



Universidad
Zaragoza

Trabajo Fin de Grado

Título del trabajo:

Redes bayesianas informadas por la física aplicadas a
sistemas estructurales

English tittle:

Physically-informed neural networks applied to structural
systems

Autor/es

Pablo Falcón Gutiérrez

Director/es

Beatriz García Moya
Elías Cueto Prendes

Ingeniería mecánica

Universidad de Zaragoza
Escuela de ingeniería y arquitectura
2022-2023



Resumen

El reciente peso dado en la inteligencia artificial a las redes neuronales bayesianas se explica por su gran aportación a la hora de capturar la incertidumbre de las predicciones. Este factor resulta especialmente útil en situaciones donde es importante entender la confianza en las predicciones, situaciones críticas donde se requiere fiabilidad y seguridad.

En el presente trabajo fin de grado se realizará la implementación de redes neuronales bayesianas informadas por la física. Para ello se parte de un modelo de red neuronal de tipo SPNN (*Structure-Preserving Neural Networks*), que es una red neuronal en la que se le implementa el formalismo GENERIC por el fin de cumplir por las leyes de la termodinámica. En el trabajo se estudian sus diferentes prestaciones ante un mismo problema: el péndulo termo-elástico.

Finalmente, se realiza la implementación de ambas tipologías de redes en un problema de monitorización estructural donde la herramienta de inteligencia artificial deberá ser capaz de detectar, localizar y dar información sobre un posible fallo.

Para ello se generan datos pseudo-experimentales mediante el software ABAQUS simulando la existencia de sensores repartidos por todo el dominio de una placa. La herramienta de inteligencia artificial consta inicialmente de un gemelo digital que aprende la física de la placa sin fallos mecánicos. Posteriormente se complementa con otra red para que así aprenda las desviaciones entre el gemelo digital y los datos recogidos por los sensores. Al modelo completo se le denomina gemelo híbrido.

Los resultados pondrán de manifiesto no solo la precisión de las redes convencionales y bayesianas en ambos casos, sino su idoneidad en función de la incertidumbre.



Contenido

| | | |
|-------|---|----|
| 1 | Introducción | 4 |
| 1.1 | Metodología | 4 |
| 1.1.1 | ¿Por qué un modelo estocástico? | 4 |
| 1.1.2 | ¿Por qué se utilizan redes informadas por la física? | 7 |
| 1.2 | GENERIC ^[1] | 8 |
| 1.2.1 | Motivación | 8 |
| 1.2.2 | Bases del formalismo | 8 |
| 1.2.3 | Algoritmo de integración ^[1] | 9 |
| 1.3 | Objetivo | 11 |
| 1.4 | Contenido del proyecto | 11 |
| 2 | Implementación | 12 |
| 2.1 | Base de datos | 12 |
| 2.2 | Entrenamiento | 13 |
| 2.3 | Variaciones de la SPNN | 15 |
| 2.3.1 | Red neuronal bayesiana | 16 |
| 2.4 | Comparación de estructuras | 17 |
| 2.4.1 | Caso base: aprendizaje de la dinámica del péndulo doble | 18 |
| 2.4.2 | Base de datos reducida | 22 |
| 3 | Aplicación de redes neuronales a SHM | 25 |
| 3.1 | Metodología | 25 |
| 3.1.1 | Creación de la base de datos | 25 |
| 3.1.2 | Entrenamiento del modelo. Gemelo digital | 25 |
| 3.1.3 | Desarrollo del gemelo híbrido | 26 |
| 3.2 | Implementación: Placa apoyada-empotrada | 27 |
| 3.2.1 | Módulo de Young | 29 |
| 3.3 | Resultados | 30 |
| 3.3.1 | Red ANN | 30 |
| 3.3.2 | Red BNN | 36 |
| 3.3.3 | Comparación ANN vs BNN | 41 |
| 4 | Conclusiones | 43 |
| 4.1 | Futuras líneas de trabajo | 43 |
| 5 | Glosario de abreviaturas | 44 |
| 6 | Bibliografía | 45 |

1 Introducción

El estudio de la construcción no sólo debe estar orientado hacia el cálculo y la utilidad de las infraestructuras, sino también a garantizar la seguridad de su uso en el tiempo de vida de las mismas. Un fallo estructural no siempre es causado por un error de cálculo o una mala ejecución de la estructura, sino que puede haber muchas otras causas que no se pueden prever. Por ejemplo, fatiga, deterioro de los materiales, condiciones ambientales adversas...

El presente TFG tiene la finalidad de predecir la evolución del comportamiento de sistemas estructurales en el tiempo o para conocer su estado actual cuando no puede ser observado directamente. Esto se logrará mediante la aplicación de la Inteligencia Artificial (IA) para predecir estados futuros bajo acciones de cargas. Las redes estarán apoyadas por la física a través de la imposición del formalismo GENERIC, que garantiza el cumplimiento de las leyes de la termodinámica. Estas redes se aplicarán a sistemas estructurales, analizando la deformación de manera temporal, para así poder introducir nociones básicas de monitorización estructural o *Structural Health Monitoring* (SHM, por sus siglas en inglés).

Los datos temporales de los desplazamientos de algunos puntos distribuidos en una gran estructura pueden mostrar alteraciones, globales o locales, que denotan un daño en la misma. El SHM se basa en la recopilación de datos de una estructura a través de sensores. Con los datos recogidos se hace un análisis de éstos para poder detectar anomalías que predigan un posible daño en la estructura. Su principal función, por tanto, consiste en detectar los posibles fallos estructurales para prevenir su agravamiento mediante el mantenimiento para la minimización de los costes, la maximización de la seguridad y gestión del ciclo de vida de la estructura.

1.1 Metodología

Este proyecto propone una estrategia de SHM basada en un modelo de IA con enfoque estocástico y físico. Para ello se propone el uso de una red informada por la física. Específicamente, este proyecto se basa en las llamadas *Structure-Preserving Neural Networks* ^[1] (SPNN, por sus siglas en inglés).

1.1.1 ¿Por qué un modelo estocástico?

El uso de sensores conlleva la adquisición de datos en los que puede haber ruido. El procesamiento de datos con ruido es complejo, ya que puede inducir inestabilidad en las predicciones y errores graves. Existen métodos de limpieza de ruido en los datos, como los filtros. Otro enfoque consiste en utilizar modelos estocásticos, que pueden sacar conclusiones ante situaciones de incertidumbre, o variabilidad de los datos.

Un modelo estocástico es aquel que se basa en la aleatoriedad para poder predecir un comportamiento mediante el aprendizaje de las características estocásticas del problema. Este tipo de modelos es lo contrario a los modelos deterministas. En un modelo determinista, las predicciones resultantes ante una misma entrada de datos siempre será la misma. En cambio, para un modelo estocástico, las predicciones no serán iguales. Un entrenamiento correcto deberá ser robusto respecto a las entradas y salidas de la red.

El proyecto realiza un desarrollo a partir de redes bayesianas o *Bayesian Neural Network* (BNN por sus siglas en inglés). Éstas son parecidas a las redes neuronales artificiales o Artificial Neural Network (ANN por sus siglas en inglés). La principal diferencia reside en que las BNN (modelos estocásticos) no aprenden parámetros concretos como las ANN (Figura 1.1.a, modelos deterministas), sino que aprenden distribuciones de los parámetros de la red (Figura 1.1.b).

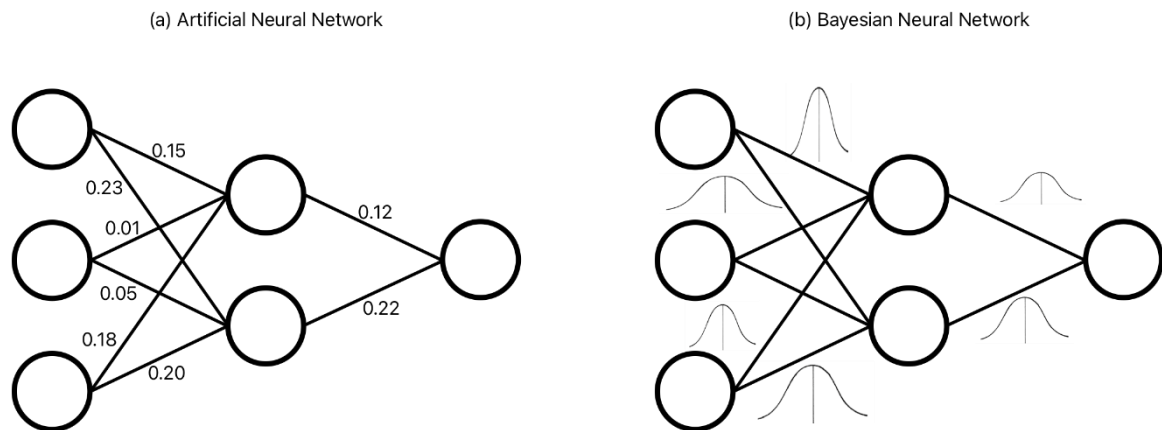


Figura 1.1 Parámetros de ANN vs BNN

Además, en las redes BNN se puede evaluar la incertidumbre de las predicciones. Es decir, se puede mostrar un intervalo de confianza de la predicción según los datos de entrenamiento utilizados.

En las siguientes figuras se muestra el ejemplo que se usa para validar las redes bayesianas. En la figura 1.2 se aprecia una red neuronal convencional y en la figura 1.3 se aprecia una red neuronal bayesiana. El modelo ANN realiza una predicción determinista. En cambio, en el modelo BNN se distingue una esperanza de los datos (línea azul oscura) y una incertidumbre (intervalo azul claro). Es notorio de destacar cómo en la zona que está más poblada de datos la incertidumbre del modelo disminuye, pero ésta aumenta conforme los datos disminuyen.

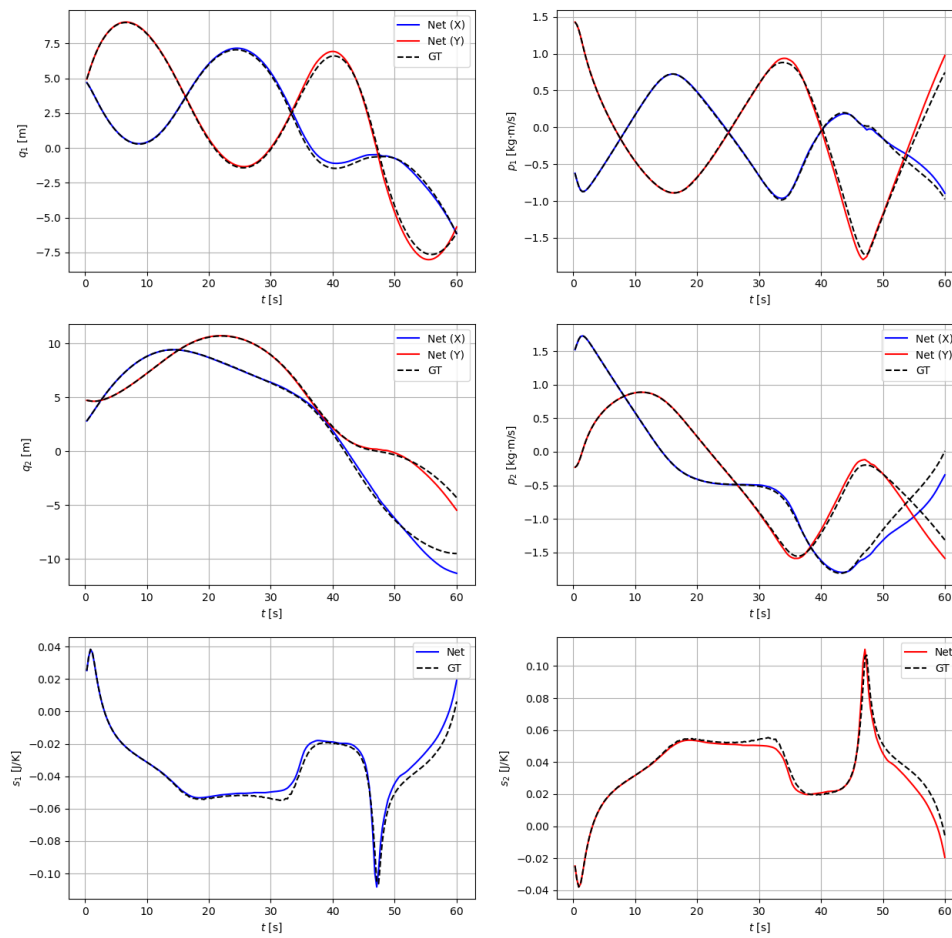


Figura 1.2 Redes neuronales convencionales.

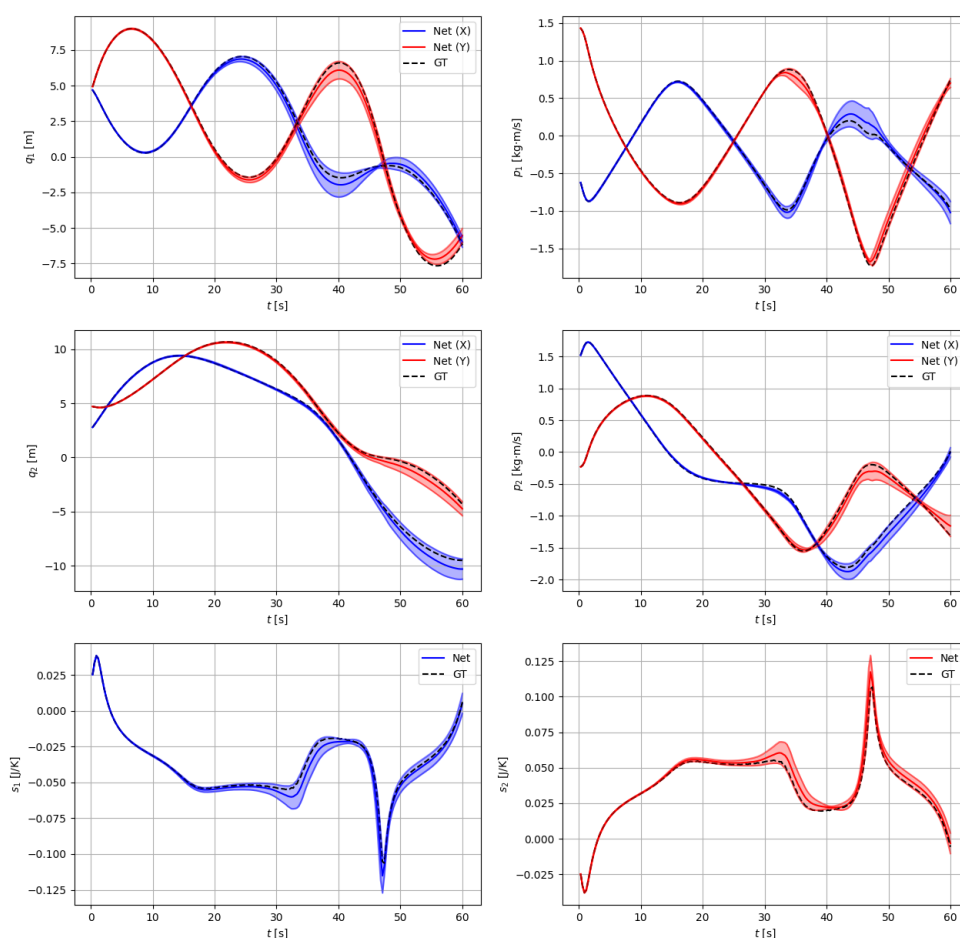


Figura 1.3 Redes neuronales bayesianas.

1.1.2 ¿Por qué se utilizan redes informadas por la física?

El formalismo GENERIC (*General Equation for Non-Equilibrium Reversible-Irreversible Coupling*)^[4] es la base teórica utilizada en el proyecto propuesto.

En general, las redes neuronales profundas son suficientes para la predicción de modelos físicos sin necesidad de la aportación de ecuaciones que cumplimenten las redes.

El problema es que las redes neuronales están limitadas por los datos de entrenamiento de la red. Aunque puedan ser capaces de predecir un modelo físico no aprenden la física que hay detrás de éste y con frecuencia proporcionan resultados inesperados cuando se evalúan en datos alejados de los empleados en el entrenamiento. Gracias a las redes informadas por la física sí que podemos asegurar la consistencia física. A través de estos cambios se puede asegurar que:

- Se aprende la física que hay detrás de los datos.
- Los resultados cumplirán con las restricciones impuestas.

Se necesitarán menos datos, lo cual es una ventaja en situaciones en las que no hay régimen de *big data* (por ejemplo, que sea caro hacer mediciones). Es lo que se llama el *smart data paradigm*, o uso inteligente de los datos. Las restricciones impuestas en el entrenamiento, sean físicas o de otra categoría, se llaman sesgos. Se aplican sesgos inductivos. Con GENERIC no es necesario conocer las ecuaciones concretas de cada modelo ya que se aprenden a partir de los datos por las características del formalismo. Además, GENERIC se puede aplicar a una gran variedad de problemas físicos ya que se puede usar para analizar casos no conservativos.

El formalismo GENERIC impone como condiciones el cumplimiento de la primera y segunda ley de la termodinámica. Es decir, se cumple con la:

- Conservación de la energía
- Desigualdad de la entropía

1.2 GENERIC ^[1]

General Equation for the Non-Equilibrium Reversible/Irreversible Coupling, o más conocido por sus siglas como GENERIC, es un formalismo creado por Prof. Miroslav Grmela y Prof. Hans Christian Öttinger en 1997^[3]. GENERIC se apoya en las leyes de la termodinámica y se utiliza para la descripción de sistemas dinámicos.

1.2.1 Motivación

El formalismo GENERIC es un sesgo inductivo que permite aplicar preceptos de la física sin conocer la ecuación de comportamiento del problema. Además, es válido tanto para problemas conservativos como disipativos.

1.2.2 Bases del formalismo

Se realiza una aproximación de sistemas dinámicos: la evolución del sistema se estudia a partir del análisis de la tasa de cambio de una selección de variables de estado impuestas para nuestro sistema.

$$(1.1) \quad \frac{d\mathbf{z}}{dt} = f(\mathbf{z}, t)$$

$$(1.2) \quad \dot{\mathbf{z}}_t = \mathbf{L} \frac{\partial E}{\partial \mathbf{z}} + \mathbf{M} \frac{\partial S}{\partial \mathbf{z}}$$

Siendo:

- $\dot{\mathbf{z}}_t$: Derivada de las variables de estado en el instante t. Las variables de estado en GENERIC son las variables físicas que describen el estado del sistema. En el

formalismo GENERIC, la selección de variables debe ser una que permita expresar la evolución de energía y entropía.

- **L**: Operador de Poisson en nomenclatura hamiltoniana. Es antisimétrico.
- **M**: Operador de fricción en nomenclatura hamiltoniana. Es simétrico y semidefinido-positivo.
- **E**: Energía del sistema
- **S**: Entropía del sistema

La ecuación (1.2) se divide en dos términos:

- Reversible: $\mathbf{L} \frac{\partial E}{\partial \mathbf{z}}$, representa a todos los fenómenos no disipativos.
- No reversible: $\mathbf{M} \frac{\partial S}{\partial \mathbf{z}}$, representa a todos los fenómenos disipativos.

Además, para completar el cumplimiento de la estructura GENERIC se deben de cumplir las condiciones de degeneración. Es decir, que el producto cruzado de los términos anteriores sea nulo.

$$(1.3) \quad \mathbf{L} \frac{\partial S}{\partial \mathbf{z}} = \mathbf{0}$$

$$(1.4) \quad \mathbf{M} \frac{\partial E}{\partial \mathbf{z}} = \mathbf{0}$$

1.2.3 Algoritmo de integración^[1]

Para la implementación del formalismo GENERIC en nuestra red neuronal se discretiza la ecuación (1.2):

$$(1.5) \quad \frac{\mathbf{z}_{n+1} - \mathbf{z}_n}{\Delta t} = \mathbf{L} \frac{DE}{D\mathbf{z}} + \mathbf{M} \frac{DS}{D\mathbf{z}}$$

La derivada temporal de la ecuación (1.5) se discretiza con un esquema *forward Euler* en incrementos de tiempo Δt donde $\mathbf{z}_{n+1} = \mathbf{z}_{n+\Delta t}$. Además, las matrices **L** y **M** aparecen en la versión discretizada. Por último, $\frac{DE}{D\mathbf{z}}$ y $\frac{DS}{D\mathbf{z}}$ representan los gradientes discretizados y se pueden aproximar como:

$$(1.6) \quad \frac{DE}{D\mathbf{z}} \approx \mathbf{A}\mathbf{z}$$

$$(1.7) \quad \frac{DS}{D\mathbf{z}} \approx \mathbf{B}\mathbf{z}$$

Donde A y B representan las matrices discretizadas de los operadores gradientes.



Por lo tanto, si se modifica la ecuación (1.5) y las ecuaciones de degeneración (1.6) y (1.7) el formalismo GENERIC quedará como:

$$(1.8) \quad \mathbf{z}_{n+1} = \mathbf{z}_n + \Delta t (\mathbf{L} \cdot \mathbf{A}\mathbf{z}_n + \mathbf{M} \cdot \mathbf{B}\mathbf{z}_n)$$

Sujetas a las condiciones de degeneración:

$$(1.9) \quad \mathbf{L} \cdot \mathbf{B}\mathbf{z}_n = \mathbf{0}$$

$$(1.10) \quad \mathbf{M} \cdot \mathbf{A}\mathbf{z}_n = \mathbf{0}$$

1.3 Objetivo

El objetivo es el análisis y comparación de redes neuronales bayesianas con redes neuronales convencionales, ambas guiadas por la física, en su aplicación a sistemas dinámicos. Como comprobación final, se aplicarán en un sistema de IA capaz de predecir los desplazamientos y tensiones (en el tiempo) de una estructura. Las dos redes neuronales estarán basadas en el formalismo GENERIC (*General Equation for the Nonequilibrium Reversible-Irreversible Coupling*).

Además, se compararán las diferentes ventajas y desventajas de usar un modelo más convencional como es el caso de las redes neuronales artificiales o uno más sofisticado con las redes neuronales bayesianas.

1.4 Contenido del proyecto

El proyecto consta de los siguientes hitos:

- Estudio de ANN + GENERIC. Péndulo doble termo-elástico ^[1] ^[2]: modelo creado y diseñado por investigadores de la Universidad de Zaragoza. Se basa en el estudio de un doble péndulo termo-elástico en el que a través de datos generados sintéticamente se acaba entrenando una red capaz de predecir el avance del péndulo dada una posición inicial.
- Implementación y comparación de BNN + GENERIC en Péndulo doble termo-elástico: Se implementa una red neuronal bayesiana en el ejemplo anterior para poder comparar los resultados y ver las ventajas y desventajas de usar un modelo estocástico frente a uno determinista.
Además, se hace otra comparación de los modelos añadiendo ruido a los datos para poder comparar su efecto en los modelos BNN y los ANN.
- Generación de datos sintéticos para problemas estructurales: Se crea un ejemplo de elementos finitos en el programa ABAQUS. Posteriormente de los informes generados y extraídos mediante un programa de *Python* y se vuelve a introducir ruido haciendo que parezcan sensores.
- Diseño del modelo ANN y BNN + GENERIC: Se adaptan las redes usadas anteriormente a los nuevos problemas estructurales y se vuelven a comparar las diferencias entre ambos modelos.
- Diseño del segundo modelo: Por último, se crea una segunda red neuronal para predecir la desviación entre la salida de la primera red y las nuevas mediciones (en un modelo con fallo), y así poder descubrir la localización del fallo en la estructura.

2 Implementación

En este capítulo se realiza la comparación entre las redes neuronales bayesianas y las SPNN en el caso dinámico del doble péndulo termo-elástico. Las redes SPNN ^[1] son una estructura neuronal basada en el formalismo GENERIC. En la figura 2.1 se observa la representación del doble péndulo termo-elástico.

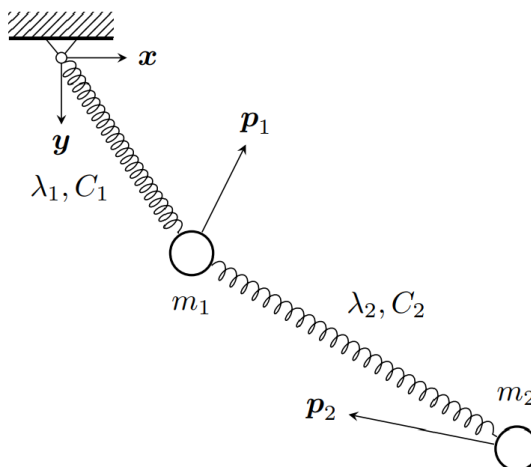


Figura 2.1. Esquema del doble péndulo termo-elástico.

2.1 Base de datos

Es de gran importancia disponer de una base de datos adecuada para que la red aprenda correctamente. En el caso del péndulo doble se parte de una base de datos que va a estar definida por un tensor de tres dimensiones:

$$\text{Dataset} = (\text{Batch size}; \text{Pasos de tiempo}; \text{Variables})$$

Siendo:

- *Batch size*: Número de simulaciones diferentes introducidas a la red neuronal (figura 2.2.C)
- Pasos de tiempo (figura 2.2.B)
- Variables: Número de variables de estado de cada instante (figura 2.2.A).

Para el ejemplo del péndulo doble en total suman 10 variables de estado:

- Posición en x
- Posición en y
- Velocidad en x
- Velocidad en y
- Entropía total del sistema
- Energía total del sistema

Tanto las variables de estado posición y velocidad están repetidas para los dos nodos del péndulo doble.

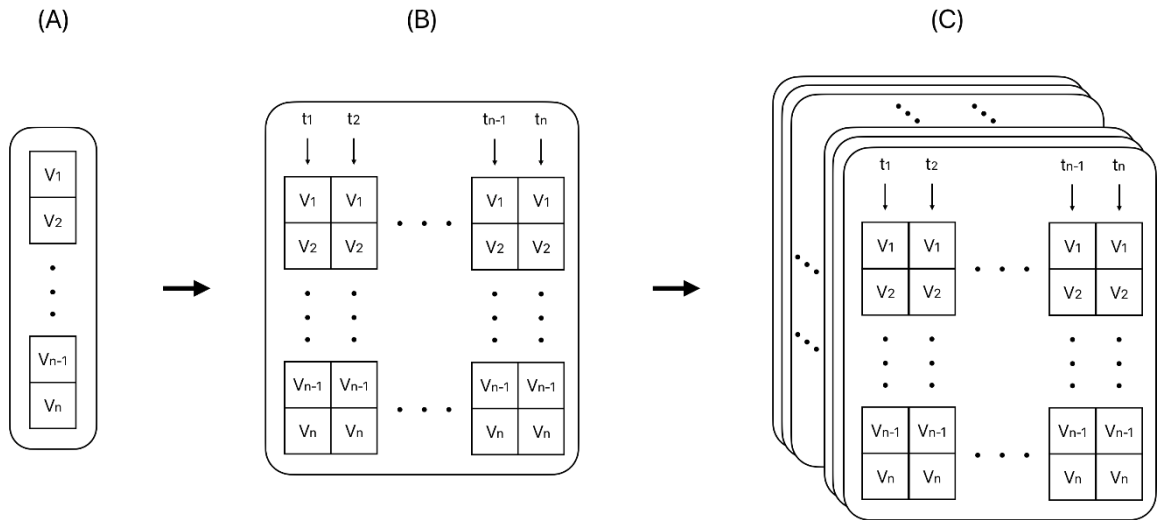


Figura 2.2. Esquema conceptual de la base de datos.

En aprendizaje automático, para poder entrenar la red, no se introduce la base de datos completa, sino que se divide una base de aprendizaje y otra de *test*. El propósito de dividirlo en dos partes tiene como objetivo asegurar que el modelo no sólo sea capaz de predecir los casos entrenados, sino que también sea capaz de predecir problemas que no haya visto previamente.

2.2 Entrenamiento

Haciendo alusión a la figura 2.3, se describe cada uno de los bloques esenciales del entrenamiento de la red.

- **Red:** Se realiza una propagación hacia delante de la red neuronal utilizada para predecir los operadores de GENERIC necesarios.
- **GENERIC:** Una vez predichos los operadores de GENERIC se realiza la integración temporal de la parte dinámica. Las matrices L y M suelen ser conocidas, pero pueden ser parte del entrenamiento para aquellos casos en los que no se conocen, como en modelos de orden reducido, o son complejas.

$$(2.1) \quad \mathbf{z}_{n+1} = \mathbf{z}_n + \Delta t (\mathbf{L} \cdot \mathbf{A} \mathbf{z}_n + \mathbf{M} \cdot \mathbf{B} \mathbf{z}_n)$$

- **Cálculo del error:** Se calcula el error entre la predicción generada por la red $\mathbf{z}_{n+1}(net)$ y el valor esperado recogido del *dataset* $\mathbf{z}_{n+1}(GT)$ (siendo GT las siglas de *Ground Truth*). Concretamente sería la suma de los errores cuadráticos de cada componente del tensor del valor predicho y el valor real. Su salida en la red es llamada como L_{data} , error de los datos.

$$(2.2) \quad MSE_i = \sum_{n=0}^{Nt} (\mathbf{z}_{n,i}^{net} - \mathbf{z}_{n,i}^{GT})^2$$

Siendo:

- MSE : error cuadrático
- n : número de step
- Nt : número total de steps
- i : componente individual de los tensores
- **Loss**: evaluación de todos los errores. Es la suma de estos tres términos:
 - L_{degen} : Error de las condiciones de degeneración.
 - L_{data} : Error de los datos. Aparece en la ecuación (2.2) como MSE.
 - L_{reg} : es un término que se añade a la pérdida global. Tiene como objetivo minimizar lo máximo posible los parámetros de la red ya que así se consigue que el modelo evite el sobreajuste y aprenda de forma general. Matemáticamente se expresa como:

(2.3)

$$L_{reg} = \sum_l^N \sum_i^{n^{[l]}} \sum_j^{n^{[l+1]}} w_{i,j}^{[l]}$$

Siendo:

- N : Número total de capas
- $n^{[l]}$: Número de neuronas de la capa l
- $n^{[l+1]}$: Número de neuronas de la capa $l+1$
- w : Parámetro
- **Actualización**: Se realiza la propagación hacia atrás, es decir, se calcula el tensor gradiente de la función del error global, para cada uno de los parámetros de la red y así poder actualizarlos.

Este procedimiento se realiza con cada una de las simulaciones de entrenamiento. Una vez que se haya realizado la vuelta completa se habrá completado una época o también llamado en inglés *epoch*.

Se itera el algoritmo sobre un número determinado de iteraciones. Este número depende de la dificultad del sistema que se está modelando.

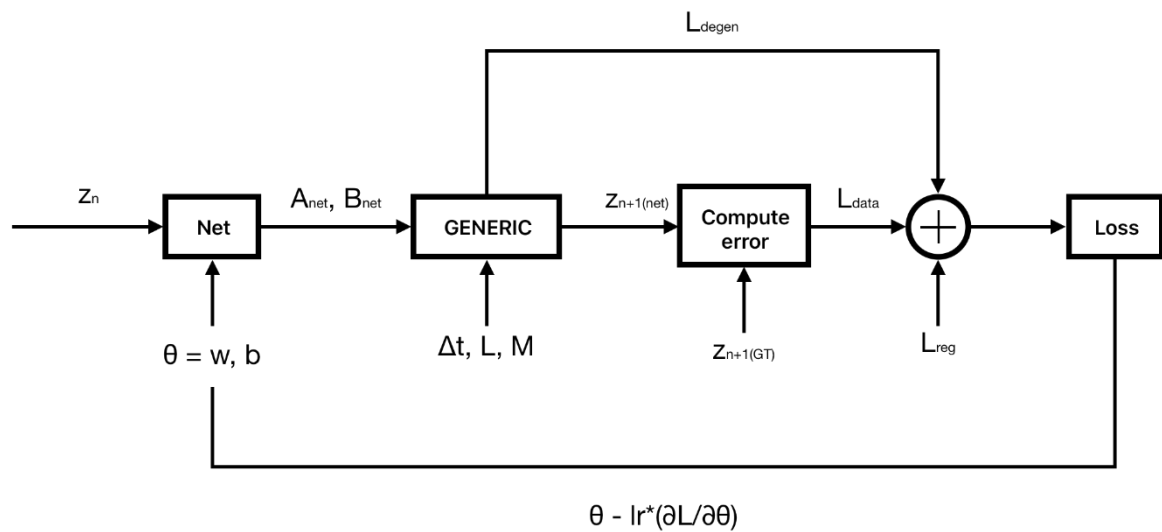


Figura 2.3 Esquema general SPNN

2.3 Variaciones de la SPNN

Respecto al esquema general de la SPNN, como se muestra en la figura 2.4, se realizarán dos cambios para adaptarla al problema concreto:

- **Red:** Este cambio permite a la red alternar entre:
 - ANN, red neuronal artificial básica:
 - Modelo determinista: siempre realiza la misma predicción ante una misma entrada.
 - BNN, red neuronal bayesiana
 - Modelo estocástico: se puede muestrear el modelo para así analizar su incertidumbre.
- **Aprendizaje de L y M:** El modelo alternará entre suponer las matrices L y M entre conocidas y desconocidas.

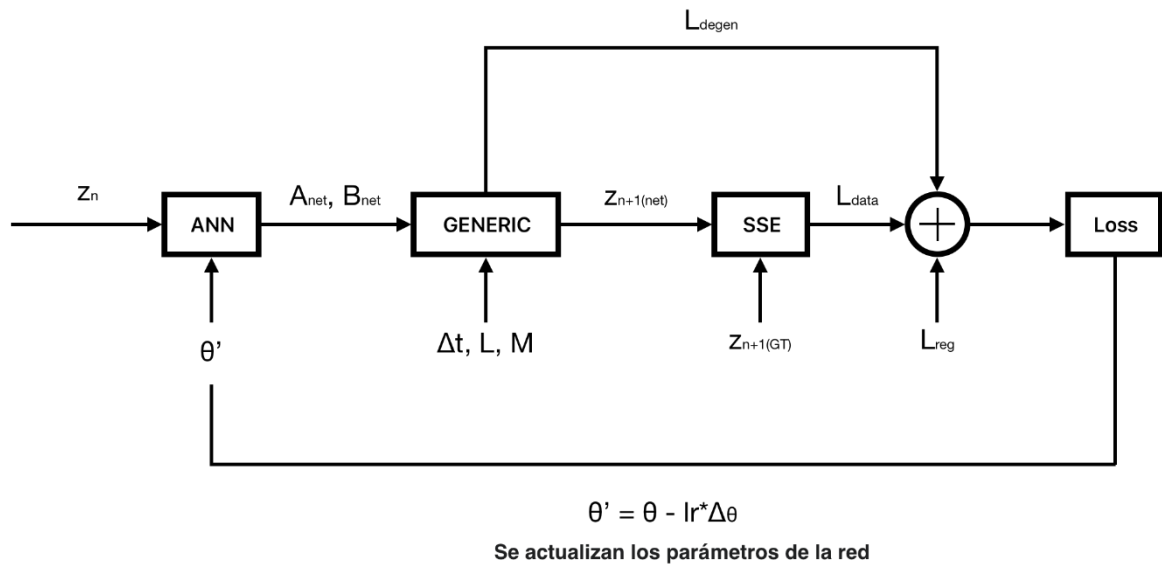


Figura 2.4 Esquema ANN sin aprender matrices L y M

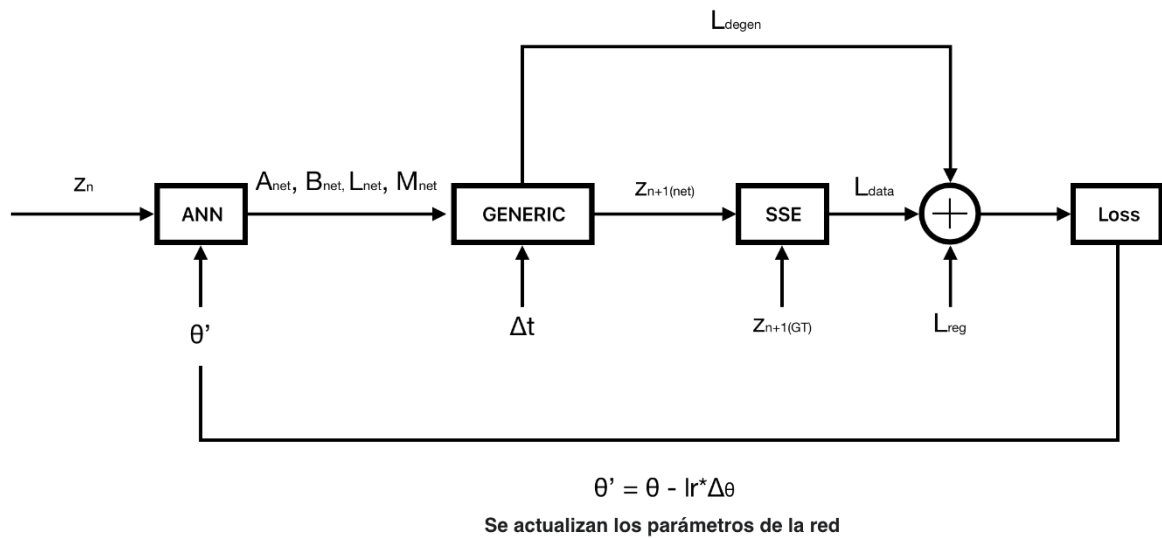


Figura 2.5 Esquema ANN sin aprender matrices L y M

2.3.1 Red neuronal bayesiana

La pérdida global para minimizar en las redes neuronales bayesianas hay que añadirle el error de la divergencia de Kullback Liebler entre la distribución a posteriori y la distribución aproximada.

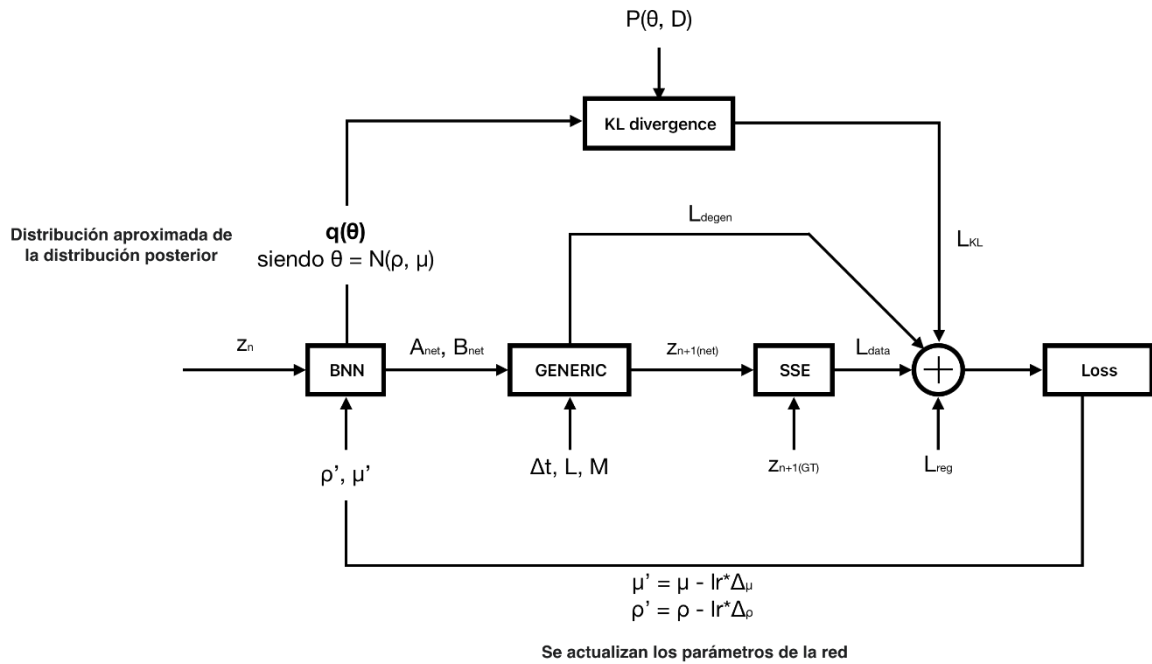


Figura 2.6 Esquema BNN sin aprender matrices L y M

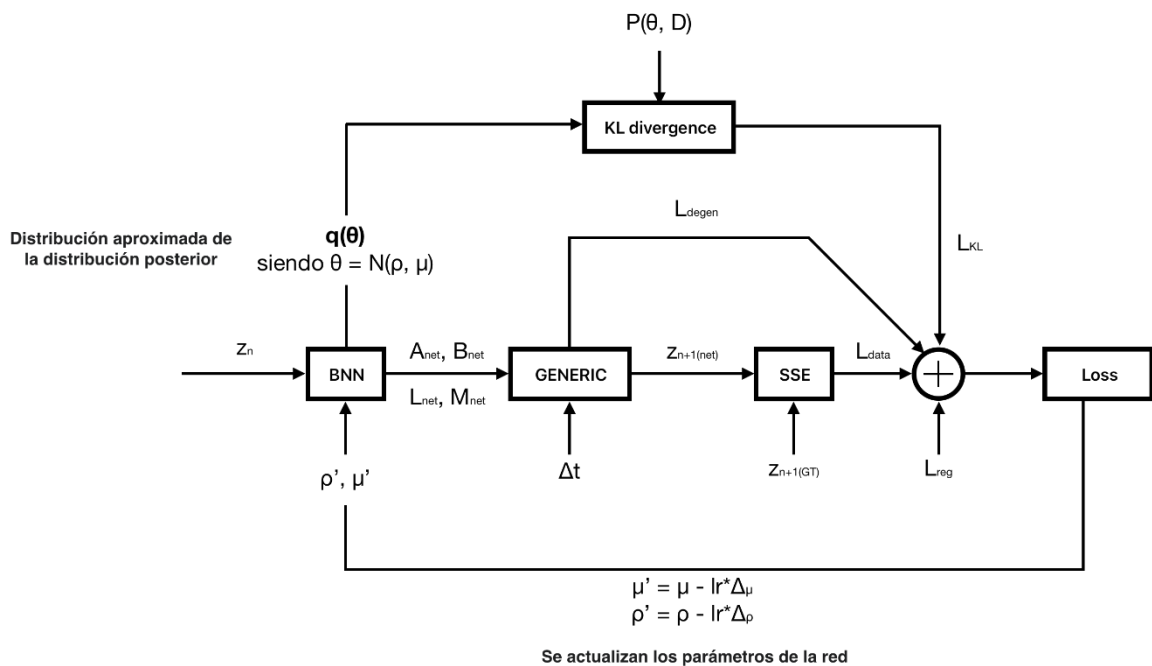


Figura 2.7 Esquema BNN aprendiendo matrices L y M

2.4 Comparación de estructuras

Para la correcta comprensión de las redes SPNN se comparan los resultados de ambos tipos de redes en un ejemplo de validación presentado en el artículo *structure-preserving neural networks* ^[1]. El artículo sólo presenta resultados basados en una red neuronal artificial con las matrices L y M definidas. En la comparación se realizan

comprobaciones en los dos casos presentados: conociendo L y M, y sin ser conocidas. Estas dos casuísticas se evaluarán con redes SPNN- ANN y SPNN- BNN.

2.4.1 Caso base: aprendizaje de la dinámica del péndulo doble

En este apartado se harán dos entrenamientos, aprendiendo las matrices L y M y dando como entrada de GENERIC las matrices.

En la tabla 2.1 se muestran los parámetros usados para el entrenamiento de las redes neuronales, aunque posteriormente serán modificados para ver cómo reaccionan ante cambios llevando las redes al límite. La descripción de los parámetros se puede ver en el anexo B.

| Parámetros | BNN | ANN |
|----------------|------------------------|------------------------|
| max_epoch | 1500 | 1500 |
| train_percent | 80% | 80% |
| lr | 1e-3 | 1e-3 |
| hidden_vec | [50, 50, 50, 50, 50] | [50, 50, 50, 50, 50] |
| miles | [500, 700, 1200, 1400] | [500, 700, 1200, 1400] |
| gamma | 5e-1 | 5e-1 |
| lambda_r | 1e-05 | 1e-05 |
| lambda_d | 4e5 | 4e5 |
| lambda_deg | 1 | 1 |
| lambda_kl | 1e-2 | - |
| n_samples | 5 | - |
| n_samples_ELBO | 3 | - |
| CI | 95% | - |
| net_init | - | kaiming_uniform |

Tabla 2.1 Configuración de simulación en condiciones normales

2.4.1.1 Simulación sin aprender matrices L y M

Para la comparación de los resultados se muestran los diagramas de caja en la figura 2.8.

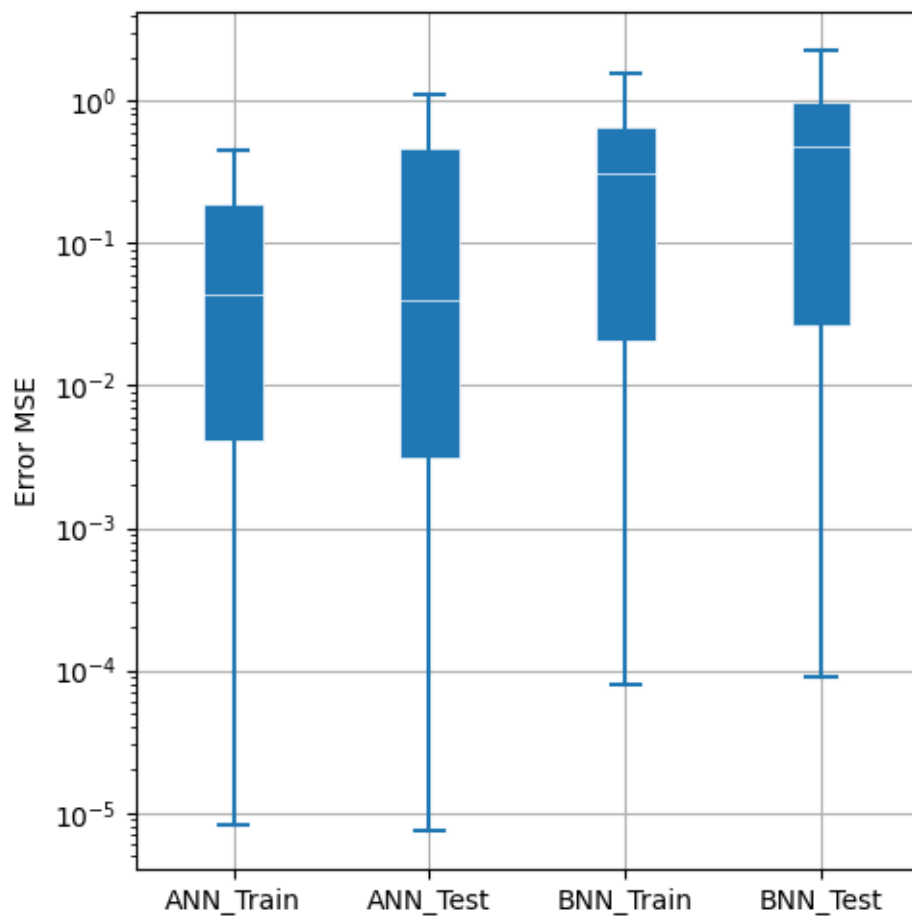


Figura 2.8 Gráfica de diagramas de caja del error medio de todas las variables de estado. No aprenden matrices L y M.

Si se analizan los errores generados por cada tipo de red las ANN son más precisas ya que sus errores son menores que las redes BNN.

Para este primer caso se mostrarán las gráficas de *test* de las dos redes para comparar visualmente los resultados.

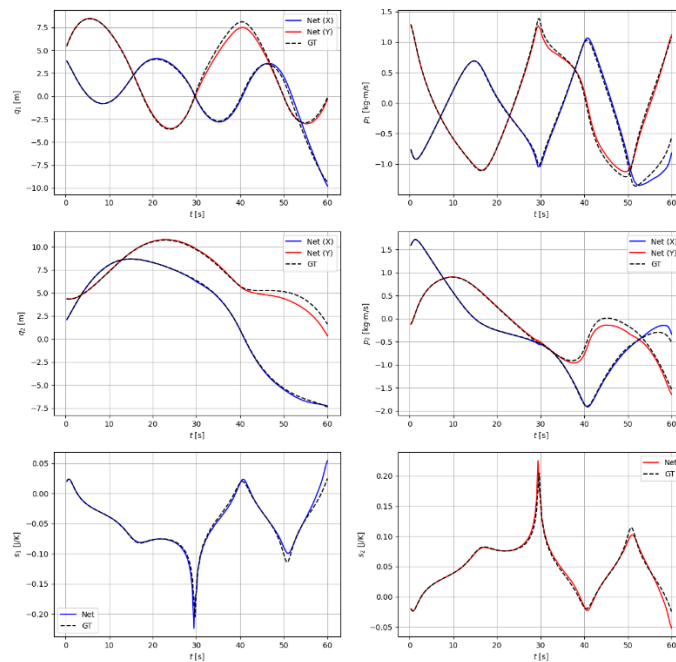


Figura 2.9 Trayectoria del test. Red ANN

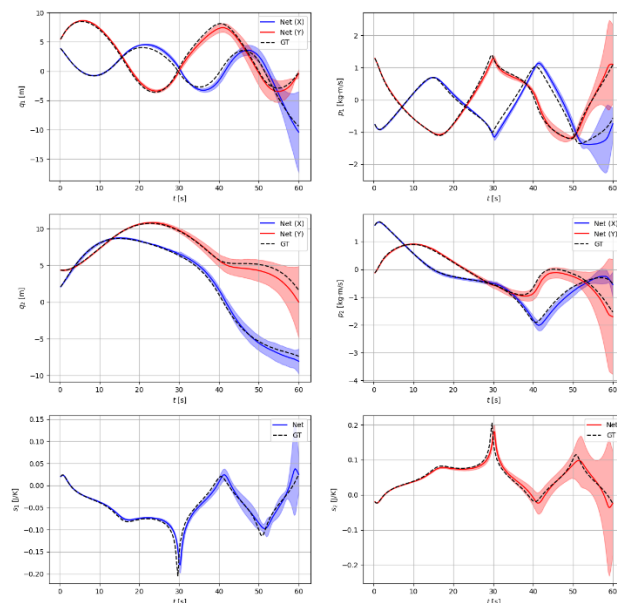


Figura 2.10 Trayectoria del test. Red BNN.

Intervalo de confianza: 95%

Lo más importante de esta gráfica, y que también ocurrirá en las siguientes, es el ensanchamiento del intervalo de confianza conforme avanza la simulación. Esto es debido a que cuanto mayor sea el tiempo de simulación, mayor es el error acumulado, y aumentará así la probabilidad de un desvío de la predicción con el problema real.

2.4.1.2 Simulación aprendiendo también matrices L y M

Se utilizan los mismos parámetros que para el caso anterior, pero ahora ya no es necesario estudiar el sistema para obtener las ecuaciones que lo rigen. Esta ventaja es notoria debido a que solo se necesitan datos, aunque hay que predecir más operadores. La pregunta es:

¿Se deteriora la solución?

Si la respuesta a la anterior pregunta es, sí ¿Cuál es la magnitud del error? ¿Justifica la deducción de todas las restricciones del sistema?

Los resultados de esta simulación se muestran en la figura 2.11.

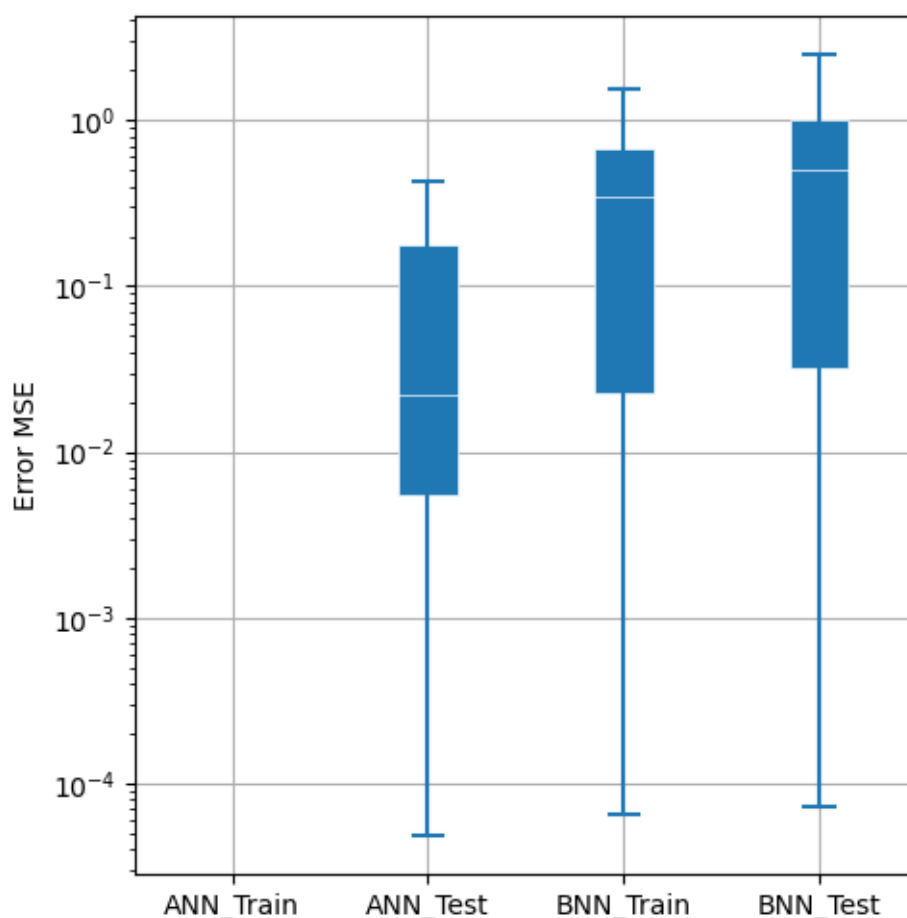


Figura 2.11 Gráfica de diagramas de caja del error medio de todas las variables de estado. Se aprenden matrices L y M. 1500 *epoch*.

Como se aprecia en la figura 2.11 el diagrama de caja de la red ANN en la etapa de entrenamiento no aparece. Esto es debido a que con estos parámetros la red ANN no es capaz de predecir todo el *dataset* de entrenamiento. Los resultados van acumulando demasiado error hasta que llega un punto en el que el resultado de la red es infinito. Sin embargo, la red BNN es más estable.

Antes de realizar otra simulación cambiando los parámetros para poder tener una comparación entre redes ANN y BNN, se hace una comparación de las redes que no estaban aprendiendo las matrices L y M (figura 2.8) con las que sí que aprenden (figura 2.11).

No se deteriora la solución para las redes BNN, aunque L y M no sea conocido.

En cambio, para las redes ANN sí que hay empeoramiento notable ya que la precisión desciende a razón de un orden de magnitud.

Para poder comparar las dos redes se decide realizar otro caso aumentando las épocas de entrenamiento hasta que las ANN arrojen un resultado en la etapa de entrenamiento. La red ANN a partir de las 4000 épocas predice adecuadamente todos los casos de simulación. Después se simula la red BNN con 4000 épocas y así se puede comparar en igualdad de condiciones tal y como se aprecia en la figura 2.14.

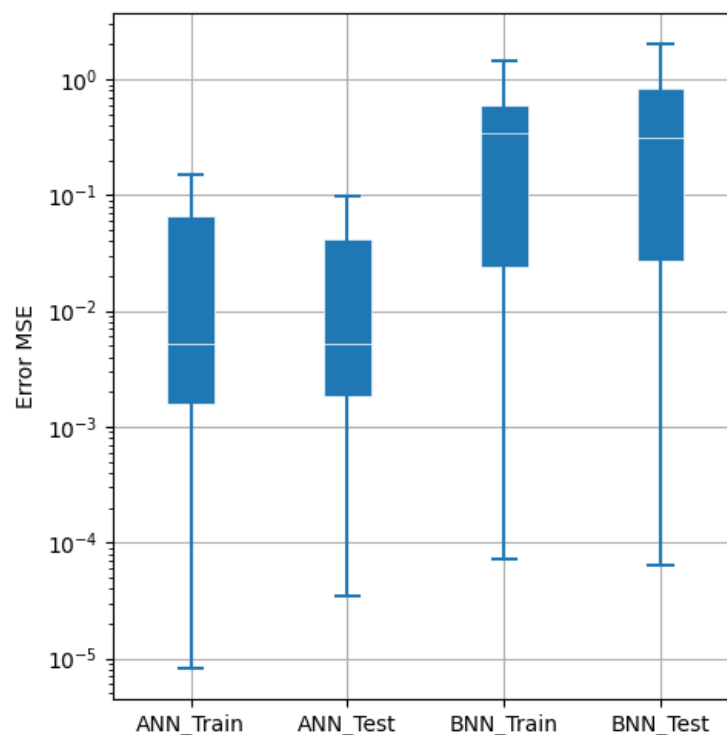


Figura 2.14 Gráfica de diagramas de caja del error medio de todas las variables de estado. Se aprenden matrices L y M. 4000 epoch.

Analizando los resultados se ve como las ANN ya no llegan a tanta precisión como en el caso de que L y M sean conocidos. Sin embargo, las BNN no mejoran, aunque se aumenten las iteraciones de entrenamiento.

2.4.2 Base de datos reducida

Se espera que, si la red es forzada a ser entrenada con los mínimos datos posibles, las redes bayesianas presenten mejores resultados gracias a su enfoque probabilístico.

Los parámetros utilizados serán los mismos que en la tabla 2.1, pero se entrenará con menos datos. Ésto implica que ahora se eliminarán varias trayectorias del *dataset* de entrenamiento y se pasarán a la fase de *test*, por lo que evaluarán muchas más trayectorias que no se han visto antes.

| Parámetros | ANN | BNN |
|---------------|-----|-----|
| train_percent | 40% | 40% |

Tabla 2.2 Configuración de simulación con *dataset* reducido

2.4.2.1 Simulación sin aprender matrices L y M

Para la comparación de los resultados se muestran los diagramas de caja en la figura 2.15. En este caso sólo se incluyen los diagramas de caja de la BNN porque las ANN no han sido capaces de extraer un resultado.

La mayor parte de las trayectorias de entrenamiento tienden a infinito, por lo que se puede concluir que las redes BNN cuando tienen menos datos se comportan mucho mejor.

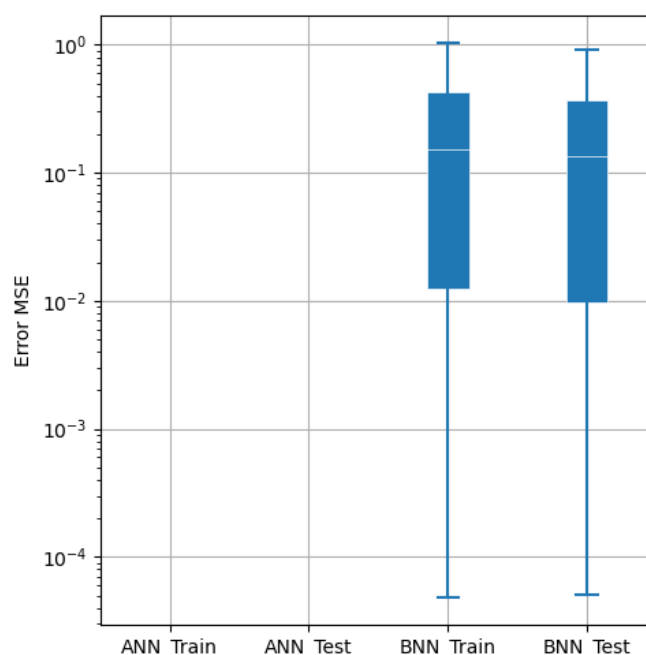


Figura 2.15 Gráfica de diagramas de caja del error medio de todas las variables de estado. *Dataset* reducido. 40% del *dataset* de entrenamiento

En la figura 2.16 se muestra la precisión de las redes BNN en el caso de reducirse el porcentaje de entrenamiento al 30%. Comparando la figura 2.16 con la 2.15 sí que se aprecia una disminución de precisión.

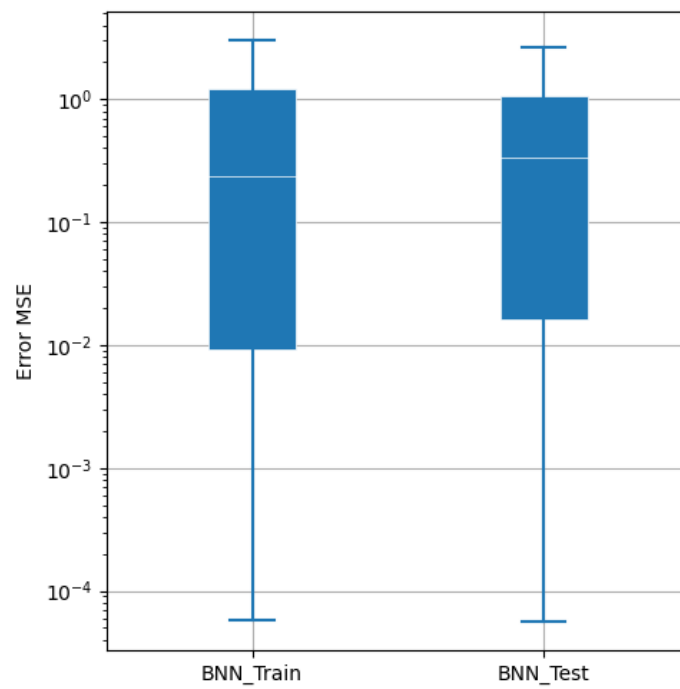


Figura 2.16 Gráfica de diagramas de caja del error medio de todas las variables de estado. *Dataset* reducido. 30% del *dataset* de entrenamiento

3 Aplicación de redes neuronales a SHM

Una vez se comprueba el funcionamiento de las redes neuronales bayesianas en el ejemplo del péndulo doble se procede a la aplicación de monitorización estructural a un ejemplo típico en el campo de la mecánica de medios continuos.

3.1 Metodología

Con el fin de poder adaptar el ejemplo estructural a la red se decide eliminar el término entropía del formalismo GENERIC. Esto es debido a que el problema es elástico lineal, no hay fenómenos disipativos. Es decir, ya no existen irreversibilidades que deba aprender la red neuronal.

Otra consecuencia de no tener término disipativo es que no es necesario aplicar las condiciones de degeneración. Las únicas dos condiciones son la aplicación del esquema GENERIC, y asegurar que L es semidefinida positiva. Por lo tanto, el formalismo se va a cumplir por construcción. En consecuencia, la ecuación (1.8) se transforma en la ecuación (3.1):

$$(3.1) \quad \mathbf{z}_{n+1} = \mathbf{z}_n + \Delta t (\mathbf{L} \cdot \mathbf{A} \mathbf{z}_n)$$

3.1.1 Creación de la base de datos

La base de datos se crea mediante simulación computacional en un software comercial de elementos finitos para obtener datos pseudo-experimentales. Se define una única geometría con unas condiciones de contorno concretas y un único caso de carga. El objetivo es caracterizar el problema bajo estas condiciones para que, al cambiar las propiedades del material, se pueda detectar los fallos de la estructura.

Es importante destacar que se crean dos modelos de elementos finitos:

- Modelo con mismo módulo de Young: modelo que representa una estructura sin fallos.
- Modelo con dos regiones con módulo de Young diferente: Misma estructura que la anterior, pero se le añade una región con un módulo de Young diferente.

3.1.2 Entrenamiento del modelo. Gemelo digital

El entrenamiento del modelo estructural es similar al del péndulo doble.

Primero se entrena el modelo con la base de datos que posee un mismo módulo de Young. En este caso, el modelo ya sería capaz de predecir la evolución de las variables de estado y módulo de Young en cualquier punto de la estructura, incluyendo los que no están presentes en el entrenamiento.

Esto constituye el gemelo digital de la estructura, es decir, un modelo virtual capaz de replicar el comportamiento físico de la estructura.

En cambio, si ahora se introduce la base de datos del modelo con el fallo estructural se observa cómo no predice correctamente su comportamiento, por lo que el gemelo requiere una actualización que permitirá analizar a que se debe el cambio detectado. En caso de aplicarlo prácticamente se requiere conocer la ubicación del fallo ya que en base a los resultados ya se intuye que el fallo existe.

3.1.3 Desarrollo del gemelo híbrido

El cambio de comportamiento se asocia a un posible fallo en la estructura. Para poder establecer la localización del fallo se introducirá el concepto de gemelo híbrido. El gemelo híbrido es una evolución del gemelo digital. Este es capaz de adaptarse a los cambios del sistema real cuando detecta una anomalía. La clave está en la conexión del gemelo híbrido en tiempo real con las fluctuaciones de la estructura que permiten ajustar su comportamiento y enriquecer la aproximación existente.

En términos matemáticos el gemelo híbrido consistiría en sumar al gemelo digital el error que está cometiendo. Por lo tanto, para poder crear este gemelo híbrido se debe de crear otra red que solo se encargue de aprender sobre el error cometido por el gemelo digital.

$$(3.2) \quad \mathbf{Z}_{GH} = \mathbf{Z}_{GD} + \mathbf{Z}_E$$

Siendo:

- \mathbf{Z}_{GH} : Datos predichos por el gemelo híbrido.
- \mathbf{Z}_{GD} : Salida de la primera red neuronal (Gemelo Digital).
- \mathbf{Z}_E : Salida de la segunda red neuronal que solo predice el error.

Para poder seguir con el desarrollo del gemelo híbrido solo se estudian unos pocos nodos repartidos uniformemente en la estructura como si fueran sensores reales.

Estos sensores sirven para hacer un seguimiento del comportamiento de la estructura. Primero se compara la predicción de la primera red con las medidas de los sensores. En este punto se verá una desviación entre las mediciones y las predicciones, dando lugar a una base de datos de errores que se empleará para entrenar la segunda red.

Posteriormente se trata de crear una segunda red que consiga aprender el error de cada variable para poder sumárselo a la predicción de la primera red y así conseguir una predicción correcta.

$$(3.3) \quad \dot{\mathbf{Z}}_E = \dot{\mathbf{Z}}_{GT} - \dot{\mathbf{Z}}_{GD}$$

$$(3.4) \quad \dot{\mathbf{Z}}_E = \mathbf{L}\nabla\mathbf{E}(\mathbf{Z}_E)$$

$$(3.5) \quad \dot{\mathbf{Z}}_{GT} = \mathbf{L}\nabla\mathbf{E}(\mathbf{Z}_{GD}) + \mathbf{L}\nabla\mathbf{E}(\mathbf{Z}_E)$$

$$(3.6) \quad \frac{Z_{E,n+1} - Z_{E,n}}{\Delta t} = LDE(Z_E)$$

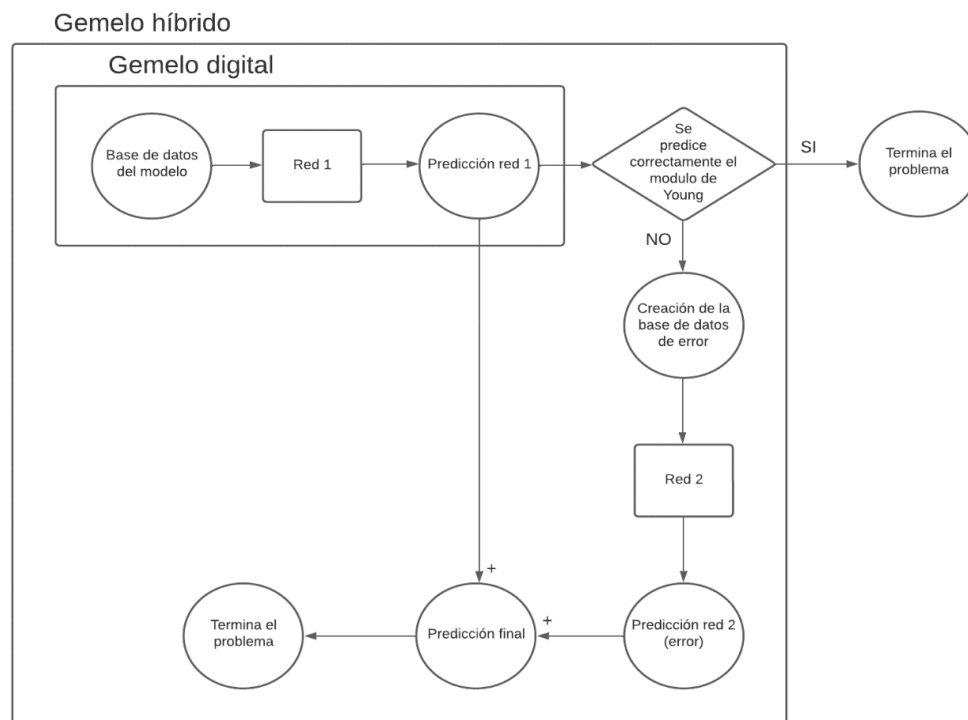


Figura 3.1 Esquema gemelo digital/híbrido

3.2 Implementación: Placa apoyada-empotrada

El ejemplo elegido para poder desarrollar la técnica del SHM va a ser un problema típico de la mecánica del sólido deformable, una placa apoyada - empotrada. Se trata de una placa cuadrada 2D apoyada en los laterales y empotrada en su base, ilustrada en la figura 3.2.

Tal y como se ha comentado se crean dos modelos. En la figura 3.2 se observa el modelo sin fallo y en la figura 3.3 se aprecia en gris la zona con el fallo.

Se le ha asignado un módulo de Young de 200.000 MPa. En cambio, para la zona afectada por el fallo se le ha reducido a un módulo de Young de 100.000 MPa. Se entiende como un modelo ideal.

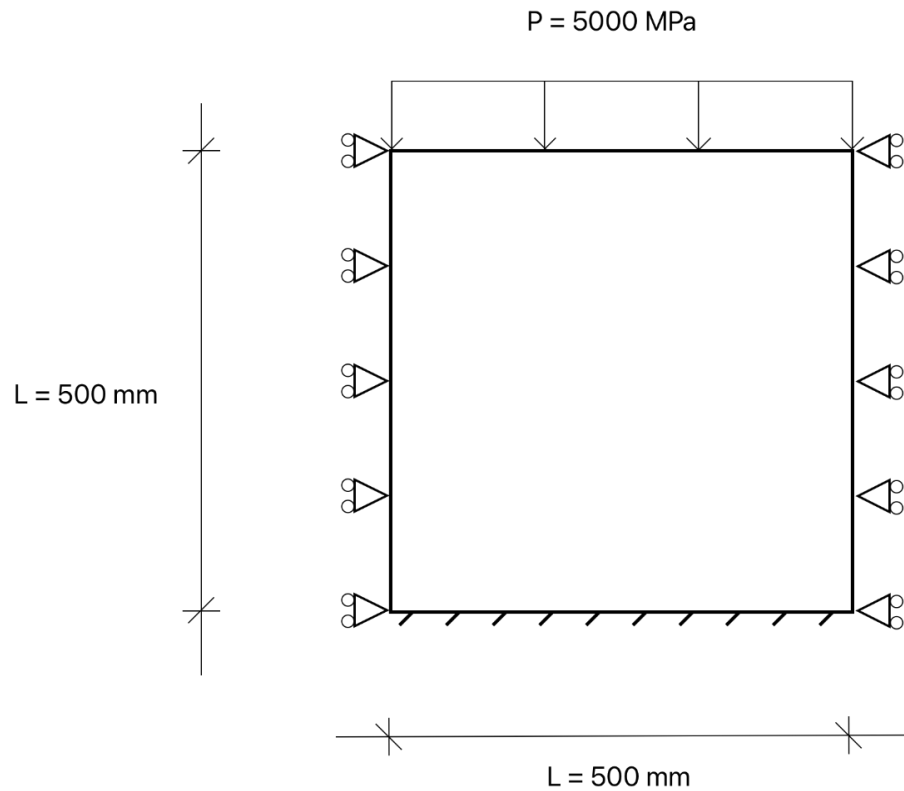


Figura 3.2 Esquema placa apoyada-empotrada sin fallo.

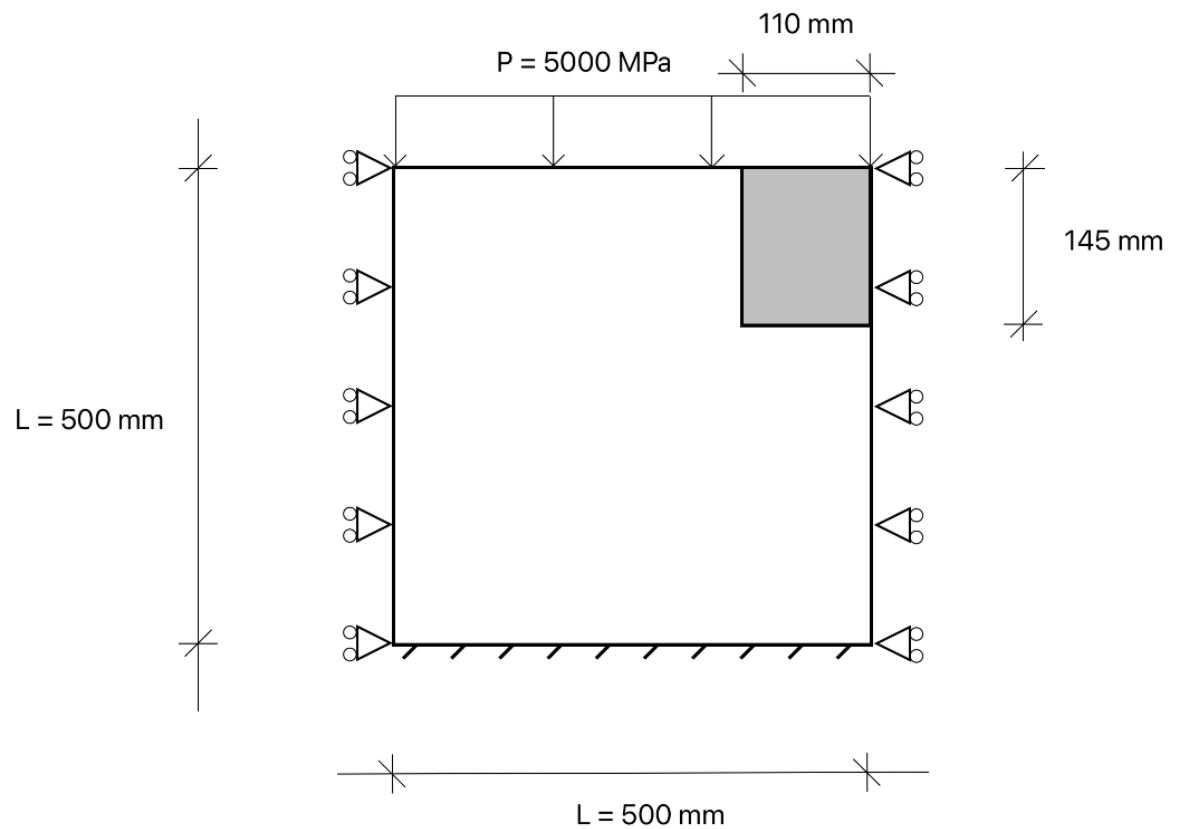


Figura 3.3 Esquema placa apoyada-empotrada con fallo.

3.2.1 Módulo de Young

Para poder determinar el fallo se debe de conocer el módulo de Young y este debe ser despejado de las ecuaciones de comportamiento.

Para describir el estado de deformaciones y tensiones se parte de:

$$(3.7) \quad \boldsymbol{\sigma} = \begin{pmatrix} \sigma_x & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & \sigma_y & \tau_{yz} \\ \tau_{xz} & \tau_{yz} & \sigma_z \end{pmatrix} \quad \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_x & \varepsilon_{xy} & \varepsilon_{xz} \\ \varepsilon_{xy} & \varepsilon_y & \varepsilon_{yz} \\ \varepsilon_{xz} & \varepsilon_{yz} & \varepsilon_z \end{pmatrix}$$

Debido a que se trata de una placa en 2D y que las caras perpendiculares son libres se está trabajando con un problema de tensión plana.

$$(3.8) \quad \sigma_z = \tau_{xz} = \tau_{yz} = 0$$

Además, en la dirección x no se puede estirar el sólido debido a que se encuentra entre dos apoyos.

$$(3.9) \quad \varepsilon_x = 0$$

Escribiendo la ecuación de Hooke de un sólido de 3 dimensiones y un material isótropo para las direcciones principales:

$$(3.10) \quad \varepsilon_x = \frac{\sigma_x - \nu(\sigma_y + \sigma_z)}{E}$$

$$(3.11) \quad \varepsilon_y = \frac{\sigma_y - \nu(\sigma_x + \sigma_z)}{E}$$

$$(3.12) \quad \varepsilon_z = \frac{\sigma_z - \nu(\sigma_x + \sigma_y)}{E}$$

Simplificando la ecuación (3.10):

$$(3.13) \quad 0 = \frac{\sigma_x - \nu \cdot \sigma_y}{E}$$
$$\nu = \frac{\sigma_x}{\sigma_y}$$

Por último, simplificando la ecuación (3.11) se obtiene el módulo de Young:

$$(3.14) \quad E = \frac{\sigma_y - \frac{\sigma_x^2}{\sigma_y}}{\varepsilon_y}$$

Siendo:

- ε_y : Deformación vertical de la placa

- σ_y : Tensión vertical de la placa, en el caso de no haber ningún fallo será constante y equivaldrá a la presión impuesta en el problema, $\sigma_y = -P$
- σ_x : Deformación horizontal de la placa

3.3 Resultados

En primer lugar, se mostrarán los resultados de la placa entrenada con la red ANN y posteriormente se estudiará la red BNN. Por último, se compararán los resultados en forma de diagramas de caja.

3.3.1 Red ANN

Primero se entrena el modelo capaz de simular la placa sin ningún fallo. La red se define con los siguientes parámetros:

| Parámetros | ANN |
|---------------|-------------------------------------|
| max_epoch | 2300 |
| train_percent | 80% |
| lr | 1e-2 |
| hidden_vec | [60, 60, 60, 60] |
| miles | [700, 1000, 1200, 1600, 1900, 2100] |
| gamma | 0.35 |
| lambda_r | 1e-05 |
| net_init | kaiming_uniform |

Tabla 2.1 Configuración del modelo de la red ANN

Resultados de la placa sin fallos:

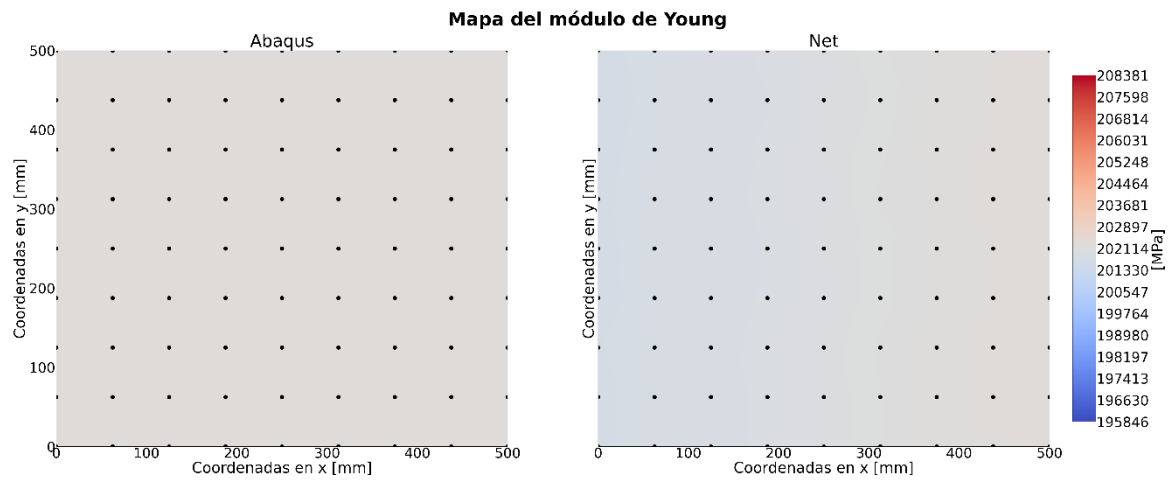


Figura 3.3 Mapa del módulo de Young de la placa sin fallo. Gemelo digital.

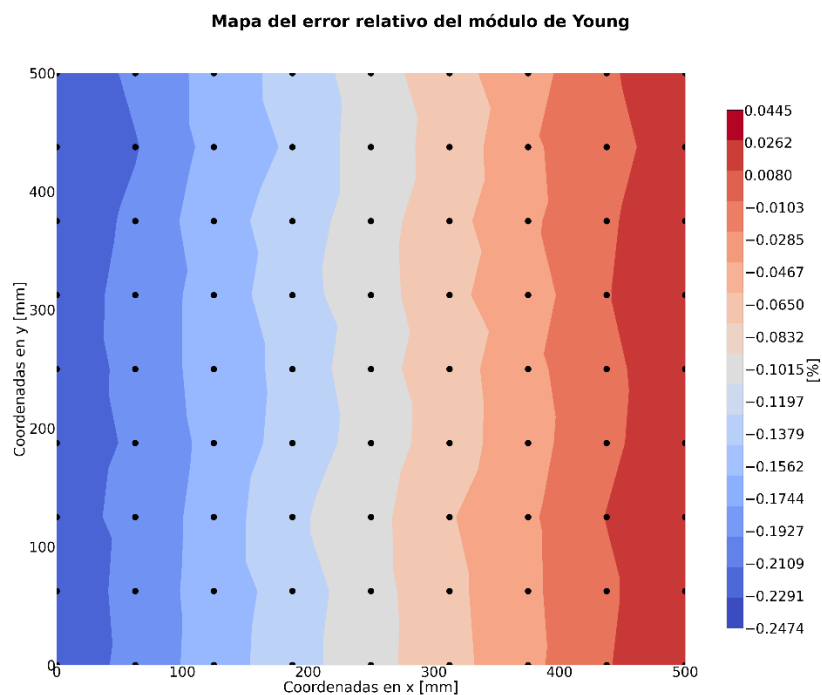


Figura 3.4 Mapa del error relativo del módulo de Young de la placa sin fallo.
Gemelo digital.

Como se puede comprobar en los mapas, la red es capaz de predecir el módulo de Young de la placa con una precisión adecuada. El resto de los resultados respecto a las otras variables de estado pueden encontrarse en el anexo C.

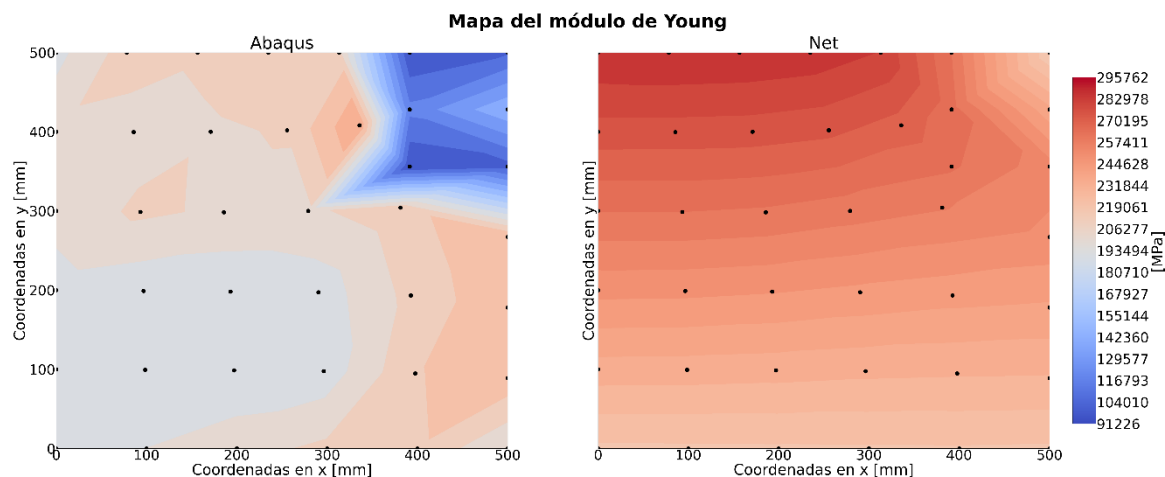


Figura 3.5 Mapa del módulo de Young de la placa con fallo. Gemelo digital.

En la figura 3.5 se observa cómo la red entrenada para la placa sin fallo no es capaz de predecir correctamente el comportamiento de la placa con el fallo.

En el caso de que se estuviera aplicando a una estructura real para poder conocer la existencia de un fallo no se dispondría de la gráfica con los datos reales (Figura 3.5, Abaqus) por lo que no se podría comparar. Sin embargo, sí que es conocido cuál es el módulo de Young del material así que sería fácilmente identificable la existencia de un fallo ya que devuelve valores extremadamente altos del módulo de Young para el material estudiado.

Ahora se conoce la existencia de un fallo en la placa, pero no se sabe el lugar del mismo. Se parte de las mediciones en la nueva placa que se supone han sido realizadas mediante sensores. Las mediciones se realizan en 40 puntos distribuidos de manera uniforme como se muestra en la figura 3.6.

Para este caso en concreto se dispondrá de una malla de 40 nodos distribuidos uniformemente, figura 3.6. Se va a entrenar con el 100 % de los nodos y después se comprobará la precisión del gemelo híbrido con otra base de datos más amplia.

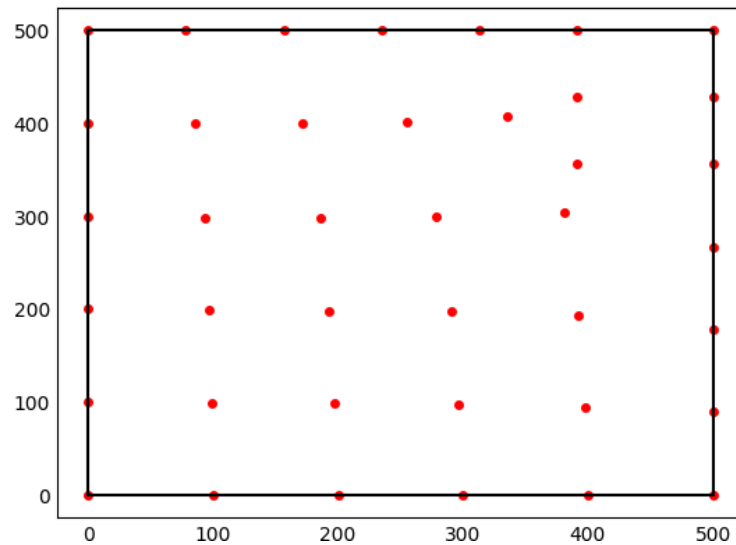


Figura 3.6 Mapa de puntos entrenados para completar el gemelo híbrido.

Después del entrenamiento de la segunda red se obtienen los siguientes resultados. Para hacer el cálculo del módulo de Young anteriormente se suman las variables de estado de la red del gemelo digital y la red que predice errores de la red. Siguiendo la ecuación (3.2).

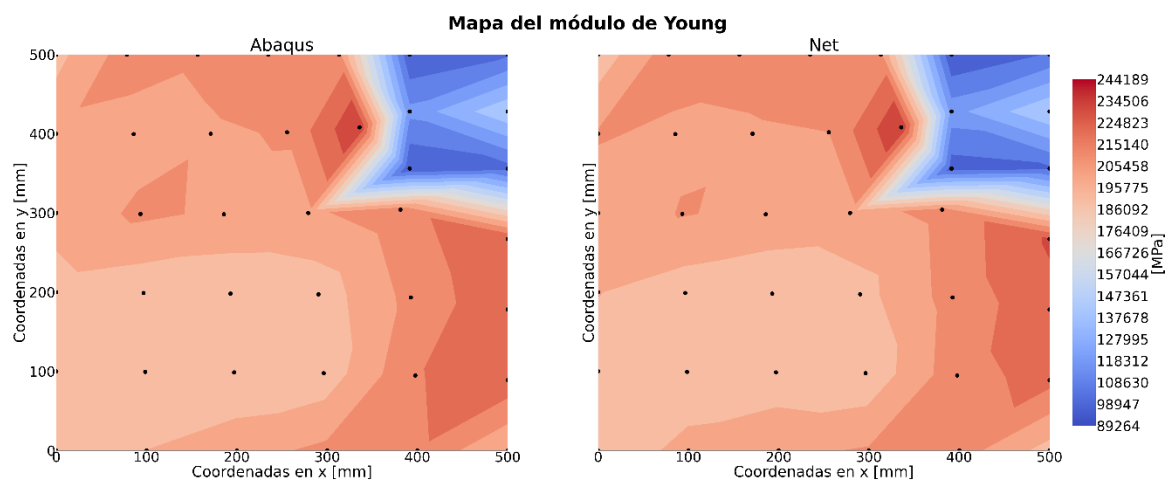


Figura 3.7 Mapa del módulo de Young de la placa con fallo. Gemelo híbrido.

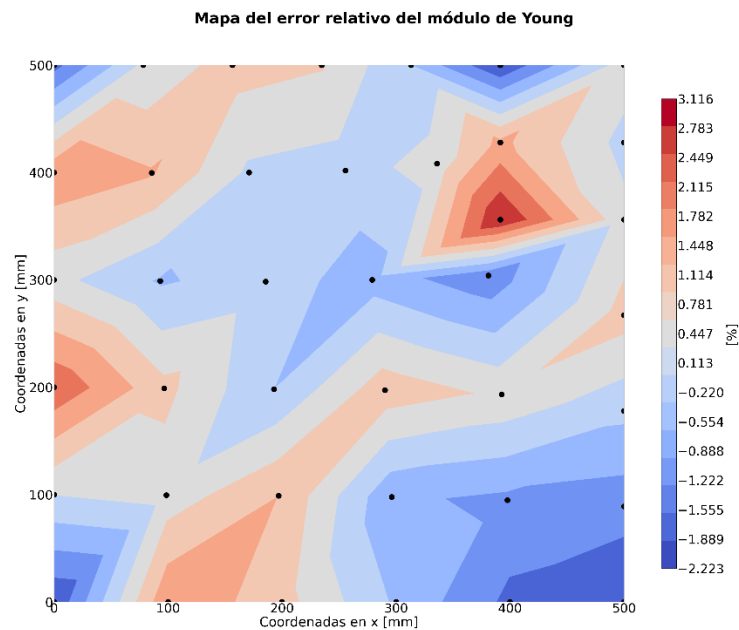


Figura 3.8 Mapa del error relativo del módulo de Young de la placa con fallo. Gemelo híbrido.

Observando los resultados se obtienen unas buenas tolerancias de error. A continuación, se analiza el comportamiento del gemelo híbrido en una malla con mayor discretización, compuesta por 827 nodos dispuestos como se muestran en la figura 3.9.

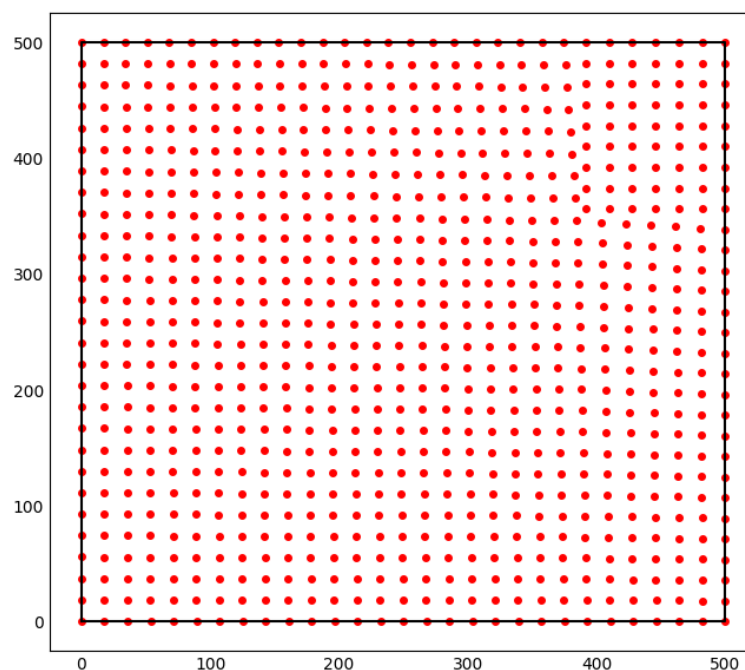


Figura 3.9 Mapa de puntos para testear el gemelo híbrido.

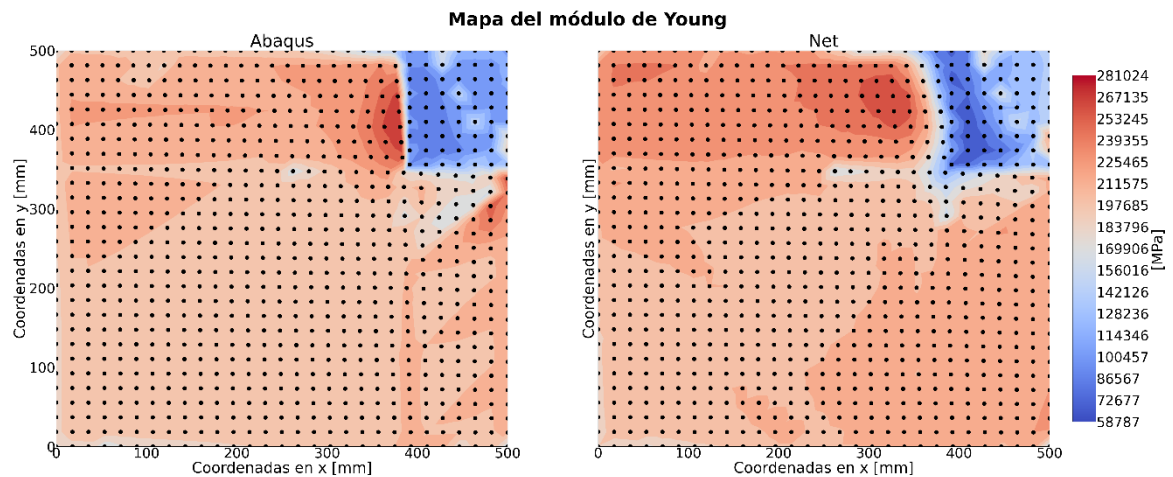


Figura 3.10 Mapa del módulo de Young de la placa con fallo. Gemelo híbrido. *Test*.

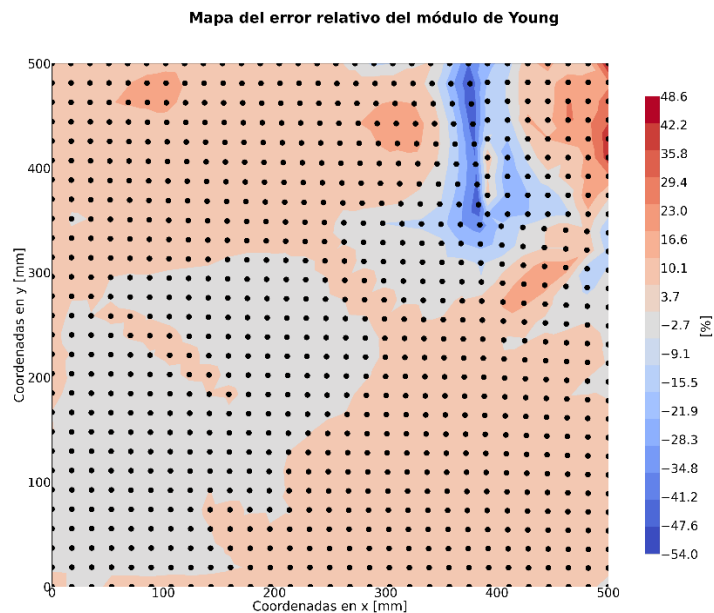


Figura 3.11 Mapa del error relativo del módulo de Young de la placa con fallo. Gemelo híbrido. *Test*.

El error queda perfectamente localizado, y la predicción del módulo de Young es precisa en la mayor parte de la placa, estando el error entorno a un $\pm 3\%$. Sin embargo, se aprecia un error mayor de manera puntual en la transición entre la zona de fallo y el resto de la placa.

Esto puede deberse al cambio brusco del módulo de Young. Las redes neuronales son capaces de generalizar para que cuando se presenten datos que no han visto todavía puedan adaptarse a ellos. En este caso hay una discontinuidad muy evidente por lo que en la parte de la frontera genera un cambio gradual en vez de una discontinuidad.

3.3.2 Red BNN

Los hiperparámetros de la red que modela la placa sin fallos en la red neuronal bayesiana son:

| Parámetros | BNN |
|----------------|---|
| max_epoch | 1300 |
| train_percent | 80% |
| lr | 1e-2 |
| hidden_vec | [30, 30, 30] |
| miles | [200, 400, 600, 750, 900, 1000, 1100, 1200] |
| gamma | 0.5 |
| lambda_r | 1e-05 |
| lambda_d | 1e09 |
| lambda_kl | 1e-2 |
| n_samples | 5 |
| n_samples_ELBO | 3 |
| CI | 95% |

Tabla 2.2 Configuración del modelo de la red BNN

A continuación, se muestran los resultados de la red bayesiana para la placa sin fallos:

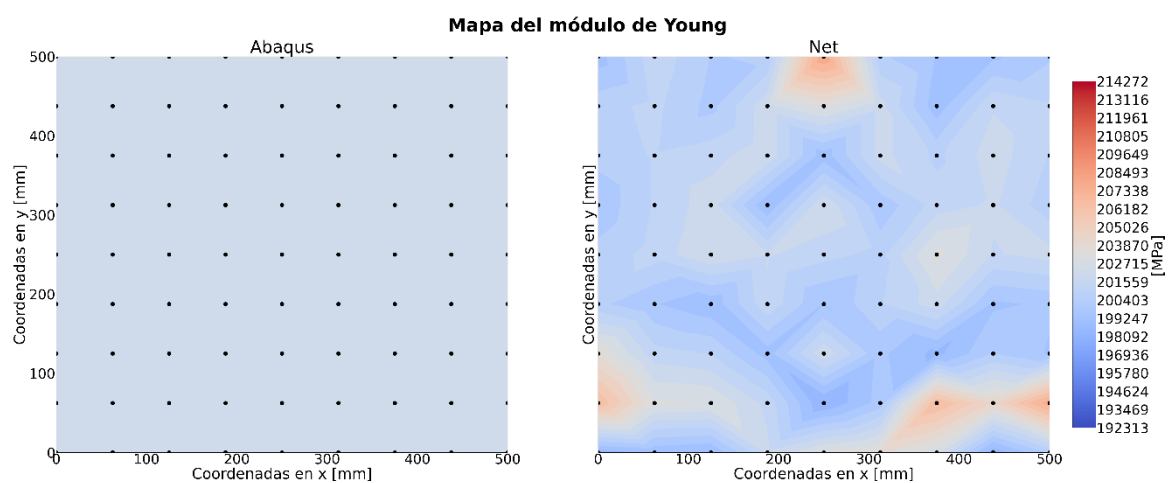


Figura 3.12 Mapa del módulo de Young de la placa sin fallo. Gemelo digital.

Sin embargo, como se trata de una red bayesiana se puede conocer los valores del módulo de Young en un intervalo de confianza. A continuación, se muestran los resultados para un intervalo de confianza del 95%:

Mapa del módulo de Young

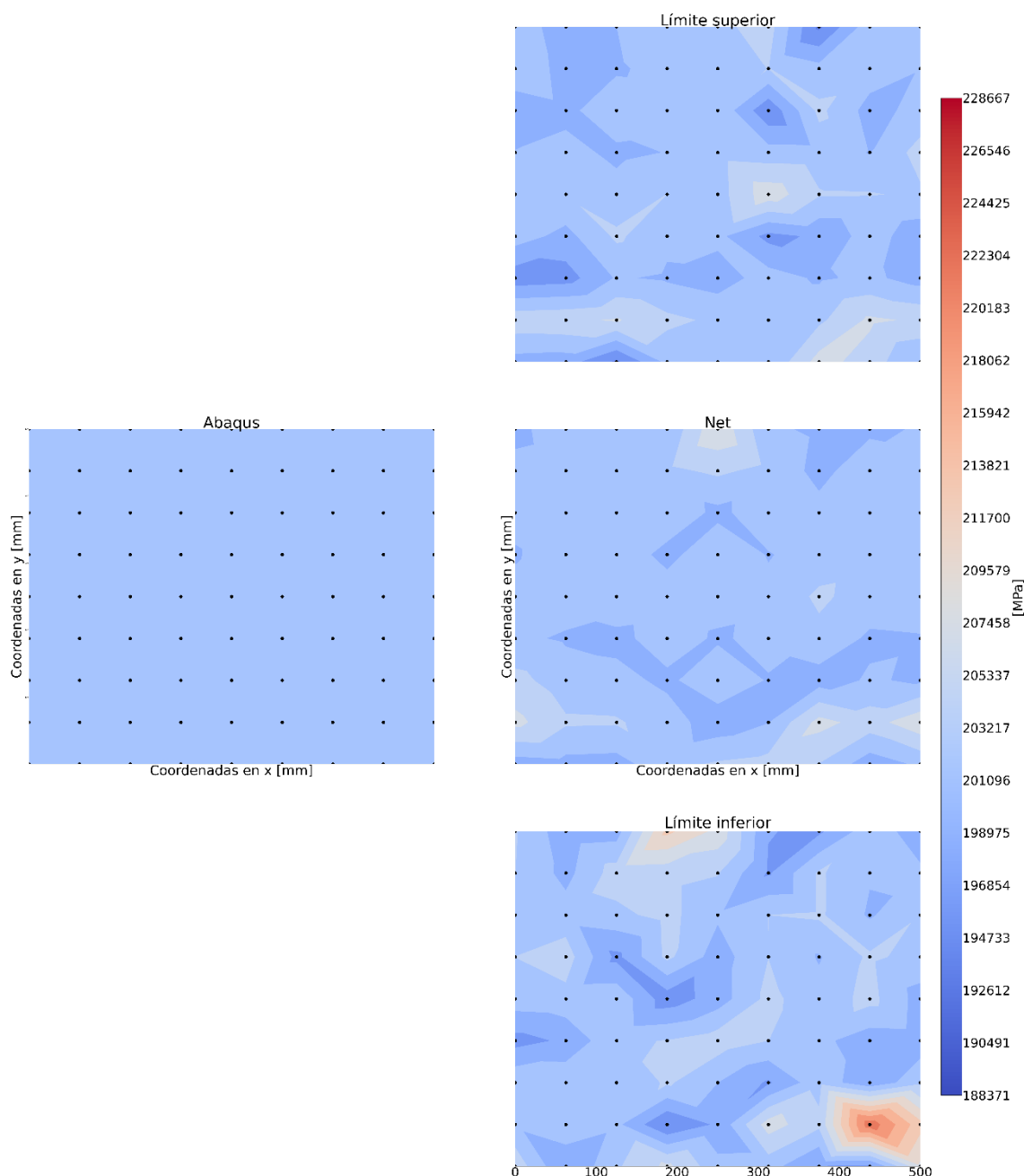


Figura 3.13 Mapa del módulo de Young de la placa sin fallo. Intervalo de confianza al 95%. Gemelo digital.

Mapa del error relativo del módulo de Young

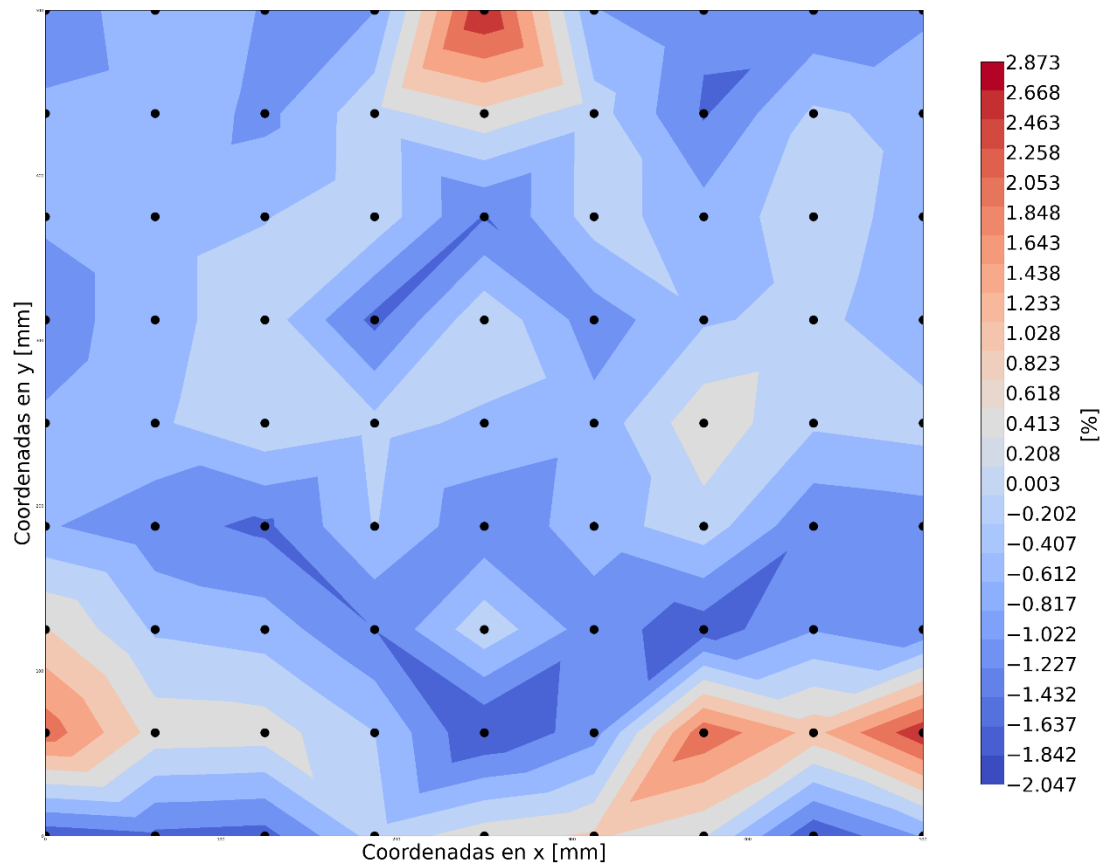


Figura 3.14 Mapa del error del módulo de Young de la placa sin fallo.
Gemelo digital.

Mapa del error relativo del módulo de Young

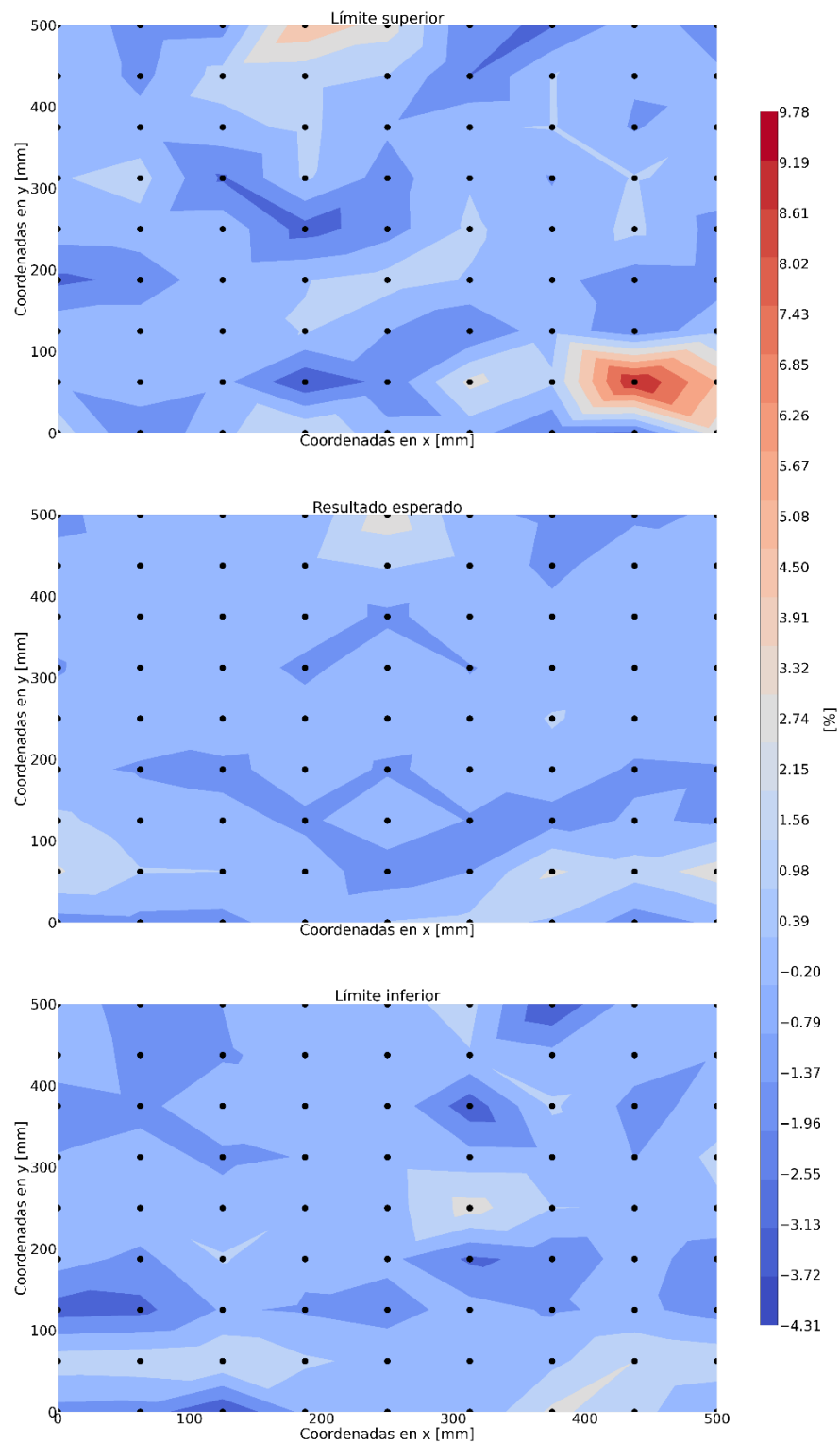


Figura 3.15 Mapa del módulo de Young de la placa sin fallo. Intervalo de confianza al 95%. Gemelo digital.

El error cometido en el módulo de Young por la red bayesiana es un poco mayor, pero a cambio también se obtiene un intervalo de confianza y así se puede vincular al grado de certeza de cada valor predicho.

Tras numerosos intentos para que la red bayesiana aprenda el segundo modelo y así pueda completar el gemelo híbrido no se consigue entrenar un modelo estable. Para ninguna configuración de los parámetros de la red. Esto se puede deber a dos factores:

- Error excesivamente pequeño para el segundo modelo del gemelo híbrido.
- Poca información para que así pueda generalizar y así poder encontrar un gradiente adecuado.

En la figura 3.16 se observa una zona en la esquina derecha superior con un módulo de Young más bajo por lo que sí que se podría localizar el fallo. Pero el error relativo cuando se compara con los datos esperados es muy grande, figura 3.17.

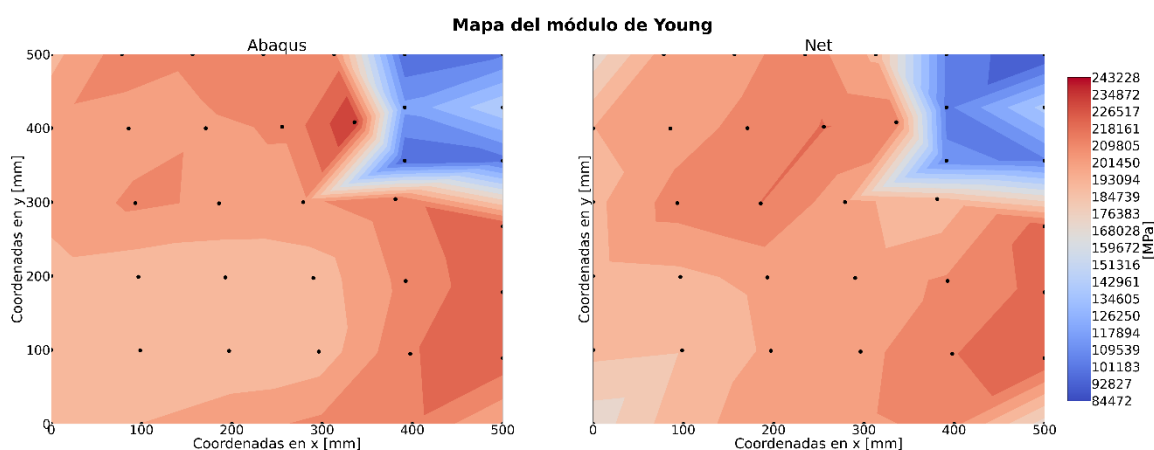


Figura 3.16 Mapa del módulo de Young de la placa con fallo. Gemelo híbrido.

Mapa del error relativo del módulo de Young

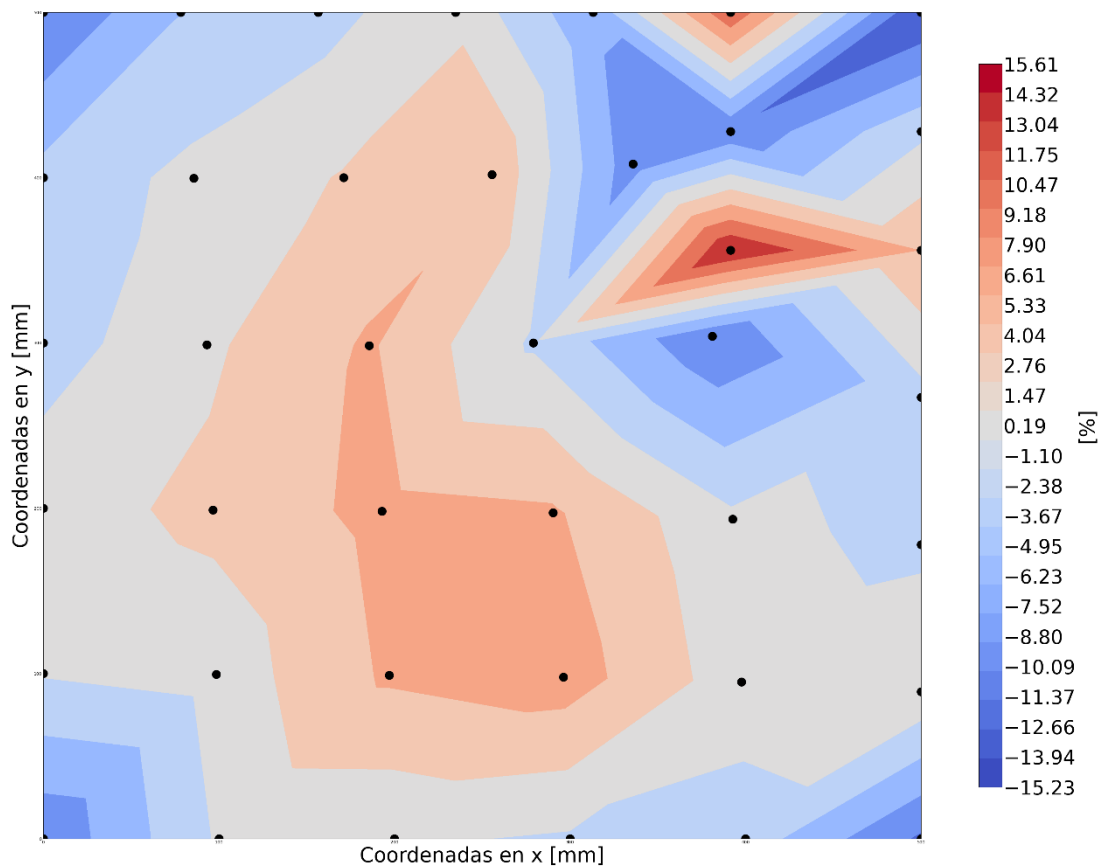


Figura 3.17 Mapa del error relativo del módulo de Young de la placa con fallo. Gemelo híbrido.

3.3.3 Comparación gemelo digital en ANN y BNN

Para poder hacer una comparación directa entre las dos redes se generan los diagramas de caja del error cometido por las redes en el entrenamiento y en la prueba.

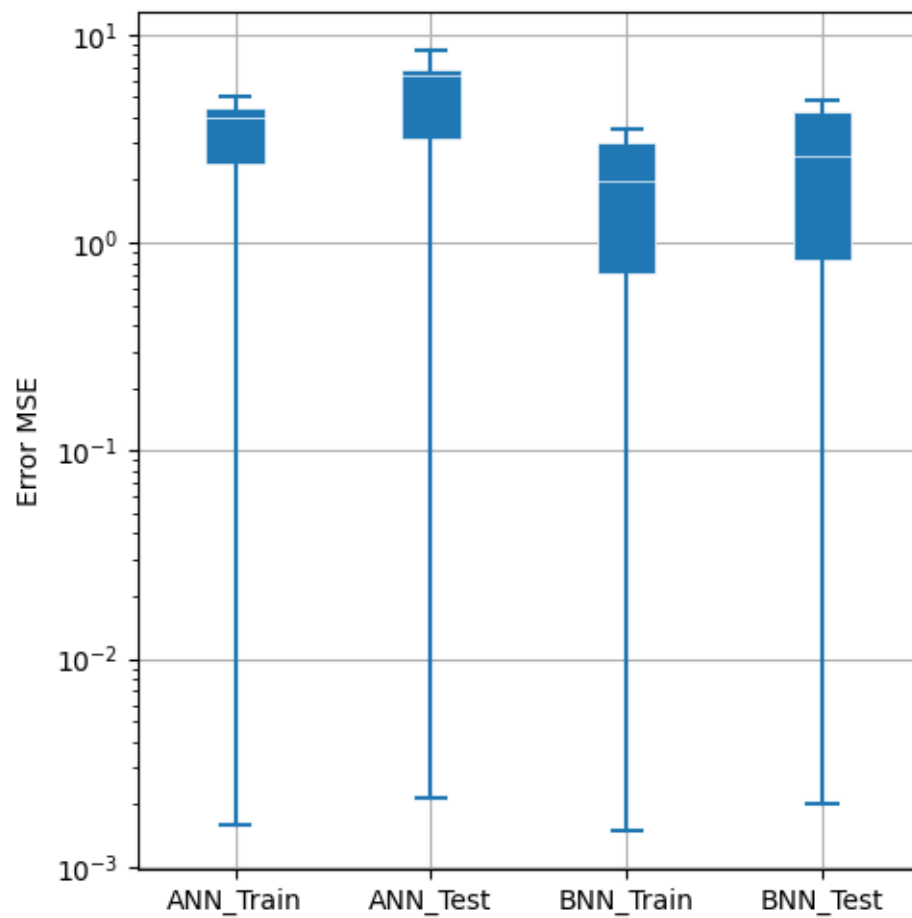


Figura 3.18 Diagrama de cajas de la red ANN y BNN para la placa sin fallo.

Las redes bayesianas son ligeramente más precisas en cuanto a las variables que predicen.

4 Conclusiones

Para todos los modelos generados en el presente trabajo de fin de grado se puede confirmar que las redes neuronales convencionales son más precisas que las redes neuronales bayesianas. La diferencia entre ambas es mínima así que no se considera una gran ventaja a favor de las redes convencionales.

En cambio, las redes bayesianas a costa de la pérdida precisión sí que traen ciertas ventajas muy notables:

- Robustez frente a bases de datos ruidosos: Se demuestra cómo las redes convencionales son menos precisas cuando se trabaja con ruido. Además, en un sistema como GENERIC se acentúa la acumulación de error porque cada iteración depende de la anterior, lo que provoca que en muchos casos la red ANN prediga valores inestables. En cambio, para las BNN, aunque se reduce la precisión siempre sigue la tendencia general de cada variable de estado.
- Cuantificación de la incertidumbre: Poder cuantificar la incertidumbre permite valorar la confianza en el modelo, y, aunque la red realice una predicción adecuada, si la incertidumbre es grande es posible que se descarte por no ser lo suficientemente fiable.

4.1 Futuras líneas de trabajo

Algunas líneas futuras de investigación son:

- Progresión a problemas 3D: Un problema con una dinámica más rica, además de suponer un avance en la simulación, experimentaría mayores variaciones que quizá, por las limitaciones numéricas de precisión, sería más fácil captar de cara al aprendizaje con una red neuronal bayesiana.
- Módulo de Young menos acusado: Sustituyendo esta discontinuidad por una mucho más suave podría ayudar a aprender de forma más eficiente los parámetros de la red.
- Ante variaciones pequeñas, se deberá explorar la normalización o tratamiento de datos necesario para poder aprender este tipo de problemas.
- Introducción de ruido en los problemas de SHM: La introducción de ruido en problemas de SHM permite asemejarse a problemas reales.



5 Glosario de abreviaturas

- TFG: Trabajo de Fin de Grado
- SPNN: *Structure-Preserving Neural Networks*
- IA: Inteligencia Artificial
- SHM: *Structural Health Monitoring*
- BNN: *Bayesian Neural Network*
- ANN: *Artificial Neural Network*
- KL: Kullback-Leibler
- lr : *Learning Rate* o tasa de aprendizaje
- GENERIC: *General Equation for the Non-Equilibrium ReversibleIrreversible Coupling*
- ELBO: *Evidence Lower BOund*
- ReLU: *Rectified Linear Unit*
- GT: *Ground Truth*
- MSE: *Mean squared Error*
- GD: Gemelo Digital
- GH: Gemelo Híbrido



6 Bibliografía

- [1] Hernández, Q., Badías, A., González, D., Chinesta, F., & Cueto, E. (2021). Structure-preserving neural networks. *Journal of Computational Physics*, 426, 109950.
- [2] <https://github.com/quercushernandez/StructurePreservingNN>
- [3] https://en.wikipedia.org/wiki/GENERIC_formalism
- [4] Miroslav Grmela and Hans Christian Öttinger. Dynamics and thermodynamics of complex fluids. i. development of a general formalism. *Physical Review E*, 56(6):6620, 1997.