Presión de salida de ECON.
PT-302	Transmisor de Presión	Presión de salida de EVAP.
PT-101	Transmisor de Presión	Control de integridad hidráulica del primario.
TT-303	Transmisor de Temp.	Temperatura de salida de turbinado.
TT-303	Transmisor de Presión	Presión de salida de turbinado.

5.1.3. Lazos de control

Lazo 100: Control de Captación y Almacenamiento de Energía (ΔT)

El objetivo principal del Lazo 100 es maximizar la eficiencia de captación del CCP evitando la degradación térmica del fluido caloportador. Para ello el fluido ha de mantenerse en el rango de temperatura 200°C a 400°C.

El actuador de TCV indica si la el fluido caloportante va a al sistema de generación o de almacenamiento en función de la tempeartura y la demanda. Si la diferencia de temperatura de captación supera la consigna definida ($\Delta T_{set} \approx 10 - 15^{\circ}\text{C}$), el variador de frecuencia (VFD) incrementa las revoluciones de la bomba P-101. Si la radiación decrece y TCV_{101} se aproxima a la temperatura de retorno, la bomba reduce su velocidad hasta el límite mínimo operativo o se detiene. Por otra parte, si la demanda es mínima o nula y el HTF tiene energía esta se dirige al sistema CTES. Y si la turbina dispone de energía y vapor para funcionar con la radiación aumentando o manteniéndose, se re-dirige el HTF al CTES.

El almacenamiento CTES no dispone de elementos finales de control propios, sino que actúa como un nodo esclavo gobernado por la posición de la válvula modulante TCV-101. El estado de carga térmico se calcula con las simulaciones y conocidas las temperaturas de entrada TCV-101 y de salida TT-105.

Lazo 200: Gestión del Agua Calentada por el Almacenamiento

El TCV-201 actúa como control de salida del sistema, si la temperatura no es suficiente continúa en el CTES.

La válvula TCV-201 en función de la temperatura del agua la envía a ACS o la devuelve al sistema a modo de Bypass.

Lazo 300: Control de Generación de Vapor

El control de lazo cerrado (*PID*) regula el caudal de vapor de agua sobrecalentado que ingresa a la turbina TUR. El sistema monitoriza la presión mediante el PT-301, PT-302 durante el proceso de calentamiento y con PSV-201 a la entrada de la turbina. Esa última válvula dispone de alivio para casos de sobrepresión. Se actúa sobre la válvula de gobierno de admisión para mantener la velocidad de rotación sincrónica necesaria en el generador eléctrico G. Paralelamente, el nivel de líquido en el generador de vapor SG-300 se estabiliza modulando la velocidad de la bomba de alimentación de agua P-301. Se tiene constancia de la temperatura y presión de salida de la turbina con las TT-303 y PT-303.

5.1.4. Lógica de Interbloqueos (*Interlocks*)

- *Interlock* I-01 (Protección contra Estancamiento): Si la temperatura medida por el TT-201 excede los 130°C debido a un fallo en las bombas del lazo primario, el

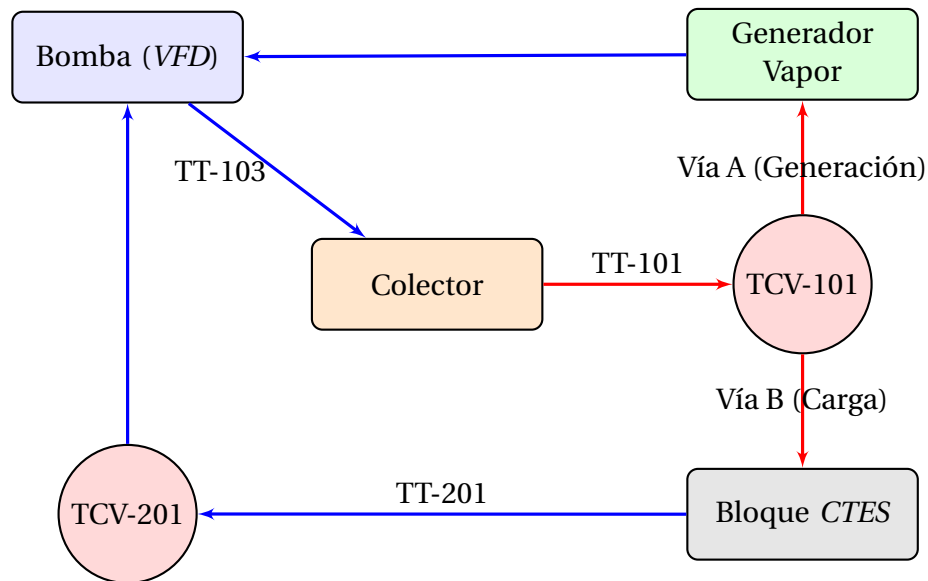


Figura 5.1.1: Esquema de bloques funcional del diagrama de tuberías primario orientado a la generación de potencia.

lazo envía una señal prioritaria para desfocar el captor solar PTC-100.

- **Interlock I-02** (Disparo de la Turbina por Condiciones de Vapor): Si el transmisor PT-302 detecta una caída de presión crítica o una falta de sobrecalentamiento que arriesgue la presencia de gotas líquidas en la expansión, se ordena el cierre de emergencia de la válvula de corte rápido y el aislamiento térmico de la turbina TUR (*Turbine Trip*), derivando el excedente hacia el condensador COND o abriendo la válvula de seguridad (PSV-301).

5.2. CÁLCULO POTENCIA CCP

La potencia necesaria se obtiene a partir de estimaciones que incluyan todas las pérdidas por los diferentes equipos:

$$Q_{SOL} = A_{CCP} \cdot Irr \quad (5.2.1)$$

Energía aportada por radiación

$$\eta_{opt} = \rho \cdot \alpha \cdot \gamma \cdot \tau \quad (5.2.2)$$

Rendimiento óptico: reflectividad del espejo (ρ), absortancia del tubo (α), factor de intercepción (γ), transmitancia de la cubierta (τ)

$$\eta_{geo} = \cos(\theta) \cdot K(\theta) \cdot F_{sombra} \quad (5.2.3)$$

Rendimiento geométrico: Efecto coseno, modificador de ángulo de incidencia, factor de sombra

$$Q_{abs} = Q_{SOL} \cdot \eta_{opt} \cdot \eta_{geo} \quad (5.2.4)$$

Calor absorbido.

$$Q_{perd} = Q_{Rad} + Q_{Conv} + Q_{Cond} \quad (5.2.5)$$

Pérdidas térmicas: Radiación vidrio-cielo, convección vidrio-ambiente, conducción en soportes de tubos.

$$Q_{útil} = Q_{abs} - Q_{perd} = m \cdot C_p \cdot (T_{sal} - T_{entr}) \quad (5.2.6)$$

Calor útil.

$$Q_{útil,col} = \eta_{Op} \cdot I_0 \cdot \cos(\theta) \cdot K(\theta) \cdot A_{col} - UL \cdot (T_{abs} - T_{amb}) \cdot A_{abs} = 563 kW \quad (5.2.7)$$

$$Q_{útil,lazo} = n_{col} \cdot Q_{útil,col} = 2253 kW \quad (5.2.8)$$

$$\dot{m}_{col} = \frac{Q_{útil,col}}{h_{sal} - h_{ent}} = 2,33 kg/s \quad (5.2.9)$$

$$\dot{m}_{lazo} = \frac{Q_{útil,lazo}}{h_{sal} - h_{ent}} = 9,34 kg/s \quad (5.2.10)$$

$$P_{elec,col} = Q_{útil,col} \cdot \eta_{elec} = 208,31 kW \quad (5.2.11)$$

$$P_{elec,lazo} = n_{col} \cdot P_{elec,col} = 833,24 kW \quad (5.2.12)$$

Adicionalmente, se debe dimensionar el sistema de forma que sea capaz de sumi-

Absortancia	96 %
Reflectividad	94 %
Transmisividad	96 %
Factor interceptación	96 %
Factor ensuciamiento	97 %
Modificador ángulo incidencia	0.99
Ángulo Incidencia	9°
Área apertura colector	817.5 m ²
Irradiancia diseño	950 W / m ²
Coefficiente global pérdidas	4.7 W / m ² K
Temperatura media absorbedor	343 °C
Temperatura ambiente	25 °C
Área tubo absorbedores	33.0 m ²
Colectores por lazo	4
Múltiplo solar	1.2
Horas equivalentes solares	6.5
Colectores por lazo	4
Entalpía HTF (293 °C)	559 kJ / kg
Entalpía HTF (393 °C)	800 kJ / kg
Rendimiento eléctrico	37 %

Tabla 5.2.1: Valores habituales para los cálculos

nistrar energía a los sistemas auxiliares de forma que pueda funcionar de forma aislada de la red:

$$P_{pump} = \frac{\rho \cdot g \cdot Q \cdot H}{eta} \quad (5.2.13)$$

Bomba de HTF

- ρ : Comprendido entre 694 y 912 $\frac{kg}{m^3}$ según la temperatura de operación. Se toma el valor más desfavorable, 912 $\frac{kg}{m^3}$.
- Q: calculada el caudal másico de un colector en [4.2.9], se puede obtener el caudal volumétrico:

$$Q = \frac{\dot{m}}{\rho} = 0,00255 \frac{m^3}{s} \quad (5.2.14)$$

- Considerando una tubería de 2 pulgadas de diámetro para la tubería de HTF para el lazo:

$$v_{HTF} = \frac{\dot{m}}{\rho_{HTF} \cdot A_{tubería}} = \frac{2,33 \cdot 4}{912 \cdot 0,008} = 1,28 m/s \quad (5.2.15)$$

- H: Pérdidas de carga por válvulas y codos, se puede aproximar con $K \cdot \frac{v^2}{2 \cdot g}$, donde $K \approx 0,9$ en codos rectos, $K \approx 2,5$ en válvulas de seguridad y $K \approx 5$ en válvulas

rectas. Tiene al menos 4 codos y una válvula recta:

$$H \approx \frac{(0,9 \cdot 4 + 5) \cdot 1,28^2}{2 \cdot 9,81} = 0,72 \quad (5.2.16)$$

- η : Rendimiento de la bomba, en su punto de diseño puede alcanzar el 90 %, en general se aproxima a 75 % en un uso con variador de frecuencia como es el caso.

$$P_{HTF} = \frac{912 \cdot 9,81 \cdot 0,01 \cdot 0,72}{0,75} = 21,9 kW \quad (5.2.17)$$

Considerando 10 horas de funcionamiento durante el día, los 365 días del año:

$$E_{HTF} = 0,355 \cdot 10 \cdot 365 = 79,96 \frac{kWh}{año} \quad (5.2.18)$$

Bomba de agua

- ρ : $1000 \frac{kg}{m^3}$.
- Q : estimado el caudal másico de $0,3 \frac{kg}{s}$ por vivienda y aplicando un 70 % de simultaneidad a 30 viviendas se obtiene $\dot{m} = 6,3 \frac{kg}{s}$ y se puede obtener el caudal volumétrico:

$$Q = \frac{\dot{m}}{\rho} = 0,0063 \frac{m^3}{s} \quad (5.2.19)$$

- Considerando una tubería de 4 pulgadas de diámetro para la tubería de ACS (para garantizar una velocidad inferior a 2 m/s) para el lazo:

$$v_{ACS} = \frac{\dot{m}}{\rho_{ACS} \cdot A_{tubería}} = \frac{7,14}{1000 \cdot 0,008} = 0,87 m/s \quad (5.2.20)$$

- H : Pérdidas de carga por válvulas y codos, se puede aproximar con $K \cdot \frac{v^2}{2 \cdot g}$, donde $K \approx 0,9$ en codos rectos, $K \approx 2,5$ en válvulas de seguridad y $K \approx 5$ en válvulas rectas. Tiene al menos 4 codos, una válvula recta y una válvula de seguridad por lo que

$$H \approx \frac{(0,9 \cdot 4 + 5 + 2,5) \cdot 0,98^2}{2 \cdot 9,81} = 0,42 \quad (5.2.21)$$

- η : Rendimiento de la bomba, en su punto de diseño puede alcanzar el 90 %, en general se aproxima a 75 % en un uso con variador de frecuencia como es el caso.

$$P_{ACS} = \frac{1000 \cdot 9,81 \cdot 0,0015 \cdot 5,4}{0,75} = 34,93 kW \quad (5.2.22)$$

Considerando 10 horas de funcionamiento durante el día, los 365 días del año:

$$E_{ACS} = 0,106 \cdot 10 \cdot 365 = 127,51 \frac{MWh}{año} \quad (5.2.23)$$

Por lo tanto, aún quedan 151.47 kW para el suministro a las viviendas. Si se calculase para un lazo, cambiando a su vez caudal y diámetro de tuberías de HTF y ACS para 120 viviendas, restarían 605.88 kW para el suministro a viviendas.

5.3. INSTALACIÓN RESIDENCIAL

5.3.1. Número de viviendas

Para el cálculo residencial, se estimará que la potencia generada por el colector sea suficiente para mantener un cierto número de casas sin caídas de potencia, para ello se estima 5 kW como potencia máxima instalada en cada vivienda, aunque esté por encima de las recomendaciones de distribuidoras como [19]. El INE aporta información sobre el número de viviendas por edificio, Figura 5.3.1, extraído de INE [20], donde se aprecia que el mayor número de viviendas son en edificios con multitud de viviendas, por lo que el cálculo se hará para un número no menor de 10 viviendas por instalación, en caso de obtener menor número de viviendas por colector se instalarán varios colectores.

	Total (valores absolutos)
Total	
Vivienda unifamiliar (chalet, adosado, pareado...)	
Total	6.226.472
Edificio con 2 viviendas	
Total	448.435
Edificio de 3 a 9 viviendas	
Total	2.382.407
Edificio con 10 o más viviendas	
Total	9.752.731

Figura 5.3.1: Número de viviendas por tipo de edificio

Potencia instalada por vivienda	6 kW
Horas equivalentes solares	6.5 h

Tabla 5.3.1: Valores habituales para el diseño de lazos

$$\frac{n_{viviendas}}{n_{colectores}} = \frac{P_{elec,col}}{P_{vivienda}} = 30,29 \frac{viv}{col} \quad (5.3.1)$$

$$\frac{n_{viviendas}}{n_{colectores}} = \frac{P_{elec,lazo}}{P_{vivienda}} = 121,18 \frac{viv}{lazo} \quad (5.3.2)$$

Se considerará que un colector puede suministrar energía a un máximo de 30 hogares. Un lazo puede suministrar energía a un máximo de 120 hogares.

5.3.2. Energía en CTES

Los cálculos para la energía consumida por vivienda se realizan basados en los datos del consumo eléctrico por vivienda [21] obteniéndose la mediana del consumo anual en España es de 3045 kWh/vivienda al año. Se debe calcular la energía que puede generar el colector para compararla con la energía consumida, la diferencia entre ambas será la que se pueda aprovechar con los CTES.

$$E_{con,viv} = 3045 \frac{kWh}{vivienda} \cdot 30viviendas = 91,35 \frac{MWh}{año} \quad (5.3.3)$$

Por parte de la energía que pueda generar el colector, se estima considerando el número de horas y el calor generado:

$$E_{disp,CTES} = P_{elec,col} \cdot 6,5h \cdot 365d - E_{con,viv} = 760331,5kWh - 103530kWh = 656,802 \frac{MWh}{año} \quad (5.3.4)$$

Este cálculo permite un predimensionado del número de equipos CTES que podrán instalarse como máximo en una instalación de estas características. Considerando el calor sensible almacenable para CTES con fibra e ilmenita con los datos de la tabla 3.2.1:

$$M_{CTES} = 250mm \cdot \pi \cdot (125 - 12,5)^2 mm^2 \cdot \rho_{C+fB+Il} = 26,22kg \quad (5.3.5)$$

Suponiendo el CTES que pueda absorber, en promedio, 100°C de temperatura en un ciclo de un día:

$$E_{CTES} = M_{CTES} \cdot C_{pC*fB+Il} \cdot 100K \cdot 2,778 \cdot 10^{-7} \frac{kWh}{J} = 0,6025kWh \quad (5.3.6)$$

$$\frac{E_{disp,CTES}}{365días} = 1,8 \frac{MWh}{día} \quad (5.3.7)$$

La cantidad de CTES que serían necesarios para almacenar toda esa energía es inviable. Se dimensiona por tanto en función de la cantidad de energía necesaria para calentar agua caliente sanitaria: El consumo total de agua en hogares fue de 128 litros por habitante y día, ver [22]. Además, el número de personas por vivienda es actualmente 2.6 personas por vivienda, tomando el total de personas de [23] y el número de viviendas de [20].

$$E_{ACS} = 128 \frac{l}{pers} \cdot 2,6 \frac{pers}{viv} \cdot 30viv \cdot 4,184 \frac{J}{kg \cdot K} \cdot 40K = 0,464kWh \quad (5.3.8)$$

Por último, se considerará que el CTES no aporta toda la energía que tiene alma-

cenada por tener pérdidas con el ambiente:

$$n_{CTES} = \frac{E_{ACS}}{E_{CTES} \cdot 0,3} = \frac{0,464}{0,6025 \cdot 0,3} = 2,57 \quad (5.3.9)$$

En conclusión, se utilizarán 3 sistemas CTES por cada colector para cubrir demandas de agua caliente sanitaria de 30 viviendas. Cada lazo necesitará 11 CTES para cubrir demandas de agua caliente sanitaria de 120 viviendas.

Capítulo 6

Estudio de viabilidad económica

La transición de un modelo numérico y experimental hacia su aplicación en un entorno operativo real requiere un análisis de su viabilidad técnico-económica. El objetivo de este capítulo es cuantificar los recursos financieros necesarios para la integración a escala del sistema de generación de energía y almacenamiento térmico (TES) optimizado, evaluando si los beneficios derivados de su eficiencia operativa e impacto ambiental justifican la inversión inicial. El análisis se basa en los siguientes cálculos:

1. **Desglose de Costes de Capital y Operación (CAPEX y OPEX):** Se auditan los costes directos de adquisición de equipos, así como los costes indirectos derivados de la ingeniería.
2. **Evaluación del Coste de la Energía:** Mediante la formulación del Coste Nivelado de la Energía (LCOE), se compara el coste de producir energía con métodos habituales con este sistema. Además, se estudia el efecto medioambiental mediante el cálculo de la huella de carbono de la implantación y la reducción de emisiones de gases de efecto invernadero (GEI) con este sistema.
3. **Indicadores de Rentabilidad y Sensibilidad:** Se calculan métricas financieras estándar: Valor Actual Neto (VAN), la Tasa Interna de Retorno (TIR) y el periodo de recuperación de la inversión (*payback*).

6.1. ESTIMACIÓN DE COSTES

6.1.1. CAPEX

Colector cilindro parabólico

Este equipo se puede dividir en 4 componentes principales:

- **Espejos:** $\approx 40\text{€}/m^2$. Considerando el sistema usado de referencia en la sección 4.2, cada colector requiere unos $820\text{ }m^2$, luego $\approx 33000\text{€}$

- **HTF:** $\approx 4€/l$. Por lo general, se requieren alrededor de 300 l por cada colector, por lo tanto $\approx 1200€$.
- **Guiado:** $\approx 5000€/colector$.
- **Motor:** $\approx 3000€/colector$.

Sistema de almacenamiento de energía térmica

Para este sistema se puede estimar simplemente considerando el precio del cemento en obra y añadiendo el los aditivos en peso. En un ajuste fino se podría considerar el precio del molde si fuera perdido.

- **Cemento:** En obra su precio es $100 \frac{€}{m^3}$, entonces $\frac{80kg}{\rho_{cemento}} \approx 0,032m^3 \cdot 100 \frac{€}{m^3} \approx 3,2€$
- **Basalto:** Su precio oscila entre $2 \frac{€}{kg}$ y $5 \frac{€}{kg}$, entonces $\approx 0,8kg \cdot 5 \frac{€}{kg} \approx 4€$
- **Ilmenita:** Su precio oscila entre $20 \frac{€}{kg}$ y $30 \frac{€}{kg}$, en este estudio se pretende usar el de deshecho de procesos de síntesis de titanio por lo que el precio será inferior. Se considerará un 50 % del precio original por mantener un margen de seguridad $\approx 4kg \cdot 15 \frac{€}{kg} \approx 60€$

Generador y turbina

Para el sistema que se está diseñando de pequeña capacidad de generación, tendrá que estimarse con equipos de 1MW que son los habituales:

- **Turbina-generador:** $\approx 90000€$

Economizador-Evaporador-Supercalentador

- **Economizador:** $\approx 70000€$
- **Evaporador** $\approx 50000€$
- **Supercalentador** $\approx 30000€$

Bombas

- **Bomba HTF:** $\approx 10000€$, Tiene que ser una bomba diseñada para soportar altas temperaturas por lo que su precio se incrementa respecto a otra similar para agua.
- **Bomba agua:** $\approx 5000€$

Equipo de control

- **Válvulas de 3 vías con control de tempertura:** $\approx 2000€$. Al tratarse de 4 válvulas sería un total de $\approx 8000€$.

- **Válvula de control de paso:** $\approx 1000\text{€}$.
- **Válvula de control de presión:** $\approx 5000\text{€}$.
- **Transmisores de temperatura industriales:** $\approx 300\text{€}$. Al tratarse de 8 transmisores sería un total de $\approx 2400\text{€}$.
- **Transmisores de presión de vapor de agua:** $\approx 1000\text{€}$. Al tratarse de 4 transmisores sería un total de $\approx 4000\text{€}$

Ver anexo 8.3.1 para el desglose total con amortizaciones y cuentas contables asignadas.

6.1.2. OPEX

El mayor coste de generación es el consumo de las bombas que ya se incluyó en el cálculo del consumo, por lo que el coste de operación pasa a ser el coste de mantenimiento y limpieza:

- Mantenimiento: Contrato para una hora de técnico al mes revisando los equipos $\approx 50 \frac{\text{€}}{\text{hora}}$
- Limpieza: Contrato para una hora a la semana limpiando los equipos $\approx 20 \frac{\text{€}}{\text{hora}}$

$$OPEX = 20 \cdot 52 \frac{\text{semanas}}{\text{año}} + 50 \cdot 12 \frac{\text{meses}}{\text{año}} = 1640 \frac{\text{€}}{\text{año}} \quad (6.1.1)$$

6.2. COSTES NIVELADOS

6.2.1. Costo Nivelado de Energía (LCOE)

También conocido como LCOE es la herramienta habitual para conocer la rentabilidad de un equipo que aporte o consuma energía. Para obtenerlo se necesitan algunos parámetros ya discutidos y otros que tendrán que estimarse:

- Inversión inicial (CAPEX)
- Costos operativos y de mantenimiento (OPEX)
- Costos de combustible: en este caso no se necesitaría salvo que se utilice para calcular el LCOE de otro equipo para comparar.
- Tasa de descuento %
- Producción de energía esperada (MWh/año).
- Vida útil del proyecto.

donde: I_t = Inversión en el año t ,
 O_t = Costos de operación y mantenimiento en el año t ,
 F_t = Costos de combustible en el año t ,
 E_t = Energía generada en el año t ,
 r = Tasa de descuento,
 n = Vida útil del proyecto (años).

$$\text{LCOE} = \frac{\sum_{t=0}^n \frac{I_t + O_t + F_t}{(1+r)^t}}{\sum_{t=0}^n \frac{E_t}{(1+r)^t}} = 48,35 \frac{\text{€}}{\text{MWh}} \quad (6.2.2)$$

6.2.2. Costo Nivelado de Almacenamiento (LCOS)

También conocido como LCOS (Levelized Cost of Storage), es la herramienta analítica habitual para evaluar la rentabilidad y el costo efectivo por unidad de energía recuperada de un sistema de almacenamiento térmico. Para obtenerlo se requieren los siguientes parámetros:

- Inversión inicial (CAPEX).
- Costos operativos y de mantenimiento (OPEX).
- Costo de la energía de carga: el precio o costo de oportunidad de la energía utilizada para abastecer el almacenamiento.
- Tasa de descuento %.
- Energía anual descargada (MWh/año): la energía útil real que el sistema entrega tras considerar las pérdidas por eficiencia global (*round-trip efficiency*).
- Vida útil del proyecto.

Se utilizará únicamente el coste de los elementos CTES y ningún coste en OPEX ya que este quedaría incluido en los costes del CCP. El coste de energía es nulo por provenir de CCP y la energía almacenada es la necesaria para calentar ACS de las 30 viviendas.

donde: I_t = Inversión en el año t ,
 O_t = Costos de operación y mantenimiento en el año t ,
 C_t = Costo de la energía de carga en el año t ,
 E_t = Energía entregada (descargada) en el año t ,
 r = Tasa de descuento,
 n = Vida útil del proyecto (años).

$$(6.2.3)$$

$$LCOS = \frac{\sum_{t=0}^n \frac{I_t + O_t + C_t}{(1+r)^t}}{\sum_{t=0}^n \frac{E_t}{(1+r)^t}} = 33,19 \frac{\text{€}}{\text{MWh}} \quad (6.2.4)$$

6.3. EFECTO MEDIOAMBIENTAL

6.3.1. Huella de carbono

La huella de carbono es la medida de *la totalidad de gases de efecto invernadero emitidos por efecto directo o indirecto por un individuo, organización, evento o producto*. Se distingue entre emisiones directas y emisiones indirectas [24]:

- **Emisiones directas de GEI:** son emisiones de fuentes que son propiedad de o están controladas por la organización.
- **Emisiones indirectas de GEI:** son emisiones consecuencia de las actividades de la organización.

La generación de energía eléctrica con un sistema renovable como son los CCP no tiene emisiones directas, por lo que bastará con conocer las emisiones indirectas y compararlas con las emisiones que se producirían con el gasto de energía eléctrica y agua caliente.

Emisiones indirectas

Las emisiones que se pueden tener en consideración son aquellas referidas al transporte de los equipos hasta su instalación.

- Transporte marítimo: los equipos más complejos o voluminosos serán enviados con seguridad desde otro país.
 - 225 toneladas de diésel marítimo al día a 24 nudos en barcos de 8000 TEU, con 3.206 toneladas de CO_2 por tonelada de diésel.
 - 8 semanas desde China, 2 desde Rotterdam.
 - $\frac{225 \text{ toneladas/día}}{8000 \text{ TEU}} \cdot 56 \text{ días} \cdot 3,206 \frac{\text{ton}CO_2}{\text{ton diésel}} = 5,05 \frac{\text{ton}CO_2}{\text{TEU}}$
 - Por la variabilidad de los equipos se considera necesario mínimo 5 TEUs distintos (Espejos y auxiliares, turbina y generador, evaporador, recuperador y super calentador).
- Transporte terrestre: tanto los equipos que requieran envío marítimo como los que no tendrán que realizar su último transporte hasta el lugar de la instalación. Suponiendo que los equipos que provengan de ultramar solo recorran 150 km mientras que los equipos de control puedan venir desde el punto más lejano del país, 800 km.
 - Para camiones de más de 3.5 toneladas, que podrían llevar cualquiera de los tipos de carga que se requiere en este cálculo, el combustible con mayor factor de emisión según [24] es el gasóleo con $0,768 \frac{\text{kg}CO_2}{\text{km}}$.
 - Maquinaria: $150 \text{ km} \cdot 0,768 \frac{\text{kg}CO_2}{\text{km}} \cdot 5 \text{ transportes} = 0,576 \text{ ton}CO_2$
 - Equipo auxiliar: $800 \text{ km} \cdot 0,768 \frac{\text{kg}CO_2}{\text{km}} \cdot 2 \text{ transportes} = 1,229 \text{ ton}CO_2$

En total se obtiene una huella de carbono por emisiones indirectas de GEI de $27,055 \text{ ton}CO_2$.

Por la potencia eléctrica instalada, se producen $494,22 \frac{MWh}{año}$, considerando que el MWh de la red en España produce un promedio de $0,2 \frac{tonCO_2}{MWh}$. Con ambos datos se obtiene que en el primer año se evitarían 98,84 toneladas de CO_2 , superando la huella de carbono emitida por el transporte en los primeros 4 meses de funcionamiento del equipo y contribuyendo durante el resto de vida útil.

6.4. INDICADORES DE RENTABILIDAD

Para los siguientes cálculos se considera lo siguiente:

- Coste medio del kWh para particular: 0,10€ .
- Vida útil del sistema: 20 años.
- Tasa de descuento: 5 %.

Año	0	1	2	3	4	5
CAPEX	-277530					
OPEX		-1640	-1640	-1640	-1640	-1640
Energía		494.215	494.215	494.215	494.215	494.215
Flujo caja	-277530	47781.5475	47781.547	47781.547	47781.547	47781.547

Tabla 6.4.1: Flujos de caja iniciales

6.4.1. VAN

El Valor Actualizado Neto es un método de valoración de inversiones que calcula el valor de todos los flujos de caja futuros actualizados a fecha inicial, restándoles la inversión inicial sin actualizar. Los flujos de caja en este caso son los ahorros obtenidos por el sistema:

$$VAN = \sum_{t=0}^n \frac{FC_t}{(1+k)^t} - I_0 = 302794\text{€} \quad (6.4.1)$$

6.4.2. TIR

La Tasa Interna de Rentabilidad es la tasa de rendimiento que genera el proyecto, se corresponde con el valor de tasa de descuento que anula el VAN.

$$TIR = 16,4\% \quad (6.4.2)$$

6.4.3. Payback Simple

Es el período transcurrido necesario para que los flujos de caja superen la inversión inicial (sin considerar valores actualizados).

$$PB = \frac{I_0}{FC_1} = 5,8 \text{ años} \quad (6.4.3)$$

Capítulo 7

Conclusiones

En este último capítulo se repasan los resultados y cálculos de los anteriores, realizando una reflexión sobre los detalles que se puedan ampliar o mejorar de cara a implementación. Por último, se exponen los conocimientos adquiridos y la relación con el máster y el ejercicio de la actividad como ingeniero industrial.

7.1. TECNOLOGÍA

El modelo subrogado permite conocer con relativa precisión el campo de temperatura y de tensiones del sistema. Esto permite no tener que realizar taladros en el CTES para colocar termopares, eso afectaría a la eficiencia del equipo. Además, la predicción permite modificar el lazo de control de forma que se optimice la absorción de calor o se eviten sobretensiones.

Los inconvenientes más destacables son los fallos acumulativos que presenta, en algún momento se tiene que utilizar algún sistema indirecto para tomar medidas y que el cálculo se reinicie sobre datos más precisos. Para ello posiblemente el mejor sistema sea midiendo la temperatura del agua que esté tomando calor de la superficie del CTES junto con la del HTF que circula por el interior. Con esas medidas debería ser sencillo ajustar la temperatura del interior con funciones de ML.

Por otra parte, las mejoras en tiempo de cómputo no solo en temperatura sino en tensiones daría una información útil para la capacidad y el mantenimiento del sistema. Con capacidad se hace referencia a la cantidad de energía no solo que está aportando en el momento sino que será capaz de aportar durante el ciclo, de forma que si en algún caso estuviera infradimensionado se pudiera calentar agua en adelante o si estuviera sobredimensionada, se almacenase ACS para evitar sobrepresiones. Y con mantenimiento, un inconveniente que no se ha tratado en profundidad es que el cemento es un material frágil, no se ha tratado porque si se puede conocer el ciclo de carga del sistema con previsión se pueden evitar los ciclos en los que se supere el umbral además de la posibilidad de programar *miners* que acumulen los ciclos previniendo las roturas por fatiga.

7.2. INSTALACIÓN

Se trata de un sistema ya conocido e implementado con éxito en estaciones de generación a gran escala al que se le añade un sistema extra que complementa y aprovecha los excesos de producción de energía que pueda producirse. La integración de un segundo lazo de control con *bypass*, control por temperatura y bombas de relativamente alto caudal aumentan la complejidad y requiere medidas extra de precaución.

Los cálculos para conocer la capacidad de dimensionar el sistema a las viviendas a las que pueda dar suministro son iterativos y han requerido reiterados intentos para obtener resultados plausibles y coherentes con el resto de cálculos y el estado actual de este tipo de sistemas. Indicar que se han realizado estos cálculos para servir de apoyo a un bloque de pisos de tamaño mediano, correspondiente al tipo de edificio residencial más común, pero se añade también el cálculo para un lazo estándar, ya que se conoce que es el método más eficiente de integrar estos bloques de generación.

El último cálculo, referido a la energía disponible para el CTES, es necesario señalar que al igual que en el resto, no es sencillo ni funcional trabajar en términos de energía aunque se intente inicialmente. Los consumos de energía pueden variar considerablemente, por lo que resulta más fiable utilizar registros estadísticos de consumo. Además, no se ha querido realizar ese cálculo tan en detalle, pero este sistema convendría que estuviera acompañado de depósitos en las viviendas, si bien de menor tamaño que los utilizados actualmente, para que el suministro no dependa del tiempo de envío desde la zona de producción de ACS hasta la vivienda.

7.3. PRESUPUESTO

El propósito de dicho capítulo es la viabilidad del equipo, el cual por los indicadores manejados se muestra como perfectamente viable. A la vista de los cálculos, podría considerarse que el CTES no se está incluyendo en los estudios al no considerarse el ACS, sin embargo, el ahorro en este caso sería no hacer un estudio detallado del sobre-coste de calentar el agua mediante energía eléctrica, sino que se tiene ya un sistema que garantiza ese suministro.

El estudio del CAPEX y OPEX ha sido realizando consultas y estimando en función a proyectos ya establecidos. Existe la posibilidad de mejorar los costes a través de economías de escala en el caso de realizar este proyecto de forma masiva ya sea acompañando a constructoras que realicen bloques residenciales o implantándose en industrias con grandes consumos eléctricos.

Si bien se tiene rentabilidad aceptable para el proyecto, los indicadores de mayor interés son el LCOE, LCOS y la huella de carbono. El LCOE está en el rango de centrales de producción hidroeléctricas, no las más eficientes, pero en el rango, por lo que se puede considerar que se trataría de un sistema que durante su funcionamiento es considerablemente eficiente. El LCOS es algo mayor, pero se encuentra también en un rango aceptable debido al bajo coste de implantación, que era la razón de la propuesta. Por otro lado, la reducción de la huella de carbono al final de su vida útil sería de alrededor de 1900 toneladas de CO_2 .

7.4. CONSIDERACIONES FINALES

Lo primero de la reflexión final es indicar los conocimientos del máster que han sido necesarios para la realización del trabajo:

- Ingeniería Estructural: la base de conocimientos de elementos finitos ha sido ampliada con el uso de modelos complejos para la simulación de elementos no necesariamente estructurales.
- Generación eléctrica: toda la generación de electricidad a través del sistema CCP está basado en los conocimientos adquiridos ahí.
- Ingeniería química: tanto los cálculos entálpicos y de combinación de elementos son propios de este sector y ha requerido repasar sus bases. Además, esta materia tiene especial interés por los diagramas P&ID, y realizar uno ha sido completamente necesario para el correcto dimensionado y presupuesto durante el proyecto.
- Financiación de la Actividad Industrial y Tecnológica: en esta materia, se adquirieron las bases para llevar a cabo el análisis económico del proyecto. Se adquirieron conocimientos para aplicar conceptos como CAPEX, OPEX y así como el estudio de la rentabilidad de una inversión.

Además, se ha realizado un estudio sobre los datos de consumo y de vivienda en España con los datos más actualizados disponibles en INE. Es una información relevante y de interés de cara a cualquier tipo de proyecto cuyo cliente final sean los residentes.

También señalar los aprendizajes realizados en materia de RRNN y programación con EEFF. Parte de estos conocimientos ya eran propios por otros estudios realizados, pero ha sido necesario profundizar y mejorar dichos conocimientos. Referido a RRNN, hubo una propuesta de realizar las simulaciones a través de PINN (*Physical Informed Neural Networks*), pero era un estudio demasiado ambicioso para el alcance del TFM, por lo que queda pendiente para próximos estudiantes una vez esa tecnología esté más asentada y extendida.

Finalmente, se consideran cubiertos los objetivos del estudio con resultados prometedores. Si bien hay mucha ingeniería de detalle que sería necesaria para una implementación correcta de todos los equipos y su interconexión tal y como se plantea en este trabajo, se prevee que las posibles ventajas en materia medioambiental y de autosuficiencia de los usuarios es suficiente para sobrellevar ese esfuerzo y coste adicional.

Capítulo 8

Anexos

8.1. CÓDIGO UTILIZADO

8.1.1. Cálculo de propiedades

```
46     import numpy as np
47
48     def get_kg(E, nu):
49         K = E / (3 * (1 - 2 * nu))
50         G = E / (2 * (1 + nu))
51         return K, G
52
53     def get_Enu(K, G):
54         E = G*(3*K+G)/(3*K+G)
55         nu = (3*K-2*G)/(6*K+2*G)
56         return E, nu
57
58     def get_isotropic_stiffness(E, nu):
59         """Retorna la matriz de rigidez (C) en notación de
60         Voigt (6x6)"""
61         lam = (E * nu) / ((1 + nu) * (1 - 2 * nu))
62         mu = E / (2 * (1 + nu))
63
64         C = np.zeros((6, 6))
65         C[0:3, 0:3] = lam
66         np.fill_diagonal(C[0:3, 0:3], lam + 2 * mu)
67         np.fill_diagonal(C[3:6, 3:6], mu)
68         return C
69
70     def get_eshelby_prolate(nu_m, aspect_ratio):
71         """
72         Tensor de Eshelby simplificado para inclusiones cilíndricas (fibras)
73         Para fibras largas (aspect_ratio >> 1), se aproxima a cilindro infinito.
74         """
```

```

74     S = np.zeros((6, 6))
75     # Aproximación para fibra larga alineada en eje 1
76     g = aspect_ratio
77     # Componentes simplificadas para cilindro
78     S[0,0] = 0 # En el eje de la fibra la deformación es
impuesta por la matriz
79     S[1,1] = S[2,2] = (5 - 4 * nu_m) / (8 * (1 - nu_m))
80     S[0,1] = S[0,2] = 0
81     S[1,0] = S[2,0] = nu_m / (2 * (1 - nu_m))
82     S[1,2] = S[2,1] = (4 * nu_m - 1) / (8 * (1 - nu_m))
83     S[3,3] = (1 - 2 * nu_m) / (4 * (1 - nu_m))
84     S[4,4] = S[5,5] = 0.25
85     return S
86
87     def mori_tanaka_composite(E_m, nu_m, E_f, nu_f, V_f,
aspect_ratio=100):
88         # 1. Matrices de rigidez
89         C_m = get_isotropic_stiffness(E_m, nu_m)
90         C_f = get_isotropic_stiffness(E_f, nu_f)
91         I4 = np.eye(6)
92
93         # 2. Tensor de Eshelby
94         S = get_eshelby_prolate(nu_m, aspect_ratio)
95
96         # 3. Tensor de concentración Diluido (A_dil)
97         # A_dil = [I + S : C_m^-1 : (C_f - C_m)]^-1
98         term = S @ np.linalg.inv(C_m) @ (C_f - C_m)
99         A_dil = np.linalg.inv(I4 + term)
100
101         # 4. Tensor de concentración Mori-Tanaka (A_MT)
102         # A_MT = A_dil : [(1-V_f)I + V_f*A_dil]^-1
103         A_MT = A_dil @ np.linalg.inv((1 - V_f) * I4 + V_f *
A_dil)
104
105         # 5. Rigidez Efectiva
106         C_eff = C_m + V_f * (C_f - C_m) @ A_MT
107         return C_eff
108
109
110
111     def orientation_average_3d(C_aligned):
112         """
113         Realiza el promediado de orientación 3D (aleatorio en
el espacio)
114         de una matriz de rigidez alineada de 6x6.
115
116         Parámetros:
117         C_aligned : numpy.ndarray
118         Matriz de rigidez (6x6) del compuesto unidireccional (p
ej. de Mori-Tanaka).
119
120         Retorna:
121         C_random : numpy.ndarray
122         Nueva matriz de rigidez (6x6) global isotrópica.
123         E_3d, nu_3d : float

```

```

124     Módulo de Young y coeficiente de Poisson globales.
125     """
126     # 1. Extraer los componentes de la matriz alineada
127     C11 = C_aligned[0, 0]
128     C22 = C_aligned[1, 1]
129     C33 = C_aligned[2, 2]
130
131     C12 = C_aligned[0, 1]
132     C23 = C_aligned[1, 2]
133     C13 = C_aligned[0, 2]
134
135     C44 = C_aligned[3, 3]
136     C55 = C_aligned[4, 4]
137     C66 = C_aligned[5, 5]
138
139     # 2. Calcular los invariantes isotrópicos:
140     Compresibilidad (K) y Corte (G)
141     K_3d = (C11 + C22 + C33 + 2 * (C12 + C23 + C13)) / 9.0
142     G_3d = (C11 + C22 + C33 - C12 - C23 - C13 + 3 * (C44 +
143     C55 + C66)) / 15.0
144
145     # 3. Calcular las propiedades de ingeniería globales (
146     Young y Poisson)
147     E_3d = (9 * K_3d * G_3d) / (3 * K_3d + G_3d)
148     nu_3d = (3 * K_3d - 2 * G_3d) / (2 * (3 * K_3d + G_3d))
149
150     # 4. Reconstruir la nueva matriz de rigidez global (
151     isotrópica)
152     lam = (E_3d * nu_3d) / ((1 + nu_3d) * (1 - 2 * nu_3d))
153     mu = G_3d # El módulo de corte es directamente G
154
155     C_random = np.zeros((6, 6))
156     C_random[0:3, 0:3] = lam
157     np.fill_diagonal(C_random[0:3, 0:3], lam + 2 * mu)
158     np.fill_diagonal(C_random[3:6, 3:6], mu)
159
160     return C_random, E_3d, nu_3d
161
162     def thermal_expansion_homogenization(alpha_m, alpha_f,
163     K_m, K_f, K_c):
164         """
165         Implementación de la Relación de Levin (Ec. 5)
166         K_m, K_f: Bulk modulus de matriz y fibra
167         K_c: Bulk modulus del compuesto (calculado previamente
168         con Mori-Tanaka elástico)
169         """
170         # Evitar división por cero si las rigideces son iguales
171         if abs(K_f - K_m) < 1e-6:
172             return alpha_m
173
174         # Relación de Levin: (af - ac)/(am - ac) = (1/Kf - 1/Kc
175         )/(1/Km - 1/Kc)
176         term_K = (1/K_f - 1/K_c) / (1/K_m - 1/K_c)
177         alpha_c = (alpha_f - term_K * alpha_m) / (1 - term_K)

```

```

172     return alpha_c
173
174
175     def thermal_complete_homogenization(k_m, k_f, V_f, cp_m
176 , cp_f, rho_m, rho_f,
177     alpha_m, alpha_f, K_m, K_f, K_c, aspect_ratio):
178         """
179         Función integral para propiedades térmicas en Abaqus
180         """
181         # 1. Densidad efectiva (Media ponderada por volumen)
182         rho_eff = rho_f * V_f + rho_m * (1 - V_f)
183
184         # 2. Calor específico (Media ponderada por masa)
185         m_f = rho_f * V_f
186         m_m = rho_m * (1 - V_f)
187         cp_eff = (cp_f * m_f + cp_m * m_m) / (m_f + m_m)
188
189         # 3. Conductividad térmica (Eje axial)
190         # Usamos la lógica de tu Ec. (6)
191         k_eff_axial = k_m + (V_f * (k_f - k_m)) / (1 + (1 - V_f
192 ) * ((k_f - k_m) / (3 * k_m)))
193
194         # 4. Coeficiente de expansión térmica (Relación de
195         Levin - Ec. 5)
196         alpha_eff = thermal_expansion_homogenization(alpha_m,
197     alpha_f, K_m, K_f, K_c)
198
199         return rho_eff, k_eff_axial, cp_eff, alpha_eff
200
201
202     #este se puede utilizar para inclusiones esféricas ==
203     ilmenita
204     def calcular_mori_tanaka_escalar(E_m, nu_m, E_f, nu_f,
205     V_f):
206         """
207         Calcula las propiedades efectivas usando las ecuaciones
208         de K y G
209         proporcionadas por el usuario.
210         """
211
212         # 1. Convertir propiedades de matriz (m) y fibra (f) a
213         K y G
214
215         K_m, G_m = get_kg(E_m, nu_m)
216         K_f, G_f = get_kg(E_f, nu_f)
217
218         # 2. Calcular factor auxiliar f_1 (según tu fórmula
219         proporcionada)
220         # f1 = G_m * (9*K_m + 8*G_m) / (6 * (K_m + 2*G_m))
221         # Nota: Esta es una simplificación común para el té
222         rmino de concentración
223         f1 = (G_m * (9 * K_m + 8 * G_m)) / (6 * (K_m + 2 * G_m)
224 )

```

```

216
217     # 3. Aplicar Ecuación (2): Módulo de Compresibilidad
Efectivo (K)
218     #  $K = K_m + (V_f * (K_f - K_m)) / (1 + (1 - V_f) * ((K_f - K_m) / (K_m + 4/3 * G_m)))$ 
219     # Nota: He ajustado a la forma estándar de MT para K
220      $K_{eff} = K_m + (V_f * (K_f - K_m)) / (1 + (1 - V_f) * ((K_f - K_m) / (K_m + (4/3) * G_m)))$ 
221
222     # 4. Aplicar Ecuación (3): Módulo de Cizalladura
Efectivo (G)
223     #  $G = G_m + (V_f * (G_f - G_m)) / (1 + (1 - V_f) * ((G_f - G_m) / (G_m + f1)))$ 
224      $G_{eff} = G_m + (V_f * (G_f - G_m)) / (1 + (1 - V_f) * ((G_f - G_m) / (G_m + f1)))$ 
225
226     # 5. Ecuación (1): Volver a E y nu efectivos
227      $E_{eff} = (9 * K_{eff} * G_{eff}) / (3 * K_{eff} + G_{eff})$ 
228      $\nu_{eff} = (3 * K_{eff} - 2 * G_{eff}) / (2 * (3 * K_{eff} + G_{eff}))$ 
229
230     return E_eff, nu_eff, K_eff, G_eff
231
232
233     def calcular_propiedades_termicas_MT(alpha_m, alpha_p,
k_m, k_p, V_p, K_m, K_p, K_c):
234         """
235         Implementación de las ecuaciones (5) y (6) de la imagen
236
237         p = partícula/fibra (f en tu imagen, usamos 'p' para no
confundir con k_f de conductividad)
238         m = matriz (cemento)
239         c = compuesto (resultado final)
240
241         # Fracción volumétrica de la matriz ( $V_m = 1 - V_p$ )
242          $V_m = 1 - V_p$ 
243
244         # --- ECUACIÓN (6): Conductividad Térmica del Compuesto
( $K_c$  en la imagen) ---
245         # Reordenando la Ec (6) para despejar  $K_c$ :
246         # La Ec (6) es una forma de Mori-Tanaka/Hashin-
Shtrikman para conductividad.
247
248         # Usamos una función de resolución para encontrar  $K_c$ 
ya que está en ambos lados
249         # No obstante, para inclusiones esféricas, la solución
explícita de  $K_c$  es:
250          $A = V_p * (k_p - k_m)$ 
251          $B = 1 + V_m * ((k_p - k_m) / (3 * k_m))$ 
252          $k_c = k_m + A / B$ 
253
254         # --- ECUACIÓN (5): Coeficiente de Expansión Térmica ( $\alpha_c$ ) ---
255         #  $\alpha_c$  es el valor que buscamos.

```

```

256     # La fórmula es: (alpha_f - alpha_c) / (alpha_m -
alpha_c) = (1/K_f - 1/K_c) / (1/K_m - 1/K_c)
257     # Donde K son los Bulk Modulus elásticos.
258     #OK
259     # Despejando alpha_c de la relación de Levin:
260     term_K = (1/K_p - 1/K_c) / (1/K_m - 1/K_c)
261     alpha_c = (alpha_p - term_K * alpha_m) / (1 - term_K)
262
263     return k_c, alpha_c
264
265     def get_hormigon_fibra(vol_fibra, E_h=30000, nu_h=0.2,
k_hormigon=0.28, cp_hormigon=880e6, rho_hormigon=2.5e-9,
alpha_hormigon=5e-6):
266
267         # --- VALORES DEFINIDOS ---
268
269
270         K_h, G_h = get_kg(E_h, nu_h)
271
272         E_basalto = 100000.0      # MPa
273         nu_basalto = 0.22
274         K_basalto, G_basalto = get_kg(E_basalto, nu_basalto)
275
276
277         L_f = 6
278         D_f = 0.018
279         AR = L_f/D_f              # Relación de aspecto L/D
280
281         C_final = mori_tanaka_composite(E_h, nu_h, E_basalto,
nu_basalto, vol_fibra, AR)
282
283         S_final = np.linalg.inv(C_final)
284         # Extracción de propiedades efectivas (E en eje 1)
285
286
287         C_random, E_final, nu_final = orientation_average_3d(
C_final)
288
289         print('### HORMIGON+FIBRA ###\n')
290         print("#"*15)
291         K_final, G_final = get_kg(E_final, nu_final)
292
293
294
295         k_basalto = 0.035          # mW/(mm*K)
296         cp_basalto = 860e6         # mJ/(t*K)
297         rho_basalto = 2.75e-9      # t/mm^3
298         alpha_basalto = 5e-6       # 1/K
299
300         # Ejecución
301         rho_eff, k_eff_axial, cp_eff, alpha_eff =
thermal_complete_homogenization(k_hormigon, k_basalto,
vol_fibra,
302         cp_hormigon, cp_basalto,
303         rho_hormigon, rho_basalto,

```

```

304     alpha_hormigon, alpha_basalto,
305     K_h, K_basalto,
306     K_final,
307     AR)
308
309
310     return {"Young": E_final, "Poisson": nu_final, "densidad":
rho_eff, "cp": cp_eff, "conductividad": k_eff_axial, "
coefi_exp": alpha_eff}
311
312
313     #####
314     def get_hormigon_ilmenita(V_i):
315     # --- VALORES TÉRMICOS (Sistema mm-MPa-Toneladas) ---
316     E_h = 30000.0      # MPa Hormigón
317     nu_h = 0.2
318     K_h, G_h = get_kg(E_h, nu_h)
319     k_hormigon = 0.28      # mW/(mm*K) equivalente a W
/(m*K)
320     cp_hormigon = 880e6      # mJ/(t*K)
321     rho_hormigon = 2.5e-9      # t/mm^3
322     alpha_hormigon = 5e-6      # 1/K
323     #ILMENITA
324
325     rho_il = 4.79e-9 #tn/mm^3
326     rho_final = rho_hormigon*(1-V_i)+rho_il*(V_i)
327     #PROPAGACIÓN DE ONDAS SÍSMICAS, ESTIMACIÓN
328     V_s = 4.2e+6 #mm/s
329     V_p = 7.7e+6 #mm/s
330     G_i = rho_il * V_s**2
331     K_i = rho_il * (V_p**2 - (4/3)*V_s**2)
332     E_i, nu_i = get_Enu(K_i, G_i)
333     # Ejecución
334     E_res, nu_res, K_res, G_res =
calcular_mori_tanaka_escalar(E_h, nu_h, E_i, nu_i, V_i)
335
336     #CALOR ESPECÍFICO
337     cp_TiO2 = 0.7 #J/gK
338     cp_FeO = 0.66 #J/gK
339     Fe_Masa = 0.47 #tanto por 1
340     cp_i = cp_FeO*Fe_Masa + cp_TiO2*(1-Fe_Masa)
341     cp_final = cp_i*V_i+cp_hormigon*(1-V_i)
342
343     Cte_Gruneissen_i = 1.21
344     alpha_i = Cte_Gruneissen_i*cp_i * rho_il / (3*K_i)
345     k_i = 3 #W/mK (ESTIMACIÓN)
346
347
348     k_final, alpha_final = calcular_propiedades_termicas_MT
(
349     alpha_hormigon, alpha_i,
350     k_hormigon, k_i,
351     V_i,
352     K_h, K_i, K_res
353 )

```



```

354
355     print("#"*15)
356     print('\n### HORMIGON+ILMENITA ###\n')
357     print("#"*15)
358
359     print(f"Volumen Ilmenita: {V_i*100}%")
360
361
362     # --- DATOS PARA EL CÁLCULO ---
363     # Propiedades Matriz (Hormigón)
364     alpha_matriz = 1.0e-5 # 1/K
365     k_matriz = 1.5 # W/mK o mW/mmK
366     K_matriz = 16666.6 # MPa (Bulk Modulus calculado
antes)
367
368     # Propiedades Fibra (Vidrio)
369     # 1/K
370     k_fibra = 1.0 # W/mK o mW/mmK
371     K_fibra = 42857.1 # MPa (Bulk Modulus calculado
antes)
372
373     # Propiedades Compuesto (Calculadas en el paso anterior
)
374     V_fibra = 0.03
375     K_compuesto = 17165.0 # MPa (Resultado de K_eff del
script anterior)
376
377     # Ejecución
378
379
380     return {"Young":E_res,"Poisson":nu_res, "densidad":
rho_final,"cp":cp_final,"conductividad":k_final,"
coefi_exp":alpha_final}
381
382     def print_results(dict_material):
383     print('RESULTADOS MECÁNICOS')
384     print(f"Módulo de Young Efectivo (E): {dict_material['
Young']:.4f} MPa")
385     print(f"Coeficiente de Poisson (nu): {dict_material['
Poisson']:.4f}")
386
387     print("RESULTADOS TÉRMICOS:")
388     print(f"Densidad {dict_material['densidad']:.4e} t/mm^3
")
389     print(f"Conductividad Térmica Efectiva (K_c): {
dict_material['conductividad']:.4f} W/m*K")
390
391     print(f"Calor Específico {dict_material['cp']:.4f} mJ/
tK")
392     print(f"Coef. Expansión Térmica (alpha_c): {
dict_material['coefi_exp']:.4e} 1/K")
393
394
395     def main():
396

```

```
397     dic_H_F = get_hormigon_fibra(0.001)
398     print_results(dic_H_F)
399     dic_H_I = get_hormigon_ilmenita(0.05)
400     print_results(dic_H_I)
401     print("AHORA SE CALCULA PARA LA SUMA DE AMBAS
INCLUSIONES AUNQUE PONGA OTRA COSA")
402
403     dic_H_F_I=get_hormigon_fibra(0.001,
404     dic_H_I['Young'],
405     dic_H_I['Poisson'],
406     dic_H_I['conductividad'],
407     dic_H_I['cp'],
408     dic_H_I['densidad'],
409     dic_H_I['coefi_exp'])
410     print_results(dic_H_F_I)
411
```

8.1.2. Cálculo automático en Abaqus

```

412 # -*- coding: utf-8 -*-
413 from abaqus import *
414 from abaqusConstants import *
415 import job
416 import interaction
417 from odbAccess import *
418 import os
419 import csv
420 import re
421 from collections import defaultdict
422
423 print("INICIO DE EJECUCIÓN")
424 # Asegurate de que el .cae existe o comenta esta linea
425 # si ya lo tienes abierto
426 openMdb(pathName="CTES_R_ThermoMeca.cae")
427 print("LECTURA DE .cae CORRECTA")
428
429 #####
430 # # # # #
431 # # # # #
432 # # # # #
433 #####
434
435 nombre_modelos = ['Model-B', 'Model-FV001', 'Model-IL05',
436 'Model-IL05FV001']
437
438 nombre_subrutina = "DFLUX.for"
439 ruta_subrutina = r'C:\temp\' + str(nombre_subrutina)
440
441 nombre_carga = 'FlujoCalorS-M'
442 superficie = 'Surf-Int-St'
443
444 temp_revisar = range(200, 404, 10)
445
446 saltos_temp = [2, 5]
447 velocidades = [1, 2]
448 #velocidades = [1]
449 nombre_csv = 'TM_cut_results'
450 name_save = "HeatTransferMec"
451
452 #####
453 # # # # #
454 # # # # #
455 # # # # #
456 #####
457
458 def calc_datos(temperatura):

```

```

458     A5=temperatura
459     den = -0.00135*A5*A5 - 0.285*A5 + 1024
460     VC=0.00000000000505*A5*A5 - 0.000000004135*A5 + 0.000001057
461     VD=0.0000000048268*A5*A5 - 0.00000413616*A5 + 0.00102858
462     CE=0.0000024*A5*A5 + 0.00146*A5 + 1.66
463     Dif= -0.00000000012962*A5*A5 + 0.00000000429*A5
+0.000073852
464     Cond= -0.0000005*A5*A5 + 0.00015*A5 + 0.12
465     datos = [den, VC, VD, CE, Dif, Cond]
466     return datos
467
468     def calc_coefi (temperatura, datos, velocidad = 2, L=0.25,
D=0.05, mm=False):
469         datos = calc_datos(temperatura)
470         Re = velocidad*L/datos[1]
471         Pr = datos[1]/datos[4]
472         Nu = 0.23 * (Re**0.8 + Pr**0.4)
473         h = Nu*datos[-1]/L
474         if mm:
475             h=h*10**-6
476         return h
477
478     def actualizar_campo_analitico(modelo, nombre_campo,
temperatura1, temperatura2, velocidad=2, L=0.250):
479         coeficiente1 = calc_coefi(temperatura1, calc_datos(
temperatura1), velocidad)
480         coeficiente2 = calc_coefi(temperatura2, calc_datos(
temperatura2), velocidad)
481
482
483         calc_pendiente = coeficiente2 - coeficiente1/(L)
484         pendiente_str = int(calc_pendiente*100)
485         pendiente_str = str(pendiente_str)
486
487         formula = "{} + {} * Z".format(float(coeficiente1),
float(calc_pendiente))
488
489         if nombre_campo in modelo.analyticalFields.keys():
490             del modelo.analyticalFields[nombre_campo]
491
492         modelo.ExpressionField(name=nombre_campo,
expression=formula,
493         localCsys=None,
494         description='H variable...')
495         return formula, pendiente_str
496
497
498
499     def modificar_subrutina_fortran(ruta_archivo,
500     h1, h2,
501     t1, t2,
502     ymin=0, ymax=0.250,
503     mm=False):
504
505     if mm:
506         ymin *= 1000

```

```

507     ymax *= 1000
508
509     with open(ruta_archivo, 'r') as f:
510         contenido = f.read()
511
512     def fmt(valor):
513         return f"{valor:.6E}".replace("E", "D")
514
515     reemplazos = {
516         "H1": fmt(h1),
517         "H2": fmt(h2),
518         "T1": fmt(t1),
519         "T2": fmt(t2),
520         "YMIN": fmt(ymin),
521         "YMAX": fmt(ymax),
522     }
523
524     for var, val in reemplazos.items():
525         contenido = re.sub(rf"{var}\s*=.*",
526             f"{var} = {val}",
527             contenido)
528
529     with open(ruta_archivo, 'w') as f:
530         f.write(contenido)
531
532     ###      #####      ###      ###      #      #      #####
533     #      #      #      #      #      #      #      #      #
534     #####      #####      #####      #      #      #      #      ###
535     #      #      #      #      #      #      #      ##      #      #
536     #      #      #####      #      #      #####      ###      #####
537
538     def get_nodal_averaged(field, node_set, variable='mises
539     '):
540         """
541         Extrapola un campo de integración a nodos y promedia
542         los valores que llegan a cada nodo desde elementos
543         adyacentes.
544         variable puede ser:
545         'mises', 'maxPrincipal', 'minPrincipal', 'data', '
546         magnitude'
547         Devuelve dict: {nodeLabel: valor_escalar_promediado}
548         """
549         if field is None:
550             return {}
551
552         try:
553             subset = field.getSubset(region=node_set, position=
554             ELEMENT_NODAL)
555         except:
556             return {}
557
558         # Acumular valores por nodo
559         node_values = defaultdict(list)
560         for val in subset.values:

```

```
558     if variable == 'mises' and hasattr(val, 'mises'):
559         node_values[val.nodeLabel].append(val.mises)
560
561     elif variable == 'maxPrincipal' and hasattr(val, '
maxPrincipal'):
562         node_values[val.nodeLabel].append(val.maxPrincipal)
563
564     elif variable == 'minPrincipal' and hasattr(val, '
minPrincipal'):
565         node_values[val.nodeLabel].append(val.minPrincipal)
566
567     elif variable == 'magnitude' and hasattr(val, '
magnitude'):
568         node_values[val.nodeLabel].append(val.magnitude)
569
570     elif hasattr(val, 'data'):
571         # Escalar genérico
572         node_values[val.nodeLabel].append(val.data)
573
574     # Promediar
575     return {node: sum(vals)/len(vals) for node, vals in
node_values.items()}
576
577
578     def Extr_ODB(csv_name, step_name, node_set_name,
odb_name):
579         error_ocurrido = False
580         with open(f'notas_{odb_name}.txt', 'w') as f:
581             print("INICIO POSTPROCESADO",file = f)
582
583         with open(csv_name, 'w') as csv_file:
584
585             writer = csv.writer(csv_file)
586
587             writer.writerow([
588                 'JobName',
589                 'Frame',
590                 'Time',
591                 'NodeLabel',
592                 'X',
593                 'Y',
594                 'Z',
595                 'NT11',
596                 'Stress_Max',
597                 'Stress_Min',
598                 'Strain'
599             ])
600
601             print(f"Procesando: {odb_name}",file = f)
602
603             try:
604
605                 odb = openOdb(odb_name)
606
607                 # Tomar último step disponible
```

```

608 step_name= odb.steps.keys()[-1]
609 step = odb.steps[step_name]
610
611 if len(step.frames) == 0:
612     print(f"No hay frames en: {odb_name}",file = f)
613     odb.close()
614     return
615
616 # Buscar el node set
617 if node_set_name not in odb.rootAssembly.nodeSets.keys
():
618     print(f"Node set no encontrado en{odb_name}",file = f)
619     print(f"Disponibles:{ odb.rootAssembly.nodeSets.keys()}
",file = f)
620     odb.close()
621     return
622
623 node_set = odb.rootAssembly.nodeSets[node_set_name]
624 except Exception as e:
625     print(f"Error procesando la lectura de {odb_name} : {
str(e)}",file = f)
626     error_ocurrido = True
627     try:
628         # Diccionario NodeLabel -> Coordenadas
629         coords = {node.label: node.coordinates
630         for node_array in node_set.nodes
631         for node in node_array}
632     except Exception as e:
633         print(f"Error procesando los nodos de {odb_name} : {str
(e)}",file = f)
634         error_ocurrido = True
635         ## Recorrer TODOS los frames
636
637     try:
638         for i, frame in enumerate(step.frames):
639
640             if 'NT11' not in frame.fieldOutputs.keys():
641                 continue
642
643             time = frame.frameValue
644
645             temp_field = frame.fieldOutputs['NT11']
646
647             # Campos adicionales (corregido)
648             stress_field = frame.fieldOutputs['S'] if 'S' in frame.
fieldOutputs.keys() else None
649             strain_field = frame.fieldOutputs['E'] if 'E' in frame.
fieldOutputs.keys() else None # o 'LE'
650
651             values = temp_field.getSubset(region=node_set).values
652
653             # En lugar de getSubset con NODAL:
654             try:
655                 stress_dict_max = get_nodal_averaged(stress_field,
node_set,variable='maxPrincipal')

```

```
656     stress_dict_min = get_nodal_averaged(stress_field,
657     node_set, variable='minPrincipal')
658     except Exception as e:
659         print(f"Error procesando las tensiones {str(e)}", file =
660         f)
661         error_ocurrido = True
662         try:
663             strain_dict = get_nodal_averaged(strain_field, node_set
664             , variable = 'mises')
665         except Exception as e:
666             print(f"Error procesando las deformaciones : {str(e)}",
667             file = f)
668             error_ocurrido = True
669             try:
670                 for val in values:
671
672                     node = val.nodeLabel
673                     x, y, z = coords[node]
674
675                     # Tensiones máximas y mínimas
676
677                     stress_max = stress_dict_max.get(node) if node in
678                     stress_dict_max else None
679                     stress_min = stress_dict_min.get(node) if node in
680                     stress_dict_min else None
681
682                     # Deformación equivalente
683                     strain = strain_dict.get(node) if node in strain_dict
684                     else None
685
686                     writer.writerow([
687                     odb_name,
688                     i,
689                     time,
690                     node,
691                     x,
692                     y,
693                     z,
694                     val.data,
695                     stress_max,
696                     stress_min,
697                     strain
698                     ])
699                     except Exception as e:
700                         print(f"Error procesando la extracción de valores en {
701                         odb_name} : {str(e)}", file = f)
702                         error_ocurrido = True
703                         odb.close()
704
705                     except Exception as e:
706                         print(f"Error procesando el bucle de {odb_name} : {str(
707                         e)}", file = f)
708                         error_ocurrido = True
```



```

702     print("POSTPROCESADO FINALIZADO", file = f)
703     return error_ocurrido
704
705
706     #####      ###      #####      ###      #      #      ###      ###      ###
707     #      #
708     #      #      #      #      #      #      #      #      #      #      #
709     ##      #
710     #####      #      #####      #      #      #      #      #      #      #
711     #      #
712     #      #      #      #      #      #      #      #      #      #      #
713     #      #
714     #####      ##      #####      ###      ###      ###      ###      ###
715     #      #
716
717     borra_odb = True
718     for nombre_modelo in nombre_modelos:
719         my_model = mdb.models[nombre_modelo]
720         if nombre_carga not in my_model.loads.keys():
721             print("ERROR CRITICO: No encuentro la carga " +
722 nombre_carga)
723         else:
724
725             for temp in temp_revisar:
726                 borra_odb = True
727                 for salto in saltos_temp:
728                     for speed in velocidades:
729                         # 1. Calcular nuevos coeficientes
730                         #los saltos de temperatura deben ser proporcionales a 5
731                         K
732                         h_calculado_1 = calc_coefi(temp, calc_datos(temp),
733 velocidad=speed)
734
735                         temp2 = temp-salto
736                         h_calculado_2 = calc_coefi(temp2, calc_datos(temp2),
737 velocidad=speed)
738
739                         #with open(f"const_{temp}.txt", 'w') as fopen:
740                         #    fopen.write("Modificando subrutina para \n")
741                         #    fopen.write(f'h1: {h_calculado_1}\t\t\t h2: {
742 h_calculado_2}\nT1: {temp}\t\t\t T2: {temp2}\n Velocidad
743 {speed}')
744
745                         # 2. Modificar el archivo Fortran
746
747                         modificar_subrutina_fortran(ruta_subrutina,
748 h_calculado_1,h_calculado_2,
749 temp,temp2)
750
751                         # 3. Configurar carga
752                         my_model.SurfaceHeatFlux(
753                             name=nombre_carga,
754                             createStepName=name_save,
755                             region=my_model.rootAssembly-surfaces['Surf-DFLUXSteel'
756 ],
757                             magnitude=1.0,

```

```
745     distributionType=USER_DEFINED
746 )
747
748
749     # 4. Preparar y lanzar Job con la subrutina vinculada
750     job_name = f'{name_save}_T{int(temp)}_S{int(salto)}_V{
int(speed*10)}'
751     if job_name in mdb.jobs.keys():
752         del mdb.jobs[job_name]
753
754     my_job = mdb.Job(name=job_name,
755                     model=nombre_modelo,
756                     userSubroutine=nombre_subrutina) # <--- VINCULACIÓN
757
758     print("Enviando: " + job_name)
759     my_job.submit()
760     my_job.waitForCompletion()
761
762     # 5. Procesado de ODB
763
764     csv_name = f"{nombre_csv}_T{temp}_S{salto}_V{int(10*
speed)}_{nombre_modelo}.csv"
765     step_name = "HeatTransfRev"
766     node_set_name = "INT_NODES"
767     odb_name = f'{job_name}.odb'
768     error_ocurrido = Extr_ODB(csv_name, step_name,
node_set_name, odb_name)
769     if temp == 400 or temp == 300 or temp == 200:
770         borra_odb = False
771         if borra_odb and not error_ocurrido:
772             for i in os.listdir():
773                 if i.startswith(job_name) or i.startswith('notas_'+
job_name):
774                     os.remove(i)
775
776
777
778     print("Estudio completado.")
779
```

8.1.3. Scripts de la red neuronal

```

780
781
782     class GridDataset(torch.utils.data.Dataset):
783     def __init__(self, grids_dict):
784         """
785         Dataset personalizado para los grids extraídos.
786         Cada archivo procesado se convierte en una muestra
787         (sample).
788         """
789         self.filenamees = list(grids_dict.keys())
790         self.grids_dict = grids_dict
791
792     def __len__(self):
793         return len(self.filenamees)
794
795     def __getitem__(self, idx):
796         filename = self.filenamees[idx]
797         grid_array = self.grids_dict[filename]
798         grid_nt11 = grid_array["NT11"] # (T,
799         12, 12)
800         grid_Stress_Max = grid_array["Stress_Max"] # (T,
801         12, 12)
802         grid_Stress_Min = grid_array["Stress_Min"] # (T,
803         12, 12)
804         # Convertimos el numpy array a un tensor de PyTorch
805         .
806         # Shape esperado: (Num_Tiempos, Res_X, Res_Y), ej:
807         (1004, 12, 12)
808         # Esto puede actuar directamente como (Canales,
809         Alto, Ancho) para una Conv2D
810         grid_tensor = torch.tensor(grid_nt11,
811         grid_Stress_Max, grid_Stress_Min, dtype=torch.float32)
812
813         # Vamos a intentar sacar las etiquetas T, S, V del
814         nombre del archivo
815         # Ejemplo: cut_results_T200_S0_V10.csv
816
817         try:
818             nombre_sin_ext = filename.replace('.csv', '')
819             partes = nombre_sin_ext.split('_')
820             if partes[-1].contains('Model'):
821                 t_val = float(partes[-4].replace('T', ''))
822                 s_val = float(partes[-3].replace('S', ''))
823                 v_val = float(partes[-2].replace('V', ''))
824                 fiber = partes[-1].split("-")
825                 if fiber[-1]=='B':
826                     fv=0
827                     ilm=0
828                 elif fiber[-1]=='IL05':
829                     fv=0
830                     ilm=0.05
831                 elif fiber[-1]=='FV001':
832                     fv=0.001

```

```
824         ilm=0
825         elif fiber[-1]=='IL05FV001':
826             fv=0.001
827             ilm=0.05
828             etiquetas = torch.tensor([t_val, s_val, v_val, fv,
829                                     ilm], dtype=torch.float32)
830
831         else:
832             t_val = float(partes[-3].replace('T', ''))
833             s_val = float(partes[-2].replace('S', ''))
834             v_val = float(partes[-1].replace('V', ''))
835             etiquetas = torch.tensor([t_val, s_val, v_val],
836                                     dtype=torch.float32)
837         except Exception:
838             etiquetas = torch.zeros(5, dtype=torch.float32)
839
840         return grid_tensor, etiquetas
```

```

840
841
842
843     class ConditionedThermalDataset(torch.utils.data.
Dataset):
844         def __init__(self, grids_dict):
845             self.samples = []
846
847         for filename, grid_array in grids_dict.items():
848             num_tiempos = grid_array.shape[0] # Ej: 1004
849
850             # --- NUEVO: Extraer condiciones del nombre ---
851             # Ejemplo: cut_results_T200_S0_V10.csv
852             try:
853                 nombre_sin_ext = filename.replace('.csv', '')
854                 partes = nombre_sin_ext.split('_')
855                 if partes[-1].contains('Model'):
856                     t_val = float(partes[-4].replace('T', ''))
857                     s_val = float(partes[-3].replace('S', ''))
858                     v_val = float(partes[-2].replace('V', ''))
859                     fiber = partes[-1].split("-")
860                     if fiber[-1]=='B':
861                         fv=0
862                         ilm=0
863                     elif fiber[-1]=='IL05':
864                         fv=0
865                         ilm=0.05
866                     elif fiber[-1]=='FV001':
867                         fv=0.001
868                         ilm=0
869                     elif fiber[-1]=='IL05FV001':
870                         fv=0.001
871                         ilm=0.05
872                 # CRÍTICO: Normalizar T, S, V (ej. 0-1) para que
coincidan con NT11
873                 # Necesitarías los máximos/mínimos globales de tu
dataset.
874                 # Aquí usamos valores de ejemplo (T_max=400, S_max
=10, V_max=50).
875                 conditions = np.array([t_val/400.0, s_val/10.0,
v_val/50.0, fv, ilm], dtype=np.float32)
876
877             except Exception as e:
878                 print(f"Error parseando {filename}: {e}. Usando
condiciones cero.")
879                 conditions = np.zeros(5, dtype=np.float32)
880
881             # --- Crear muestras (t, t+1) ---
882             for t in range(num_tiempos - 1):
883                 self.samples.append({
884                     'input_grid': grid_array[t],           # Mapa t
885                     'target_grid': grid_array[t+1],         # Mapa t+1
886                     'conditions': conditions                 # T, S, V
normalizados

```

```
887         })
888
889         def __len__(self):
890             return len(self.samples)
891
892         def __getitem__(self, idx):
893             sample = self.samples[idx]
894
895             # Grid: (1, 12, 12)
896             x_grid = torch.tensor(sample['input_grid'], dtype=
torch.float32).unsqueeze(0)
897             # Target: (1, 12, 12)
898             y_grid = torch.tensor(sample['target_grid'], dtype=
torch.float32).unsqueeze(0)
899             # Conditions: (3,)
900             x_cond = torch.tensor(sample['conditions'], dtype=
torch.float32)
901
902             return x_grid, x_cond, y_grid
903
904
```

```

905         class ConditionedThermalDataset_N(torch.utils.data.
Dataset):
906             def __init__(self, grids_dict, n_steps: int = 1):
907                 """
908                 grids_dict : dict[str, dict[str, np.ndarray]]
909                 Salida de points_to_grid con múltiples fields.
910                 Estructura: {filename: {"NT11": (T,12,12), "
Stress_Max": ..., "Stress_Min": ...}}
911                 n_steps : int
912                 n_steps=1 -> y_grid shape (1, 12, 12)
913                 n_steps>1 -> y_grid shape (n_steps, 1, 12, 12)
914                 """
915                 self.samples = []
916                 self.n_steps = n_steps
917
918                 for filename, fields_dict in grids_dict.items():
919                     # Extraer arrays de cada campo
920                     grid_nt11 = fields_dict["NT11"] # (T,
12, 12)
921                     grid_Stress_Max = fields_dict["Stress_Max"] # (T,
12, 12)
922                     grid_Stress_Min = fields_dict["Stress_Min"] # (T,
12, 12)
923                     num_tiempos = grid_nt11.shape[0]
924
925                     # Extraer condiciones del nombre del fichero
926                     try:
927                         nombre_sin_ext = filename.replace('.csv', '')
928                         partes = nombre_sin_ext.split('_')
929
930                         if 'Model' in partes[-1]: # <- fix: 'in'
en lugar de .contains()
931                             t_val = float(partes[-4].replace('T', ''))
932                             s_val = float(partes[-3].replace('S', ''))
933                             v_val = float(partes[-2].replace('V', ''))
934                             fiber = partes[-1].split("-")
935                             tag = fiber[-1]
936                         else:
937                             t_val = float(partes[-3].replace('T', ''))
938                             s_val = float(partes[-2].replace('S', ''))
939                             v_val = float(partes[-1].replace('V', ''))
940                             tag = None
941
942                     # Decodificar tag de fibra
943                     fv, ilm = 0.0, 0.0
944                     if tag == 'B':
945                         fv, ilm = 0.0, 0.0
946                     elif tag == 'IL05':
947                         fv, ilm = 0.0, 0.05
948                     elif tag == 'FV001':
949                         fv, ilm = 0.001, 0.0
950                     elif tag == 'IL05FV001':
951                         fv, ilm = 0.001, 0.05
952

```

```

953         conditions = np.array(
954             [t_val/400.0, s_val/10.0, v_val/50.0, fv/0.001, ilm
955              /0.05],
956             dtype=np.float32
957         )
958
959         except Exception as e:
960             print(f"Error parseando '{filename}': {e}. Usando
961             condiciones cero.")
962             conditions = np.zeros(5, dtype=np.float32)
963
964         # Crear muestras
965         for t in range(num_tiempos - n_steps):
966             self.samples.append({
967                 # Input: los 3 campos apilados (3, 12, 12)
968                 'input_grid': np.stack([
969                     grid_nt11[t],
970                     grid_Stress_Max[t],
971                     grid_Stress_Min[t],
972                     ], axis=0),
973                 # Targets: secuencia de n_steps para cada campo
974                 'target_nt11': grid_nt11[t+1 : t+1+
975                 n_steps], # (n_steps, 12, 12)
976                 'target_Stress_Max': grid_Stress_Max[t+1 : t+1+
977                 n_steps],
978                 'target_Stress_Min': grid_Stress_Min[t+1 : t+1+
979                 n_steps],
980                 'conditions': conditions,
981             })
982
983         def __len__(self):
984             return len(self.samples)
985
986         def __getitem__(self, idx):
987             sample = self.samples[idx]
988
989             # (3, 12, 12) 3 canales: NT11, Stress_Max,
990             Stress_Min
991             x_grid = torch.tensor(sample['input_grid'], dtype=
992             torch.float32)
993
994             # (5,) T, S, V, fv, ilm normalizados
995             x_cond = torch.tensor(sample['conditions'], dtype=
996             torch.float32)
997
998             # Apilar targets en canal (n_steps, 3, 12, 12)
999             target = np.stack([
1000                 sample['target_nt11'],
1001                 sample['target_Stress_Max'],
1002                 sample['target_Stress_Min'],
1003                 ], axis=1) # (n_steps, 3, 12, 12)
1004
1005             if self.n_steps == 1:
1006                 # (3, 12, 12) compatibilidad hacia atrás
1007                 y_grid = torch.tensor(target[0], dtype=torch.

```



```
float32)
000     else:
001         # (n_steps, 3, 12, 12)
002         y_grid = torch.tensor(target, dtype=torch.float32)
003
004     return x_grid, x_cond, y_grid
005
006
007
```

```

008
009     class ConditionedThermalCNN(nn.Module):
010         """
011         CNN condicional configurable para predicción de
012         mapas térmicos.
013
014         Parámetros
015         -----
016         grid_size : int
017         Lado del grid cuadrado de entrada (default 12  12
018         x12).
019         channels_in : int
020         Canales de la imagen de entrada (default 1).
021         conv_channels : list[int]
022         N de canales de salida de cada bloque convolucional
023         .
024         Len(conv_channels) == número de capas Conv+Pool.
025         Ejemplo: [16, 32]  2 bloques, 16 y 32 canales.
026         cond_input_dim : int
027         Dimensión del vector de condiciones (default 3: T,
028         S, V).
029         cond_hidden_dims : list[int]
030         Dimensiones ocultas del MLP de condiciones (sin
031         incluir
032         la entrada ni la salida final).
033         Ejemplo: [16]  Linear(3,16)  Linear(16,16).
034         cond_out_dim : int
035         Dimensión de la representación de condición (
036         default 16).
037         regressor_hidden_dims : list[int]
038         Dimensiones ocultas del MLP de regresión (sin
039         incluir
040         fusión ni salida).
041         Ejemplo: [256]  Linear(fusion,256)  Linear(256,out)
042         .
043         cnn_activation : str
044         Función de activación en la rama CNN.
045         mlp_activation : str
046         Función de activación en los MLPs de condición y
047         regresión.
048         n_recursive_steps : int
049         N de pasos recursivos para series temporales (
050         default 1  sin
051         recursión). Con n > 1 el modelo se auto-aplica n-1
052         veces.
053         detach_recursive : bool
054         Si True, desconecta el gradiente entre pasos
055         recursivos
056         (modo "teacher forcing" / inferencia eficiente).
057         Si False, el gradiente fluye a través de todos los
058         pasos.
059         """
060
061     def __init__(

```

```

049     self,
050     grid_size: int = 12,
051     channels_in: int = 3,
052     conv_channels: Optional[List[int]] = None,
053     cond_input_dim: int = 5,
054     cond_hidden_dims: Optional[List[int]] = None,
055     cond_out_dim: int = 16,
056     regressor_hidden_dims: Optional[List[int]] = None,
057     cnn_activation: str = "relu",
058     mlp_activation: str = "relu",
059     n_recursive_steps: int = 1,
060     detach_recursive: bool = False,
061 ):
062     super().__init__()
063
064     # defaults
065     if conv_channels is None:
066         conv_channels = [16, 32]
067     if cond_hidden_dims is None:
068         cond_hidden_dims = [16]
069     if regressor_hidden_dims is None:
070         regressor_hidden_dims = [256]
071
072     self.grid_size = grid_size
073     self.channels_in = channels_in
074     self.n_recursive_steps = n_recursive_steps
075     self.detach_recursive = detach_recursive
076
077     # Rama 1: CNN Espacial
078     # Cada bloque: Conv2d -> Activación -> MaxPool2d
079     (2,2)
080     # Cada MaxPool divide el espacial por 2.
081     conv_layers: List[nn.Module] = []
082     ch_prev = channels_in
083     for ch_out in conv_channels:
084         conv_layers += [
085             nn.Conv2d(ch_prev, ch_out, kernel_size=3, padding
086             =1),
087             _get_activation(cnn_activation),
088             nn.MaxPool2d(2, 2),
089         ]
090         ch_prev = ch_out
091
092     self.conv_branch = nn.Sequential(*conv_layers)
093
094     # Tamaño espacial tras N poolings de factor 2
095     n_pools = len(conv_channels)
096     out_spatial = grid_size // (2 ** n_pools)
097     if out_spatial < 1:
098         raise ValueError(
099             f"Demasiadas capas convolucionales ({n_pools}) para
100             "
101             f"un grid de tamaño {grid_size}. El espacial llegar
102             ía "
103             f"a {out_spatial}."

```

```

100         )
101         self.cnn_feat_dim = conv_channels[-1] * out_spatial
102         * out_spatial
103
104         # Rama 2: MLP de Condiciones
105         cond_dims = [cond_input_dim] + cond_hidden_dims + [
106         cond_out_dim]
107         self.cond_branch = _build_mlp(cond_dims, activation
108         =mlp_activation)
109
110         # Rama 3: Fusión + Regresión ConditionedThermalCNN
111         fusion_dim = self.cnn_feat_dim + cond_out_dim
112         output_dim = channels_in * grid_size * grid_size
113
114         reg_dims = [fusion_dim] + regressor_hidden_dims + [
115         output_dim]
116         self.regressor = _build_mlp(reg_dims, activation=
117         mlp_activation)
118
119         # Info de configuración
120         self._config = dict(
121         grid_size=grid_size,
122         channels_in=channels_in,
123         conv_channels=conv_channels,
124         cond_input_dim=cond_input_dim,
125         cond_hidden_dims=cond_hidden_dims,
126         cond_out_dim=cond_out_dim,
127         regressor_hidden_dims=regressor_hidden_dims,
128         cnn_activation=cnn_activation,
129         mlp_activation=mlp_activation,
130         n_recursive_steps=n_recursive_steps,
131         detach_recursive=detach_recursive,
132         cnn_feat_dim=self.cnn_feat_dim,
133         fusion_dim=fusion_dim,
134         )
135
136         # Forward de un solo paso
137
138         def _forward_one_step(
139         self, x_grid: torch.Tensor, x_cond: torch.Tensor
140         ) -> torch.Tensor:
141             """
142             Un paso de inferencia: (B, C, H, W) + (B, cond_dim)
143             (B, C, H, W)
144             """
145             B = x_grid.size(0)
146
147             # 1. Rama CNN
148             cnn_feat = self.conv_branch(x_grid)
149             cnn_feat = cnn_feat.view(B, -1) # (
150             B, cnn_feat_dim)
151
152             # 2. Rama Condición
153             cond_feat = self.cond_branch(x_cond) # (
154             B, cond_out_dim)

```

```

147
148     # 3. Fusión
149     combined = torch.cat((cnn_feat, cond_feat), dim=1)
150
151     # 4. Regresión
152     out_flat = self.regressor(combined) # (
153     B, C*H*W)
154     return out_flat.view(
155         B, self.channels_in, self.grid_size, self.grid_size
156     )
157
158     # Forward con recursión
159
160     def forward(
161         self,
162         x_grid: torch.Tensor,
163         x_cond: torch.Tensor,
164         n_steps: Optional[int] = None,
165     ) -> torch.Tensor:
166         """
167         Propagación hacia adelante con (opcional) recursión
168         temporal.
169
170         Parámetros
171         -----
172         x_grid : Tensor (B, C, H, W)
173         Grid de entrada en t=0.
174         x_cond : Tensor (B, cond_dim) o (B, n_steps,
175         cond_dim)
176         Vector de condiciones.
177         - Si tiene 2 dimensiones se reutiliza el mismo
178         vector en
179         todos los pasos recursivos.
180         - Si tiene 3 dimensiones se usa x_cond[:, step, :]
181         en cada
182         paso (condiciones que evolucionan con el tiempo).
183         n_steps : int, opcional
184         N de pasos a predecir. Si no se especifica se usa
185         self.n_recursive_steps.
186
187         Retorna
188         -----
189         Si n_steps == 1 -> Tensor (B, C, H, W) [
190         compatibilidad hacia atrás]
191         Si n_steps > 1 -> Tensor (B, n_steps, C, H, W)
192         """
193         n_steps = n_steps or self.n_recursive_steps
194         cond_is_sequence = (x_cond.dim() == 3) # (B, T,
195         cond_dim)
196
197         if n_steps == 1:
198             cond = x_cond[:, 0, :] if cond_is_sequence else
199             x_cond
200         return self._forward_one_step(x_grid, cond)

```

```

194         # Bucle recursivo
195         predictions: List[torch.Tensor] = []
196         current = x_grid
197
198         for step in range(n_steps):
199             if cond_is_sequence:
200                 cond = x_cond[:, step, :]
201             else:
202                 cond = x_cond
203
204             pred = self._forward_one_step(current, cond) # (B,
C, H, W)
205             predictions.append(pred)
206
207             # El grid predicho se convierte en la entrada del
siguiente paso
208             if self.detach_recursive:
209                 current = pred.detach()
210             else:
211                 current = pred
212
213             # (B, n_steps, C, H, W)
214             return torch.stack(predictions, dim=1)
215
216         # Utilidades
217
218         # def config(self) -> dict:
219         #     """Devuelve el diccionario de configuración
del modelo."""
220         #     return dict(self._config)
221         def config(self) -> dict:
222             """Devuelve el diccionario de configuración del
modelo."""
223             # Opción B: excluir claves calculadas que no son
args del __init__
224             _COMPUTED_KEYS = {"cnn_feat_dim", "fusion_dim"}
225             return {k: v for k, v in self._config.items() if k
not in _COMPUTED_KEYS}
226
227         def __repr__(self) -> str:
228             cfg = self._config
229             lines = [
230                 "ConditionedThermalCNN(",
231                 f"    grid           : {cfg['grid_size']}x{cfg['
grid_size']}",
232                 f"    conv_channels : {cfg['conv_channels']} (act={
cfg['cnn_activation']})",
233                 f"    cnn_feat_dim  : {cfg['cnn_feat_dim']}",
234                 f"    cond_mlp      : {cfg['cond_input_dim']} -> {cfg
['cond_hidden_dims']} -> {cfg['cond_out_dim']}",
235                 f"    fusion_dim    : {cfg['fusion_dim']}",
236                 f"    regressor     : {cfg['regressor_hidden_dims']}
(act={cfg['mlp_activation']})",
237                 f"    recursive      : {cfg['n_recursive_steps']}
pasos (detach={cfg['detach_recursive']})",

```

```
238         ")",  
239     ]  
240     return "\n".join(lines)  
241  
242
```

```

243     def training(
244         grid_dataset,
245         epochs: int = 50,
246         batch_size: int = 32,
247         lr: float = 0.001,
248         model_kwargs: dict = None,
249         n_steps: int = 1,
250         criterion = nn.MSELoss,
251         optimizer = torch.optim.Adam
252     ):
253         """
254         Entrena el modelo ConditionedThermalCNN.
255
256         Parámetros
257         -----
258         grid_dataset : Dataset
259             Debe devolver tuplas (x_grid, x_cond, y_target)
260         donde:
261             - n_steps == 1 -> y_target shape (B, C, H, W) [
262               igual que antes]
263             - n_steps > 1 -> y_target shape (B, n_steps, C, H,
264               W)
265             x_cond puede ser (B, cond_dim) ó (B, n_steps,
266               cond_dim)
267             model_kwargs : dict, opcional
268               Argumentos para ConditionedThermalCNN.__init__.
269               Ejemplo: dict(conv_channels=[16,32],
270               n_recursive_steps=5)
271             n_steps : int
272               Número de pasos temporales a predecir en cada
273               forward.
274             Debe coincidir con lo que devuelve el dataset.
275         """
276         device = torch.device("cuda" if torch.cuda.
277             is_available() else "cpu")
278         print(f"Entrenando en: {device} | pasos
279             recursivos: {n_steps}")
280
281         dataloader = DataLoader(grid_dataset, batch_size=
282             batch_size, shuffle=True)
283
284         kwargs = model_kwargs or {}
285         kwargs.setdefault("n_recursive_steps", n_steps)
286         model = ConditionedThermalCNN(**kwargs).to(device)
287         print(model)
288
289         # criterion = nn.MSELoss()
290         optimizer = optimizer(model.parameters(), lr=lr)
291
292         model.train()
293
294         for epoch in range(epochs):
295             running_loss = 0.0

```



```

288         for x_grid, x_cond, y_target in dataloader:
289             x_grid = x_grid.to(device) # (B, C, H, W)
290             x_cond = x_cond.to(device) # (B, cond_dim) ó (B,
n_steps, cond_dim)
291             y_target = y_target.to(device) # (B, C, H, W) ó (B,
n_steps, C, H, W)
292
293             # Forward: n_steps=1 -> (B,C,H,W) ; n_steps>1 -> (B
,T,C,H,W)
294             outputs = model(x_grid, x_cond, n_steps=n_steps)
295
296             # La loss se calcula sobre toda la secuencia
predicha
297             loss = criterion(outputs, y_target)
298
299             optimizer.zero_grad()
300             loss.backward()
301             optimizer.step()
302
303             running_loss += loss.item()
304
305             avg_loss = running_loss / len(dataloader)
306             if (epoch + 1) % 5 == 0:
307                 print(f"Epoch [{epoch+1}/{epochs}] Loss MSE: {
avg_loss:.6f}")
308
309             return model
310

```

```

311
312     def objective(trial, ds_train, ds_test, n_steps=1):
313         """
314         Objective function for Optuna to optimize
hyperparameters.
315         """
316         # 1. Define the hyperparameter search space
317         # Suggest continuous values (Learning Rate) log-
uniformly
318         lr = trial.suggest_float("lr", 1e-4, 1e-2, log=True)
319
320         # Suggest categorical or discrete values
321         batch_size = trial.suggest_categorical("batch_size"
, [32, 64, 128])
322         epochs = trial.suggest_int("epochs", 10, 30)
323
324         # Suggest architecture parameters
325         conv_channels_choice = trial.suggest_categorical("
conv_channels", ["small", "medium", "large"])
326         if conv_channels_choice == "small":
327             conv_channels = [8, 16]
328         elif conv_channels_choice == "medium":
329             conv_channels = [16, 32]
330         else:
331             conv_channels = [32, 64]

```

```

332         regressor_dim = trial.suggest_categorical("
333 regressor_dim", [128, 256, 512])
334
335     # Reconstruct your model_kwargs with Optuna's
336 suggestions
337     model_kwargs = {
338         "grid_size": 12,
339         "conv_channels": conv_channels,
340         "cond_hidden_dims": [16],
341         "cond_out_dim": 16,
342         "regressor_hidden_dims": [regressor_dim],
343         "n_recursive_steps": n_steps # matched with
344 n_steps below
345     }
346
347     # 2. Run the training
348     # Note: Ensure your training function returns or
349 allows tracking validation losses if you want pruning.
350     modelo_entrenado = training(
351         ds_train,
352         epochs=epochs,
353         batch_size=batch_size,
354         lr=lr,
355         model_kwargs=model_kwargs,
356         n_steps=n_steps
357     )
358
359     # 3. Evaluate the model
360     metricas = testing(modelo_entrenado, ds_test,
361 n_steps=n_steps)
362
363     # 4. Return the metric you want to MINIMIZE (e.g.,
364 Mean Squared Error / Loss)
365     # Adjust 'loss' or 'mse' depending on what your
366 testing() function returns.
367     val_loss = metricas.get("loss") or metricas.get("
368 mse")
369
370     # Save the best model state to the trial metadata
371 so we can fetch it later if it wins
372     trial.set_user_attr("model_state", modelo_entrenado
373 .state_dict())
374     trial.set_user_attr("config", modelo_entrenado.
375 config())
376
377     return val_loss
378
379
380 def main():
381     # Fix the indexing bug: Select the specific model
382 string you want to work with
383     chosen_model = 'ALL'
384     variable = 'NT11' # Focus on optimizing one
385 variable configuration at a time

```

```
374         # 1. Load data once (outside Optuna loop to save
375         time)
376         path = obtener_ruta_entorno(f'/home/mdlsg/Documents
377         /TFM/Data/TM_results/TM_cut_{chosen_model}')
378         todos_los_grids = cargar_todos_los_csv(path,
379         variable)
380
381         # 2. Train/Test Split
382         filenames = list(todos_los_grids.keys())
383         split_idx = int(len(filenames) * 0.8)
384
385         train_filenames = filenames[:split_idx]
386         test_filenames = filenames[split_idx:]
387
388         dict_train = {f: todos_los_grids[f] for f in
389         train_filenames}
390         dict_test = {f: todos_los_grids[f] for f in
391         test_filenames}
392
393         # 3. Instantiate Datasets
394         n_steps = 10
395         ds_train = ConditionedThermalDataset_N(dict_train,
396         n_steps=n_steps)
397         ds_test = ConditionedThermalDataset_N(dict_test,
398         n_steps=n_steps)
399
400         # 4. Optuna Study Setup
401         # We use a lambda to pass the datasets into the
402         objective function
403         study = optuna.create_study(direction="minimize")
404         study.optimize(lambda trial: objective(trial,
405         ds_train, ds_test, n_steps=n_steps), n_trials=20)
406
407         # 5. Results & Saving the Best Model
408         print("Best trial:")
409         best_trial = study.best_trial
410         print(f"    Value (Loss): {best_trial.value}")
411         print("    Params: ")
412         for key, value in best_trial.params.items():
413             print(f"        {key}: {value}")
414
415         # Retrieve the best model's parameters and
416         configuration from user attributes
417         best_model_state = best_trial.user_attrs["
418         model_state"]
419         best_config = best_trial.user_attrs["config"]
420
421         best_lr = best_trial.params["lr"]
422         best_batch = best_trial.params["batch_size"]
423         best_epochs = best_trial.params["epochs"]
424
425         # Save the absolute best model found during the
426         study
427         save_path = f"/home/mdlsg/Documents/TFM/Data/Models
```

```

/OPTUNA_modelo_termico_E{best_epochs}_B{best_batch}_LR{
best_lr:.5f}_nS{n_steps}_V{variable}.pth"
417     torch.save({
418         "state_dict": best_model_state,
419         "config": best_config,
420     }, save_path)
421     print(f"Best model saved successfully to {save_path
}!")
422
423     if __name__ == "__main__":
424         main()
425
426

```

8.1.4. Resultados modelos

Mejores modelos temperatura

A continuación se listan los errores de los mejores modelos obtenidos

```

427
428     --- Resultados del Test ---
429     Pasos temporales evaluados : 1
430     MSE (Error Cuadrático Medio) : 0.042543
431     MAE (Error Absoluto Medio)   : 0.066572
432     R2 (Ajuste)                  : 1.0000 (1.0 =
perfecto)
433     [I 2026-05-28 13:13:06,096] Trial 10 finished with
value: 0.04254300892353058 and parameters:
434     {'lr': 0.0006819992604538296,
435      'batch_size': 128,
436      'epochs': 30,
437      'conv_channels': 'large',
438      'regressor_dim': 512}.
439     Best is trial 10 with value: 0.04254300892353058.
440
441

```

```

442
443     --- Resultados del Test ---
444     Pasos temporales evaluados : 10
445     MSE (Error Cuadrático Medio) : 1.184578
446     MAE (Error Absoluto Medio)   : 0.555093
447     R2 (Ajuste)                  : 0.9999 (1.0 =
perfecto)
448
449     Desglose por paso temporal:
450     t+01 MSE=0.231543 MAE=0.282661 C
451     t+02 MSE=0.274956 MAE=0.315424 C
452     t+03 MSE=0.441605 MAE=0.367666 C
453     t+04 MSE=0.668425 MAE=0.431072 C

```

```

454         t+05    MSE=0.932280    MAE=0.500922 C
455         t+06    MSE=1.218089    MAE=0.574710 C
456         t+07    MSE=1.519346    MAE=0.650960 C
457         t+08    MSE=1.837965    MAE=0.729370 C
458         t+09    MSE=2.178447    MAE=0.808997 C
459         t+10    MSE=2.543121    MAE=0.889148 C
460         [I 2026-05-28 17:45:30,464] Trial 13 finished with
value: 1.1845777034759521 and parameters:
461         {'lr': 0.00011643391186535657,
462          'batch_size': 32,
463          'epochs': 25,
464          'conv_channels': 'small',
465          'regressor_dim': 512}.
466         Best is trial 12 with value: 0.8380922079086304.
467

```

Mejores modelos tensión máxima

```

468         --- Resultados del Test ---
469         Pasos temporales evaluados : 1
470         MSE (Error Cuadrático Medio) : 173772029952.000000
471         MAE (Error Absoluto Medio)   : 81988.882812
472         R2 (Ajuste)                   : 0.9997 (1.0 =
perfecto)
473         [I 2026-05-28 13:31:41,932] Trial 15 finished with
value: 173772029952.0 and parameters:
474         {'lr': 0.0007682915385824046,
475          'batch_size': 64,
476          'epochs': 24,
477          'conv_channels': 'small',
478          'regressor_dim': 512}.
479         Best is trial 15 with value: 173772029952.0.
480
481

```

```

482         --- Resultados del Test ---
483         Pasos temporales evaluados : 10
484         MSE (Error Cuadrático Medio) :
2040934825984.000000
485         MAE (Error Absoluto Medio)   : 462787.687500
486         R2 (Ajuste)                   : 0.9963 (1.0 =
perfecto)
487
488         Desglose por paso temporal:
489         t+01    MSE=1146799783936.000000    MAE=280983.468750 C
490         t+02    MSE=1566400839680.000000    MAE=308810.718750 C
491         t+03    MSE=1873158078464.000000    MAE=347849.250000 C
492         t+04    MSE=2215500578816.000000    MAE=392794.812500 C
493         t+05    MSE=2538402742272.000000    MAE=439997.718750 C
494         t+06    MSE=2746974994432.000000    MAE=487595.437500 C
495         t+07    MSE=2737763516416.000000    MAE=533259.312500 C
496

```

```

497         t+08    MSE=2436068802560.000000    MAE=574964.125000 C
498         t+09    MSE=1849134940160.000000    MAE=613274.750000 C
499         t+10    MSE=1299146604544.000000    MAE=648347.125000 C
500         [I 2026-05-28 15:44:54,760] Trial 6 finished with
value: 2040934825984.0 and parameters:
501         {'lr': 0.00022583716702850965,
502          'batch_size': 64,
503          'epochs': 19,
504          'conv_channels': 'small',
505          'regressor_dim': 512}.
506         Best is trial 6 with value: 2040934825984.0.
507
508

```

Mejores modelos tensión mínima

```

509         --- Resultados del Test ---
510         Pasos temporales evaluados : 1
511         MSE (Error Cuadrático Medio) : 90687832064.000000
512         MAE (Error Absoluto Medio)   : 34570.210938
513         R2 (Ajuste)                   : 0.9998 (1.0 =
perfecto)
514         [I 2026-05-28 14:08:58,706] Trial 1 finished with
value: 90687832064.0 and parameters:
515         {'lr': 0.0008042290467770933,
516          'batch_size': 32,
517          'epochs': 22,
518          'conv_channels': 'small',
519          'regressor_dim': 256}.
520         Best is trial 1 with value: 90687832064.0.
521
522

```

```

523         --- Resultados del Test ---
524         Pasos temporales evaluados : 10
525         MSE (Error Cuadrático Medio) :
1798563430400.000000
526         MAE (Error Absoluto Medio)   : 443343.812500
527         R2 (Ajuste)                   : 0.9954 (1.0 =
perfecto)
528
529         Desglose por paso temporal:
530         t+01    MSE=784085876736.000000    MAE=355818.531250 C
531         t+02    MSE=1078677864448.000000    MAE=394677.781250 C
532         t+03    MSE=1321063415808.000000    MAE=415873.125000 C
533         t+04    MSE=1615861645312.000000    MAE=433414.343750 C
534         t+05    MSE=1922246508544.000000    MAE=448586.531250 C
535         t+06    MSE=2179307929600.000000    MAE=461198.093750 C
536         t+07    MSE=2339933519872.000000    MAE=471241.125000 C
537         t+08    MSE=2385056366592.000000    MAE=479122.218750 C
538         t+09    MSE=2300445720576.000000    MAE=485007.250000 C
539

```

```
540         t+10    MSE=2058955522048.000000    MAE=488499.250000 C
541         [I 2026-05-28 20:40:26,200] Trial 11 finished with
value: 1798563430400.0 and parameters:
542         {'lr': 0.00032654954215186293,
543          'batch_size': 64,
544          'epochs': 21,
545          'conv_channels': 'small',
546          'regressor_dim': 512}.
547         Best is trial 11 with value: 1798563430400.0.
548
549
550
```

8.2. PLANOS

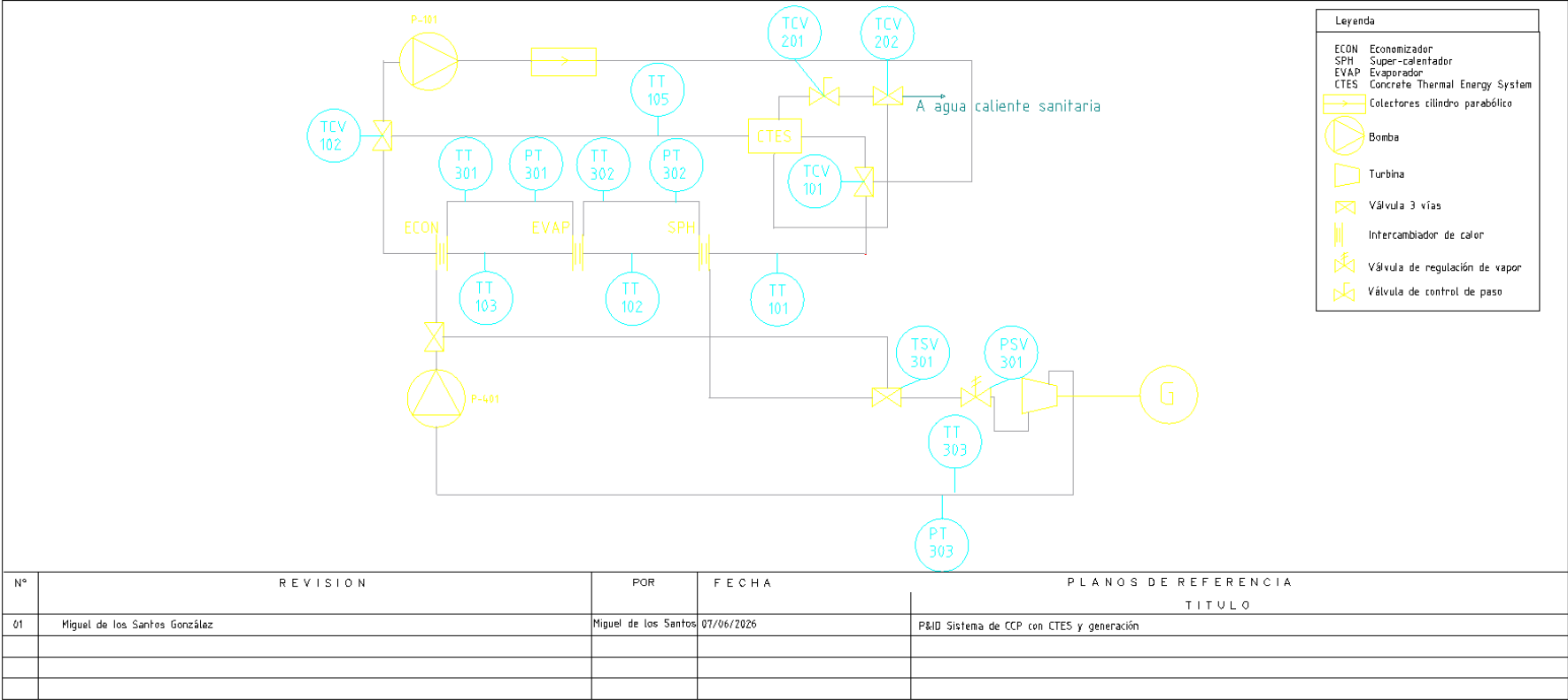


Figura 8.2.1: Diagrama de elementos de sistema CCP con CTES

8.3. PRESUPUESTOS

PLAN DE INVERSIONES		CAPITAL EXPENDITURES (CAPEX)								
CONCEPTO	TIPO	CUENTA CONTABLE	VALOR BRUTO	D&A	¿SE AMORTIZA?	VIDA ÚTIL	% Amortiz.	TABLAS FISCALES		21%
								MÁX. AÑOS	% MÁX. PERMITIDO	IVA
Espejos cilindro parabólicos	Material	Construcciones	131'200 €	6'560 €	SI	20	5.0%	68	3%	27'552 €
Aceite caloportante	Material	Consumible	4'800 €	960 €	SI	5	20.0%	18	12%	1'008 €
Guiado de espejos	Material	Equipos proceso informac	20'000 €	1'000 €	SI	20	5.0%	18	12%	4'200 €
Servomotores de posicionamiento	Material	Maquinaria	12'000 €	600 €	SI	20	5.0%	18	12%	2'520 €
CTES	Material	Construcciones	280 €	28 €	SI	10	10.0%	18	12%	59 €
Sistema de tuberías HFT	Material	Construcciones	12'000 €	600 €	SI	20	5.0%	18	12%	2'520 €
Sistema de tuberías ACS	Material	Construcciones	6'000 €	300 €	SI	20	5.0%	18	12%	1'260 €
Bomba HFT	Material	Maquinaria	10'000 €	1'000 €	SI	10	10.0%	14	12%	2'100 €
Bomba agua	Material	Maquinaria	5'000 €	500 €	SI	10	10.0%	14	12%	1'050 €
Turbina	Material	Maquinaria	70'000 €	3'500 €	SI	20	5.0%	14	16%	14'700 €
Generador	Material	Maquinaria	20'000 €	1'000 €	SI	20	5.0%	14	16%	4'200 €
Economizador	Material	Maquinaria	50'000 €	2'500 €	SI	20	5.0%	14	16%	10'500 €
Evaporador	Material	Maquinaria	30'000 €	1'500 €	SI	20	5.0%	14	16%	6'300 €
Supercalentador	Material	Maquinaria	20'000 €	1'000 €	SI	20	5.0%	14	16%	4'200 €
Equipo de control	Material	Equipos proceso informac	20'400 €	1'020 €	SI	20	5.0%	20	10%	4'284 €
Equipo informático	Material	Equipos proceso informac	2'500 €	500 €	SI	5	20.0%	8	25%	525 €
Sistema SCADA y simulación	Inmaterial	Aplicaciones Informáticas	2'000 €	400 €	SI	5	20.0%	6	33%	420 €
Registro de Marca	Inmaterial	Propiedad Industrial	1'000 €	100 €	SI	10	10.0%		10%	- €
TOTAL		TOTAL CAPEX	417'180 €	23'068 €						87'398 €

Figura 8.3.1: Plan de inversiones con amortizaciones y cuentas contables para un colector

PLAN DE INVERSIONES		CAPITAL EXPENDITURES (CAPEX)								
CONCEPTO	TIPO	CUENTA CONTABLE	VALOR BRUTO	D&A	¿SE AMORTIZA?	VIDA ÚTIL	% Amortiz.	TABLAS FISCALES		21%
								MÁX. AÑOS	% MÁX. PERMITIDO	IVA
Espejos cilindro parabólicos	Material	Construcciones	131'200 €	6'560 €	SI	20	5.0%	68	3%	27'552 €
Aceite caloportante	Material	Consumible	4'800 €	960 €	SI	5	20.0%	18	12%	1'008 €
Guiado de espejos	Material	Equipos proceso informac	20'000 €	1'000 €	SI	20	5.0%	18	12%	4'200 €
Servomotores de posicionamiento	Material	Maquinaria	12'000 €	600 €	SI	20	5.0%	18	12%	2'520 €
CTES	Material	Construcciones	520 €	52 €	SI	10	10.0%	18	12%	109 €
Sistema de tuberías HFT	Material	Construcciones	12'000 €	600 €	SI	20	5.0%	18	12%	2'520 €
Sistema de tuberías ACS	Material	Construcciones	6'000 €	300 €	SI	20	5.0%	18	12%	1'260 €
Bomba HFT	Material	Maquinaria	10'000 €	1'000 €	SI	10	10.0%	14	12%	2'100 €
Bomba agua	Material	Maquinaria	5'000 €	500 €	SI	10	10.0%	14	12%	1'050 €
Turbina	Material	Maquinaria	70'000 €	3'500 €	SI	20	5.0%	14	16%	14'700 €
Generador	Material	Maquinaria	20'000 €	1'000 €	SI	20	5.0%	14	16%	4'200 €
Economizador	Material	Maquinaria	50'000 €	2'500 €	SI	20	5.0%	14	16%	10'500 €
Evaporador	Material	Maquinaria	30'000 €	1'500 €	SI	20	5.0%	14	16%	6'300 €
Supercalentador	Material	Maquinaria	20'000 €	1'000 €	SI	20	5.0%	14	16%	4'200 €
Equipo de control	Material	Equipos proceso informac	20'400 €	1'020 €	SI	20	5.0%	20	10%	4'284 €
Equipo informático	Material	Equipos proceso informac	2'500 €	500 €	SI	5	20.0%	8	25%	525 €
Sistema SCADA y simulación	Inmaterial	Aplicaciones Informáticas	2'000 €	400 €	SI	5	20.0%	6	33%	420 €
Registro de Marca	Inmaterial	Propiedad Industrial	1'000 €	100 €	SI	10	10.0%		10%	- €
TOTAL		TOTAL CAPEX	417'420 €	23'092 €						87'448 €

Figura 8.3.2: Plan de inversiones con amortizaciones y cuentas contables para un lazo

Bibliografía

- [1] Xu, H. J., et al. (2025). *Progress on thermal storage technologies with high heat density in renewables: Latent and thermochemical energy storage*. Renewable and Sustainable Energy Reviews. <https://doi.org/10.1016/j.rser.2025.115587>
- [2] MDPI (2025). *Thermal Energy Storage Technology Roadmap for Decarbonising Medium-Temperature Heat Processes*. Processes Journal. <https://www.mdpi.com/2071-1050/17/21/9693>
- [3] AT Tafere. *Thermal Energy Storage Technologies: A Review of Current Landscape and Future Directions*. F1000Research 2026 15:83, <https://doi.org/10.12688/f1000research.176639.1>
- [4] Barrera-Díaz, C.-D., Ortiz-Vásquez, L.-E., Valbuena-Niño, E.-D., Fuentes, D., Montero-Chacón, E, y Endrino, J.-L. (2023). *Simulation of an Organic Rankine Cycle driven by solar energy based on the absorptance of NFT and Pyromark coatings in a Parabolic Trough Collector*. DYNA, 100(5), 436–443. <https://doi.org/10.52152/D11456>
- [5] Goldleaf Labs (s.f.). *Características del HTF Dowtherm A, compuesto por óxido de difenilo y bifenilo*. Recuperado de: <https://www.goldleaflabs.com/dowtherm-a.html>
- [6] Liu, Donghuan, Liu, Yinghua, Applications of Discontinuous Galerkin Finite Element Method in *Thermomechanical Coupling Problems with Imperfect Thermal Contact, Mathematical Problems in Engineering*, 2013, 861417, 14 pages, 2013. <https://doi.org/10.1155/2013/861417>
- [7] A. K. Sakban, M. Mashreib *Revista ingeniería de construcción*, ISSN-e 0718-5073, Vol. 35, N. 2, 2020, págs. 148-169
- [8] DITTLER, R. A., ALVAREZ HOSTOS, J. C., ALBANESI, A. E. (2024). *Solución Numérica de Problemas Transitorios de Conducción de Calor con Histéresis de Cambio de Fase Sólido-Líquido*. AJEA <https://doi.org/10.33414/ajea.1716.2024>
- [9] Miguel Prieto Palomera, Delfín Silio Salcines. *Análisis térmico por métodos numéricos para la optimización de una matriz de contenedores de residuos radiactivos en un almacén temporal*.
- [10] Prince, S. J. D. (2025). *Understanding Deep Learning*, The MIT Press. <http://udlbook.com>

- [11] Khadilkar A, Wang J, Rai R (2019) *Deep learning-based stress prediction for bottom-up SLA 3D printing process*. Int J Adv Manuf Technol 102(5):2555–2569. <https://doi.org/10.1007/s00170-019-03363-4>
- [12] Leon Herrmann, Stefan Kollmannsberger *Deep learning in computational mechanics: a review* <https://doi.org/10.1007/s00466-023-02434-4>
- [13] Zhenguo N, Haoliang J, Burak KL (2020) *Stress field prediction in cantilevered structures using convolutional neural networks*. J Comput Inform Sci Eng 20(1):011002. <https://doi.org/10.1115/1.4044097>
- [14] Wasim Khaliq, Venkatesh Kodur, *Thermal and mechanical properties of fiber reinforced high performance self-consolidating concrete at elevated temperatures*, <https://doi.org/10.1016/j.cemconres.2011.06.012>
- [15] Jianxun Liu, Meirong Chen, Jianping Yang & Zhishen Wu (2020), *Study on Mechanical Properties of Basalt Fibers Superior to E-glass Fibers*, *Journal of Natural Fibers*, <https://doi.org/10.1080/15440478.2020.1764438>
- [16] Baghban, M. H., Hovde, P. J., y Jacobsen, S. (2013). *Analytical and experimental study on thermal conductivity of hardened cement pastes*. Materials and Structures. <https://doi.org/10.1617/s11527-012-9995-y>
- [17] *Concrete: Microstructure, Properties, and Materials*, 3rd ed., The McGraw-Hill Companies, Inc., New York, USA, 2006
- [18] Ghatage, P. S., Kar, V. R., y Sudhagar, P. E. (2020). *On the numerical modelling and analysis of multi-directional functionally graded composite structures: A review*. Composite Structures. <https://doi.org/10.1016/j.compstruct.2019.111837>
- [19] Endesa *Recomendaciones de Endesa para contratar potencia*. Recuperado de: <https://www.endesa.com/es/blog/blog-de-endesa/luz/que-potencia-electrica-necesitas>
- [20] Instituto Nacional de Estadística *Viviendas totales*. Recuperado de: <https://www.ine.es/jaxi/Datos.htm?tpx=59531>
- [21] Instituto Nacional de Estadística *Viviendas por intensidad de uso a partir del consumo eléctrico*. Recuperado de: <https://www.ine.es/jaxi/Datos.htm?tpx=59531>
- [22] Instituto Nacional de Estadística *Estadística sobre el Suministro y Saneamiento del Agua. Año 2022*. Recuperado de: https://www.ine.es/dyngs/INEbase/es/operacion.htm?c=Estadistica_C&cid=1254736176834&menu=ultiDatos&idp=1254735976602
- [23] Instituto Nacional de Estadística *Población residente por fecha, sexo y edad*. Recuperado de: <https://ine.es/jaxiT3/Datos.htm?t=56934>
- [24] Ministerio para la Transición Ecológica y el Reto Demográfico *Guía para el cálculo de la huella de carbono y para la elaboración de un plan de reducción de emisiones de una organización* Recuperado de: https://www.miteco.gob.es/content/dam/miteco/es/cambio-climatico/temas/mitigacion-politicas-y-medidas/guia_huella_carbono_tcm30-479093.pdf