



Universidad
Zaragoza

Trabajo Fin de Máster

Desarrollo de técnicas de inteligencia artificial para
la creación de gemelos humanos digitales.

Development of artificial intelligence techniques for
the creation of digital human twins.

Autor

Lucas Tesán Altelarrea

Director

David González Ibáñez

ESCUELA DE INGENIERÍA Y ARQUITECTURA
2023-2024



DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./D^a. Lucas Tesán Altelarrea,

con nº de DNI 73050906P en aplicación de lo dispuesto en el art.

14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Máster _____, (Título del Trabajo)

Desarrollo de técnicas de inteligencia artificial para la
creación de gemelos humanos
digitales

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 23/09/23

Fdo: Lucas Tesán Altelarrea

AGRADECIMIENTOS

Es con profundo agradecimiento que dedico estas palabras a aquellos cuyo apoyo y orientación han sido fundamentales en la realización de este Trabajo de Fin de Máster.

En primer lugar, agradezco a David González por brindarme la oportunidad de embarcarme en este proyecto y por su excelente orientación a lo largo del mismo. Entré en su despacho en busca de un desafío, y lo que encontré fue mucho más que un simple proyecto.

Quiero expresar mi agradecimiento también a Elías Cueto y a Iciar Alfaro por toda su paciencia, por estar siempre disponibles para nuestras dudas, errores e ideas descabelladas, por hacer de este grupo de investigación un verdadero equipo y por confiar en nosotros como si fuéramos investigadores ya formados.

Hablo en plural porque detrás de esta inteligencia artificial hay un gran grupo de investigadores y unos increíbles compañeros de laboratorio que han hecho de esta etapa una de las más gratificantes de mi vida. Si no es suficiente motivo de felicidad estar trabajando en tu campo de vocación, lo es el tener un ambiente de trabajo tan bonito como este.

Quiero reconocer especialmente a ciertas personas sin las cuales este trabajo no sería lo que es ahora: a Ángela, por ser un apoyo incondicional en los momentos más difíciles y emocionales del proyecto; a Hugo, por tu confianza ciega en mí y en mis decisiones, por tu inagotable admiración y toda tu ayuda en estos últimos meses tan duros ; a mis amigos Eduardo, José Antonio y Alessandro, por ser mi soporte a diario y por escucharme delirar sobre mates y ciencia sin juzgar mi cordura; a Pablo Dolado, quien sin darse cuenta, despertó mi interés por la programación, la termodinámica y la inteligencia artificial cuando era apenas un crio que acababa su primera experiencia laboral; a Quercus por permitirme entrar en su mundo y empezar a entender la inteligencia artificial como siempre había soñado; a Mikel, por ser la voz crítica e inconformista que me ha impulsado a ver otras perspectivas y que sin darse cuenta ha conseguido hacer de este proyecto algo de lo que podemos estar orgullosos.

No puedo pasar por alto el apoyo incondicional de mi familia, especialmente de mi padre y mi madre, por creer en mi visión de la vida y por no intentar influir en mis decisiones. Su comprensión, amor incondicional y admiración hacia mi desarrollo son invaluableles.

Tampoco puedo olvidar que este trabajo fue posible gracias a la beca proporcionada por el I3A y a los recursos facilitados por el grupo AMB. Sin ellos, todo el trabajo que está detrás de estas páginas habría sido inviable.

A todos ustedes, mi más sincero agradecimiento.

RESUMEN

La simulación numérica se ha consolidado como un recurso fundamental en la modelización del comportamiento de sistemas físicos, permitiendo la realización de cálculos de gran magnitud. No obstante, la aparición de modelos biológicos más sofisticados y la necesidad de potenciar el desempeño de estos cálculos, con miras a objetivos más ambiciosos, nos motiva a explorar alternativas a los métodos tradicionalmente establecidos, como el método de elementos finitos (MEF).

Dentro de estos emergentes modelos biológicos destaca el Gemelo Digital Humano, que se define como una copia o contrapartida en el ciberespacio de una persona real en el mundo físico. En esencia, es un modelo basado en datos como la edad, el peso, el sexo, el tipo de tejido, las interacciones con su entorno, su historial pasado de eventos ...etc. La idea es que estos datos cambien en tiempo real como lo harían en el caso de la persona real. Estos datos pueden proceder de muchas fuentes distintas, como pueden ser los dispositivos inteligentes, los historiales médicos, o ser añadidos manualmente.

Sin embargo, este modelo no es solo una mera base de datos, también hay muchos cálculos detrás de él, analizando todos los tipos de datos que se le han enviado y utilizando la computación en nube, el aprendizaje automático y varias herramientas basadas en IA, para obtener información sobre el estado físico de la persona y la forma en que podría verse influido por eventos específicos (ingesta de medicamentos, exposición a ciertas sustancias, etc.). Incluso podría proporcionar diagnósticos, predicciones de enfermedades futuras o consejos de bienestar a los pacientes.

Una perspectiva prometedora para abordar esta creciente complejidad está constituida por los modelos de aprendizaje profundo geométrico, sustentados en la termodinámica del problema y su capacidad de identificar la dinámica de sistemas desconocidos, incluso en presencia de comportamientos altamente no lineales.

Siguiendo la línea de estos modelos de aprendizaje, nuestro trabajo quiere validar su aplicabilidad a sistemas biológicos y explorar las nuevas posibilidades que la inteligencia artificial híbrida puede ofrecer a este campo.

Índice

1. Introducción	1
1.1. Contexto y motivación	1
1.2. Límites de la simulación numérica	2
1.3. Estudios previos	3
1.4. Objetivo	4
2. Metodología	6
2.1. Obtención y preprocesado del modelo geométrico	6
2.2. Creación de la base de datos mediante métodos numéricos	7
2.2.1. Iteración y extracción mediante elementos finitos	8
2.3. Aprendizaje automático científico.	10
2.3.1. Redes de grafos guiadas únicamente por datos	11
2.3.2. Redes de grafos híbridas	15
3. Resultados	20
3.1. Redes de grafos guiadas únicamente por datos: modelo simplificado . .	20
3.1.1. Modelo resultante y métricas	20
3.2. Redes de grafos guiadas únicamente por datos: modelo general	21
3.2.1. Modelo resultante y métricas	22
3.3. Inteligencia artificial híbrida: modelo general	22
3.3.1. Modelos resultantes y métricas	22
3.4. Estudio de robustez frente a cambios de geometría	24
3.4.1. Inferencia en entrenamientos sin variabilidad geométrica	25
3.4.2. Inferencia en entrenamientos con variabilidad geométrica	25
4. Conclusiones	28
5. Bibliografía	30
Lista de Figuras	34

Lista de Tablas	43
Anexos	46
A. Modelo simplificado - Redes guiadas únicamente por datos	47
A.1. Barrido de hiperparámetros	47
A.2. Comparativas y métricas adicionales	49
B. Modelo completo - Redes guiadas únicamente por datos	52
B.1. Barrido de Hiperparámetros	52
B.1.1. Robustez en inferencia por medio del ruido	52
B.1.2. Escalado constante del ruido gaussiano	53
B.1.3. Ruido mixto	55
B.2. Redes con mayor cantidad de parámetros entrenables	57
B.3. Comparativas y medidas adicionales	57
C. Modelo completo - Redes híbridas	60
C.1. Barrido de hiperparámetros	60
C.1.1. Escalado constante del ruido gaussiano	60
C.1.2. Ruido gaussiano proporcional	61
C.1.3. Ruido mixto	62
C.2. Influencia del procesador de la red de grafos	65
C.2.1. Barrido de hiperparámetros para redes con pesos compartidos	65
C.3. Redes con mayor cantidad de parámetros entrenables	66
C.4. Comparativas y métricas adicionales	67
D. Algoritmos de entrenamiento, validación y testeo	71
D.1. Redes guiadas por datos	71
D.2. Redes híbridas	72
E. Marco teórico	74
E.1. Modelo biológico	74
E.1.1. Anatomía y fisiología del hígado	74
E.1.2. Propiedades de los tejidos hepáticos	75
E.2. Características generales de las redes neuronales	79
E.2.1. Teoría básica de las redes neuronales, el perceptrón multicapa	79
E.2.2. Encoder-decoder y el espacio latente	81
E.2.3. Redes de grafos y el sesgo geométrico inductivo	82
E.2.4. Redes informadas por física y la inteligencia artificial híbrida	84

F. Justificación de un sistema cerrado por medio del port metripleptico	85
F.1. Formalismo GENERIC para un sistema abierto	85
G. Normalización de los datos de entrenamiento	87
G.1. Distribución de los datos	87
G.2. Estandarización Z	88
G.3. Normalización MinMax	89
G.4. Otras normalizaciones	89
G.4.1. Box-Cox	90
G.4.2. Log-Normal	90
H. Robustez frente a cambios de geometría	91
H.1. Modelos pre-entrenados con una geometría	91
H.1.1. Redes guiadas por datos	91
H.2. Redes híbridas	93
I. Arquitecturas recurrentes	95
I.1. Contexto y metodología	95
I.2. Resultados	97
I.3. Conclusiones	97
J. Velocidad de respuesta: resultados y comparativas	99
K. Valoración de robustez ante mallas densas	100
K.1. Contexto y resultados	100
K.2. Representaciones gráficas	101
L. Robustez adquirida desde el entrenamiento con bases de datos mixtas	103
L.1. Resultados y métricas	103
L.2. Comparativas tridimensionales en inferencia	104
M. Especificaciones técnicas	108
M.1. Entorno de desarrollo	108
M.2. Aceleración de cálculos	108
M.3. Computación en la nube	108
M.4. Modelado geométrico	109
M.5. Modelado de elementos finitos	109

Capítulo 1

Introducción

1.1. Contexto y motivación

Los gemelos digitales surgen inicialmente como respuesta al avance de los modelos computacionales (*in silico*), con el propósito de simular con gran detalle el funcionamiento fisiológico del cuerpo humano. Siendo herramientas indispensables desde el punto de vista de la sostenibilidad, los gemelos digitales ofrecen una alternativa prometedora para reducir la dependencia de los costosos y éticamente cuestionables ensayos clínicos en humanos y animales. Sin embargo, la complejidad de estos modelos biológicos ha experimentado un crecimiento exponencial, lo que ha limitado nuestro entendimiento y capacidad de simulación incluso de pequeñas porciones de órganos.

Generalmente, los métodos numéricos han constituido las bases para simular sobre estas representaciones digitales de la anatomía humana. Sin embargo el estado del arte de la simulación numérica, más concretamente los elementos finitos, no ha tenido avances significativos durante los últimos años. Este factor sumado con la creciente popularidad y evolución de las unidades de procesamiento gráfico (GPUs) han impulsado una nueva dinámica de trabajo donde las redes neuronales buscan sustituir a los métodos numéricos ya establecidos.

Este proyecto se centra en analizar y aplicar las últimas tendencias utilizando algunas de las arquitecturas más innovadoras en el ámbito de las redes neuronales. Integrando en su conjunto diferentes técnicas ya consolidadas como los auto-codificadores, las redes de grafos o las redes informadas por física.

1.2. Límites de la simulación numérica

La simulación numérica se ha afianzado como una herramienta confiable para enfrentar problemas que requieren cálculos a gran escala. No obstante, el rendimiento de este método para abordar modelos más complejos encuentra limitaciones debido a la finita capacidad de cálculo de los ordenadores actuales. Esto ocurre debido a que los recursos computacionales requeridos para iterar un método numérico aumentan de manera exponencial con la complejidad y precisión del modelo, por lo que existe una posibilidad tangible de que lleguemos a un punto en el cual, incluso las computadoras más potentes, no puedan manejar una cantidad tan abrumadora de datos y operaciones.

Una solución factible y existente son las redes neuronales guiadas por datos. Estas redes tienen la capacidad de aprender la física subyacente a un problema únicamente a partir de los datos disponibles, sin requerir conocimiento previo de las leyes físicas. No obstante, este enfoque requiere una gran cantidad de datos para un entrenamiento efectivo, siendo su inferencia totalmente sesgada por sus propios datos de entrenamiento y sin saber realmente que está ocurriendo dentro de ese conjunto de miles y miles de operaciones (modelos de caja negra). El enfoque propuesto para este proyecto se basa en el modelo de caja gris, detallado por Serg Masis en su libro “Interpretable Machine Learning with Python” [1], el cual se distingue por proporcionar un mayor entendimiento de los procesos de aprendizaje y inferencia de la red, necesitando una cantidad de datos de entrenamiento relativamente menor que para las arquitecturas guiadas por datos.

La convergencia del conocimiento humano y la inteligencia artificial, en conjunción con el enfoque citado, crean la rama que fundamenta el objetivo de este proyecto: la inteligencia artificial híbrida (Figura 1.1).

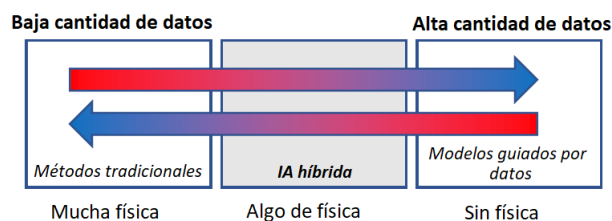


Figura 1.1: Esquematización que compara, respecto a otras dos ramas más tradicionales, la idea detrás de la inteligencia artificial híbrida. Incorporando la física de los métodos tradicionales y el aprendizaje guiado por datos, encontrando un balance que optimice el rendimiento del sistema.

Sin embargo, esta tecnología está en su etapa más inicial y su robustez es incierta: la línea que separan los modelos de caja negra y los modelos de caja gris es realmente fina. Pero las ultimas tendencias de investigación ofrecen algo de esperanza en un futuro donde las redes neuronales y su aprendizaje dejen de ser un misterio.

1.3. Estudios previos

La problemática expuesta no nace en este siglo, durante los años noventa ya se buscaban metodologías numéricas que facilitasen los cálculos masivos de datos. No obstante, estas técnicas no involucraban la conocida inteligencia artificial, sino que se nutrían de herramientas basadas en la reducción de dimensionalidad. Uno de estos enfoques es la técnica denominada como “Proper Generalized Decomposition”, la cual define una solución multiparamétrica compuesta o formada por productos de funciones de menor (o baja) dimensionalidad, mejorando la eficiencia del cálculo numérico sin comprometer enormemente la precisión del modelo [2] y llegando a tener aplicabilidad incluso en entornos biomecánicos con tejidos blandos [3], siendo destacable estudios que obtenían hasta 1KHz de velocidad para la simulación en entornos hápticos. [4].

Todas estas metodologías comparten la característica de ejecutarse en la CPU. Sin embargo, en la actualidad, la creciente popularidad y la potencia que ofrecen las GPUs están permitiendo definir un nuevo enfoque para este tipo de aplicaciones. Este punto de inflexión llega con la aparición de la inteligencia artificial en el campo de la simulación numérica, inicialmente como un complemento a los análisis por elementos finitos (FEA) [5, 6] pero poco a poco estableciéndose como un sustituto a esta metodología [7].

Lejos de la concepción de caja negra, las redes neuronales tienen una capacidad inexplorada de aprender de la mano de la física, evolucionando poco a poco hasta el concepto teórico mencionado de caja gris. A raíz de este enfoque, surgen dos grandes sesgos: el sesgo geométrico [8] aprovecha la estructura geométrica del problema y por ende, de sus datos para establecer dependencias y guiar el aprendizaje. Mientras que el sesgo físico [9] se atribuye a la regulación del aprendizaje de la red por medio de las ecuaciones que rigen el comportamiento teórico definido por la física del problema. Siendo ambos sesgos los componentes diferenciadores del aprendizaje híbrido.

Durante esta ultima década, la similitud entre las redes de grafos y la discretización espacial de los métodos basados en elementos finitos ha permitido a la inteligencia artificial entender las relaciones no lineales de la mecánica de solidos deformables sin ser entrenadas con física: DeepMind inicialmente lo propuso para la interacción de geometrías complejas [10], sin embargo NVIDIA lo ha extrapolado, más recientemente, para medicina de precisión cardiovascular [11]. No obstante, la creciente divergencia sobre los modelos de caja negra ha inspirado al grupo de investigación AMB, perteneciente a la Universidad de Zaragoza, a desarrollar nuevas tecnologías que mejoren la interpretabilidad de la inteligencia artificial: en 2020 Beatriz Moya presentó su metodología de aprendizaje informado por física para la dinámica de fluidos [12]. En el mismo año, Alberto Badías integró la reducción de dimensionalidad dentro de una aplicación para la simulación en tiempo real, siendo capaz de simular la colisión y deformación de dos objetos en contacto [13]. Posteriormente, en 2021, Quercus Hernandez utilizo la formulación GENERIC para informar físicamente a la red y componer un robusto integrador temporal [14]. Pero no fue hasta 2023 que se integraron las redes de grafos y el formalismo GENERIC, desarrollando las redes de grafos informadas por la termodinámica del sistema dentro de la inteligencia artificial híbrida [15, 16].

1.4. Objetivo

El objetivo es evaluar la capacidad y robustez de la inteligencia artificial híbrida para la simulación de gemelos digitales de un órgano con un comportamiento mecánico complejo, aportando nuevas herramientas y obteniendo respuestas a los defectos de las arquitecturas existentes.

En primer lugar, se analizarán los resultados de arquitecturas guiadas por datos utilizando una base de datos de alta fidelidad, con el objetivo de servir como comparativa al aprendizaje híbrido. Esta base será representativa de un hígado no patológico, partiendo desde una posición de reposo absoluto y siendo sometido a un desplazamiento impuesto, normal a la superficie del modelo indeformado. Evaluando el error en inferencia de las tres variables de estado que representan el sistema: posición, velocidad y tensión. De esta forma, se establecerán datos de control con el objetivo de no solo reducir el error, sino también de aumentar la capacidad de generalización del modelo ante cambios en la geometría dentro de los límites fisiológicos.

En segundo lugar, se valorará la eficiencia del sesgo físico en las redes híbridas y se comparará, para diferentes casos, su rendimiento frente a las redes guiadas por datos. Estableciendo conclusiones objetivas sobre esta metodología y valorando sus límites, con la ambición de implementar su desarrollo a entornos hápticos.

Finalmente, se estudiará la influencia de la variabilidad geométrica dentro de la base de datos como herramienta para mejorar la inferencia sobre geometrías no entrenadas. Analizando también la amplificación de datos mediante la adición de ruido durante el entrenamiento (Figura 1.2), con el objetivo de mejorar la robustez del despliegue temporal de los datos, dado que se prevé que la acumulación de ruido sea uno de los principales cuellos de botella del proyecto.

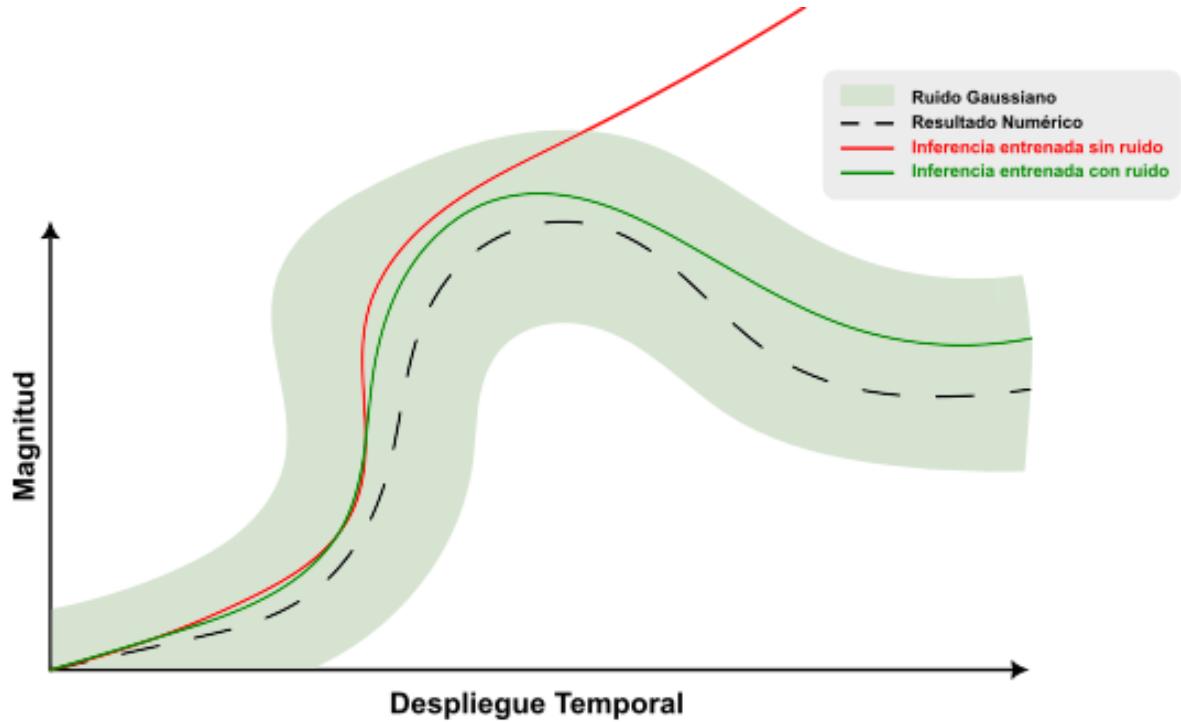


Figura 1.2: Representación del efecto del ruido sobre el despliegue temporal de una variable de estado cualquiera, evitando que la acumulación de error genere divergencias o grandes errores relativos.

Capítulo 2

Metodología

2.1. Obtención y preprocesado del modelo geométrico

Dentro del contexto del proyecto, la obtención y preprocesado del modelo geométrico definen la parte central de la creación de la base de datos. Esta etapa constituye el núcleo central, ya que la calidad y realismo del modelo geométrico impactan directamente en la eficacia del entrenamiento y, por ende, en la relevancia de los resultados obtenidos.

Inicialmente, la geometría se estandariza como un archivo .STL (Stereolithography) de resolución variable. Este formato, ampliamente usado para impresión 3D se caracteriza por la representación tridimensional mediante una malla de triángulos o polígonos. Sin embargo, no almacena información sobre los colores, texturas ni propiedades físicas del modelo (Figura 2.1).

El procesamiento de esta malla se lleva a cabo empleando la aplicación Blender [17], que incorpora una API (Interfaz de Programación de Aplicaciones) basada en el lenguaje de programación Python [18]. Esta API habilita al usuario para crear un conjunto de instrucciones, comúnmente conocidas como *macros*, con el propósito de automatizar las transformaciones deseadas.

El objetivo de la macro es regular tanto la escala como la cantidad de polígonos del modelo, reduciendo la resolución del mismo a un mallado superficial de aproximadamente mil nodos, facilitando su manipulación y simplificando aristas o bordes pronunciados que puedan resultar de la segmentación del órgano.

El archivo “.STL” resultante es únicamente un modelo geométrico tipo cáscara, sin volumen. Por lo que es necesario continuar su preprocesado con GMesh [19] para generar una malla de volumen que coincida con la malla superficial. Adicionalmente se generan con MeshLab [20] los vectores normales a los nodos superficiales.

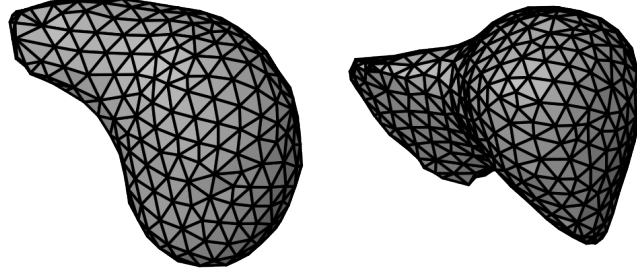


Figura 2.1: Modelo tridimensional del hígado reducido a una menor resolución poligonal con Blender. El total de nodos presentes en la malla es de 530 y su escala es 1/1 con la real.

2.2. Creación de la base de datos mediante métodos numéricos

La siguiente etapa define el marco de trabajo que automatiza la creación de una base de datos para el entrenamiento y predicción de la red. Este proceso cubre desde el planteamiento numérico de la dinámica del sistema, hasta la extracción de datos y post-procesado de los mismos.

El fenómeno físico simulado puede expresarse como un sistema de ecuaciones diferenciales que codifican la evolución temporal de sus variables de estado, $\mathbf{z}$. Por lo que se puede definir, dentro de un sistema cerrado, que la variación temporal del mismo es dependiente de este conjunto de variables de estado (Ecuación 2.1),

$$\dot{\mathbf{z}} = \frac{d\mathbf{z}}{dt} = F(\mathbf{z}, t), \quad t \in I = (0, T] \quad (2.1)$$

Donde t se refiere a la coordenada temporal en el intervalo de tiempo I y $F(\mathbf{z}, t)$ se refiere a una función diferencial no lineal arbitraria. Para que la red aprenda $F(\mathbf{z}, t)$, en un problema mecánico, no lineal, es necesario definir un total de doce variables de estado. Siendo las tres dimensiones de la posición, las tres de la velocidad y las seis del tensor simétrico de tensiones de Cauchy, las que físicamente definen el comportamiento del sistema.

Las simulaciones realizadas mediante elementos finitos son implícitas, concretamente cuasi-estáticas, con una interpolación a 20 pasos de tiempo cada una y con el objetivo de, tras aproximar las velocidades, simular los resultados de un ensayo explícito.

El intervalo completo, denotado como I , y el cálculo integral de todos sus pseudo-tiempos se referirán en adelante como “rollout”.

2.2.1. Iteración y extracción mediante elementos finitos

Para la extracción de las tres componentes de posición y las seis del tensor de tensiones, la malla de volumen es importada a ABAQUS [21] donde se le asignan las propiedades mecánicas ya definidas en el Anexo E. De esta forma se establece un ensayo de referencia que actúa como guía para la creación de las bases de datos posteriores. A continuación se implementa un marco automatizado que emplea tanto la API de ABAQUS como Matlab [22] para calcular y extraer los resultados en cada pseudotiempo (Figura 2.2).

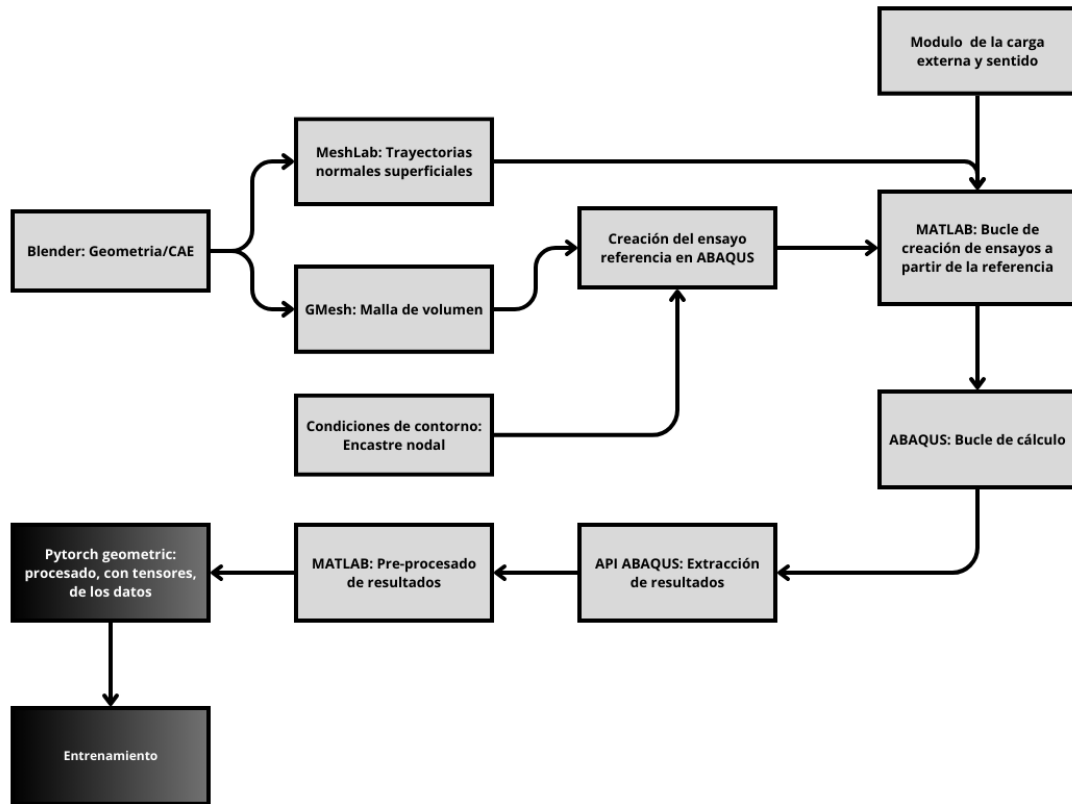


Figura 2.2: Esquematzación del proceso de creación y preprocesado de la base de datos.

Sin embargo, no todas las variables de estado tienen una extracción directa desde los ensayos cuasi-estáticos. En el caso de las velocidades, se pueden establecer dos metodologías de cálculo en función de la precisión necesaria:

Denominando inicialmente la posición de un sistema cualquiera como x , el planteamiento de primer orden se calcula mediante su diferencia lineal dividida entre el pseudo-tiempo h . No obstante, la aproximación de primer orden puede introducir errores significativos si h no es lo suficientemente pequeño, resultando inadecuada para sistemas con cambios rápidos en la posición.

Por otra parte, el planteamiento de segundo orden se calcula mediante el método de diferencias finitas, permitiendo capturar las transiciones dinámicas del sistema con mayor precisión. La aplicación de las diferencias finitas se subdivide en: hacia adelante (2.2), hacia atrás (2.3) o de manera central (2.4), dependiendo de las necesidades específicas del análisis:

$$\text{Diferencias hacia adelante (Forward): } f'(x_i) \approx \frac{f(x_{i+1}) - f(x_i)}{h}, \quad (2.2)$$

$$\text{Diferencias hacia atrás (Backward): } f'(x_i) \approx \frac{f(x_i) - f(x_{i-1})}{h}, \quad (2.3)$$

$$\text{Diferencias centrales (Central): } f'(x_i) \approx \frac{f(x_{i+1}) - f(x_{i-1}))}{2h}, \quad (2.4)$$

siendo la metodología de segundo orden la empleada mayormente para el cálculo de las velocidades, comenzando para todas las bases de datos desde la posición de reposo y evitando que se filtre información posicional del movimiento en su inicio.

Entrando en las características generales de las condiciones de contorno, los desplazamientos puntuales se establecen nodalmente en dirección normal a la superficie, siendo un único desplazamiento puntual por ensayo. (Figura 2.3).

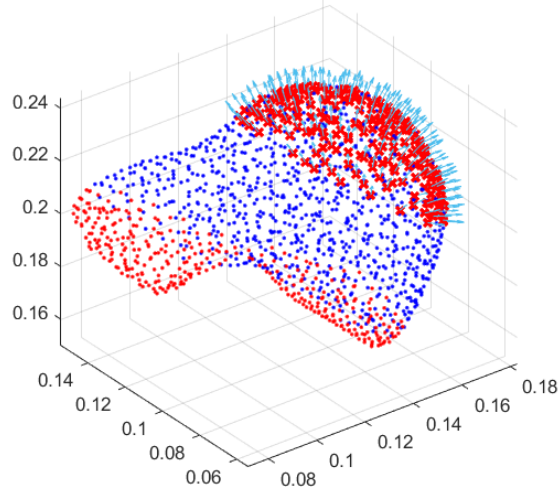


Figura 2.3: Representación visual de la base de datos empleada. Los nodos en rojo muestran los nodos encastrados, mientras que se dibujan en conjunto todos los diferentes puntos de aplicación de los desplazamientos con sus respectivas normales, siendo para cada cada ensayo un único punto de aplicación y un único desplazamiento impuesto.

2.3. Aprendizaje automático científico.

El objetivo de la red es crear un integrador temporal, partiendo del estado de referencia ($\mathbf{z}^t$), debe de ser capaz de predecir cual va a ser su evolución temporal ($\dot{\mathbf{z}}$) para el siguiente pseudo-tiempo (Figura 2.4). Evaluando al final cual es su robustez al integrar recurrentemente hasta 20 pseudo-tiempos, solo conociendo el instante inicial del sistema (reposo, $\mathbf{z}^0$).

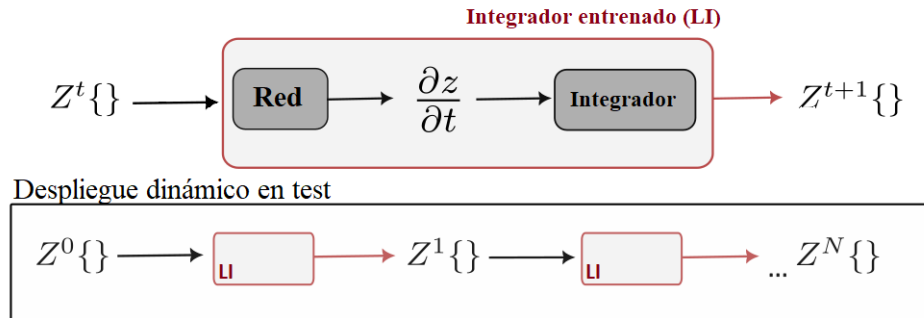


Figura 2.4: Esquematización del como la red entrena sobre una integración temporal entre dos pseudo-tiempos del modelo, sin embargo para validar estos resultados se calcula la integración temporal completa del modelo.

El proyecto se divide en dos metodologías: las arquitecturas híbridas y las arquitecturas guiadas por datos. Las cuales pretenden componer un robusto integrador temporal para posición, velocidad y tensión (denominado a partir de este punto como modelo general).

No obstante, aunque el sistema puede estar definido por tantas variables de estado como dicte la física de su comportamiento, es factible realizar simplificaciones gracias a la capacidad de la red de inferir relaciones no lineales. Esto permite comprobar el rendimiento de un modelo simplificado que solo contempla variables de estado dependientes exclusivamente del aporte de los desplazamientos para comprobar dicho comportamiento con el modelo general ya mencionado.

2.3.1. Redes de grafos guiadas únicamente por datos

El concepto de cómo las redes de grafos interpretan la física y la operan en su espacio latente (Anexo E: Encoder-Decoder y el espacio latente) se plantea a partir del intercambio de información entre un actuador y un receptor [10].

Si el objetivo inicial es lograr un integrador temporal de posición, una de las alternativas algorítmicas disponibles son los “MeshGraphNets” [10]. Este enfoque se basa en la codificación de las coordenadas normalizadas de los nodos, cada uno con una etiqueta correspondiente (como encastrado, actuador, receptor, entre otros), y las longitudes relativas de las aristas. Estos datos, tanto en su estado inicial como en cada iteración, determinan la deformación del modelo.

El funcionamiento de estas redes se alimenta de la información asignada a cada elemento del grafo (Figura 2.5), sin embargo una de las ventajas de las ‘MeshGraphNets’ es la baja cantidad de datos que necesitan para iterar buenos resultados. Para este proyecto, se realiza una modificación previa a la arquitectura original, descrita por Pfaff et al. [10], para adaptarla a unas condiciones de contorno completamente diferentes.

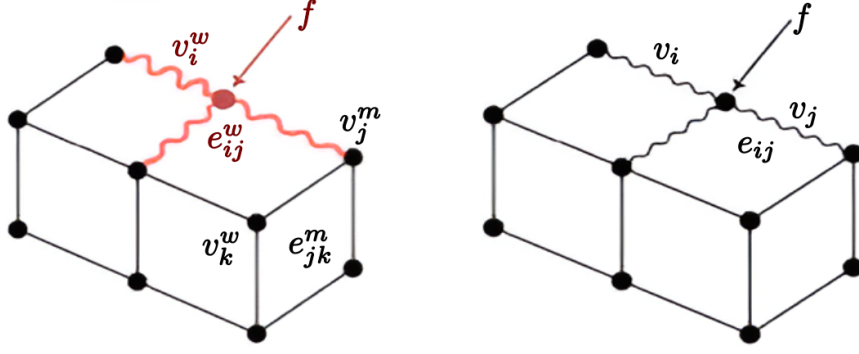


Figura 2.5: Comparativa entre un esquema típico de grafos con codificación de vértices y aristas (derecha), con el esquema propuesto (izquierda), el cual define el nodo de carga como actuador (v_i^w) y su conectividad como aristas globales que transmitirán la información del desplazamiento al modelo, e_{ij}^w . Mientras que el resto de conexiones que no se relacionen con el actuador compondrán la conectividad de la malla, e_{ij}^m .

En el artículo mencionado [10], la fuerza exterior se presenta como un contacto entre dos superficies, sin embargo, para este proyecto la carga exterior se define como un desplazamiento impuesto en un único punto. El nodo en el que se aplica esta condición de contorno de Dirichlet se representa como el actuador v_i^w , mientras que su conectividad da lugar a las aristas de conexión, denominadas “world edges” en el artículo original, e_{ij}^w . Los nodos restantes, junto con sus uniones, forman el subconjunto de nodos y aristas de la malla: v_j^m, v_k^m, e_{jk}^m . No obstante, al estar especificado la tipología de nodos como una etiqueta dentro de la variable v_i , se redefine la notación de las variables de entrada a la red como: v_i, e_{ij}^w, e_{ij}^m ,

$$\begin{aligned}
 \varepsilon_v : \mathbb{R}^{F_v} &\rightarrow \mathbb{R}^{F_h} \\
 \mathbf{v}_i &\longrightarrow \mathbf{x}_i \\
 \varepsilon_m : \mathbb{R}^{F_e} &\rightarrow \mathbb{R}^{F_h} \\
 \mathbf{e}_{ij}^m &\longrightarrow \mathbf{x}_{ij}^m \\
 \varepsilon_w : \mathbb{R}^{F_e} &\rightarrow \mathbb{R}^{F_h} \\
 \mathbf{e}_{ij}^w &\longrightarrow \mathbf{x}_{ij}^w
 \end{aligned} \tag{2.5}$$

La arquitectura parte de estas características iniciales (F_v representa la dimensión inicial de las variables almacenadas en el nodo y F_e de las variables almacenadas en las aristas) para codificar la información a un espacio latente de mayor dimensión (F_h): $x_i \in \mathbb{R}^{F_h}$, $x_{ij}^m \in \mathbb{R}^{F_h}$ y $x_{ij}^w \in \mathbb{R}^{F_h}$, donde los codificadores $\varepsilon_m, \varepsilon_w, \varepsilon_v$ se definen como MLPs, Ec. (2.5). A continuación se procesa la información en ambos tipos de aristas y se integra todo en una función de agregación encargada de distribuir la información a cada nodo según la matriz de conectividad. Este proceso define la transmisión de mensaje

en la red y es el encargado de informar tanto a los nodos cercanos como lejanos de la influencia de la carga:

$$\begin{aligned} \mathbf{x}_{ij}'^m &\leftarrow f^m(\mathbf{x}_{ij}^m, \mathbf{x}_i, \mathbf{x}_j), \\ \mathbf{x}_{ij}'^w &\leftarrow f^w(\mathbf{x}_{ij}^w, \mathbf{x}_i, \mathbf{x}_j), \\ \mathbf{x}_i' &\leftarrow f^v\left(\mathbf{x}_i, \sum_j \mathbf{x}_{ij}'^m, \sum_j \mathbf{x}_{ij}'^w\right), \end{aligned} \quad (2.6)$$

concretamente siendo f^m , f^w y f^v MLPs con conexiones residuales (Anexo E).

El procesamiento se repite un numero de pasos predefinido como hiperparámetro (M) para finalmente decodificarse , extrayendo directamente la variación temporal del sistema para cada variable de estado ($\mathbf{z}^{t+1}$, Figura 2.6).

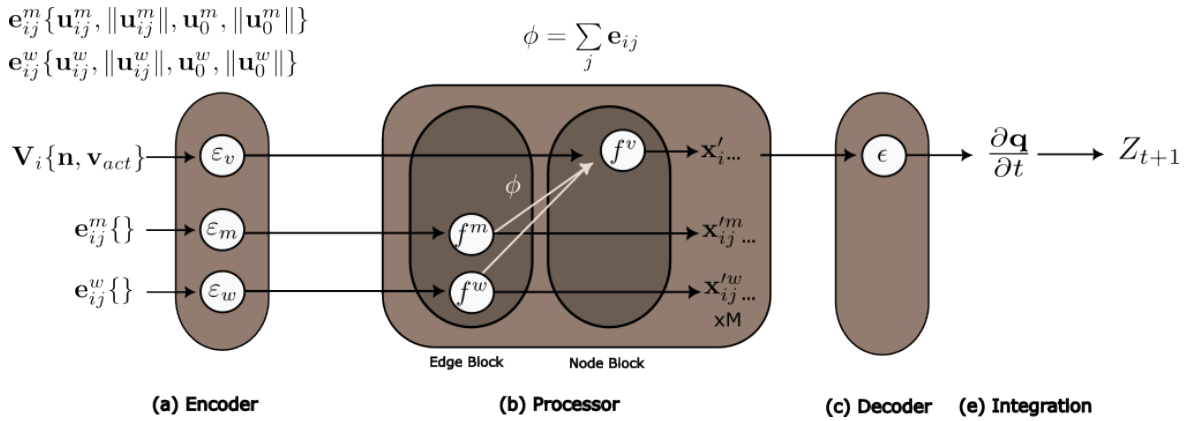


Figura 2.6: Diagrama de bloques del algoritmo utilizado para estimar la variable de estado en el siguiente instante de tiempo. (a) El codificador transforma las características de los nodos y las aristas a un espacio latente. (b) El procesador comparte la información nodal a través del grafo mediante M módulos de transmisión de mensajes. (c) El decodificador extrae la variación temporal de posición. (d) El integrador predice la posición para el siguiente paso de tiempo. Todo el proceso se repite iterativamente para obtener la dinámica del sistema físico.

No obstante, este planteamiento no es sólido con el objetivo final, el cual consiste en integrar tanto la posición como las velocidades y tensiones, especialmente en un enfoque viscoelástico donde las tensiones resultantes deben considerar la velocidad del modelo. Por lo que se propone otra modificación a la arquitectura existente donde se agrega en el espacio latente la información codificada de velocidad y tensión ($\epsilon_\sigma, \epsilon_{\dot{q}}$, denotan los codificadores respectivamente), concatenándola con el tensor nodal $\mathbf{V}_i$, según lo

propuesto en la Figura 2.7 y triplicando el tamaño del espacio latente $F_{h'} = 3F_h$,

$$\begin{aligned}
\varepsilon_S &: \mathbb{R}^{F_S} \rightarrow \mathbb{R}^{F_h} \\
\sigma_i &\longrightarrow \sigma'_i \\
\varepsilon_\sigma &: \mathbb{R}^{F_q} \rightarrow \mathbb{R}^{F_h} \\
\dot{\mathbf{q}}_i &\longrightarrow \dot{\mathbf{q}}'_i \\
\mathbf{x}_i &\longrightarrow [\varepsilon(\mathbf{v}_i), \sigma'_i, \dot{\mathbf{q}}'_i] \in \mathbb{R}^{F_{h'}}
\end{aligned}$$

En consecuencia se homogeneiza el tamaño de las agregaciones al mismo espacio latente $F_{h'}$ de la información en los vértices,

$$\begin{aligned}
\varepsilon_m &: \mathbb{R}^{F_e} \rightarrow \mathbb{R}^{F_{h'}} \\
\mathbf{e}_{ij}^m &\longrightarrow \mathbf{x}_{ij}^m \\
\varepsilon_w &: \mathbb{R}^{F_e} \rightarrow \mathbb{R}^{F_{h'}} \\
\mathbf{e}_{ij}^w &\longrightarrow \mathbf{x}_{ij}^w
\end{aligned}$$

Esta metodología agrega adicionalmente dos decodificadores con el objetivo de extraer la derivada temporal las diferentes variables de estado. Esta información se decodifica desde el espacio latente una vez procesados los pasos de la red de grafos, resultando en la salida de tres vectores que definen el diferencial de cada conjunto de variables de estado.

Para definir la función de coste de ambos modelos se calcula la norma 2 del error en la derivada temporal de las variables de estado

$$\mathcal{L}_{\text{data}_n} = \left\| \frac{\partial \mathbf{z}_{\text{GT}}}{\partial t} - \frac{\partial \mathbf{z}_{\text{net}}}{\partial t} \right\|_2^2 \quad (2.7)$$

Finalmente la función de coste global ($\mathcal{L}$) resulta en una media ponderada entre todos los lotes que componen cada época del entrenamiento,

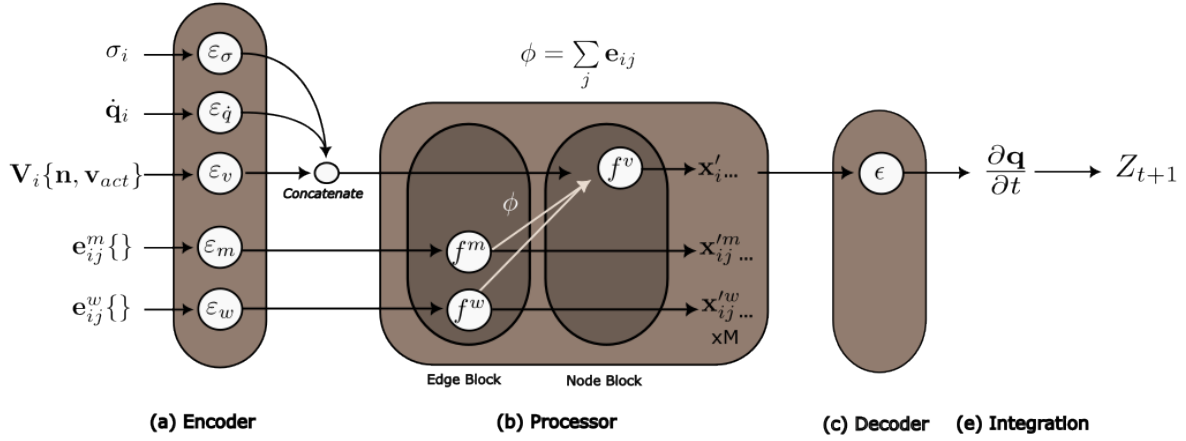


Figura 2.7: Diagrama de bloques del algoritmo utilizado para predecir un cambio de variable de estado de un solo paso en el tiempo. (a) El codificador transforma las características de los nodos y las aristas a un espacio latente. (b) El procesador comparte la información nodal a través del grafo mediante (M) módulos de paso de mensajes. (c) El decodificador triple extrae la variación temporal de los tres conjuntos de variables de estado. (d) El integrador predice la posición para el siguiente paso de tiempo. Todo el proceso se repite iterativamente para obtener la dinámica del sistema físico.

Después de calcular el coste para un lote de datos,

$$\mathcal{L} = \frac{1}{N_{\text{batch}}} \sum_{n=0}^{N_{\text{batch}}} \mathcal{L}_{\text{data}_n} \quad (2.8)$$

se aplica un proceso conocido como retropropagación (“backpropagation”)[23] para ajustar los parámetros del modelo y minimizar este coste, generalmente utilizando un algoritmo de optimización como Adam [24]. Cuando todos los lotes del conjunto de entrenamiento se han optimizado, se completa la época,

2.3.2. Redes de grafos híbridas

Las redes de grafos informadas por física [15] añaden un sesgo físico a la red de grafos, de esta forma y en conjunto con el sesgo geométrico, se asegura una convergencia mas eficiente hacia la solución que mejor satisfaga los datos de entrenamiento.

Formalismo GENERIC

Se puede introducir GENERIC como una generalización del paradigma hamiltoniano a sistemas disipativos [15, 25]. GENERIC afirma que la evolución temporal de un sistema que esta fuera de su equilibrio se expresa según la ecuación

$$\dot{\mathbf{z}} = \{\mathbf{z}, E\} + [\mathbf{z}, S] \quad (2.9)$$

La contribución conservativa se asume de forma hamiltoniana, con un potencial de energía $E = E(\mathbf{z})$ y un corchete de Poisson $\{\mathbf{z}, E\}$ que actúa sobre el vector de estado, representando la dinámica reversible en la mecánica clásica. La contribución irreversible al balance energético es generada por el potencial de entropía fuera del equilibrio $S = S(\mathbf{z})$, con un corchete irreversible o de fricción $[\mathbf{z}, S]$, [25].

Para facilitar su uso en un problema práctico se reescribe la notación de corchetes usando dos operadores lineales ($\mathbf{L}$ y $\mathbf{M}$), los cuales heredan las propiedades matemáticas de la formulación original,

$$\dot{\mathbf{z}} = \mathbf{L} \frac{\partial E}{\partial \mathbf{z}} + \mathbf{M} \frac{\partial S}{\partial \mathbf{z}} \quad (2.10)$$

Sin embargo, las restricciones matemáticas en $\mathbf{L}$ y $\mathbf{M}$ no son suficientes, por lo que para garantizar una descripción termodinámicamente consistente del sistema se introducen dos condiciones de degeneración $\{S, \mathbf{z}\} = [E, \mathbf{z}] = \mathbf{0}$. Estas condiciones aseguran la naturaleza reversible de la contribución hamiltoniana y la conservación de la energía total del sistema. Las condiciones de degeneración se expresan en forma matricial en términos de los operadores $\mathbf{L}$ y $\mathbf{M}$,

$$\mathbf{L} \frac{\partial S}{\partial \mathbf{z}} = \mathbf{M} \frac{\partial E}{\partial \mathbf{z}} = \mathbf{0} \quad (2.11)$$

por lo que se satisface, adicionalmente, la conservación de la energía y el aumento de la entropía que dictan la primera y segunda ley de la termodinámica,

$$\frac{dE}{dt} = 0 \quad (2.12)$$

$$\frac{dS}{dt} \geq 0 \quad (2.13)$$

Cuando el sistema planteado no es cerrado, por ejemplo en el caso de la existencia de acciones externas al sistema, se hace uso de formulaciones port-metripléticas. En el anexo F puede consultarse su definición e implementación.

Redes informadas por la termodinámica del sistema

La incorporación de GENERIC como sesgo físico a una red de grafos se denomina a partir de este punto como TIGNN o “Thermodynamics-Informed Graph Neural Networks” [15], con el mismo objetivo de construir un integrador temporal de la variable de estado z para un sistema excitado por un desplazamiento impuesto nodal, denotado como f_i .

Analogamente al planteamiento anterior, es necesario transformar nuestras variables de estado a un espacio latente de mayor dimensión, esta tarea se realiza con tres

codificadores compuestos por una MLP cada uno $(\varepsilon_v, \varepsilon_e, \varepsilon_f)$ para transformar las variables de estado, a partir de ahora denotadas como características de la red de grafos, a un espacio latente de mayores dimensiones $x_i \in \mathbb{R}^{F_h}$, $x_{ij} \in \mathbb{R}^{F_h}$ y $f_i \in \mathbb{R}^{F_h}$ respectivamente,

$$\begin{aligned}
\varepsilon_e &: \mathbb{R}^{F_e} \rightarrow \mathbb{R}^{F_h} \\
\mathbf{e}_{ij} &\longrightarrow \mathbf{x}_{ij} \\
\varepsilon_v &: \mathbb{R}^{F_v} \rightarrow \mathbb{R}^{F_h} \\
\mathbf{v}_i &\longrightarrow \mathbf{x}_i \\
\varepsilon_f &: \mathbb{R}^{F_f} \rightarrow \mathbb{R}^{F_h} \\
\mathbf{f}_i &\longrightarrow \mathbf{f}'_i
\end{aligned}$$

Esta información entra al procesador de la red de grafos, el cual tiene como función principal compartir estos datos entre los nodos y aristas de la red. Primero una MLP (π_e) calcula las nuevas características de las aristas (x'_{ij}) basándose en sus actuales características y en las de los dos nodos que lo forman,

$$\begin{aligned}
\pi_e &: \mathbb{R}^{3F_h+F_g} \rightarrow \mathbb{R}^{F_h} \\
\pi_e(\mathbf{x}_{ij}, \mathbf{x}_i, \mathbf{x}_j) &\longrightarrow \mathbf{x}'_{ij}
\end{aligned} \tag{2.14}$$

Posteriormente para cada nodo se agrupa esta información con una función invariante a permutación (ϕ) basada en la conectividad local $N_i = \{j \in V \mid (i, j) \in E\}$ del nodo i (o nodo receptor). Finalmente son implementadas a una segunda MLP (π_v) todas las características actuales del nodo junto con el resultado de la función de agregación y las cargas o desplazamientos externos.

$$\begin{aligned}
\pi_v &: \mathbb{R}^{2F_h+F_f+F_g} \rightarrow \mathbb{R}^{F_h} \\
\pi_v(\mathbf{x}_i, \phi(\mathbf{x}'_{ij}), \mathbf{f}'_i) &\longrightarrow \mathbf{x}'_i
\end{aligned} \tag{2.15}$$

Este paso es el equivalente a la transmisión del mensaje de un nodo adyacente al receptor. Para obtener la influencia de nodos más lejanos, se repite este bloque M veces.

Tras el procesado, los tensores resultantes para cada nodo se implementan como entrada al decodificador (ϵ), el cual se encarga de extraer la información física relevante del sistema. Concretamente para este proyecto y siguiendo el formalismo GENERIC impuesto, el decoder predice los potenciales de energía (E) y entropía (S), junto con

los operadores $\mathbf{l}$ y $\mathbf{m}$ en forma de vector[15].

$$\begin{aligned} \epsilon &: \mathbb{R}^{F_h} \rightarrow \mathbb{R}^{F_y} \\ \mathbf{x}_i' &\longrightarrow \mathbf{y}_i = (\mathbf{l}, \mathbf{m}, \mathbf{E}, \mathbf{S}) \end{aligned}$$

Finalmente para formar el integrador, se reparametrizan los operadores aplanados para obtener los operadores lineales ($\mathbf{L}$ y $\mathbf{M}$), respetando la antisimetría de la matriz $\mathbf{L}$ y la semi-definición positiva de la matriz $\mathbf{M}$:

$$\mathbf{L} = \mathbf{l} - \mathbf{l}^T, \quad \mathbf{M} = \mathbf{m}\mathbf{m}^T$$

Se genera por medio de esta metodología un integrador termodinámicamente estable, Ec. (2.16), Figura 2.8, que permite predecir el comportamiento temporal de las variables de estado de un sistema no lineal,

$$\dot{\mathbf{z}}_{t+1} = \dot{\mathbf{z}}_t + \Delta t \left(\mathbf{L} \frac{\partial E}{\partial \mathbf{z}_t} + \mathbf{M} \frac{\partial S}{\partial \mathbf{z}_t} \right) \quad (2.16)$$

Debido a esta estructura impuesta no se debe evaluar únicamente la pérdida de datos relativa a las variables de estado, sino que hay que asegurar que durante el entrenamiento se cumplan las condiciones de degeneración definidas anteriormente. En este caso, la función multicoste $\mathcal{L}$ (Ecuación 2.19) estará compuesta por un término que estimará la norma 2 del error en la derivada temporal,

$$\mathcal{L}_{\text{data}_n} = \left\| \frac{\partial \mathbf{z}_{\text{GT}}}{\partial t} - \frac{\partial \mathbf{z}_{\text{net}}}{\partial t} \right\|_2^2 \quad (2.17)$$

mientras que otro término evaluará, también con la norma 2, el coste generado por los términos de degeneración,

$$\mathcal{L}_{\text{deg}_n} = \left\| \mathbf{L} \frac{\partial S}{\partial \mathbf{z}_n} \right\|_2^2 + \left\| \mathbf{M} \frac{\partial E}{\partial \mathbf{z}_n} \right\|_2^2 \quad (2.18)$$

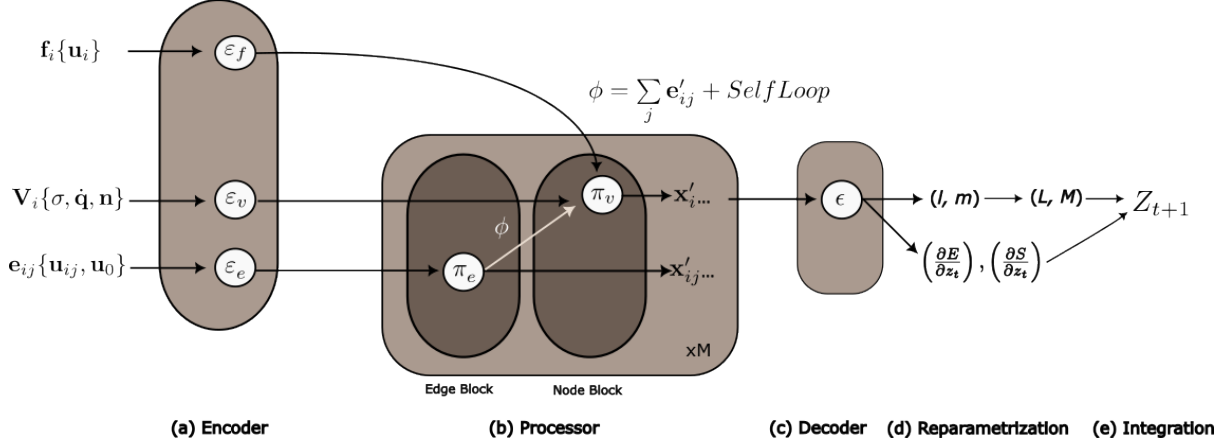


Figura 2.8: Diagrama de bloques del algoritmo utilizado para predecir un cambio de variable de estado de un solo paso en el tiempo. (a) El codificador transforma las características de los nodos y las aristas a un espacio latente. (b) El procesador comparte la información nodal a través del grafo mediante M módulos de paso de mensajes. (c) El decodificador extrae los potenciales de energía y entropía, además de los operadores aplanados. (d) Se reparametrizan los operadores L y M y se calcula los gradientes del potencial con respecto a la entrada de la red. (e) El integrador predice las variables de estado del siguiente paso en el tiempo basándose en la formulación GENERIC. Todo el proceso se repite iterativamente para obtener la dinámica del sistema físico [15].

Finalmente la función de coste global ($\mathcal{L}$) resulta en una media ponderada entre todos los lotes que componen cada época del entrenamiento

$$\mathcal{L} = \frac{1}{N_{\text{batch}}} \sum_{n=0}^{N_{\text{batch}}} (\lambda \mathcal{L}_{\text{data}_n} + \mathcal{L}_{\text{deg}_n}) \quad (2.19)$$

Como se ha explicado anteriormente, tras calcular el coste para un lote de datos, se aplica un proceso conocido como retropropagación (“backpropagation”) [23] para ajustar los parámetros del modelo y minimizar este coste, generalmente utilizando un algoritmo de optimización como Adam [24]. Cuando todos los lotes del conjunto de entrenamiento se han optimizado, se completa la época.

Como punto final a la metodología, se incluye información mas detallada sobre el marco teórico en el Anexo E, mientras que los pseudo-códigos de entrenamiento y testeo para ambas redes están disponibles en el Anexo D. De la misma forma las especificaciones técnicas estarán disponibles en el Anexo M.

Capítulo 3

Resultados

3.1. Redes de grafos guiadas únicamente por datos: modelo simplificado

En esta sección se describen los resultados obtenidos para el integrador de posición con redes de grafos. Definiendo el conjunto de datos de entrenamiento, validación y testeo con 85 simulaciones, con 20 pasos de tiempo cada una.

Como primer punto se define un barrido estocástico de hiperparámetros, Anexo A, para obtener información sobre la sensibilidad de la red. Es de remarcar que el barrido no optimiza el coste en validación, sino que optimiza una métrica propia (γ , también denominado como MRE) que mide el error relativo medio durante el “rollout”,

$$\gamma = \frac{1}{N} \sum_{n=0}^N \frac{\|\mathbf{q}^{GT} - \mathbf{q}^{net}\|_2}{\|\mathbf{q}^{GT}\|_2} \quad (3.1)$$

siendo N el numero de pasos por el numero de simulaciones en el conjunto de test.

3.1.1. Modelo resultante y métricas

En la Tabla 3.1 se exponen los dos mejores modelos seleccionados del barrido de hiperparámetros descrito en el Anexo A.

Tabla 3.1: Métricas e hiperparámetros de los dos mejores ensayos para el integrador de desplazamientos guiado por datos. Respecto a los hiperparámetros: *d.o.* se refiere a la dimensión del espacio latente y de la capa oculta de las MLPs, *ruido* al valor que escala la distribución gaussiana de ruido, *pasos* a la cantidad de pasos que se iteran en el procesador de grafos.

Entrenamiento	<i>d.o.</i>	<i>lr</i>	ruido	pasos	γ	MSE_{train}	MSE_{val}
3h 1’	150	0.0001	0.001	10	5.47e-04	3.04e-05	7.83e-04
2h 49’	100	0.0001	0.001	10	5.59e-04	6.81e-05	4.20e-04

El gráfico de caja mostrado en la Figura 3.1, verifica que el modelo predice peor en inferencia, lo que es natural debido a que son posiciones de carga no vistas durante el

entrenamiento. Por lo que la red extrapola la información de los casos vistos y predice para los conjuntos de validación y testeo.

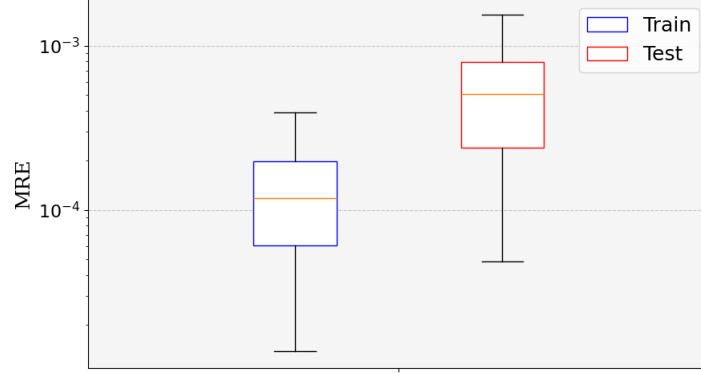


Figura 3.1: Gráfico de caja que compara la inferencia durante el rollout para los datos de entrenamiento y testeo en la red de grafos guiada por datos propuesta para construir el integrador de desplazamientos.

Adicionalmente en el Anexo A se muestran más resultados.

3.2. Redes de grafos guiadas únicamente por datos: modelo general

En este apartado se describen los resultados del integrador informado por datos para el mismo conjunto de entrenamiento, test y validación ya definido.

Con un enfoque equivalente al integrador de posición, se definen tres métricas de error relativo promedio para cada conjunto de variables de estado que comparten las mismas unidades físicas durante el rollout, Ecs. (3.2) - (3.5), donde la variable a optimizar durante el barrido es una media aritmética de las tres,

$$\gamma_q = \frac{1}{N} \sum_{n=0}^N \frac{\|\mathbf{q}^{GT} - \mathbf{q}^{net}\|_2}{\|\mathbf{q}^{GT}\|_2} \quad (3.2)$$

$$\gamma_{\dot{\mathbf{q}}} = \frac{1}{N} \sum_{n=0}^N \frac{\|\dot{\mathbf{q}}^{GT} - \dot{\mathbf{q}}^{net}\|_2}{\|\dot{\mathbf{q}}^{GT}\|_2} \quad (3.3)$$

$$\gamma_{\sigma} = \frac{1}{N} \sum_{n=0}^N \frac{\|\sigma^{GT} - \sigma^{net}\|_2}{\|\sigma^{GT}\|_2} \quad (3.4)$$

$$\gamma = \frac{\gamma_q + \gamma_{\dot{\mathbf{q}}} + \gamma_{\sigma}}{3} \quad (3.5)$$

siendo N el numero de pasos por el numero de simulaciones en el conjunto de test.

3.2.1. Modelo resultante y métricas

Después de analizar todas las etapas de exploración (Anexo B), se elige un grupo de modelos representativos para las redes guiadas por datos, los cuales están configurados con la siguiente combinación de hiperparámetros:

Tabla 3.2: Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las redes guiadas por datos. Respectivamente: *d.o.* equivale a la dimensión del espacio latente y de la capa oculta de las MLPs, mientras que los parámetros con ε_c se relacionan con la constante de ruido y los de ε_p con un valor proporcional. Los dos primeros modelos tienen aproximadamente tres millones de parámetros entrenables, mientras que los dos últimos con once millones.

d.o.	$\varepsilon_p\sigma$	$\varepsilon_c\sigma$	ε_pq	ε_cq	$\varepsilon_p\dot{q}$	$\varepsilon_c\dot{q}$	pasos	γ_σ	γ_q	$\gamma_{\dot{q}}$
70	0	0.5	0.005	0	0	0.5	12	0.2679	0.0008	0.0702
70	0	0.5	0.005	0	0.005	0.5	12	0.2762	0.0009	0.0741
150	0.1	0.5	0	0	0.005	0.5	12	0.2539	0.0006	0.0722
150	0.1	0.5	0	0	0.005	0.5	12	0.2584	0.0006	0.0733

Obteniendo valores de errores relativos en posición semejantes al modelo simplificado, pero ligeramente superiores para velocidades y tensiones. No obstante y en base a las representaciones expuestas en el Anexo B se puede validar que la inferencia resultante es realista y semejante a su solución numérica. Siendo destacable que el error se acumula en los concentradores de tensión propios de la zona de contacto para todas las arquitecturas y simulaciones.

3.3. Inteligencia artificial híbrida: modelo general

Análogamente al planteamiento anterior, la métrica a optimizar durante el barrido sera γ , Ec. (3.5) y su análisis estará descrito al detalle en el Anexo C.

3.3.1. Modelos resultantes y métricas

Tras condensar los resultados de todos los barridos, se selecciona finalmente un conjunto de modelos representativos para las redes híbridas con la siguiente combinación de hiperparámetros:

Tabla 3.3: Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las híbridas. Respectivamente: $d.o.$ equivale a la dimensión del espacio latente y de la capa oculta de las MLPs, mientras que los parámetros con ε_c se relacionan con la constante de ruido y los de ε_p con un valor proporcional. Los tres primeros modelos tienen aproximadamente tres millones de parámetros entrenables, mientras que los dos últimos con once millones.

d.o.	λ_d	$\varepsilon_p\sigma$	$\varepsilon_c\sigma$	$\varepsilon_p q$	$\varepsilon_c q$	$\varepsilon_p \dot{q}$	$\varepsilon_c \dot{q}$	pasos	γ_σ	γ_q	$\gamma_{\dot{q}}$
125	50	0.05	0.5	0.005	0.1	0.005	0.5	12	0.2279	0.0023	0.0797
150	100	0.1	0.5	0.1	0.01	0.1	0.5	12	0.2293	0.0025	0.0760
150	100	0.001	0.0	0.05	0.0	0.1	0.0	12	0.2910	0.0007	0.0744
250	100	0.05	0.1	0.005	0.1	0	0.1	16	0.258	0.0010	0.0711
250	50	0.05	0.1	0	0.1	0.005	0.1	16	0.244	0.0013	0.074

Adicionalmente, en el Anexo C están expuestas una mayor cantidad de métricas que pueden dar una mejor visión del rendimiento del algoritmo. En la Tabla 3.4, también sobresalen otras redes híbridas más masivas que han logrado las métricas de error más destacadas entre todos los modelos evaluados. Rondando su tamaño entre 16 y 45 millones de parámetros entrenables.

Tabla 3.4: Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las híbridas con mayores cantidades de parámetros entrenables.

d.o.	λ_d	$\varepsilon_p\sigma$	$\varepsilon_c\sigma$	$\varepsilon_p q$	$\varepsilon_c q$	$\varepsilon_p \dot{q}$	$\varepsilon_c \dot{q}$	pasos	γ_σ	γ_q	$\gamma_{\dot{q}}$
500	100	0.05	0.1	0	0.1	0.005	0.1	16	0.24	0.0006	0.066
350	50	0.1	0.1	0.005	0.1	0.005	0.1	25	0.24	0.0007	0.0678

3.4. Estudio de robustez frente a cambios de geometría

Una de las premisas respecto a esta tecnología es la robustez frente a cambios de malla y de geometría. Por lo que se establecen un total de cuatro geometrías de trabajo que expresan diferencias tanto el mallado como en su anatomía (Figura 3.2).

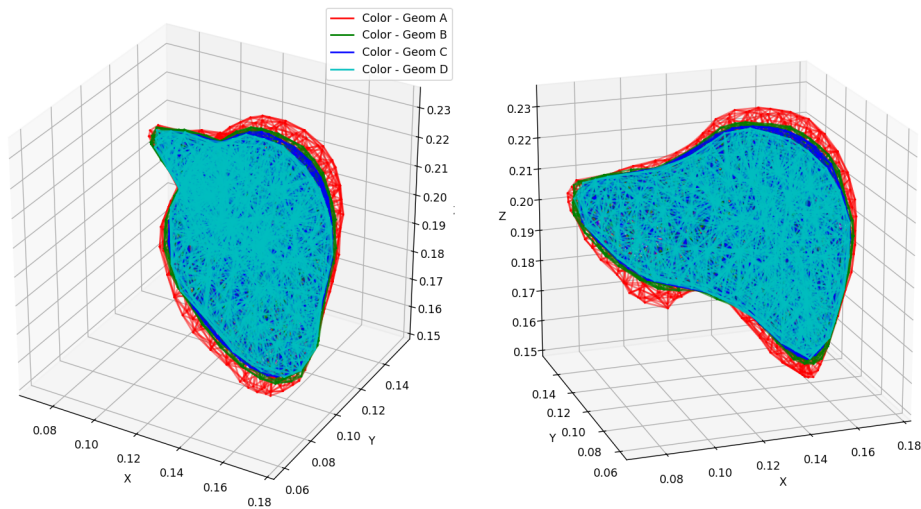


Figura 3.2: Comparativa de las cuatro geometrías superpuestas con su respectivo mallado.

Definiendo en su conjunto este grupo de geometrías por sus características como:

Tabla 3.5: Las características de la base de datos están determinadas por dos factores principales: el número total de nodos y la variación de volumen con respecto a la geometría inicial.

Geometría	Nodos	% Variación en Volumen
A	530	-
B	460	15.75
C	492	25.08
D	639	27.5

Siendo la Geometría A la utilizada para los ensayos “in silico” que componen las bases de datos que constan de una única geometría.

3.4.1. Inferencia en entrenamientos sin variabilidad geométrica

Para los entrenamientos con una sola geometría vista (la Geometría A), se evalúa su robustez ante cambios de volumen y cargas no vistas. Resultando en las métricas expuestas con mayor de talle en el Anexo H y en la comparativa definida para la Figura 3.3, donde se evalúa la diferencia entre el error relativo del despliegue dinámico de cada simulación segun el punto de aplicación de la carga, tal y como expresa la métrica 3.6, siendo γ_{DD} el error relativo de cada simulación para una red únicamente guiada por datos, mientras que γ_{Hib} corresponde al error relativo de la red híbrida,

$$\gamma_{dif} = \|\gamma_{DD} - \gamma_{Hib}\| \quad (3.6)$$

Obteniendo mejores métricas de error en posición y velocidad para las redes guiadas por datos, mientras que las redes híbridas muestran un mayor rendimiento en lo que respecta a la predicción del campo de tensiones.

3.4.2. Inferencia en entrenamientos con variabilidad geométrica

Al implementar la misma comparativa para entrenamientos con mayor variabilidad de geometrías (Geometrías A, B, C y evaluadas en D), se obtienen resultados también altamente relevantes.

Tanto para la arquitectura guiada por datos, como para la híbrida, se describe en el Anexo L un decremento de error notable al entrenar con bases de datos multi-variadas en geometría y inferir en nuevas geometrías no vistas. **Alcanzando mejores rendimientos con modelos hasta once veces mas pequeños.**

La metodología híbrida se diferencia positivamente para este punto debido a un mayor aumento de la robustez. Como se puede verificar en la Figura 3.4 (en mayor detalle expuesta en el Anexo L), su error en inferencia es menor para dos de los tres grupos de variables de estado.

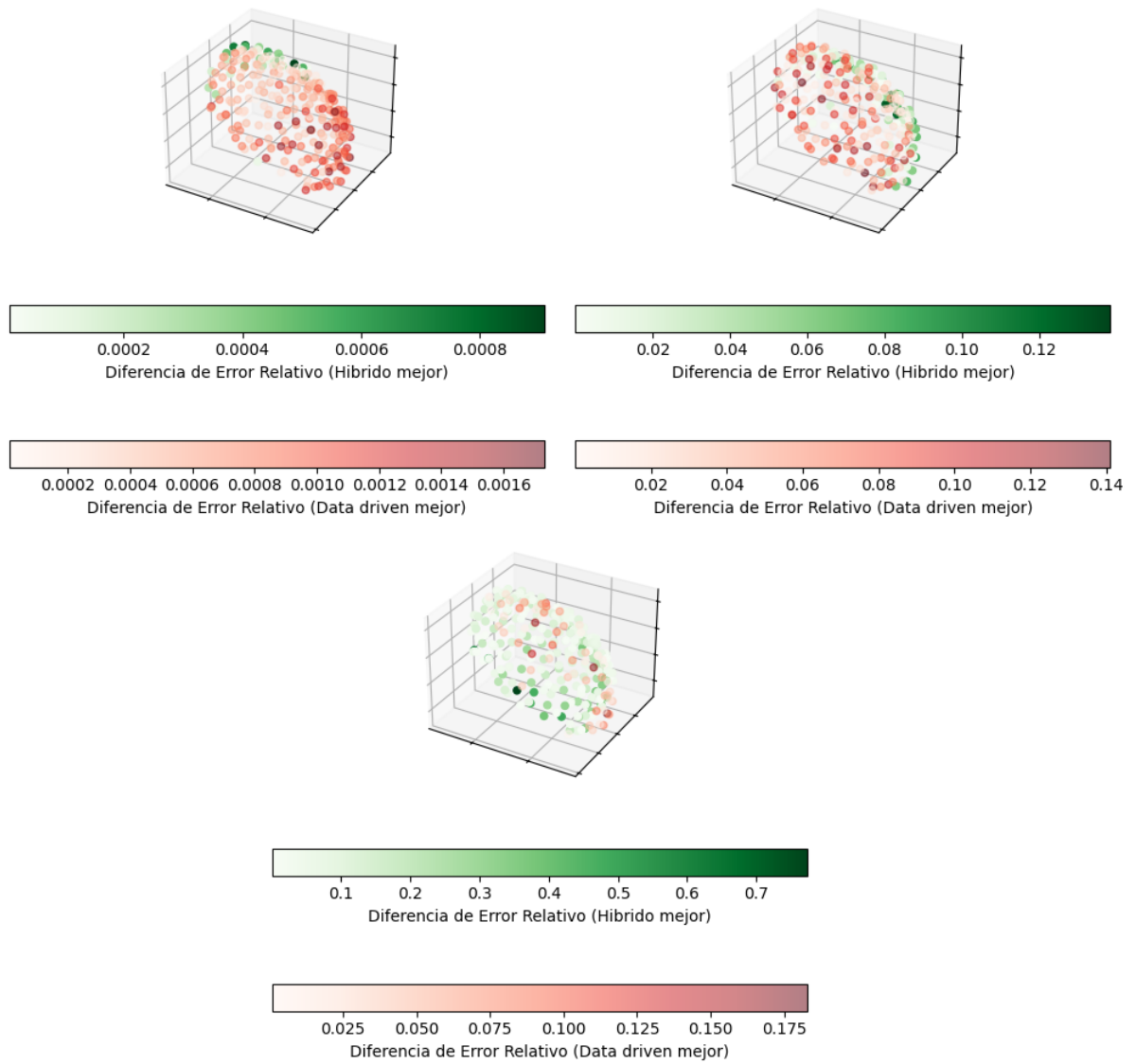


Figura 3.3: Comparativa tridimensional que representa la diferencia entre el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para el conjunto de geometrías no vistas (B, C y D) de las redes guiadas por datos e híbridas. A la izquierda para posición, en el derecha para velocidad y en la parte inferior para tensiones, siendo en verde los modelos con mejor inferencia para arquitecturas híbridas, mientras que en rojo tienen un mejor rendimiento los guiados por datos.

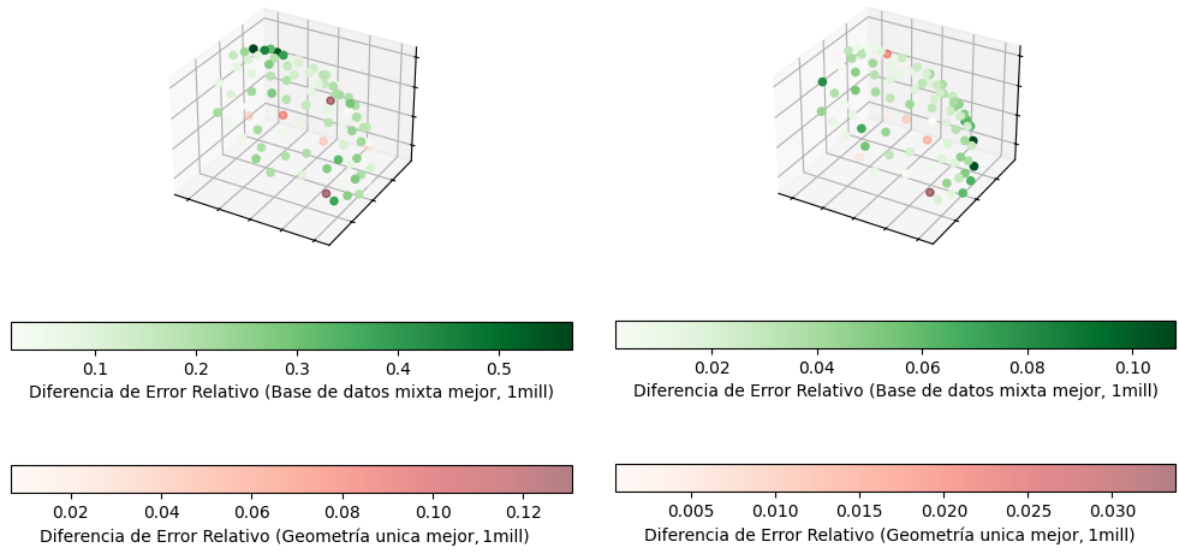


Figura 3.4: Comparativa tridimensional que representa la diferencia entre el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para la geometría no vista (D) de las redes híbridas y guiadas por datos entrenadas con bases de datos de geometrías mixtas. A la izquierda para tensiones y en el derecha para velocidad, siendo en verde los modelos con mejor inferencia para bases de datos con geometrías mixtas, mientras que en rojo tienen un mejor rendimiento los modelos con geometrías únicas.

Capítulo 4

Conclusiones

Contextualizando las conclusiones del proyecto, se han abordado diferentes puntos que engloban, en su conjunto, el desarrollo de algoritmos basados en inteligencia artificial que puedan describir y entender la compleja biomecánica de un tejido hepático.

Los barridos de hiperparámetros realizados para todas las arquitecturas (Anexos A, B y C), han revelado que un aumento en el número de pasos de la red de grafos conlleva a una mejora en la inferencia. Validando la hipótesis principal de la transmisión de mensaje que define a las redes de grafos. Además, confirmando el efecto negativo del sobre-ajuste en la predicción del rollout, lo que enfatiza la importancia de mecanismos que doten al modelo de mayor robustez.

El ruido, como herramienta de aumento de datos, ha evidenciado su utilidad para evitar dicho sobre-ajuste (Anexos B y C). Asimismo, la estandarización se ha identificado como la más efectiva entre todas las normalizaciones implementadas (Anexo G).

El estudio de las redes únicamente guiadas por datos ha validado su capacidad para deducir la física de un sistema, de manera robusta, a través de los propios datos de entrenamiento. Resaltando su notable estabilidad al predecir los desplazamientos del sistema. No obstante, su entrenamiento es largo y su convergencia está definida por tiempos superiores a los modelos híbridos.

En cuanto a las redes híbridas, se ha encontrado que, a pesar de diferir del esquema de caja negra, igualan e incluso superan los resultados de las redes guiadas por datos. Obteniendo mejores métricas respecto a la estimación de tensiones, pero introduciendo ligeramente más ruido en posición y velocidad cuando se infiere en bases de datos pequeñas sin mucha variabilidad.

Los modelos más masivos generan buenas inferencias para bases de datos entrenadas y representadas con una sola geometría, donde los datos son más semejantes entre si, sin embargo cuando queremos predecir sobre geometrías no vistas, el error se escala independientemente de que arquitectura o planteamiento utilizamos.

Una de las hipótesis principales del proyecto define que las redes optan por dos caminos, si los datos son semejantes, la red aprende a simplificarlos e infiere buenos resultados cuando le preguntas por situaciones semejantes. Pero cuando los datos no son tan similares entre sí, la red se ve obligada a aprender la física (o al menos una representación no lineal de ella), ya que es su única forma de generalizar entre los datos de entrenamiento, siendo capaz ahora de inferir en situaciones distantes de lo que ya ha visto.

Esta hipótesis se nutre de como la predicción de una geometría no vista es mejor para modelos entrenados con varias geometrías, que los entrenados con una única (en igualdad de número de simulaciones). Destacando como modelos once veces más pequeños infieren errores menores, y destacando también cómo el sesgo híbrido puede ser el responsable de incrementar esta mejora.

Otros puntos no menos destacables del proyecto definen como aunque el cómputo del modelo crece exponencialmente con el número de nodos, los datos de error se mantienen robustos ante redes de mayor densidad (Anexo K). Recalcando que el error relativo a estas redes es invariable ante el mallado, a diferencia que los elementos finitos. Ligado a este argumento, la velocidad de cálculo de la red respecto a la simulación numérica ha sido hasta 230 veces superior. Posicionándose como un posible sustituto en entornos hápticos que busquen frecuencias de respuesta mucho mayores a las proporcionadas por los métodos numéricos ya conocidos (Anexo J).

En contraste a estos resultados, las arquitecturas híbridas con historia de la evolución temporal del modelo, es decir, con cierta recurrencia, no han resultado óptimas debido a un sobre-ajuste excesivo de los datos (Anexo I). Aunque es factible que se puedan encontrar otros mecanismos que mejoren la convergencia por medio de la historia temporal.

Capítulo 5

Bibliografía

- [1] Serg Masís. *Interpretable Machine Learning with Python - Second Edition: Build explainable, fair, and robust high-performance models with hands-on, real-world examples*. 2nd ed. edition, 2024.
- [2] Francisco Chinesta, Pierre Ladeveze, and Elias Cueto. A short review on model order reduction based on proper generalized decomposition. *Archives of Computational Methods in Engineering*, 2011.
- [3] Sina Niroomandi, Diego González, Imanol Alfaro, Fabien Bordeu, Adrien Leygue, Elias Cueto, and Francisco Chinesta. Real-time simulation of biological soft tissues: a pgd approach. *International Journal for Numerical Methods in Biomedical Engineering*.
- [4] David González, Alberto García-González, Francisco Chinesta, and Elías Cueto. A data-driven learning method for constitutive modeling: Application to vascular hyperelastic soft tissues. *Materials*, 13(10), 2020.
- [5] Giovanni C. Silva, Vinicius C. Beber, and Diogo B. Pitz. Machine learning and finite element analysis: An integrated approach for fatigue lifetime prediction of adhesively bonded joints. 2021.
- [6] Rishith E. Meethal, Anoop Kodakkal, Mohamed Khalil, Aditya Ghantasala, Birgit Obst, Kai-Uwe Bletzinger, and Roland Wüchner. Finite element method-enhanced neural network for forward and inverse problems. 2023.
- [7] Liang Liang, Minliang Liu, Caitlin Martin, and Wei Sun. A deep learning approach to estimate stress distribution: a fast and accurate surrogate of finite-element analysis. *PubMed*, 2018.
- [8] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv*, 2021.

- [9] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, and et al. Physics-informed machine learning. *Nat Rev Phys*, 3:422–440, 2021.
- [10] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks. *Deepmind*, 2024.
- [11] Harpreet Sethi, Mohammad Nabian, Ram Cherukuri, Luca Pegolotti, Eric Darve, and Alison Marsden. Enabling greater patient-specific cardiovascular care with ai surrogates. *Journal of Cardiovascular Engineering and Technology*, November 2023.
- [12] Beatriz Moya, Iciar Alfaro, David Gonzalez, Francisco Chinesta, and Elías Cueto. Physically sound, self-learning digital twins for sloshing fluids. 2020.
- [13] A. Badias, I. Alfaro, D. Gonzalez, F. Chinesta, and E. Cueto. Real-time interaction of virtual and physical objects in mixed reality applications. *International Journal for Numerical Methods in Engineering*, 2020.
- [14] Quercus Hernández, Alberto Badiás, David González, Francisco Chinesta, and Elías Cueto. Structure-preserving neural networks. *Journal of Computational Physics*, page 109950, 2020.
- [15] Quercus Hernandez, Alberto Badias, Francisco Chinesta, and Elias Cueto. Thermodynamics-informed graph neural networks. *IEEE Transactions on Artificial Intelligence*, 2022.
- [16] Quercus Hernández, Alberto Badiás, Francisco Chinesta, and Elías Cueto. Thermodynamics-informed neural networks for physically realistic mixed reality. *Computer Methods in Applied Mechanics and Engineering*, 407:115912, 2023. ArXiv preprint, Download from the publisher, Open Access.
- [17] Blender Foundation. *Blender*, 1994.
- [18] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, USA, 3rd edition, 2009.
- [19] Christophe Geuzaine and Jean-François Remacle. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 2018.
- [20] Paolo Cignoni, Massimiliano Corsini, and Guido Ranzuglia. Meshlab: an open-source 3d mesh processing system. *ERCIM News*, 2008.

- [21] Dassault Systèmes. Abaqus: Finite Element Analysis Software, 1978.
- [22] MathWorks, Inc. MATLAB. <https://es.mathworks.com/products/matlab.html>, R2023a.
- [23] Satyen Kale, Ayush Sekhari, and Karthik Sridharan. Sgd: The role of implicit regularization, batch-size and multiple-epochs, 2021.
- [24] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2017.
- [25] Miroslav Grmela and Hans Christian Öttinger. Dynamics and thermodynamics of complex fluids. i. development of a general formalism. *Phys. Rev. E*, 56:6620–6632, Dec 1997.
- [26] W.S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5, 1943.
- [27] Frank Rosenblatt. The perceptron, a perceiving and recognizing automaton project. *Cornell Aeronautical Laboratory*, 85, 1957.
- [28] Roberto Tornero Costa and Sandra Martínez Sanchís. Desarrollo de modelos biomécánicos del hígado mediante método de elementos finitos, 2020.
- [29] Erin Trefts, Michelle Gannon, and David H. Wasserman. The liver. *Curr Biol*, 2017.
- [30] Stéphanie Marchesseau, Simon Chatelin, and Hervé Delingette. Nonlinear biomechanical model of the liver. In Yohan Payan and Jacques Ohayon, editors, *Biomechanics of Living Organs*, pages 243–265. Elsevier, 2017.
- [31] F. Martínez Martínez. *Determining the Biomechanical Behavior of the Liver Using Medical Image Analysis and Evolutionary Computation*. PhD thesis, Universitat Politècnica de València, 2014.
- [32] Y. C. Fung. *Biomechanics: Mechanical Properties of Soft Tissues*. Springer-Verlag, 2nd edition, 1996.
- [33] K. Miller. Constitutive modelling of abdominal organs. *Journal of Biomechanics*, 33(3):367–373, 2000.
- [34] E. O. Carew, J. E. Barber, and I. Vesely. Role of preconditioning and recovery time in repeated testing of aortic valve tissues: Validation through quasilinear viscoelastic theory. *Annals of Biomedical Engineering*, 28:1093–1100, 2000.

- [35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An open source deep learning platform, 2019.
- [36] Abhimanyu Das, Andrew Leach, Rajat Sen, Rose Yu, and Weihao Kong. Long horizon forecasting with tide: Time-series dense encoder. *Transactions on Machine Learning Research*, 2023.
- [37] Hans Christian Öttinger. Nonequilibrium thermodynamics for open systems. *Physical Review E*, 73(3):036126, Mar 2006.
- [38] Quercus Hernández, Alberto Badiás, Francisco Chinesta, and Elías Cueto. Port-metriplectic neural networks: thermodynamics-informed machine learning of complex physical systems. *Computational Mechanics*, 72(3):553–561, March 2023.
- [39] A. Moses Badlyan, B. Maschke, C. Beattie, and V. Mehrmann. Open physical systems: From generic to port-hamiltonian systems. *arXiv preprint arXiv:1804.04064*, 2018.

Lista de Figuras

1.1. Esquematización que compara, respecto a otras dos ramas más tradicionales, la idea detrás de la inteligencia artificial híbrida. Incorporando la física de los métodos tradicionales y el aprendizaje guiado por datos, encontrando un balance que optimice el rendimiento del sistema.	2
1.2. Representación del efecto del ruido sobre el despliegue temporal de una variable de estado cualquiera, evitando que la acumulación de error genere divergencias o grandes errores relativos.	5
2.1. Modelo tridimensional del hígado reducido a una menor resolución poligonal con Blender. El total de nodos presentes en la malla es de 530 y su escala es 1/1 con la real.	7
2.2. Esquematización del proceso de creación y preprocesado de la base de datos.	8
2.3. Representación visual de la base de datos empleada. Los nodos en rojo muestran los nodos encastrados, mientras que se dibujan en conjunto todos los diferentes puntos de aplicación de los desplazamientos con sus respectivas normales, siendo para cada cada ensayo un único punto de aplicación y un único desplazamiento impuesto.	10
2.4. Esquematización del como la red entrena sobre una integración temporal entre dos pseudo-tiempos del modelo, sin embargo para validar estos resultados se calcula la integración temporal completa del modelo. . .	10
2.5. Comparativa entre un esquema típico de grafos con codificación de vértices y aristas (derecha), con el esquema propuesto (izquierda), el cual define el nodo de carga como actuador (v_i^w) y su conectividad como aristas globales que transmitirán la información del desplazamiento al modelo, e_{ij}^w . Mientras que el resto de conexiones que no se relacionen con el actuador compondrán la conectividad de la malla, e_{ij}^m	12

2.6.	Diagrama de bloques del algoritmo utilizado para estimar la variable de estado en el siguiente instante de tiempo. (a) El codificador transforma las características de los nodos y las aristas a un espacio latente. (b) El procesador comparte la información nodal a través del grafo mediante M módulos de transmisión de mensajes. (c) El decodificador extrae la variación temporal de posición. (d) El integrador predice la posición para el siguiente paso de tiempo. Todo el proceso se repite iterativamente para obtener la dinámica del sistema físico.	13
2.7.	Diagrama de bloques del algoritmo utilizado para predecir un cambio de variable de estado de un solo paso en el tiempo. (a) El codificador transforma las características de los nodos y las aristas a un espacio latente. (b) El procesador comparte la información nodal a través del grafo mediante (M) módulos de paso de mensajes. (c) El decodificador triple extrae la variación temporal de los tres conjuntos de variables de estado. (d) El integrador predice la posición para el siguiente paso de tiempo. Todo el proceso se repite iterativamente para obtener la dinámica del sistema físico.	15
2.8.	Diagrama de bloques del algoritmo utilizado para predecir un cambio de variable de estado de un solo paso en el tiempo. (a) El codificador transforma las características de los nodos y las aristas a un espacio latente. (b) El procesador comparte la información nodal a través del grafo mediante M módulos de paso de mensajes. (c) El decodificador extrae los potenciales de energía y entropía, además de los operadores aplanados. (d) Se reparametrizan los operadores L y M y se calcula los gradientes del potencial con respecto a la entrada de la red. (e) El integrador predice las variables de estado del siguiente paso en el tiempo basándose en la formulación GENERIC. Todo el proceso se repite iterativamente para obtener la dinámica del sistema físico [15].	19
3.1.	Gráfico de caja que compara la inferencia durante el rollout para los datos de entrenamiento y testeo en la red de grafos guiada por datos propuesta para construir el integrador de desplazamientos.	21
3.2.	Comparativa de las cuatro geometrías superpuestas con su respectivo mallado.	24

3.3.	Comparativa tridimensional que representa la diferencia entre el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para el conjunto de geometrías no vistas (B, C y D) de las redes guiadas por datos e híbridas. A la izquierda para posición, en el derecha para velocidad y en la parte inferior para tensiones, siendo en verde los modelos con mejor inferencia para arquitecturas híbridas, mientras que en rojo tienen un mejor rendimiento los guiados por datos.	26
3.4.	Comparativa tridimensional que representa la diferencia entre el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para la geometría no vista (D) de las redes híbridas y guiadas por datos entrenadas con bases de datos de geometrías mixtas. A la izquierda para tensiones y en el derecha para velocidad, siendo en verde los modelos con mejor inferencia para bases de datos con geometrías mixtas, mientras que en rojo tienen un mejor rendimiento los modelos con geometrías únicas.	27
A.1.	Barrido de hiperparámetros para el modelo simplificado con las MeshGraphNets. Resultando en mejores valores de error para bajos valores de la tasa de aprendizaje, así como para modelos con mayor cantidad de pasos. Siendo deficientes los resultados para modelos con menos de cinco pasos.	47
A.2.	Barrido de hiperparámetros para el modelo de integración de posición con las MeshGraphNets tras filtrar los mejores entrenamientos. Reafirmando la hipótesis de que los modelos con mayor espacio latente y mayor cantidad de pasos convergen mejor.	48
A.3.	Barrido de hiperparámetros para el modelo de integración de posición con las MeshGraphNets tras filtrar los mejores entrenamientos. Validando el ruido intra-entreno no solo evita el sobre-ajuste, sino que mejora la inferencia durante el rollout.	49
A.4.	Comparativa gráfica entre la solución numérica y la predicción de la red para una posición de carga no vista en el entrenamiento, con una representación en escala de color de la norma del desplazamiento en metros. Los nodos oscuros, visibles para ambos modelos, están definidos como apoyos. Siendo el modelo de la izquierda la predicción de la red guiada por datos y el de la derecha el generado por elementos finitos.	50

A.5. Representación gráfica del error absoluto cuadrático para posiciones en el último paso de tiempo para la misma simulación vista en la Figura A.4. A la izquierda para la dimensión x , en el centro para la dimensión y y a la derecha para z . Los nodos oscuros están definidos como encastrados, por lo que no tienen error al ser una condición de contorno conocida.	50
A.6. Renderizado de la malla para comparar la solución numérica (izquierda) con la predicción de la red guiada por datos (derecha) en una situación de carga no entrenada. La escala de color representa el módulo de los desplazamientos en metros.	51
A.7. Representación gráfica de la inferencia en el “rollout” frente a los datos reales (gt) para las tres variables de estado presentes en el modelo simplificado.	51
B.1. Barrido de hiperparámetros para el modelo completo con las “MeshGraphNets”, pero únicamente prestando atención a la regulación de la constante que escala el ruido. Resultando en mejor convergencia para valores de ruido altos en velocidad, mínimos en tensión y nulos en posición. Las escalas (λ_c) se denotan de forma estandar para este barrido como: $noise_var_S$, $noise_var_v$, $noise_var_S$	54
B.2. Comparativa entre la inferencia de las “MeshGraphNets” con ruido escalado por una constante, donde $L2$ representa el error relativo (γ) para cada variable de estado.	54
B.3. Barrido de hiperparámetros para el modelo completo con las “MeshGraphNets”. Convergiendo para valores altos en tensión y velocidad constante, junto con valores notables en el ruido proporcional de posición.	56
B.4. Barrido de hiperparámetros para las “MeshGraphNets” con ruido mixto, pero únicamente prestando atención a la regulación de ruido para los mejores resultados. Resultando en mejor convergencia para valores de ruido altos en velocidad, mínimos en tensión y nulos en posición.	56
B.5. Comparativa para la inferencia de una simulación con condiciones de contorno no vistas. A la izquierda se muestra el resultado para el ultimo pseudo-tiempo generado por las redes guiadas por datos, mientras que a la derecha se evalúa el resultado del modelo de elementos finitos.	58

B.6.	Gráfico tridimensional que representa el error relativo de cada nodo para una geometría y situación de carga no utilizada durante el entreno para las redes guiadas por datos. A la izquierda se representa el error relativo para posiciones, en el centro para velocidad y a la derecha para tensiones.	59
B.7.	Renderizado de la malla para comparar la solución numérica (izquierda) con la predicción de la red guiada por datos (derecha) en una situación de carga no entrenada. La escala de color representa el módulo de los desplazamientos en metros.	59
C.1.	Barrido de los hiperparámetros que regulan el ruido para las TIGNNs, donde los mejores resultados se adquieren con ruidos altos en posición y velocidad, junto con ruido moderado en tensiones.	61
C.2.	Barrido de los hiperparámetros que regulan el ruido proporcional para las TIGNNs, donde los mejores resultados se adquieren con ruidos altos en posición y velocidad, junto con ruido moderado en tensiones.	62
C.3.	Barrido de los hiperparámetros que regulan el ruido mixto para las TIGNNs. Validando como diferentes cantidades de ruido mejoran la robustez del modelo para una selección de los mejores ensayos.	63
C.4.	Barrido de hiperparámetros para las TIGNNs con pesos no compartidos.	64
C.5.	Barrido específico para λ_d que refleja ese mejor rendimiento con valores de λ_d mayores que 20.	64
C.6.	Barrido de hiperparámetros para las TIGNNs con pesos compartidos. .	65
C.7.	Comparativa en inferencia de las tres variables de estado para los procesadores de pesos compartidos.	66
C.8.	Comparativa en inferencia de las tres variables de estado para los procesadores de pesos no compartidos.	66
C.9.	Comparativa para la inferencia de dos simulaciones con condiciones de contorno no vistas. A la izquierda se muestran dos resultados para el ultimo pseudo-tiempo generado por las redes híbridas, mientras que a la derecha se evalúa el resultado del modelo de elementos finitos.	68
C.10.	Error relativo nodal para la inferencia de las dos simulaciones expuestas en la Figura C.9, con condiciones de contorno no vistas. A la izquierda para la norma de las tres dimensiones de posición, al centro para las tres dimensiones de velocidad y a la derecha para las seis del tensor de tensiones	69

C.11. Renderizado de la malla para comparar la solución numérica (izquierda) con la predicción de la red híbrida (derecha) en una situación de carga no entrenada. La escala de color representa el módulo de los desplazamientos en metros.	69
C.12. Error relativo nodal para la inferencia de las dos simulaciones expuestas en la Figura C.9, con condiciones de contorno no vistas. Las dos primeras filas describen nodos alejados del desplazamiento impuesto, mientras que las dos siguientes son mucho mas próximos a la carga.	70
E.1. Esquema anatómico del hígado (fuente: SlideShare, recurso educativo).	75
E.2. Fijaciones anatómicas del hígado al cuerpo humano [28].	75
E.3. Representación geométrica de un lóbulo hepático [29].	76
E.4. Curvas de fuerza-tiempo adquiridas a partir de la prueba de tracción uniaxial para las 39 muestras de los tres hígados de cordero [31].	78
E.5. Curvas de ajuste de los modelos hiperelásticos a partir de datos experimentales de prueba de tensión-deformación uniaxial ex vivo [31].	78
E.6. Curva de ajuste de los datos de pruebas de tensión uniaxial para el modelado de la dependencia temporal de uno de los tres hígados testeados por Martinez Martinez [31].	79
E.7. Esquema simple de un perceptrón multicapa con varias neuronas de entrada y una única salida.	80
E.8. Diagrama de bloques de una MLP con conexión residual, donde la salida x'_i actualiza el estado de entrada mediante una suma, por lo que la dimensión de entrada y salida deberá ser la misma.	81
E.9. Esquema de Encoder-Decoder para la transformación de una entrada RGB hacia un espacio latente con dimensiones reducidas y mayor cantidad de canales. Esto permite una mayor expresividad de los distintos componentes principales. Este enfoque es similar al utilizado en el proyecto, pero emplea redes convolucionales en lugar de Perceptrones Multicapa (MLPs), dado que los datos del ejemplo son bidimensionales (imágenes) en lugar de unidimensionales (vectores).	82
E.10. Esquemmatización de las diferentes arquitecturas y sus sesgos geométricos.	83
E.11. Dominio de un sistema físico discretizado en una malla con variables de estado nodales z_i , distancias nodales relativas q_{ij} , interacciones externas f_i y propiedades globales u (izquierda). Representación del mismo sistema con atributos de nodo y arista: v_i y e_{ij} (derecha).	83

F.1. Esquematización de un sistema complejo creado como un grupo conectado de subsistemas Ω_i . La dinámica de cualquier subsistema se describe utilizando una formulación GENERIC que incluye términos conservativos y disipativos, teniendo también en cuenta las contribuciones externas en los términos de Γ . Los subsistemas se comunican entre sí a través de puertos mediante el intercambio de energía y entropía [38]	86
G.1. Histogramas de las normas de las tres variables de estado en el conjunto de entrenamiento. A la izquierda posición, al centro velocidad y a la derecha tensiones.	87
H.1. Gráfico de caja del error relativo medio de cada simulación testeada para el conjunto de geometrías no vistas (B, C y D) de las redes guiadas por datos. A la izquierda para posición, en el centro para velocidad y a la derecha para tensiones.	92
H.2. Gráfico tridimensional que representa el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para el conjunto de geometrías no vistas (B, C y D) de las redes guiadas por datos. A la izquierda para posición, en el centro para velocidad y a la derecha para tensiones.	93
H.3. Gráfico de caja del error relativo medio de cada simulación testeada para el conjunto de geometrías no vistas (B, C y D) de las redes híbridas. A la izquierda para posición, en el centro para velocidad y a la derecha para tensiones.	94
H.4. Gráfico tridimensional que representa el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para el conjunto de geometrías no vistas (B, C y D) de las redes híbridas. A la izquierda para posición, en el centro para velocidad y a la derecha para tensiones.	94
I.1. Diagrama de bloques de una TIGNNs con recurrencia codificada y agregada en el procesador de la red de grafos. En el caso de ser el primer paso pseudo-temporal el vector recurrente sera nulo.	96
I.2. Métrica representativa del error cuadrático medio durante el entrenamiento de la red recurrente, en el cual se aprecia el sobre-ajuste y el empeoramiento del error en validación durante el avance del entreno.	98

I.3. Métrica representativa del error relativo medio en tensiones para el “rollout” de validación durante el entrenamiento de la red recurrente, con una clara divergencia al empeorar también durante el avance del entreno.	98
K.1. Gráfico de caja que compara el error relativo en inferencia entre los conjuntos de entrenamiento y test. Concretamente con redes guiadas por datos y mallas de alta densidad.	101
K.2. Gráfico tridimensional que compara la inferencia de una geometría y carga no vista en un entreno (izquierda) con su solución numérica respectiva (derecha) para una malla de alta densidad en una arquitectura guiada por datos. $S11$ es la variable representativa en ABAQUS de σ_{11}	102
K.3. Gráfico tridimensional que representa el error relativo de cada nodo para una geometría y situación de carga no vista durante el entreno para mallas de alta densidad. A la izquierda se representa el error relativo para posiciones y a la derecha para tensiones.	102
L.1. Comparativa tridimensional que representa la diferencia entre el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para la geometría no vista (D) de las redes guiadas únicamente por datos entrenadas con diferentes bases de datos. A la izquierda para posición, en el derecha para velocidad y en la parte inferior para tensiones, siendo en verde los modelos con mejor inferencia para bases de datos con geometrías mixtas, mientras que en rojo tienen un mejor rendimiento los modelos con geometrías únicas.	105
L.2. Comparativa tridimensional que representa la diferencia entre el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para la geometría no vista (D) de las redes híbridas entrenadas con diferentes bases de datos. A la izquierda para posición, en el derecha para velocidad y en la parte inferior para tensiones, siendo en verde los modelos con mejor inferencia para bases de datos con geometrías mixtas, mientras que en rojo tienen un mejor rendimiento los modelos con geometrías únicas.	106

L.3. Comparativa tridimensional que representa la diferencia entre el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para la geometría no vista (D) de las redes híbridas y guiadas por datos entrenadas con bases de datos de geometrías mixtas. A la izquierda para posición, en el derecha para velocidad y en la parte inferior para tensiones, siendo en verde los modelos con mejor inferencia para bases de datos con geometrías mixtas, mientras que en rojo tienen un mejor rendimiento los modelos con geometrías únicas.	107
--	-----

Lista de Tablas

3.1.	Métricas e hiperparámetros de los dos mejores ensayos para el integrador de desplazamientos guiado por datos. Respecto a los hiperparámetros: <i>d.o</i> se refiere a la dimensión del espacio latente y de la capa oculta de las MLPs, <i>ruido</i> al valor que escala la distribución gaussiana de ruido, <i>pasos</i> a la cantidad de pasos que se iteran en el procesador de grafos.	20
3.2.	Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las redes guiadas por datos. Respectivamente: <i>d.o.</i> equivale a la dimensión del espacio latente y de la capa oculta de las MLPs, mientras que los parámetros con ε_c se relacionan con la constante de ruido y los de ε_p con un valor proporcional. Los dos primeros modelos tienen aproximadamente tres millones de parámetros entrenables, mientras que los dos últimos con once millones.	22
3.3.	Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las híbridas. Respectivamente: <i>d.o.</i> equivale a la dimensión del espacio latente y de la capa oculta de las MLPs, mientras que los parámetros con ε_c se relacionan con la constante de ruido y los de ε_p con un valor proporcional. Los tres primeros modelos tienen aproximadamente tres millones de parámetros entrenables, mientras que los dos últimos con once millones.	23
3.4.	Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las híbridas con mayores cantidades de parámetros entrenables.	23
3.5.	Las características de la base de datos están determinadas por dos factores principales: el número total de nodos y la variación de volumen con respecto a la geometría inicial.	24
B.1.	Resumen de los mejores modelos obtenidos para el estudio de ruido escalado por una constante en las redes guiadas por datos.	55
B.2.	Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las redes guiadas por datos.	57

B.3.	Resumen de los mejores modelos obtenidos para el estudio de ruido constante y estocástico en las redes guiadas por datos de mayor dimensión. ε_c se relacionan con la constante que escala el ruido.	57
B.4.	Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las redes guiadas por datos de mayor dimensión. Respectivamente <i>d.o.</i> equivale a la dimensión en la capa oculta, mientras que los parámetros con ε_c se relacionan con la constante de ruido y los de ε_p con un valor proporcional.	57
C.1.	Resumen de los mejores modelos obtenidos para el estudio de ruido constante y estocástico en las TIGNNs.	60
C.2.	Resumen de los mejores modelos obtenidos para ruido proporcional. . .	62
C.3.	Resultados con mejor convergencia en tensiones. Respectivamente <i>d.o.</i> equivale a la dimensión en la capa oculta, mientras que los parámetros con ε_c se relacionan con la constante que escala el ruido y los de ε_p con un valor proporcional.	63
C.4.	Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las informadas por física de mayor dimensión. Respectivamente <i>d.o.</i> equivale a la dimensión en la capa oculta, mientras que los parámetros con ε_c se relacionan con la constante que escala el ruido y los de ε_p con un valor proporcional.	67
G.1.	Estadísticas de una muestra de datos de entrenamiento sin normalizar y normalizados con la estandarización Z.	88
G.2.	Estadísticas de una muestra de datos de entrenamiento sin normalizar y normalizados con la normalización MinMax, para las tensiones y velocidades	89
H.1.	Resultado en inferencia para los diferentes modelos propuestos, pero únicamente entrenando con datos pertenecientes a la geometría original como indica la columna <i>Vista</i>	91
H.2.	Resultado en inferencia para los diferentes modelos propuestos, pero únicamente entrenando con datos pertenecientes a la geometría original como indica la columna <i>Vista</i>	92
H.3.	Resultado en inferencia para los diferentes modelos propuestos, pero únicamente entrenando con datos pertenecientes a la geometría original como indica la columna <i>Vista</i>	93

I.1.	Mettricas de las TIGNNs con recurrencia entrenadas para 2000 épocas, 150 neuronas en capas ocultas y 10 pasos. La tasa de aprendizaje y λ_d fijadas semejantemente a los modelos presentados en la memoria. El ruido es gaussiano y controlado por una constante que escala su distribución.	97
J.1.	Tiempos de cálculo en ABAQUS CAE para distintos mallados de una misma anatomía hepática. El procesador utilizado es un Inter Core i7-8700K	99
J.2.	Tiempos de respuesta en diferentes arquitecturas con una RTX 3090, para dos de las mallas utilizadas en el proyecto.	99
K.1.	Resultado en inferencia para los diferentes modelos propuestos con el mallado denso.	100
L.1.	Resultado en inferencia para las bases de datos mixtas sobre el conjunto de validación en geometrías vistas.	104
L.2.	Resultado en inferencia para los diferentes modelos propuestos en la geometría D, siendo para todos ellos no vista durante el entrenamiento.	104

Anexos

Anexos A

Modelo simplificado - Redes guiadas únicamente por datos

A.1. Barrido de hiperparámetros

Se consideran cuatro hiperparámetros principales para el barrido (Figura A.1): las dimensiones en la capa oculta y del espacio latente (dim_hidden), la tasa de aprendizaje (lr), la escala de ruido impuesta en el entrenamiento ($noise_var$) y los pasos de la red de grafos ($passes$). Siendo ($loss$) la variable representativa del MRE durante el rollout y $loss_z_val$ el MSE durante la validación.

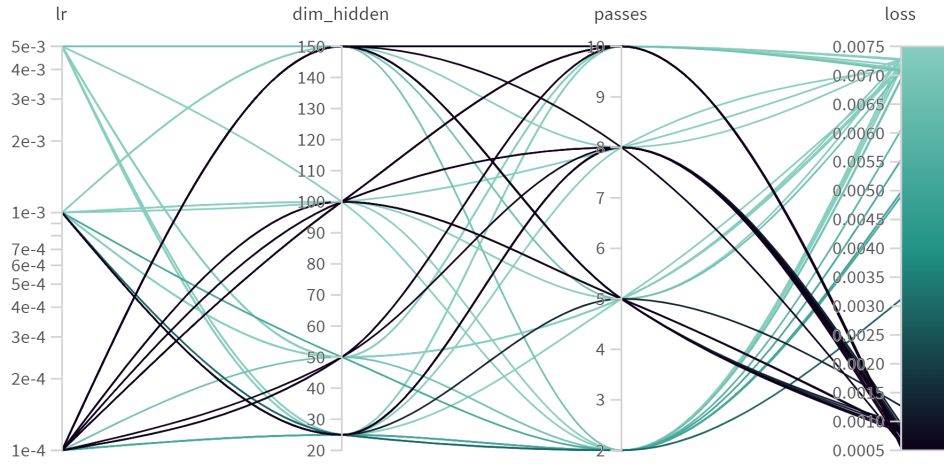


Figura A.1: Barrido de hiperparámetros para el modelo simplificado con las MeshGraphNets. Resultando en mejores valores de error para bajos valores de la tasa de aprendizaje, así como para modelos con mayor cantidad de pasos. Siendo deficientes los resultados para modelos con menos de cinco pasos.

La metodología de barrido se basa en una búsqueda aleatoria que explora todas las combinaciones posibles de hiperparámetros, siendo eficiente para establecer conclusiones sobre la sensibilidad del modelo. Otras metodologías como la optimización

bayesiana son descartadas debido a que pueden dejar sin explorar ciertas combinaciones de hiperparámetros al tender a un mínimo local rápidamente.

El primer análisis arroja mejores resultados para números bajos de tasa de aprendizaje, y para mayores pasos de transmisión de mensaje. Resultando en valores no óptimos para menos de dos pasos y tasas de aprendizaje mayores a $1e - 3$ (Figura A.1).

A continuación y en base a las primeras conclusiones, se realiza un filtrado de los mejores ensayos (Figura A.2), validando con mayor claridad que la convergencia es mas dependiente de la cantidad de pasos que del tamaño de la red.

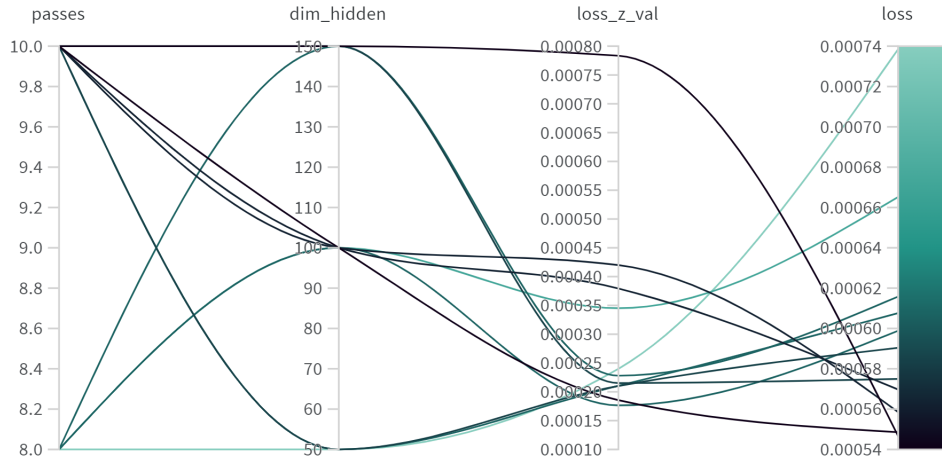


Figura A.2: Barrido de hiperparámetros para el modelo de integración de posición con las MeshGraphNets tras filtrar los mejores entrenamientos. Reafirmando la hipótesis de que los modelos con mayor espacio latente y mayor cantidad de pasos convergen mejor.

Esta segunda hipótesis relaciona el sobre-ajuste durante el entreno con la robustez en inferencias recurrentes. Planteando un escenario donde la red es realmente buena prediciendo una única integración temporal, sin embargo no es robusta ante el error acumulado durante el rollout.

Debido a este motivo se procede a estudiar si el ruido utilizado durante el entreno, como herramienta para evitar este sobre-ajuste, tiene la influencia esperada. Resultando tras otro filtrado en una validación de la hipótesis mostrada en la Figura A.3.

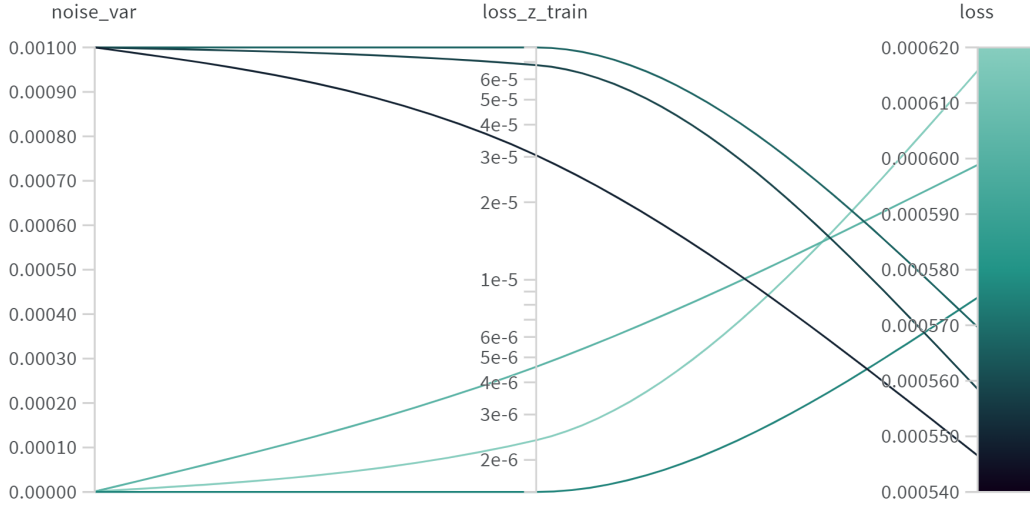


Figura A.3: Barrido de hiperparámetros para el modelo de integración de posición con las MeshGraphNets tras filtrar los mejores entrenamientos. Validando el ruido intra-entreno no solo evita el sobre-ajuste, sino que mejora la inferencia durante el rollout.

Verificando que el comportamiento estocástico del ruido evita el sobre-ajuste, obteniendo errores más altos en la validación, pero añadiendo robustez en la inferencia recurrente del rollout.

A.2. Comparativas y métricas adicionales

Para no sobrecargar la memoria se añaden en este anexo varios resultados gráficos relevantes, que pueden ayudar a entender el rendimiento obtenido.

Para la inferencia de una misma simulación se presentan tres representaciones gráficas adicionales: la Figura A.4 refleja una comparativa de la norma dos de los desplazamientos entre la predicción y el resultado real, mientras que la Figura A.5 muestra el error cuadrático en posiciones para cada nodo en el ultimo paso de tiempo. Por otra parte se añade un renderizado del resultado a nivel de malla, imitando las visualizaciones típicas de programas comerciales de elementos finitos (Figura A.6).

Asimismo, el “rollout” se describe como la evolución temporal de estas variables de estado, por lo que es necesario estimar de forma objetiva la calidad de la respuesta de la red respecto a los datos reales. La Figura A.7 muestra para un nodo no estático, próximo a la zona de carga, como el ajuste es indistinguible de la realidad.

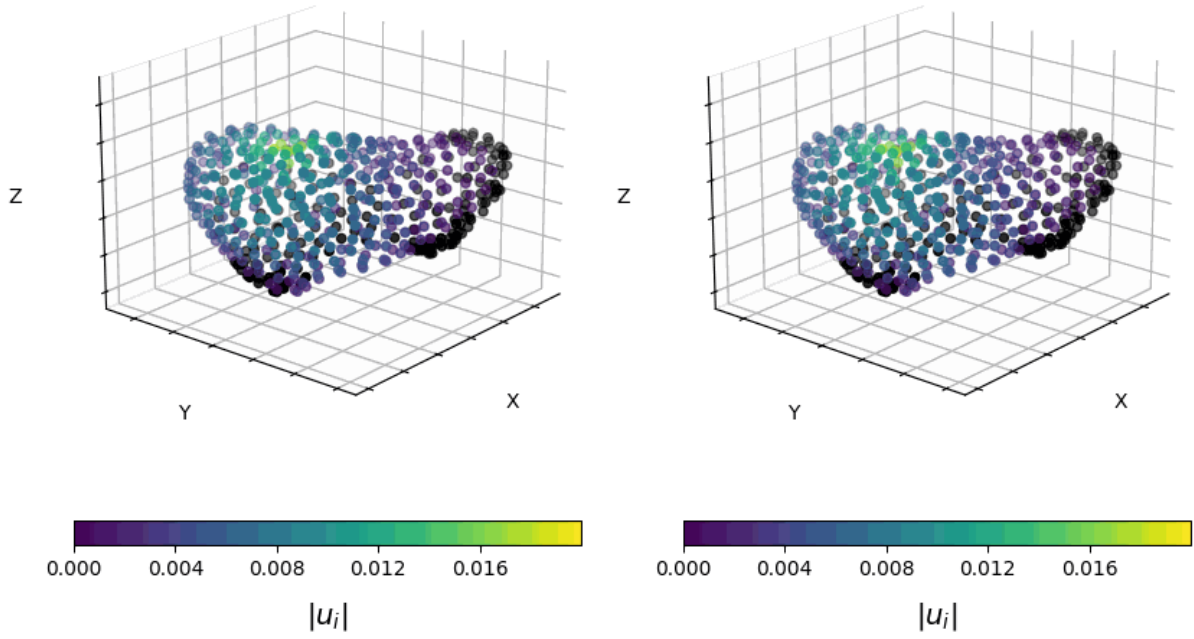


Figura A.4: Comparativa gráfica entre la solución numérica y la predicción de la red para una posición de carga no vista en el entrenamiento, con una representación en escala de color de la norma del desplazamiento en metros. Los nodos oscuros, visibles para ambos modelos, están definidos como apoyos. Siendo el modelo de la izquierda la predicción de la red guiada por datos y el de la derecha el generado por elementos finitos.

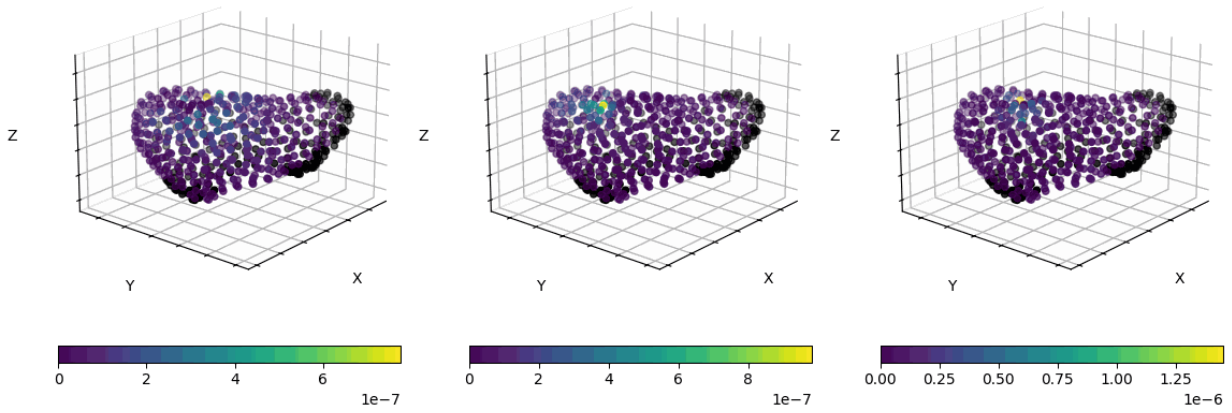


Figura A.5: Representación gráfica del error absoluto cuadrático para posiciones en el último paso de tiempo para la misma simulación vista en la Figura A.4. A la izquierda para la dimensión x , en el centro para la dimensión y y a la derecha para z . Los nodos oscuros están definidos como encastres, por lo que no tienen error al ser una condición de contorno conocida.

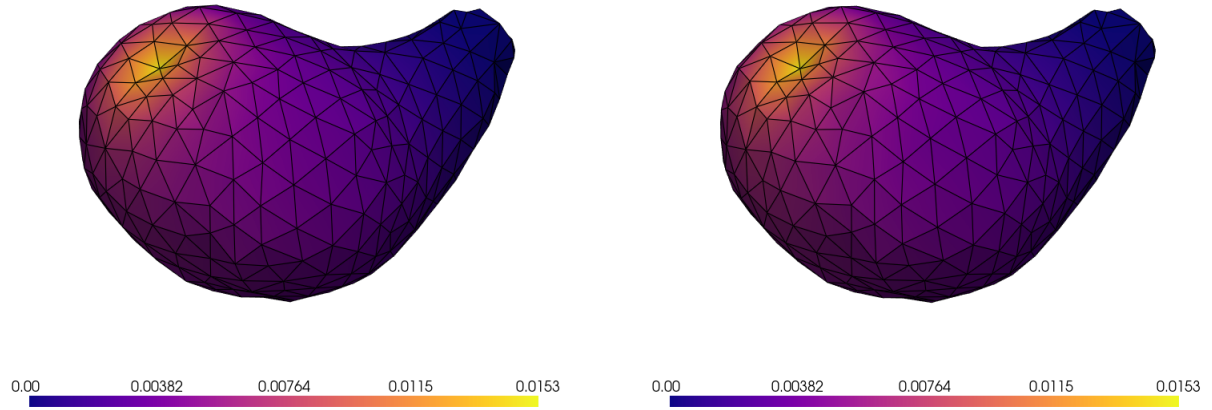


Figura A.6: Renderizado de la malla para comparar la solución numérica (izquierda) con la predicción de la red guiada por datos (derecha) en una situación de carga no entrenada. La escala de color representa el módulo de los desplazamientos en metros.

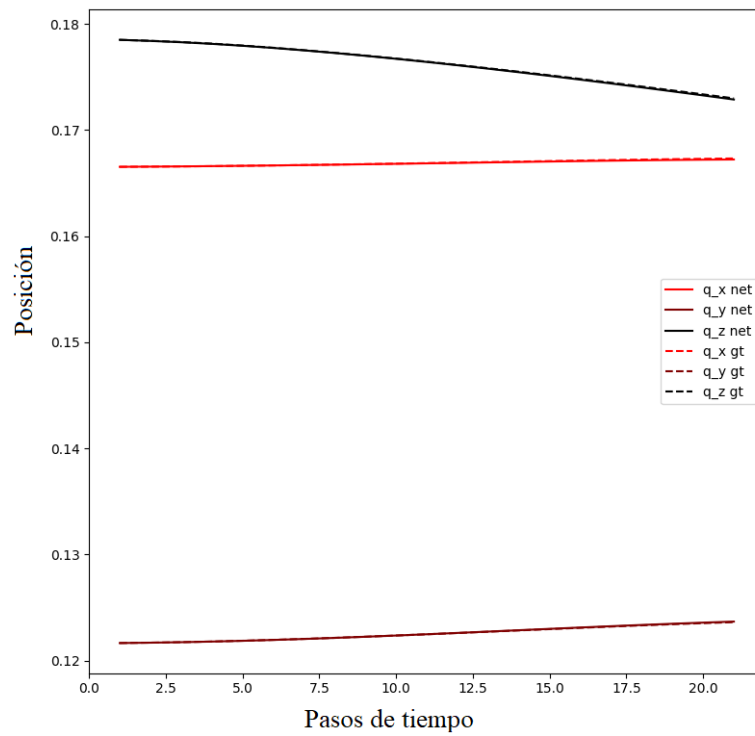


Figura A.7: Representación gráfica de la inferencia en el “rollout” frente a los datos reales (*gt*) para las tres variables de estado presentes en el modelo simplificado.

Anexos B

Modelo completo - Redes guiadas únicamente por datos

B.1. Barrido de Hiperparámetros

B.1.1. Robustez en inferencia por medio del ruido

El ruido se utiliza como herramienta para añadir robustez a una gran cantidad de redes, siendo un recurso utilizado en infinidad de problemas resueltos con aprendizaje automático.

El planteamiento más común es realizar una eliminación de ruido o “denoising” como término anglosajón. Donde a un conjunto de entrada perteneciente al subconjunto total de datos: $z \subseteq Z$, también denominado paquete o Batch, se le añade una constante de ruido estocástica y generalmente gaussiana, cuya media es regulada como un hiperparámetro: ε ,

$$\mathbf{z}_{\text{noise}}^t = \mathbf{z}^t + \varepsilon \quad (\text{B.1})$$

$$\mathbf{z}_{\text{noise}}^t \longrightarrow \text{Net} \longrightarrow \dot{\mathbf{z}}_{\text{net}} \quad (\text{B.2})$$

Tras evaluar el resultado en la red, Ec. (B.2), es preciso detallar que en un proceso de eliminación de ruido, no se tendría en cuenta dicha variación dentro de la función de pérdida del modelo:

$$\frac{\partial \mathbf{z}_{gt}}{\partial t} = \frac{\mathbf{z}^{t+1} - \mathbf{z}^t}{dt} \quad (\text{B.3})$$

$$\mathcal{L} = \left(\frac{\partial \mathbf{z}_{gt}}{\partial t} - \frac{\partial \mathbf{z}_{\text{net}}}{\partial t} \right)^2 \quad (\text{B.4})$$

No obstante, para la algorítmica presentada cumple otro papel fundamental: el aumento de datos. Donde el ruido tiene como objetivo evitar el sobre-ajuste de datos en la red,

mejorando su robustez en el “rollout”, con un planteamiento análogo donde la función de coste evalúa (al contrario de lo visto anteriormente) el valor de ruido añadido,

$$\frac{\partial \mathbf{z}_{gt}}{\partial t} = \frac{\mathbf{z}^{t+1} - \mathbf{z}_{noise}^t}{\partial t} \quad (\text{B.5})$$

Se proponen dos formas de añadir la componente estocástica y gaussiana. Como una constante que escala la distribución, equivalente la expresión B.1, o como una variable proporcional ε' al valor de z^t que de igual forma es una distribución gaussiana:

$$\mathbf{z}_{noise}^t = \mathbf{z}^t + \varepsilon(\mathbf{z}^t) \quad (\text{B.6})$$

De esta forma se definen tres metodologías: ruido con distribución constante, ruido con distribución proporcional y ruido mixto. Siendo el segundo solo implementado como testeo para la arquitectura híbrida del Anexo C.

B.1.2. Escalado constante del ruido gaussiano

Al igual que para el modelo simplificado, el ajuste de hiperparámetros no es trivial. Es necesario iterar diferentes combinaciones para ver como evoluciona la métrica a optimizar, siendo de nuevo γ ,

$$\varepsilon_c = \lambda_c \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (\text{B.7})$$

La estrategia de generación de ruido empleada sigue un enfoque análogo al modelo simplificado, generando tres parámetros (λ_n) que escalan las distribuciones gaussianas para cada conjunto de variables de estado (Ecuación B.7). Adicionalmente, se mantienen los hiperparámetros que controlan la dimensión oculta y del espacio latente, así como los pasos de procesamiento de la red de grafos para un total de parámetros entrenables que ronda los tres millones.

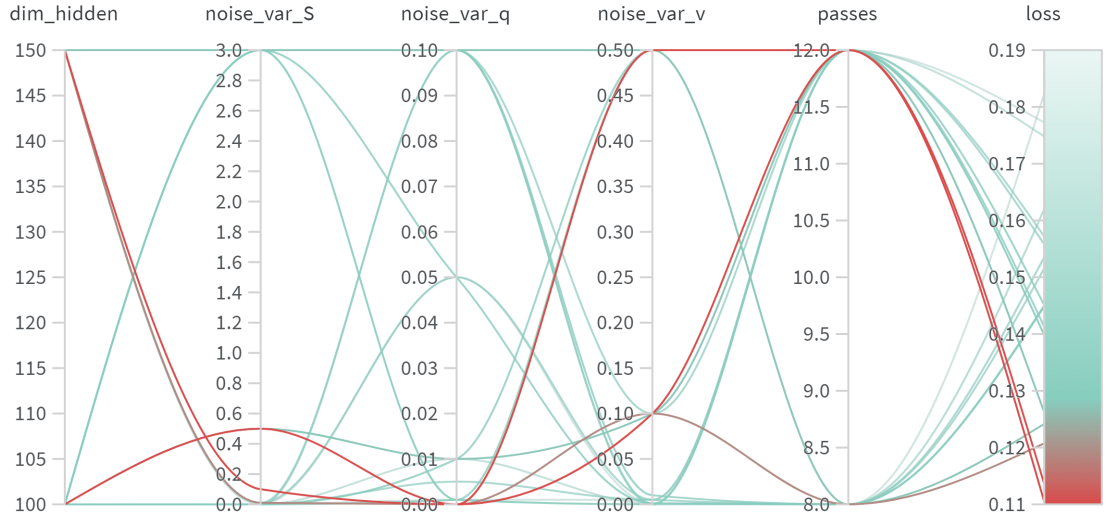


Figura B.1: Barrido de hiperparámetros para el modelo completo con las “MeshGraphNets”, pero únicamente prestando atención a la regulación de la constante que escala el ruido. Resultando en mejor convergencia para valores de ruido altos en velocidad, mínimos en tensión y nulos en posición. Las escalas (λ_c) se denotan de forma estandar para este barrido como: *noise_var_S*, *noise_var_v*, *noise_var_S*

La valoración inicial del barrido, visto en la Figura B.2, concluye que el rendimiento en inferencia se ve beneficiado de mayores cantidades de ruido en tensión y velocidad, además de una dimensión mayor del espacio latente, lo que tiene correlación con el aumento de sus parámetros entrenables. Por la parte de λ_d no se establecen conclusiones a priori fuera de su capacidad de regular en inicio del entrenamiento y su bajada de coste inicial.

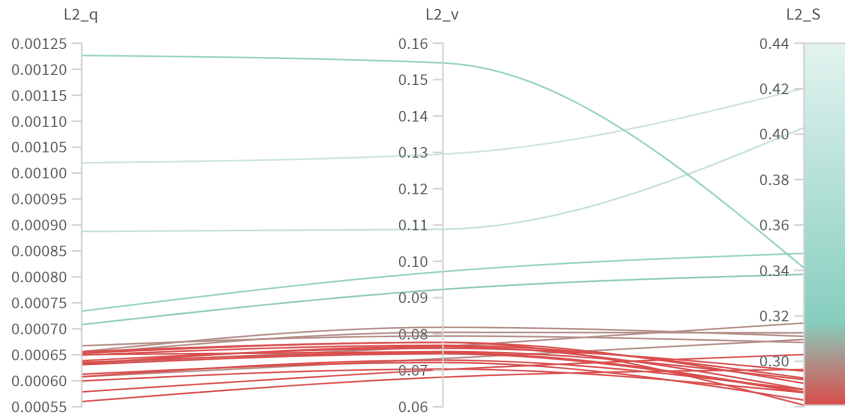


Figura B.2: Comparativa entre la inferencia de las “MeshGraphNets” con ruido escalado por una constante, donde $L2$ representa el error relativo (γ) para cada variable de estado.

En relación con la estabilidad del error relativo medio para cada grupo de variables de estado, es importante destacar que, como se ilustra en la Figura B.2, se observa un patrón significativo. Cuando un error converge hacia un mínimo en particular, se evidencia una correlación entre este fenómeno y la disminución de otros errores relacionados. Este hallazgo sugiere la posibilidad de que la red neuronal alcance un mínimo global que incida en todas las variables de estado simultáneamente. Este fenómeno es fundamental, ya que implica que la optimización de la red no solo afecta a un conjunto específico de variables, sino que tiene un impacto más amplio y sistemático en la totalidad de las variables de estado consideradas.

Como resumen, se agrupan los mejores ensayos resultantes para este barrido en la siguiente tabla:

parámetros	dimensión oculta	$\varepsilon_c \sigma$	$\varepsilon_c q$	$\varepsilon_c \dot{q}$	pasos	γ_σ	γ_q	$\gamma_{\dot{q}}$
3mill	70	0.5	0	0.5	12	0.2862	0.0006	0.0704
3mill	70	0.1	0	0.5	12	0.2964	0.0006	0.0681

Tabla B.1: Resumen de los mejores modelos obtenidos para el estudio de ruido escalado por una constante en las redes guiadas por datos.

B.1.3. Ruido mixto

En el penúltimo análisis y debido a la mayor cantidad de hiperparámetros, se emplea la optimización bayesiana para ajustar seis los valores que conforman una función de ruido mixta. La nueva función combina un término proporcional, ε_p (Ecuación B.7), y uno constante, ε_c , con el objetivo de aprovechar las ventajas ofrecidas por ambas estrategias,

$$z_{\text{noise}}^t = z^t + \varepsilon_p(z^t) + \varepsilon_c \quad (\text{B.8})$$

$$\varepsilon_p = z(t) \lambda_p \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (\text{B.9})$$

Las componentes constantes (λ_c) se denotan con una *c*: *noise_var_S_c*, *noise_var_q_c*, *noise_var_v_c*, mientras que las proporcionales (λ_p) se mantienen con la notación: *noise_var_S*, *noise_var_q*, *noise_var_v*.

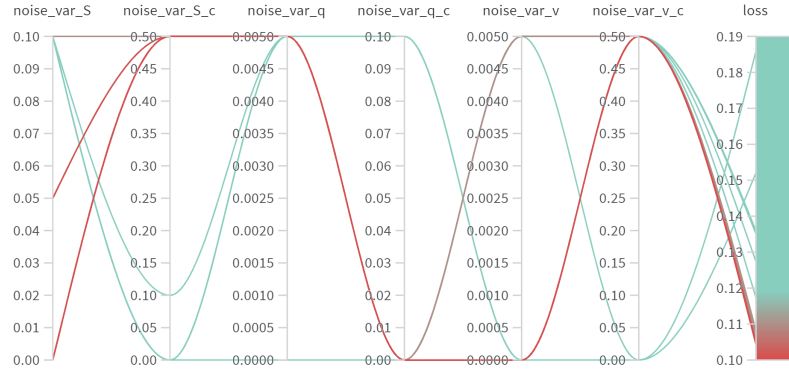


Figura B.3: Barrido de hiperparámetros para el modelo completo con las “MeshGraphNets”. Convergiendo para valores altos en tensión y velocidad constante, junto con valores notables en el ruido proporcional de posición.

Aunque la dificultad del ajuste escala exponencialmente al introducir más términos, se puede validar qué valores altos en tensión y velocidad constante, junto con valores notables en el ruido proporcional de posición, combinan bien para lograr buenas métricas de error en inferencia (Figura B.3). Asimismo, la convergencia del error relativo se mantiene estable, obteniendo además, mejores métricas para las tensiones (Figura B.4).

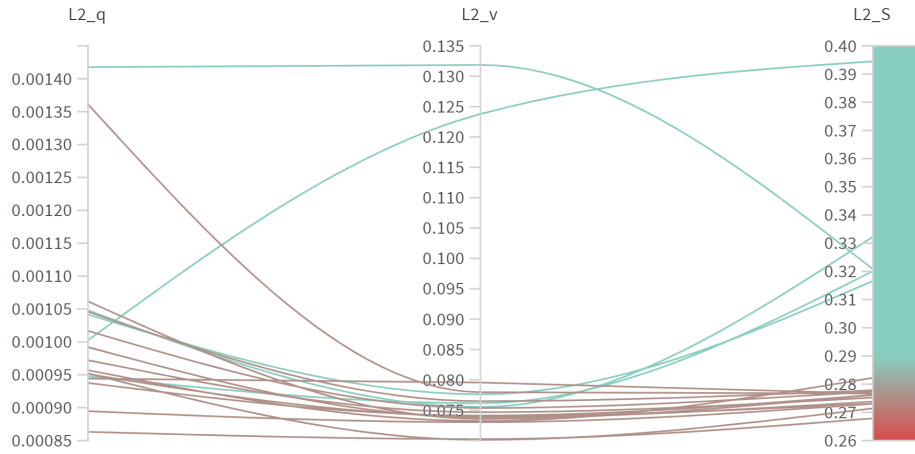


Figura B.4: Barrido de hiperparámetros para las “MeshGraphNets” con ruido mixto, pero únicamente prestando atención a la regulación de ruido para los mejores resultados. Resultando en mejor convergencia para valores de ruido altos en velocidad, mínimos en tensión y nulos en posición.

Los mejores resultados para ruido mixto y un tamaño aproximado de tres millones de parámetros entrenables se agrupan en la siguiente tabla:

Tabla B.2: Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las redes guiadas por datos.

d.o.	$\varepsilon_p\sigma$	$\varepsilon_c\sigma$	$\varepsilon_p q$	$\varepsilon_c q$	$\varepsilon_p \dot{q}$	$\varepsilon_c \dot{q}$	pasos	γ_σ	γ_q	$\gamma_{\dot{q}}$
70	0	0.5	0.005	0	0	0.5	12	0.2679	0.0008	0.0702
70	0	0.5	0.005	0	0.005	0.5	12	0.2762	0.0009	0.0741

B.2. Redes con mayor cantidad de parámetros entrénales

Por último, se analiza el impacto de espacios latentes de mayor tamaño, lo que conlleva un incremento en la cantidad de parámetros entrenables, tanto para modelos con ruido escalado por una constante (Tabla B.3) como para modelos con ruido mixto (Tabla B.4).

Tabla B.3: Resumen de los mejores modelos obtenidos para el estudio de ruido constante y estocástico en las redes guiadas por datos de mayor dimensión. ε_c se relacionan con la constante que escala el ruido.

parámetros	dimensión oculta	$\varepsilon_c\sigma$	$\varepsilon_c q$	$\varepsilon_c \dot{q}$	pasos	γ_σ	γ_q	$\gamma_{\dot{q}}$
11mill	150	0.1	0	0.5	12	0.2814	0.0006	0.0717
5mill	100	0	0.001	0.5	12	0.3459	0.0006	0.0734
8mill	150	0.01	0	0.1	8	0.3115	0.0006	0.0805

Tabla B.4: Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las redes guiadas por datos de mayor dimensión. Respectivamente $d.o.$ equivale a la dimensión en la capa oculta, mientras que los parámetros con ε_c se relacionan con la constante de ruido y los de ε_p con un valor proporcional.

d.o.	$\varepsilon_p\sigma$	$\varepsilon_c\sigma$	$\varepsilon_p q$	$\varepsilon_c q$	$\varepsilon_p \dot{q}$	$\varepsilon_c \dot{q}$	pasos	γ_σ	γ_q	$\gamma_{\dot{q}}$
150	0.1	0.5	0	0	0.005	0.5	12	0.2539	0.0006	0.072
150	0.1	0.5	0	0	0.005	0.5	12	0.2584	0.0006	0.0733
150	0.1	0.5	0	0	0.005	0.1	12	0.2586	0.0007	0.0918

Se deduce de este análisis que para modelos menos masivos, una aproximación simplificada al ruido se muestra como la elección preferida, mientras que para redes más complejas, una metodología mixta de adición de ruido tiende a potenciar la inferencia en tensiones.

B.3. Comparativas y medidas adicionales

A continuación se exponen algunas representaciones tridimensionales para los mejores resultados de las arquitecturas guiadas por datos: entre ellas una comparativa

tridimensional de la predicción frente a la solución numérica (Figura B.5), junto a su error relativo nodal para la misma simulación, Figura B.6,

$$\varepsilon_i = \frac{\|z_i^{GT} - z_i^{Net}\|_2}{\|z_i^{GT}\|_2} \quad (\text{B.10})$$

En la Figura B.7 se expone también un renderizado de malla para una de las situaciones de carga no entrenadas (Figura B.7). Imitando el formato de representación típico de los elementos finitos y validando la baja diferencia entre la predicción y la solución numérica para el módulo de desplazamientos en metros.

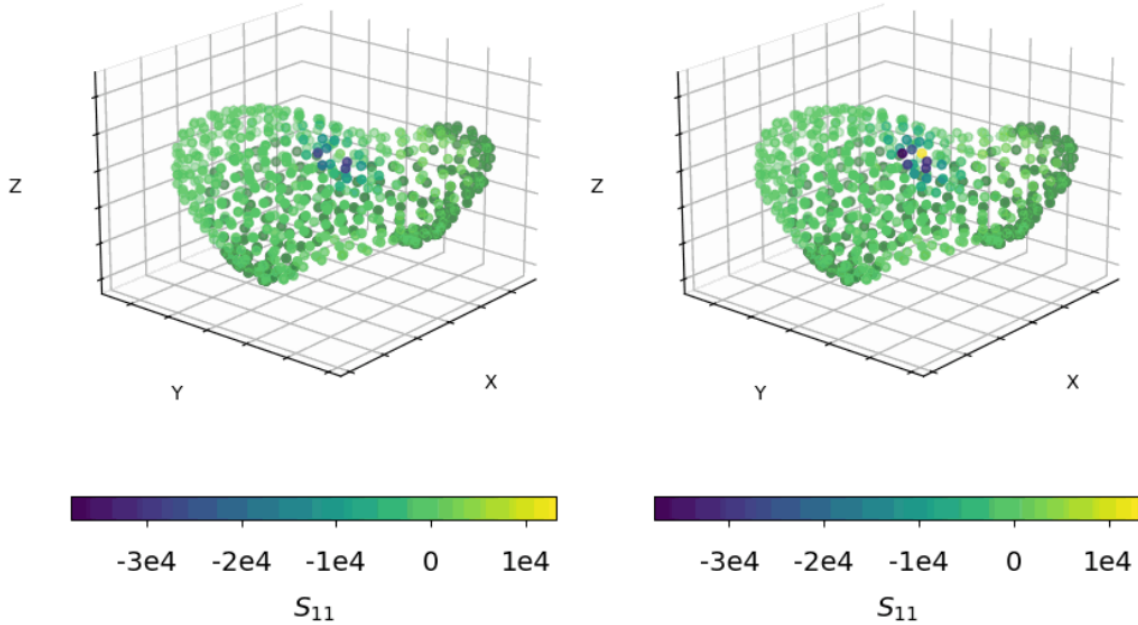


Figura B.5: Comparativa para la inferencia de una simulación con condiciones de contorno no vistas. A la izquierda se muestra el resultado para el ultimo pseudo-tiempo generado por las redes guiadas por datos, mientras que a la derecha se evalúa el resultado del modelo de elementos finitos.

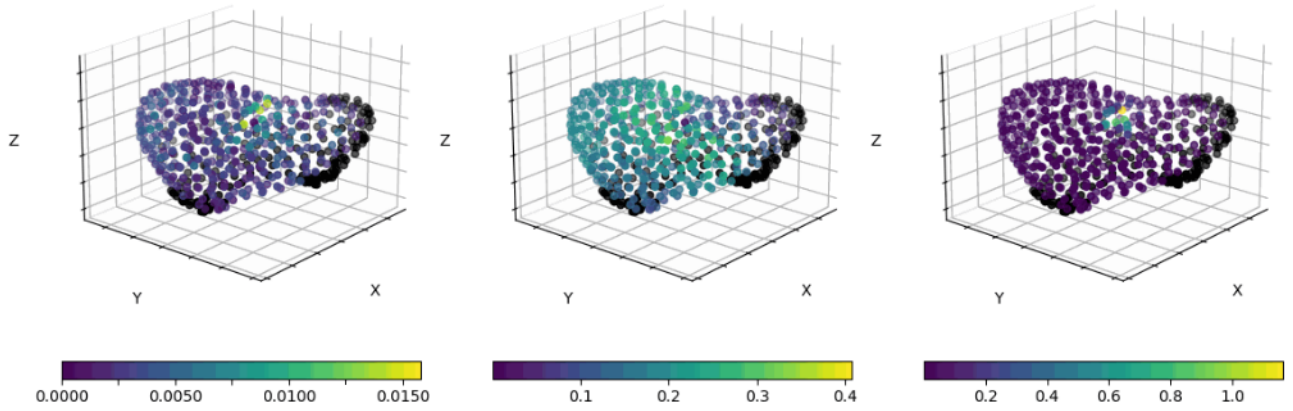


Figura B.6: Gráfico tridimensional que representa el error relativo de cada nodo para una geometría y situación de carga no utilizada durante el entrenamiento para las redes guiadas por datos. A la izquierda se representa el error relativo para posiciones, en el centro para velocidad y a la derecha para tensiones.

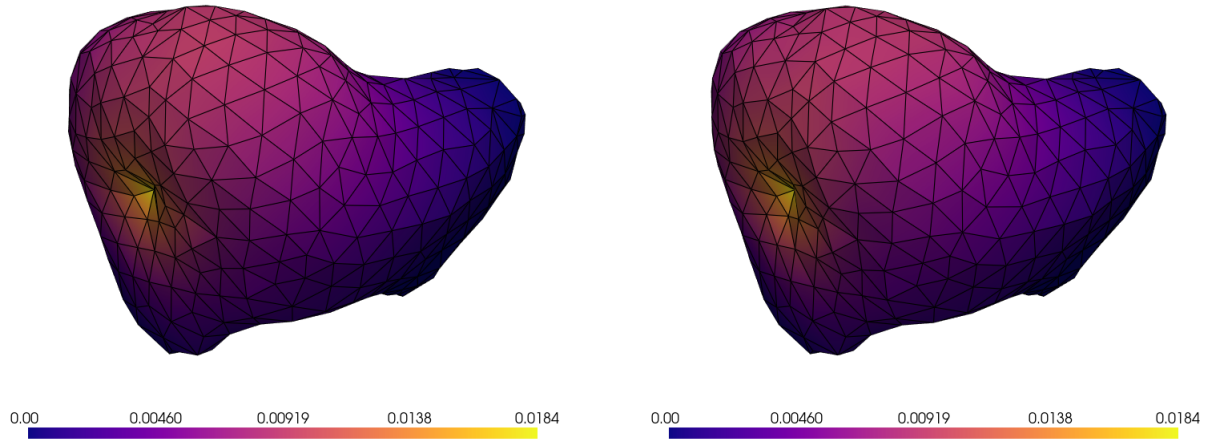


Figura B.7: Renderizado de la malla para comparar la solución numérica (izquierda) con la predicción de la red guiada por datos (derecha) en una situación de carga no entrenada. La escala de color representa el módulo de los desplazamientos en metros.

Anexos C

Modelo completo - Redes híbridas

C.1. Barrido de hiperparámetros

C.1.1. Escalado constante del ruido gaussiano

La variabilidad de la constante que escala el ruido se establece como tres valores independientes que afectan a cada subconjunto de variables de estado tal y como se expone en la ecuación B.1. Siendo respectivamente *noise_var_S* el asociado a las seis tensiones, *noise_var_q* el asociado a las tres dimensiones de la posición y *noise_var_v* a las tres dimensiones de la velocidad.

El barrido mostrado en la Figura C.1 resulta de un filtrado aplicado a un barrido mas general de hiperparámetros, seleccionando los modelos semejantes con mejores resultados respecto a la métrica impuesta (Tabla C.1).

dimensión oculta	λ_d	$\varepsilon_c \sigma$	$\varepsilon_c q$	$\varepsilon_c \dot{q}$	pasos	γ_σ	γ_q	$\gamma_{\dot{q}}$
150	100	0.5	0.05	0.5	12	0.2391	0.0021	0.0835
100	20	0.5	0.01	0.005	12	0.2599	0.0017	0.1111
150	100	0.001	0.05	0.1	12	0.2910	0.0007	0.0744

Tabla C.1: Resumen de los mejores modelos obtenidos para el estudio de ruido constante y estocástico en las TIGNNs.

La optimización concluye que bajas cantidades de ruido benefician a la inferencia, no obstante en velocidades, mayores valores repercuten positivamente sobre el rendimiento de la red. De la misma forma, altas cantidades de ruido en posición y tensiones causan divergencias durante el entreno, empeorando la capacidad predictiva de la red.

Es interesante analizar como en los mejores modelos, a veces se sacrifica la precisión

en una variable para mejorar los resultados en otra. Este fenómeno evidencia la complejidad inherente en la búsqueda de soluciones óptimas, donde mantener errores mínimos en una dimensión puede generar perdidas de precisión para otras variables de estado.

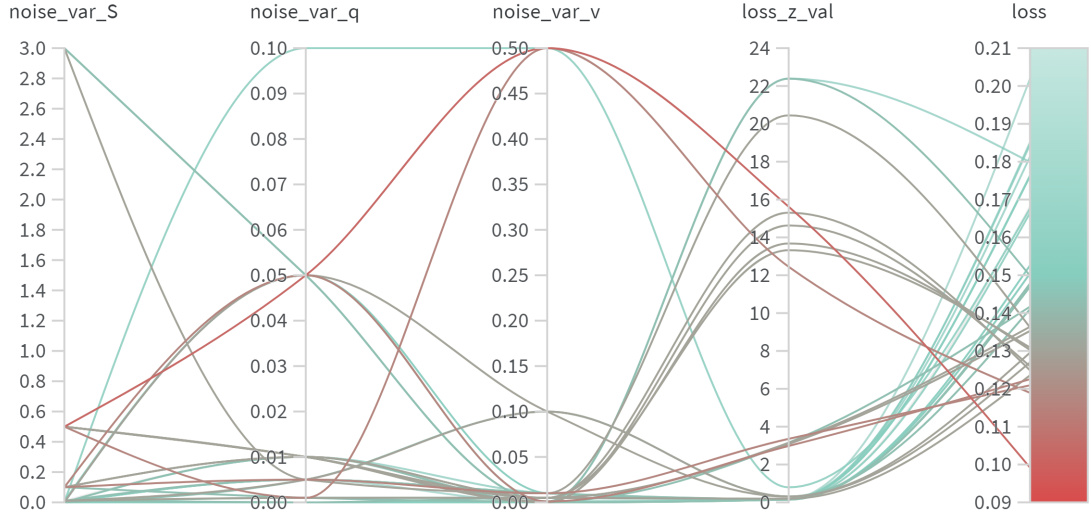


Figura C.1: Barrido de los hiperparámetros que regulan el ruido para las TIGNNs, donde los mejores resultados se adquieren con ruidos altos en posición y velocidad, junto con ruido moderado en tensiones.

C.1.2. Ruido gaussiano proporcional

La variable proporcional de ruido se establece como tres valores independientes que afectan a cada subconjunto de variables de estado tal y como se expone en la ecuación B.6. Siendo respectivamente *noise_var_S* el asociado a las seis tensiones, *noise_var_q* el asociado a las tres dimensiones de la posición y *noise_var_v* a las tres dimensiones de la velocidad.

Análogamente al apartado anterior, el barrido mostrado en la Figura C.2 resulta de un filtrado aplicado a un barrido mas general de hiperparámetros, seleccionando los modelos semejantes con mejores resultados respecto a la métrica impuesta.

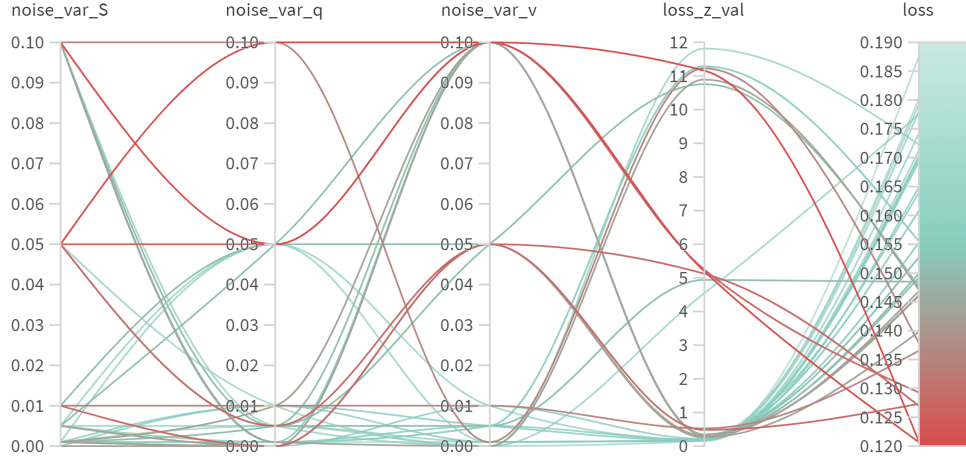


Figura C.2: Barrido de los hiperparámetros que regulan el ruido proporcional para las TIGNNs, donde los mejores resultados se adquieren con ruidos altos en posición y velocidad, junto con ruido moderado en tensiones.

Realizando un análisis similar a lo anterior, se obtienen unas conclusiones relativamente similares a las del barrido con el ruido estocástico, aunque en este caso la sensibilidad del modelo ante valores mas altos de ruido es realmente superior. En conjunto con lo expuesto en la Tabla C.2 se aprecia que las redes son menos sensibles a la incorporación de ruido proporcional, obteniendo buenas convergencias para los tres conjuntos de variables de estado.

dimensión oculta	λ_d	$\varepsilon_c \sigma$	$\varepsilon_c q$	$\varepsilon_c \dot{q}$	pasos	γ_σ	γ_q	$\gamma_{\dot{q}}$
150	5	0.05	0.1	0.1	12	0.2953	0.0010	0.1045
150	5	0.1	0.05	0.1	12	0.3285	0.0013	0.0997
150	5	0.05	0.005	0.05	8	0.3331	0.0014	0.0968
150	5	0.01	0	0.05	12	0.3153	0.0011	0.0919
150	20	0.05	0.05	0.1	8	0.3131	0.0011	0.0876

Tabla C.2: Resumen de los mejores modelos obtenidos para ruido proporcional.

Sin embargo, los resultados, aun siendo mas homogéneos entre si, son obtenidos de modelos con ligeramente peores métricas que con el ruido estocástico.

C.1.3. Ruido mixto

En el siguiente análisis, se emplea la optimización bayesiana para ajustar seis nuevos hiperparámetros que conforman una función de ruido mixta. Se introduce una nueva función que combina un término proporcional, ε_p , y uno constante, ε_c , con el objetivo de aprovechar las ventajas ofrecidas por ambas estrategias, Ecuación B.8.

Las componentes constantes se denotan con una c : $noise_var_S_c$, $noise_var_q_c$, $noise_var_v_c$, mientras que las proporcionales se mantienen con la notación estándar: $noise_var_S$, $noise_var_q$, $noise_var_v$.

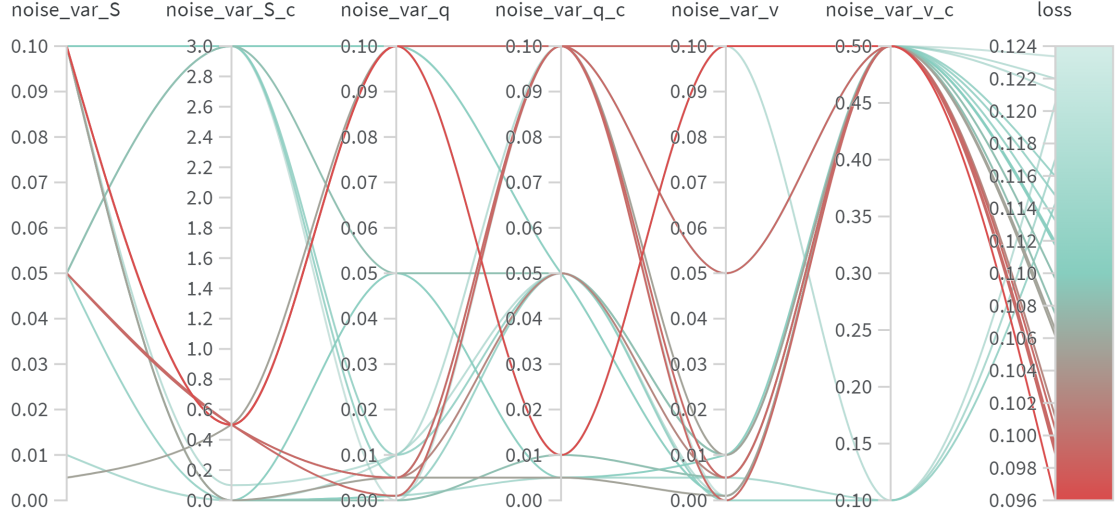


Figura C.3: Barrido de los hiperparámetros que regulan el ruido mixto para las TIGNNs. Validando como diferentes cantidades de ruido mejoran la robustez del modelo para una selección de los mejores ensayos.

El barrido presente en la Figura C.3 agrupa los resultados de la optimización bayesiana para una selección de los mejores errores relativos obtenidos. Estos resultados confirman que diversas combinaciones de ruido no nulo, tanto constante como proporcional, logran los mejores resultados entre los tres estudios (Tabla C.3).

Tabla C.3: Resultados con mejor convergencia en tensiones. Respectivamente $d.o.$ equivale a la dimensión en la capa oculta, mientras que los parámetros con ε_c se relacionan con la constante que escala el ruido y los de ε_p con un valor proporcional.

d.o.	λ_d	$\varepsilon_p\sigma$	$\varepsilon_c\sigma$	ε_pq	ε_cq	$\varepsilon_p\dot{q}$	$\varepsilon_c\dot{q}$	pasos	γ_σ	γ_q	$\gamma_{\dot{q}}$
125	50	0.05	0.5	0.005	0.1	0.005	0.5	12	0.2279	0.0023	0.0797
150	100	0.1	0.5	0.1	0.01	0.1	0.5	12	0.2293	0.0025	0.0760
125	50	0.05	0.5	0.001	0.1	0	0.5	12	0.2311	0.0029	0.0802

El resto de hiperparámetros no son despreciados durante el barrido, su monitorización es continua pero no se presentan cambios en las hipótesis establecidas desde los modelos guiados por datos (Figura C.4).

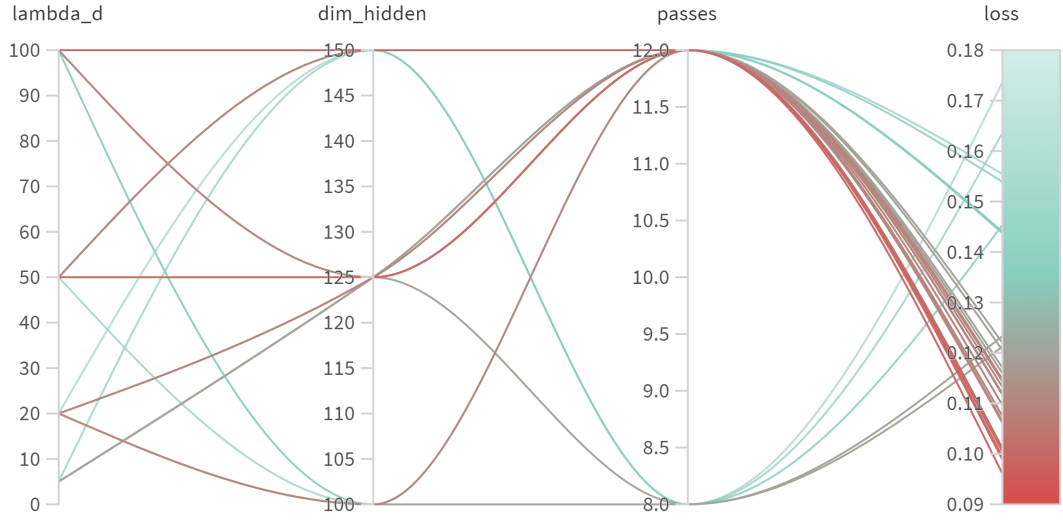


Figura C.4: Barrido de hiperparámetros para las TIGNNs con pesos no compartidos.

Si que es destacable como adicionalmente se introduce el término de regularización del coste relacionado con las condiciones de degeneración (λ_d). Este término, inicialmente, no muestra correlaciones evidentes. Sin embargo, al analizar un subgrupo de ensayos con métricas de error mas bajas, se observa que cuando la componente guiada por grupos es dominante, es decir, cuando el valor de lambda es mayor, el rendimiento de la red mejor (Figura C.5).

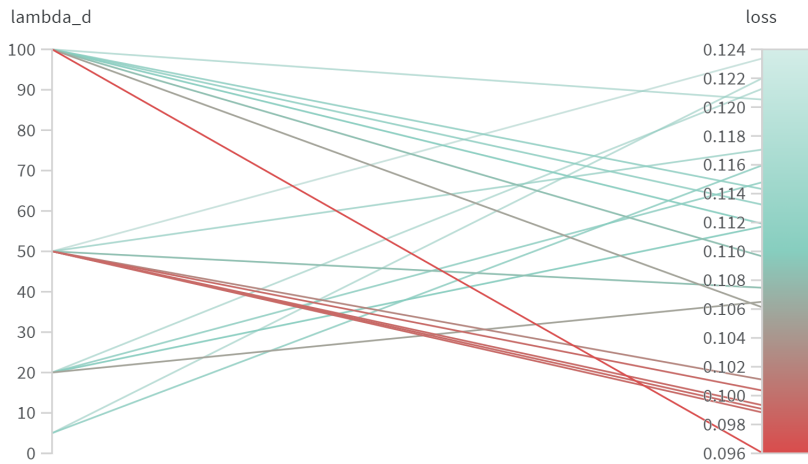


Figura C.5: Barrido específico para λ_d que refleja ese mejor rendimiento con valores de λ_d mayores que 20.

C.2. Influencia del procesador de la red de grafos

Otro hiperparámetro relevante es la opción de compartir o no los pesos entre los perceptrones multicapa en el procesador de la red de grafos. En un modelo sin pesos compartidos, cada paso implica una MLP independiente, mientras que en el caso de pesos compartidos, se utiliza una única MLP y los datos son procesados iterativamente en cada paso a través de ella.

En consecuencia, los modelos sin pesos compartidos tienen una mayor cantidad de parámetros que los modelos con pesos compartidos, por lo que es indispensable estudiar si realmente esta mayor cantidad de pesos y sesgos mejoran las capacidades de la red.

C.2.1. Barrido de hiperparámetros para redes con pesos compartidos

Al igual que para los casos anteriores donde los pesos no eran compartidos, se implementa una metodología de ruido mixto (tras ver que es el mas flexible y efectivo) que a primeras conclusiones, como se ve en la Figura C.6, resulta en una optimización muy similar a la presentada en el Apartado C.1.3.

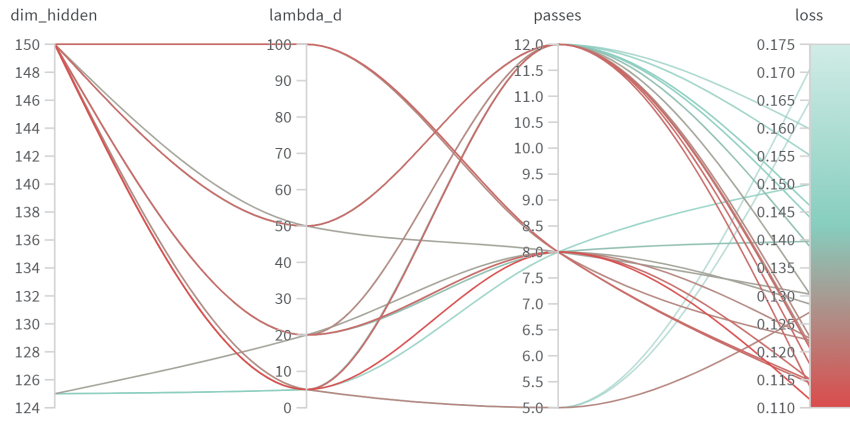


Figura C.6: Barrido de hiperparámetros para las TIGNNs con pesos compartidos.

Sin embargo, al comparar las métricas de error, se observa que ambas metodologías no muestran el mismo rendimiento en inferencia. Al analizar el error relativo medio para las tres variables de estado en cada modelo, deberíamos notar una correlación relativamente uniforme entre las tres métricas (ver Figuras C.7 y C.8). Asumiendo como hipótesis que una buena inferencia en tensiones, por ejemplo, no debería implicar un peor resultado en posición.

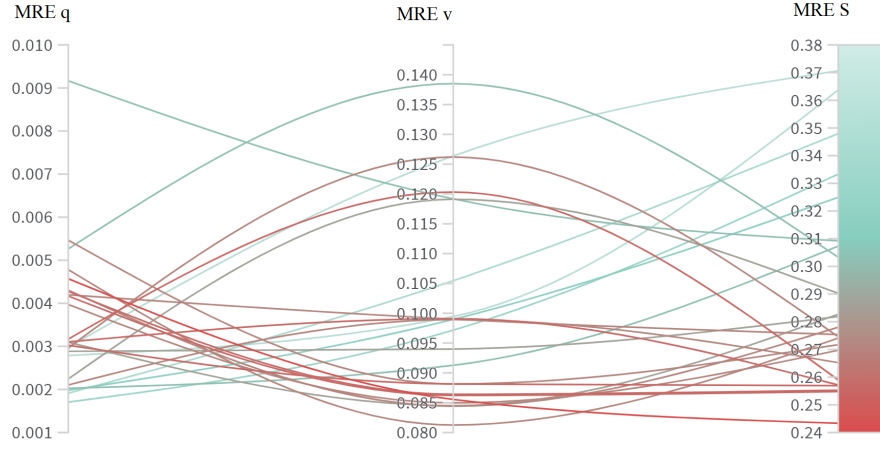


Figura C.7: Comparativa en inferencia de las tres variables de estado para los procesadores de pesos compartidos.

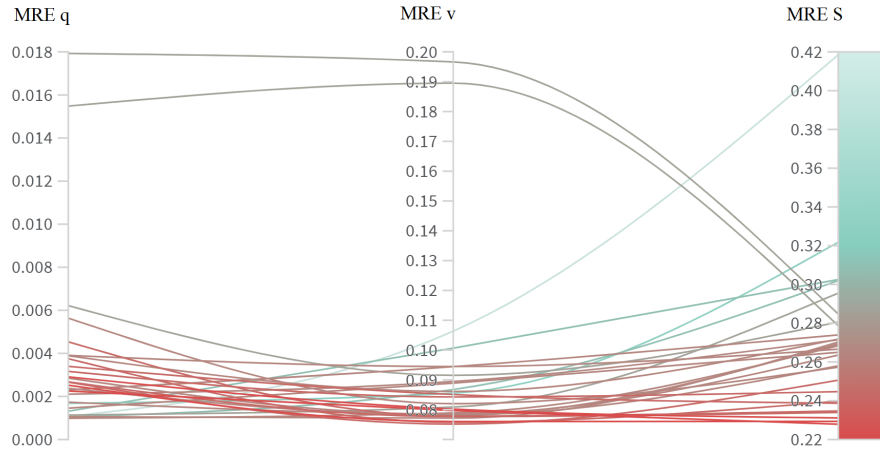


Figura C.8: Comparativa en inferencia de las tres variables de estado para los procesadores de pesos no compartidos.

Queda claro como los modelos de pesos compartidos son mucho más sensibles, siendo realmente difícil conseguir la convergencia para esta variación de la metodología original.

C.3. Redes con mayor cantidad de parámetros entrenables

Equivalentemente a las redes guiadas por datos, se valora la influencia de aumentar el numero de parámetros entrenables por medio de un espacio latente mayor. Esto

implica otra mejora en inferencia para los modelos mucho mas grandes.

Tabla C.4: Resumen de los mejores modelos obtenidos para el estudio de ruido mixto en las informadas por física de mayor dimensión. Respectivamente $d.o.$ equivale a la dimensión en la capa oculta, mientras que los parámetros con ε_c se relacionan con la constante que escala el ruido y los de ε_p con un valor proporcional.

d.o.	λ_d	$\varepsilon_p\sigma$	$\varepsilon_c\sigma$	$\varepsilon_p q$	$\varepsilon_c q$	$\varepsilon_p \dot{q}$	$\varepsilon_c \dot{q}$	pasos	γ_σ	γ_q	$\gamma_{\dot{q}}$
500	100	0.05	0.1	0	0.1	0.005	0.1	16	0.2401	0.0007	0.0668
350	50	0.1	0.1	0.005	0.1	0.005	0.1	25	0.2433	0.0008	0.0671
250	50	0.1	0.5	0.005	0.1	0.005	0.5	25	0.2094	0.0017	0.0737

No obstante el aumento del tamaño de la red también tiene correlación con una mejora en rendimiento en las redes guiadas por datos, por lo que lo mas objetivo sería realizar entrenamientos con la misma cantidad de parámetros entrenables para ambos modelos. Tarea que se dificulta enormemente debido a que la arquitectura guiada por datos es mucho mas lenta, por lo que se desestima al no ser un objetivo del presente trabajo.

C.4. Comparativas y métricas adicionales

Al igual que con las demás arquitecturas, otras métricas y representaciones tridimensionales de los resultados de la red están expuestas en esta sección del Anexo.

En la Figura C.9 se representan los modelos tridimensionales para poder comparar los resultado entre la simulación numérica y la inferencia de la red, siendo ambos modelos cargas no vistas durante el entrenamiento. Por otra parte, la Figura C.10 muestra el error relativo por nodo Ecuación B.10 para los dos mismos modelos, en dicha representación se muestran errores mas elevados en la zona proximal al desplazamiento puntual. Mientras que la Figura C.11 describe una comparativa renderizada como malla para una mejor comparativa de la precisión del modelo sobre la carga no vista.

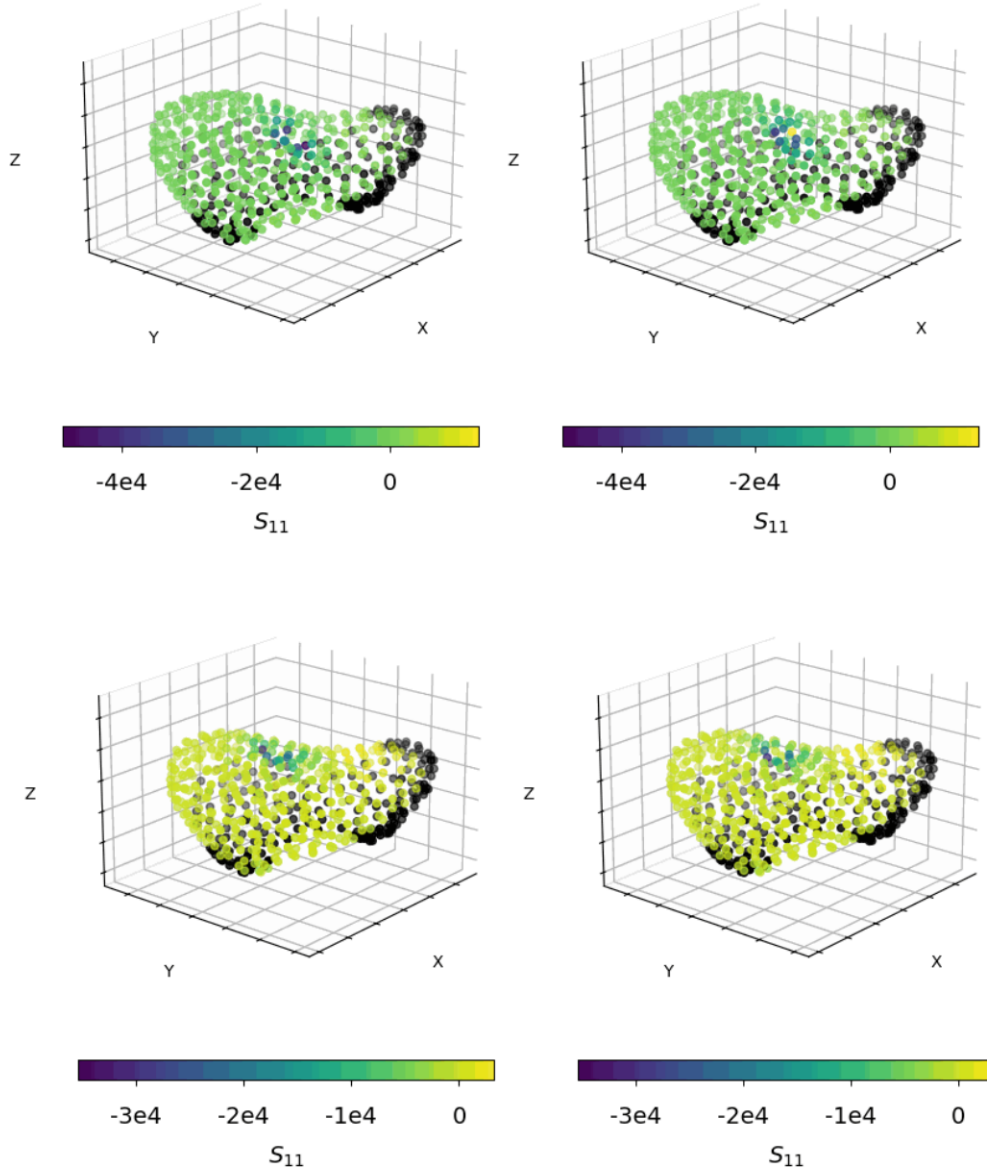


Figura C.9: Comparativa para la inferencia de dos simulaciones con condiciones de contorno no vistas. A la izquierda se muestran dos resultados para el ultimo pseudo-tiempo generado por las redes híbridas, mientras que a la derecha se evalúa el resultado del modelo de elementos finitos.

Por último, con el propósito de validar el despliegue proporcionado por la red basándose en las observaciones de los modelos tridimensionales, se han generado varias figuras que confirman el rendimiento del integrador. Es importante destacar que, inicialmente y al mantenerse lo suficientemente alejados de la zona de carga, se obtienen los mejores resultados para los tres grupos de variables de estado (Figura C.12).

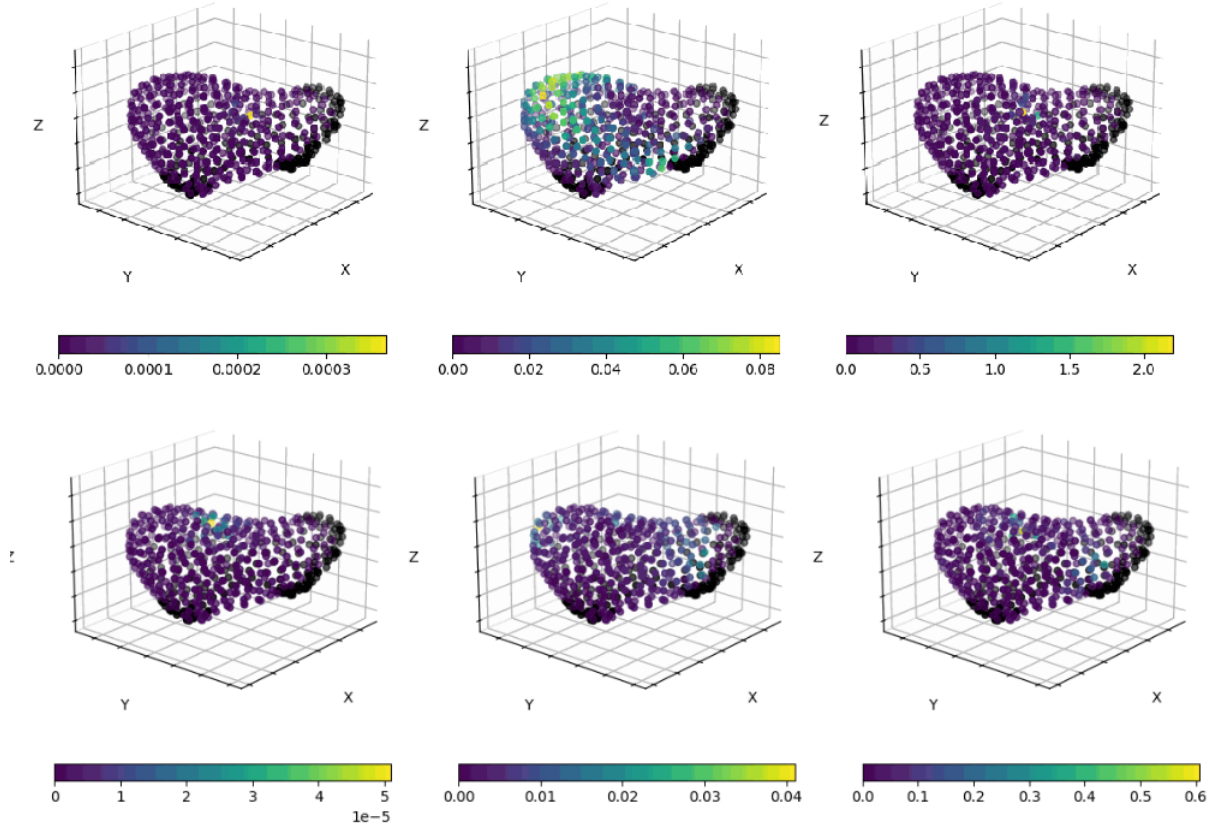


Figura C.10: Error relativo nodal para la inferencia de las dos simulaciones expuestas en la Figura C.9, con condiciones de contorno no vistas. A la izquierda para la norma de las tres dimensiones de posición, al centro para las tres dimensiones de velocidad y a la derecha para las seis del tensor de tensiones

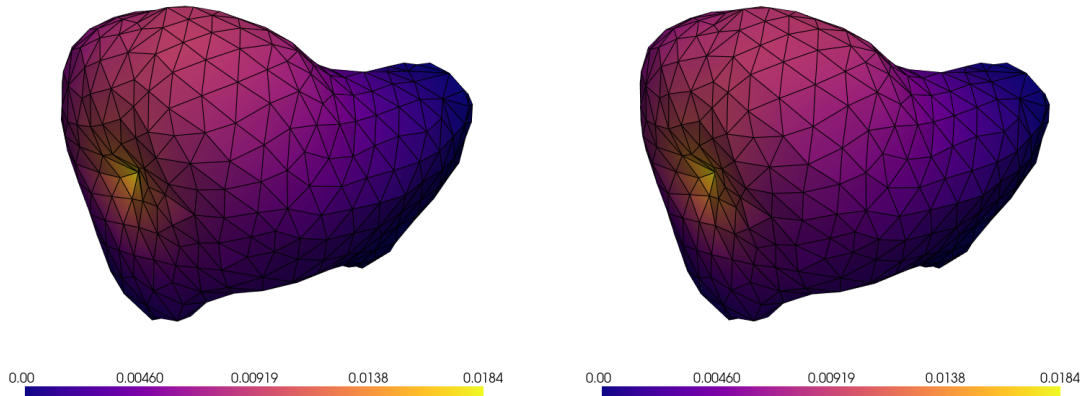


Figura C.11: Renderizado de la malla para comparar la solución numérica (izquierda) con la predicción de la red híbrida (derecha) en una situación de carga no entrenada. La escala de color representa el módulo de los desplazamientos en metros.

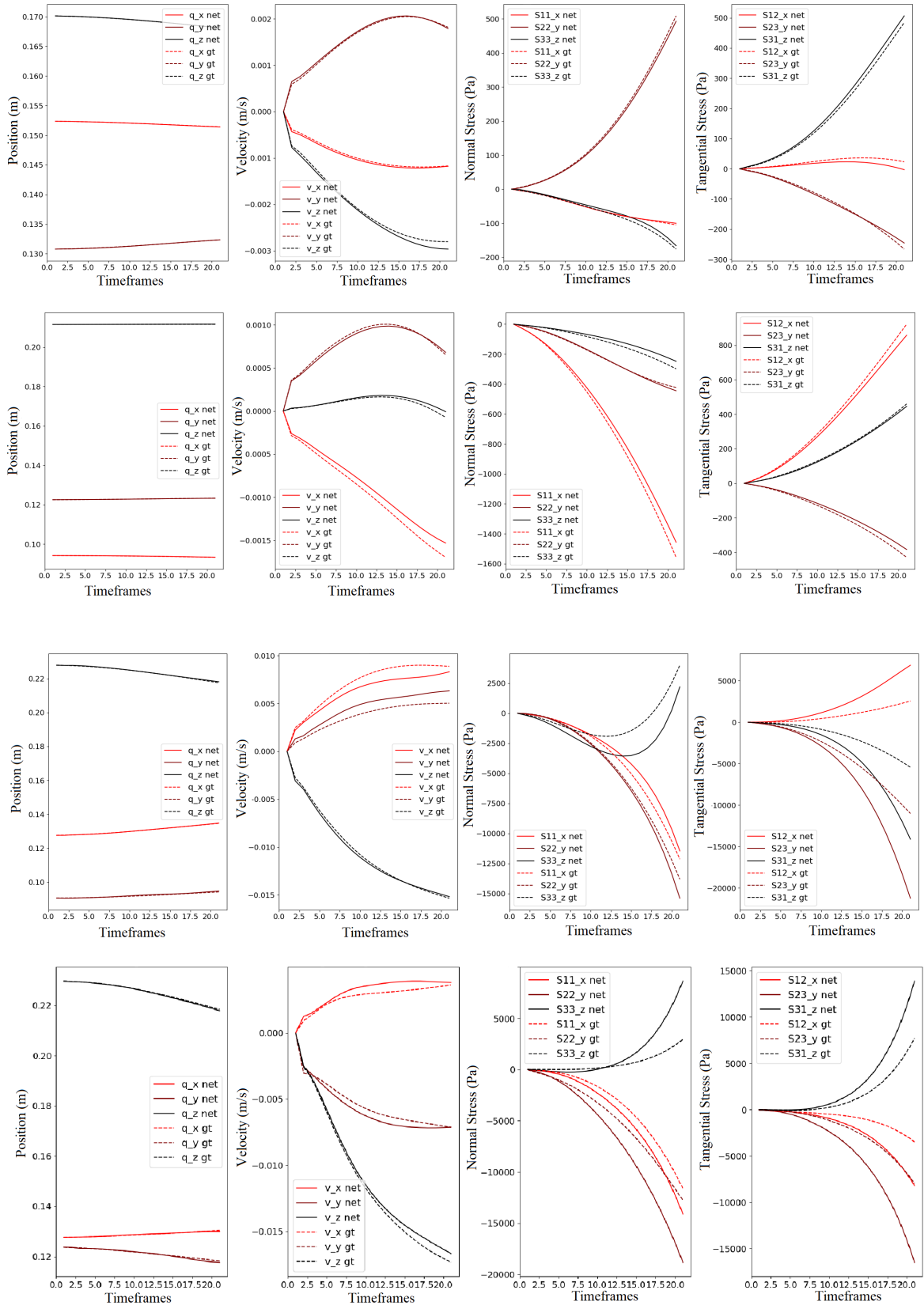


Figura C.12: Error relativo nodal para la inferencia de las dos simulaciones expuestas en la Figura C.9, con condiciones de contorno no vistas. Las dos primeras filas describen nodos alejados del desplazamiento impuesto, mientras que las dos siguientes son mucho mas próximos a la carga.

Anexos D

Algoritmos de entrenamiento, validación y testeo

D.1. Redes guiadas por datos

A continuación, se presentan los pseudocódigos que definen tanto el algoritmo de entrenamiento [1] como el de testeo de los resultados del modelo [2]:

Algorithm 1 Pseudocódigo para el algoritmo de entrenamiento

```
1: Extracción de los datos de entrenamiento:  $z$ 
2: Definir arquitectura y coste.
3: Definir hiperparámetros:  $N_{hidden}, N_{layers}, n_{epoch}, lr, \lambda_d...etc$ 
4: for época  $\leftarrow 1$  to  $n_{epoch}$  do
5:   for caso de entrenamiento  $\leftarrow 1$  to  $N_{train}$  do
6:     for snapshot  $\leftarrow 1$  to  $N_T$  do
7:       Normalización de los datos:  $z_{GT}^n$ 
8:       Propagación hacia adelante:  $[\frac{\partial z}{\partial t}] \leftarrow \text{Net}(z_{GT}^n)$ 
9:       Integrador temporal:  $z_{net}^{n+1} \leftarrow z_{net}^n + \Delta t \frac{\partial z}{\partial t}$ 
10:      Actualización del coste:  $L_{data} \leftarrow L_{data} + L_{data}^n$ 
11:    end for
12:    Función de coste:  $L \leftarrow L_{data}$ 
13:    Retropropagación
14:    Actualización del optimizador
15:  end for
16: end for
```

Algorithm 2 Pseudocódigo para el algoritmo de test

```
1: Cargar base de datos de test:  $z$ 
2: Cargar parámetros de la red
3: for caso de test  $\leftarrow 1$  to  $N_{\text{test}}$  do
4:   Inicializar vector de estado:  $z_{\text{net}}^0 \leftarrow z_{GT}^0$ 
5:   for snapshot  $\leftarrow 1$  to  $N_T$  do
6:     Propagación hacia adelante:  $[\frac{\partial z}{\partial t}] \leftarrow \text{Net}(z_{GT}^n)$ 
7:     Integrador temporal:  $z_{\text{net}}^{n+1} \leftarrow z_{\text{net}}^n + \Delta t \frac{\partial z}{\partial t}$ 
8:     Actualizar vector de estado:  $z_{\text{net}}^n \leftarrow z_{\text{net}}^{n+1}$ 
9:     Actualizar paso de tiempo:  $n \leftarrow n + 1$ 
10:  end for
11:  Calcular:  $MSE_{\text{sim}}$ 
12: end for
13: Calcular:  $MSE_{\text{test}}$ 
14: Calcular:  $error_{\gamma}$ 
```

En su conjunto, y para todas las arquitecturas informadas por datos, estos algoritmos permiten a la red aprender a construir el integrador temporal.

D.2. Redes híbridas

A continuación, se presentan los pseudocódigos que definen tanto el algoritmo de entrenamiento [3] como el de testeo de los resultados del modelo [4]:

Algorithm 3 Pseudocódigo para el algoritmo de entrenamiento

```
1: Extracción de los datos de entrenamiento:  $z$ 
2: Definir arquitectura y coste.
3: Definir hiperparámetros:  $N_{\text{hidden}}, N_{\text{layers}}, n_{\text{epoch}}, lr, \lambda_d \dots etc$ 
4: for época  $\leftarrow 1$  to  $n_{\text{epoch}}$  do
5:   for caso de entrenamiento  $\leftarrow 1$  to  $N_{\text{train}}$  do
6:     for snapshot  $\leftarrow 1$  to  $N_T$  do
7:       Normalización de los datos:  $z_{GT}^n$ 
8:       Propagación hacia adelante:  $[l_{\text{net}}^n, m_{\text{net}}^n, dE_{\text{net}}^n, dS_{\text{net}}^n] \leftarrow \text{Net}(z_{GT}^n)$ 
9:       Reparametrización:  $[L_{\text{net}}^n, M_{\text{net}}^n] \leftarrow [l_{\text{net}}^n, m_{\text{net}}^n]$ 
10:      Integrador temporal:  $z_{\text{net}}^{n+1} \leftarrow z_{\text{net}}^n + \Delta t (L \cdot dE_{\text{net}}^n + M \cdot dS_{\text{net}}^n)$ 
11:      Actualización del coste:  $L_{\text{data}} \leftarrow L_{\text{data}} + L_{\text{data}}^n$ 
12:      Actualización del coste de degeneración:  $L_{\text{degen}} \leftarrow L_{\text{degen}} + L_{\text{degen}}^n$ 
13:    end for
14:    Función de multi-coste agrupada:  $L \leftarrow \lambda_d L_{\text{data}} + L_{\text{degen}}$ 
15:    Retropropagación
16:    Actualización del optimizador
17:  end for
18: end for
```

Algorithm 4 Pseudocódigo para el algoritmo de test

```
1: Cargar base de datos de test:  $z$ 
2: Cargar parámetros de la red
3: for caso de test  $\leftarrow 1$  to  $N_{\text{test}}$  do
4:   Inicializar vector de estado:  $z_{\text{net}}^0 \leftarrow z_{GT}^0$ 
5:   for snapshot  $\leftarrow 1$  to  $N_T$  do
6:     Propagación hacia adelante:  $[l_{\text{net}}^n, m_{\text{net}}^n, dE_{\text{net}}^n, dS_{\text{net}}^n] \leftarrow \text{Net}(z_{GT}^n)$ 
7:     Reparametrización:  $[L_{\text{net}}^n, M_{\text{net}}^n] \leftarrow [l_{\text{net}}^n, m_{\text{net}}^n]$ 
8:     Integrador temporal:  $z_{\text{net}}^{n+1} \leftarrow z_{\text{net}}^n + \Delta t(L \cdot dE_{\text{net}}^n + M \cdot dS_{\text{net}}^n)$ 
9:     Actualizar vector de estado:  $z_{\text{net}}^n \leftarrow z_{\text{net}}^{n+1}$ 
10:    Actualizar paso de tiempo:  $n \leftarrow n + 1$ 
11:   end for
12:   Calcular:  $MSE_{\text{sim}}$ 
13: end for
14: Calcular:  $MSE_{\text{test}}$ 
15: Calcular:  $error_{\gamma}$ 
```

En su conjunto, y para todas las arquitecturas informadas por física, estos algoritmos permiten a la red aprender a construir el integrador temporal.

Anexos E

Marco teórico

E.1. Modelo biológico

El concepto de «gemelo digital» engloba una gran cantidad de modelos y métodos. No obstante, en el contexto de este trabajo, se define como una copia virtual de un órgano, con la finalidad de replicar con la mayor precisión su comportamiento biomecánico.

El enfoque seleccionado para llevar a cabo el estudio se centra en el modelo de un hígado, simplificando su composición a la de un tejido hepático homogéneo en todo su dominio sin tener en cuenta vascularización, lípidos, tendones ni musculatura circundante.

E.1.1. Anatomía y fisiología del hígado

El hígado desempeña un papel central en una amplia variedad de procesos fisiológicos. Estos incluyen el metabolismo de macronutrientes, la regulación del volumen sanguíneo, el apoyo al sistema inmunológico, el control endocrino de las vías de señalización del crecimiento, la homeostasis de lípidos y colesterol, y la descomposición de compuestos xenobióticos, incluyendo muchos medicamentos actuales [29]. Este conjunto de funciones vitales, además de su apoyo a los procesos digestivos y linfáticos, lo convierten en un órgano clave para el correcto funcionamiento del cuerpo humano.

Además de esta clara importancia fisiológica, el hígado es la víscera más voluminosa y pesada del cuerpo. Dividiendo su geometría en dos claros lóbulos, el izquierdo y el derecho, además de dos lobulillos menores, el cuadrado y el caudado (Figura E.1). Dicha separación lobular esta llevada a cabo por el ligamento falciforme, el cual tiene como función adicional el unir su cara diafragmática al propio diafragma. Por otro lado

la cara visceral contiene a la vacularización principal del hígado, la vena y la arteria hepática, además del complejo biliar.

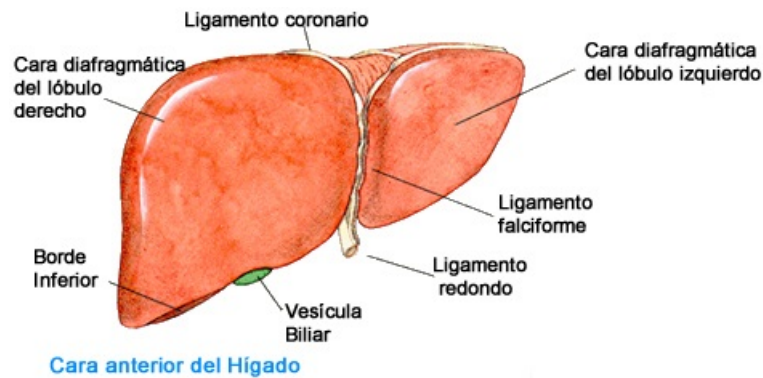


Figura E.1: Esquema anatómico del hígado (fuente: SlideShare, recurso educativo).

El hígado está anclado a otras estructuras anatómicas mediante diversos tejidos, lo que proporciona estabilidad. La figura muestra que varios ligamentos se conectan al hígado, contribuyendo a su fijación. Además de estos ligamentos, la propia vascularización del hígado desempeña un papel significativo en esta sujeción [28].

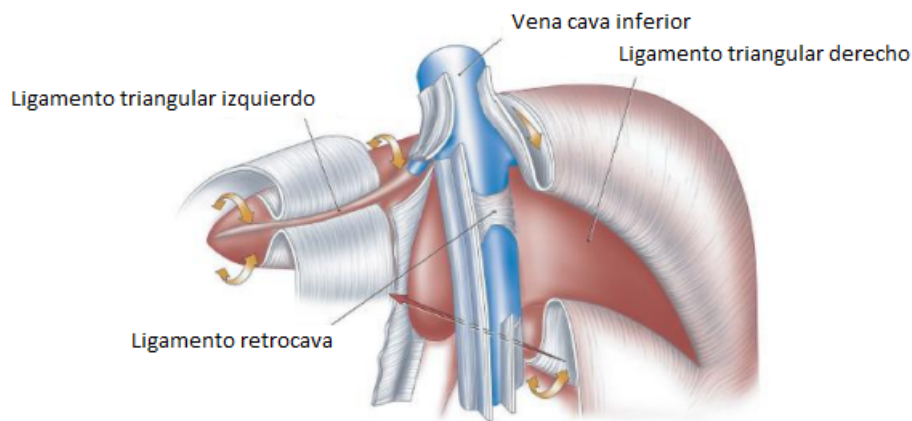


Figura E.2: Fijaciones anatómicas del hígado al cuerpo humano [28].

E.1.2. Propiedades de los tejidos hepáticos

Entrando en un análisis a menor escala, las células del hígado están organizadas alrededor de la unidad funcional del hígado, que es el lóbulo. Este consiste en cordones de hepatocitos organizados en una forma hexagonal alrededor de la vena central (Figura E.3). En los vértices de este hexágono se encuentran los tríos portales, que consisten

en ramificaciones cercanas de la arteria hepática, la vena porta y los conductos biliares [29].

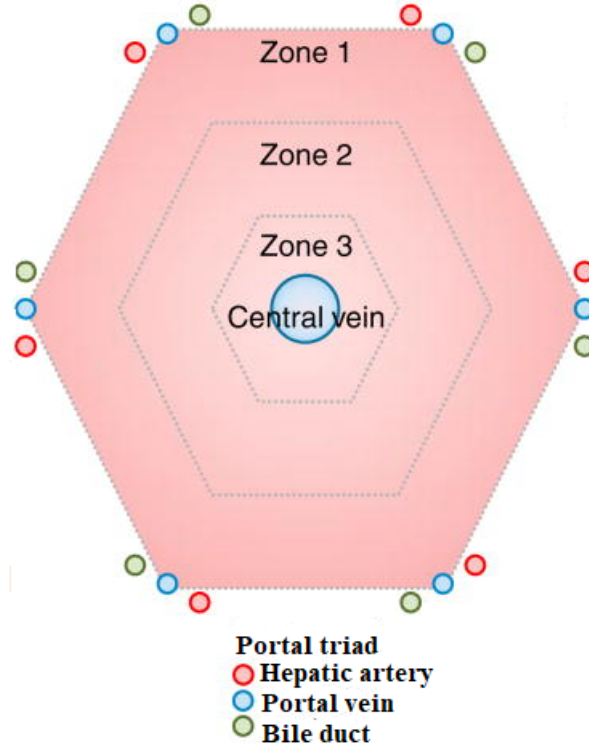


Figura E.3: Representación geométrica de un lóbulo hepático [29].

Esta intrincada arquitectura, donde las células hepáticas interactúan con su matriz extracelular y con todos los fluidos circundantes, hace que su caracterización siga siendo sumamente compleja, incluso en la actualidad. La mayoría de modelos no lineales dedicados a tejidos biológicos blandos se basan en la identificación de las leyes constitutivas no lineales, que están definidas mediante una función específica de energía de deformación: W [30].

Una de las opciones más populares para aproximar este comportamiento no lineal son los modelos hiperelásticos [30], los cuales definen la energía de deformación con diferentes formas polinómicas. Aunque su aplicabilidad a tejidos blandos biológicos esta generalmente delimitada a considerar pequeñas deformaciones.

$$W_{NH} = \frac{\mu}{2}(I_1 - 3) + \frac{1}{d}(J - 1)^2 \quad (\text{E.1})$$

El modelo neo-Hookeano, Ec. (E.1) es la función polinómica más simple, basada en la termodinámica estadística y la conservación de la entropía de las cadenas de polímeros entrecruzados, donde μ es el módulo de corte inicial del material, I_1 es el

primer invariante de la tensión desviadora, d es un parámetro de incompresibilidad del material y J es el determinante del gradiente de deformación elástica [31]

$$W_{MR} = \sum_{i,j=0}^N C_{ij}(I_1 - 3)^i(I_2 - 3)^j + \sum_{i=1}^N \frac{1}{d_i}(J - 1)^{2i} \quad (\text{E.2})$$

El modelo Mooney-Rivlin (Ecuación E.2), el cual consiste en una generalización del modelo Neo-Hookeano mediante la adición del segundo invariante del tensor de estiramiento, donde N representa el orden del modelo, C_{ij} son constantes del material y I_2 es el segundo invariante del tensor desviador [31]

$$W_{Ogden} = \sum_{i=1}^N \frac{\mu_i}{\alpha_i}(\lambda_1^{\alpha_i} + \lambda_2^{\alpha_i} + \lambda_3^{\alpha_i} - 3) + \sum_{i=1}^N \frac{1}{d_i}(J - 1)^{2i} \quad (\text{E.3})$$

El modelo de Ogden, Ec. (E.3), por otro lado, es una generalización aún más amplia y flexible. Utiliza una función de energía de deformación que incluye múltiples parámetros, entre ellos α_i , μ_i y d_i , los cuales corresponden a diferentes propiedades del material, mientras que λ_{1-3} denotan los ratios principales de deformación del material.

Adicionalmente, la presencia de fluidos tanto en la matriz extra-celular como en la propia vascularización revelan que estos tejidos pueden tener un comportamiento viscoelástico. Este modelo biomecánico inicialmente propuesto por Fung[32] y correctamente aplicado a una gran variedad de tejidos biológicos[34][33] define que el comportamiento mecánico se puede subdividir en dos partes: una respuesta no lineal elástica y otra dependiente del tiempo que se relaciona con el comportamiento disipativo del tejido blando[31]. Este último componente del comportamiento se modela con una función de relajación $G(t)$, la cual se expresa para este trabajo como una serie de Prony

$$G(t) = G_0(\alpha_\infty + \sum_{k=1}^N \alpha_k e^{-t/\tau_k}) \quad (\text{E.4})$$

G_0 representa el módulo de corte en el límite de carga rápida, mientras que α_∞ , que resulta de operar $\alpha_\infty = \frac{G_\infty}{G_0}$, es el módulo relativo para cuando $t = \infty$. α_k y τ_k son los coeficientes de la serie de Prony y constantes dependientes del material en cuestión.

Es debido a esta naturaleza mixta entre un modelo hiperelástico y uno viscoelástico que Martínez Martínez[31] presenta durante su tesis un modelo visco-hiperelástico que reproduce satisfactoriamente el comportamiento del tejido hepático.

Los resultados obtenidos en la tesis muestran una marcada variabilidad entre los ensayos uniaxiales realizados en un mismo tejido (Figura E.4). Sin embargo, al calcular un promedio de todos los ensayos, se logra obtener una aproximación satisfactoria de

su comportamiento hiperelástico (Figura E.5). Como resultado, se ha seleccionado el modelo de Ogden como la representación elegida para describir la hiperelasticidad del tejido.

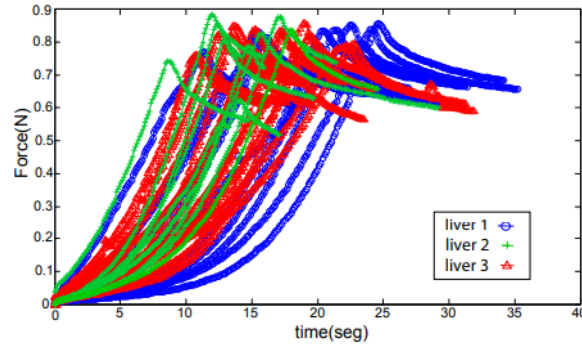


Figura E.4: Curvas de fuerza-tiempo adquiridas a partir de la prueba de tracción uniaxial para las 39 muestras de los tres hígados de cordero [31].

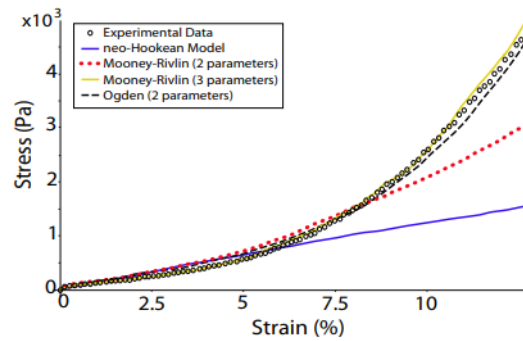


Figura E.5: Curvas de ajuste de los modelos hiperelásticos a partir de datos experimentales de prueba de tensión-deformación uniaxial ex vivo [31].

Por otra parte, el comportamiento viscoelástico experimenta esta misma variabilidad intra-muestral que el hiperelástico, Ec. (E.6). Análogamente se extrae numéricamente el promedio y se obtienen los coeficientes de la serie de Prony que mejor ajustan dicha media.

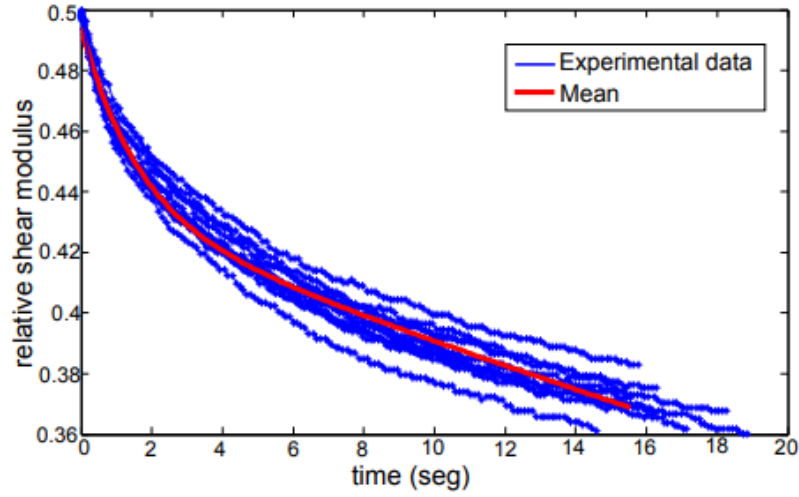


Figura E.6: Curva de ajuste de los datos de pruebas de tensión uniaxial para el modelado de la dependencia temporal de uno de los tres hígados testeados por Martinez Martinez [31].

En síntesis de estos resultados de Martinez Martinez [31], se concluyen como coeficientes de la respuesta viscoelástica los siguientes:

$$\mu_L(P_a) = 364,74 \quad (\text{E.5})$$

$$\alpha_L = 16,19 \quad (\text{E.6})$$

$$d(P_{a-1}) = 9,525 \times 10^{-5} \quad (\text{E.7})$$

Mientras que la respuesta elástica será modelada por una serie de Prony con las siguientes constantes:

$$\alpha_1 = 0,56057 \quad (\text{E.8})$$

$$\tau_1(s) = 6,36173 \quad (\text{E.9})$$

$$\alpha_2 = 0,40585 \quad (\text{E.10})$$

$$\tau_2(s) = 0,73348 \quad (\text{E.11})$$

E.2. Características generales de las redes neuronales

E.2.1. Teoría básica de las redes neuronales, el perceptrón multicapa

En los años 40 Warren McCulloch junto con Walter Pitts crearon un modelo que representaba matemáticamente el funcionamiento de las neuronas [26]. Era simple, una ecuación lineal que dependía de una serie de pesos (tantos como entradas tuviera la neurona) y de un sesgo. Posteriormente se añadió una función de activación a la salida

de esta operación, lo que atribuía al sistema la capacidad de representar sistemas no lineales (Figura E.7).

$$\begin{aligned} z_j^{(l)} &= \sum_{i=1}^{n^{(l-1)}} w_{ij}^{(l)} a_i^{(l-1)} + b_j^{(l)} \\ a_j^{(l)} &= f(z_j^{(l)}) \end{aligned} \tag{E.12}$$

Donde:

$z_j^{(l)}$ es la entrada neta (ponderada) a la neurona j en la capa l .

$w_{ij}^{(l)}$ es el peso de la conexión entre la neurona i en la capa $l - 1$ y la neurona j en la capa l .

$a_i^{(l-1)}$ es la salida de la neurona i en la capa $l - 1$.

$b_j^{(l)}$ es el sesgo de la neurona j en la capa l .

$a_j^{(l)}$ es la salida de la neurona j en la capa l .

$f(\cdot)$ es la función de activación.

Una década después, Frank Rosenblatt desarrolló el primer algoritmo que permitía a esta ecuación aprender, modificando tanto los pesos como el sesgo y creando el llamado “perceptrón” [27]. La agrupación de distintas capas de estos perceptrones forma lo denominado “perceptrón multicapa” o “MLPs”, siendo la base de todos los modelos de redes neuronales establecidos:

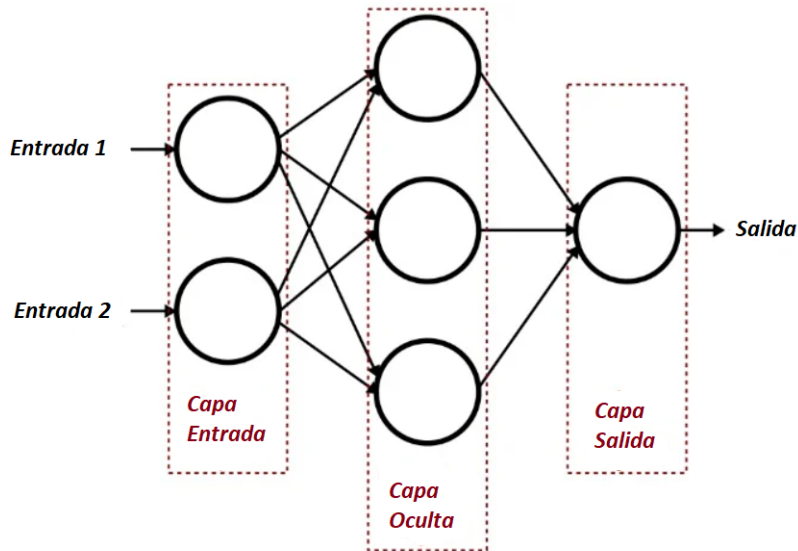


Figura E.7: Esquema simple de un perceptrón multicapa con varias neuronas de entrada y una única salida.

En el contexto del proyecto, la MLP será el componente principal en la mayoría de arquitecturas. Sin embargo en los procesadores de las redes de grafos se definirá una modificación que es la MLP o “perceptrón multicapa con conexiones residuales”, lo que simplemente actualiza el estado de los datos de entrada x_i mediante una agregación de la salida de la red, x'_i (Figura E.8).

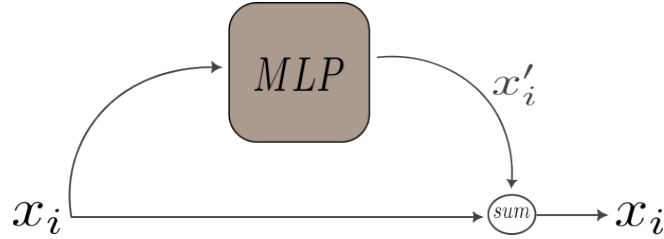


Figura E.8: Diagrama de bloques de una MLP con conexión residual, donde la salida x'_i actualiza el estado de entrada mediante una suma, por lo que la dimensión de entrada y salida deberá ser la misma.

Esta técnica fue popularizada con las redes residuales y ayuda a evitar la pérdida de información cuando empieza a tener una cierta profundidad. Aliviando el problema de la degradación de gradiente y mejorando el flujo de información del algoritmo de aprendizaje.

E.2.2. Encoder-decoder y el espacio latente

Este simple modelo se acaba generalizando según la complejidad y necesidades del problema. La arquitectura de “Encoder-Decoder” es un marco conceptual comúnmente utilizado en el campo de las redes neuronales y mayormente utilizada en el campo del procesamiento del lenguaje natural (NLP) y de la visión por computador.

Definiendo sus dos principales componentes, el “Encoder” o Codificador es responsable de condensar la información de entrada en un espacio latente, por lo general, con dimensiones distintas a las de la entrada de datos. Mientras que el “Decoder” es el encargado de traducir la información del espacio latente en la variable de salida esperada para el problema (Figura E.9).

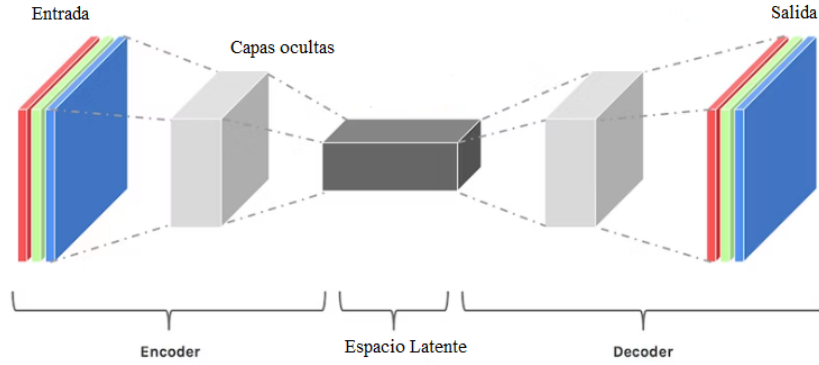


Figura E.9: Esquema de Encoder-Decoder para la transformación de una entrada RGB hacia un espacio latente con dimensiones reducidas y mayor cantidad de canales. Esto permite una mayor expresividad de los distintos componentes principales. Este enfoque es similar al utilizado en el proyecto, pero emplea redes convolucionales en lugar de Perceptrones Multicapa (MLPs), dado que los datos del ejemplo son bidimensionales (imágenes) en lugar de unidimensionales (vectores).

Concretamente para este proyecto, tanto el codificador como el decodificador están compuestos por una MLP definida por dos hiperparámetros: la cantidad de capas ocultas y la dimensión de dicha capa oculta. Siendo, el ultimo, el encargado de definir la dimensión del espacio latente para los codificadores empleados.

Esta metodología de transformación de datos se denomina, en su conjunto, “autoencoder” o auto-codificador.

E.2.3. Redes de grafos y el sesgo geométrico inductivo

Generalmente las arquitecturas basadas en perceptrones multicapa son agnósticas a la estructura de los datos, asumiendo como *prior* que no hay una relación inicial entre los datos de entrada. Es por ello que en estas capas densas, todas las neuronas están conectadas con todas y es tarea de la red encontrar estas relaciones o patrones de información. Sin embargo durante los últimos años se han explotado diferentes arquitecturas que para diferentes estructuras de datos deciden imponer un sesgo inicial al entrenamiento, un sesgo geométrico, superando el rendimiento de las MLPs y estableciéndose como las arquitecturas más dominantes para cada uno de sus campos.

Este sesgo capacita a las redes convolucionales para detectar características locales en una imagen al entender la relación entre píxeles cercanos. Del mismo modo, habilita a las redes recurrentes para comprender la temporalidad de los datos y en el contexto de este proyecto, este sesgo posibilita a los grafos establecer relaciones específicas o predeterminadas entre todos los elementos de su sistema (Figura E.10).

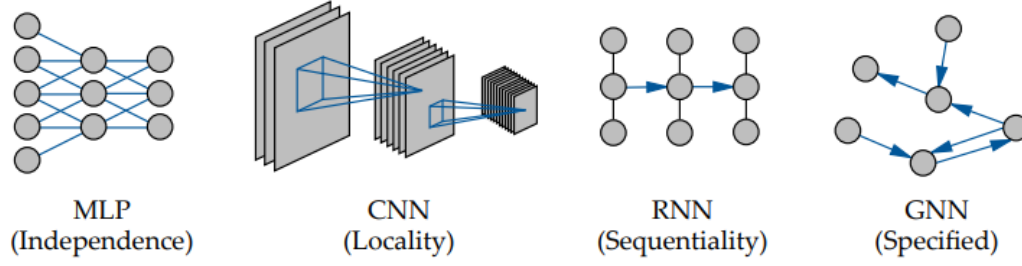


Figura E.10: Esquematización de las diferentes arquitecturas y sus sesgos geométricos.

Los grafos son estructuras matemáticas que consisten en nodos (también llamados vértices, V), aristas (también llamadas bordes, ε) que conectan estos nodos y características globales que afectan a todo el grafo, u . La utilidad principal es que estas estructuras son utilizadas para modelar relaciones entre entidades, es decir, son capaces de transmitir mensajes entre nodos y aristas según su función de agregación. Por lo tanto, cuantas más iteraciones de esta transmisión de mensajes se realicen, más información tendrá cada elemento de la red sobre su entorno. Con ello se induce un sesgo geométrico al entrenamiento en relación a una matriz de conectividad, A , que puede o no tener en cuenta la información propia de cada nodo durante la agregación,

$$A = \{a_{ij} = 1 \text{ si } (i, j) \in E \text{ si no, } 0\} \quad (\text{E.13})$$

Dicha característica presenta un símil con la discretización espacial del método de elementos finitos, los elementos de una malla transmiten las cargas de unos a otros y tienen en cuenta dicha conectividad. Como ya se ha definido, el problema físico expuesto está definido por una serie de variables de estado: la velocidad, el tensor de tensiones y la fuerzas exteriores se procesaran en los nodos de la arquitectura, mientras que el desplazamiento relativo y las deformaciones entre nodos se procesarán en los vértices.

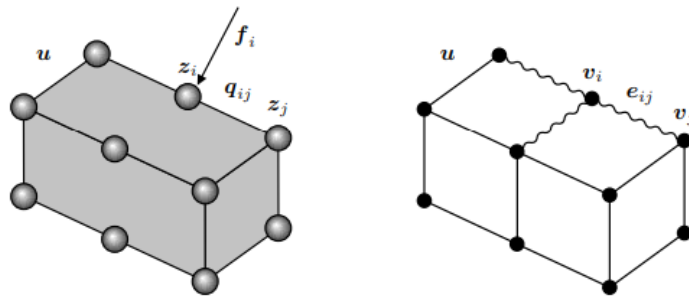


Figura E.11: Dominio de un sistema físico discretizado en una malla con variables de estado nodales z_i , distancias nodales relativas q_{ij} , interacciones externas f_i y propiedades globales u (izquierda). Representación del mismo sistema con atributos de nodo y arista: v_i y e_{ij} (derecha).

E.2.4. Redes informadas por física y la inteligencia artificial híbrida

Las Redes Neuronales Informadas por Física (PINNs, por su terminología anglosajona) son una popular aplicación de las redes neuronales en la resolución de problemas de física y matemáticas. Estas redes incorporan directamente las leyes físicas o ecuaciones matemáticas relevantes como parte integral de su arquitectura. Minimizando el residuo de la ecuación diferencial y satisfaciendo tanto las condiciones iniciales, como las de contorno como funciones de coste complementarias a la optimización de los datos.

Estas redes son especialmente útiles para resolver Ecuaciones en Derivadas Parciales (EDPs) o Ecuaciones Diferenciales Ordinarias (EDOs), cuyas soluciones teóricas suelen ser difíciles de encontrar debido a su complejidad exponencial.

Este planteamiento discierne de la idea de una caja negra que resuelve un problema dado. Al incorporar la EDP a la estructura de la red, no solo se impone que el flujo de datos respete la estructura matemática de la EDP, sino que es crucial construir una función de coste que al minimizarla, satisfaga la ecuación diferencial. En consecuencia, la nueva función, ahora denominada de multi-coste $\mathcal{L}$, esta compuesta por hasta cuatro términos: la función de pérdida, $\mathcal{L}_z$, el coste asociado al residuo de la EDP, $\mathcal{L}_{EDP}$, el coste asociado a las condiciones de contorno, $\mathcal{L}_{BC}$ y el coste asociado a las condiciones iniciales, $\mathcal{L}_{BI}$

$$\mathcal{L} = \mathcal{L}_z + \lambda_{EDP}\mathcal{L}_{EDP} + \lambda_{BC}\mathcal{L}_{BC} + \lambda_{CI}\mathcal{L}_{CI} \quad (\text{E.14})$$

Siendo λ el parámetro de regularización del coste asociado a la EDP, de tal forma hiperparametrizando la influencia de la física durante el entrenamiento.

Esta influencia física, es denominada también como sesgo físico. Lo que en conjunto con el sesgo geométrico constituyen un ejemplo perfecto de como la simbiosis entre el conocimiento humano y la inteligencia artificial, creando una potente herramienta que se enmarca dentro del concepto teórico de la inteligencia artificial híbrida.

Anexos F

Justificación de un sistema cerrado por medio del port metripleptico

F.1. Formalismo GENERIC para un sistema abierto

En el contexto del proyecto, se asume el comportamiento mecánico de un tejido hepático como hiperelástico. Con una componente viscoelástica disipativa que es función de la velocidad del sistema. Donde el formalismo ya mencionado busca establecer una función $f(\mathbf{z}, t)$ que satisfaga la termodinámica de un sistema cerrado en desequilibrio

$$\dot{\mathbf{z}} = \frac{dz}{dt} = f(\mathbf{z}, t), \quad \mathbf{z}(0) = \mathbf{z}_0 \quad (\text{F.1})$$

No obstante, el sistema propuesto no se define a priori como cerrado, ya que el desplazamiento impuesto está introduciendo energía en el tejido y en consecuencia su Hamiltoniano no es constante en el tiempo, o lo que es lo mismo, su energía mecánica no es constante en el tiempo.

En consecuencia a este planteamiento, surgen las primeras aproximaciones a una re-formulación del formalismo para sistemas abiertos [37, 39, 38]. Cuya solución deriva en el uso de los operadores de frontera (Γ_i) como puertas de intercambio de energía y entropía entre los sub-sistemas locales (Ω_i , Figura F.1). Por lo que definiendo el sistema global como (Ω_{global}) y sus subsistemas locales como Ω_1 (tejido hepático) y Ω_2 (carga), junto con el operador de frontera entre ambos (Γ), se satisface que:

$$\dot{\mathbf{z}}_1 = \{\mathbf{z}, E\} + [\mathbf{z}, S] = \{\mathbf{z}, E\}_1 + [\mathbf{z}, S]_1 + \{\mathbf{z}, E\}_\Gamma + [\mathbf{z}, S]_\Gamma \quad (\text{F.2})$$

$$\dot{\mathbf{z}}_2 = \{\mathbf{z}, E\} + [\mathbf{z}, S] = \{\mathbf{z}, E\}_2 + [\mathbf{z}, S]_2 + \{\mathbf{z}, E\}_\Gamma + [\mathbf{z}, S]_\Gamma \quad (\text{F.3})$$

Lo que transformado a la notación ya vista, con los operadores lineales resulta para

el subsistema hepático en:

$$\dot{\mathbf{z}}_1 = \mathbf{L} \frac{\partial E}{\partial \mathbf{z}} + \mathbf{M} \frac{\partial S}{\partial \mathbf{z}} = \mathbf{L}_1 \frac{\partial E_1}{\partial \mathbf{z}_1} + \mathbf{M}_1 \frac{\partial S_1}{\partial \mathbf{z}_1} - \mathbf{L} \Gamma \frac{\partial E \Gamma}{\partial \mathbf{z}} - \mathbf{M} \Gamma \frac{\partial S \Gamma}{\partial \mathbf{z}} \quad (\text{F.4})$$

Siendo ahora satisfechas las condiciones de degeneración en el sistema global tal que el primer y segundo principio de la termodinámica se cumple según:

$$\mathbf{L}_{\Omega_1}(\mathbf{z}) \frac{\partial S_{\Omega_1}}{\partial \mathbf{z}} = 0, \quad (\text{F.5})$$

$$\mathbf{M}_{\Omega_1}(\mathbf{z}) \frac{\partial E_{\Omega_1}}{\partial \mathbf{z}} = 0 \quad (\text{F.6})$$

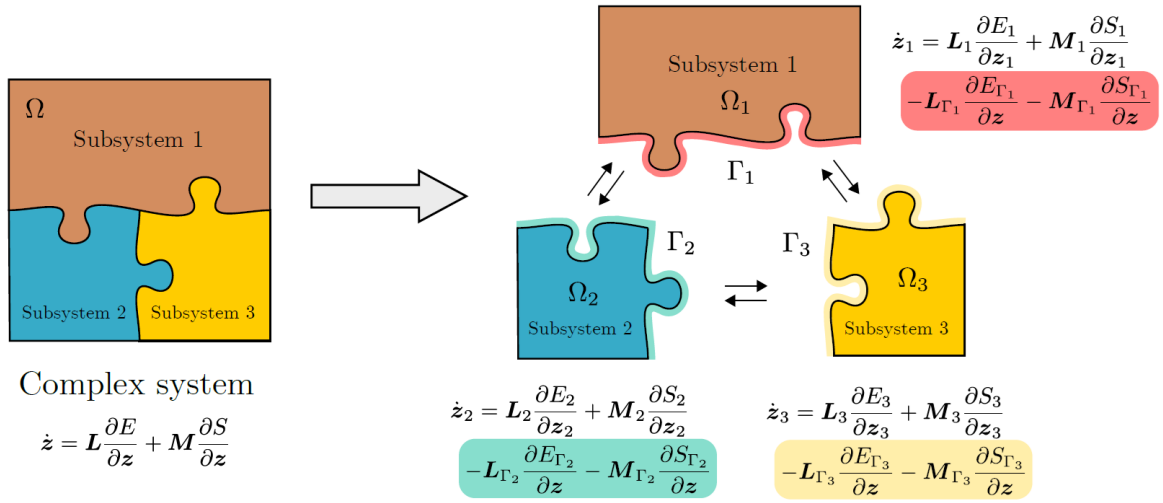


Figura F.1: Esquematización de un sistema complejo creado como un grupo conectado de subsistemas Ω_i . La dinámica de cualquier subsistema se describe utilizando una formulación GENERIC que incluye términos conservativos y disipativos, teniendo también en cuenta las contribuciones externas en los términos de Γ . Los subsistemas se comunican entre sí a través de puertos mediante el intercambio de energía y entropía [38]

Por lo que en resumen y contextualizando con el aprendizaje automático, la salida de la red predecirá: $\mathbf{L}$, $\mathbf{M}$, $\mathbf{E}$ y $\mathbf{S}$, que según lo expuesto en la Ecuación F.4, satisface las condiciones de degeneración para $\dot{\mathbf{z}}_1$ y justifica el empleo del formalismo para un sistema cerrado compuesto por dos subsistemas abiertos.

Anexos G

Normalización de los datos de entrenamiento

G.1. Distribución de los datos

Las doce variables de estado del problema, definidas como $\mathbf{Z}^t\{\mathbf{q}_x, \mathbf{q}_y, \mathbf{q}_z, \mathbf{v}_x, \mathbf{v}_y, \mathbf{v}_z, \sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{xz}\}$, donde σ representa tensor de Cauchy, se agrupan en tres subconjuntos definidos por la homogeneidad de unidades físicas, de manera que $\mathbf{Z}$ se redefine como $\mathbf{Z}^t\{\mathbf{q}, \dot{\mathbf{q}}, \sigma\}$.

De esta forma, la evaluación de la distribución de cada componente de las variables de estado se realiza en una dimensión única, como se muestra en la Figura G.1. Se observa que ninguna de las tres componentes se ajusta a una distribución Gaussiana, y se destaca la alta desviación típica de las tensiones debido a los concentradores de tensión producidos por el desplazamiento impuesto.

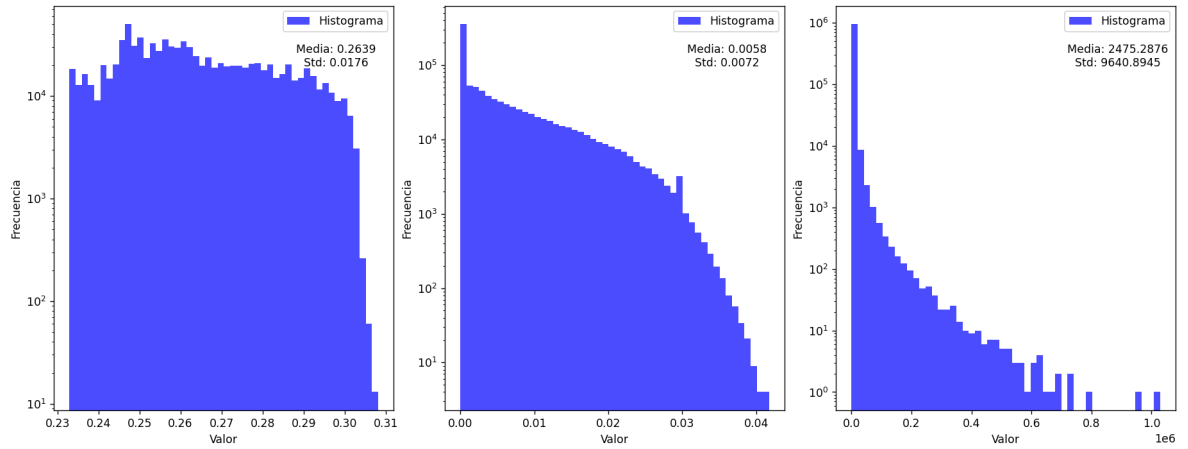


Figura G.1: Histogramas de las normas de las tres variables de estado en el conjunto de entrenamiento. A la izquierda posición, al centro velocidad y a la derecha tensiones.

Adicionalmente se evalúan la distribución de los desplazamientos u_{ij} , lo que resulta en una distribución Gaussiana.

G.2. Estandarización Z

Originalmente para el planteamiento de las TIGNNs [15] se establece, como base para la normalización de los datos, la conocida estandarización Z (G.1). Este modelo de normalización es un proceso estadístico que transforma una variable aleatoria para que tenga una media de cero y una desviación estándar de uno. Siendo realmente útil para comparar variables con diferentes unidades de medida, ya que todas quedan en una escala común,

$$z = \frac{X - \mu}{\sigma} \quad (\text{G.1})$$

donde:

$X > 0$ es la variable aleatoria.

μ es la media de la distribución logarítmica.

σ es la desviación estándar de la distribución logarítmica.

La fórmula para la estandarización z ajusta cada valor en función de la media y la desviación estándar de la distribución original (X , G.1). Esta normalización es especialmente útil cuando se trabaja con datos que se asumen tener una distribución Gaussiana (normal), sin embargo, cuando se desconoce la estructura de los mismos puede no resultar idónea y arrojar ruido durante la transformación.

Para un conjunto de datos de entrenamiento aleatorio se calcula su media y desviación típica antes y después de la normalización. Comprobando en la Tabla G.1 que las estadísticas normalizadas para una muestra de datos del tamaño de un “batch” o paquete.

Tabla G.1: Estadísticas de una muestra de datos de entrenamiento sin normalizar y normalizados con la estandarización Z.

Variable	Media	Desviación Estándar
v	-1.3407e-05	0.0028
σ	-499.4769	2047.6873
v (norm z)	0.1888	0.5044
σ (norm z)	-0.0278	0.3721

G.3. Normalización MinMax

Por otra parte, la normalización Min-Max es una técnica también utilizada en el preprocesamiento de datos para escalar valores dentro de un rango específico, concretamente para el problema presentado entre -1 y 1 (Ecuación G.3). Esto difiere del rango típico de 0 y 1 (Ecuación G.2) debido a que se busca mantener los valores nulos, además de conservar la importancia física del signo asociada a las variables de estado,

$$z = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \quad (\text{G.2})$$

Donde:

$X_{\min}$ es el mínimo de la distribución.

$X_{\max}$ es el máximo de la distribución.

$$z = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \times 2 - 1 \quad (\text{G.3})$$

A diferencia de la estandarización, la distribución resultante conserva la relación entre la media y la desviación típica. No obstante la sensibilidad a valores atípicos es realmente alta, lo que perjudica por igual a las velocidades y tensiones debido a la naturaleza de sus distribuciones. En la Tabla G.2 se contrasta la nueva desviación estándar de la muestra para velocidades y tensiones, concretamente con la MinMax expuesta en la Ecuación G.3.

Tabla G.2: Estadísticas de una muestra de datos de entrenamiento sin normalizar y normalizados con la normalización MinMax, para las tensiones y velocidades

Variable	Media	Desviación Estándar
$\dot{q}$	-1.3407e-05	0.0028
σ	-499.4769	2047.6873
$\dot{q}$ (Minmax)	-0.0010	0.0475
σ (Minmax)	-0.0006	0.0026

G.4. Otras normalizaciones

En esta sección se definen otras transformaciones que han sido testeadas en el transcurso del proyecto.

G.4.1. Box-Cox

La transformación Box-Cox busca estabilizar la varianza y hacer que los datos se ajusten mejor a la distribución normal. Definiendo un parámetro de transformación λ en base a una maximización de la función de verosimilitud o utilizando métodos de validación cruzada para encontrar el valor óptimo. Adicionalmente al tener un logaritmo, solo sirve para valores positivos no nulos, por lo que se establece un término ε para desplazar la distribución,

$$z(\lambda, \varepsilon) = \begin{cases} \frac{(X + \varepsilon)^\lambda - 1}{\lambda}, & \text{si } \lambda \neq 0 \\ \ln(X + \varepsilon), & \text{si } \lambda = 0 \end{cases}$$

Donde:

$X > 0$ es la variable original que deseamos transformar.

λ es el parámetro de transformación que determina la forma de la transformación.

ε es un pequeño valor positivo añadido a X para manejar valores cercanos a cero.

G.4.2. Log-Normal

Debido a los valores atípicos, se considera también una normalización con una distribución Log-normal, donde la función de densidad de probabilidad (PDF) está dada por:

$$f(x; \mu, \sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(\ln(x)-\mu)^2}{2\sigma^2}} \quad (\text{G.4})$$

Donde:

$x > 0$ es la variable aleatoria.

μ es la media de la distribución logarítmica.

σ es la desviación estándar de la distribución logarítmica.

Por lo que análogamente a la estandarización simple se puede fácilmente operar como:

$$z = \frac{\ln(X) - \mu}{\sigma} \quad (\text{G.5})$$

Donde:

$X > 0$ es la variable aleatoria.

μ es la media de la distribución logarítmica.

σ es la desviación estándar de la distribución logarítmica.

Anexos H

Robustez frente a cambios de geometría

El propósito de este Anexo es ofrecer una descripción más detallada de la robustez de las arquitecturas analizadas frente a diversas geometrías y, por ende, ante condiciones de contorno no vistas. De esta forma se busca delimitar su capacidad de generalización en términos de inferencia y cuantificar el aumento de error, tanto para modelos entrenados con una geometría, como para modelos con mayor variabilidad anatómica entre los datos de entrenamiento.

H.1. Modelos pre-entrenados con una geometría

Engloba, mas concretamente, a todos los modelos entrenados con un total de 85 simulaciones cuya geometría y malla es la misma.

H.1.1. Redes guiadas por datos

La primera parte aborda la evaluación de la robustez de las redes entrenadas con datos frente a situaciones de inferencia que no fueron contempladas durante el proceso de entrenamiento, tanto en términos de cargas como de geometrías.

Modelo Simplificado

Base de datos	γ_q	Vista
Geometría A	$5,47 \times 10^{-4}$	Si
Geometría B	$2,82 \times 10^{-3}$	No
Geometría C	$2,86 \times 10^{-3}$	No
Geometría D	$3,05 \times 10^{-3}$	No

Tabla H.1: Resultado en inferencia para los diferentes modelos propuestos, pero únicamente entrenando con datos pertenecientes a la geometría original como indica la columna *Vista*.

Resultando en un primer argumento que correla la evolución del error del algoritmo con el incremento en la variación de volumen.

Modelo General

Base de datos	γ_s	γ_q	$\gamma_{\dot{q}}$	Vista
Geometría A	0.253	$5,96 \times 10^{-4}$	$7,21 \times 10^{-2}$	Si
Geometría B	0.45	$2,55 \times 10^{-3}$	$2,51 \times 10^{-1}$	No
Geometría C	0.658	$1,97 \times 10^{-3}$	$2,06 \times 10^{-1}$	No
Geometría D	0.775	$2,47 \times 10^{-3}$	$2,89 \times 10^{-1}$	No

Tabla H.2: Resultado en inferencia para los diferentes modelos propuestos, pero únicamente entrenando con datos pertenecientes a la geometría original como indica la columna *Vista*.

Donde en mayor detalle y con ayuda de la Figura H.2, se valida la obtención de predicciones más precisas para condiciones de carga más centradas, lo que está en línea con la hipótesis de que estas son las cargas más comunes durante los entrenamientos. Esta característica deriva de la configuración de la base de datos, ya que al alejar la zona de carga de la zona de encastre con un cierto margen, el grupo de nodos seleccionados adquiere una forma cónica suavizada, conllevando a una mayor cantidad de nodos cuyo comportamiento es similar alrededor de la zona central. Esta variabilidad entre buenas y malas métricas se acentúa para las tensiones, como se observa en el gráfico de caja respectivo (Figura H.1).

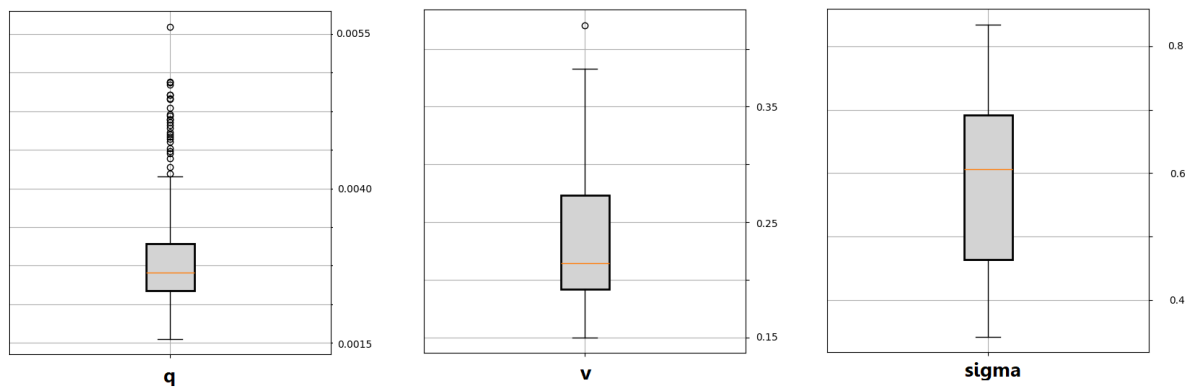


Figura H.1: Gráfico de caja del error relativo medio de cada simulación testeada para el conjunto de geometrías no vistas (B, C y D) de las redes guiadas por datos. A la izquierda para posición, en el centro para velocidad y a la derecha para tensiones.

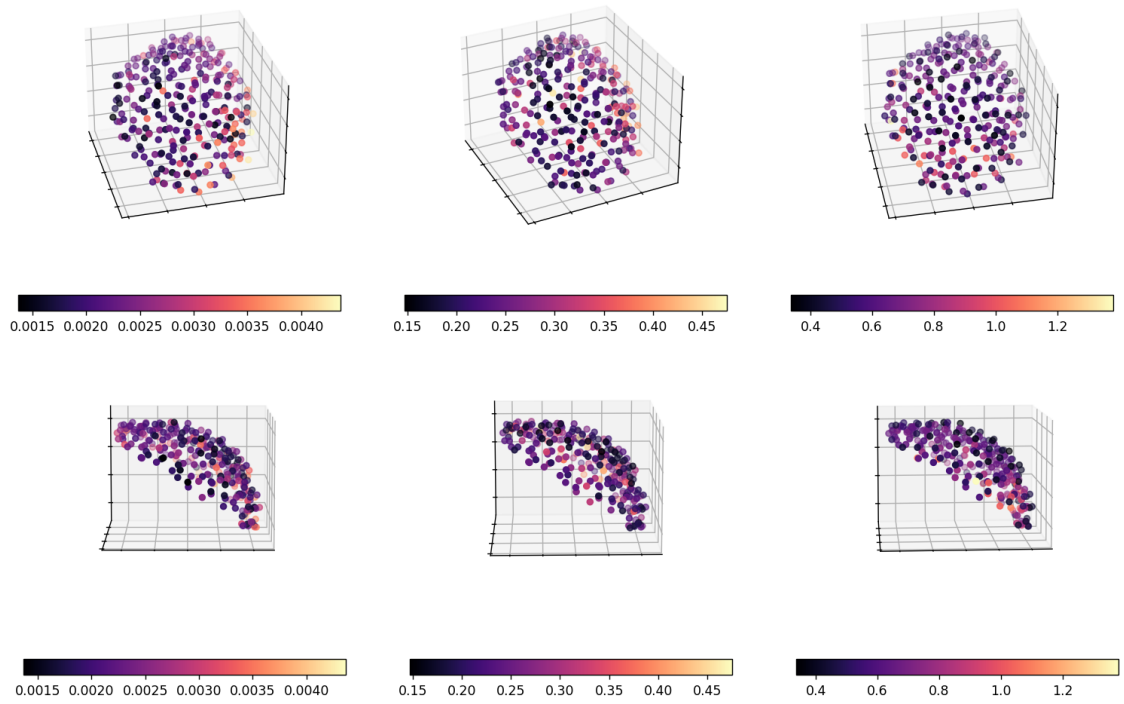


Figura H.2: Gráfico tridimensional que representa el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para el conjunto de geometrías no vistas (B, C y D) de las redes guiadas por datos. A la izquierda para posición, en el centro para velocidad y a la derecha para tensiones.

H.2. Redes híbridas

La segunda parte aborda la evaluación de la robustez de las redes híbridas frente a los mismos casos de carga y geometrías evaluados en las redes informadas por datos.

Modelo General

Base de datos	γ_{σ}	γ_q	$\gamma_{\dot{q}}$	Vista
Geometría A	0.252	$1,06 \times 10^{-3}$	$7,40 \times 10^{-2}$	Si
Geometría B	0.428	$3,08 \times 10^{-3}$	$2,80 \times 10^{-1}$	No
Geometría C	0.652	$2,40 \times 10^{-3}$	$2,11 \times 10^{-1}$	No
Geometría D	0.573	$2,99 \times 10^{-3}$	$3,11 \times 10^{-1}$	No

Tabla H.3: Resultado en inferencia para los diferentes modelos propuestos, pero únicamente entrenando con datos pertenecientes a la geometría original como indica la columna *Vista*.

Representando con la misma escala los gráficos tridimensionales así como el gráfico de caja (Figura H.4 y Figura H.3) de forma de comparativa.

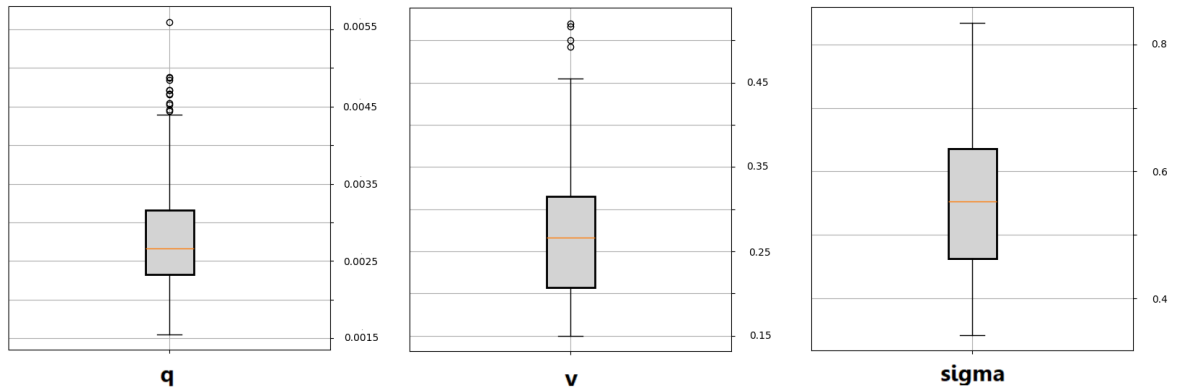


Figura H.3: Gráfico de caja del error relativo medio de cada simulación testeada para el conjunto de geometrías no vistas (B, C y D) de las redes híbridas. A la izquierda para posición, en el centro para velocidad y a la derecha para tensiones.

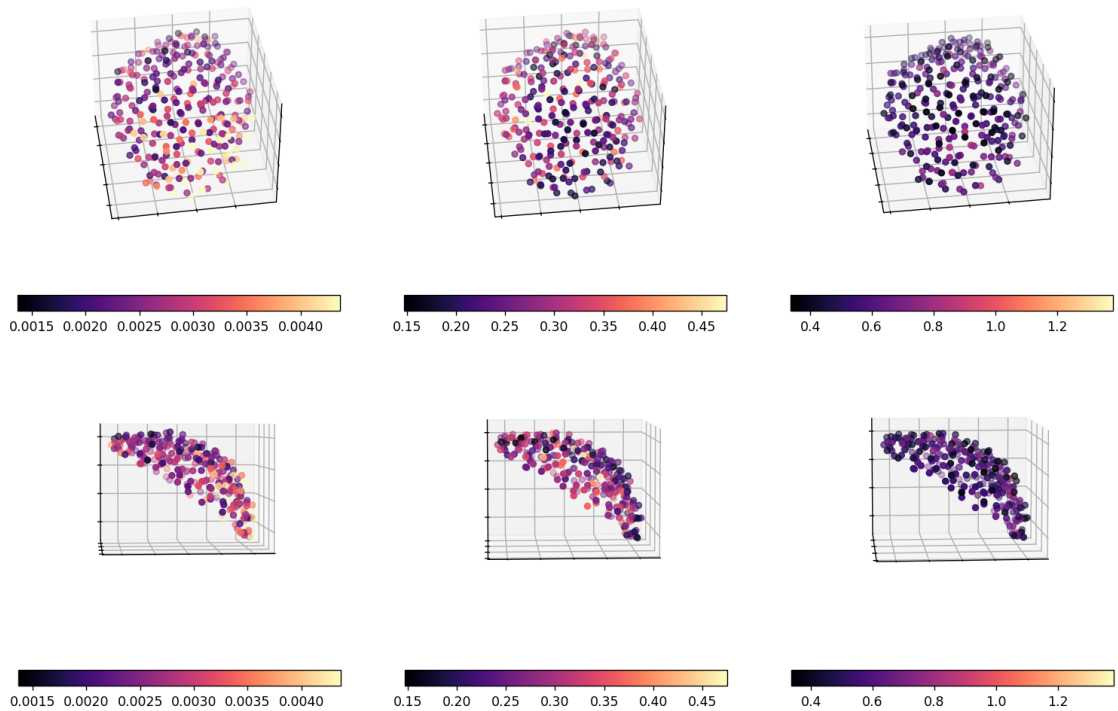


Figura H.4: Gráfico tridimensional que representa el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para el conjunto de geometrías no vistas (B, C y D) de las redes híbridas. A la izquierda para posición, en el centro para velocidad y a la derecha para tensiones.

Al igual que para las redes guiadas por datos, la inferencia es mejor en la región más céntrica. No obstante, si comparamos la precisión en tensiones, la metodología híbrida generaliza mejor para variaciones de geometría y volumen, a cambio de un mayor ruido en la predicción de posición y velocidad.

Anexos I

Arquitecturas recurrentes

I.1. Contexto y metodología

En diciembre del 2023, durante el desarrollo de este proyecto se publicó un artículo desde Google que remarcaba influencias positivas de incluir historia durante la codificación de los datos de entrada [36]. Asimismo este concepto es implementado en el artículo de DeepMind para la integración de modelos con mayor cantidad de pseudo-pasos, de tal forma que no se perdiera percepción del sistema inicial [10]. Remarcando, en ese caso, que esta metodología podría acelerar el sobre-ajuste de los modelos y en consecuencia empeorando su inferencia,

$$\begin{aligned}\varepsilon_{h_0} : \mathbb{R}^{F_e} &\rightarrow \mathbb{R}^{F_h} \\ \mathbf{h}_0 &\longrightarrow \mathbf{h}'_0\end{aligned}\tag{I.1}$$

Debido a este motivo se plantea una modificación sobre las redes de grafos informadas por física propuestas que añade cierta recurrencia al sistema. Este planteamiento se basa en re-circular la evolución temporal del paso anterior ($\dot{z}$), para incluirla como entrada (equivalentemente a la fuerza exterior) durante el procesamiento de la red de grafos (Figura I.1).

Se denota la historia de la red como h_0 para posteriormente codificarse al mismo espacio latente que las demás variables: $h'_0 \in \mathbb{R}^{F_h}$ (Ecuación I.1).

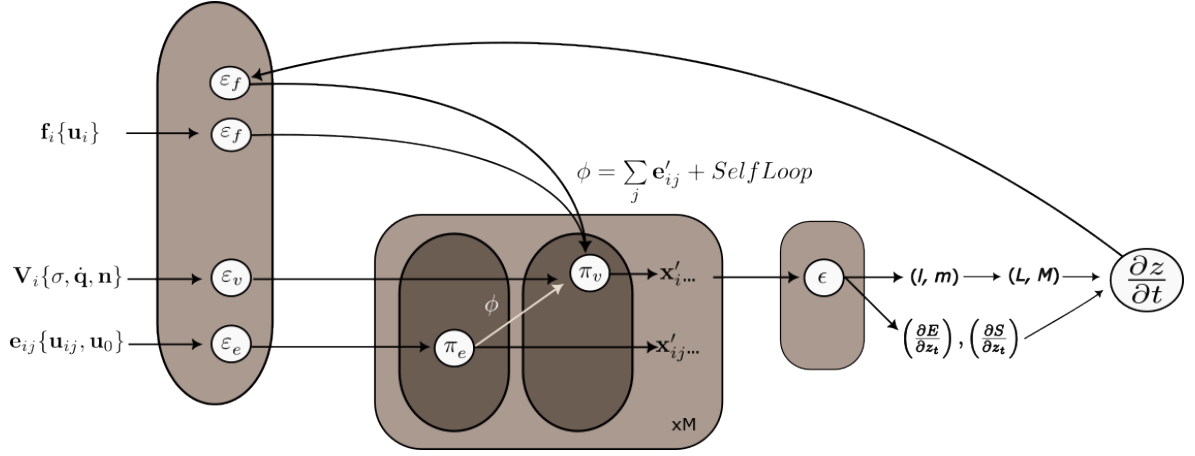


Figura I.1: Diagrama de bloques de una TIGNNs con recurrencia codificada y agregada en el procesador de la red de grafos. En el caso de ser el primer paso pseudo-temporal el vector recurrente sera nulo.

De tal forma, cuando la MLP (π_v) procesa la información nodal, agrega adicionalmente como entrada la componente de la memoria en espacio latente

$$\pi_v : \mathbb{R}^{3F_h + F_f + F_{h_0}} \rightarrow \mathbb{R}^{F_h}$$

$$F_h(\mathbf{x}_i, \phi(\mathbf{x}'_{ij}), \mathbf{f}'_i, \mathbf{h}'_0) \rightarrow \mathbf{x}'_i$$

(I.2)

I.2. Resultados

En un primer momento, se llevaron a cabo diversas pruebas para validar la capacidad predictiva del modelo con hiperparámetros conocidos. Sin embargo, la mayoría de los modelos probados mostraron escasa convergencia y falta de robustez. En la Tabla I.1, se detallan las métricas obtenidas en estas pruebas, las cuales presentaron resultados inferiores a los previamente reportados con otros modelos.

Ensayo	Tiempo	γ_σ	γ_q	$\gamma_{\dot{q}}$	MSE train	MSE val
Sin Ruido	7h 16'	7.77e-01	6.10e-03	4.37e-01	7.74e-04	5.22e-02
Ruido Alto	7h 20'	1.02e+00	5.08e-03	4.66e-01	2.73e-02	6.68e-02
Ruido Bajo	8h 5'	7.68e-01	7.61e-03	4.46e-01	1.11e-02	4.63e-02

Tabla I.1: Metricas de las TIGNNs con recurrencia entrenadas para 2000 épocas, 150 neuronas en capas ocultas y 10 pasos. La tasa de aprendizaje y λ_d fijadas semejantemente a los modelos presentados en la memoria. El ruido es gaussiano y controlado por una constante que escala su distribución.

Es destacable que también fueron validados pequeños cambios en la circulación de la recurrencia: tanto como entrada en el codificador, como en el decodificador o incluso agregándola al espacio latente del nodo por medio de una suma. Obteniendo para todos estos modelos resultados divergentes y descartables. Por lo que debido a estas bajas expectativas no se decide dedicar recursos a un barrido de hiperparámetros más completo.

I.3. Conclusiones

En base a los resultados expuestos en la Tabla I.1 se observa que el coste (MSE) en validación es realmente bajo, estableciendo una hipótesis de sobre-ajuste durante el entrenamiento. En esta suposición, la red ha aprendido a predecir realmente bien el siguiente pseudo-tiempo basandose en los dos anteriores, no obsante a medida que se acumula error durante el rollout, la robustez de ve más comprometida al ser función de dos entradas con ruido.

Estas conjeturas se respaldan con las métricas mostradas en las Figura I.2 y I.3. Donde la rápida bajada del coste en entrenamiento genera a posteriori una divergencia y separación del coste durante la validación, mientras que las métricas respectivas del rollout divergen equitativamente en el mismo punto de ruptura.

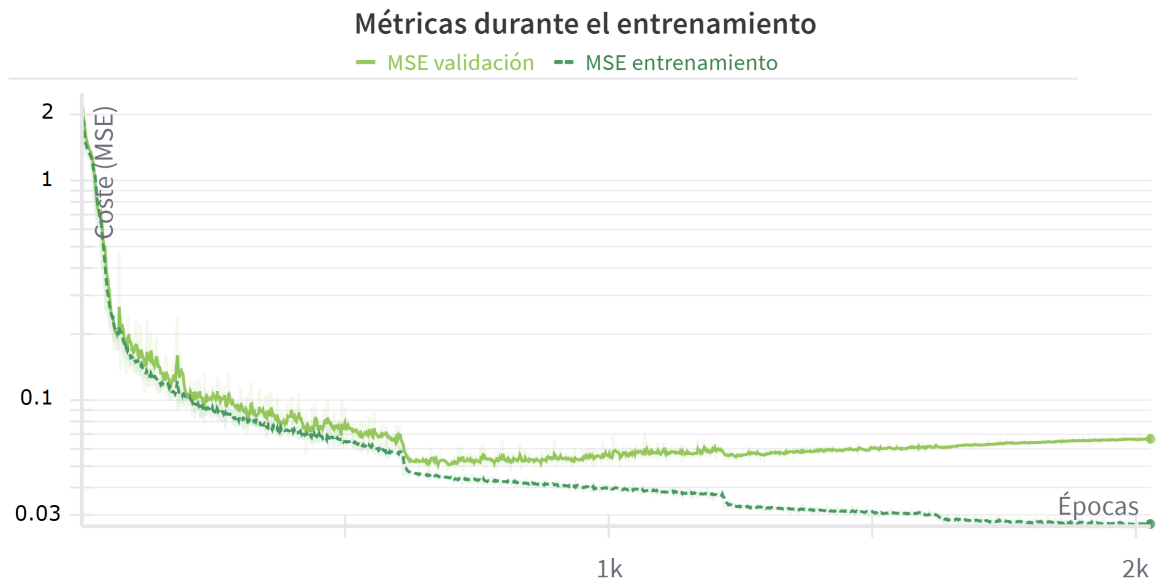


Figura I.2: Métrica representativa del error cuadrático medio durante el entrenamiento de la red recurrente, en el cual se aprecia el sobre-ajuste y el empeoramiento del error en validación durante el avance del entreno.

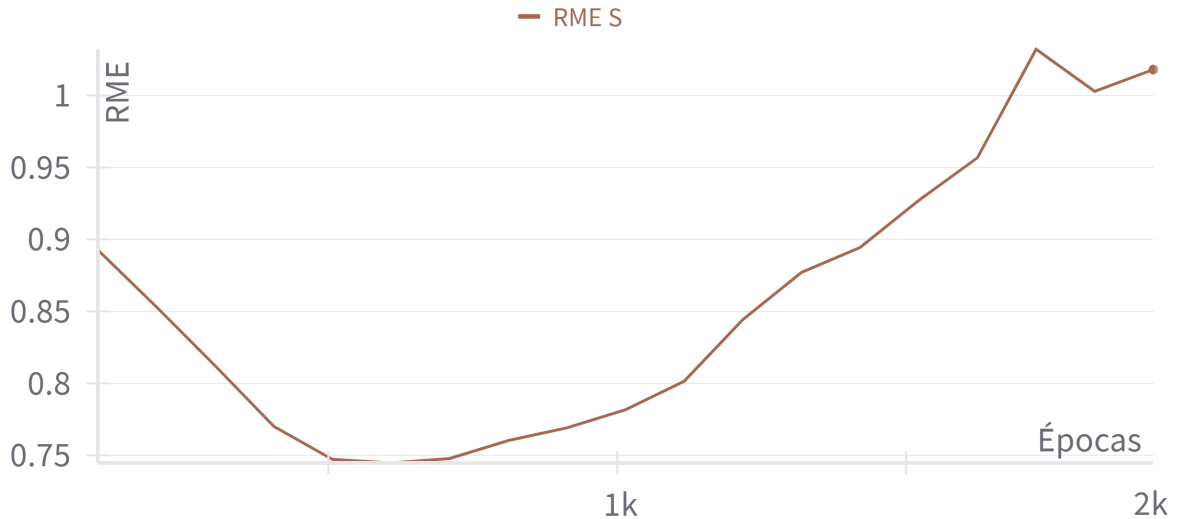


Figura I.3: Métrica representativa del error relativo medio en tensiones para el “rollout” de validación durante el entrenamiento de la red recurrente, con una clara divergencia al empeorar también durante el avance del entreno.

Asimismo todas las representaciones tridimensionales resultan pobres y poco realistas con este planteamiento, lo que en su conjunto concluye que al menos para la metodología propuesta, no se obtiene una mejora respecto a las arquitecturas anteriores.

Anexos J

Velocidad de respuesta: resultados y comparativas

Otro de los objetivos principales de estos algoritmos es optimizar la velocidad de cálculo en comparación con los métodos numéricos convencionales. Para ello, se comienza estimando el tiempo necesario para calcular alguno de modelos de elementos finitos utilizados (Tabla J.1).

Nodos	Tiempo Total (seg)	Frecuencia (Hz)
530	17.2	1.16
1395	21.32	0.93

Tabla J.1: Tiempos de cálculo en ABAQUS CAE para distintos mallados de una misma anatomía hepática. El procesador utilizado es un Inter Core i7-8700K

Asimismo, las arquitecturas analizadas han mostrado significativas mejoras en velocidad y rendimiento, incluso sin pulir la eficiencia computacional de los algoritmos (Tablas J.2). Lo que sugiere que aun hay margen para optimizar considerablemente los tiempos de cálculo para todos los modelos.

Nodos	Modelo	Parametros	Frame (seg)	Rollout (seg)	Frecuencia (Hz)
530	Guiado por datos	11mill	0.015	0.3	66.66666667
530	Guiado por datos	3mill	0.012	0.24	83.33333333
1395	Guiado por datos	11mill	0.0135	0.27	74.07407407
530	Híbrido	3mill	0.0044	0.088	227.2727273
530	Híbrido	2mill	0.0046	0.092	217.3913043
530	Híbrido	11mill	0.0088	0.176	113.6363636
1395	Híbrido	3mill	0.011	0.22	90.90909091
1395	Híbrido	11mill	0.022	0.44	45.45454545

Tabla J.2: Tiempos de respuesta en diferentes arquitecturas con una RTX 3090, para dos de las mallas utilizadas en el proyecto.

Anexos K

Valoración de robustez ante mallas densas

K.1. Contexto y resultados

El objetivo de este anexo es analizar la sensibilidad de la red frente a una mayor densidad de elementos, dado que las mallas empleadas en el proyecto tienen un promedio entre 400 y 600 nodos.

El nuevo mallado constará de 1395 nodos, siendo su densidad casi tres veces superior a la utilizada para los otros modelos. Manteniendo la proporción de casos de entrenamiento en un 80 % y aumentando el número de simulaciones totales a 340.

Modelo	Parámetros	γ_σ	γ_q	$\gamma_{\dot{q}}$
Redes Guiadas por Datos	$2,5 \cdot 10^6$	0.25	0.08	0.00083
Redes Híbridas	$3 \cdot 10^6$	0.3025	0.0903	0.00084

Tabla K.1: Resultado en inferencia para los diferentes modelos propuestos con el mallado denso.

Tras entrenar las dos arquitecturas definidas para este proyecto, se confirma que incluso frente a modelos más complejos, con mayores tiempos de procesamiento y mayor carga computacional, la predicción se mantiene sólida para los tres conjuntos de variables de estado.

Los modelos basados en datos muestran en la Tabla K.1 ligeramente mejores resultados, lo cual tiene una explicación lógica: al contar con más casos similares debido a la mayor cantidad de datos de carga semejantes, es posible inferir resultados más consistentes con experiencias previas.

No obstante, para este anexo no se han desarrollado barridos extensos, por lo que es probable que ambas metodologías presenten un rendimiento mas similar entre ellas. Validando que, para ambas, es factible utilizar redes de mayor resolución.

Sin embargo, el objetivo de este proyecto no es incrementar la densidad de manera indefinida para explorar los límites de ambas arquitecturas, ya que esto podría ir en contra del mecanismo de transmisión de mensaje establecido y aumentar exponencialmente el coste computacional del sistema. En su lugar, se busca obtener buenos resultados con una menor cantidad de datos, aprovechando los sesgos para ser mas eficiente durante los entrenamientos.

K.2. Representaciones gráficas

Finalmente se exponen tanto los gráficos de caja (Figura K.1) de la mejor arquitectura resultante, siendo la guiada por datos con tres millones de parámetros, junto con una comparativa tridimensional de la inferencia (Figura K.2) en una carga no vista y de su error relativo nodal para la misma simulación(Figura K.3).

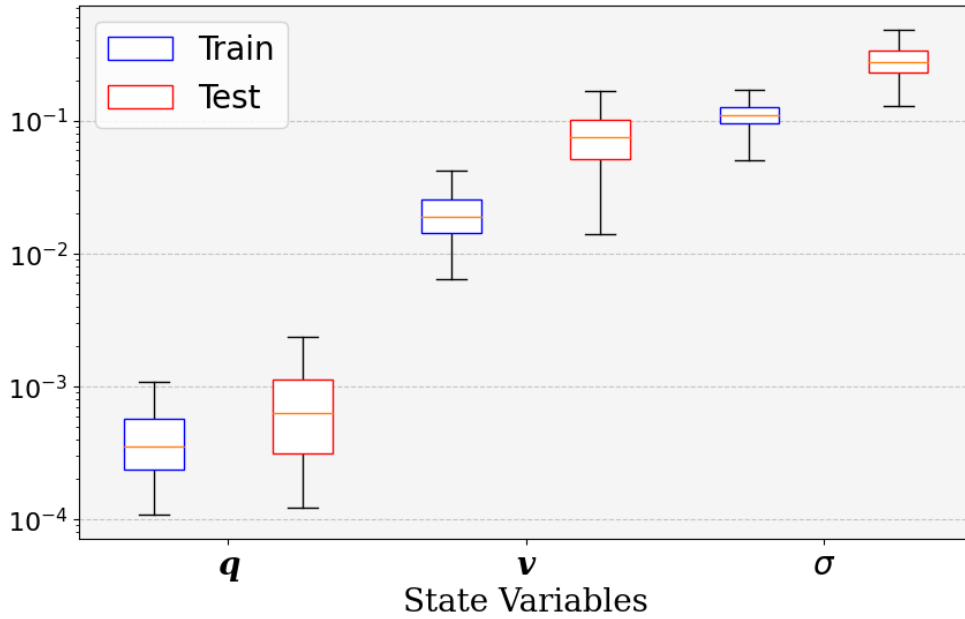


Figura K.1: Gráfico de caja que compara el error relativo en inferencia entre los conjuntos de entrenamiento y test. Concretamente con redes guiadas por datos y mallas de alta densidad.

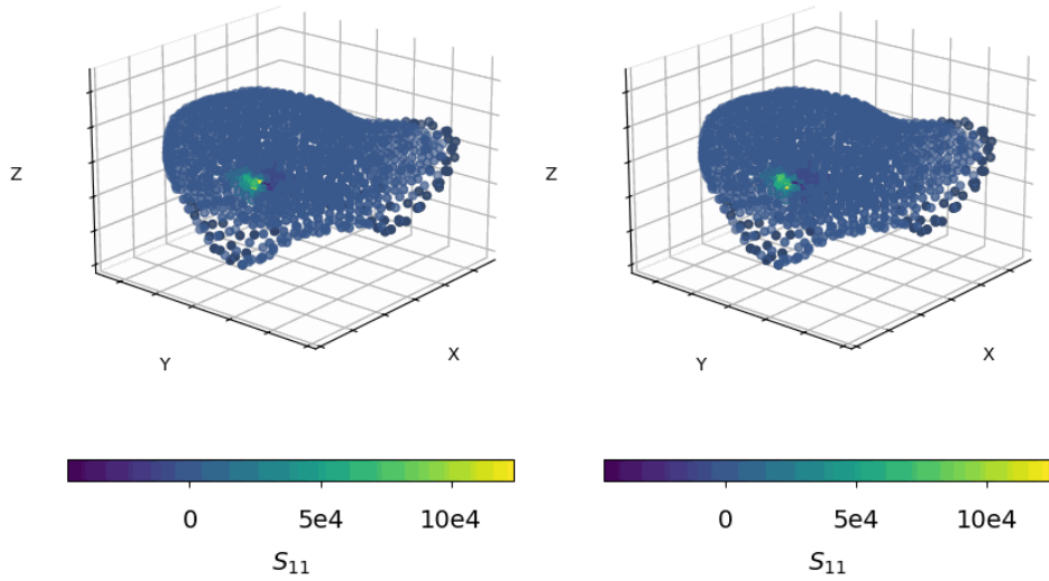


Figura K.2: Gráfico tridimensional que compara la inferencia de una geometría y carga no vista en un entreno (izquierda) con su solución numérica respectiva (derecha) para una malla de alta densidad en una arquitectura guiada por datos. S_{11} es la variable representativa en ABAQUS de σ_{11} .

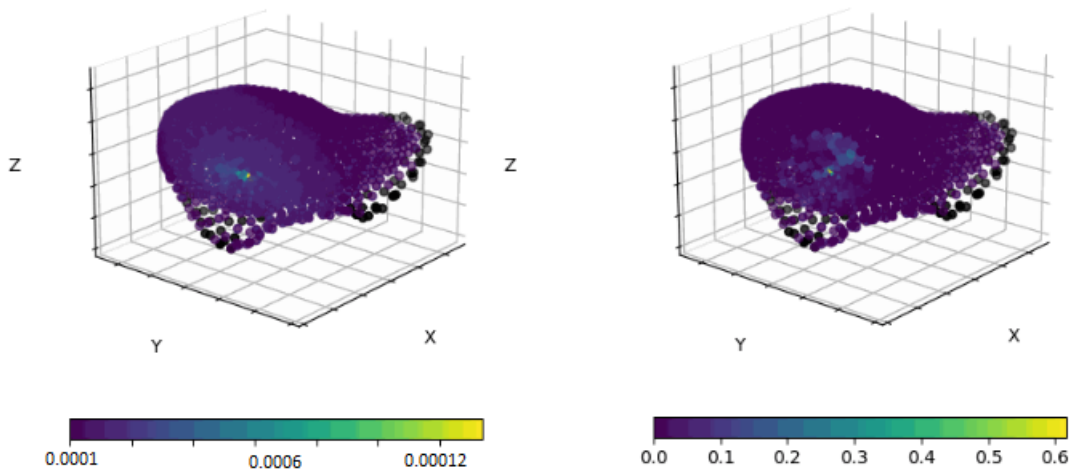


Figura K.3: Gráfico tridimensional que representa el error relativo de cada nodo para una geometría y situación de carga no vista durante el entreno para mallas de alta densidad. A la izquierda se representa el error relativo para posiciones y a la derecha para tensiones.

Anexos L

Robustez adquirida desde el entrenamiento con bases de datos mixtas

Como se ha demostrado, los modelos basados en grafos tienen cierta capacidad de generalización con geometrías distintas incluso sin que esta robustez sea implícita de sus datos de entrenamiento.

¿Pero que ocurre si se incrementa dicha robustez durante el entrenamiento? Es debida a esta pregunta que se ha creado una base de datos mixta utilizando particiones de tres geometrías diferentes (A, B y C), resultando en un total de 281 simulaciones, de las cuales el 80 % se destinará al entrenamiento. Por otro lado, la evaluación de la inferencia se realizará de dos formas: utilizando las métricas establecidas a lo largo del proyecto en el 20 % restante y, además, empleando una nueva geometría no vista para evaluar su capacidad de extrapolación.

L.1. Resultados y métricas

En primer punto, se valora la predicción sobre el conjunto de testeo en la propia base de datos mixta. La Tabla L.1, muestra los resultados en error relativo, obteniendo métricas semejantes a los modelos entrenados con una geometría. No obstante, el tamaño empleado es de un millón de parámetros entrenables. Al comparar este modelo con otros más grandes, no se observa una diferencia significativa en su desempeño ni en su capacidad de generalización. Por ello, se ha optado por modelos más ligeros y rápidos.

Base de datos	parámetros	γ_σ	γ_q	$\gamma_{\dot{q}}$
Mixto - Guiada por datos	10^6	0.493	$8,88 \times 10^{-4}$	$1,07 \times 10^{-1}$
Mixto - Híbrido	10^6	0.359	$2,26 \times 10^{-3}$	$1,18 \times 10^{-1}$

Tabla L.1: Resultado en inferencia para las bases de datos mixtas sobre el conjunto de validación en geometrías vistas.

Si por otra parte, evaluamos su rendimiento en una geometría no vista (Geometría D), obtenemos las métricas expuestas en la Tabla L.2.

Base de datos	γ_σ	γ_q	$\gamma_{\dot{q}}$
Singular - Guiada por datos	0.658	$1,97 \times 10^{-3}$	$2,06 \times 10^{-1}$
Singular - Híbrido	0.652	$2,40 \times 10^{-3}$	$2,11 \times 10^{-1}$
Mixto - Guiada por datos	0.616	$2,02 \times 10^{-3}$	$2,17 \times 10^{-1}$
Mixto - Híbrido	0.469	$2,67 \times 10^{-3}$	$1,74 \times 10^{-1}$

Tabla L.2: Resultado en inferencia para los diferentes modelos propuestos en la geometría D, siendo para todos ellos no vista durante el entrenamiento.

Obteniendo la mejor robustez para el planteamiento híbrido tanto como para tensiones como para velocidades, manteniéndose semejantes para las posiciones.

L.2. Comparativas tridimensionales en inferencia

De forma adicional a la métrica relativa, es relevante visualizar el error por simulación y punto de carga. Si en primer punto, comparamos los modelos guiados por datos, obtenemos que el añadir diferentes geometrías, mejora el error en tensiones, pero añade ruido en posición como puede ser lógico (Figura L.1). Sin embargo para los modelos híbridos (Figura L.2) es notable esta mejora, evolucionando de manera destacable en tensiones y velocidad.

Asimismo se plantea una comparativa entre la inferencia para la Geometría D en los modelos guiados por datos y los modelos híbridos, ambos entrenados con la misma base de datos mixta descrita en el inicio del anexo. Validando en la Figura L.3 que la robustez del modelo guiado por física es superior para dos de las tres variables de estado, manteniéndose la robustez intrínseca de las metodología guiada por datos para posición.

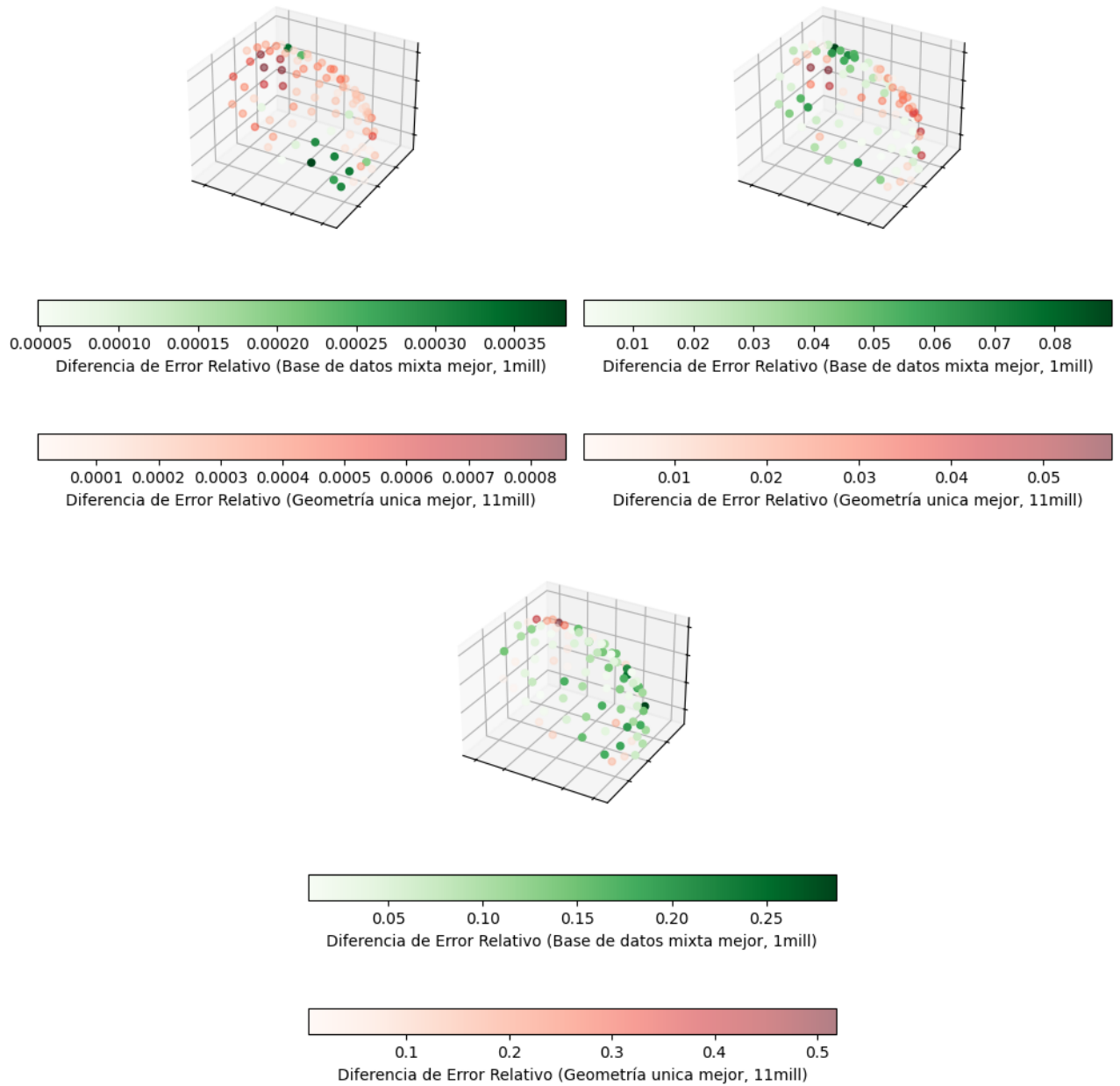


Figura L.1: Comparativa tridimensional que representa la diferencia entre el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para la geometría no vista (D) de las redes guiadas únicamente por datos entrenadas con diferentes bases de datos. A la izquierda para posición, en el derecha para velocidad y en la parte inferior para tensiones, siendo en verde los modelos con mejor inferencia para bases de datos con geometrías mixtas, mientras que en rojo tienen un mejor rendimiento los modelos con geometrías únicas.

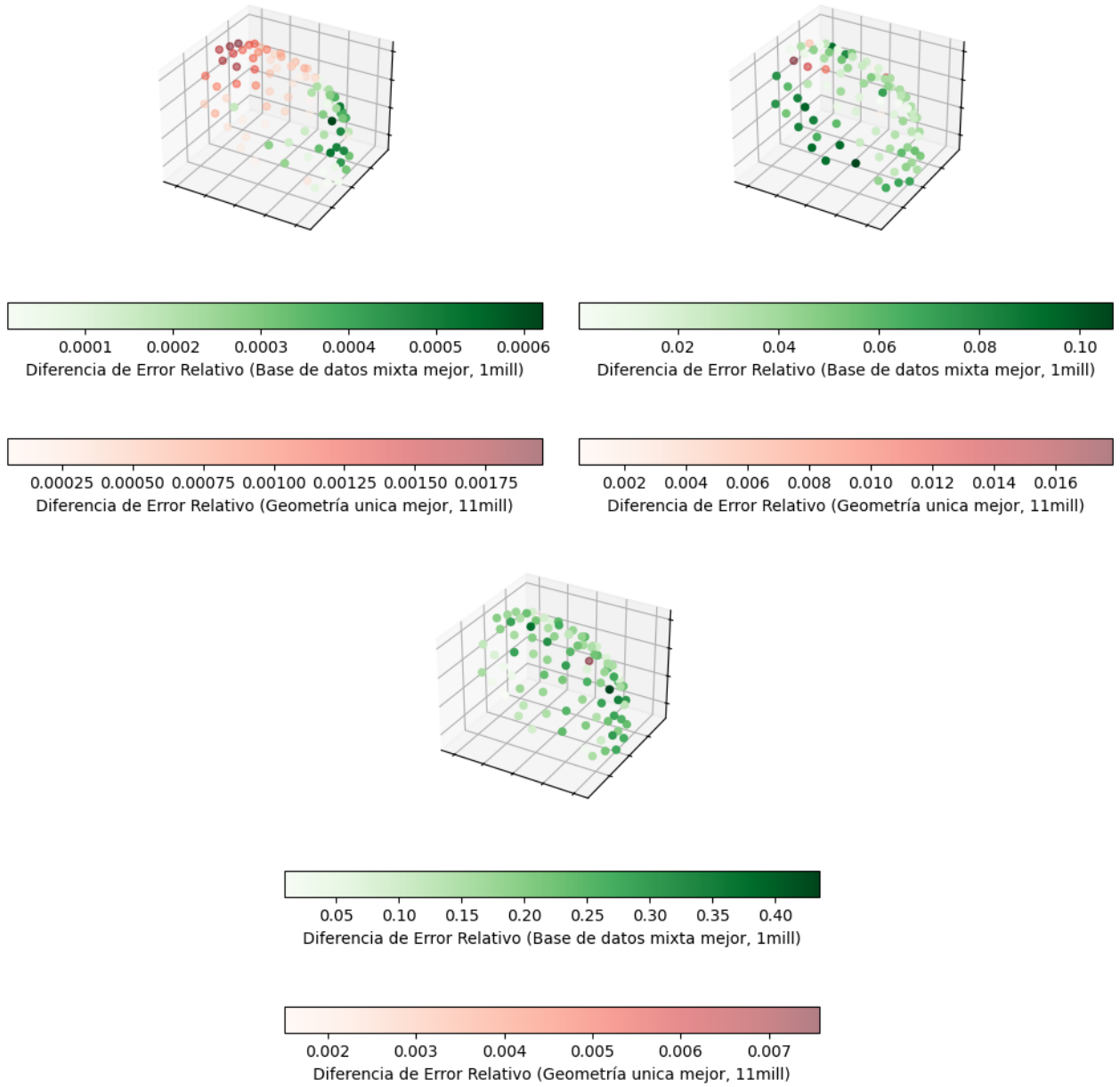


Figura L.2: Comparativa tridimensional que representa la diferencia entre el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para la geometría no vista (D) de las redes híbridas entrenadas con diferentes bases de datos. A la izquierda para posición, en el derecha para velocidad y en la parte inferior para tensiones, siendo en verde los modelos con mejor inferencia para bases de datos con geometrías mixtas, mientras que en rojo tienen un mejor rendimiento los modelos con geometrías únicas.

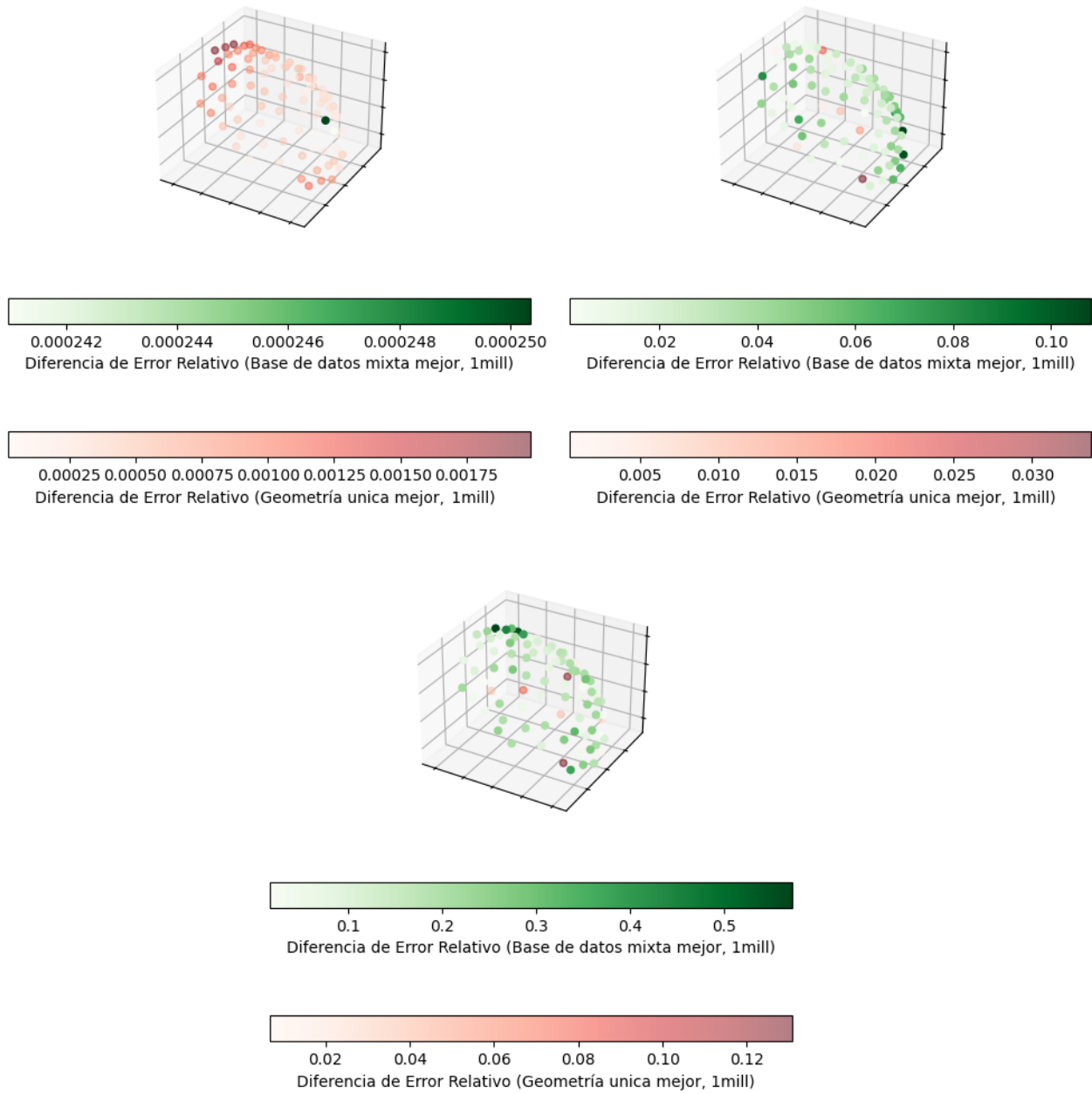


Figura L.3: Comparativa tridimensional que representa la diferencia entre el error relativo medio de cada simulación dependiendo de su punto de aplicación de la carga para la geometría no vista (D) de las redes híbridas y guiadas por datos entrenadas con bases de datos de geometrías mixtas. A la izquierda para posición, en el derecha para velocidad y en la parte inferior para tensiones, siendo en verde los modelos con mejor inferencia para bases de datos con geometrías mixtas, mientras que en rojo tienen un mejor rendimiento los modelos con geometrías únicas.

Anexos M

Especificaciones técnicas

El desarrollo del proyecto se ha llevado a cabo empleando las siguientes tecnologías y herramientas:

M.1. Entorno de desarrollo

- Python: La versión utilizada es Python 3.11.5 de forma local.
- Pytorch: Se ha empleado la versión 2.0.1 y 2.1.1 de Pytorch [35].

M.2. Aceleración de cálculos

- GPU NVIDIA GeForce RTX 2070: Se ha aprovechado esta tarjeta gráfica para optimizar el rendimiento durante el entrenamiento mediante la paralelización de cálculos de forma local.
- NVIDIA GeForce RTX 3090: se han utilizado las NVIDIA GeForce RTX 3090 disponibles en el superordenador del I3A.
- NVIDIA GeForce A100: también se ha entrenado con las NVIDIA GeForce A100 disponibles en el superordenador del I3A.
- CUDA (Compute Unified Device Architecture): Se ha empleado esta plataforma desarrollada por NVIDIA para la coordinación de los entrenamientos, facilitando la integración de los dispositivos GPU con Pytorch.

M.3. Computación en la nube

- Condor: Se ha utilizado la versión 9.0.9 de Condor para la computación distribuida en la nube, junto con Python 3.9 y sus respectivos paquetes.

M.4. Modelado geométrico

- Blender: La versión 3.6 de Blender como programa de edición de geometría 3D.
- MeshLab: Se ha empleado MeshLab en su versión 2022.02, como herramienta de procesamiento de mallas 3D.
- Gmesh: La versión utilizada es Gmesh 4.11.1, un software de código abierto especializado en la generación de mallas para simulaciones numéricas.

M.5. Modelado de elementos finitos

- ABAQUS CAE: Se ha utilizado la versión 6.16 para el modelado de elementos finitos.

Adicionalmente, todas las arquitecturas pertenecientes a este proyecto están disponibles en GitHub.