Trabajo Fin de Grado
Ingeniería en Tecnologías Industriales

Gemelos digitales basados en modelos subrogados para mantenimiento predictivo de infraestructuras ferroviarias

Autor: Fernando Blanco Jiménez
Tutor: Antonio Romero Ordóñez

Dpto. de Mecánica de Medios Continuos y Teoría de Estructuras
Escuela Técnica Superior de Ingeniería

Sevilla, 2024

Trabajo Fin de Grado
Ingeniería en Tecnologías Industriales

Gemelos digitales basados en modelos subrogados para mantenimiento predictivo de infraestructuras ferroviarias

Autor:
Fernando Blanco Jiménez

Tutor:
Antonio Romero Ordóñez

Dpto. de Mecánica de Medios Continuos y Teoría de Estructuras
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2024

Trabajo Fin de Grado: Gemelos digitales basados en modelos subrogados para mantenimiento predictivo de infraestructuras ferroviarias

Autor: Fernando Blanco Jiménez

Tutor: Antonio Romero Ordóñez

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2024

El Secretario del Tribunal

A mi familia

A mis amigos

A mis maestros

Agradecimientos

En primer lugar, quería agradecer a Dios por haberme ayudado a ser capaz de afrontar con ganas e ilusión estos 4 años de carrera y especialmente este Trabajo de Fin de Grado. En segundo lugar, quiero felicitar y agradecer a mi tutor por su infinita paciencia y constancia en estos meses de trabajo. Me he considerado siempre un alumno con mucha suerte durante toda mi etapa académica, tanto por mis profesores, como por mi tutor como por mis compañeros de clase.

Por otro lado, quería agradecer a mi Familia, a mis padres. Javier e Inma, por su apoyo incondicional durante estos 4 años de estudio que han sido extremadamente bonitos y difíciles. Quiero agradecer también a mis hermanos, a Javi, a Pablo y a Luisito, por estar siempre a mi lado, por sacarme una sonrisa todos y cada uno de los días de mi vida. Muchas gracias Familia.

Además, quería agradecer también las risas y momentos vividos durante estos 4 años a mis Amigos. Son unos pocos los que han estado ahí, pero dicen que es mejor calidad que cantidad, y yo los he tenido de una calidad inigualable. Gracias a todos.

Por último, quería agradecer estos 4 años al Colegio Mayor Almonte, sitio donde he vivido durante toda mi etapa universitaria. No puedo estar más feliz de haber elegido Almonte como sitio en el que vivir. me han ayudado todos los días a ser mejor persona, a crecer profesional y humanamente, pero sobre todo, a crecer en mi relación con Dios. Muy agradecido a todas las personas que se han preocupado por mi día tras día de manera incansable. Gracias.

Con mucho cariño,

Fernando Blanco Jiménez

Resumen

En este Trabajo de Fin de Grado se realizará un modelo subrogado que posteriormente será calibrado a través de unas medidas experimentales. A modo general, se establecerá un contexto acerca del mantenimiento preventivo y los gemelos digitales que ayudarán a comprender mejor la finalidad de este trabajo. Se desarrollará en profundidad los conceptos de algoritmo genético y subrogación que son herramientas útiles para realizar procesos de optimización. Por otro lado, se realizará un caso práctico para implementar en la práctica, los conceptos estudiados de los métodos de optimización. En último lugar, se estudiará el caso de estudio de este trabajo: La estructura E9 – Línea 1 de Metro de Sevilla. Se llevará a cabo un proceso de optimización de las variables sensibles de dicha estructura con el objetivo de establecer un modelo apto para poder determinar los valores de dichas variables críticas de manera que se pueda garantizar la integridad de la estructura siempre y cuando se realicen ensayos experimentales con una cierta periodicidad.

Para la realización del trabajo, se exponen las definiciones de los métodos de optimización mediante algoritmos genéticos y subrogación, así como el funcionamiento externo e interno de los mismos. En el caso práctico de una viga simplemente apoyada que se presenta para contemplar dichos métodos, se llevan a cabo unos análisis analíticos y análisis mediante Elementos Finitos. Por último, se analizará la estructura de metro en la que se desarrollarán la definición de la misma, así como las pruebas de carga realizadas sobre esta. Se implementará un modelo de Elementos Finitos en ANSYS que será posteriormente una herramienta fundamental en el proceso de optimización llevado a cabo en MATLAB. De dicho proceso, se obtendrán una serie de resultados y se redactarán una serie de conclusiones.

Abstract

In this Bachelor's Thesis, a surrogate model will be created and subsequently calibrated through experimental measurements. Generally, a context will be established about preventive maintenance and digital twins to help better understand the purpose of this work. The concepts of genetic algorithms and surrogacy, which are useful tools for optimization processes, will be developed in depth. On the other hand, a practical case will be carried out to implement the studied concepts of optimization methods in practice. Finally, the case study of this work will be examined: The E9 structure – Line 1 of the Seville Metro. An optimization process of the sensitive variables of this structure will be carried out to establish a suitable model to determine the values of these critical variables, ensuring the integrity of the structure as long as experimental tests are carried out periodically.

To carry out the work, the definitions of optimization methods using genetic algorithms and surrogacy are presented, as well as their external and internal functioning. In the practical case of a simply supported beam presented to contemplate these methods, analytical analyses and analyses using Finite Elements are carried out. Finally, the metro structure will be analyzed, including its definition and the load tests performed on it. A Finite Element model will be implemented in ANSYS, which will later become a fundamental tool in the optimization process carried out in MATLAB. From this process, a series of results will be obtained, and a series of conclusions will be drafted.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN Y OBJETIVOS

Las estructuras de ingeniería civil están expuestas a peligros naturales que pueden causar daños estructurales o incluso el colapso. Un fallo estructural imprevista puede ser catastrófica no sólo en términos de vidas y pérdidas económicas, sino también en términos de impactos sociales posteriores. Por lo tanto, la detección de daños estructurales es importante, especialmente en el estado de daño temprano, para evitar fallas repentinas y mejorar la seguridad y confiabilidad de las estructuras.

Por esta razón, el mantenimiento preventivo de infraestructuras ferroviarias es fundamental para garantizar la seguridad, eficiencia y duración del sistema de transporte. Tradicionalmente, el mantenimiento de estas estructuras se ha basado en inspecciones periódicas y mantenimiento poco frecuente, lo cual puede resultar en tiempos de inactividad no planificados y costes elevados. En este sentido, la aplicación de gemelos digitales basados en modelos subrogados surge como una solución innovadora que permite una gestión continua y optimizada de las infraestructuras ferroviarias.

Un gemelo digital es una réplica virtual de una estructura física que utiliza datos en tiempo real y simulaciones avanzadas para reflejar el estado y comportamiento de la estructura a lo largo del tiempo. En el caso de las estructuras ferroviarias, los gemelos digitales permiten monitorizar continuamente el estado de las vías, puentes y otros componentes críticos, facilitando la detección temprana de fallos y la planificación eficiente del mantenimiento.

El objetivo de este trabajo de fin de grado es desarrollar un modelo subrogado capaz de ajustarse a los resultados experimentales disponibles, con el fin de determinar de manera aproximada y precisa el valor de una serie de variables significativas en el comportamiento de las estructuras ferroviarias. Un modelo subrogado es una versión simplificada y computacionalmente eficiente del modelo original, que mantiene la capacidad de realizar aproximaciones precisas. Este enfoque es especialmente útil en situaciones donde las simulaciones completas del modelo original son demasiado costosas o lentas.

Para la creación del modelo subrogado, se partirá de datos experimentales obtenidos de pruebas realizadas en infraestructuras ferroviarias (en este trabajo se tratarán como pruebas de carga). Estos datos incluyen medidas de deformación, vibración, temperatura, entre otros parámetros relevantes, que reflejan el comportamiento estructural bajo diversas condiciones operativas y ambientales (en este Trabajo de Fin de Grado se parte de medidas de desplazamiento únicamente). A través del uso de programas de cálculo avanzados se desarrollará un modelo que pueda aproximar con precisión estos resultados experimentales.

En un proceso de toma de datos, en la primera etapa del desarrollo se recopilarán datos de sensores instalados en las estructuras ferroviarias, los cuales proporcionarán información continua y en tiempo real sobre el estado de la infraestructura. Estos datos serán preprocesados para asegurar su calidad y consistencia, mediante la eliminación de datos erróneos o ruidosos, la normalización de escalas y la imputación de datos faltantes.

A continuación, se procede con el desarrollo del modelo subrogado mediante el uso de programas de cálculo. Este proceso incluirá la selección de las técnicas de modelización adecuadas, el entrenamiento del modelo con los datos preprocesados y la optimización de sus parámetros para minimizar el error entre las aproximaciones del modelo y los resultados experimentales. Se evalúa el modelo para asegurar precisión y fiabilidad y en caso de ser necesario, se realizarán ajustes adicionales para mejorar el rendimiento del modelo.

Una vez validado, el modelo subrogado se utilizará para realizar simulaciones que permitan analizar el comportamiento futuro de las estructuras ferroviarias. Estas simulaciones facilitarán la identificación de posibles fallos y puntos críticos antes de que ocurran. Los resultados de las simulaciones proporcionarán información valiosa para la toma de decisiones, optimizando las estrategias de mantenimiento y minimizando los tiempos de inactividad y costos asociados, como se ha comentado anteriormente.

El desarrollo de este modelo subrogado tiene múltiples beneficios para el mantenimiento de infraestructuras ferroviarias. Permite una evaluación continua y precisa del estado de las estructuras, reduce la necesidad de inspecciones físicas frecuentes, mejora la capacidad de predecir fallos y planificar el mantenimiento para evitar dichos fallos. Además, contribuye a la sostenibilidad del sistema de transporte ferroviario al optimizar el uso de recursos y prolongar la vida útil de las infraestructuras. En última instancia, este Trabajo de Fin de Grado busca demostrar entre otras cosas, que los gemelos digitales basados en modelos subrogados son una herramienta eficaz para el mantenimiento preventivo de estructuras ferroviarias, mejorando significativamente la gestión y estudio de estas infraestructuras críticas.

1.1 Mantenimiento preventivo y detección de daño

Antiguamente, el mantenimiento de infraestructuras se basaba principalmente en enfoques correctivos, donde se realizaban inspecciones regulares y se abordaban los problemas una vez surgían. Sin embargo, con el desarrollo de tecnologías de sensores más sofisticadas, análisis de datos avanzados y sistemas de monitorización en tiempo real, el mantenimiento preventivo ha ganado popularidad como una forma más eficiente y efectiva de gestionar activos críticos. En lugar de esperar a que ocurran problemas, el mantenimiento preventivo permite predecir y prevenir fallos y problemas antes de que afecten negativamente a la infraestructura y a la operación, así como a los costos elevados que requiere una reparación de un daño en una infraestructura.

De manera general (posteriormente entraremos un poco en profundidad) en la actualidad, el número de proyectos de I+D+i de nuevos sistemas de mantenimiento preventivo en infraestructuras crece cada año. Las técnicas de detección de daño se clasifican en:

1) Ensayos de auscultación no destructivos
2) Métodos de identificación basados en el análisis de la respuesta dinámica de la estructura [1]

El primer tipo son métodos locales que no permiten detectar fácilmente daños internos en la estructura. Por el contrario, el segundo método de identificación estudia cambios en el comportamiento dinámico de la estructura y se consideran métodos globales.

Esta línea se enmarca en el desarrollo de sistemas de detección de daño basados en el estudio del comportamiento dinámico de la estructura [2], abordándose la medida y la adquisición experimental de datos, y los métodos de detección de daño. Este tipo de método usa algoritmos de actualización de modelos en los que se modifican propiedades como la masa, la rigidez y el amortiguamiento, para detectar cambios en el comportamiento e identificar la presencia de daño, su posible ubicación, y cuantificar el alcance de éste.

La identificación de daño puede tratarse como un problema de reconocimiento de patrones que se divide en dos etapas [3]:

1) La adquisición de datos experimentales
2) La extracción y clasificación de características

El objetivo de la extracción de características es el ajuste del modelo de la estructura, identificando los parámetros que son más sensibles al daño. Posteriormente, con las características seleccionadas, se utiliza un algoritmo de clasificación con el fin de determinar la presencia, la ubicación y la severidad del daño. La mayoría de los algoritmos de detección de daño se basan en métodos de aprendizaje supervisado que requieren características de los estados dañados y no dañados de la estructura para establecer un modelo estadístico durante el proceso de identificación [4]. Los datos de la estructura dañada se pueden obtener a partir de pruebas de laboratorio o bien mediante simulaciones numéricas empleando gemelos digitales.

1) Detección de daños en infraestructuras

Por ende, con respecto a la detección de daños estructurales, Rytter [5] dividió la identificación de daños en cuatro etapas: determinación de la presencia de daños, localización precisa de los mismos, evaluación de la magnitud del daño y, por último, estimación de la vida útil restante de las estructuras, clasificados como Niveles 1 a 4, respectivamente. La mayoría de las investigaciones actuales se enfocan en los tres primeros niveles [6].

Como afirman C.R. Farrar, S.W. Doebling y D.A. Nix [2], en términos generales el daño puede definirse como los cambios introducidos en un sistema que afectan negativamente el rendimiento actual o futuro de ese sistema. Implícito en esta definición está el concepto de que el daño no es significativo sin una comparación entre dos estados diferentes del sistema, uno de los cuales se asume que representa el estado inicial, y a menudo sin daño. Esta discusión se centra en el estudio de la identificación de daños en sistemas estructurales y mecánicos. Por lo tanto, la definición de daño se limitará a los cambios en las propiedades materiales y geométricas de estos sistemas, incluidos los cambios en las condiciones de contorno y la conectividad del sistema, que afectan negativamente el rendimiento actual o futuro de los sistemas.

Los métodos de detección de daños basados en la respuesta dinámica (o basada en vibraciones) de la estructura se definen como un conjunto de métodos que utiliza la respuesta dinámica de una estructura para evaluar su estado y detectar daños. Estos métodos incluyen la detección basada en la frecuencia natural, la forma modal, la curvatura modal, la energía de deformación modal y la flexibilidad modal. Aunque principalmente se centra en estudios basados en experimentos, también se consideran algunos estudios numéricos que pueden motivar investigaciones adicionales.

H. Rongrong y X. Yong [1], así como C.R. Farrar, S.W. Doebling y D.A. Nix [2], presentan sus respectivos puntos de vista acerca de los métodos de detección de daños basados en la respuesta dinámica de una estructura, lo que nos permite comprender de manera más completa los avances realizados en las últimas décadas, la utilidad de la detección de daños y su potencial para llevar a cabo un mantenimiento preventivo efectivo. Entre estos, destacamos los siguientes puntos:

- Las nuevas investigaciones han impulsado el desarrollo y la implementación de tecnologías de sensores avanzadas. Se ha progresado desde los sistemas cableados y de fibra óptica hasta sensores inalámbricos y tecnologías sin contacto como cámaras y vibrómetros. Aunque los sensores inalámbricos enfrentan limitaciones debido al consumo de energía, los sistemas de fibra óptica y las técnicas basadas en visión han mostrado un gran potencial para aplicaciones futuras, especialmente cuando se combinan con tecnologías de inteligencia artificial.
- La integración de modelos FE con técnicas que combinan muchos datos de diferentes fuentes han permitido generar respuestas estructurales no observadas por los sistemas de monitorización. Además, el uso de algoritmos de aprendizaje automático (‘machine learning’), especialmente el aprendizaje profundo, ha mejorado significativamente la capacidad de identificar y clasificar características sensibles al daño. Estos algoritmos permiten simplificar grandes volúmenes de datos de monitoreo en índices de daño y representaciones gráficas útiles, destacando la evolución desde estrategias de aprendizaje no supervisado a supervisado.
- A pesar de los avances, existen desafíos significativos en la implementación de métodos de mantenimiento preventivo de infraestructuras (‘Structural Health Monitoring’, SHM) a gran escala, especialmente en estructuras grandes como puentes. Las estrategias de identificación de daños han pasado de enfoques globales a aplicaciones más localizadas para aumentar la rentabilidad. No obstante, la detección de daños basada en vibraciones y otras técnicas no destructivas aún enfrenta dificultades debido a la variabilidad ambiental y operativa. La comunidad de SHM sigue siendo ambiciosa en la búsqueda de métodos que puedan identificar daños ocultos en una escala más amplia, subrayando la necesidad de métodos robustos y eficientes para la toma de decisiones basadas en los datos obtenidos del monitoreo.

2) Problemas de reconocimiento de patrones para la identificación de daños

Como se ha dicho anteriormente, estos métodos de detección de daños se pueden abordar como problemas de reconocimiento de patrones, es decir, se recolectan una serie de datos experimentales obtenidos a base de pruebas y ensayos en la infraestructura de estudio mediante instrumentos de medida y posteriormente, examinamos y clasificamos las características de la propia estructura, determinando por ejemplo, los parámetros que son más sensibles al daño en la misma. Los métodos de reconocimiento de patrones pueden incluir técnicas de aprendizaje automático, como redes neuronales [7], clasificación estadística, análisis de datos de vibración, entre otros. Esta aproximación permite una detección más automatizada y sistemática de daños, lo que puede ser especialmente útil en la monitorización continua de grandes estructuras o en situaciones donde la inspección manual es difícil o costosa.

E. Figueiredo y J. Brownjohn [3] explican en uno de sus artículos las siguientes afirmaciones acerca de las técnicas de reconocimiento de patrones e instrumentos utilizados para ello:

- La evolución de la evaluación operativa ha pasado de estrategias de identificación de daños globales a aplicaciones más localizadas, aumentando así la tasa de retorno de la inversión. Sin embargo, sigue existiendo un fuerte interés en la comunidad de SHM por identificar daños basados en mediciones de campo completo en un enfoque global. La identificación y el uso efectivo de los beneficios obtenidos a través de sistemas SHM son cruciales para el apoyo en la toma de decisiones.
- Se han observado desarrollos significativos en la adquisición de datos, desde sensores cableados, de fibra óptica e inalámbricos hasta sensores sin contacto como cámaras y vibrómetros. Cada tipo de sensor presenta ventajas y limitaciones específicas, con una tendencia hacia tecnologías más avanzadas como la detección distribuida de fibra óptica y técnicas basadas en visión, que ofrecen una mayor cobertura y resolución. La integración de estas tecnologías con inteligencia artificial promete mejorar aún más el monitoreo y la detección de daños.

3) Redes neuronales como técnica para reconocimiento de patrones

En cuanto a las redes neuronales, podemos decir que son, en el contexto de reconocimiento de patrones, algoritmos que pueden aprender y reconocer patrones complejos en conjuntos de datos, lo que las convierte en una herramienta poderosa para la clasificación, la predicción y la detección de anomalías en diversas aplicaciones.

Como mencionan N. Bakhary, H. Hao y A. Deeks. [4], las redes neuronales (‘Artificial Neural Network’, ANN) junto con la aplicación de una técnica de subestructuración (que consiste básicamente en dividir una estructura completa en subestructuras más pequeñas y analizar cada una de manera independiente, de manera que permite una identificación más efectiva de daños locales en la estructura, ya que se pueden detectar cambios específicos en el comportamiento de cada subestructura) para la identificación de daños en una infraestructura, muestran las siguientes evidencias analizadas:

La técnica propuesta, que emplea subestructuración junto con un ANN multietapa, demuestra ser más eficiente y fiable en la detección de daños locales en estructuras en comparación con técnicas convencionales. Al dividir la estructura completa en subestructuras y analizar cada una de manera independiente, se puede identificar mejor el daño local. Por otro lado, el uso de un modelo ANN de una sola etapa para la detección de daños en estructuras grandes requiere un tiempo de computación excesivo y una gran cantidad de memoria. La técnica propuesta reduce significativamente el tamaño de los modelos ANN necesarios, disminuyendo así el esfuerzo computacional. Esto hace que la aproximación sea más práctica para aplicaciones a gran escala.

1.2 Gemelo digital

En términos globales, el gemelo digital ('Digital Twins', DT) es una especie de union entre el mundo físico y el mundo digital. Dicho de otra forma, un gemelo digital es un modelo virtual de un sistema físico. A este sistema físico se le denomina como gemelo físico. Los intentos de aproximar o simular el comportamiento de los sistemas físicos son un componente clave de la ingeniería y la ciencia. Sin embargo, la diferencia entre un gemelo digital y un modelo es que el gemelo digital se actualiza a sí mismo para seguir al gemelo físico mediante el uso de sensores, análisis de datos, aprendizaje automático (o 'machine learning') y el Internet de las cosas. Idealmente, los gemelos físicos y digitales deben lograr una sincronización temporal.

En este Trabajo de Fin de Grado, el gemelo físico sería la infraestructura correspondiente de estudio que se mostrará en el apartado 4.1, y el gemelo digital es el modelo que realiza la simulación de la estructura, se mostrará en el apartado 4.3.

Tal y como desarrollan S. Chakraborty, S. Adhikari y R. Ganguli [8], vemos a continuación cómo los gemelos digitales permiten el seguimiento y predicción de la evolución de sistemas físicos, así como el objetivo de su implementación y su historia y evolución. Por último, comentaremos sus aplicaciones en la industria actual.

El gemelo digital de un sistema sigue su evolución a lo largo del tiempo. El sistema real y su gemelo digital están sujetos a dos escalas de tiempo. La primera escala de tiempo es de tiempo instantáneo, y la segunda escala de tiempo es de tiempo lento, que captura la evolución de las propiedades del sistema real. Por ejemplo, en los diagnósticos de turbinas de gas, la eficiencia de los módulos se degrada lentamente a medida que el motor entra en servicio, siendo un cambio lineal una buena aproximación. El crecimiento del daño en los componentes ocurre con el tiempo debido a la fatiga y puede representarse como un ajuste de curva. Este ajuste de curva captura los modos de daño físico de fisuración de matriz, delaminación y rotura de fibras.

La conectividad generalizada de los sistemas físicos a Internet permite el seguimiento de millones de gemelos físicos en tiempo real. Sin embargo, el objetivo es asegurar que el gemelo digital imite lo máximo posible al sistema físico con el paso del tiempo. En otras palabras, el gemelo digital debe evolucionar con el gemelo físico a lo largo de su ciclo de vida. Al final de la vida útil, tanto los gemelos digitales como los físicos terminarán. Los problemas importantes en el desarrollo de la tecnología de gemelos digitales están relacionados con la modelización y simulación del gemelo físico, la precisión y velocidad de la transmisión de datos entre el gemelo físico y donde reside el gemelo digital, el almacenamiento y procesamiento de los grandes volúmenes de datos creados por los sensores a lo largo del ciclo de vida, el procesamiento informático eficiente del modelo de gemelo digital, y posiblemente la activación del gemelo físico mediante señales enviadas por el gemelo digital a los actuadores del gemelo físico.

Varios autores han considerado el concepto de gemelo digital en las últimas dos décadas. Un artículo reciente sobre gemelos digitales en el contexto de la predicción de vida útil de estructuras de aeronaves fue publicado por E. J. Tuegel, A. R. Ingraffea, T. G. Eason y S. M. Spottswood [9]. Su objetivo era utilizar un modelo de ultra alta fiabilidad de una aeronave individual identificada por su número de cola para predecir su vida estructural. Esta investigación fue motivada por el creciente poder de las computadoras y la evolución de la computación de alto rendimiento. La idea clave introducida en este artículo fue la necesidad de rastrear cada aeronave individual por su número de cola, que es un identificador único. Así, cada aeronave física en la flota tendría un gemelo digital separado, que evolucionará de manera diferente en comparación con los otros gemelos, ya que cada aeronave enfrenta combinaciones únicas de misiones, cargas, ambiente, comportamiento del piloto, situación de los sensores, etc. Este concepto de gemelos digitales se amplió posteriormente a cualquier sistema físico, como automóviles, servidores informáticos, locomotoras, turbinas, herramientas de máquinas, etc. Las aplicaciones principales de los gemelos digitales han sido en producción, diseño de productos, y pronóstico y gestión de la salud.

La tecnología de gemelos digitales ha encontrado muchas aplicaciones industriales. El gemelo digital es una tecnología habilitadora para realizar la manufactura inteligente y la Industria 4.0. En la Industria 4.0, se imaginan fábricas con conectividad inalámbrica y sensores que permiten que la línea de producción funcione automáticamente. A medida que los sistemas físicos se enriquecen con sensores, la tecnología de transmisión de datos permite la recolección de datos a lo largo de la etapa de vida del sistema físico. Este conjunto de datos

es típicamente muy grande, y se necesitan análisis de 'big data' para encontrar causas de fallos, optimizar cadenas de suministro y mejorar la eficiencia de la producción. Los gemelos digitales también se han utilizado en pronóstico y gestión del mantenimiento de sistemas ciberfísicos y procesos de manufactura aditiva. La fusión de los sistemas físicos y cibernéticos sigue siendo el desafío clave para el uso de los gemelos digitales, y un enfoque para abordar este problema es hacer que las tareas de modelización y simulación sean más eficientes. La modelización por sustitución podría ser un enfoque para aumentar la eficiencia del proceso de simulación.

1.3 Métodos de optimización (Algoritmo Genético y Subrogación)

Teniendo en cuenta que ya se ha definido previamente, un gemelo digital puede ser objeto de optimización porque, al ser una representación virtual precisa y en tiempo real de un objeto, sistema o proceso físico, permite experimentar y ajustar diversos parámetros sin afectar directamente a su contraparte física. Por esta razón, a modo de ejemplo, en la industria de la manufactura, un gemelo digital de una máquina puede utilizar algoritmos genéticos para optimizar parámetros de operación, reducir el desgaste, mejorar la eficiencia y minimizar los tiempos de inactividad. Al igual que los algoritmos genéticos, en la simulación de un avión, un modelo subrogado puede aproximar el comportamiento aerodinámico sin necesidad de realizar simulaciones computacionalmente intensivas para cada iteración.

1.3.1 Algoritmo genético

Los algoritmos genéticos son un método de búsqueda y optimización inspirado en los principios de la evolución biológica, como la selección natural de Charles Darwin y la genética. Utiliza técnicas como la selección, el cruce y la mutación para generar soluciones progresivamente mejores a un problema determinado. Es decir, en cada sucesiva generación la población evoluciona hacia una solución óptima.

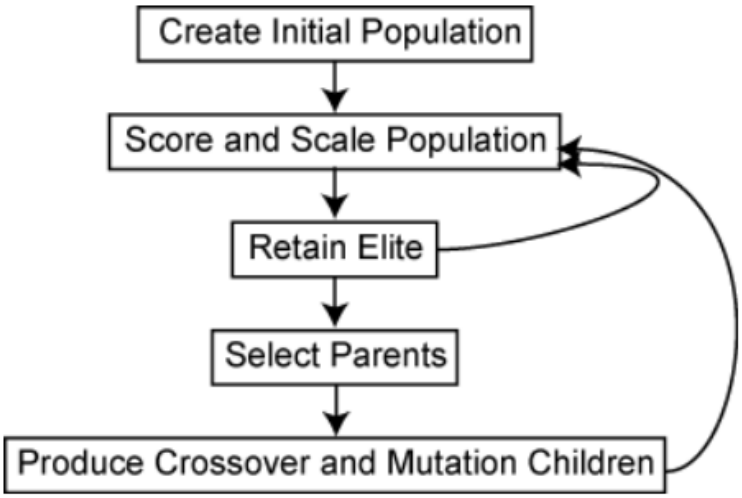

Figura 1.1 Proceso de creación de generaciones de los algoritmos genéticos.

En la figura 1.1 se muestra el proceso que sigue el algoritmo genético cuando se utiliza como herramienta principal en un problema de optimización (explicaremos en detalle dicho proceso posteriormente). Sin embargo, de manera general, el proceso interno que forma el algoritmo genético es el siguiente: En primer lugar, crea una población inicial aleatoria, en segundo lugar, puntúa y escala dicha población. Por consiguiente, entre los puntos mejor valorados se seleccionan a los 'padres', y a raíz de ellos, se realizan los

cruzamientos y las mutaciones para generar los 'hijos'. Los puntos que no fueron seleccionados como 'padres' así como los hijos producidos por las mutaciones y los cruzamientos, pasan a formar parte de la nueva población, la 'siguiente generación'.

1.3.2 Subrogación

La subrogación se refiere a la creación de modelos simplificados o aproximados que reemplazan a modelos más complejos y computacionalmente intensivos. Estos modelos subrogados permiten realizar simulaciones y análisis de manera más rápida y eficiente, sin que se vea comprometida significativamente la precisión de los resultados. En otras palabras, es una función que aproxima otra función y emplea poco tiempo de cálculo. Para encontrar un mínimo, evalúa cientos de puntos y toma el mejor valor como aproximación del minimizador de la función objetivo. Es especialmente útil para funciones objetivo que requieren mucho tiempo de evaluación.

El algoritmo de subrogación trata de encontrar un mínimo global empleando pocas iteraciones. A su vez, trata de equilibrar el proceso de optimización entre 2 objetivos principales:

1. Exploración, para buscar un mínimo global.
2. Velocidad, para obtener una buena solución en pocas evaluaciones de la función objetivo.

Por lo general, el criterio de parada del algoritmo suele ser por máximo número de funciones evaluadas ('MaxFunctionEvaluations') o por alguna de las opciones propias del algoritmo que se explicarán más en mayor medida en el apartado 4.3.

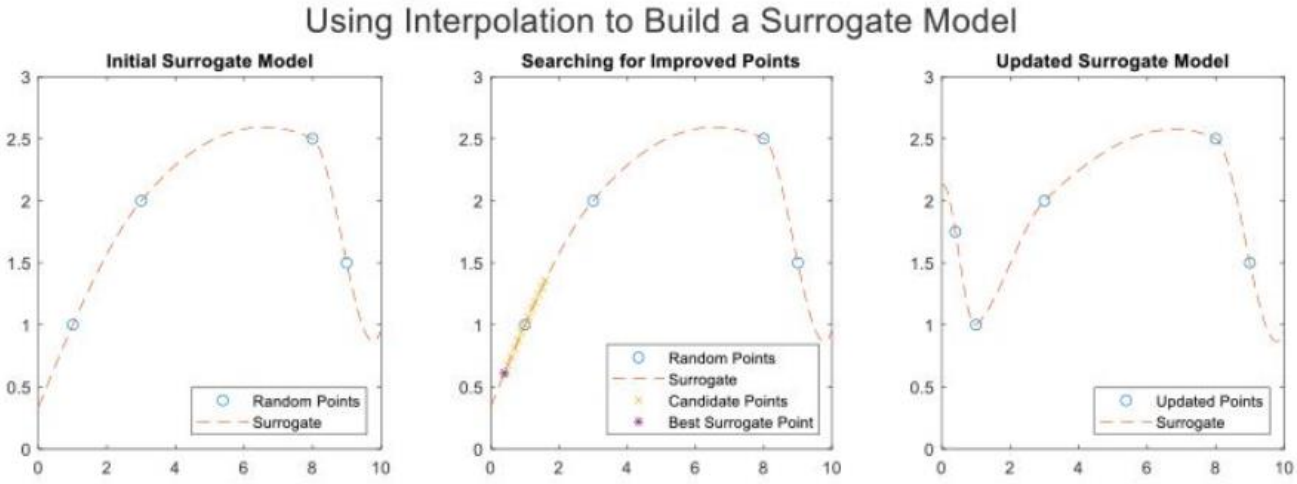

Figura 2.2 Proceso de selección de candidatos en el método de subrogación.

En la figura 1.2 se puede observar el proceso general que sigue el algoritmo de subrogación (que posteriormente comentaremos en mayor detalle) en el que se generan una serie de puntos aleatorios cerca de uno de los puntos del modelo subrogado inicial, que posteriormente son evaluados en la función objetivo y actualizan el modelo subrogado en función del valor que se obtiene.

1.4 Objetivos y contribuciones originales

En este apartado veremos una serie de aspectos que destacan especialmente en este Trabajo de Fin de Grado:

- Se aportarán una serie de pruebas de carga que se realizan sobre la estructura de metro a partir de las cuales podemos realizar la optimización. Estas pruebas de carga constan de 3 pruebas individuales. De dichas pruebas de carga extraemos una serie de resultados experimentales en desplazamientos, que serán la raíz a partir de la cual construiremos nuestro modelo subrogado (o modelo sustituto).
- Se expondrá el funcionamiento de los métodos de optimización (algoritmos genéticos y subrogación) de manera teórica. Servirá de base para una posterior aplicación práctica en un caso sencillo de estudio.
- La aplicación de los métodos de optimización en un caso simple de estudio de una viga simplemente apoyada que nos ayudará a comprender, una vez realizado el análisis, las ventajas e inconvenientes de ambos métodos de optimización. Realizaremos un estudio tanto analítico como digital, es decir, a través de Elementos Finitos.
- El desarrollo de un gemelo digital basado en subrogación. Es decir, a partir del modelo digital que desarrollamos en ANSYS y mediante el uso de MATLAB, procederemos a realizar un proceso de optimización a través del método de subrogación. En dicho método de optimización estudiaremos las variables críticas de la estructura y trataremos de minimizar la función objetivo con el propósito de obtener unos resultados precisos y que se ajusten a la realidad. Haremos uso también de dicho gemelo digital para realizar un proceso de optimización mediante algoritmos genéticos. Se partirá de un modelo de Elementos Finitos que simula el comportamiento de la estructura de metro de Sevilla que veremos posteriormente. Este modelo nos permite estudiar adecuadamente el comportamiento de esta con el objetivo de estudiar los parámetros críticos, es decir, los más sensibles al daño, con el objetivo de determinar su valor a partir de unos resultados experimentales extraídos de una serie de ensayos.
- La calibración del modelo digital mediante el cálculo de las variables críticas de la estructura (módulos de elasticidad tanto del acero como del hormigón, resistencias a tracción y compresión) a través de un análisis estático de la estructura con el propósito de ajustar el comportamiento del modelo digital al comportamiento de la estructura en la vida real.
- En último lugar, se redactarán una serie de conclusiones después de analizar con detalle el caso de estudio: Estructura E9-Línea 1 Metro de Sevilla. Estas conclusiones nos pueden ayudar a tener una idea muy completa del objetivo de este trabajo, así como de los conceptos desarrollados y el análisis realizado.

2 MÉTODOS DE OPTIMIZACIÓN

En este apartado se expondrá el funcionamiento interno de los métodos de optimización que emplearemos en este Trabajo de Fin de Grado, optimización mediante Algoritmos Genéticos (Genetic Algorithm, GA) y optimización mediante Subrogación.

2.1 Algoritmo Genético

Definimos primeramente el problema de optimización que queremos resolver. Tenemos una función objetivo que queremos minimizar, unos determinados límites que irán asociados a una serie de variables.

La función objetivo que vamos a minimizar es la siguiente:

$$Fobj = (Fobj + sqrt(Phi' * Phi) * iPrueba^{-1})/(sqrt(2 * nPost) * lCase)$$

A continuación, se detalla el funcionamiento básico de un algoritmo genético.

1) Creación de la población inicial

El proceso comienza con la creación de una población inicial aleatoria. Esta población inicial es un conjunto de posibles soluciones al problema que se está intentando resolver. Cada solución en la población se representa como un individuo, y cada individuo se caracteriza por un conjunto de parámetros, conocidos como genes. Se muestra en la Figura 2.1 una población inicial aleatoria.

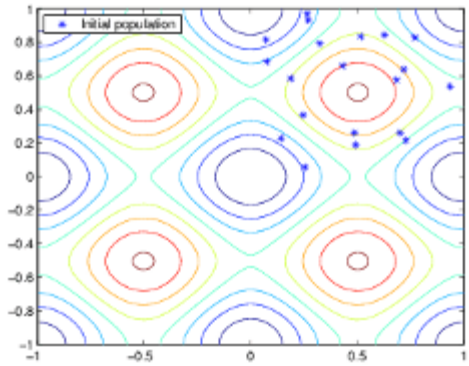

Figura 2.1 Población inicial aleatoria en el proceso de optimización mediante GA.

2) Evaluación de la población

Una vez creada la población inicial, el algoritmo evalúa a cada individuo para determinar su aptitud (‘fitness value’). La aptitud de un individuo es una medida de cómo de buena es la solución que representa para el problema en cuestión. Este valor se conoce como puntuación de aptitud bruta (‘raw fitness scores’).

3) Escalamiento de la puntuación de aptitud

La puntuación de aptitud bruta se escala para convertirla en un rango de valores más utilizable. Estos valores escalados, llamados valores esperados (‘expectation values’), se utilizan para determinar la probabilidad de selección de cada individuo para la reproducción.

4) Selección de los ‘padres’ (‘parents’)

Basándose en los valores esperados, se seleccionan individuos de la población actual para actuar como ‘padres’. La selección favorece a los individuos con mayor aptitud, asegurando que las características favorables se transmitan a la siguiente generación.

5) Selección de los élite (‘elite’)

Algunos de los individuos con mayor aptitud de la población actual se seleccionan como élite y se trasladan directamente a la siguiente generación. Esto asegura que las mejores soluciones encontradas hasta el momento no se pierdan.

6) Generación de los ‘hijos’ (‘children’) a partir de los ‘padres’

A partir de los ‘padres’ seleccionados, el algoritmo genera nuevos individuos, llamados hijos. Los hijos se producen mediante dos procesos principales: mutación y cruce.

1. Mutación: Se realizan cambios aleatorios en un solo padre para producir un hijo. Esto introduce diversidad en la población y ayuda a explorar nuevas áreas del espacio de búsqueda. Es como cambiar la forma o geometría o cualquier variable similar.
2. Cruce: Se combinan las entradas de los vectores de un par de padres para crear un hijo. Esto permite que las características favorables de diferentes individuos se combinen en la descendencia. Es como combinar una parte de uno de los ‘padres’ y otra parte del otro ‘padre’.

En la Figura 2.2 se puede apreciar la mutación y el cruce, así como la nueva generación de individuos que conforman la nueva población.

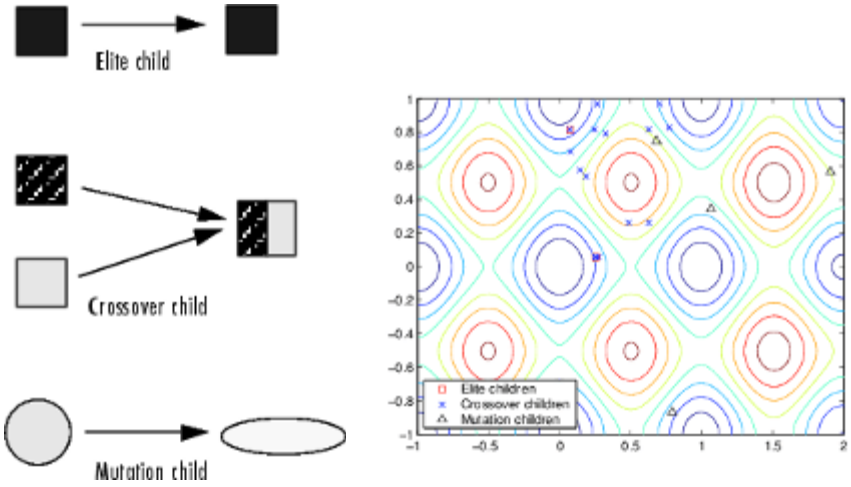

Figura 2.2 Generación de nuevos individuos (mutaciones y cruces) y nueva generación.

7) Creación de la nueva generación

La población actual se reemplaza con los nuevos individuos generados para formar la siguiente generación. Este proceso de evaluación, selección y generación de nuevos individuos se repite en cada generación, ver Figura 2.3.

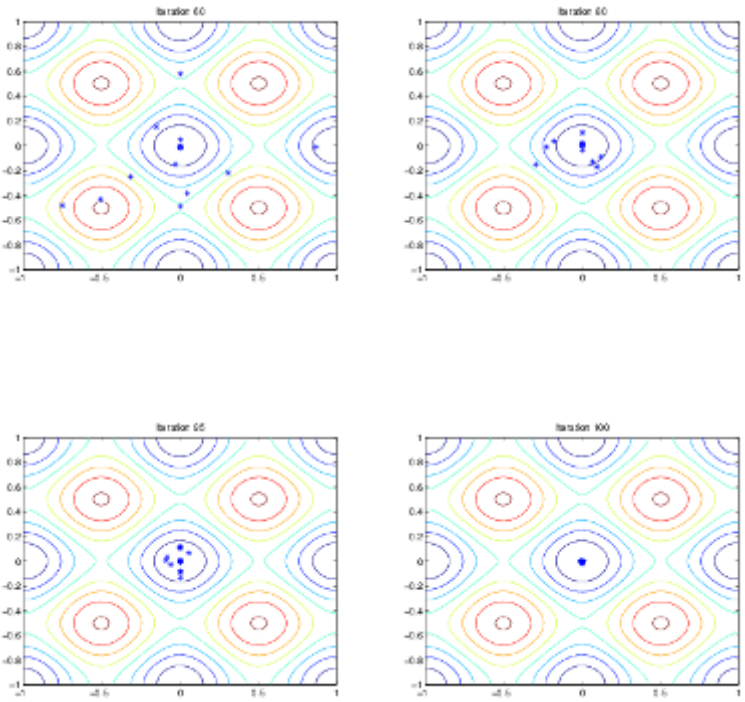

Figura 2.3 Proceso de convergencia del algoritmo para encontrar la solución óptima.

8) Criterios de parada

El algoritmo continúa iterando a través de las generaciones hasta que se cumple uno de los criterios de parada predefinidos. Estos criterios pueden incluir un número máximo de generaciones, una mejora mínima en la aptitud de las soluciones, o un tiempo de ejecución límite. El algoritmo incluye varias opciones modificables, lo que permite detener el algoritmo cuando se alcanza un valor específico en cualquiera de estas opciones de parada.

9) Adaptaciones para restricciones

Los algoritmos genéticos pueden adaptarse para manejar diferentes tipos de restricciones, en primer lugar, las lineales y enteras, y en segundo lugar, las no lineales:

1. Restricciones lineales y enteras: Se aplican modificaciones al proceso de selección y generación de nuevos individuos para asegurar que se respeten las restricciones lineales y enteras del problema.
2. Restricciones no lineales: Se utilizan algoritmos especializados para manejar restricciones no lineales, asegurando que las soluciones generadas sean viables dentro del espacio de búsqueda definido por estas restricciones.

En último lugar, presentamos un ejemplo típico, una población inicial contiene 20 individuos, todos ubicados en un rango específico del espacio de búsqueda. Si se sabe aproximadamente dónde se encuentra el punto mínimo de una función, se puede ajustar el rango inicial para que incluya ese punto. En cada generación, el algoritmo selecciona ‘padres’ de la población actual y genera ‘hijos’ mediante cruce y mutación. Los individuos ‘élite’ de la generación anterior se trasladan a la siguiente generación. Este proceso se repite hasta que se cumple un criterio de parada.

Es importante tener en cuenta que se garantice una cierta diversidad puesto que una diversidad considerable aumenta la probabilidad de que el algoritmo encuentre buenas soluciones. A continuación definiremos dicho concepto.

La diversidad se refiere a la distancia promedio entre los individuos de una población. Una población tiene alta diversidad si la distancia promedio es grande; de lo contrario, tiene baja diversidad. En la figura siguiente, la población de la izquierda tiene alta diversidad, mientras que la población de la derecha tiene baja diversidad.

La diversidad es fundamental para el algoritmo genético porque permite al algoritmo explorar una región más amplia del espacio. Esto significa que con una mayor diversidad, el algoritmo tiene más oportunidades de encontrar soluciones óptimas al problema, ya que no se concentra en una única área del espacio de búsqueda.

Mantener una alta diversidad dentro de la población puede lograrse mediante varias técnicas, como la implementación de mutaciones más frecuentes o intensas, o utilizando métodos de selección que favorezcan la variedad genética en lugar de concentrarse únicamente en los individuos con las mejores aptitudes. Por ejemplo, en lugar de seleccionar siempre a los individuos más aptos como ‘padres’, se podría implementar un enfoque de selección por ‘torneo’, donde se elige un subconjunto aleatorio de la población y se selecciona al mejor de ese grupo, lo que introduce variabilidad adicional en el proceso de selección.

En la siguiente Figura 2.4 la población de la izquierda tiene alta diversidad, mientras que la población de la derecha tiene baja diversidad.

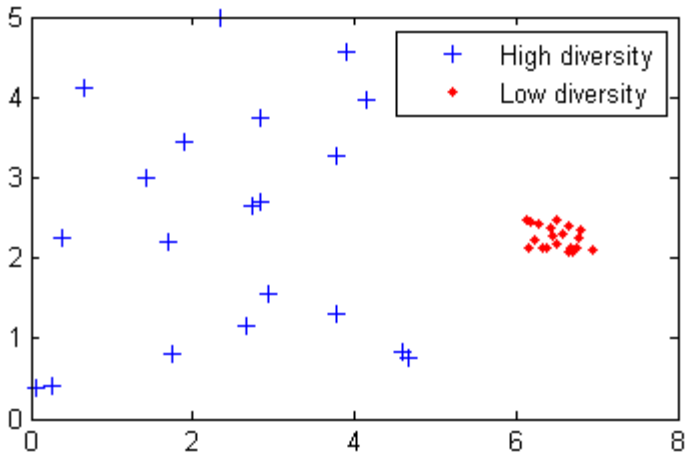

Figura 2.4 Alta diversidad y baja diversidad en una población.

A modo de conclusión, este es el proceso interno que sigue un algoritmo genético para encontrar una solución óptima, a través de la generación de nuevas poblaciones mediante selección de individuos creados a partir de mutaciones y cruces. Este método, por lo general, es una gran opción para resolver problemas de optimización. Sin embargo, el coste de computación que requiere es elevado.

2.2 Subrogación

Los modelos sustitutos son aproximaciones simplificadas de modelos más complejos y de orden superior. Se utilizan para asignar datos de entrada a salidas cuando la relación real entre los dos es desconocida o su evaluación es costosa desde el punto de vista computacional. Relacionan estadísticamente estos datos de entrada con los de salida, que se recopilan ejecutando la complicada simulación del sistema tal y como explican S. Davis et al.[10].

Los modelos clásicos de caja negra (black-box model) son modelos que relacionan entradas con salidas, pero sin exponer el funcionamiento interno del mismo (tratan de minimizar una función objetivo sin comprender exactamente lo que se está haciendo) y son modelos computacionalmente costosos. Los modelos subrogados a diferencia de los modelos de caja negra, tratan de realizar una aproximación explícita de la función objetivo y tratan de minimizarla para facilitar el proceso de optimización, y además son computacionalmente más eficientes y rápidos.

2.2.1 Funcionamiento externo de la optimización mediante subrogación

La base de los modelos subrogados reside en una técnica de interpolación denominada modelo de Kigring. En el siguiente artículo de Xiuming Yang, Xinglin Guo, Huajiang Ouyang y Dongsheng Li [11], se explica profundamente el funcionamiento de dicho modelo. Explicamos brevemente el proceso de construcción de este modelo para tener una idea general:

El modelo Kriging es una técnica de interpolación basada en un proceso estocástico [12]. En la teoría de la probabilidad, un proceso estocástico es un concepto matemático que sirve para representar magnitudes aleatorias que varían con el tiempo o para caracterizar una sucesión de variables aleatorias (estocásticas) que evolucionan en función de otra variable, generalmente el tiempo. Cada una de las variables aleatorias del

proceso tiene su propia función de distribución de probabilidad y pueden o no estar correlacionadas entre sí. Este modelo se utiliza comúnmente como base para los modelos subrogados debido a su capacidad para manejar datos no lineales y proporcionar predicciones precisas.

El proceso de iteración que utiliza el modelo de Kigring es el siguiente (Figura 2.5), se puede resumir en los siguientes:

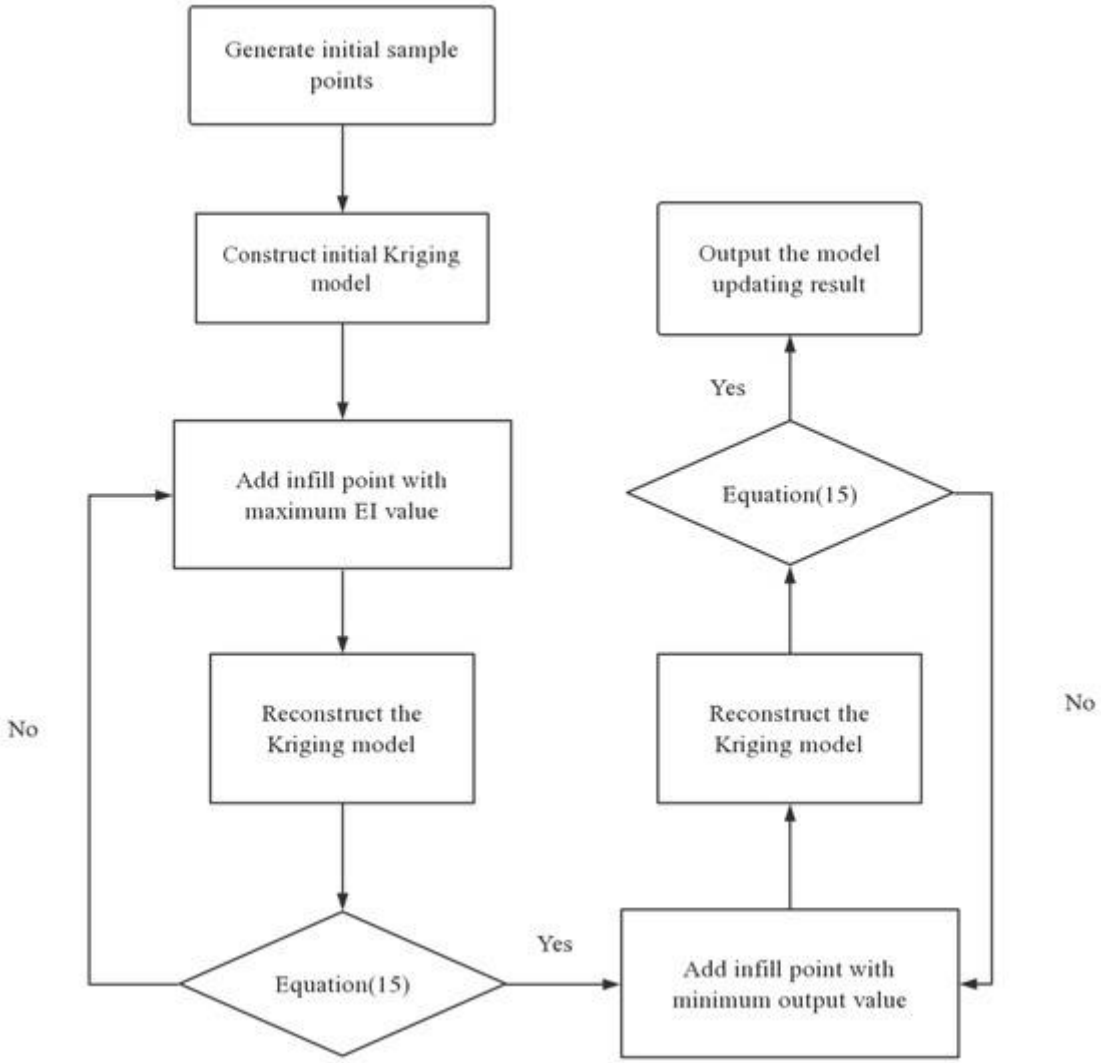

Figura 2.5 Proceso de optimización del modelo de Kigring.

1) Generar puntos de muestra iniciales de los parámetros de actualización mediante DOE (‘Design of Experiments’, diseño de experimentos).
2) Ejecutar el programa de análisis FE para calcular el vector de salida de la función objetivo de los puntos de muestra y construir el modelo Kriging inicial.
3) Encontrar el punto con el valor máximo de ‘mejora esperada’ (‘expected improvement’, EI) del modelo Kriging actual y añadirlo al conjunto de puntos de muestra.
4) Calcular la respuesta verdadera del nuevo punto de muestra y reconstruir el modelo Kriging con el nuevo conjunto de puntos de muestra y sus respuestas.
5) Verificar si se cumple el primer criterio de terminación (ecuación (15)). Si se cumple, se va a 6). De lo contrario, volver a 3) y continuar añadiendo nuevos puntos.

6) Encontrar el punto con el valor mínimo del modelo Kriging actual y añadirlo al conjunto de puntos de muestra, calcular la respuesta verdadera y reconstruir el modelo Kriging.
7) Verificar si se cumple el segundo criterio de terminación (ecuación (16)). Si se cumple, detener la actualización del modelo Kriging y usar la muestra correspondiente al valor mínimo en el vector de salida como el resultado de actualización correcto. De lo contrario, volver a 4) y continuar añadiendo nuevos puntos.

siendo,

Ecuación (15):

$$\max\big(EI(x)\big) < \varepsilon_1$$

Ecuación (16):

$$\| x_{new} - x_{min} \| \leq \varepsilon_2$$

en las que:

- EI es la 'mejora esperada' que proviene de la función de densidad de probabilidad Gaussiana.
- ε_1 es un pequeño número positivo que controla la precisión y velocidad de convergencia del algoritmo
- $\| x_{new} - x_{mín} \|$ es la distancia euclídea entre el nuevo punto de muestra x_{new} y $x_{mín}$ cuya respuesta de salida es mínima entre los puntos de muestra actuales
- ε_2 denota la tolerancia para la terminación de la iteración

Este modelo de Kigring verdaderamente es lo que hay en el interior del algoritmo de interpolación del modelo de subrogación. Un modelo subrogado está formado principalmente por 2 cosas:

1. Las entradas X y las salidas Y del modelo de Elementos Finitos (FEM).
2. El método de interpolación de Kigring.

En la siguiente figura se muestra un modelo subrogado, formado por los puntos [X,Y] y el método de interpolación de Kigring.

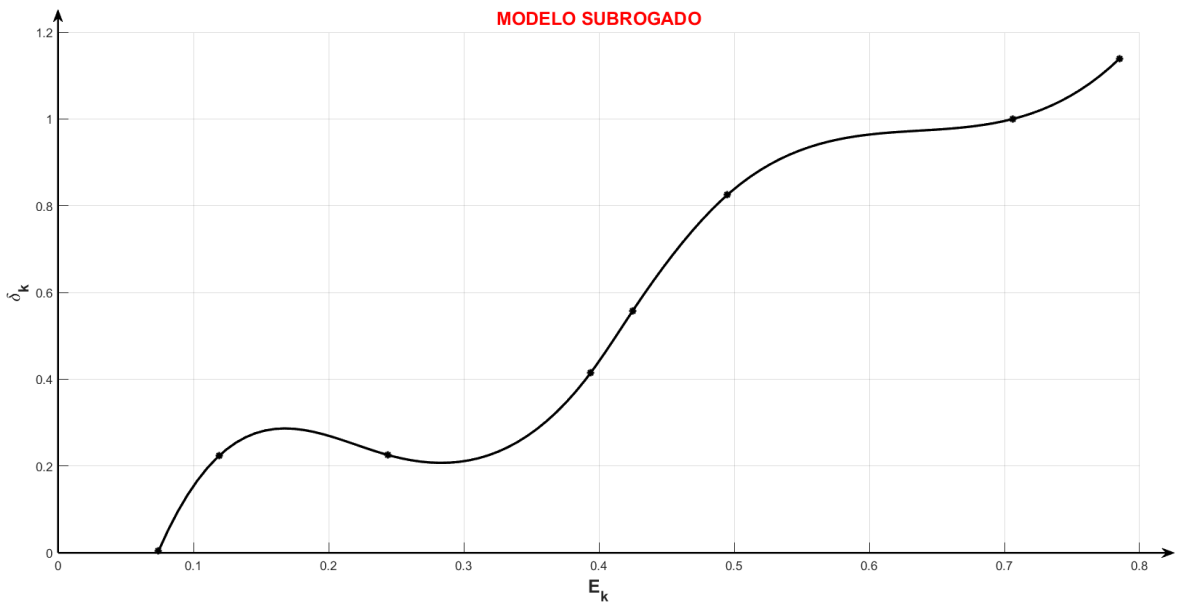

Figura 2.6 Construcción de un modelo subrogado mediante MEF y Kigring.

Se puede observar en la Figura 2.6 los puntos obtenidos mediante Elementos Finitos y cómo el método de Kigring anteriormente mencionado, interpola dichos puntos para construir el modelo subrogado con el que trataremos de optimizar las variables críticas de la estructura de estudio.

Se expone ahora en la Figura 2.7, el funcionamiento externo de un proceso de optimización (calibración) mediante un modelo subrogado. La ecuación (1) se muestra en el punto número 3:

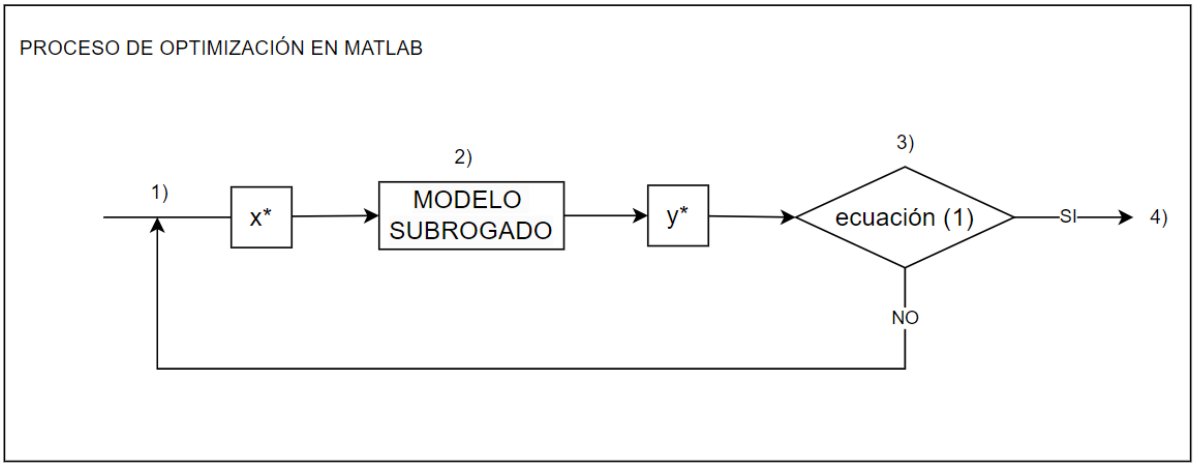

Figura 2.7 Proceso de optimización llevado a cabo en MATLAB.

En este diagrama se aprecia el proceso de optimización que se realiza en MATLAB cuando queremos calibrar un modelo. Los pasos a seguir son los siguientes:

1. El algoritmo de subrogación propone unos datos de entrada x*. Estos datos de entrada x* son los mejores candidatos de todos los evaluados internamente en el algoritmo. Con estos datos, se entra al modelo de subrogación.

2. El modelo subrogado tiene la siguiente forma (Figura 2.8). Se explicará después el modelo de Elementos Finitos en el apartado 4.2. Se aportan unos valores de entrada x* y el modelo de EF devuelve unos resultados en forma de desplazamientos y* (δ_k), que posteriormente evaluaremos en el paso 3).

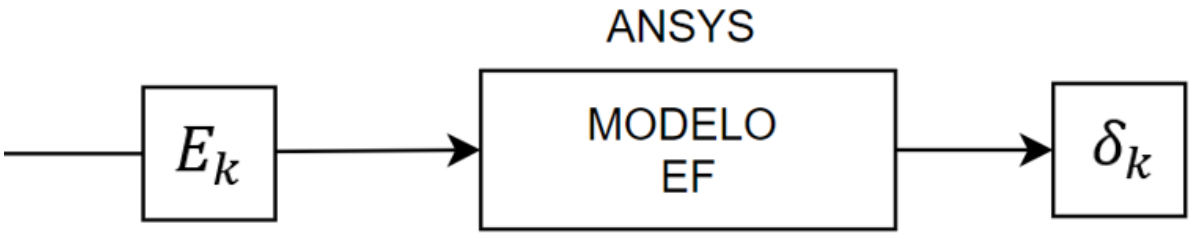

Figura 2.8 Funcionamiento externo del modelo subrogado.

3. En este paso se comparan los resultados obtenidos del modelo subrogado con los resultados experimentales (y^{exp}) obtenidos de los ensayos realizados en la estructura (que se explican en el apartado 4.1). La ecuación (1) es la siguiente:

   Ecuación (1):

   $$¿\ y^* \backsimeq y^{exp}\ ?$$

   Si se cumple esta ecuación, se avanza hacia el paso 4). Por el contrario, si no se cumple dicha ecuación, se vuelve hacia el paso 1), y se entra de nuevo con otros valores x* de entrada.

4. El proceso de optimización ha terminado y la solución del problema serían los valores de x* de entradas correspondientes a los valores de salida y* tales que $y^* \backsimeq y^{exp}$.

El proceso de optimización realmente puede terminar por varias razones, pero siempre se trata de minimizar lo máximo posible la función objetivo empleada, que al fin y al cabo es lo mismo que decir $y^* \backsimeq y^{exp}$. Sin embargo, no siempre podemos llegar a cumplir la ecuación (1). Es por esta razón que se establecen una serie de restricciones para el proceso de optimización. Se establecen criterios de parada diferentes al de minimizar la función objetivo. Se establece por ejemplo número máximo de iteraciones en la optimización, así como también, un tiempo máximo de cálculo.

2.2.2 Funcionamiento interno de la optimización mediante subrogación

Después de haber explicado el proceso que sigue externamente la optimización mediante subrogación, pasamos a explicar a continuación el proceso interno del método de optimización que sigue la subrogación para encontrar un mínimo global:

En el proceso se distinguen 2 fases principales:

1) 1ª fase: Fase de construcción de la subrogación

Dividimos esta primera fase en 3 partes:

a. Se generan una serie de puntos cuasi-aleatorios dentro de los límites superiores e inferiores (‘ub’ y ‘lb’) para cumplir con el mínimo de puntos de subrogación. El usuario podrá aportar un conjunto de puntos iniciales y el algoritmo hará uso de estos puntos iniciales y creará otros puntos cuasi-aleatorios adicionales si así fuese necesario (este módulo es modificable en las opciones de funcionamiento del algoritmo, se puede establecer en cualquier valor). Es decir, si el número mínimo de puntos de subrogación fuese 50 y el usuario aportará un conjunto de 30 puntos iniciales, el algoritmo generará cuasi-aleatoriamente 20 puntos adicionales para llegar al número mínimo de puntos de subrogación.

b. Una vez generados, se procede a evaluar dichos puntos en la función objetivo.

c. En último lugar, se interpola la función objetivo utilizando un interpolador de función de base radial (RBF, funciones cuyo valor depende únicamente de la distancia desde un punto central). La interpolación RBF tiene varias propiedades convenientes que la hacen adecuada para la construcción de un modelo subrogado [13]:

   - Un interpolador RBF se define utilizando la misma fórmula en cualquier número de dimensiones y con cualquier número de puntos.
   - Un interpolador RBF toma los valores prescritos en los puntos evaluados.
   - Evaluar un interpolador RBF toma poco tiempo.
   - Agregar un punto a una interpolación existente toma relativamente poco tiempo.
   - Construir un interpolador RBF implica resolver un sistema lineal de ecuaciones de N x N, donde N es el número de puntos del subrogado.
   - El algoritmo utiliza una ‘cubic RBF with a linear tail’ (en este caso, la función es cúbica, lo que significa que su forma se asemeja a una función cúbica y la cola lineal indica que la función de base radial tiene una componente adicional que se comporta linealmente a medida que la distancia desde el punto central aumenta). Esta RBF minimiza una medida de irregularidad.

El modelo de Kriging se usa principalmente cuando se necesita una estimación de la incertidumbre y se puede definir un modelo de covarianza adecuado. La predicción juega un papel importante en esta técnica de interpolación.

Las funciones de base radial (RBF) se utilizan cuando se necesita una interpolación rápida y eficiente en problemas de alta dimensión y cuando la incertidumbre no es el principal foco. Es común en algoritmos de optimización global como ‘surrogateopt’.

Por lo tanto, en los modelos subrogados, se puede utilizar tanto Kriging como RBF dependiendo de las necesidades del problema específico. El algoritmo surrogateopt, en particular, utiliza RBF para la interpolación.

En la siguiente Figura 2.9 se aprecian los 3 pasos que forman esta primera fase de la construcción de la subrogación:

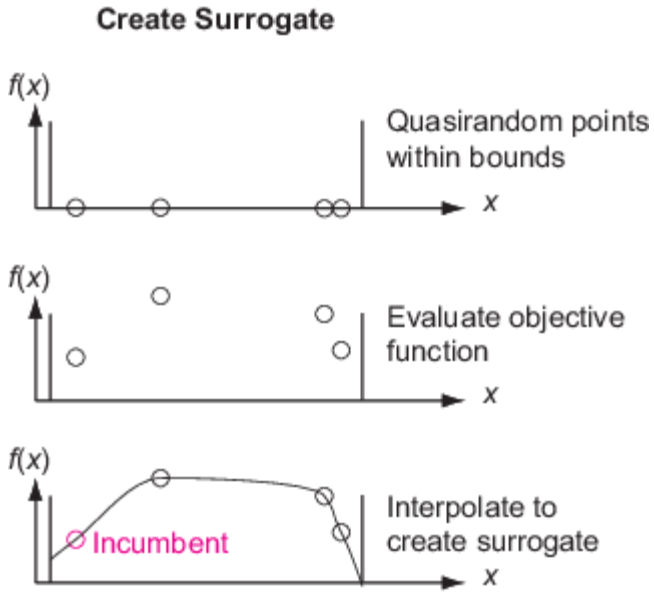

Figura 2.9 Fase de construcción de la subrogación.

2) 2ª fase: Fase de búsqueda del mínimo

En esta segunda y última fase, podemos dividir el proceso de la búsqueda del mínimo en 4 partes:

a. El solucionador ('surrogateopt') busca el mínimo a través de un proceso relacionado con la búsqueda local. El solucionador inicializa una escala para la búsqueda con el valor 0.2. La escala es como un radio de la región de búsqueda o el tamaño de la malla en una búsqueda de patrón. El solucionador comienza desde el 'incumbent point', que es el punto con el menor valor de la función objetivo desde el último reinicio ('reset') del subrogado. El solucionador busca un mínimo de una 'función de mérito' que se relaciona tanto con el subrogado como con la distancia desde los puntos de búsqueda existentes, tratando de equilibrar la minimización del subrogado y la exploración del espacio. El solucionador añade cientos o miles de vectores pseudoaleatorios con longitudes escaladas cercanos al 'incumbent' para obtener puntos de muestra ('sample points').

b. Una vez obtenidos los puntos de muestra, estos son evaluados en la 'función de mérito' pero no en ningún punto que esté a una distancia menor que la 'mínima distancia entre puntos' de un punto previamente evaluado. Esta restricción asegura que el solucionador no evalúe la 'función de mérito' en nuevos puntos que estén muy cerca de cualquier punto previamente evaluado. Esto evita redundancias y asegura que la búsqueda explore nuevas áreas del espacio de soluciones en lugar de concentrarse demasiado en regiones ya exploradas. El punto con el valor más bajo de la función de mérito se llama punto adaptativo ('adaptive point').

c. En tercer lugar, el punto adaptativo es evaluado por el solucionador en la función objetivo y actualiza el modelo subrogado con este valor en la función objetivo.

d. Por último, si el valor de la función objetivo en el punto adaptativo es suficientemente menor que el valor del 'incumbent', entonces el solucionador considera la búsqueda exitosa y establece el punto adaptativo de estudio como el nuevo 'incumbent'. De lo contrario, el solucionador considera la búsqueda como no exitosa y no cambia el 'incumbent'.

Uno de los aspectos a tener en cuenta es la escala utilizada en cada evaluación. El solucionador cambia la escala cuando se cumple la primera de estas condiciones:

- Ocurren tres búsquedas exitosas desde el último cambio de escala. En este caso, la escala se duplica, hasta una longitud máxima de escala de 0.8 veces el tamaño de la caja especificada por los límites.
- Ocurren max(5, nvar) búsquedas no exitosas desde el último cambio de escala, donde nvar es el número de variables del problema.

En la siguiente Figura 2.10 se muestran gráficamente los 4 pasos que componen la fase de búsqueda del mínimo.

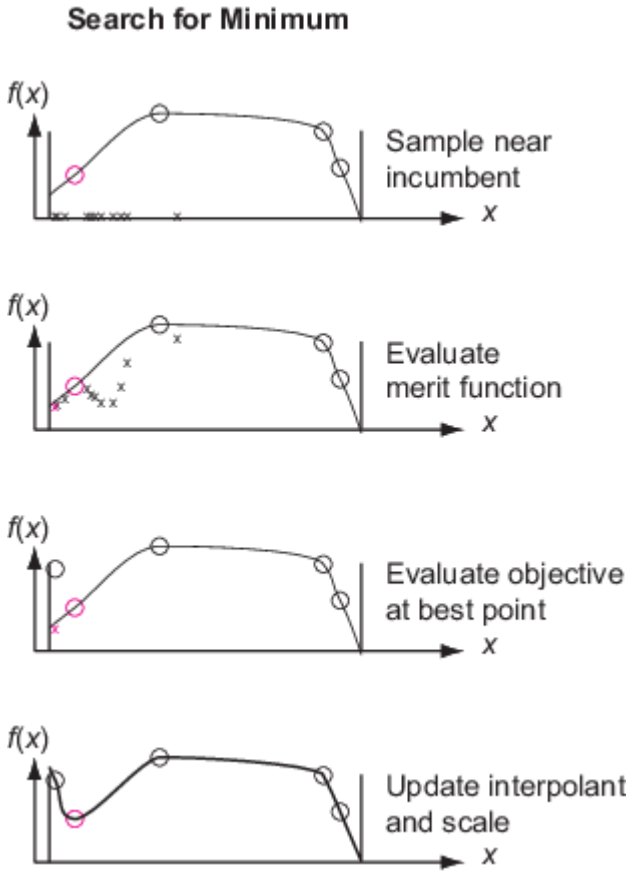

Figura 2.10 Fase de la búsqueda del mínimo del proceso de optimización mediante subrogación.

Pasamos a detallar la 'función de mérito'. La función de mérito está compuesta por 2 términos:

1. Subrogado escalado ('scaled surrogate')

Definimos los siguientes parámetros:

- $s_{mín}$: valor mínimo del subrogado entre los puntos de muestra ('sample points')
- $s_{máx}$: valor máximo del subrogado entre los puntos de muestra ('sample points')
- $s(x)$: valor del subrogado en el punto x ('sample points')

Entonces, el subrogado escalado $S(x)$ es:

$$S(x) = \frac{s(x) - s_{mín}}{s_{máx} - s_{mín}}$$

donde, $S(x)$ es no negativo (es decir, mayor o igual que cero) y es cero en los puntos x que tienen el valor mínimo del subrogado entre los puntos de muestra.

2. Distancia escalada ('scaled distance')

Definimos los siguientes parámetros:

- $x_j,\ j = 1, \ldots, k$. Siendo k el número de puntos evaluados
- d_{ij}: distancia del punto de muestra i al punto evaluado j
- $d_{mín} = min(d_{ij})$
- $d_{máx} = max(d_{ij})$

donde, el mínimo y el máximo se toman sobre todos los i y j.

Entonces, la distancia escalada $D(x)$ es:

$$D(x) = \frac{d_{máx} - d(x)}{d_{máx} - d_{mín}}$$

donde, $d(x)$ es la distancia mínima entre x y un punto evaluado y $D(x)$ es no negativa (es decir, mayor o igual que cero) y es cero en los puntos x que están a la máxima distancia de los puntos evaluados. Así, minimizar $D(x)$ lleva al algoritmo a puntos que están lejos de los puntos evaluados.

La función de mérito es una función pondera por estos 2 términos. Es una combinación convexa del 'subrogado escalado' y la 'distancia escalada'. Una combinación convexa es una mezcla de dos o más términos de tal manera que los coeficientes de ponderación suman uno y cada coeficiente es no negativo. En el caso de la función de mérito, una combinación convexa garantiza que la suma de las ponderaciones del subrogado escalado $S(x)$ y la distancia escalada $D(x)$ es igual a 1, y cada ponderación está entre 0 y 1.

Para un peso w con $0 < w < 1$, la función de mérito es:

$$f_{merit} = wS(x) + (1 - w)D(x)$$

Un valor grande de w da importancia a los valores del subrogado, haciendo que la búsqueda minimice el subrogado. Un valor pequeño de w da importancia a los puntos que están lejos de los puntos evaluados, llevando la búsqueda a nuevas regiones. Durante esta segunda fase de búsqueda del mínimo, el peso w cicla a través de estos cuatro valores, como sugieren Regis y Shoemaker [14]: 0.3, 0.5, 0.8 y 0.95.

El objetivo de la función de mérito es equilibrar la búsqueda entre minimizar el valor del subrogado (una aproximación de la función objetivo) y explorar nuevas regiones del espacio de búsqueda que están lejos de los puntos ya evaluados. Esto ayuda a guiar el algoritmo de optimización hacia soluciones óptimas de manera eficiente.

A modo de conclusión, la optimización a través de un modelo subrogado es un método altamente eficiente para problemas con evaluaciones de funciones costosas, ya que utiliza aproximaciones rápidas para guiar la búsqueda hacia óptimos, reduciendo significativamente el número de evaluaciones directas necesarias y acelerando el proceso de optimización.

3 APLICACIÓN DE MÉTODOS DE OPTIMIZACIÓN EN VIGA SIMPLEMENTE APOYADA

En este apartado se entenderán mejor los conceptos teóricos anteriormente explicados en el apartado 2. Concretamente, analizaremos un caso práctico en el que someteremos a flexión (carga uniforme q) a una viga simplemente apoyada (viga con libertad de desplazamiento en dirección longitudinal) a través de los métodos de optimización vistos previamente, algoritmo genético y optimización mediante subrogación. En este caso práctico abordaremos además las principales diferencias entre ambos métodos, ventajas e inconvenientes de uno respecto al otro (se menciona posteriormente en el apartado 3.2.2 por ejemplo la opción 'trials' que ofrece la optimización mediante subrogación).

Para presentar los métodos de optimización en nuestro caso práctico, nos enfocaremos en características críticas. Estas características son cruciales ya que fueron determinantes para optar por un enfoque basado en subrogación (o sustitución) en lugar de un algoritmo genético. Nos referimos a la precisión de los resultados, el tiempo de computación y la robustez o convergencia del algoritmo. Estas cualidades se traducen en la exactitud de los resultados obtenidos, el tiempo necesario para resolver el problema de optimización y la capacidad del algoritmo para encontrar soluciones válidas, incluso en entornos con ruido o incertidumbre.

Para evaluar las características mencionadas, realizaremos un total de seis análisis. Tres de estos análisis utilizarán un enfoque de subrogación, mientras que los otros tres emplearán un algoritmo genético.

- Primeros Cuatro Análisis:

En estos primeros cuatro análisis, utilizaremos una sola variable de cálculo, específicamente la variable del módulo elástico a compresión del hormigón correspondiente a 40 GPa.

- Tipos de Análisis:
    - Dos Análisis Analíticos: Aquí, el valor del desplazamiento de referencia será uno que proponemos.
    - Dos Análisis mediante Elementos Finitos: Utilizaremos el método de elementos finitos para obtener resultados.

- Últimos Dos Análisis:

En estos dos últimos análisis, introduciremos dos variables adicionales para hacer el cálculo más complejo.

- Tipos de Análisis:
    - Método de Elementos Finitos: Ambos análisis se realizarán utilizando el método de elementos finitos para evaluar mejor la precisión, el tiempo de computación y la robustez del algoritmo.

Este enfoque nos permitirá comparar de manera clara y efectiva las diferencias en precisión, tiempo de computación y robustez entre los métodos de subrogación y los algoritmos genéticos en contextos de complejidad creciente.

El caso de aplicación de los métodos de optimización lo dividiremos en 2 problemas: La viga formada por un solo material, y por otro lado, la viga formada por tres materiales (primeros casos de estudio y últimos respectivamente). Definimos las propiedades de ambas vigas a continuación:

Estudio con 1 variable

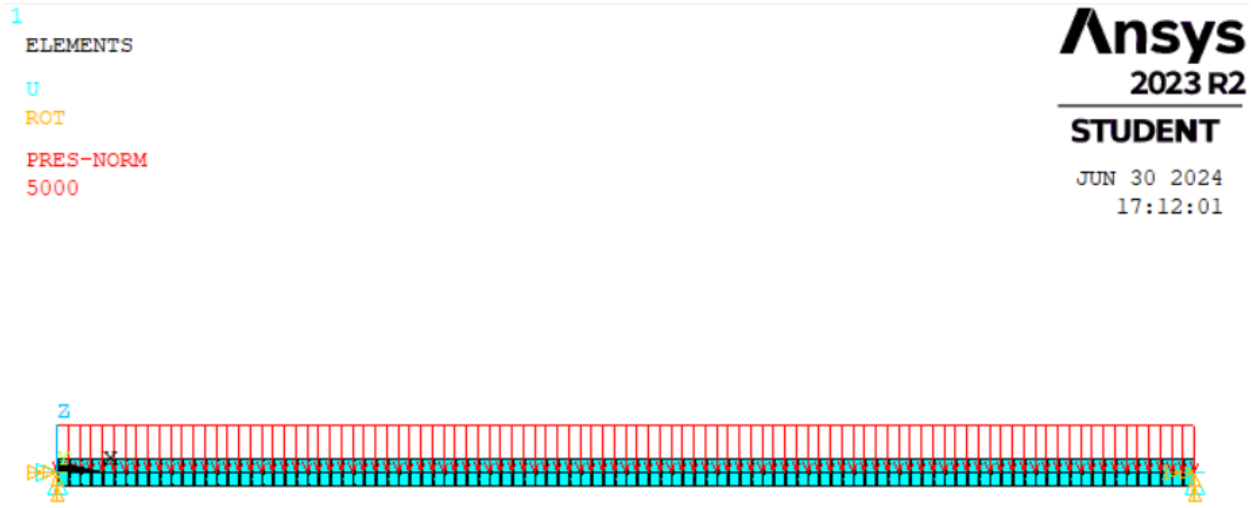

Figura 3.1 Viga simplemente apoyada compuesta por un único material.

Estudio con 3 variables

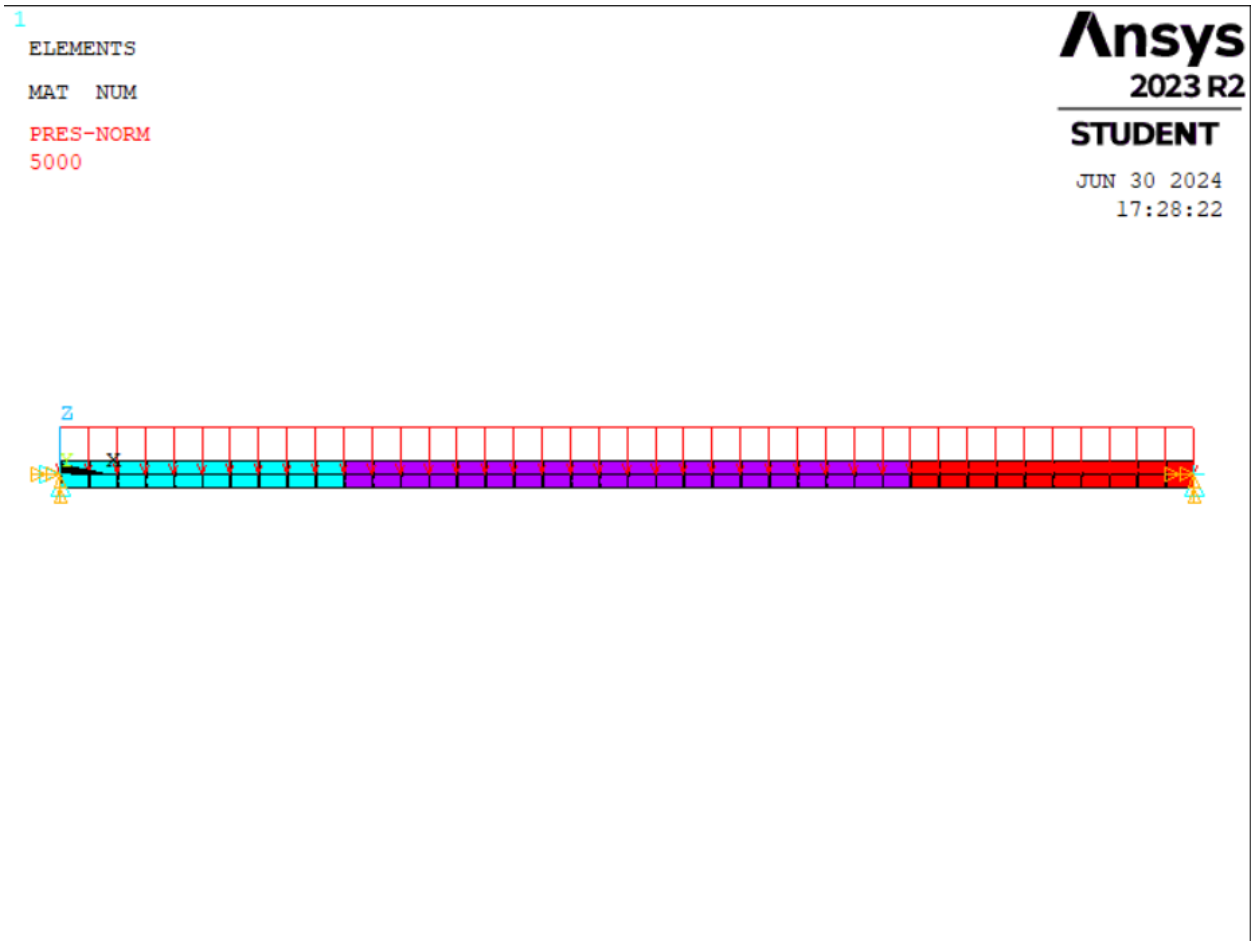

Figura 3.2 Viga simplemente apoyada compuesta por tres materiales.

En el caso de estudio con 1 variable, la viga está formada por el hormigón de 40 GPa de módulo elástico a compresión a largo de toda la viga y una sección de dimensiones b x h. Como se puede observar en la Figura 3.1, este es el problema que resolveremos en los 4 primeros análisis (se puede apreciar las condiciones de contorno y las fuerzas de presión actuando en la viga). Por otro lado, en la Figura 3.2 observamos el caso de estudio con 3 variables, en el que la viga está formada por un hormigón de 40 GPa, otro de 50 GPa de resistencia a compresión y un acero de 200 GPa de resistencia a tracción. El hormigón con el valor más bajo se encuentra en el primer cuarto de la viga. El hormigón de 50 GPa se extiende desde el primer cuarto hasta el tercer cuarto de la viga. Finalmente, el acero con 200 GPa de resistencia está ubicado en el último cuarto de la viga.

3.1 Análisis analíticos

En este apartado resolveremos el problema de la figura 3.3:

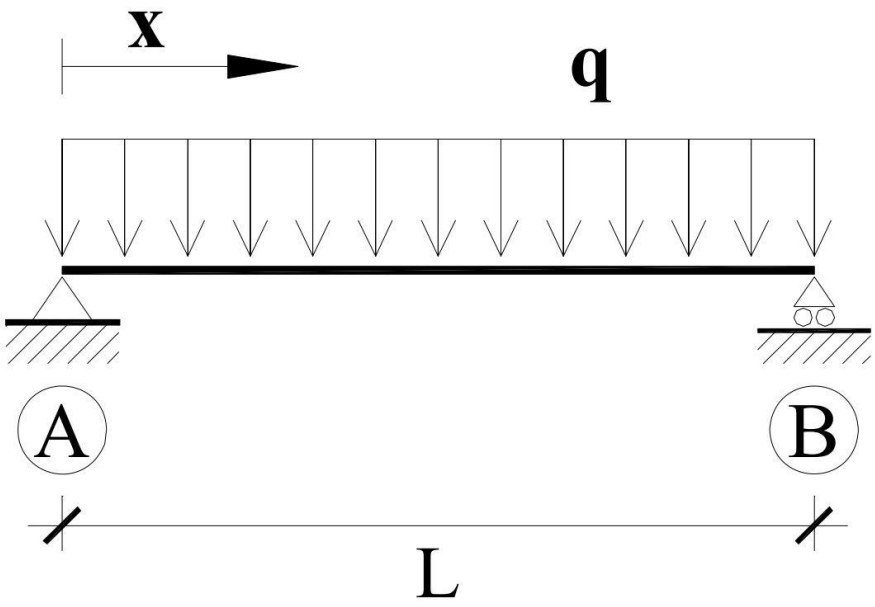

Figura 3.3 Viga simplemente apoyada sometida a una carga uniforme q.

Para los dos primeros análisis, crearemos un código de MATLAB en el que la solución sea analítica, es decir, el comportamiento de la flecha en el vano (flecha en función del punto de estudio en el vano) se regirá por una función que implementaremos nosotros. Esta función es válida evidentemente para ambos análisis, uno a través del método de subrogación y el otro análisis mediante el método de algoritmo genético. La variable de cálculo será el material del cual está formado la viga, 40 GPa de hormigón. En este apartado no hará falta ningún modelo de ANSYS puesto que la solución de referencia es analítica, es decir, viene dada por una función ya definida. La función en cuestión es la siguiente:

$$flecha\_obj = \frac{Px}{24EI_{obj}} * (x^3 - 2lx^2 + l^3)$$

siendo,

- EI_{obj} , la rigidez a flexión de la viga (valor de e40 objetivo, lo fijamos en 42 GPa, por ejemplo, la inercia I toma un valor de 1000e-5)

$$EI_{obj} = 42e9 * 1000e - 5$$

- P , la carga distribuida uniforme aplicada en la viga (toma 5000 N/m de valor, por ejemplo)
- x , el número de puntos de estudio (para estos dos análisis estudiaremos el comportamiento de la viga en 3 puntos de estudio. El primer punto de estudio es a L/3, el segundo de ellos, a L/2, y el último punto a 3L/4)
- l , la longitud de la viga (en este caso la viga medirá 10 metros de longitud)

Por otro lado, debemos definir otros parámetros importantes a la hora del cálculo (todos los parámetros que se mencionan a continuación se explican detalladamente en el apartado 4.3):

- En cuanto a los límites superiores (ub) e inferiores (lb) de la variable de estudio (e40), los fijaremos en los siguientes valores:
  - ub = [35e9]

- lb = [45e9]

- El número máximo de iteraciones (MFE) lo fijaremos en 100 iteraciones.
- El generador de números aleatorios (rng) se establece para un valor por defecto. Esto se debe a que el valor por defecto de 'rng' otorga una gran reproducibilidad en los resultados, puesto que resetea el estudio en caso de obtener resultados similares en repetidas ocasiones con el objetivo de lograr una mayor diversidad en el estudio y garantizar así un mayor rango de puntos estudios para realizar un análisis lo más completo posible.
- Por último, en cuanto a la función objetivo ('Fobj'), establecemos una función objetivo estándar. La norma de la diferencia entre resultados (resta entre resultado experimental (objetivo) y resultado analítico). En otras palabras, es el error absoluto entre el valor exacto (obj) y el valor aproximado (trial).

```
Fobj = norm (flecha_obj – flecha_trial)
```

En el apéndice A.2 encontramos el código de MATLAB que implementa los cálculos analíticos que se exponen en este apartado.

3.1.1 Resultados de los estudios analíticos

Pasamos a estudiar las soluciones obtenidas. Se muestran a continuación las figuras (Figura 3.4.a y 3.4.b) en las que se observa el proceso de optimización seguido en los dos primeros análisis y posteriormente, la diferencia de resultados en los 3 puntos de estudio establecidos:

En primer lugar, el análisis mediante subrogación en la Tabla 3.1:

Estudio analítico de subrogación	
Fobj	**X_optím**
0	4.20e+10 Pa

Tabla 3.1 Resultados del estudio analítico mediante subrogación.

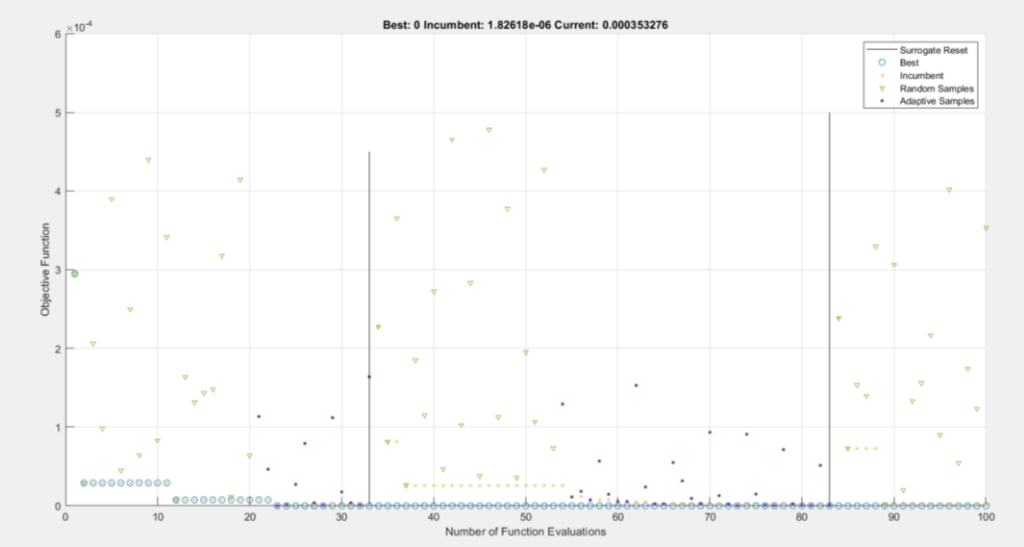

Figura 3.4.a Resultados de MATLAB para subrogación.

En segundo lugar, el estudio analítico mediante algoritmo genético en la Tabla 3.2:

Estudio analítico de algoritmos genéticos	
Fobj	**X_optím**
3.028006614488681e-06	4.194828625375817e+10 Pa

Tabla 3.2 Resultados del estudio analítico mediante algoritmos genéticos.

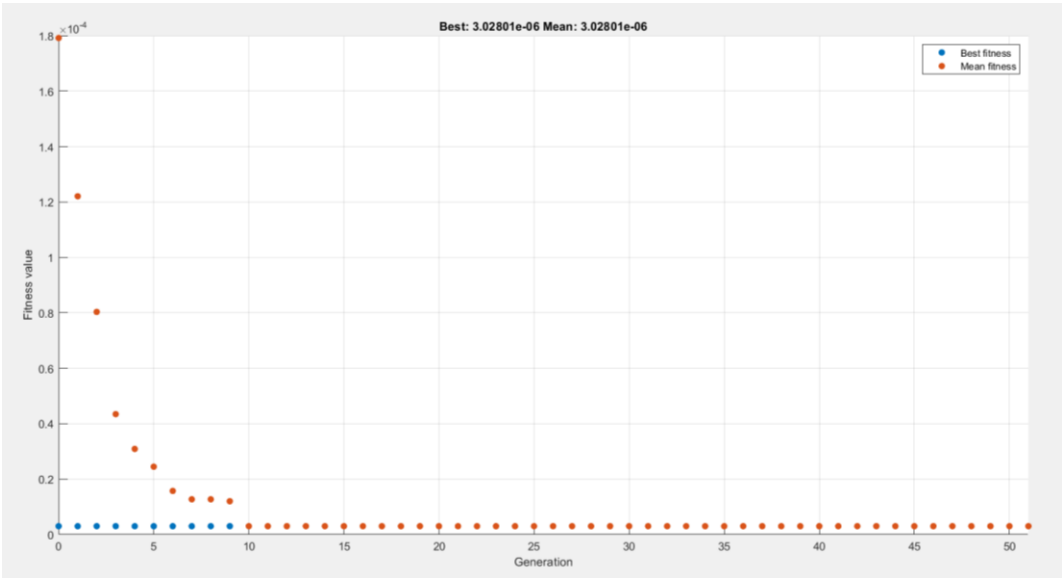

Figura 3.4.b Resultados de MATLAB para algoritmos genéticos.

Como podemos apreciar en estas 2 figuras, ambos métodos son muy precisos y el análisis se ha realizado en pocos segundos. De momento, no hay diferencia apenas en los resultados. Si es cierto que la diferencia en los resultados de los 3 puntos de estudio es mínima. Vemos que, en el caso de la subrogación, el resultado es exacto, y en el algoritmo genético hay una pequeña diferencia. Debemos tener en cuenta que este caso de estudio es un ejemplo simple de una viga simplemente apoyada. Es un problema que ambos métodos de optimización resuelven con facilidad.

Se puede observar además que, en el caso del algoritmo genético, se han realizado 51 iteraciones puesto que por una de las opciones por defecto, se ha parado el análisis, por haber convergido por debajo de una tolerancia de cálculo establecida por defecto. Dicha tolerancia es la misma para ambos análisis puesto que si no lo fuesen, el análisis no sería comparable.

3.2 Análisis a través de Elementos Finitos

Para afrontar los dos primeros análisis mediante ANSYS, debemos elaborar un modelo de Elementos Finitos en el que diseñaremos una viga simplemente apoyada sometida a una carga distribuida uniforme q, y que está formada por un solo material, el hormigón de 40 GPa de módulo elástico a compresión (variable de estudio, e40).

Para hacer frente a los dos últimos análisis mediante ANSYS, a diferencia de los dos primeros, emplearemos 3 materiales en la viga. Un hormigón de 40 GPa, como en el primer caso, y un hormigón de 50 GPa. Y por otro lado, para complementarse con el hormigón un acero de 200 GPa de resistencia a tracción (variables de estudio en el análisis, e40, e50 y e210).

Por tanto, en el apéndice A.3 se muestra dicho código que posteriormente será implementado en ANSYS. Una vez implementado el modelo de Elementos Finitos, procedemos a crear el código que ejecutaremos en MATLAB para realizar los procesos de optimización. En dicho código de MATLAB modificaremos respecto a los estudios analíticos una serie de parámetros, que son los siguientes:

- El número máximo de iteraciones (MFE) lo fijaremos en 50 iteraciones para la evaluación mediante subrogación y 30 iteraciones para la evaluación mediante algoritmo genético. Al ser un modelo más complejo que el anterior, se ha establecido un número diferente de iteraciones por ahorro de tiempo de cálculo.
- En segundo lugar, en cuanto a la 'Fobj', hemos simplificado su definición. Hemos decidido establecer una simple resta para la función objetivo, el resultado exacto menos el resultado aproximado, para obtener directamente el valor de la diferencia como valor de la función objetivo. Se ha tomado esta decisión por simplicidad en el cálculo.

$$F_{obj} = Resultado\ de\ referencia - Resultado\ aproximado$$

siendo,

- Resultado de referencia: Resultado exacto obtenido de ANSYS estableciendo como objetivo un valor de e40 igual a 40 GPa.
- Resultado aproximado: Resultado obtenido de la evaluación en el modelo de ANSYS del punto proporcionado por el algoritmo en cada iteración realizada.

- En tercer lugar, hemos modificado el número de puntos de estudio por simplicidad en el modelo. Hemos establecido un solo punto de estudio. Para demostrar las ventajas e inconvenientes de ambos métodos de optimización no será necesario implementar un modelo complejo. Aun así, en el estudio de la estructura que llevamos a cabo más tarde, veremos con mucha claridad las notables diferencias que presentan. En esta aplicación de métodos de optimización en modelo simple de una viga simplemente apoyada se muestra en menor medida, aunque suficiente.
- Por último, es importante destacar que los valores de los límites superior e inferior ('ub' y 'lb' respectivamente), así como el valor de 'rng', se mantienen igual respecto a los estudios anteriormente realizados.

Estos valores modificados son principalmente para los dos primeros análisis que vamos a realizar a través de Elementos Finitos. Sin embargo, muchas de estas variaciones nos serán de gran utilidad para los últimos dos análisis a realizar.

En el apéndice A.4 encontramos el código de ANSYS implementado en estos análisis a través de Elementos Finitos.

3.2.1 Resultados de los estudios mediante Elementos Finitos

Los resultados obtenidos son los que se muestran a continuación en las Figuras 3.5.a y 3.5.b, en primer lugar, mediante subrogación, y en segundo lugar mediante algoritmo genético (ver Tablas 3.3 y 3.4):

Estudio FEM de subrogación		
Fobj	**X_optím**	**elapsed_time**
7.711498177034870e-06	3.999384017280749e+10 Pa	127.123568 s

Tabla 3.3 Resultados del estudio analítico mediante algoritmos genéticos.

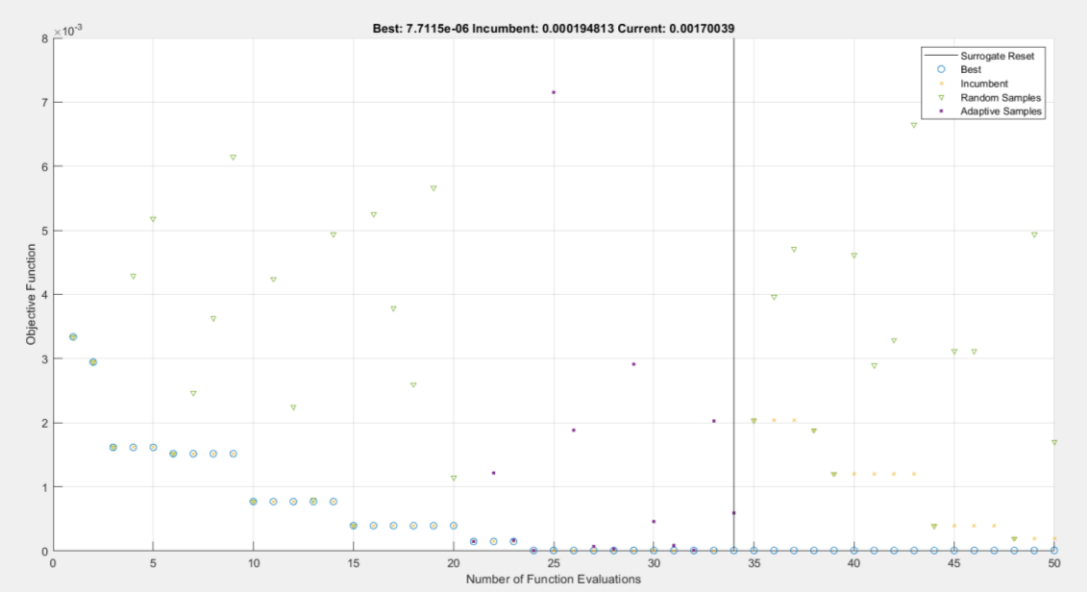

Figura 3.5.a Resultados de MATLAB para subrogación.

Estudio FEM de algoritmos genéticos		
Fobj	**X_optím**	**elapsed_time**
1.284500544475597e-04	3.989764395788231e+10 Pa	3165.401336 s

Tabla 3.4 Resultados del estudio analítico mediante algoritmos genéticos.

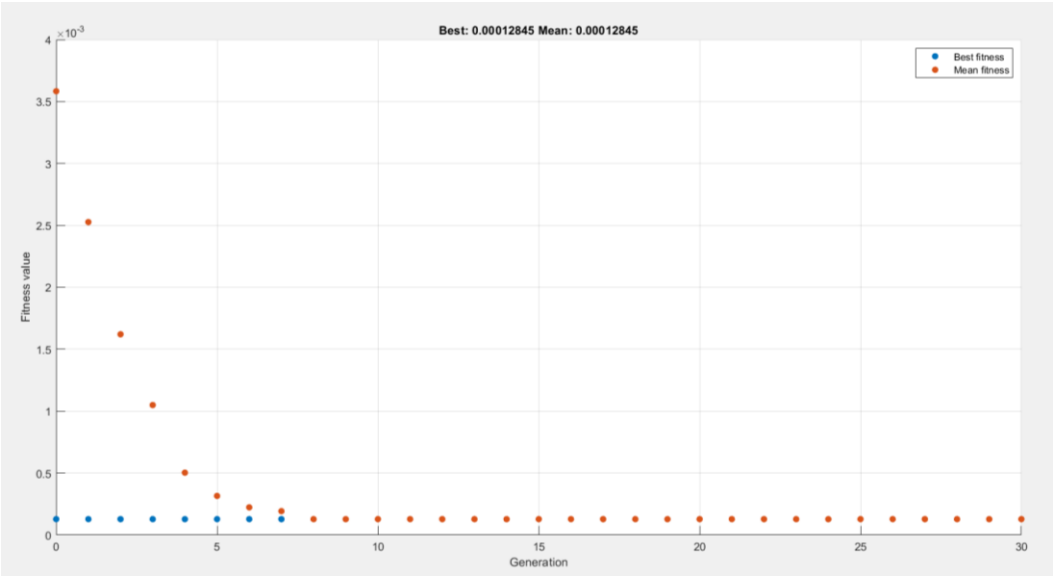

Figura 3.5.b Resultados de MATLAB para algoritmos genéticos.

En este caso, apreciaremos las primeras diferencias entre métodos. Vemos una diferencia en el tiempo de

computación bastante grande, a pesar de ser un modelo simple. Observamos en el caso del algoritmo genético un tiempo de computación de casi 53 minutos, por el contrario, en el caso de la subrogación obtenemos un tiempo de cálculo de apenas 2 minutos y escasos segundos. Por otro lado, vemos una diferencia en el valor de la función objetivo, vemos una mayor precisión en el caso de la subrogación aún sabiendo que la diferencia al ser un modelo simple, es mínima.

En último lugar, apreciaremos el estudio mediante Elementos Finitos de los análisis que incluyen 3 variables de estudio. En estos últimos dos estudios, debemos tener en cuenta las siguientes consideraciones, además de las anteriormente mencionadas en los análisis mediante MEF que acabamos de realizar.

- En cuanto a los límites superiores (ub) e inferiores (lb) de la variable de estudio (e40), los fijaremos en los siguientes valores:
  - ub = [35e9 45e9 190e9]
  - lb = [45e9 55e9 210e9]
- Los valores objetivo (los de referencia) se obtienen a través del modelo de ANSYS definiendo como valores objetivo 40 GPa, 50 GPa y 200 GPa (e40, e50 y e210 respectivamente).
- El resto de los parámetros se mantienen igual en estos dos análisis.

Resolviendo el problema que se muestra en la Figura 3.2, obtenemos los siguientes resultados, en primer lugar, el estudio mediante optimización, y posteriormente, el estudio mediante algoritmo genético (Tabla 3.5 y 3.6):

Estudio FEM de subrogación				
Fobj	**X_optím**			**elapsed_time**
0	4.0e+10 Pa	5.0e+10 Pa	20.0e+10 Pa	148.055250 s

Tabla 3.5 Resultados del estudio analítico mediante subrogación.

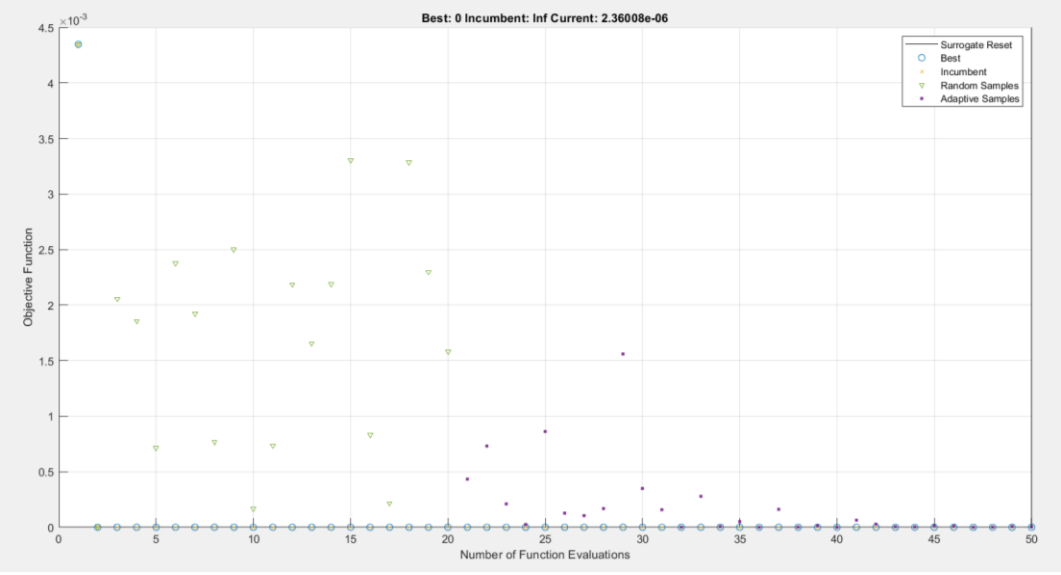

Figura 3.6.a Resultados de MATLAB para subrogación.

Estudio FEM de algoritmos genéticos				
Fobj	**X_optím**			**elapsed_time**
1.265228592658985e-07	4.206046088019609e+10 Pa	4.973288848902729e+10 Pa	19.51974080570131e+10 Pa	3306.844579 s

Tabla 3.6 Resultados del estudio analítico mediante algoritmos genéticos.

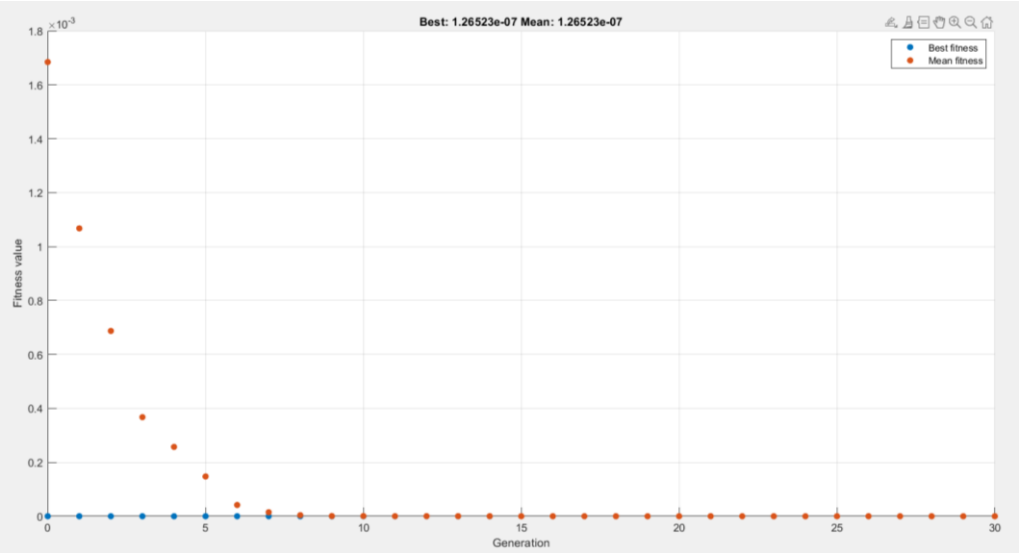

Figura 3.6.b Resultados de MATLAB para algoritmos genéticos.

En este estudio, al igual que en el anterior, observamos en las Figuras 3.6.a y 3.6.b que el método de subrogación produce resultados con una precisión superior y requiere un menor tiempo de computación. No obstante, aunque esperábamos una diferencia más notable en términos de precisión, el hecho de que se trate de un modelo de una viga simple dificulta que un método de optimización tan eficaz genere resultados deficientes.

3.2.2 Utilidad de 'trials' en la función de subrogación

Demostramos a continuación una de las dos utilidades que caracterizan la robustez y adaptabilidad del método de subrogación. En primer lugar, veremos la utilidad de los 'trials'. Veremos cómo se puede aprovechar este parámetro, para qué sirve, etc. En segundo y último lugar, veremos cómo se adapta el método de subrogación a una posible variación en el problema de estudio (ya sea porque la estructura en cuestión se ha deteriorado, ha sido dañada o lo que fuere).

Para exponer la interesante utilidad que ofrece el módulo de 'trials', estudiaremos el caso de carga número 10 de la primera prueba de carga. Se puede estudiar dicha utilidad sin necesidad de recurrir a las pruebas de carga. Sin embargo, al ser un problema sencillo, la diferencia entre implementar los 'trials' en un modelo y no implementarlos, sería prácticamente nula. Es por eso que nos centramos en el caso de carga número 10 de la

primera prueba de carga, puesto que es un problema más complejo y que requiere un mayor coste computacional tanto de tiempo como de cálculo.

El procedimiento a seguir es el siguiente: Primero, estudiaremos el décimo caso de carga realizando 100 iteraciones sin utilizar el módulo de 'trials'. Para evaluar la utilidad del uso de 'trials', realizaremos aproximadamente 100 iteraciones, comenzando con 20 iteraciones iniciales como base. Luego, continuaremos con intervalos de 15 iteraciones, evaluando los puntos analizados para que el algoritmo aproveche la información previa y encuentre un mejor valor sin necesidad de completar 100 iteraciones para obtener un valor aceptable. Los resultados son los siguientes:

Estudio de 100 iteraciones sin uso de 'trials'
Fobj_mín
0.0170788

Tabla 3.7 Valor de la función objetivo 1ª iteración.

En el caso de la implementación de 'trials' en el cálculo se presentan varias tablas que nos mostrarán el proceso que seguimos para examinar los puntos evaluados y que el algoritmo de subrogación sea consciente de que se han evaluado una serie de puntos previos en los cuales puede apoyarse para proseguir con su optimización y que sea más precisa y ligera, ver Tabla 3.8, 3.9, 3.10 y 3.11.

Estudio de 100 iteraciones mediante el uso de 'trials' **Primeras 20 iteraciones**
Fobj_mín
0.032130514869396

Tabla 3.8 Valor de la función objetivo 2ª iteración.

Estudio de 100 iteraciones mediante el uso de 'trials' **Próximas 15 iteraciones (suma total : 35)**
Fobj_mín
0.028904242488669

Tabla 3.9 Valor de la función objetivo 3ª iteración.

Estudio de 100 iteraciones sin uso de ‘trials’ Próximas 15 iteraciones (suma total : 50)
Fobj_mín
0.016609159167847

Tabla 3.10 Valor de la función objetivo 4ª iteración.

Estudio de 100 iteraciones mediante el uso de ‘trials’ Próximas 15 iteraciones (suma total : 90)
Fobj_mín
0.016074709792218

Tabla 3.11 Valor de la función objetivo última iteración.

El análisis de las últimas 50 iteraciones no muestra una diferencia significativa en la precisión debido a que el problema planteado, con límites superiores e inferiores amplios, es complejo y encontrar una solución con una función objetivo pequeña resulta difícil. Además, la proximidad al vano número 13 introduce una cierta incertidumbre, ya que este vano, por su geometría curva, sufre torsión, como se verá más adelante en el apartado 4.3.3.

A pesar de estos inconvenientes, hemos logrado una mejor solución en comparación con el análisis que no utiliza 'trials', y con menos iteraciones. Aunque en este caso la diferencia es pequeña, en estudios estructurales más complejos la diferencia será mayor, por lo que el uso de 'trials' será muy útil. El módulo 'trials' puede ayudarnos a ahorrar muchos costos computacionales. En este caso concreto, nos ha permitido ahorrar aproximadamente 50 iteraciones.

Además, el uso de 'trials' puede proporcionarnos una estimación de la convergencia del problema. En otras palabras, al utilizar 'trials' en el análisis, podemos realizar un menor número de iteraciones que resultan muy útiles. Si observamos que el valor de la función objetivo varía poco a lo largo de las iteraciones, podemos concluir que el problema ha convergido y detener el análisis. Por el contrario, en un análisis con un número predeterminado de iteraciones, el proceso debe completarse por completo antes de evaluar la convergencia.

3.2.3 Adaptabilidad de la subrogación en problemas de optimización

En segundo lugar, analizaremos la robustez del método de subrogación. Para ello, retomamos el caso analítico presentado en el apartado 3.1, donde observamos que el valor objetivo que buscamos obtener a través de ‘flecha_trial’ es fijado por el usuario. En la vida real, una estructura en una determinada parte local posee un valor específico para una cierta propiedad. Sin embargo, en cualquier momento, esa parte puede experimentar una deformación excesiva por diversas causas, sufrir un exceso de tensiones que comprometa la sección, o verse afectada por un accidente automovilístico o ferroviario en el caso de infraestructuras civiles o ferroviarias, lo que puede causar un daño considerable y alterar las variables de estudio de la estructura. Es por esta razón que, en el caso analítico de estudio de una variable (e40), se puede apreciar claramente lo

mencionado anteriormente.

Una vez iniciado un estudio, ya tengo una serie de puntos evaluados. Si la estructura sufre un daño que afecta alguna variable de estudio, no es necesario comenzar de nuevo un análisis desde cero. La adaptabilidad de este método permite reevaluar los puntos previamente analizados en el contexto del nuevo problema (que en realidad es el mismo problema con un valor diferente para una o varias variables específicas). Esto nos permite aprovechar el análisis previo para llegar a una solución precisa en un tiempo de computación reducido.

Vamos a realizar 30 iteraciones en el problema planteado en el apartado 3.1 (caso analítico). El valor objetivo inicial es de 40 GPa. Obtenemos los siguientes resultados después de 30 iteraciones (límites superiores e inferiores iguales a los vistos en el apartado 3.1):

Valor de la función objetivo a las 30 iteraciones
Fobj_mín
3.972187710997309e-07

Tabla 3.12 Valor de la función objetivo para 30 iteraciones inicales (estructura sin daños).

Supongamos ahora que la estructura sufre un daño y el valor del módulo elástico a compresión del hormigón que compone la viga, pasa a tener un valor de 36 GPa a causa del daño sufrido. Realizaremos 10 iteraciones, evaluando los puntos anteriormente analizados en el 'nuevo problema' para aprovechar el anterior análisis. Veremos ahora cuantas iteraciones son necesarias para encontrar un valor aceptable de la variable de estudio e40 despues de que la estructura haya sufrido un daño:

Valor de la función objetivo a las 30 iteraciones
Fobj_mín
0

Tabla 3.13 Valor de la función objetivo para 10 iteraciones (estructura con daños).

Ya en la iteración número 4 obtenemos el valor nulo de la función objetivo, no ha sido necesario empezar un nuevo análisis, simplemente evaluar los puntos anteriores en el 'nuevo problema' planteado. Con este ejemplo se muestra la adaptabilidad que nos ofrece el método de subrogación en problemas de optimización.

Esta reevaluación de puntos previos se realiza a través del módulo 'trials' mencionado anteriormente. Gracias a los 'trials', conocemos los puntos evaluados en el análisis anterior. Tomamos estos puntos y los evaluamos en la nueva función objetivo, que ahora tiene un valor diferente para 'ei_obj' debido al daño sufrido por la estructura. Así, en cuestión de segundos, obtenemos una base de puntos que nos permite continuar el análisis como si hubiésemos realizado cientos de iteraciones previamente (en este ejemplo sencillo se ha hecho con pocas iteraciones para mayor claridad).

Por lo tanto, a modo de conclusión, la función de optimización subrogada se destaca como una excelente opción para realizar estudios de detección de daños y análisis preventivo de infraestructuras en general. Su gran adaptabilidad le permite ajustarse rápidamente a cambios inesperados en las condiciones de la estructura, mientras que su robustez asegura que los análisis sean consistentes y fiables. Además, la precisión en los resultados, obtenida mediante la reutilización de puntos evaluados previamente, contribuye a una evaluación más exacta de las condiciones estructurales. Todo esto se logra además, con una notable rapidez de comutación, lo que permite realizar análisis complejos en menos tiempo.

4 CASO DE ESTUDIO : ESTRUCTURA E9 – LÍNEA 1 DE METRO DE SEVILLA

La estructura que será objeto de estudio en este Trabajo de Fin de Grado es la estructura E9 – Línea 1 de Metro de Sevilla. Después de haber desarrollado los conceptos necesarios acerca del mantenimiento preventivo, optimización y demás, llevaremos a cabo en la práctica todo lo aprendido.

En este capítulo definiremos la estructura que vamos a estudiar, se mostrará una explicación detallada sobre el modelo de Elementos Finitos ('EF') desarrollado en ANSYS para su posterior implementación en MATLAB. Una vez definido el modelo de EF, se procederá al proceso de optimización de las variables críticas de la estructura, y por último, analizaremos los resultados y sacaremos unas breves conclusiones de los mismos.

4.1 Estructura de estudio y Pruebas de Carga

En este apartado definiremos la estructura que vamos a estudiar y detallaremos las pruebas de carga que se han realizado sobre la misma. De dichas pruebas de carga se obtienen los resultados experimentales necesarios para llevar a cabo la optimización de las variables críticas de la estructura que veremos en el apartado 4.3.

La ubicación de la estructura E9, que forma parte de la Línea 1 del Metro de Sevilla, es fundamental para comprender su importancia en el entorno de la provincia y la mejora de accesibilidad a la ciudad para las personas. A continuación, se detalla su localización y características del emplazamiento.

La estructura E9 de la Línea 1 del Metro de Sevilla se extiende desde la estación de San Juan Bajo hasta la estación de Blas Infante. Las coordenadas geográficas de la estructura son 6° 01′ 10.37″ W, 37° 22′ 03.56″ N (EPSG:4326).

Esta estructura se encuentra en un entorno metropolitano, cruzando el río Guadalquivir y conectando áreas clave de San Juan de Aznalfarache con el barrio de Los Remedios en Sevilla. Al oeste, la estructura se inicia en San Juan Bajo, una zona principalmente residencial. Al este, termina en Blas Infante, ubicada en una zona urbana con acceso a importantes vías de comunicación y servicios.

La integración urbana de esta estructura es notable, dado que mejora significativamente la conectividad entre San Juan de Aznalfarache y Sevilla, promoviendo la conectividad y el desarrollo social de la ciudad de Sevilla. La estructura no solo facilita el transporte diario de miles de personas, sino que también contribuye a la reducción del tráfico vehicular y la contaminación en la ciudad.

Las siguientes figuras muestran la ubicación y el emplazamiento de la estructura (Figura 4.1 y 4.2), así como imágenes en las que se puede observar la estructura en su conjunto (Figura 4.3, 4.4.a, 4.4.b y 4.5), con sus respectivos vanos, pilares, etc. Además, se muestran también los tramos elevados sobre el río Guadalquivir, que forman parte de la estructura.

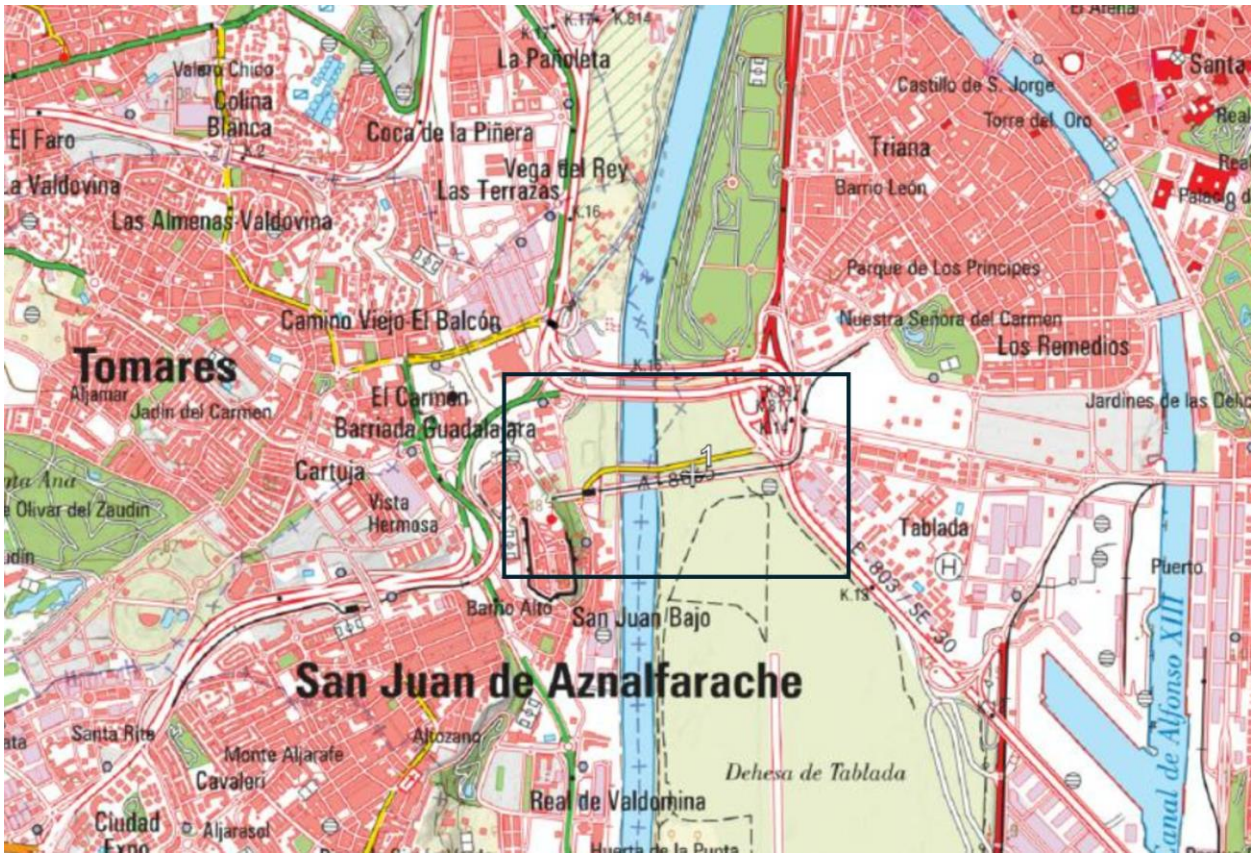

Figura 4.1 Ubicación de la estructura. Escala 1:50.000.

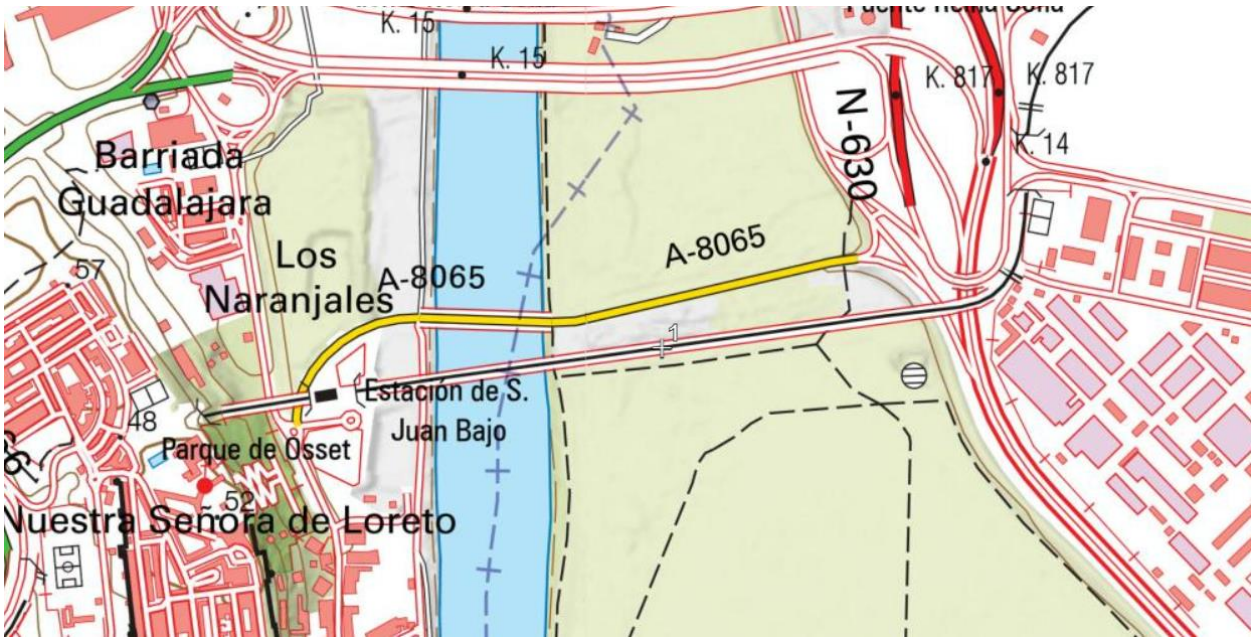

Figura 4.2 Emplazamiento de la estructura. Escala 1:5.000.

Figura 4.3 Vista en planta de la estructura.

Figura 4.4.a Vista aérea de la estructura (1).

En esta última figura, se puede apreciar cuales serán exactamente los 13 vanos que serán objeto de estudio. Los vanos de estudio de la estructura E9 comienzan una vez cruzado el Río Guadalquivir, tal y como se indica en esta última imagen (Figura 4.4.a).

Figura 4.4.b Vista aérea de la estructura (2).

Figura 4.5 Vista trasera de la estructura.

A continuación, observamos imágenes con mayor detalle de la zona inferior del puente (tablero y apoyos). En dichas imágenes se puede apreciar con mayor claridad el tablero y las secciones inferiores del puente que veremos posteriormente en el apartado 4.2, así como los diversos apoyos que tiene el puente:

Figura 4.6 Vista contrapicada de la estructura.

Figura 4.7 Vista longitudinal de la estructura.

Figura 4.8 Vista próxima a un apoyo de la estructura.

Figura 4.9 Vista contrapicada de uno de los apoyos de la estructura.

En la Figura 4.8 y 4.9 se puede apreciar los apoyos de la estructura. Dichos apoyos son apoyos tipo POT, es decir, apoyos que permiten el giro y los desplazamientos en una o dos direcciones. En el caso de de los apoyos situados en el lateral izquierdo del puente, se permiten los desplazamientos en el plano XY, plano perpendicular al eje de altura Z. Por otro lado, en el lateral derecho solo se permiten desplazamientos en la dirección longitudinal de la estructura, es decir, el eje X. Dichos apoyos se modelizarán en el apartado 4.2.

En la segunda sección de este apartado se muestran las pruebas de carga proporcionadas a raíz de un estudio independiente y realizadas en la estructura a través de unos ensayos estáticos para medir desplazamientos, giros, etc.

Las pruebas de carga se dividen en 3 pruebas. La primera de ellas, realizada individualmente en cada vano. En segundo lugar, la segunda prueba se realiza en vanos contiguos. Es decir, se colocan una serie de camiones en el vano 1 y 2, y se miden los parámetros correspondientes, después, en los vanos 2 y 3, y se hace lo mismo. Por último, la tercera prueba de carga consiste en una serie de camiones que cargan el puente en uno de los laterales para medir desplazamientos, giros y parámetros similares.

En primer lugar, en la Figura 4.10 se muestran los puntos de medida de cada vano, que hacen un total de 30 puntos de estudio. Así mismo, en la Figura 4.11 y 4.12 se representa la geometría de los camiones, la carga por eje y la disposición en la estructura. A continuación, se definen las pruebas de carga:

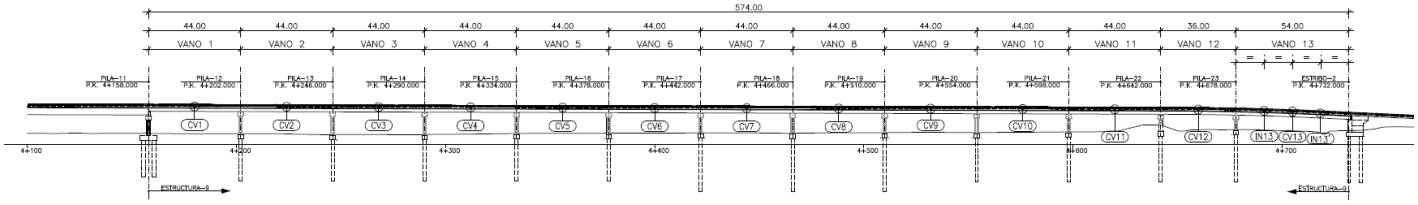

Figura 4.10 Puntos de medida a lo largo de la estructura.

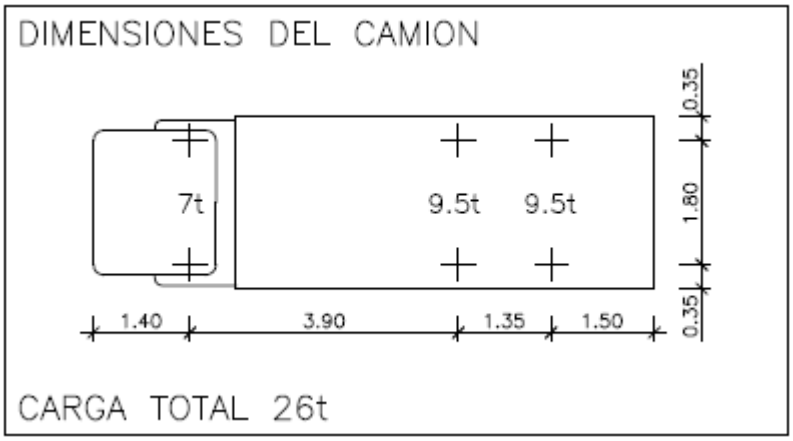

Figura 4.11 Geometría de los camiones.

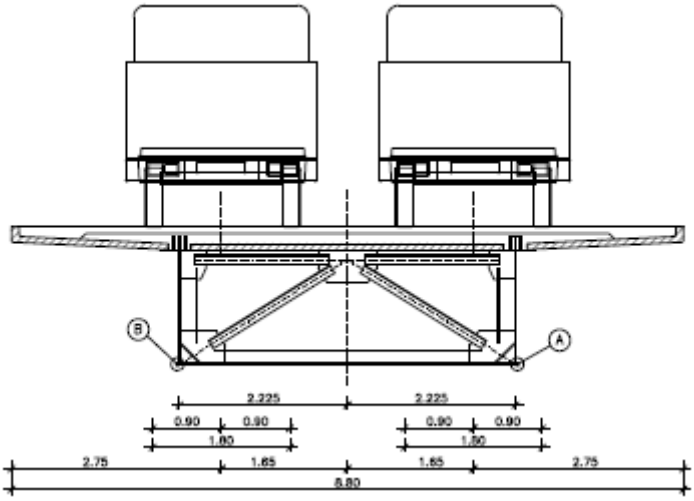

Figura 4.12 Disposición de los camiones en la estructura.

Se puede observar en la Figura 4.11 y 4.12 el peso total de 26 T de los camiones junto con sus dimensiones, así como la distribución a lo largo del ancho de dichos camiones, equidistantes del centro del ancho vano.

A continuación, se muestran las 3 pruebas de carga:

- En la primera de ellas se verán 12 camiones distribuidos a lo largo de cada vano. 6 hasta la mitad del vano, y otros 6 en el final del vano. A lo ancho, 1 camión en la mitad izquierda del ancho, y otro en la otra mitad.
- En segundo lugar, la segunda prueba de carga consiste en la distribución de 16 camiones a lo largo de dos vanos contiguos. Es decir, se sitúan los camiones en los vanos 1 y 2 y se miden los parámetros correspondientes de interés. A continuación, se cargan los vanos 2 y 3 y se sigue el mismo procedimiento. Por tanto, la distribución de estos 16 camiones es 8 camiones en el primero de los vanos de carga, y los otros 8 en el vano contiguo.
- Por último, la tercera prueba de carga consistirá en la distribución de 4 camiones en un lateral del puente. Dichos camiones se situarán en el vano 3.

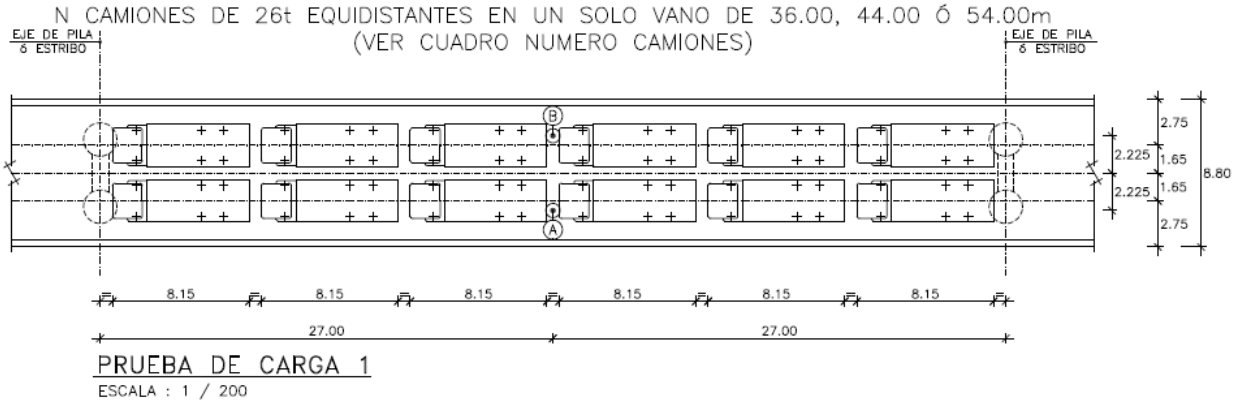

Figura 4.13 Prueba de carga 1 realizada en la estructura.

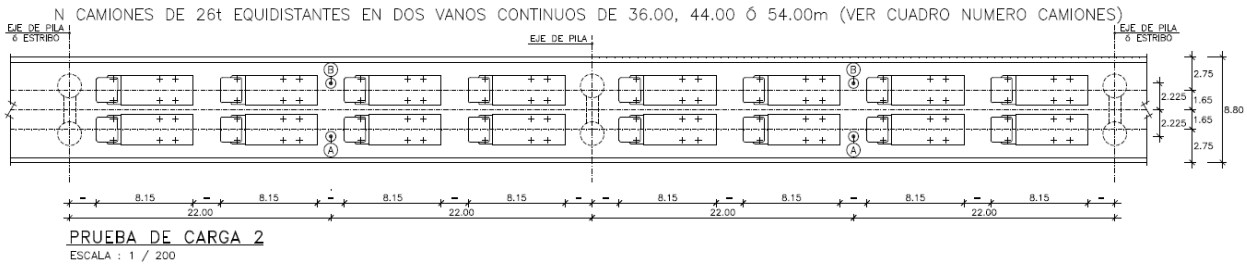

Figura 4.14 Prueba de carga 2 realizada en la estructura.

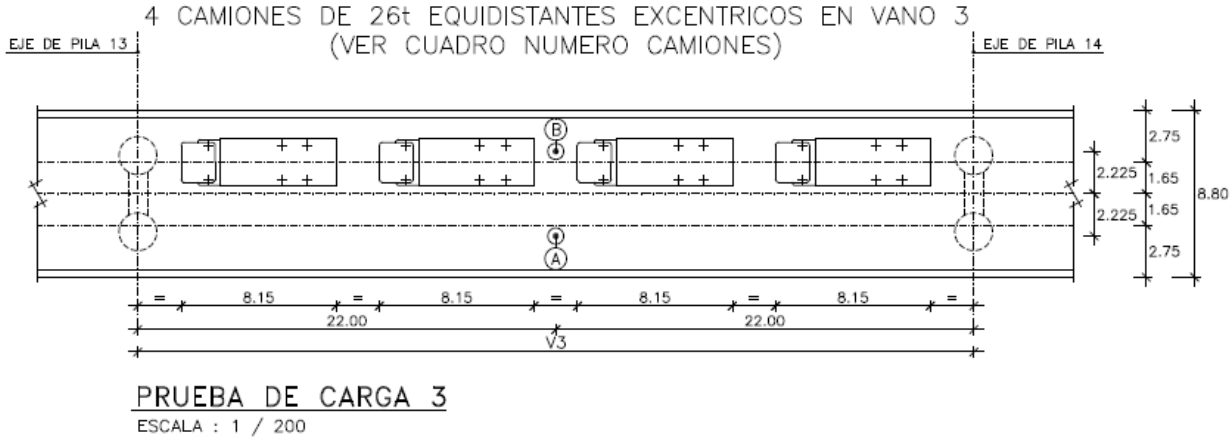

Figura 4.15 Prueba de carga 3 realizada en la estructura.

En último lugar, para finalizar este apartado, se presentan a continuación los resultados de las pruebas de carga. Recordemos que estas pruebas de carga son pruebas de carga estáticas, lo que significa que los resultados obtenidos son desplazamientos, medidos en milímetros (mm).

Se muestra la primera prueba de carga. Tal y como se aprecian en la Figura 4.13, 4.14 y 4.15 los puntos de estudios A y B se encuentran en el centro de cada vano, el punto A en un lateral, y el punto B en el otro. Hay un total de 30 puntos de estudio, 24 puntos de estudio en los primeros 12 vanos (2 por cada vano, A y B) y los últimos 6 puntos se encuentran en el vano 13 distribuidos de la siguiente manera (Figura 4.16):

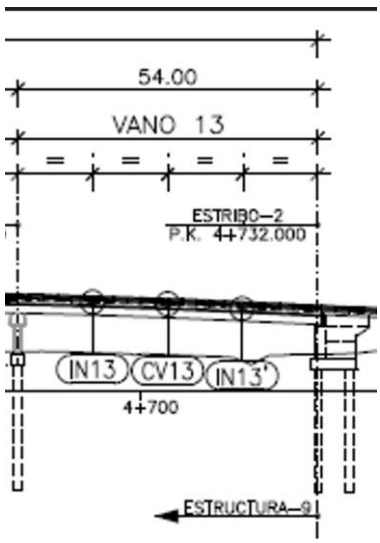

Figura 4.16 Distribución de los puntos de estudio en el vano número 13.

En esta figura apreciamos que los puntos están distribuidos de manera equidistante. El primer par de puntos de estudio de este vano se encuentra a un tercio del vano, el segundo a la mitad, y el último se encuentra a un tercio del final del vano. Por lo tanto, suman un total de 6 puntos de estudio a lo largo del vano, lo que significa que tenemos un total de 30 puntos de estudio. Por tanto, los resultados experimentales obtenidos de la primera prueba de carga, a partir de los cuáles realizaremos el proceso de optimización, son los siguientes (Tabla 4.1):

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
1	-50	22	-7	2	-1	0	0	0	0	0	0	0	0	0	0
2	21	-40	18	-6	2	-1	0	0	0	0	0	0	0	0	0
3	-7	18	-40	18	-6	2	0	0	0	0	0	0	0	0	0
4	2	-6	18	-40	18	-6	2	-1	0	0	0	0	0	0	0
5	-1	2	-6	18	-40	18	-6	2	-1	0	0	0	0	0	0
6	0	-1	2	-6	18	-40	18	-6	2	-1	0	0	0	0	0
7	0	0	-1	2	-6	18	-40	18	-6	2	-1	0	0	0	0
8	0	0	0	-1	2	-6	18	-40	18	-6	2	-1	0	0	0
9	0	0	0	0	-1	2	-6	18	-40	18	-6	2	-1	0	0
10	0	0	0	0	0	-1	2	-6	18	-39	17	-4	3	0	0
11	0	0	0	0	0	0	-1	2	-6	17	-37	13	-8	-7	-5
12	0	0	0	0	0	0	0	0	1	-4	11	-18	14	12	9
13	0	0	0	0	0	0	0	0	-1	3	-10	21	-72	-47	-56

Tabla 4.1 Resultados en mm de la primera prueba de carga (13 casos de carga x 15 ptos. de estudio).

Esta tabla pertenece a la primera prueba de carga. Es una tabla 13x15 en la que cada fila significa cada caso de carga, y cada una de las columnas significa los 15 puntos de estudio que hay a lo largo de toda la estructura, puesto que esta tabla es únicamente el punto A de cada vano. Los resultados de los puntos pertenecientes al lateral de B son muy similares a los del punto A. Varían apenas varios milímetros en los últimos 2 casos de carga.

Se muestran a continuación los resultados de la segunda prueba de carga. Recordemos que esta segunda prueba de carga tiene 1 caso menos de carga que la primera prueba. Se presentan a continuación en la Tabla 4.2 los resultados de los puntos A de cada vano:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	-23	-15	9	-3	1	1	0	0	0	0	0	0	0	0	0
2	11	-18	-17	9	-3	1	0	0	0	0	0	0	0	0	0
3	-4	9	-18	-17	9	-3	1	0	0	0	0	0	0	0	0
4	1	-3	9	-17	-17	9	-3	1	0	0	0	0	0	0	0
5	0	1	-3	9	-17	-17	9	-3	1	0	0	0	0	0	0
6	0	0	1	-3	9	-17	-17	9	-3	1	0	0	0	0	0
7	0	0	0	1	-3	9	-17	-17	9	-3	1	0	0	0	0
8	0	0	0	0	1	-3	9	-17	-17	9	-3	1	0	0	0
9	0	0	0	0	0	1	-3	9	-17	-17	9	-2	1	0	0
10	0	0	0	0	0	0	1	-3	9	-18	-16	7	-4	0	0
11	0	0	0	0	0	0	0	1	-4	11	-21	-4	4	4	3
12	0	0	0	0	0	0	0	0	0	0	1	2	-46	-28	-37

Tabla 4.2 Resultados en mm de la segunda prueba de carga (12 casos de carga x 15 ptos. de estudio).

Nuevamente, los puntos de estudio del lateral de B tienen resultados también similares a los de A, a excepción de la última fila. En último lugar, se presentan los resultados de la tercera prueba de carga, un único caso de carga en el vano número 3.

Los resultados en A son los siguientes. En cuanto a los resultados de B, varía un par de milímetros en uno de los puntos de estudio (Tabla 4.3):

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
3	-4	9	-19	9	-3	1	0	0	0	0	0	0	0	0	0

Tabla 4.3 Resultados en mm de la tercera prueba de carga (1 caso de carga x 15 ptos. de estudio).

4.2 Modelo de Elementos Finitos

En este apartado se explica el modelo de Elementos Finitos empleado en ANSYS. Dicho programa realizará el cálculo de desplazamientos, a través de la implementación del modelo que se explica a continuación, que se producen en los 30 puntos de estudio que hay en la estructura tal y como se comenta en el apartado 4.1. Estos desplazamientos se calculan a partir de unos datos de entrada, que son los módulos de Young (e30, e40, e50 y e210), variables críticas de estudio.

a. Introducimos a través de un fichero externo de datos de entrada
b. Se definen varios parámetros como por ejemplo el número de puntos de estudio, número de vanos, etc.
c. Se definen los elementos de nuestro modelo
d. Se definen las propiedades del material junto con las constantes reales
e. Determinamos las distintas secciones que formarán parte de los vanos del puente
f. Se define la geometría de la estructura. Nodos, 'keypoints' (puntos clave), líneas, etc.
g. Se procede a mallar el modelo para su posterior resolución
h. Establecemos las condiciones de contorno del problema (aquí entra por ejemplo la rigidez longitudinal del vano número 13)
i. Resolución del problema (se establecen condiciones de carga, se resuelve y se exportan resultados)

Este es el esquema general que se sigue en el modelo de Elementos Finitos. A continuación, desarrollaremos cada una de las partes mencionadas. En el apéndice A.1 se encuentra el código en su totalidad.

a. Introducimos a través de un fichero externo de datos de entrada

Al principio del código del modelo de la estructura, introducimos los datos de entrada a través de un fichero externo de texto (llamémoslo 'input_data'). Dicho fichero incluye las 4 variables que será objeto de estudio. El módulo de Young del acero y los módulos de elasticidad a compresión de los diferentes hormigones que conforman el puente. Hormigones de 30 GPa, 40 Gpa y 50 Gpa.

b. Se definen varios parámetros de gran relevancia como por ejemplo el número de puntos de estudio, número de vanos, etc.

Definimos el número de vanos del puente ('NumberOfSpans'), las longitudes de cada vano del puente, así como sus respectivas coordenadas en el modelo en ANSYS ('LengthOfSpan' y 'CoordOfSpan' respectivamente). Por otro lado, el número de puntos de estudio ('NumberOfNodesPost'), que suman un total de 15. El parámetro 'ElementLength' será de utilidad para el momento del mallado del modelo. En último lugar, ancho del tablero y propiedades del pavimento.

c. Se definen los elementos que serán el pilar fundamental de nuestro modelo

En el apartado 4.3.7 se explica en profundidad la razón por las que se escoge el elemento MPC184. Sin embargo, los otros dos elementos son los siguientes:

BEAM188 porque son elementos viga que nos servirán en gran medida para modelar el comportamiento del puente a flexión, torsión y deformación axial, aunque nos interese principalmente el comportamiento a flexión. Sin embargo, es un elemento que representa de manera precisa y detallada el comportamiento estructural de un puente.

LINK11 puesto que en el vano 13 hay un muelle longitudinal que aportará rigidez en esa dirección. Por lo tanto, este elemento nos ayudará a modelar dicho muelle.

d. Se definen las propiedades del material junto con las constantes reales

Definimos el acero y los 3 hormigones que serán objeto de estudio en el proceso de optimización, un total de 4 variables. En el caso del acero, tiene un módulo de Young de e210 GPa (variable a determinar evidentemente, la proporcionará el algoritmo en cada iteración), un coeficiente de Poisson de 0.3 y una densidad de 7850 kg/m^3. En el caso de los hormigones, elasticidad a compresión de e30, e40 y e50 GPa respectivamente. Tienen un coeficiente de Poisson de 0.2 y una densidad de hormigón de 2500 kg/m^3.

Por otro lado, definimos la constante real del muelle longitudinal que se encuentra en el vano número 13.

e. Determinamos las distintas secciones que formarán parte de los vanos del puente

En este punto definiremos las secciones de la estructura. Ésta cuenta con un total de 16 secciones a lo largo de su longitud. Por lo tanto, definimos 16 secciones con SECTYPE. Desde ficheros externos, vamos leyendo cada una de las 16 secciones con SECREAD y establecemos como punto de referencia de cada una de las secciones, el origen de coordenadas de cada una de estas secciones, a través de SECOFFSET. En último lugar, añadimos masa por unidad de longitud del pavimento (ancho tablero x espesor x densidad) con SECCONTROL.

Recordemos que:

- MAT 1: ACERO 210 GPa
- MAT 2: HORMIGÓN 30 GPa
- MAT 3: HORMIGÓN 40 GPa
- MAT 4: HORMIGÓN 50 GPa

Se muestran a continuación las secciones más relevantes en este modelo de Elementos Finitos (Figura 4.17, 4.18 y 4.19):

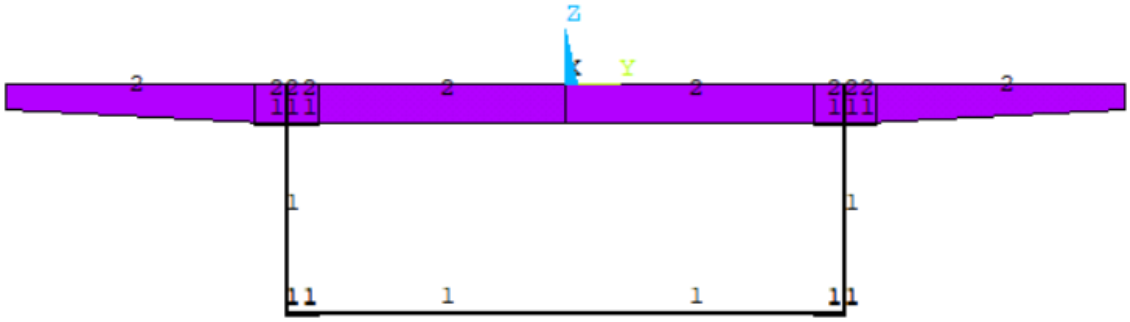

Figura 4.17 Sección de viga en cajón.

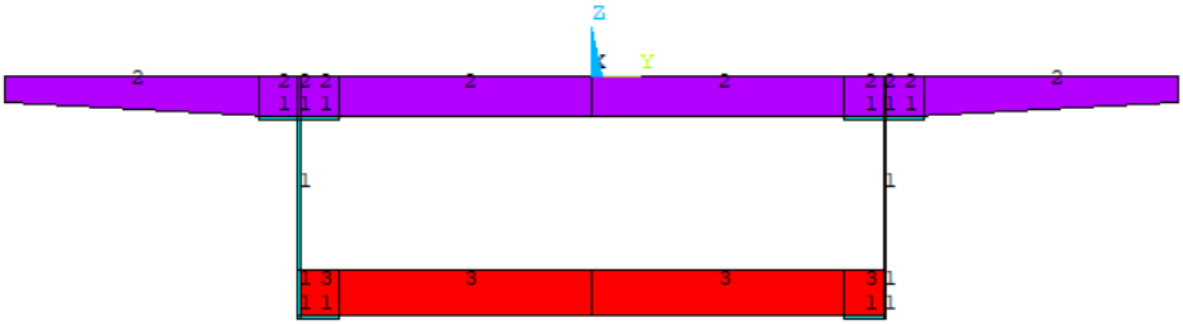

Figura 4.18 Sección de viga en cajón.

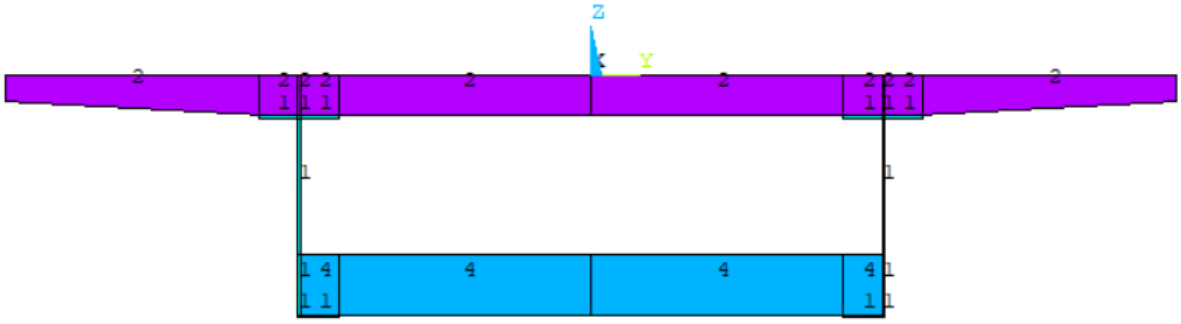

Figura 4.19 Sección de viga en cajón.

Estas son las tres secciones significativas del modelo. En la primera de ellas (Figura 4.17) podemos observar resistencia proporcionada por el hormigón e30 en la zona superior y sin hormigón en la zona inferior. En la segunda de las secciones (Figura 4.18), se añade una resistencia en la zona inferior proporcionada por el hormigón e40. En último lugar, en la Figura 4.19 apreciamos una resistencia proporcionada por un hormigón e50 en la zona inferior.

a. Se define la geometría de la estructura. Nodos, 'keypoints' (puntos clave), líneas, etc.

En el apéndice A.1 se pueden encontrar las 4 partes en las que se podría dividir la construcción de la geometría del modelo. Las cuatro partes son el vano 1, los vanos 2-11 (que tienen la misma geometría), el vano 12 y por último, el vano 13 que difiere significativamente del resto puesto que es un vano curvo y por ejemplo, experimenta torsión en mayor medida que los vanos anteriores.

En definitiva, lo que estamos haciendo con todos estos comandos es definir las líneas que forman parte de cada vano con el objetivo de mallarlas posteriormente. A cada línea se le asigna un tipo de elemento (en este caso, elementos viga) junto con la sección que corresponde a cada vano.

b. Se procede a mallar el modelo para su posterior resolución

En este punto se malla el modelo para poder resolverlo. Para ello, mallamos todas las líneas creadas en el punto anterior con una longitud de división de 'ElementLength'. Y por otro lado, creamos una serie de puntos 'npt' que serán los puntos de estudio en los que extraeremos los resultados (desplazamientos, giros, etc.).

c. Establecemos las condiciones de contorno del problema (aquí entra por ejemplo la rigidez longitudinal del vano número 13)

Restringimos ahora desplazamientos en nodos seleccionados para poder resolver el problema. Estas condiciones de contorno simulan con una alta precisión el comportamiento de los apoyos de la estructura en la vida real. En el apartado 4.1 hemos visto los tipos de apoyos que tiene la estructura. En este apartado se modelizan en el programa de ANSYS. Se pasa de un modelo físico a un modelo digital. Se restringen desplazamientos en la dirección vertical (dirección del eje Z) en todos los apoyos de cada vano. En la dirección positiva del eje Y (lateral izquierdo del puente visto desde arriba), se restringe desplazamientos en esta misma dirección, así como en dirección z también. En el lateral contrario solo se restringe en dirección z en cada uno de los apoyos. Además, en el último vano, se restringen los desplazamientos y giros en todas las direcciones.

d. Resolución del problema (se establecen condiciones de carga, se resuelve y se exportan resultados)

Para resolver el problema de Elementos Finitos, una vez establecidas las condiciones de contorno, debemos establecer los casos de carga, que dependerán de la prueba de carga que queramos evaluar, puesto que cada una de las pruebas de carga contiene diferentes casos de carga. Introducimos un fichero externo que calculará y extraerá los resultados a un fichero de texto, que posteriormente, será implementado en MATLAB para ejecutar el proceso de optimización. El fichero 'prueba_carga_aro_surrogate' se divide en 2 partes. La primera de ellas se encargará de establecer los resultados de carga para cada prueba de carga respectivamente y la segunda parte, extraerá los resultados obtenidos para cada prueba de carga respectivamente de nuevo.

Se muestra a continuación el fichero 'prueba_carga_aro_surrogate'. En este fichero hay varios parámetros que dependen de la prueba de carga de interés. En el proceso de optimización se analizan casos de carga individuales, pruebas de carga completas, y análisis completo de las 3 pruebas de carga simultáneamente. Por lo tanto, aquí se muestra por ejemplo el estudio completo de la tercera prueba de carga, es decir, un solo caso de carga simultáneamente (caso de carga del vano 3). El código completo se encuentra en el apéndice A.1.

En primer lugar, 'iPrueba' tomará un valor de 3 por ser la tercera prueba de carga. Por lo tanto, solo si el parámetro 'lSpan' vale 3, y no otro valor, se establecen las cargas correspondientes tal y como se explica en el apartado 4.1 de las pruebas de carga. Se aplica una presión y un momento alrededor del eje x, a lo largo de todo el vano número 3.

Por último, la segunda parte de este punto (en el módulo post procesador) es extraer los resultados obtenidos en el vano 3 (en este caso). Para ello, generamos los puntos de estudio en los que queremos obtener los resultados y a través del comando '*vwrite' le pedimos a ANSYS que nos genere un fichero de texto de salida en el que se

recogen los desplazamientos, giros, etc.

Se muestra a continuación en la Figura 4.20, 4.21 y 4.22 una representación del modelo implementado en ANSYS. En la Figura 4.20 observamos la estructura al completo, vista desde arriba. Se aprecian los 13 vanos que conforman el puente con sus respectivas modelizaciones de apoyos.

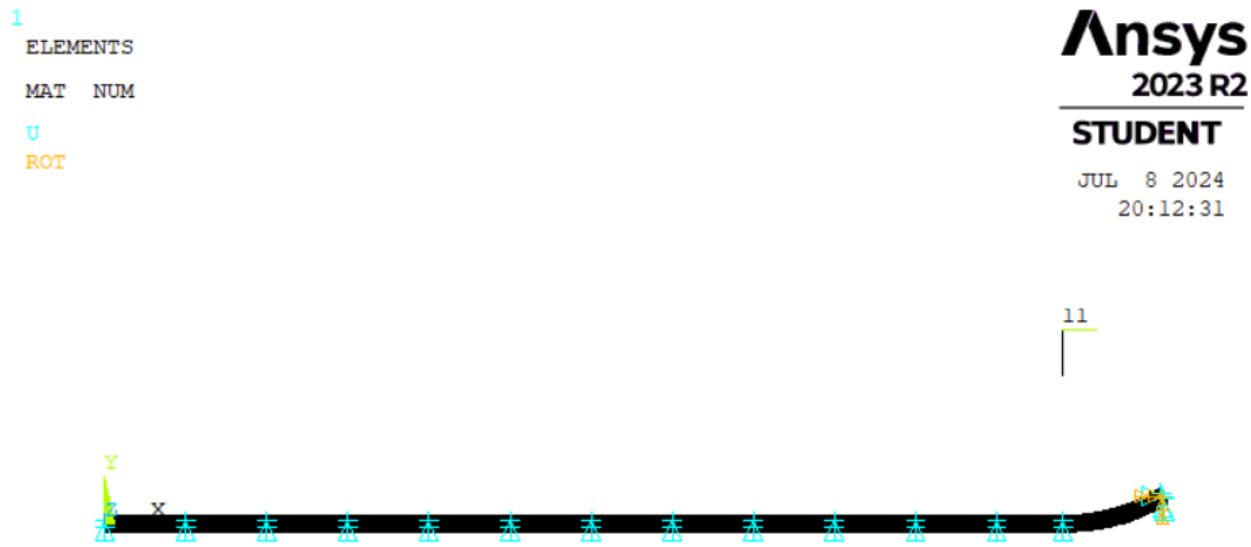

Figura 4.20 Modelo de la estructura en ANSYS. Vista desde arriba.

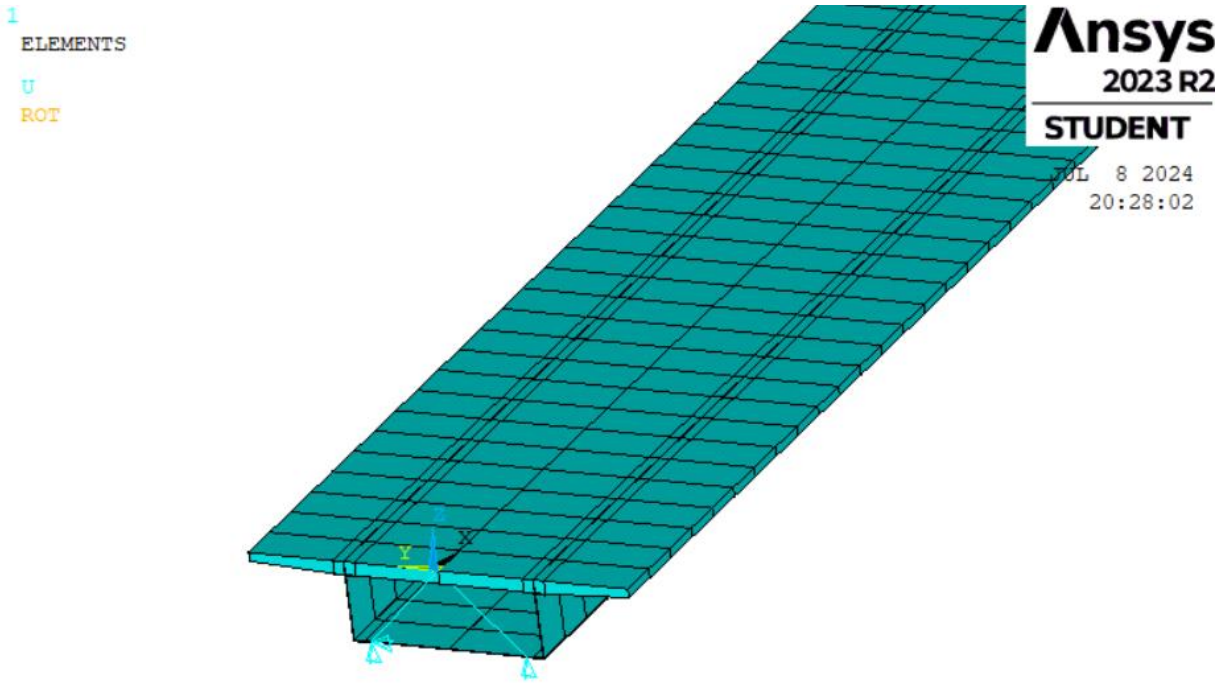

Figura 4.21 Modelo del primer vano de la estructura en ANSYS. Vista oblicua.

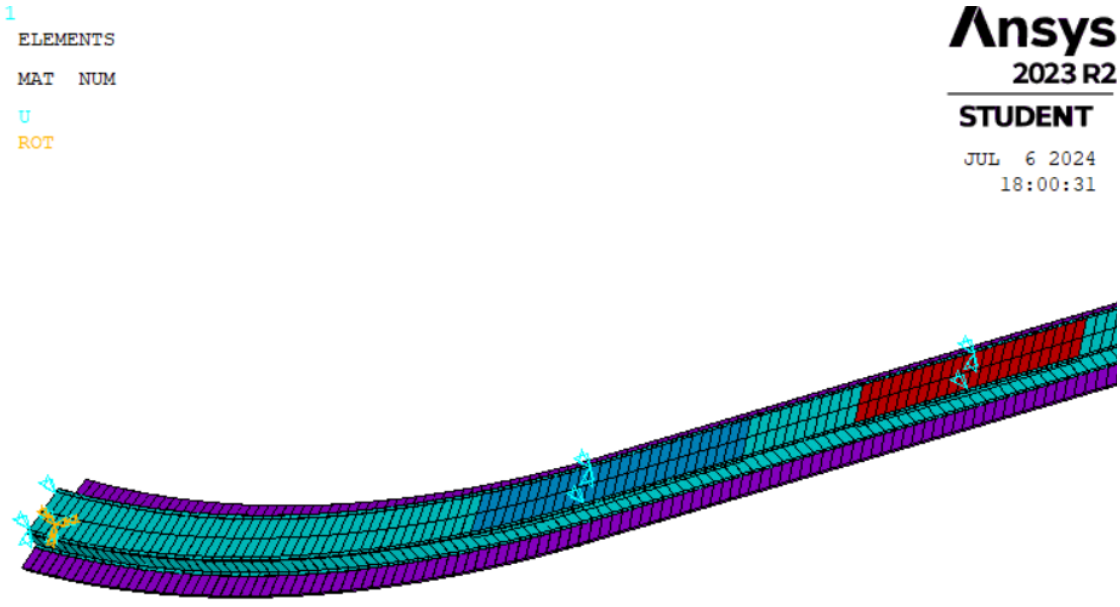

Figura 4.22 Secciones características de la estructura. Hormigón 30 GPa (morado), 40 GPa (roja) y 50 GPa (azul).

En la Figura 4.21 se muestra una sección cualquiera de la estructura, para apreciar bien la sección en cajón junto con el tablero. En la figura 4.22 se muestra las 3 secciones características vistas desde abajo para una mejor comprensión. Se puede apreciar la sección en la que únicamente está la sección en cajón de acero. Por otro lado, la sección compuesta por un hormigón de 40 GPa (rojo) en la parte inferior de la sección en cajón (Figura 4.18) y por último, una sección compuesta por un hormigón de 50 GPa (azul) en la parte inferior de la sección en cajón (Figura 4.19).

4.3 Optimización

La optimización del modelo en estudio se ha realizado con MATLAB a través de la implementación de un código que se caracteriza por los parámetros que a continuación se describen:

- Función objetivo (fobj)
- Límites superiores (ub) e inferiores (lb) de los parámetros de entrada x = [e210 e30 e40 e50 kx]
- Generador de números aleatorios (rng)
- Opciones del algoritmo de resolución (optimoptions)
- Algoritmo de resolución (surrogateopt)

4.3.1 Parámetros principales del proceso de optimización

En el código implementado tenemos en cuenta los parámetros mencionados. Se ha partido de una hipótesis inicial de valores de cada uno de estos parámetros para empezar a realizar la optimización. Por tanto, establecemos como función objetivo inicial:

$$Fobj = (Fobj + sqrt(Phi' * Phi))/(2 * nPost * Precision * nSpan)$$

siendo,

- `Fobj`, la función objetivo
- `Phi,` la diferencia entre desplazamientos experimentales y de elementos finitos en los 30 puntos de estudio que tiene la estructura
- `nPost,` el número de puntos de estudio, que son 15
- `Precision`, exactitud o calidad de resultado de las pruebas de carga realizadas sobre la estructura. Su valor es 1 mm (1e-3)
- `nSpan`, número de vanos de los que consta la estructura, 13 vanos

En un proceso de optimización, establecemos una función objetivo para definir claramente lo que estamos tratando de maximizar o minimizar. Esta función objetivo representa la medida cuantitativa del éxito o el rendimiento que queremos optimizar en relación con ciertas variables o restricciones. En otras palabras, la función objetivo es una forma de tener una métrica para saber si el proceso de optimización evoluciona de mejor o peor manera. Por tanto, para una correcta elección de la función objetivo, han de tenerse en cuenta las siguientes consideraciones.

Debemos determinar las métricas de rendimiento que están directamente relacionadas con nuestros objetivos. Es decir, en un proceso de optimización en el que se trata de calcular 5 variables a partir de unos resultados experimentales, una buena métrica sería por ejemplo, la diferencia entre resultados analíticos y reales o experimentales.

Por otro lado, una vez que se ha seleccionado la métrica de rendimiento correspondiente, debemos determinar los objetivos en términos matemáticos para crear una función objetivo. En otras palabras, la maximización o minimización de la métrica de rendimiento seleccionada.

Por tanto, una buena opción para empezar a definir una función objetivo es la definición de error. Es decir, la diferencia entre 2 valores cualesquiera. La raíz cuadrada de la norma de la variable '`Phi`'. Al incluir el término `(2*nPost*Precision*nSpan)` en el denominador, estamos normalizando la función objetivo de acuerdo con la cantidad de puntos estudiados, la precisión de las pruebas de carga realizadas y el número de vanos de la estructura.

Las variables de estudio en este problema de optimización serán los módulos elásticos a compresión de los diferentes hormigones que conforman la estructura. Las variables e30, e40, e50 respectivamente y e210 que es la resistencia a tracción del acero. Dichos parámetros tendrán los siguientes límites que establecemos por dos razones. La primera de ellas por ahorro de tiempo de cómputo, y en segundo lugar, porque conocemos las propiedades del puente y sabemos que el valor que vamos a obtener está precisamente dentro de estos límites. Los valores nominales son 30 GPa, 40 GPa, 50 GPa y 210 GPa respectivamente. El límite superior (ub) será 1.05 veces el valor nominal. El límite inferior (lb) será 0.8 veces el valor nominal a excepción del caso de la variable e50 que se establece en 0.7 por razones que se verán posteriormente en el apartado 4.3.7.

ub = [1.05 1.05 1.05 1.05]

lb = [0.8 0.8 0.8 0.7]

En cuanto a los límites superiores e inferiores, el planteamiento inicial son límites superiores 5 veces mayor al valor nominal (valor nominal = [210e9 30e9 40e9 50e9 1e8]), el límite inferior en 10 veces menor al valor nominal. La hipótesis inicialmente planteada es un rango amplio en el que estamos seguros de que se encontrarán los valores de las 5 variables que buscamos encontrar.

El parámetro generador de números aleatorios `rng` se ha establecido en el valor predeterminado que MATLAB ofrece, 'default', debido a varias razones:

1. Consistencia: Utilizar el valor predeterminado garantiza que los resultados sean consistentes entre diferentes sesiones y diferentes usuarios (diferentes usuarios y un mismo 'script' garantiza gracias a rng default, los mismos resultados), lo que facilita la reproducibilidad de los resultados.

2. Compatibilidad: Muchos códigos y funciones de MATLAB están diseñados para funcionar mejor con el valor predeterminado del rng. Cambiar el valor puede afectar la forma en que se comportan ciertas funciones, especialmente aquellas que utilizan generación de números aleatorios.

3. Confianza en los resultados: Al utilizar el valor predeterminado del generador de números aleatorios (rng) en MATLAB, podemos tener confianza en la integridad de los resultados obtenidos. Esto significa que podemos estar seguros de que los resultados no están influenciados por la elección de un rng específico.

El 4º parámetro que tenemos en cuenta, 'optimoptions' son las opciones del problema de subrogación que queremos definir. Posteriormente, comprenderemos el sentido de todos y cada uno de los conceptos y parámetros que hemos definido.

```
('surrogateopt','ObjectiveLimit',0.010,'MaxFunctionEvaluations',
                MFE,'PlotFcn','surrogateoptplot')
```

Estos son los módulos de las opciones de subrogación que serán diferentes de sus valores por defecto.

- '`ObjectiveLimit`' es la tolerancia en el valor que toma la función objetivo. Es decir, si el valor de la función objetivo en el punto de estudio es menor que el valor de 'ObjectiveLimit', el algoritmo se para automáticamente. Escogemos como valor de este módulo, 0.010 debido a la siguiente razón:

El valor máximo (mínimo en términos matemáticos) que puede tomar la función objetivo lo podemos determinar de manera arbitraria o según criterios específicos establecidos por nosotros como parte del proceso de optimización. Esta decisión puede basarse en consideraciones sobre los límites físicos o prácticos del problema, la calidad deseada de la solución, o cualquier otro factor relevante para el contexto en el que se aplica la optimización. Al establecer un valor máximo para la función objetivo, estamos definiendo un límite superior para la búsqueda de soluciones óptimas dentro del espacio de búsqueda definido por el problema de optimización. Por lo tanto, una primera hipótesis podría ser fijar el valor de la precisión que queremos obtener. Por ejemplo, hemos decidido tomar una precisión de 0.1 mm. Es decir, 1e-4.

Si el término `sqrt (Phi'*Phi)` toma como valor 1e-4, considerando los valores fijos del denominador de la función objetivo, obtenemos un valor máximo (mínimo en términos matemáticos) de la función objetivo de 0.0033.

Una vez aproximado el valor máximo que tomará la función objetivo en nuestro proceso de optimización, podemos determinar el valor inicial que vamos a considerar en el parámetro '`ObjectiveLimit`'. Sabiendo que el valor aproximadamente máximo que vamos a obtener de la función objetivo es 0.0033, no es mala opción escoger por ejemplo como '`ObjectiveLimit`' el valor 0.010, porque está cerca del 0.0033 máximo que vamos a obtener y no es una mala precisión, además de que implementamos dicho modelo para que en el caso de que se supere dicho umbral, se ahorre coste computacional en el cálculo, así como evitar la sobreoptimización.

- A su vez, el módulo '`MaxFunctionEvaluations`' nos permite también ahorrar coste computacional en el cálculo si no se ha superado el umbral de 'ObjectiveLimit'. 'MaxFunctionEvaluations' significa que el algoritmo se detendrá automáticamente cuando llegue al número de iteraciones establecido 'MFE'. Dicho valor puede ser 50, 100, 200, etc. Lo que convenga según cada caso de estudio.
- Por último, '`PlotFcn`' es un módulo de las opciones que te permite 'plotear' el progreso del solucionador, es decir, los valores de 'Fobj' que se van obteniendo en cada iteración.

El último parámetro a tener en cuenta en el proceso de optimización es su algoritmo de resolución. '`surrogateopt`' es un solucionador global para funciones objetivos que consumen mucho tiempo. Dicho solucionador cuenta con unos parámetros de entrada previamente mencionados y unos parámetros de salida que describimos a continuación:

```
[x,fval,exitflag,output,trials] = surrogateopt(_)
```

- '**`x`**' es la solución de nuestro problema de optimización. Son los valores de las variables que queremos calcular, [e210 e30 e40 e50 kx].
- '**`fval`**' es el valor que toma la función objetivo para los valores de '**`x`**'.
- '**`exitflag`**' es la razón por la cual se detiene el algoritmo de cálculo. Toma un valor concreto. Dependiendo del valor que tome, sabremos que el algoritmo de resolución ha sido detenido por una razón u otra. Puede tomar como valor 10, 3, 1, 0, -1, -2. Por ejemplo, si toma 1 como valor, significa que el algoritmo de resolución ha sido detenido por llegar a '`MaxFunctionEvaluations`' y no seguirá iterando.
- '**`output`**' es información sobre el proceso de optimización, devuelto como una estructura de datos, que contiene por ejemplo el tiempo de computación empleado en el proceso de resolución, el número total de evaluaciones realizadas, etc.
- '**`trials`**' tal y como se ha comentado anteriormente en el apartado 3.2, es una estructura de datos también, que incluye únicamente 2 parámetros, la solución '**`x`**' y su respectiva imagen '**`fval`**'. Guarda estos 2 parámetros de todas y cada una de las iteraciones que se realicen en el proceso de resolución.

Describimos a continuación, el funcionamiento del proceso de optimización de manera general, y procederemos a realizar cálculos de manera más particular hasta llegar a una solución global.

El código implementado para resolver el problema de optimización consta de dos funciones. La función '`runansys`' y la función '`obj_f`'. El objetivo de la primera función es evaluar los valores de entrada en ANSYS con el fin de obtener los resultados a partir de elementos finitos. Una vez obtenidos estos resultados, los evaluamos en la función objetivo '`obj_f`' para conocer el valor de la función objetivo de los resultados previamente recogidos. El objetivo es minimizar el valor de dicha función objetivo.

Para ello, tal y como se ha mencionado en el apartado 2.2, en cada iteración, el algoritmo se encargará de generar valores aleatorios de los parámetros de entrada de manera que, teniendo en cuenta la nube de puntos que con cada iteración va almacenando el algoritmo, el valor de la función objetivo disminuya conforme avance el número de puntos estudiados.

Antes de proceder con el cálculo, dotaremos de sentido y coherencia a los conceptos anteriormente comentados. Para obtener unos resultados plausibles hemos realizado una serie de análisis, que nos permiten validar que los resultados globales son coherentes a partir de los resultados locales.

En otras palabras, empezaremos estudiando un caso de carga de una de las pruebas de carga, posteriormente, estudiaremos todos los casos de carga de la respectiva prueba de carga. Al finalizar la prueba de carga al completo, estudiaremos por completo el resto de las pruebas de carga. Una vez realizados todos los cálculos, procederemos a estudiar las 3 pruebas de carga al mismo tiempo, con el objetivo de conseguir la solución del problema global.

4.3.2 Parámetros fundamentales para garantizar una comparación de resultados precisa

Los cálculos mencionados carecen de sentido si no se definen los parámetros anteriormente mencionados. Para poder realizar un contraste de resultados adecuado y preciso, el proceso de optimización que ha de llevarse a cabo debe contar con el mismo valor en los parámetros siguientes:

- `rng default` debido a que en el proceso de optimización, la reproducibilidad de los resultados ha de ser la misma en los diferentes cálculos que ejecutemos. En el caso de utilizar '`rng`' diferentes, no se generarán los mismos números aleatorios y perderá la reproducibilidad por lo que no se obtendrán los mismos resultados, además de que el rendimiento será peor, y no generará números aleatorios de calidad.
- Límites '`lb`' y '`ub`'. La importancia de mantener los mismos límites para las diferentes evaluaciones a ejecutar nos garantiza que el algoritmo generará números aleatorios dentro del mismo rango. Es decir, el algoritmo generará números aleatorios dentro de [(1/10)*ubnominal , 5*ubnominal], siendo ubnominal=[210e9, 30e9, 40e9, 50e9, 1e8].
- En cuanto a '`MaxFunctionEvaluations`' y '`ObjectiveLimit`' han de tener el mismo valor en cada uno de los procesos de optimización que vamos a llevar a cabo. Esto se debe a que el criterio de parada del algoritmo debe ser el mismo porque ha de asegurarse una comparación de resultados justa entre las evaluaciones particulares y las globales.

Para poder afrontar correctamente el problema de subrogación, tal y como se ha mencionado antes, partimos de soluciones particulares a soluciones globales. Para ello, empezamos estudiando la prueba de carga número 1 (`iPrueba=1`), y en ella, obtenemos las soluciones de cada caso de carga individualmente. Es decir, trataremos de optimizar la función objetivo para la prueba de carga número 1, para el caso de carga número 1 (`iPrueba=1 && lSpan=1`). Esto implica el siguiente proceso:

1. El algoritmo selecciona un punto aleatorio. Cuando hablamos de un punto, nos referimos a 5 variables/parámetros de entrada, que son [e210, e30, e40, e50, kx].
2. Entramos en la función '`runansys`' con estos 5 parámetros de entrada y calculamos el fichero '`prueba1_1.txt`' que incluye los desplazamientos y giros de los 15 puntos de estudio establecidos. Obtenemos el valor de '`ResultadoFe`' (desplazamientos en la dirección vertical, UZ) que guardamos en

'trials'. Evaluamos dichos puntos en la función objetivo 'obj_f'.

3. En la función 'obj_f' simplemente calculamos el valor de función objetivo correspondiente para los datos de entrada que nos ha proporcionado el algoritmo.
4. Una vez hechos estos cálculos, MATLAB procede a realizar la siguiente iteración, el algoritmo nos vuelve a proporcionar otros valores nuevos para las nuevas variables de entrada, volvemos a calcular los desplazamientos, evaluamos en la función objetivo, etc.
5. Se repite el proceso hasta que el algoritmo se pare automáticamente porque ha llegado al 'MaxFunctionEvaluations' o ha obtenido un valor menor a 'ObjectiveLimit'.

El código que hemos implementado para el cálculo de '**x**' y '**fval**' de los 13 casos de carga de la primera prueba de carga es el que se muestra a continuación. Para simplificar el cálculo y ahorrar coste de computación, recorremos con la variable 'kSpan' los 13 casos de carga.

En el código de EF también se utiliza la variable 'kSpan' para poder realizar los cálculos. Para guardar los resultados de los 13 casos de carga hemos creado la variable 'file_name2' que guarda en un fichero con extensión '.mat' los resultados de los 13 casos de carga ('resultados5.mat').

Por lo tanto, a modo de conclusión, en el apéndice A.5 se muestra el código de MATLAB llevado a cabo en el proceso de optimización, donde hemos implementado los parámetros anteriormente comentados, del estudio inicial realizado.

4.3.3 Resultados de las pruebas de carga al completo (aproximación inicial)

En este apartado analizaremos los resultados que se obtienen del modelo inicial estudiado empleando una serie de variables de estudio, opciones determinadas de la función de subrogación empleada, etc. Estos resultados nos han ayudado a evaluar y modificar este modelo inicial del que partimos, con el objetivo de llegar a unos resultados coherentes. Se han evaluado las pruebas de carga individualmente, así como cada una de ellas al completo pero independientes. Posteriormente se ha estudiado las tres pruebas de carga al completo y se han obtenido los siguientes resultados. Se muestran dichos resultados puesto que son los más relevantes para modificar el modelo inicial empleado. A raíz de estos resultados se redactan las correspondientes conclusiones.

Los resultados obtenidos son los siguientes:

Prueba TOTAL SUBROGACIÓN						
Fobj_mín	**X_optím**					**elapsed_time**
0.522143704573266	1.0076e+11 Pa	1.0841e+11 Pa	1.7883e+11 Pa	1.0328e+11 Pa	2.9374e+08 Pa	5970.952266 s

Tabla 4.4 Resultados totales correspondiente a las tres pruebas de carga en su conjunto (aproximación inicial).

Obtenemos los resultados mostrados en la Tabla 4.4. Con dichos resultados, sacamos las siguientes conclusiones:

- El valor de la función objetivo es poco preciso, lejos del valor mínimo establecido de 0.010. Teóricamente, la función objetivo ha de ser lo mínimo posible para poder afirmar con evidencia empírica que hemos obtenido unos resultados válidos y aceptables. No obstante, no necesariamente es siempre así, como podemos apreciar, los valores de las funciones objetivo en los casos individuales son muy bajos, cerca del 'ObjectiveLimit', y sin

embargo, la gran mayoría de los casos de carga tienen unos valores de parámetros de '**x**' muy parecidos a los obtenidos en esta prueba. Lo que implica cierta convergencia. No deja de ser una prueba de carga de las 3 existentes.

- En cuanto a los parámetros de la solución, analizamos primeramente el valor del módulo de elasticidad del acero. Tal y como esperábamos, se obtiene un valor bajo de módulo de elasticidad (100 GPa), parecido a los valores obtenidos en los casos individuales. El único punto positivo es que todo parece converger. Sin embargo, si finalmente e210, e30, e40, e50 y kx mantienen estos valores en la solución global, los resultados no son congruentes y por lo tanto nos veríamos obligados a recalibrar el modelo.
- En cuanto a las resistencias a compresión de los hormigones e30, e40 y e50, al igual que el módulo de Young del acero, muy parecidas a las del análisis individual. Obtenemos e30 alrededor de los 110 GPa, el e40 próximo a los 180 GPa, y por último, el e50 muy lejano a su valor nominal, 103 GPa. Vemos que son valores desorbitados, no encaja con los valores de compresión del hormigón que esperamos, valores con los que se ha construido el modelo. Es un claro ejemplo de sobredimensionamiento de armadura de hormigón y dimensionamiento pobre en acero. En este caso, el modelo desarrollado no se ajustaría a la realidad y puede no ser válido si finalmente los resultados globales son estos.

En conclusión, hemos observado resultados consistentes con nuestras expectativas y con los hallazgos de estudios previos. Globalmente, encontramos que el acero presenta una baja resistencia a tracción, alrededor de 100 GPa. Además, los hormigones exhiben un módulo elástico a compresión notablemente alta, con valores de 100 GPa para el e30 y el e50, y 180 GPa para el e40. La variable de rigidez longitudinal (kx) ha mostrado una estabilidad prácticamente constante de 2.8e8 N/m a lo largo de todas las evaluaciones realizadas. En cuanto al tiempo de computación, MATLAB ha tardado aproximadamente 1.66 horas, es decir, 1 hora y 40 minutos aproximadamente. Es un tiempo de computación ligero para un proceso de optimización de esta envergadura. Hemos de tener en cuenta las características previamente comentadas del ordenador en el que se ha realizado este Trabajo de Fin de Grado.

Sin embargo, estos resultados presentan discrepancias con los valores típicos de estas variables en la práctica real, lo que pone en duda la validez de los resultados obtenidos. Por lo tanto, consideramos que estos resultados no son válidos y, como consecuencia, estamos preparados para buscar una solución alternativa más sólida y fiable.

Tras examinar detenidamente los resultados obtenidos y la notoria discrepancia entre ellos, se plantean tres posibles escenarios a considerar: el riesgo de colapso de la estructura, la potencial imprecisión en las mediciones experimentales, o un posible desajuste en el modelo empleado. Al evaluar cada uno de estos escenarios, llegamos a la conclusión de que:

Los dos primeros escenarios se encuentran fuera del alcance de este Trabajo de Fin de Grado, sin embargo, no se vislumbra un peligro inminente de colapso en la estructura, dado el contexto actual y las condiciones en las que se encuentra el puente. Las evaluaciones indican que las bases estructurales son sólidas y no hay indicios de que estén comprometidas, lo que descarta esta posibilidad como la causa principal de la disparidad en los resultados.

Las mediciones experimentales, realizadas meticulosamente con la ayuda de sensores y técnicas adecuadas, parecen ser confiables y precisas. Esto sugiere que es poco probable que la discrepancia provenga de errores en la recopilación o interpretación de datos.

Por lo tanto, la causa más probable de la discrepancia radica en un posible desajuste en el modelo utilizado para predecir los resultados. En consecuencia, vamos a recalibrar el modelo revisando minuciosamente aspectos como la geometría, las condiciones de contorno, las cargas aplicadas y otros parámetros relacionados con el cálculo. Este proceso nos permitirá identificar posibles fallos y realizar los ajustes necesarios para mejorar la precisión y la fiabilidad del modelo, asegurando así que las futuras predicciones se alineen más estrechamente con los datos experimentales y las condiciones reales de la estructura.

La implementación del modelo inicial nos ha ayudado a identificar los problemas de los que partía nuestro modelo con el objetivo de corregir dichos errores y garantizar una evaluación mediante subrogación de manera correcta.

4.3.4 Recalibración del modelo

Para recalibrar el modelo, hemos de revisar exhaustivamente todos los parámetros y decisiones y parámetros que hemos incluido a lo largo de todo el cálculo realizado, tanto en ANSYS como en MATLAB. Por lo tanto, revisaremos el modelo implementado en ANSYS y el código que lleva a cabo el proceso de optimización en MATLAB.

Después de una exhaustiva revisión tanto del modelo implementado en ANSYS como del código MATLAB utilizado para el proceso de optimización, se han realizado evaluaciones detalladas en diversos aspectos. Se han revisado minuciosamente la geometría del modelo, las cargas aplicadas, las condiciones de contorno y las propiedades de la sección, incluyendo el material utilizado, los tipos de elemento (conocidos como 'element type' en ANSYS), así como las constantes reales ('real constants' en ANSYS). Además, se ha examinado el módulo de solución utilizado en ANSYS para asegurar su idoneidad en relación con el problema planteado.

En cuanto al código MATLAB, se ha llevado a cabo una revisión exhaustiva de los argumentos de entrada, las variables involucradas en la función de subrogación utilizada en el proceso de optimización, así como las opciones de dicha función. Se han evaluado las funciones objetivo y las funciones utilizadas para ejecutar ANSYS ('runansys'), así como los tiempos de computación y las variables relevantes para el cálculo.

Basándonos en estas evaluaciones, hemos llegado a las siguientes conclusiones y se han implementado los siguientes cambios:

El primer cambio efectuado es en el código del proceso de optimización en MATLAB, concretamente, la función objetivo. La nueva función objetivo nos aportará ahora una mejor interpretación de los resultados, así como un mayor sentido físico, más completo. Exponemos porqué:

La función objetivo se divide en dos partes. La primera parte, que es el numerador, calcula el error absoluto, es decir, la diferencia entre los resultados obtenidos mediante elementos finitos y los resultados experimentales. La segunda parte, que es el denominador, se utiliza principalmente para normalizar lo previamente calculado y dar un sentido físico a la función objetivo.

Al dividir por el número de puntos de estudio, que son 30, normalizamos la cantidad de puntos de estudio disponibles para el análisis de la estructura. Además, al dividir por 'lCase', que es el recuento de evaluaciones efectuadas en cada iteración, normalizamos el número de evaluaciones, lo que otorga coherencia y sentido físico a la función objetivo, expresándola en milímetros. Esto permite comparar directamente el valor de la función objetivo con las medidas en milímetros, asegurando que hemos establecido una métrica adecuada. Estas son las principales diferencias con la función objetivo definida en la aproximación inicial, que no contaba con un sentido físico claro.

Por lo tanto, después de entender la importancia de esta modificación, pasamos a definir nuestra nueva Fobj:

$$Fobj = (Fobj + sqrt(Phi' * Phi) * iPrueba^{-1})/(sqrt(2 * nPost) * lCase)$$

siendo,

- `Fobj`, la función objetivo
- `Phi,` la diferencia entre desplazamientos experimentales y de elementos finitos en los 30 puntos de estudio que tiene la estructura
- `nPost,` el número de puntos de estudio, que son 15
- `iPrueba,` el número de pruebas de carga realizadas, que son 3
- `lCase,` un parámetro de ponderación de las 3 pruebas de carga. Prueba número uno se pondera a la unidad. Prueba número dos se pondera a la mita. Prueba número tres se pondera un tercio del total

El cálculo de la función objetivo se desglosa en dos etapas distintas para asegurar su coherencia y relevancia física. En la primera etapa, se lleva a cabo la agregación de las funciones objetivo ponderadas, las cuales representan la contribución relativa de cada prueba de carga al resultado global. Aquí, el término

`(Fobj+sqrt(Phi'*Phi)*iPrueba^-1)` se ocupa de este proceso. Una vez completada esta suma ponderada, se procede a la segunda etapa del cálculo.

En la segunda etapa, se normaliza la función objetivo resultante dividiéndola por el producto de la raíz cuadrada del doble del número de puntos de medición posteriores (`nPost`) y la longitud característica del caso de estudio (`lCase`). Este paso de normalización es crucial para otorgar un significado físico al valor de la función objetivo, permitiendo una comparación apropiada entre diferentes casos de estudio y asegurando que el resultado final sea representativo en términos de las dimensiones y escalas del problema físico llevado a cabo.

En segundo lugar, el segundo ajuste efectuado es en el modelo implementado en ANSYS, en el que en la definición de tipos de elemento (‘type element’ según ANSYS) teníamos los siguientes tipos de elementos definidos anteriormente en la aproximación inicial:

ET, 1, BEAM188, , ,

ET, 2, MPC184, 1, ,

ET, 3, LINK11

Y en cuanto al reajuste realizado, tenemos la siguiente configuración de tipos de elementos:

ET, 1, BEAM188, , ,

KEYOPT, 1, 12, 1

ET, 2, MPC184, 1, ,

KEYOPT, 2, 2, 1

ET, 3, LINK11

La preferencia por utilizar la primera configuración, que incluye la especificación de KEYOPT para los elementos, se fundamenta en varios aspectos clave relacionados con el modelado y el análisis en ANSYS, así como con los métodos para establecer condiciones de contorno:

Control de comportamiento del elemento: Mediante la especificación de KEYOPT para los elementos BEAM188 y MPC184, se puede controlar de manera precisa y ajustar el comportamiento de estos elementos durante el análisis. Esto es especialmente relevante para optimizar la precisión y eficacia del modelo en situaciones específicas.

Imposición precisa de condiciones de contorno: La utilización de KEYOPT para el elemento MPC184 facilita la imposición de condiciones de contorno complejas o específicas, como restricciones cinemáticas o uniones múltiples, de manera precisa y controlada. Esto garantiza una representación adecuada del comportamiento del sistema físico y evita posibles errores en la modelización.

Por otro lado, la segunda configuración, que no incluye la especificación de KEYOPT para los elementos, podría limitar la precisión y eficacia del modelo en la predicción de comportamientos físicos reales. Además, al no emplear un método específico como el de Lagrange que conserva todos los grados de libertad, podría haber una pérdida de información importante para la precisión del análisis. En contraste, el método de eliminación directa, aunque simplifica el sistema de ecuaciones al eliminar los grados de libertad redundantes, también puede afectar la precisión del modelo al descartar información potencialmente

relevante para el comportamiento físico real del sistema.

En resumen, la inclusión de la especificación de KEYOPT para los elementos BEAM188 y MPC184 proporciona un mayor control, precisión y adaptabilidad en el modelado y análisis en ANSYS, así como un método más efectivo para establecer condiciones de contorno, lo que resulta en modelos más precisos y representativos de los sistemas físicos bajo estudio.

En tercer lugar, volviendo de nuevo al código de MATLAB que utilizamos para ejecutar el proceso de optimización, tras revisar detalladamente la función 'runansys', hemos optado por realizar un cambio específico en los comandos encargados de llamar a ANSYS para su ejecución. Originalmente, estos comandos se configuraban de la siguiente manera:

```
% run ansys
ansys_root='C:\Program Files\ANSYS Inc\ANSYS Student\v232\ansys\bin\winx64\ansys232';
ansys_inp='bridge_e9.ans';
ansys_exe=sprintf('"%s" -b -i %s -o report.out -np 1',ansys_root,ansys_inp);
[status,cmdout] =  system(ansys_exe);
```

Sin embargo, tras una evaluación cuidadosa, hemos tomado la decisión de cambiar a la siguiente configuración:

```
% run ansys
ansys_root='C:\Program Files\ANSYS Inc\ANSYS Student\v232\ansys\bin\winx64\ansys232';
ansys_inp='bridge_e9.ans';
ansys_exe=sprintf('"%s" -b -i %s -o report.out',ansys_root,ansys_inp);
[status,cmdout] =  system(ansys_exe);
```

Este cambio radica en la eliminación del argumento "-np 1", que limitaba la ejecución de ANSYS a un solo procesador. Al quitar esta restricción, permitimos que ANSYS utilice todos los recursos disponibles de manera óptima durante la ejecución.

Cuando ANSYS utiliza múltiples procesadores, se aprovecha de la capacidad de procesamiento paralelo para dividir la carga de trabajo entre varios núcleos de procesador. Es esencial considerar las características del ordenador utilizado para llevar a cabo el Trabajo de Fin de Grado. Estas especificaciones son fundamentales para el análisis que estamos realizando. Esto conlleva las siguientes ventajas, importantes para el cálculo que queremos realizar:

- Mayor velocidad de cálculo: Al distribuir la carga de trabajo entre múltiples procesadores, ANSYS puede completar el análisis más rápidamente en comparación con el uso de un solo procesador. Esto es especialmente útil para modelos grandes y complejos que requieren un tiempo considerable para resolver.
- Mejor rendimiento en análisis intensivos: En análisis que requieren una gran cantidad de cálculos, como análisis estructurales complejos, el uso de múltiples procesadores puede mejorar significativamente el rendimiento y reducir el tiempo de ejecución.
- Escalabilidad: ANSYS está diseñado para escalar de manera eficiente con el número de procesadores disponibles. Esto significa que cuanto más procesadores se utilicen, más lineal será el aumento en el rendimiento. Sin embargo, este aumento puede verse afectado por la naturaleza del problema, la eficiencia del algoritmo y la capacidad del hardware.
- Requisitos de memoria: El uso de múltiples procesadores puede aumentar los requisitos de memoria del sistema, ya que cada procesador puede requerir su propia porción de memoria para almacenar datos temporales y resultados parciales. Es importante asegurarse de que el sistema tenga suficiente memoria disponible para admitir el uso de múltiples procesadores de manera eficiente.

En resumen, la utilización de múltiples procesadores por parte de ANSYS puede ofrecer ventajas significativas en términos de velocidad de cálculo, rendimiento y escalabilidad, especialmente para análisis intensivos y modelos grandes. Sin embargo, es importante tener en cuenta los requisitos de hardware y configurar adecuadamente el software para aprovechar al máximo esta capacidad.

En cuarto lugar, el tercer ajuste realizado tiene implicaciones en esta cuarta modificación. Al efectuar los cálculos con ANSYS implementando el cálculo en paralelo, detectamos un error en el programa. En el último vano, el número 13, la sección "TAPER" previamente definida no funciona bien con secciones variables, que es la forma en que está construido el puente. Por esta razón, decidimos crear una sección media significativa de la variable e50. Debido a esto, ahora hay 16 secciones en lugar de 14.

Esta modificación se realizó para poder continuar los cálculos en paralelo, ya que no es factible realizar cálculos paralelos para algunas secciones y cálculos con un solo procesador para el resto de las secciones.

Por último, el ajuste final que hemos implementado en esta recalibración es la exclusión en el proceso de optimización del parámetro de rigidez longitudinal (muelle en el vano 13), kx. En la aproximación inicial realizada anteriormente, observamos que el valor del parámetro kx presenta variaciones insignificantes y su influencia en el cálculo es mínima.

Argumentamos a continuación porqué el parámetro kx no tiene influencia destacable en el cálculo.

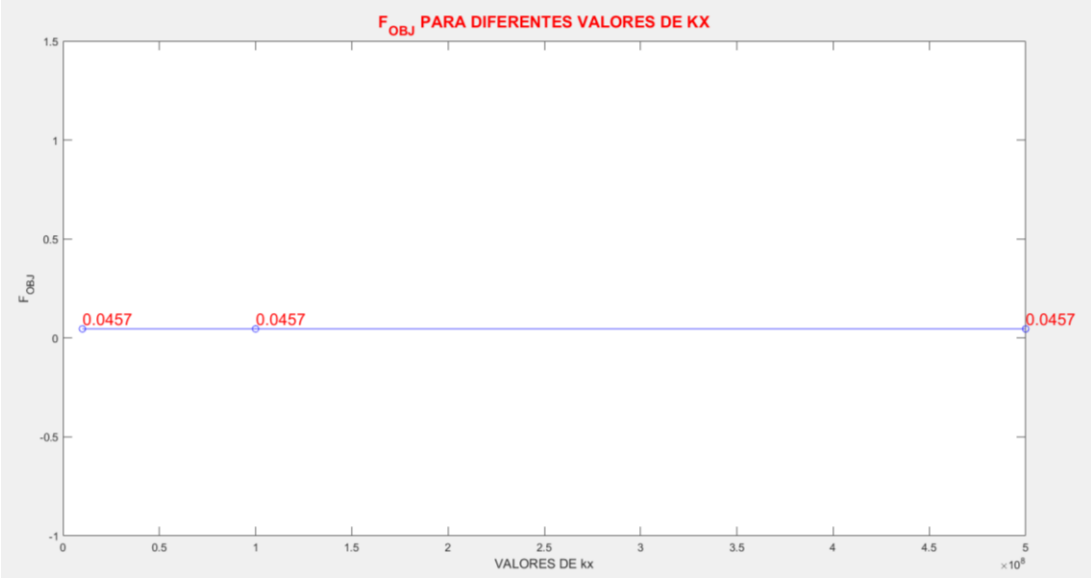

Figura 4.23 Valor de la función objetivo para diferentes valores de ‘kx’ (rigidez longitudinal).

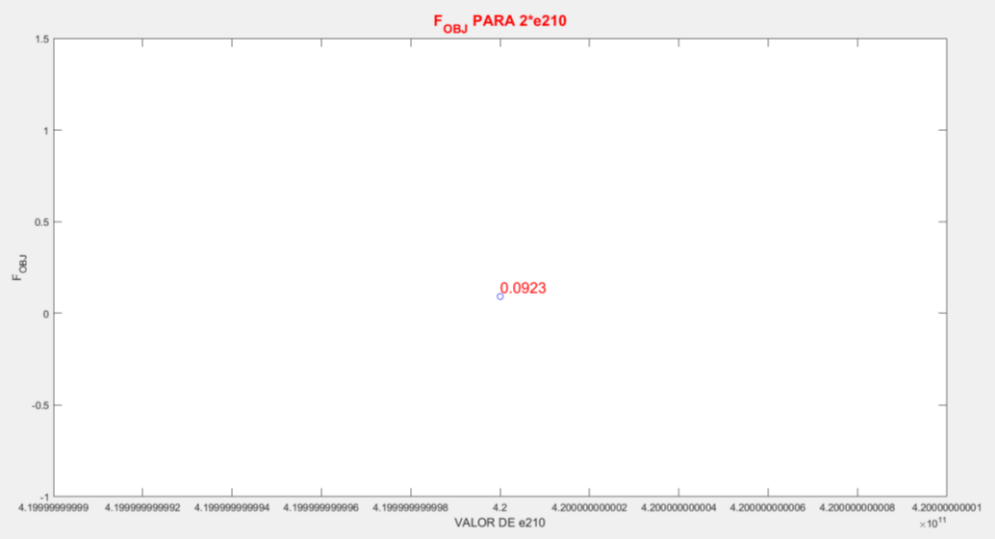

Figura 4.24 Valor de la función objetivo para una variación de 'e210' (módulo elástico del acero).

Podemos observar en la Figura 4.23 los tres valores de 'Fobj' son iguales para los diferentes valores de kx evaluados:

- 1er punto: kx = 5e8 (5 veces el valor nominal)
- 2º punto: kx = 1e8 (valor nominal)
- 3er punto kx = 1e7 (0.1 veces el valor nominal)

En los tres casos, el valor de la función objetivo es 0.0457.

Sin embargo, en la Figura 4.24 podemos ver que al variar el valor de la variable e210, el valor de la función objetivo cambia significativamente, obtenemos un valor de función objetivo cercano a 0.1. Por lo tanto, podemos concluir que el parámetro kx tiene un impacto poco significativo en este análisis. Esto tiene sentido, ya que las cargas aplicadas son verticales y kx, definido como rigidez longitudinal, no tiene un efecto importante en un análisis con cargas verticales.

Debido a que este parámetro no tiene un impacto considerable en los resultados, hemos decidido no incluirlo en la nueva aproximación. Esta decisión no solo simplifica el modelo, sino que también reduce el coste computacional y aumenta la velocidad de los cálculos. Hemos decidido fijar dicho valor en 1e8 N/m, valor muy cercano al obtenido en el cálculo inicial.

Hemos modificado además, con el fin de ahorrar coste computacional y lograr una mayor precisión, los siguientes parámetros:

- Se ha fijado el valor de 'ObjectiveLimit' en 4e-4. Valor correspondiente a la máxima precisión que obtendremos de nuestra función objetivo.
- Se ha fijado también el valor de 'MaxFunctionEvaluations' en 200, valor por defecto, para no sobrecargar computacionalmente el análisis, es un valor razonable y aceptable.
- Los límites de cálculo, límite superior 'ub' e inferior 'lb' se han fijado en los siguientes valores (con el objetivo de ser más precisos en el cálculo y sabiendo que el rango de valores que buscamos se encuentra en los definido a continuación):

  ub = [1.05 1.05 1.05 1.05]

  lb = [0.8 0.8 0.8 0.7]

Por lo tanto, a modo de conclusión, en el apéndice A.5 se muestra el código de MATLAB en el que llevamos a

cabo el proceso de optimización con todas las modificaciones de la recalibración implementadas.

4.3.5 Resultados Prueba de Carga 1 (aproximación final)

Al igual que en la aproximación inicial, comenzaremos con un análisis específico, evaluando casos individuales, antes de avanzar hacia una solución más global, como las evaluaciones del 1 al 13 o la resolución conjunta de las tres pruebas de carga. En esta nueva aproximación, y para simplificar el proceso, hemos decidido normalizar las variables a calcular ([e210, e30, e40, e50]) en términos de 1. De esta manera, podremos medir la desviación entre el valor calculado y el valor nominal de cada variable de manera más eficiente, por lo tanto, veremos lo próximo o lejano que se encuentra el valor calculado del valor nominal de la variable en cuestión.

Por tanto, se muestran a continuación los resultados de las pruebas individuales de la primera prueba de carga:

Prueba 1_2				
Fobj_mín	**X_optím**			
0.00036514837167011	0.9328125	1.0265625	0.9015625	0.8421875

Tabla 4.5 Resultados del segundo caso de carga correspondientes a la primera prueba de carga.

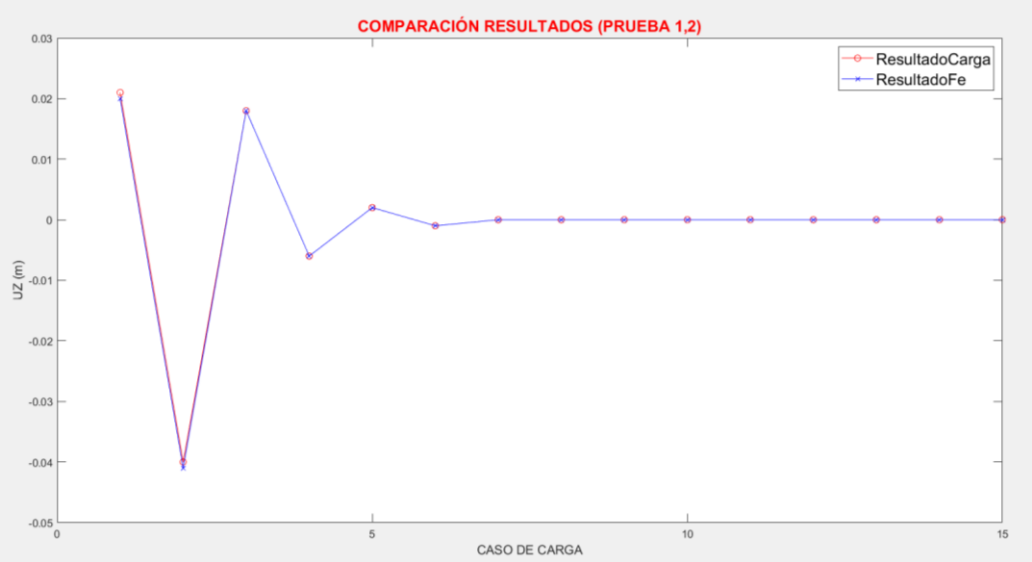

Figura 4.25 Resultados gráficos del segundo caso de carga. Desplazamientos aproximados vs referencia.

Prueba 1_3				
Fobj_mín	**X_optím**			
0.00036514837167011	0.92890625	0.91328125	1.02265625	1.00078125

Tabla 4.6 Resultados del tercer caso de carga correspondientes a la primera prueba de carga.

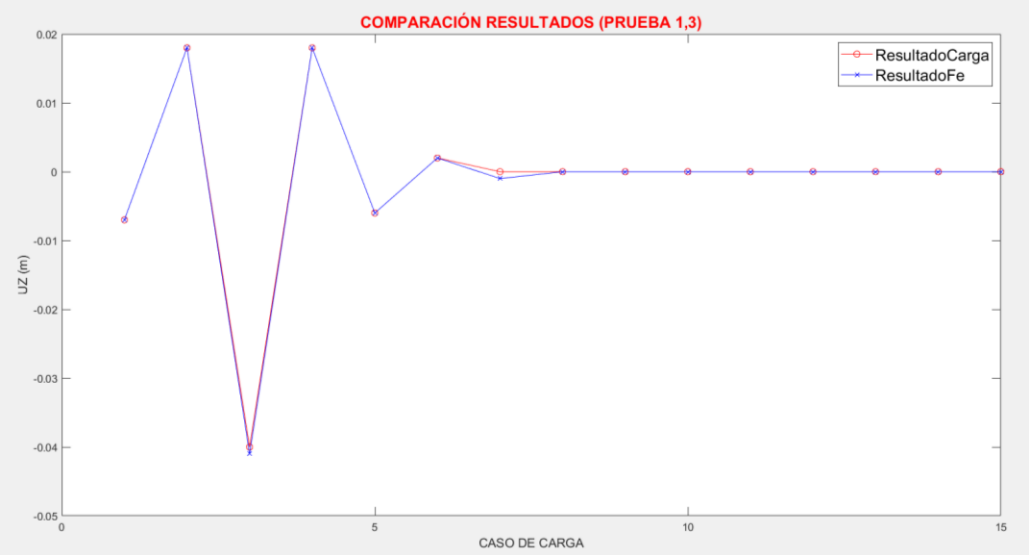

Figura 4.26 Resultados gráficos del tercer caso de carga. Desplazamientos aproximados vs referencia.

Prueba 1_12				
Fobj_mín	**X_optím**			
0.000605530070819498	0.8400390625	0.9943359375	1.0119140625	0.9939453125

Tabla 4.7 Resultados del decimosegundo caso de carga correspondientes a la primera prueba de carga.

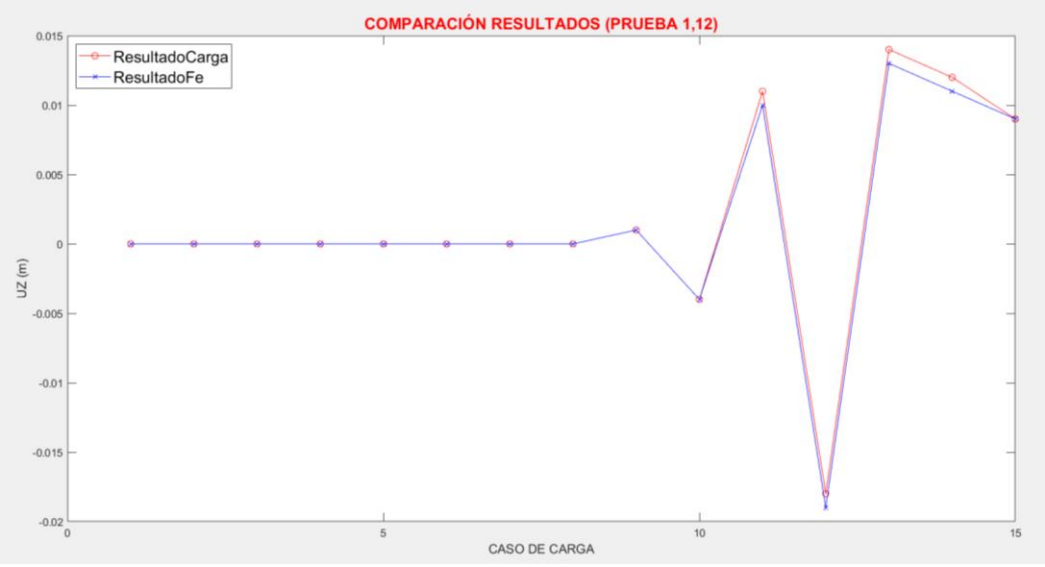

Figura 4.27 Resultados gráficos del decimosegundo caso de carga. Desplazamientos aproximados vs referencia.

Prueba 1_13				
Fobj_mín	**X_optím**			
0.00230217288664427	1.03046875	0.98359375	1.03046875	0.81484375

Tabla 4.8 Resultados del decimotercero caso de carga correspondientes a la primera prueba de carga.

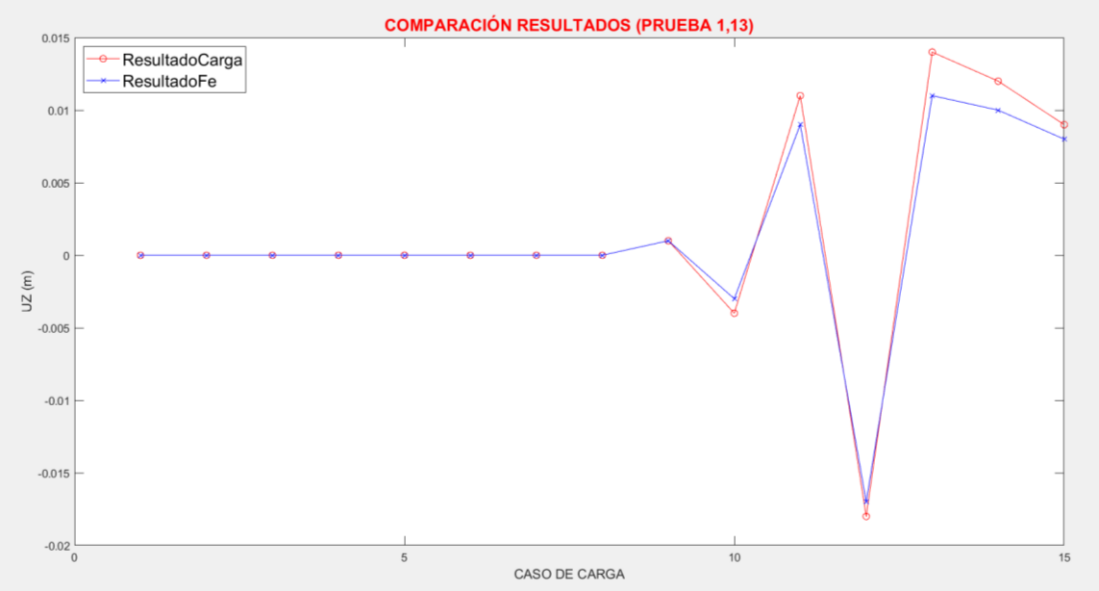

Figura 4.28 Resultados gráficos del decimotercero caso de carga. Desplazamientos aproximados vs referencia.

Se observan en la Tabla 4.5, 4.6, 4.7 y 4.8 los resultados de las pruebas evaluadas, así como en la Figura 4.25, 4.26, 4.27 y 4.28 los resultados gráficos correspondientes a dichas pruebas. Esta vez, a diferencia de la aproximación inicial anteriormente realizada, obtenemos una precisión mucho mayor, aproximadamente de una milésima. Este incremento en precisión se debe, entre otras cosas, a la implementación de una nueva función objetivo. Esta nueva función objetivo ha sido mejorada mediante la ponderación de las pruebas, lo cual hemos comentado anteriormente. En este contexto, la ponderación de pruebas significa que hemos asignado diferentes niveles de importancia a las diferentes pruebas, de manera que la tercera prueba, que corresponde a un vano sometido a torsión (vano 13), no afecta de manera significativa al análisis global.

En cuanto a las cuatro variables que hemos considerado, podemos observar que, en general, están muy cercanas a los valores nominales esperados. Vamos a desglosar cada uno de estos valores óptimos y su significado:

En primer lugar, las soluciones de los casos de carga 2 y 3 muestran que las variables de diseño están muy cercanas a los valores nominales, indicando una solución optimizada y coherente con los resultados reales del modelo.

En cuanto a los casos de carga 12 y 13, observamos un aumento significativo en el valor de la función objetivo, lo que indica menor precisión en la optimización para este vano. Esto es esperado debido a la naturaleza particular del vano 13 que se somete a torsión.

Prestando atención a las variables de acero y hormigón vemos una alta resistencia por ambas partes. La combinación de estos materiales es ideal porque el acero, con su alta resistencia a tracción, y el hormigón, con su

alta resistencia a compresión, se complementan perfectamente. En una estructura compuesta, el acero será el principal encargado de soportar los esfuerzos de tracción, mientras que el hormigón se encargará de los esfuerzos de compresión. Esto permite una distribución eficiente de las cargas, mejorando la resistencia y durabilidad de la estructura.

Además, como se mencionó anteriormente, la rigidez longitudinal (kx) se ha fijado en 1e8 N/m. Esta alta rigidez asegura que la deformación longitudinal de los vanos bajo carga sea mínima, contribuyendo a la estabilidad global de la estructura.

Finalmente, se puede observar claramente que los resultados en los vanos principales (cuyos resultados son extrapolables a los vanos centrales) son más precisos. Esto se debe a que estos vanos están sujetos a condiciones de carga más uniformes y previsibles. En contraste, los vanos finales presentan funciones objetivo mayores y resultados menos precisos, debido a la complejidad adicional de las condiciones de carga y soporte en esos extremos de la estructura.

En resumen, la optimización ha logrado resultados muy precisos para la mayoría de los vanos, con algunas variaciones esperadas en los vanos sometidos a condiciones más complejas, gracias a la mejora en la función objetivo y la adecuada ponderación de las pruebas. Sin embargo, esto no deja de ser un análisis particular en el que no podemos determinar con exactitud una solución global. En estas últimas figuras (Figura 4.29 y 4.30) se muestran los resultados de todos los casos de carga individualmente, en las que podemos apreciar la similitud de resultados tanto en las funciones objetivo como en el valor de las 4 variables de estudio, especialmente en los vanos más centralizados.

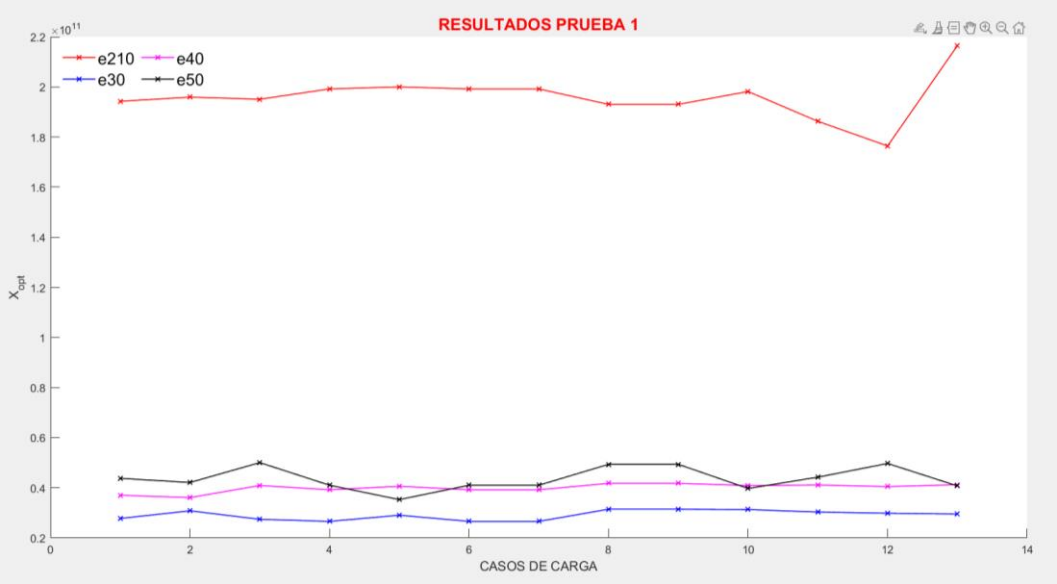

Figura 4.29 Variables de estudio para cada uno de los casos de carga.

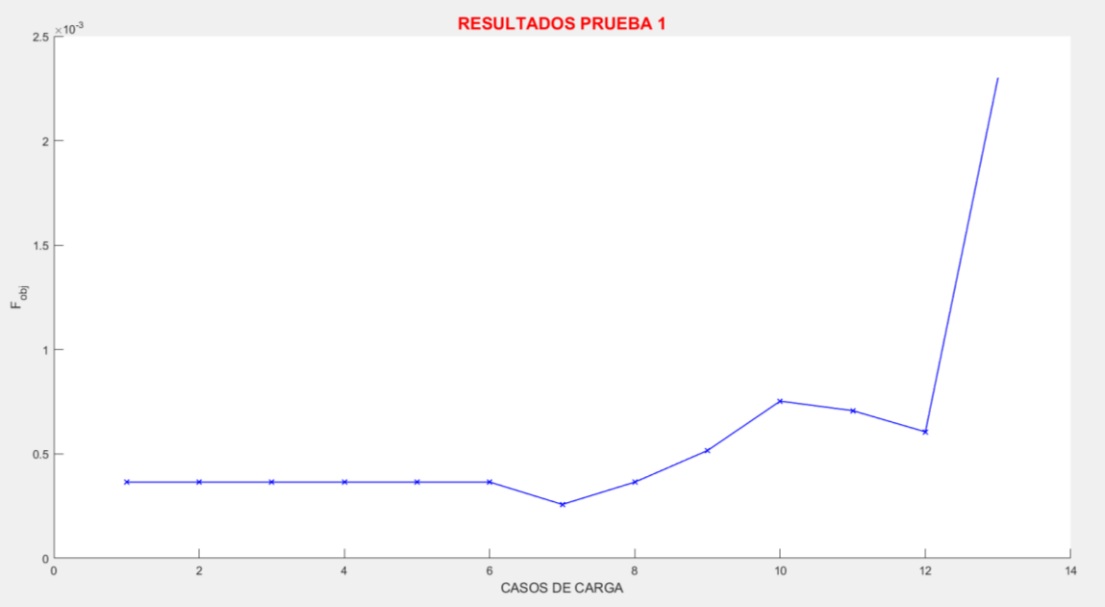

Figura 4.30 Funciones objetivo en función de cada caso de carga.

Pasamos a estudiar la solución en la que se incluyen todos los casos de carga de la primera prueba para ver qué obtenemos.

A continuación, vemos los resultados obtenidos del estudio de la primera prueba de carga al completo, los 13 casos de carga a la vez. Es importante tener en cuenta que, la 'Fobj' ahora pasa a ser un sumatorio de varios términos en vez de un solo término:

$$Fobj = \sum_{lSpan=1}^{nSpan} \sqrt{\phi^2} * (iPrueba)^{-1}$$

Y una vez terminado el sumatorio, se normaliza la función objetivo:

$$Fobj = \frac{Fobj}{lCase * Precision * \sqrt{2 * nPost}}$$

Este cálculo, explicado también anteriormente, se llevará a cabo en cada iteración.

Los resultados obtenidos en este análisis son los siguientes (Tabla 4.9):

Prueba 1 del 1 al 13				
Fobj_mín	**X_optím**			
0.00061193	0.914428461948563	1.05	1.05	1.04461024939155

Tabla 4.9 Resultados totales correspondientes a la primera prueba de carga en su conjunto.

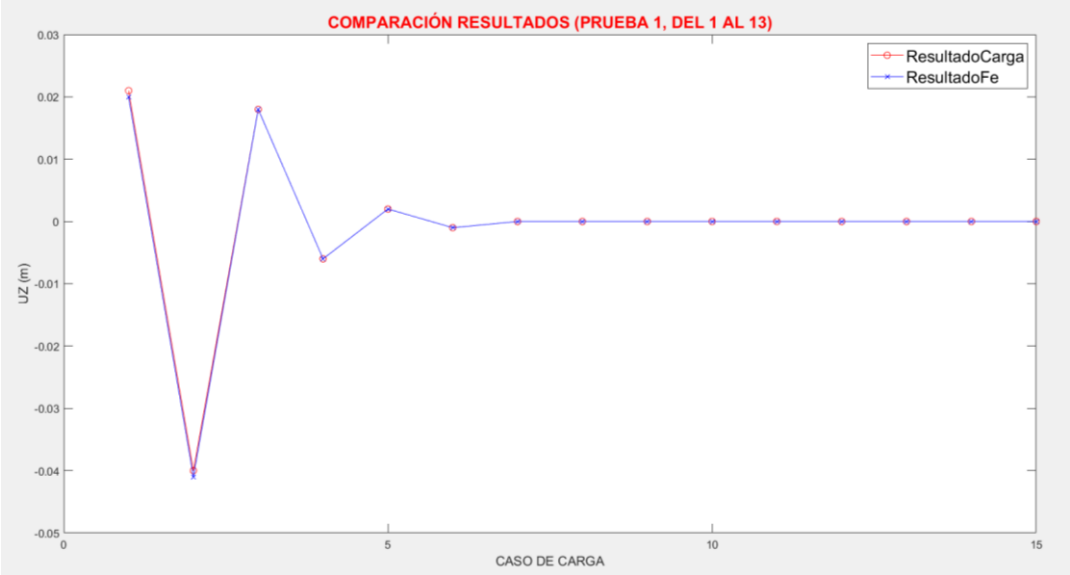

Figura 4.31 Segundo caso de carga correspondiente al estudio total de la primera prueba de carga.

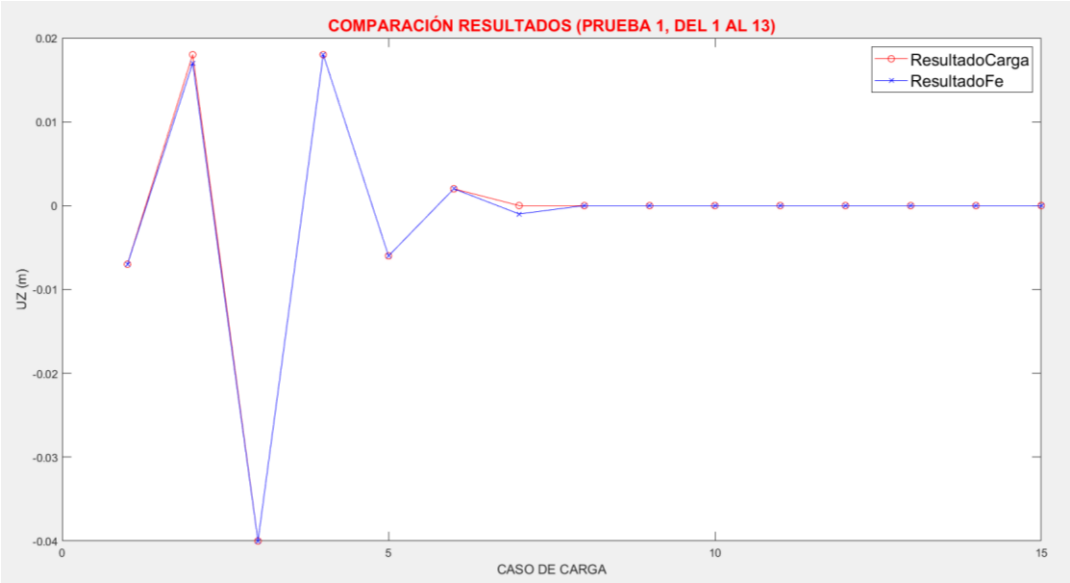

Figura 4.32 Tercer caso de carga correspondiente al estudio total de la primera prueba de carga.

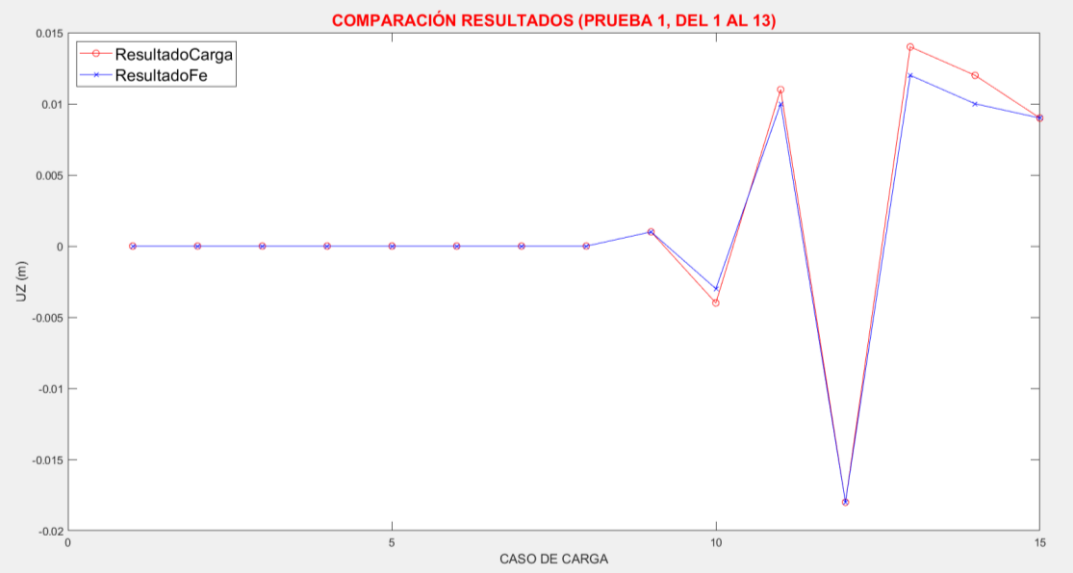

Figura 4.33 Decimosegundo caso de carga correspondiente al estudio total de la primera prueba de carga.

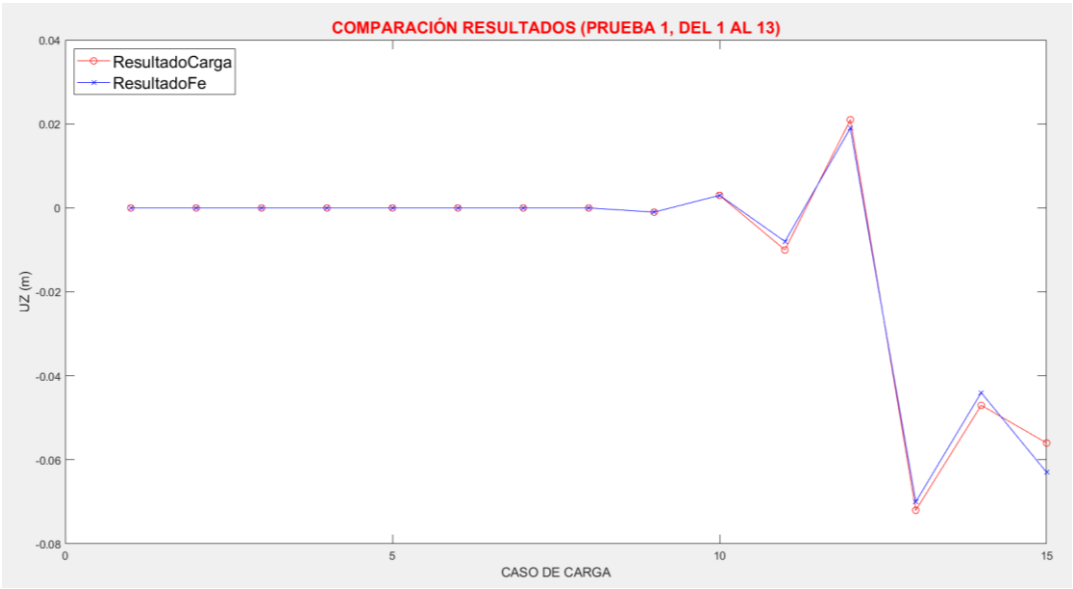

Figura 4.34 Decimotercero caso de carga correspondiente al estudio total de la primera prueba de carga.

A partir de los resultados mostrados encima de las figuras (Figura 4.31, 4.32, 4.33 y 4.34), llegamos a las siguientes conclusiones:

- Las funciones objetivo obtenidas son muy precisas y cercanas al 'ObjectiveLimit'. El valor mínimo establecido es de 4e-4, lo que indica que hemos obtenido unos resultados válidos y aceptables. Además, se puede ver que los resultados son extrapolables en el vano central y que obtenemos unos resultados uniformes.
- En cuanto a los parámetros de la solución, analizamos primeramente el valor del módulo de elasticidad del acero. Tal y como esperábamos, se obtiene un valor muy bueno y preciso de módulo de elasticidad (aproximadamente 200 GPa). Esta es una aproximación muy buena y consistente con los valores obtenidos en los casos individuales.

- En cuanto a los módulos elásticos a compresión de los hormigones e30, e40 y e50, los valores obtenidos son muy cercanos a sus valores nominales (30 GPa, 40 GPa y 50 GPa respectivamente), mostrando una buena resistencia a compresión y permitiendo una buena compenetración con el acero. Parece que los resultados empiezan a converger, lo cual indica que los resultados tienen muy buena pinta.

A continuación, se muestra una figura (Figura 4.35) en la que se puede observar el proceso de optimización realizado a través de la función 'surrogateopt'. En ella podemos ver cómo se evalúan unos puntos iniciales (proporcionados por el algoritmo de subrogación). Una vez evaluados los primeros 20 puntos, podemos ver un amplio rango de puntos que son puntos adaptados, tal y como se ha explicado anteriormente en el apartado 2.2. Posteriormente, alrededor del punto 80, podemos apreciar un reseteo de puntos, que nos aporta nuevos puntos aleatorios de muestreo, etc. De nuevo, más puntos adaptados, y finalmente, al no superar el umbral de 'ObjectiveLimit' establecido, llegamos al máximo de iteraciones establecido por defecto, 200 iteraciones.

Este ejemplo de proceso de optimización es extrapolable a todos los procesos de optimización que se llevan a cabo en este trabajo.

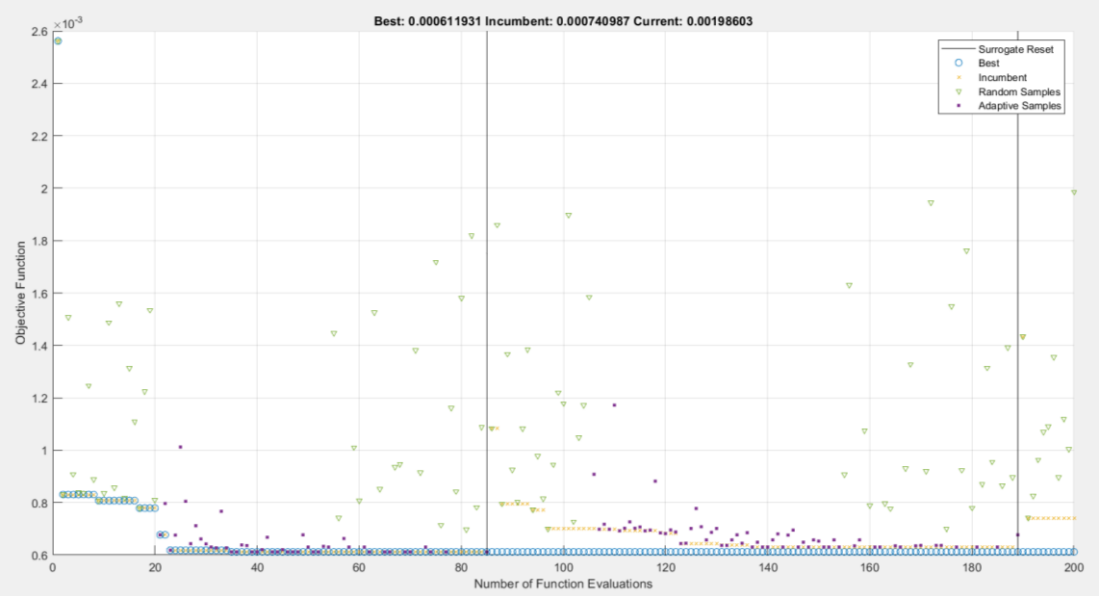

Figura 4.35 Proceso de optimización llevado a cabo a través de MATLAB.

4.3.6 Resultados Prueba de Carga 2 (aproximación final)

Pasamos ahora a estudiar los resultados de la segunda prueba de carga individualmente en esta aproximación final que estamos realizando. Este segundo estudio nos guiará en gran parte hacia los resultados finales que llegaremos a obtener, después de haber evaluado dos pruebas de carga, debemos estar cerca de la convergencia de los resultados.

Los resultados obtenidos son los siguientes:

Prueba 2_2				
Fobj_mín	**X_optím**			
0.00036514837167011	0.9875	0.9875	0.9875	0.7875

Tabla 4.10 Resultados del segundo caso de carga correspondientes a la segunda prueba de carga.

Figura 4.36 Resultados gráficos del segundo caso de carga. Desplazamientos aproximados vs referencia.

Prueba 2_3				
Fobj_mín	**X_optím**			
0.00036514837167011	0.9875	0.9875	0.9875	0.7875

Tabla 4.11 Resultados del tercer caso de carga correspondientes a la segunda prueba de carga.

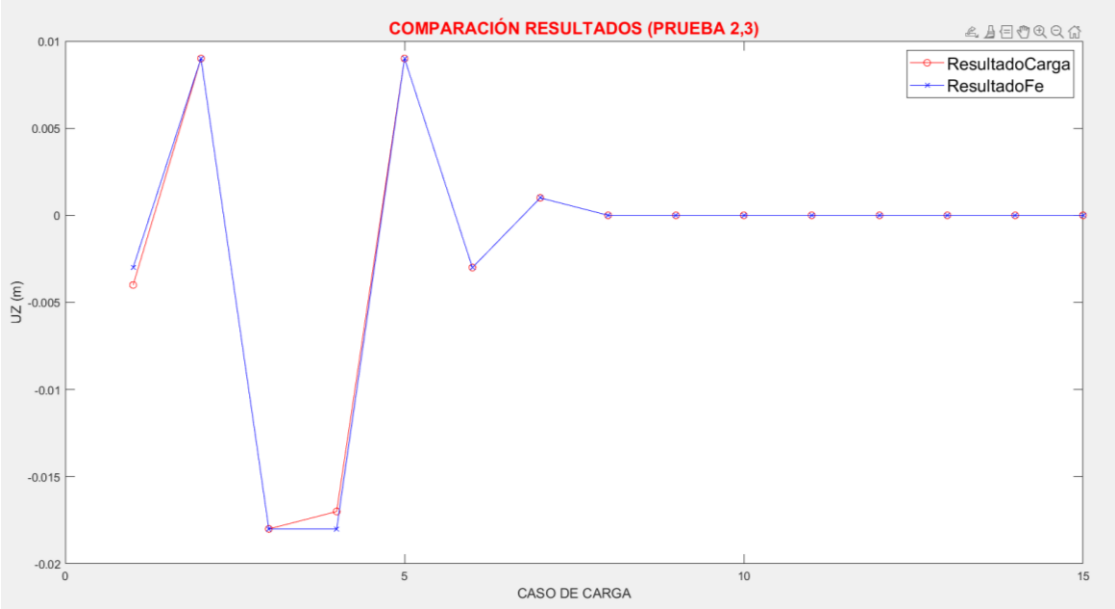

Figura 4.37 Resultados gráficos del tercer caso de carga. Desplazamientos aproximados vs referencia.

Prueba 2_11				
Fobj_mín	**X_optím**			
0.000516397779494322	1.0312186445246	1.0425136334279	0.96428274271958	0.75734590864264

Tabla 4.12 Resultados del decimoprimero caso de carga correspondientes a la segunda prueba de carga.

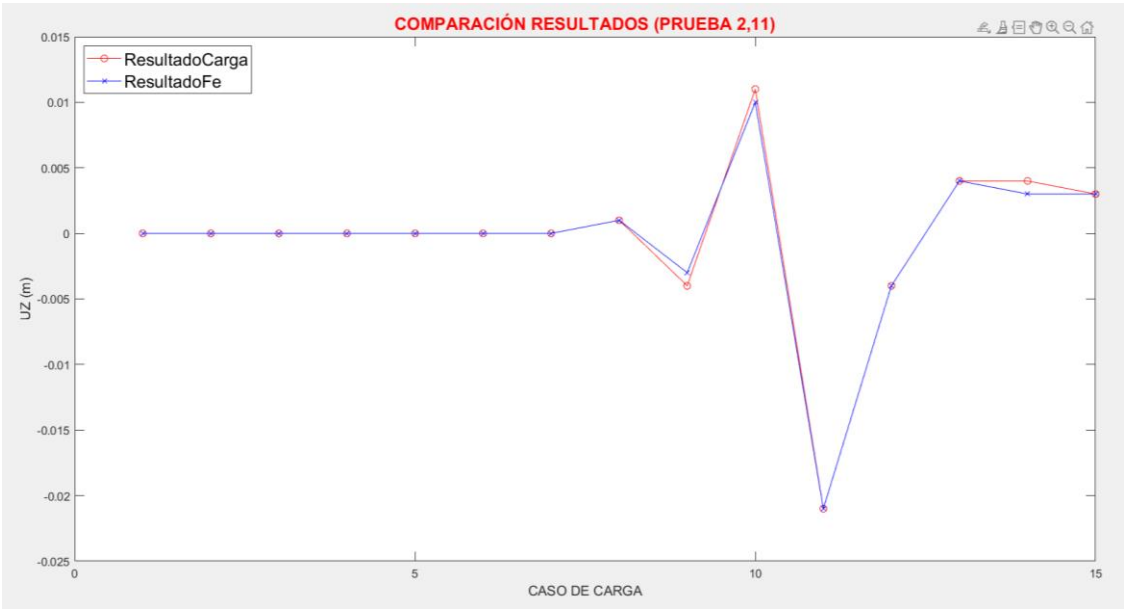

Figura 4.38 Resultados gráficos del decimoprimero caso de carga. Desplazamientos aproximados vs referencia.

Prueba 2_12				
Fobj_mín	**X_optím**			
0.00135400640077266	1.05	1.05	0.896687293364056	1.05

Tabla 4.13 Resultados del decimosegundo caso de carga correspondientes a la segunda prueba de carga.

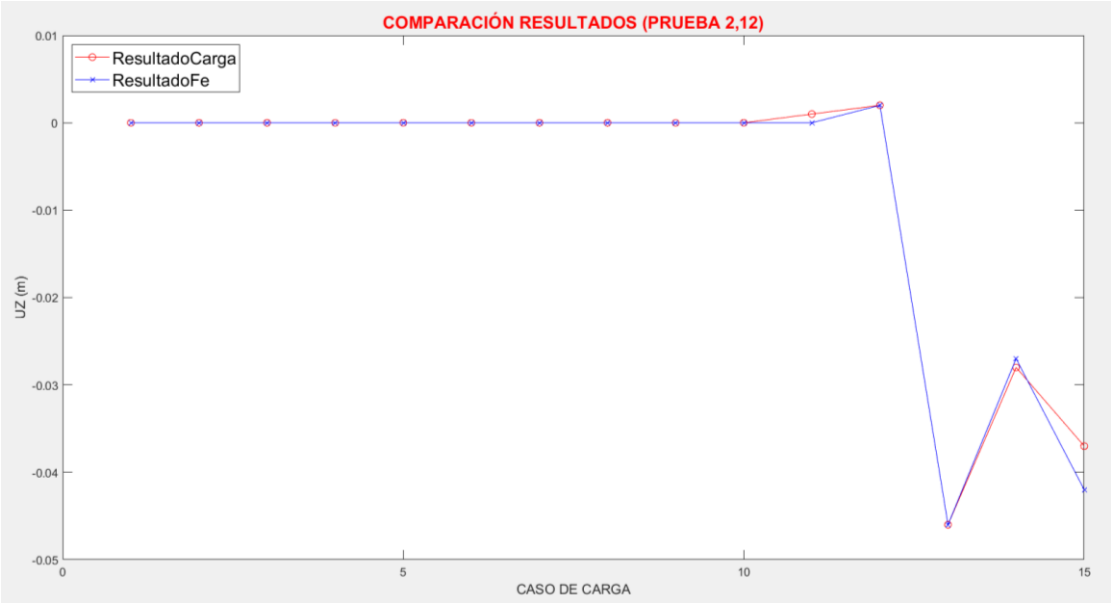

Figura 4.39 Resultados gráficos del decimosegundo caso de carga. Desplazamientos aproximados vs referencia.

En último lugar, se presentan a continuación los resultados que comparan los 12 casos de carga que contiene la segunda prueba de carga (Figura 4.40 y 4.41):

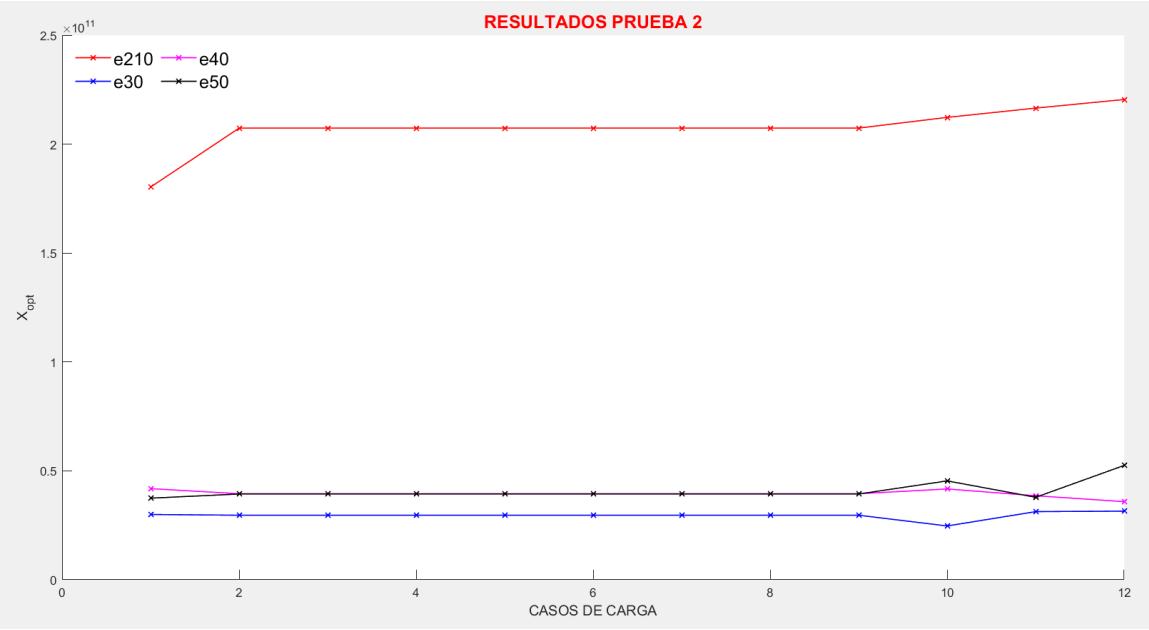

Figura 4.40 Variables de estudio en función del caso de carga estudiado.

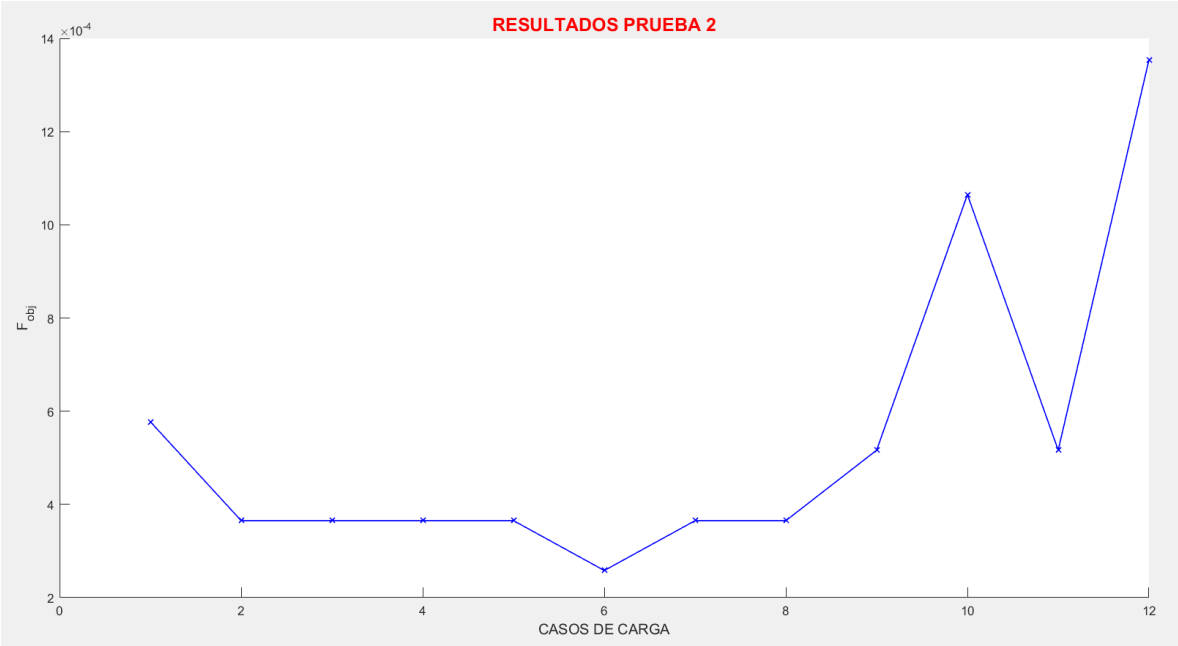

Figura 4.41 Función objetivo en función del caso de carga estudiado.

Los resultados obtenidos en esta segunda prueba de carga muestran una gran similitud con los de la primera prueba. Los valores de la función objetivo (Fobj) son muy precisos y, en muchos casos, especialmente en los vanos centrales, están por debajo del valor de 4e-4 del 'ObjectiveLimit'.

En cuanto a las cuatro variables de estudio [e210, e30, e40, e50], los valores que presentan en los diferentes casos de carga son prácticamente idénticos en todo el puente y se aproximan mucho a sus valores nominales. Como esperábamos, los resultados en los vanos finales (casos de carga 11 y 12) son menos precisos y los valores de las cuatro variables de estudio son un poco más dispersos. Esto se debe a la curvatura del último vano y a su correspondiente solicitación a torsión, como se ha mencionado en varias ocasiones. Es crucial destacar la función objetivo empleada y su ponderación en estas pruebas para asegurarse de que este vano no tenga una influencia excesivamente negativa.

La convergencia observada es notable, indicando que estamos cerca de alcanzar la solución final. No obstante, es fundamental realizar un análisis global posterior para confirmar y validar estos resultados de manera definitiva.

Evaluamos a continuación la prueba de carga número 2 en su conjunto. Los 12 casos de carga a la vez.

Prueba 2 del 1 al 12				
Fobj_mín	**X_optím**			
0.00056713	0.9710	1.0421	1.0220	1.0444

Tabla 4.14 Resultados totales correspondientes a la segunda prueba de carga en su conjunto.

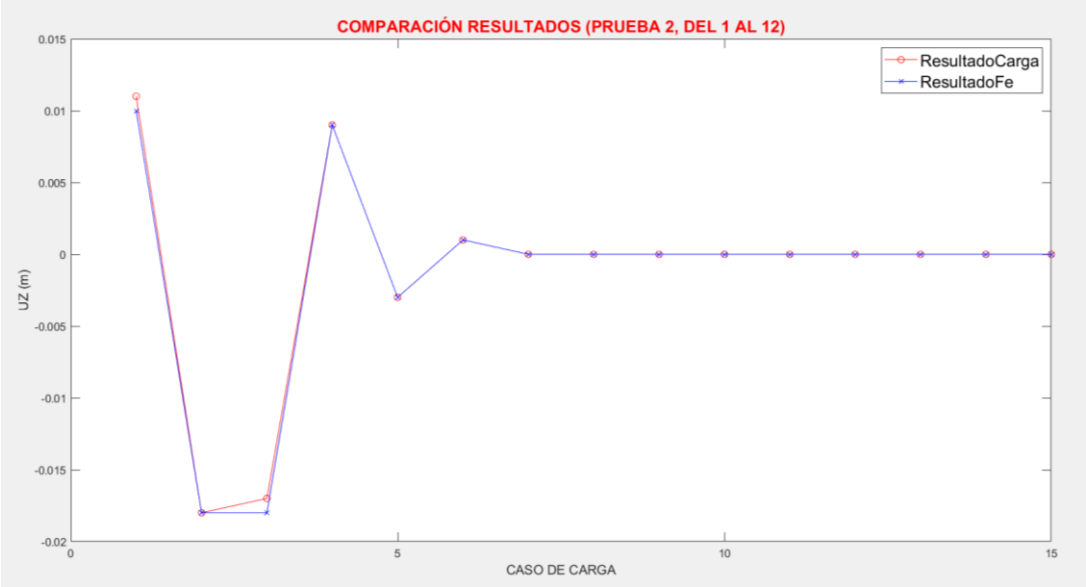

Figura 4.42 Segundo caso de carga correspondiente al estudio total de la segunda prueba de carga.

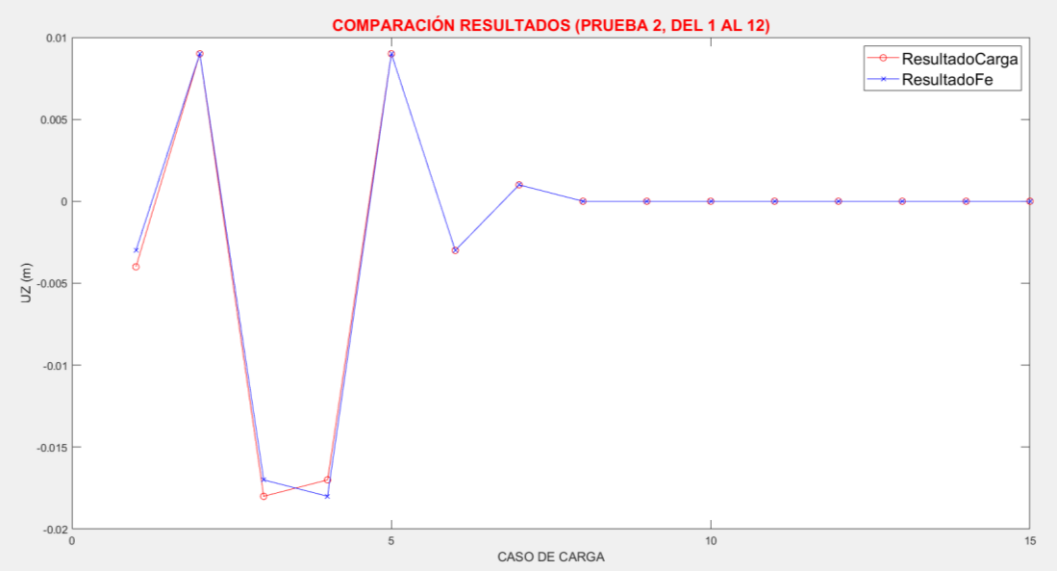

Figura 4.43 Tercer caso de carga correspondiente al estudio total de la segunda prueba de carga.

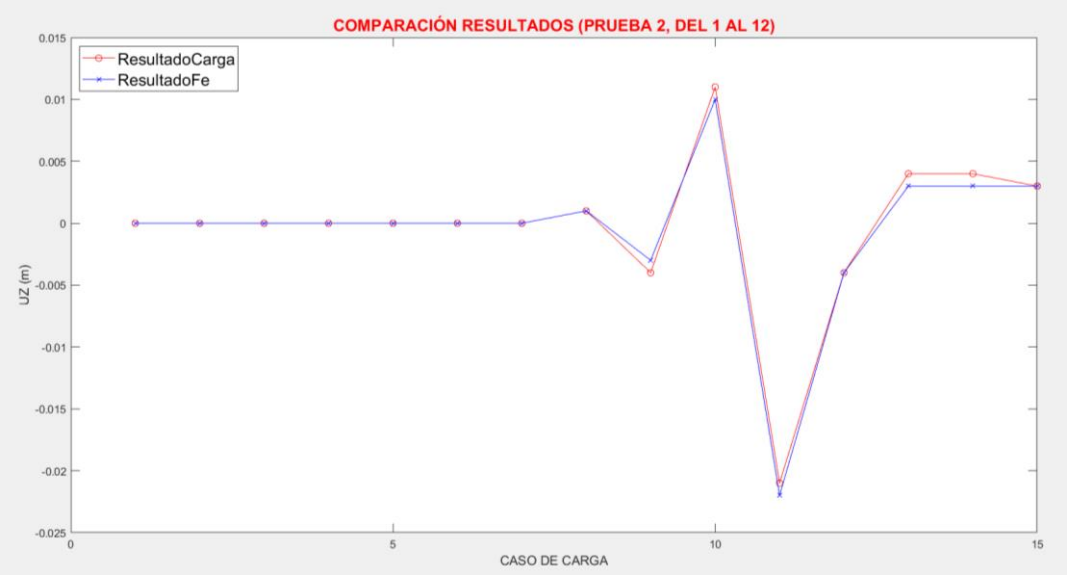

Figura 4.44 Decimoprimero caso de carga correspondiente al estudio total de la segunda prueba de carga.

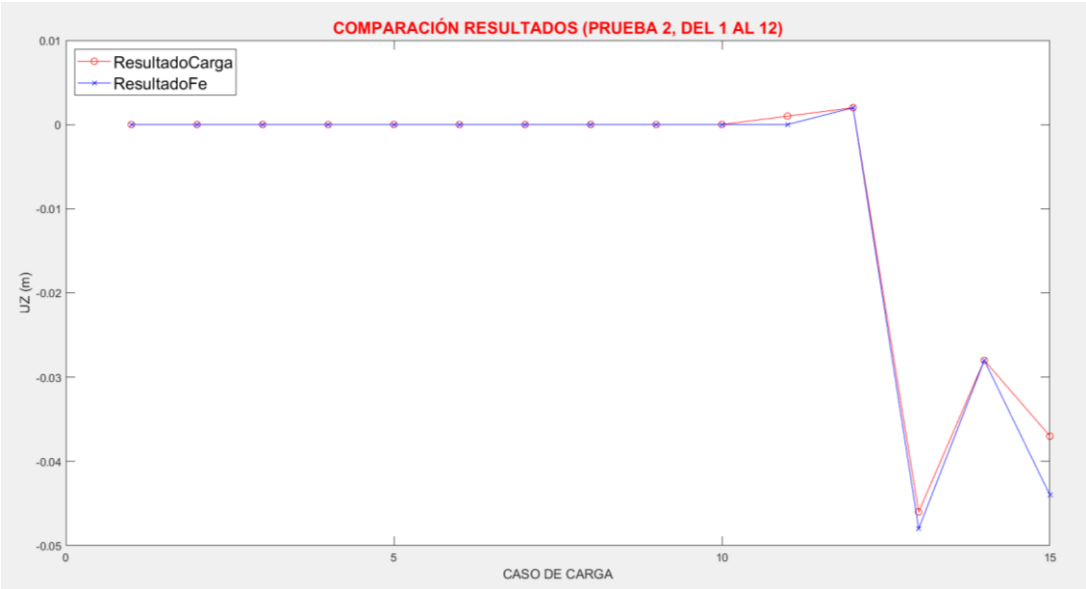

Figura 4.45 Decimosegundo caso de carga correspondiente al estudio total de la segunda prueba de carga.

Las soluciones obtenidas (Tabla 4.14, Figura 4.42, 4.43, 4.44 y 4.45) son coherentes con los anteriores análisis realizados. Procederemos posteriormente a realizar el análisis global en el que veremos los valores finales de las 4 variables de estudio. Seguramente, los valores que vamos a obtener sean muy similares a los que estamos obteniendo en estas evaluaciones particulares que estamos realizando. Pasamos a calcular las soluciones de la tercera prueba de carga.

4.3.7 Resultados Prueba de Carga 3 (aproximación final)

Por último, analizamos la tercera prueba de carga que está compuesta únicamente de un solo caso de carga en el vano número tres, tal y como se ha descrito en el apartado 4.1, para medir la rigidez a torsión del puente. Obtenemos los siguientes resultados de la última prueba de carga:

Prueba 3_3				
Fobj_mín	**X_optím**			
0.000730296743340222	0.8	0.810677165128203	1.00133389091694	0.7

Tabla 4.15 Resultados del tercer caso de carga correspondientes a la tercera prueba de carga.

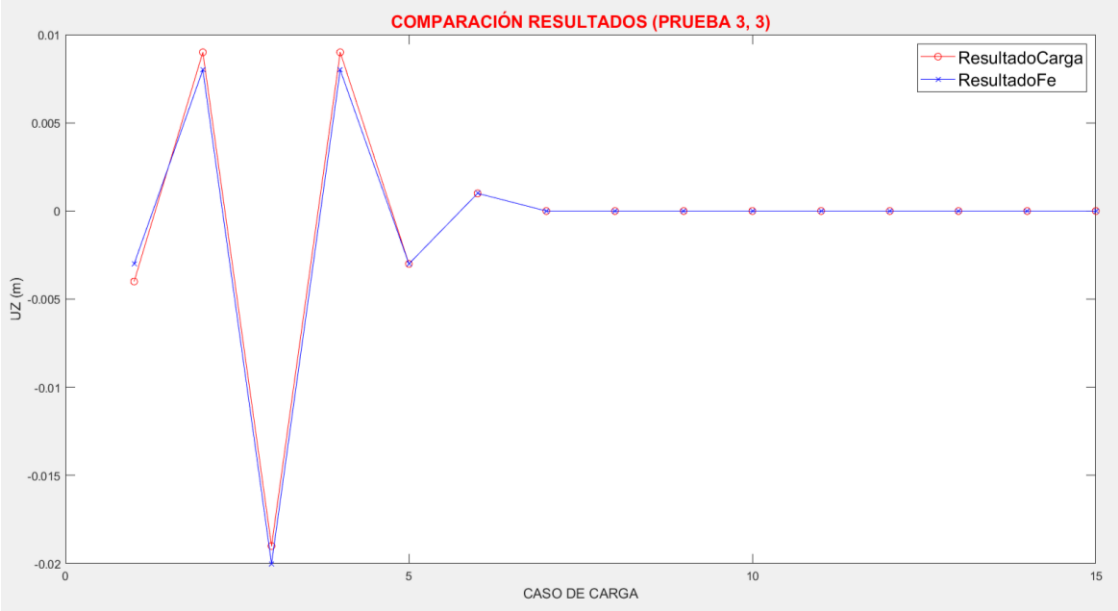

Figura 4.46 Resultados del tercer y único caso de carga correspondientes a la tercera prueba de carga.

Resultados coherentes con los resultados obtenidos en las anteriores evaluaciones realizadas.

4.3.8 Resultados globales de las 3 Pruebas de Carga conjuntas (aproximación final)

Vamos a examinar en último lugar los resultados obtenidos en la solución total a través del algoritmo de subrogación.

Al igual que en la aproximación inicial, procedemos al cálculo global de las tres pruebas de carga conjuntas. Destacamos de nuevo que en este proceso de optimización, se tienen en cuenta 26 casos de carga conjuntamente en cada una de las iteraciones.

Los resultados obtenidos son los siguientes (Tabla 4.16):

Prueba TOTAL SUBROGACIÓN					
Fobj_mín	**X_optím**				**elapsed_time**
0.000646316558586918	0.915858670606212	1.043750	1.043750	1.041250	6715.737011 s

Tabla 4.16 Resultados totales correspondiente a las tres pruebas de carga en su conjunto (aproximación final).

PRUEBA 1

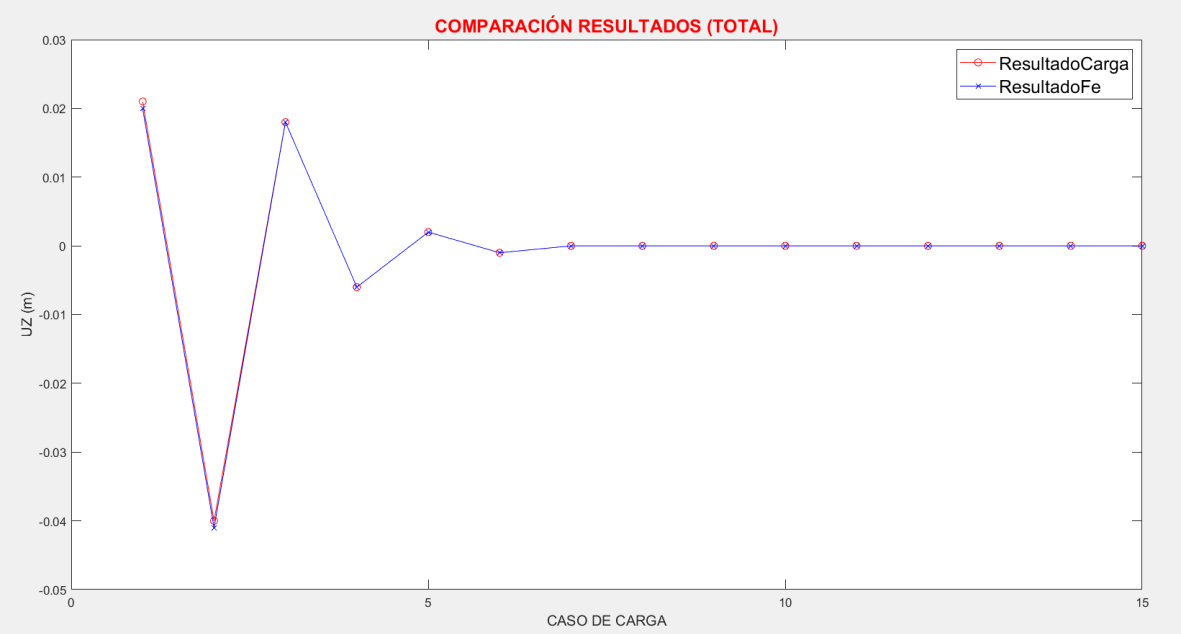

Figura 4.47 Segundo caso de carga correspondiente al estudio total de las tres pruebas de carga.

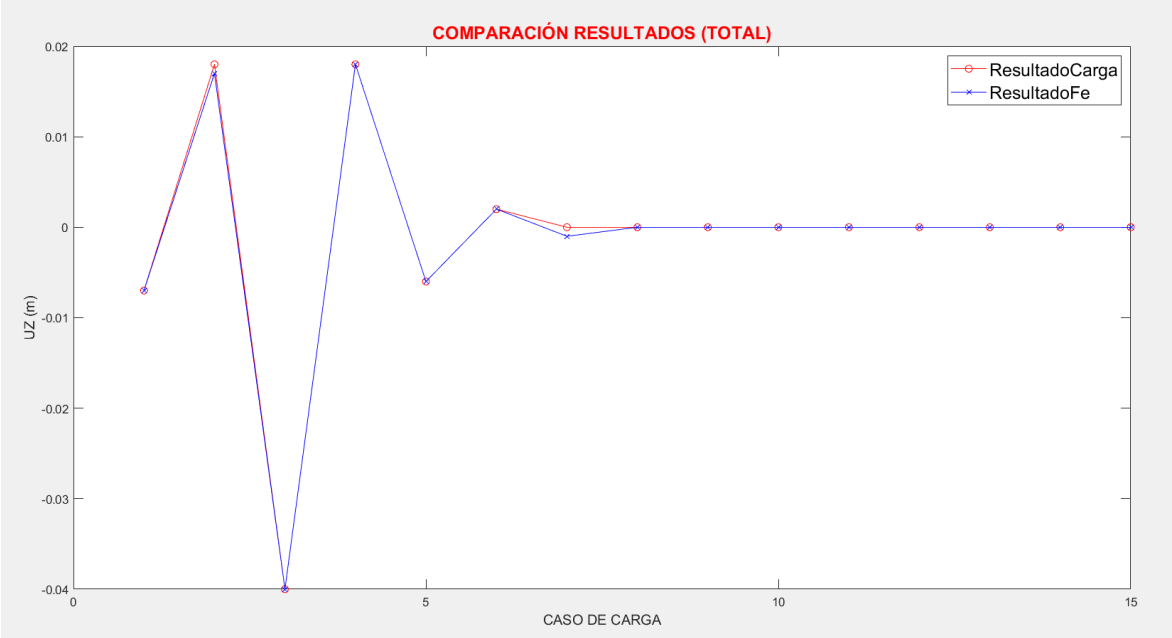

Figura 4.48 Tercer caso de carga correspondiente al estudio total de las tres pruebas de carga.

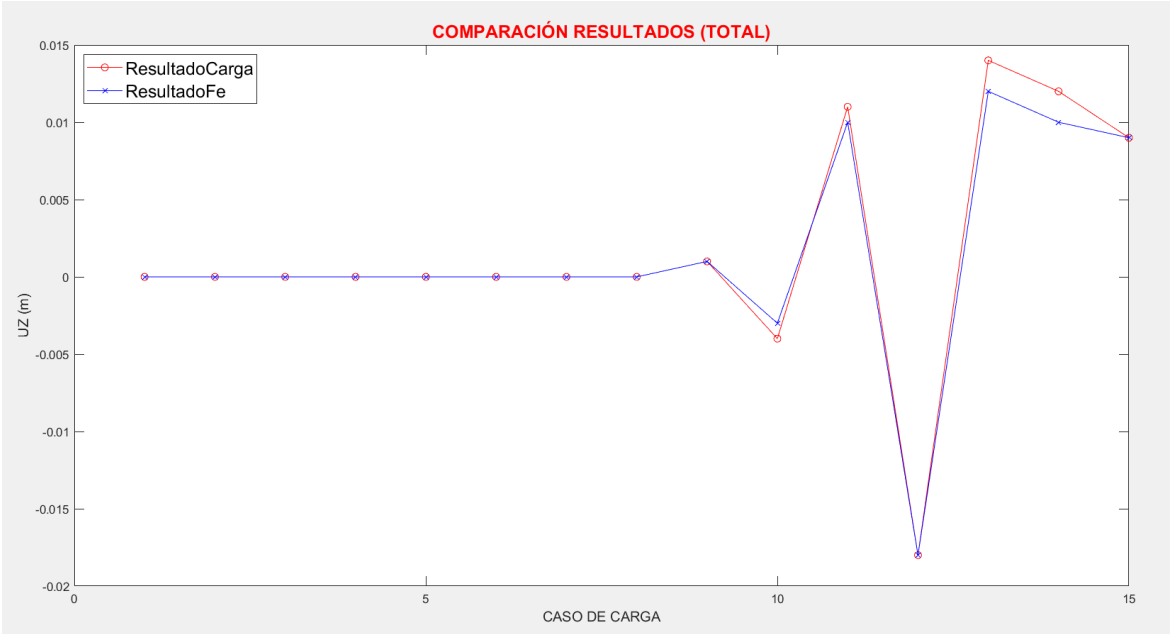

Figura 4.49 Decimosegundo caso de carga correspondiente al estudio total de las tres pruebas de carga.

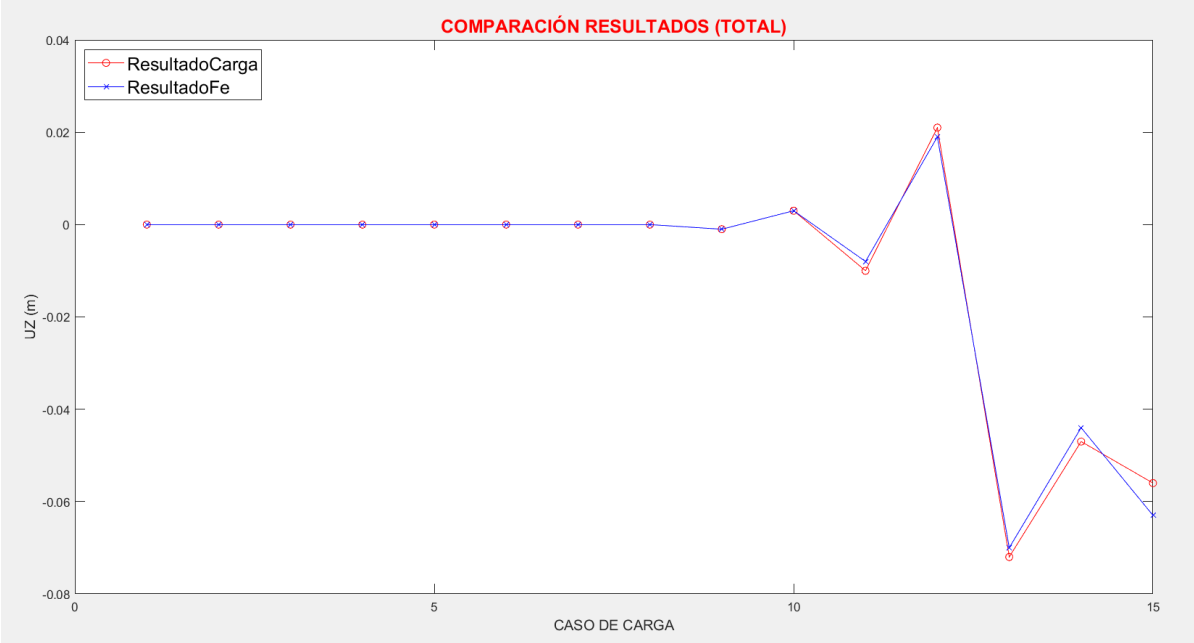

Figura 4.50 Decimotercero caso de carga correspondiente al estudio total de las tres pruebas de carga.

PRUEBA 2

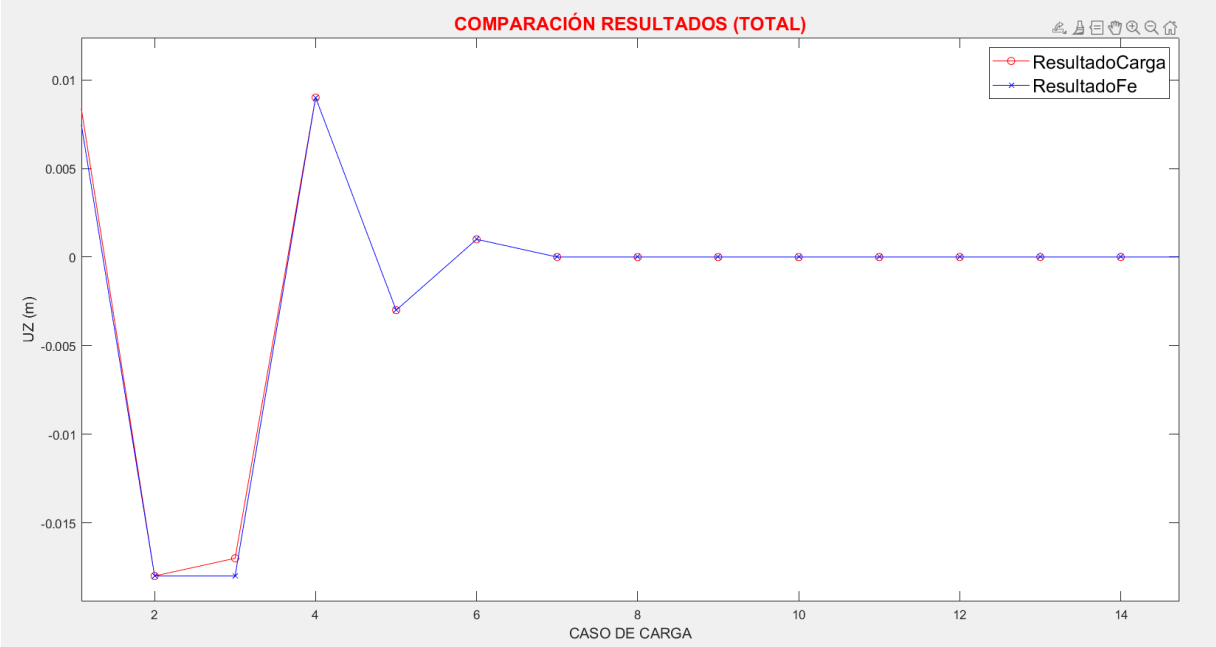

Figura 4.51 Segundo caso de carga correspondiente al estudio total de las tres pruebas de carga.

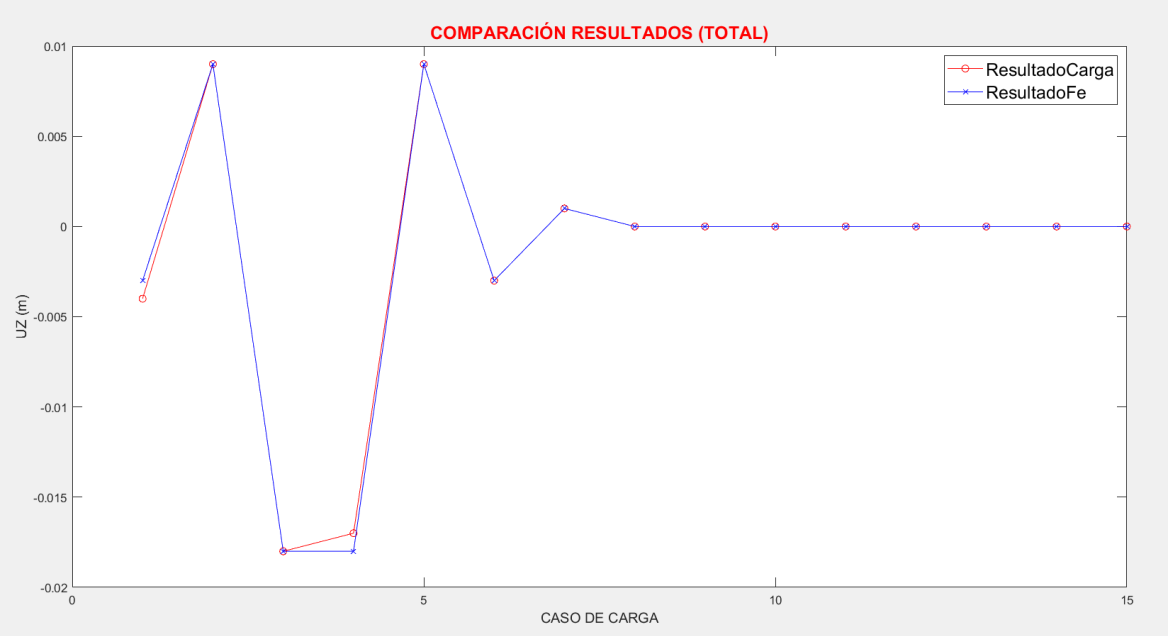

Figura 4.52 Tercer caso de carga correspondiente al estudio total de las tres pruebas de carga.

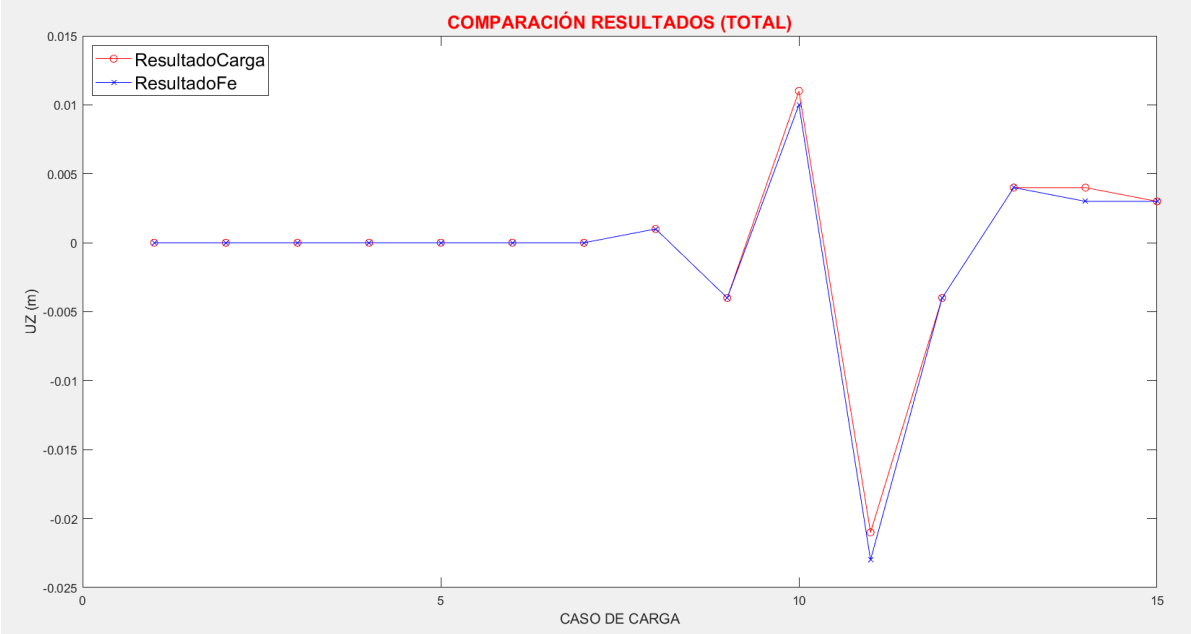

Figura 4.53 Decimoprimero caso de carga correspondiente al estudio total de las tres pruebas de carga.

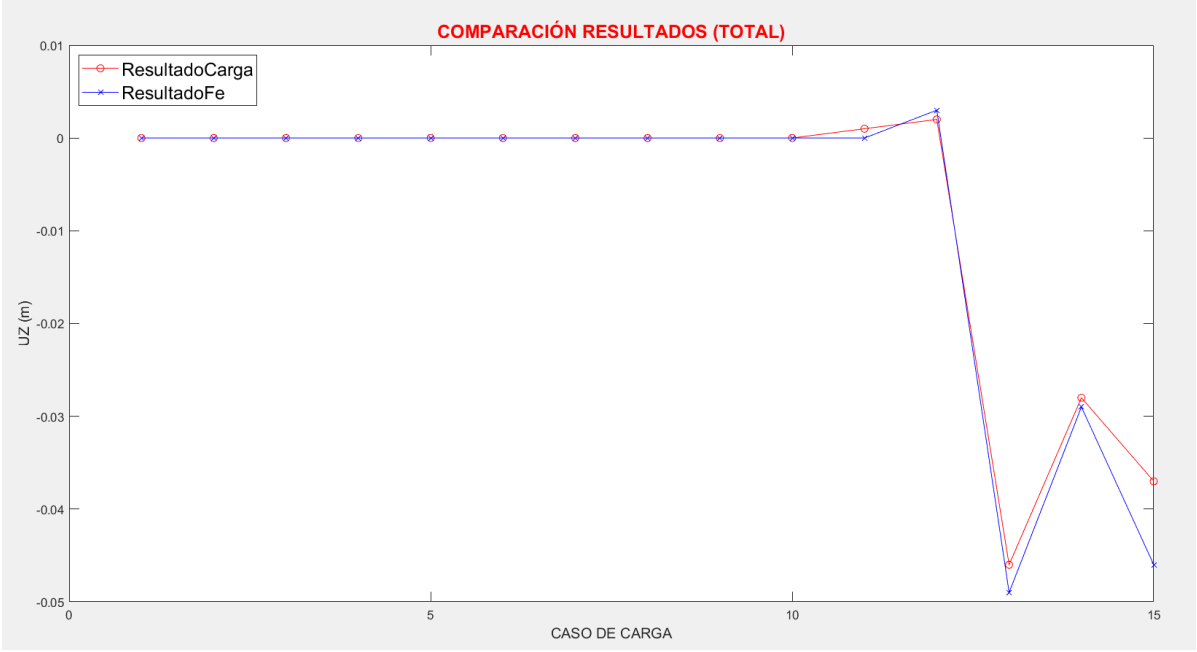

Figura 4.54 Decimosegundo caso de carga correspondiente al estudio total de las tres pruebas de carga.

PRUEBA 3

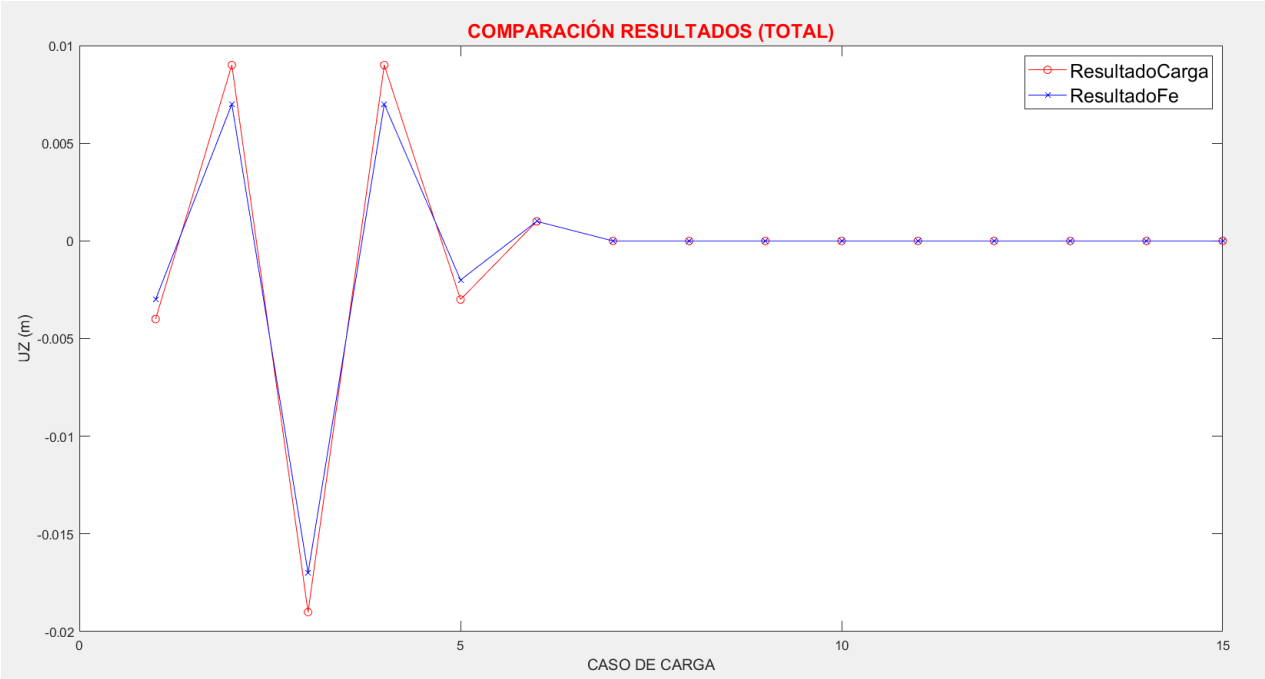

Figura 4.55 Tercer caso de carga correspondiente al estudio total de las tres pruebas de carga.

En conclusión, hemos observado (Figura 4.47 – 4.55) resultados consistentes con nuestras expectativas y con los hallazgos de estudios previos. Globalmente, encontramos que el acero presenta una alta resistencia a tracción, alrededor de 192 GPa. Además, los hormigones exhiben un módulo elástico a compresión notablemente alta, con valores de 31 GPa para el e30, 42 GPa para el e40, y 52 GPa para el e50. En cuanto al tiempo de computación, MATLAB ha tardado aproximadamente 1.87 horas, es decir, alrededor de 1 hora y 52 minutos. Este es un tiempo de procesamiento bastante reducido para un proceso de optimización de esta magnitud, especialmente considerando que en la aproximación inicial realizada anteriormente, el tiempo fue 12 minutos menor. Es importante tener en cuenta que en este caso se han llevado a cabo 200 iteraciones en muchos casos de carga porque no se ha alcanzado el 'ObjectiveLimit'. En el análisis anterior, como máximo, llegábamos a 100 iteraciones en algunos casos de carga. Notamos significativamente la influencia de la recalibración realizada. Además, debemos recordar las características previamente mencionadas del ordenador utilizado para este Trabajo de Fin de Grado.

Por lo tanto, los resultados obtenidos son extremadamente precisos y las cuatro variables de estudio se encuentran muy cerca de sus respectivos valores nominales, alineándose estrechamente con los valores utilizados para implementar el modelo. La precisión milimétrica en los cálculos y la notable cercanía a los valores esperados indican que el modelo está excepcionalmente bien calibrado. Los ajustes realizados han sido exitosos, logrando una exactitud y consistencia destacables. En resumen, todos los aspectos del modelo funcionan correctamente, reflejando una implementación y ejecución exitosa.

4.4 Análisis y Resultados

En este apartado veremos los resultados finales que se han obtenido del proceso de optimización. Veremos las soluciones mediante algoritmo genético del estudio total, así como la solución mediante subrogación del estudio completo de las 3 pruebas de carga. Por otro lado, veremos un estudio adicional acerca de las funciones objetivo, que consistirá principalmente en reemplazar las soluciones obtenidas de cada caso de carga en su respectiva prueba de carga, aplicándolas posteriormente en el resto de casos de carga de las pruebas correspondientes.

En primer lugar, en la Tabla 4.17 presentamos la solución mediante el uso de algoritmos genéticos.

Prueba TOTAL GA					
Fobj_mín	**X_optím**				**elapsed_time**
6.545854858209e-04	0.922441098947	1.02040473963	1.0445077794305	1.02528508984	33525.743156 s

Tabla 4.17 Resultados totales mediante el uso de algoritmos genéticos.

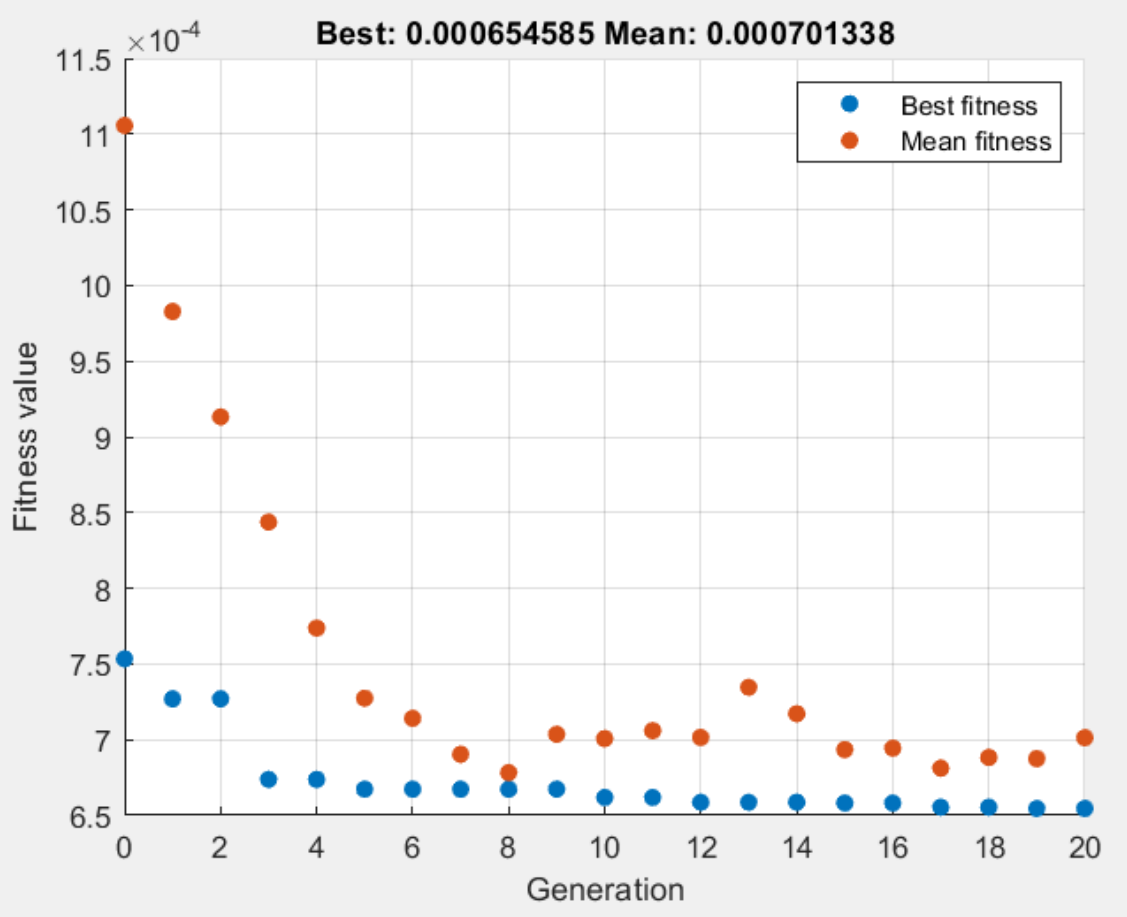

Figura 4.56 Proceso de optimización en MATLAB mediante algoritmos genéticos.

En la Figura 4.56, se aprecia una optimización mediante algoritmos genéticos con un total de 20 iteraciones que se han ejecutado en un total de 9.31 horas, es decir, 9 horas y 19 minutos aproximadamente. Se puede apreciar que no termina de converger la solución, pero sin embargo, se obtiene una buena solución con una gran precisión. Es evidente que la gran diferencia entre otras, de ambos métodos de optimización es el tiempo de computación, podemos apreciar en los resultados de subrogación esta gran discrepancia.

A continuación, se presenta en la Tabla 4.18 la solución del problema de optimización mediante subrogación.

Prueba TOTAL SUBROGACIÓN					
Fobj_mín	**X_optím**				**elapsed_time**
0.000646316558586918	0.915858670606212	1.043750	1.043750	1.041250	6715.737011 s

Tabla 4.18 Resultados totales mediante el uso de subrogación.

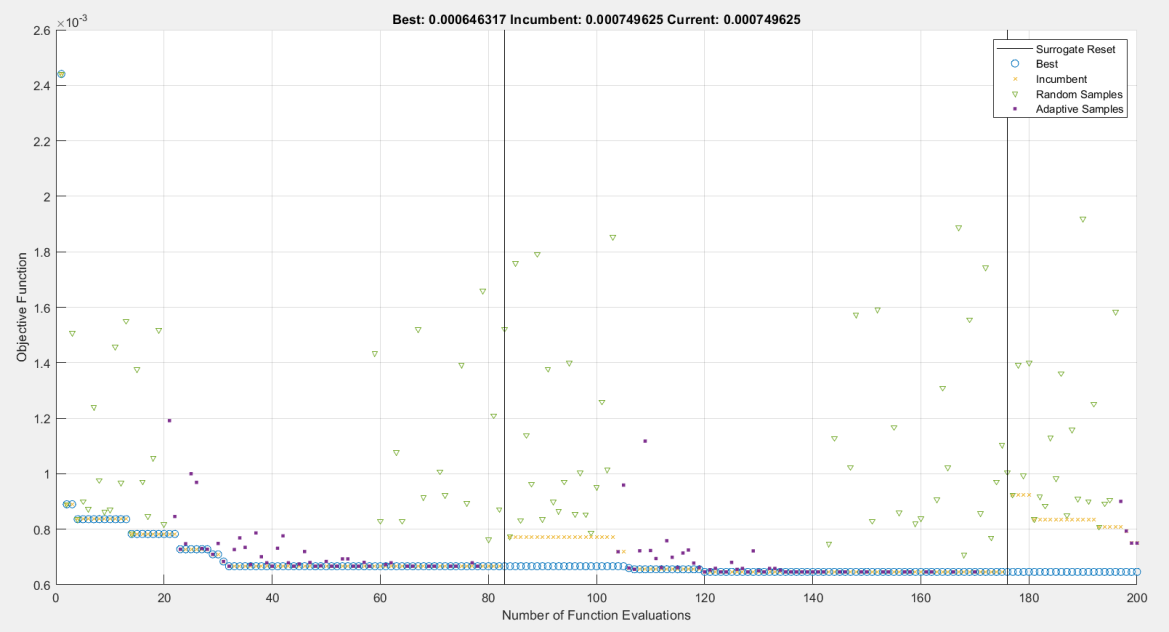

Figura 4.57 Proceso de optimización en MATLAB mediante subrogación.

En este estudio mediante subrogación se realizan 200 iteraciones en 1.87 horas, aproximadamente 1 hora y 53 minutos. La precisión es muy parecida, un poco superior la de subrogación, pero ambas son buenas. Las 2 soluciones son muy buenas, con valores de las 4 variables de estudio muy cerca de sus valores nominales.

Para finalizar el estudio de optimización de la estructura E9 mediante subrogación, hemos elaborado un análisis adicional, en el que compararemos las funciones objetivos individualmente en cada caso de carga de la respectiva prueba de carga que se esté analizando.

Para ello, empezamos con la primera prueba de carga. Habiendo recogido los resultados de las pruebas individuales de cada caso de carga en el apartado 4.3.5, realizaremos dos análisis, el de la prueba de carga número 1 en primer lugar, y posteriormente, analizaremos la segunda prueba de carga. Por tanto, para el primer caso, se obtiene una tabla 13x13 que se muestra en la siguiente imagen de MATLAB en la que podemos apreciar la uniformidad de los resultados obtenidos así como alguna peculiaridad interesante (Figura 4.58):

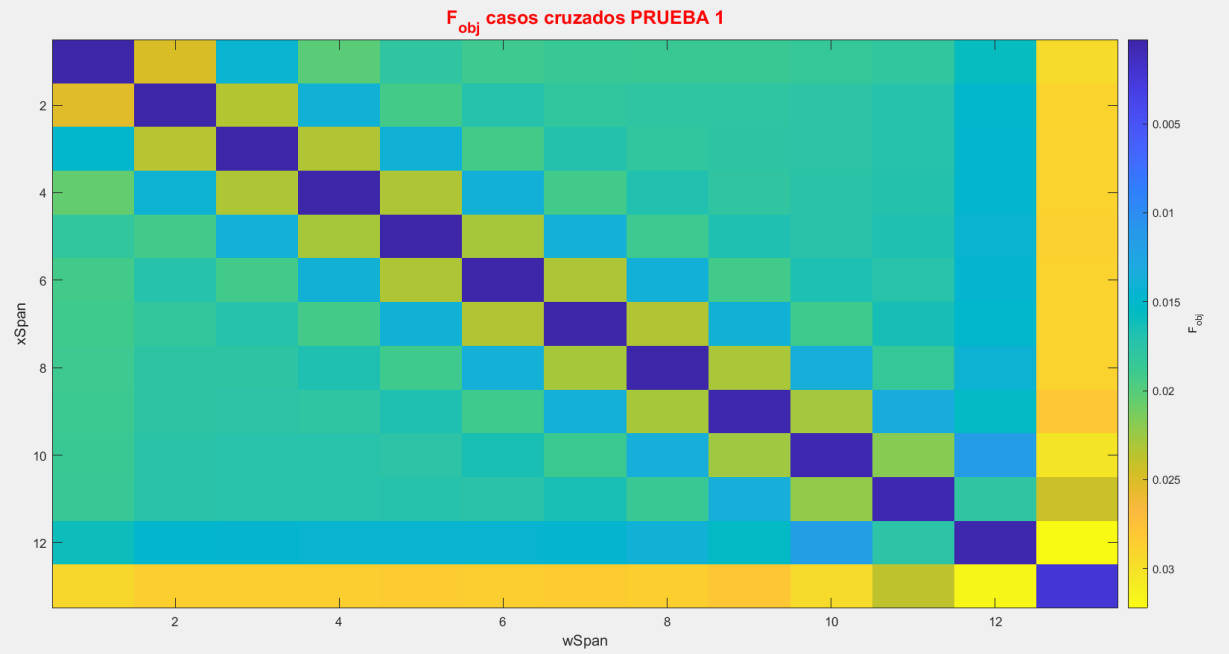

Figura 4.58 Funciones objetivo de cada caso de carga implementadas en cada uno de los casos de carga.

Este tipo de visualización es útil porque resume de manera eficiente las interacciones complejas entre múltiples escenarios, permitiendo identificar rápidamente qué casos podrían requerir una investigación adicional debido a su alto impacto cuando se aplican a otros casos. En este caso, podemos apreciar una notable característica del análisis, la diagonal principal, que al fin y al cabo, son los casos de carga evaluados individualmente, es decir, las funciones objetivo vistas en los apartados 4.3.5 y 4.3.6 anteriormente explicados, es por esta razón por la que son los valores con el color más oscuro de la figura.

Por otro lado, podemos observar un claro contraste de colores en las evaluaciones realizadas en el decimotercer vano. Dicho vano es curvo tal y como se ha dicho anteriormente en repetidas ocasiones y por tanto, puede dar lugar a incertidumbre en los resultados. Debida a esta razón, los valores de entrada de todos los casos de carga al introducirlos en el solucionador del vano número 13, salen resultados dispares, al igual que si se introduce los resultados del vano número 13 en el resto de vanos de estudio.

En segundo lugar, estudiamos los resultados de la segunda prueba de carga, que recordemos, solo tiene 12 casos de carga al ser casos de carga en los que se tienen en cuenta los vanos contiguos. Por lo tanto, obtenemos la siguiente matriz 12x12 de resultados que al igual que en el primer caso, se muestra la Figura 4.59 en la que se recoge dicha matriz:

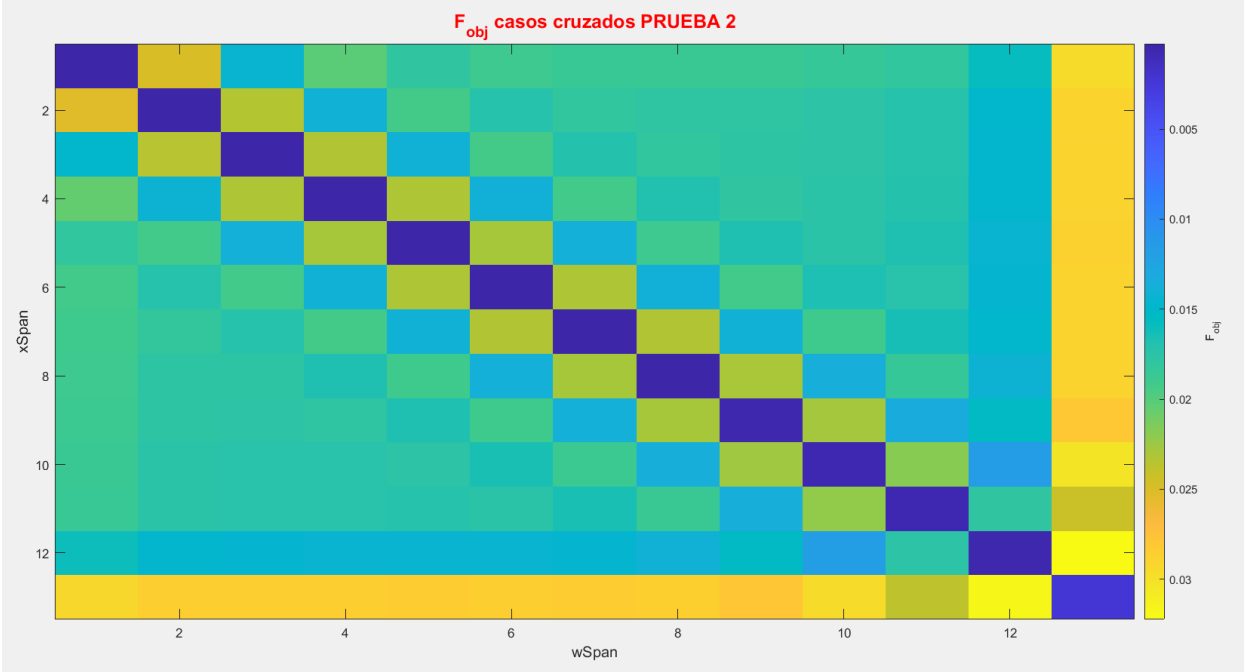

Figura 4.59 Funciones objetivo de cada caso de carga implementadas en cada uno de los casos de carga.

Los resultados de la segunda prueba de carga son notablemente similares a los de la primera prueba de carga y se obtienen por lo tanto, las mismas conclusiones.

En conclusión, este análisis ayuda a comprender de una mejor manera, la uniformidad y convergencia de los resultados. Dicha convergencia y uniformidad se hace presente especialmente en los vanos centrales de la estructura, que poseen una distribución, propiedades y características muy similares. Podemos concluir dicho estudio en el que se han llevado a cabo una serie de análisis y evaluaciones de un problema realmente complejo pero que con la ayuda de la subrogación se hace menos costoso en todos los ámbitos, especialmente en el ámbito de la computación. Se puede apreciar en el apéndice A.6 el código implementado para la obtención de este análisis.

5 CONCLUSIONES Y TRABAJOS FUTUROS

De manera general, hemos sido capaces de elaborar un modelo subrogado e implementarlo en un caso práctico real a través de la utilización de los programas de MATLAB y ANSYS. Hemos comprendido en profundidad el funcionamiento de los métodos de optimización de algoritmos genéticos y subrogación. Además, hemos obtenido una serie de conclusiones fundamentales en el uso de ambos métodos de optimización. Por último, hemos podido calibrar un modelo subrogado a partir de unos datos experimentales que nos ha servido para finalizar la primera parte de una herramienta de gran relevancia en la estructura E9 – Línea 1 de Metro de Sevilla.

Haber comprendido los métodos de optimización de los gemelos digitales nos ha sido de gran utilidad en vistas al caso práctico elaborado de una viga simplemente apoyada en el que hemos visto las ventajas e inconvenientes principales de ambos métodos. La rapidez computacional y la importancia de los 'trials' tanto para el ahorro en el tiempo de computación como para adaptar la función de subrogación a una posible variación en el problema (por un daño producido en la estructura, por ejemplo). En último lugar, el estudio de la estructura de metro del que hemos obtenido un modelo subrogado calibrado con sus respectivas variables críticas calibradas, cercanas a su valor nominal, tal y como indican los resultados experimentales de los que se parten.

A lo largo de este trabajo, hemos ido detectando aspectos en los que nos gustaría profundizar en un futuro, pero que por falta de resultados experimentales, no hemos sido capaces de abordar hasta ahora. La segunda parte de esta futura herramienta sería:

- Elaborar un modelo subrogado calibrado a partir de unos resultados experimentales en los que se recojan datos dinámicos de la estructura y a partir de los cuales, establezcamos una herramienta digital (teniendo en cuenta modos de vibración, parámetros modales, etc.) mediante la que podamos mantener la estructura de manera preventiva en su totalidad.

APÉNDICE A CÓDIGOS COMENTADOS

A.1 fitsurrogate.m

Código de MATLAB de optimización mediante subrogación para el cálculo de los casos individuales de la primera prueba de carga.

```
clear all; clc; close all;
format long

global kSpan file_name file_name2

resultados=zeros(13,6);

for kSpan=1:13
file_name2=sprintf('resultados5.mat');
file_name=sprintf('fitsurrogate_%i.mat',kSpan);

% Find minimum of function using genetic algorithm
nvars=5;
fun=@(x) optim_obj(x);

MFE=100;

rng default % For reproducibility

% [e210,e30,e40,e50,kx]
ubnominal=[210e9, 30e9, 40e9, 50e9, 1e8];

ub=ubnominal*5;
lb=ubnominal/10;

% para empezar
trials.X=[];
trials.Fval=[];
save(file_name,'trials')

options = optimoptions
('surrogateopt','ObjectiveLimit',0.010,'MaxFunctionEvaluations',MFE,'PlotFcn','surrogat
eoptplot');
options.InitialPoints=trials;
[x,fval,exitflag,output,trials] = surrogateopt(fun,lb,ub,options);


err=optim_obj(x);
```

```
resultados(kSpan,1)=fval;
resultados(kSpan,2:6)=x;
end

save(file_name2,'resultados')

%%

function err=optim_obj(x)
err=runansys(x(1),x(2),x(3),x(4),x(5));
end

%%

function Fobj=obj_f(ResultadoFe)


global kSpan

nSpan=13; % 13 vanos
nPost=15; % 15 puntos de estudio a lo largo de todo el puente
Precision=1e-3;

% read experimental test
PruebaCarga;

Fobj=0;

for iPrueba=1:1 %3     !!!!!!!!!!!!
    %
    for lSpan=kSpan:kSpan %1:nSpan !!!!!!!!!!!!!!!!!!!
        analysis_type=sprintf('prueba%i_%i',iPrueba,lSpan);

        if iPrueba==1 || (iPrueba==2 && lSpan<nSpan) || (iPrueba==3 && lSpan==3)


             Phi=[squeeze(ResultadoCarga(iPrueba,lSpan,:,1)-
ResultadoFe(iPrueba,lSpan,:,1));squeeze(ResultadoCarga(iPrueba,lSpan,:,2)-
ResultadoFe(iPrueba,lSpan,:,2))];
             Fobj=Fobj+sqrt(Phi'*Phi)/nPost/Precision/nSpan/2;
           % 0.0033     MAXIMA PRECISION QUE VAMOS A OBTENER. 1E-4/1E-3/30

        end
    end
end


end


%% RUN ANSYS FUNCTION
function err=runansys(e210,e30,e40,e50,kx)

global kSpan file_name

% read experimental test
PruebaCarga;

%
```

```
nSpan=13; % 13 vanos
nPost=15; % 15 puntos de estudio a lo largo de todo el puente

% write input file file
fileID = fopen('input_data.txt','w');
fprintf(fileID,"E210=%d\n",e210);
fprintf(fileID,"E30=%d\n",e30);
fprintf(fileID,"E40=%d\n",e40);
fprintf(fileID,"E50=%d\n",e50);
fprintf(fileID,"Kx=%d\n",kx);
fprintf(fileID,"kSpan=%i\n",kSpan);
fclose(fileID);

% run ansys
ansys_root='C:\Program Files\ANSYS Inc\ANSYS Student\v232\ansys\bin\winx64\ansys232';
ansys_inp='bridge_e9.ans';
ansys_exe=sprintf('"%s" -b -i %s -o report.out -np 1',ansys_root,ansys_inp);
[status,cmdout] =  system(ansys_exe);

% allocate fe results and error
ResultadoFe=zeros(3,nSpan,nPost,2); % prueba*nSpan*nPost*2

for iPrueba=1:1 %3     !!!!!!!!!!!!!
    %
    for lSpan=kSpan:kSpan %1:nSpan !!!!!!!!!!!!!!!!!!!!
        analysis_type=sprintf('prueba%i_%i',iPrueba,lSpan);

        if iPrueba==1 || (iPrueba==2 && lSpan<nSpan) || (iPrueba==3 && lSpan==3)

            % post
            filename=sprintf('%s.txt',analysis_type);
            AnsysResult = readtable(filename);

            % para componer giro en vano 13
            TH=zeros(15,1);
            TH(13)=asin((AnsysResult.X(13)-520)/105);
            TH(14)=asin((AnsysResult.X(14)-520)/105);
            TH(15)=asin((AnsysResult.X(15)-520)/105);

            ResultadoFe(iPrueba,lSpan,:,1)=AnsysResult.UZ-
            2.225*(AnsysResult.ROTX.*cos(TH)+AnsysResult.ROTY.*sin(TH));

            ResultadoFe(iPrueba,lSpan,:,2)=AnsysResult.UZ+2.225*(AnsysResult.ROTX.*cos(
            TH)+AnsysResult.ROTY.*sin(TH));

            % 0.0033     MAXIMA PRECISION QUE VAMOS A OBTENER. 1E-4/1E-3/30
        end
    end
end

% para guardar puntos iniciales
load(file_name,'trials')

trials.X(end+1,:)=[e210 e30 e40 e50 kx];

for iPrueba=1:1 %1:3
    for lSpan=kSpan:kSpan %1:nSpan
        tam=size(trials.Fval,1);
        if tam==0
        trials.Fval(iPrueba,lSpan,:,:,end)=ResultadoFe(iPrueba,lSpan,:,:);
```

```
        else
        trials.Fval(iPrueba,lSpan,:,:,end+1)=ResultadoFe(iPrueba,lSpan,:,:);
        end
    end
end

save(file_name,'trials')

Fobj=obj_f(trials.Fval(:,:,:,:,end));

err=Fobj;
disp(err)

%
end
```

A.2 surrogate_analit.m

Código de MATLAB del cálculo analítico del caso práctico a través de los métodos de optimización de algoritmos genéticos y subrogación. Ha de cambiarse la función de optimización en función del método que se desee emplear.

```
clear all; clc; close all;
% Find minimum of function using surrogate model
nvars=1;
fun=@(x) optim_obj(x);

% [e40]
lb=[35e9];
ub=[45e9];
MFE=100;
rng default % For reproducibility

% para empezar
trials.X=[];
trials.Fval=[];
save trials_analit.mat trials

% evaluo funcion objetivo
% load trials_analit.mat trials
% trials.Fval=objective_fun(trials.Fval);

% defino puntos iniciales
options.InitialPoints=trials;

% surrogacion
options = optimoptions
('surrogateopt','MaxFunctionEvaluations',MFE,'PlotFcn','surrogateoptplot')

[x,fval,exitflag,output,trials] = surrogateopt(fun,lb,ub,options); % estamos
resolviendo con SURROGACIÓN.

% check fit: 199446645305.976   29633103012.4187   39398755154.6921   49602923354.5523
      13511232.3174059
```

```
err=optim_obj(x);

%%
function err=optim_obj(x)
err=runansys(x(1));
end

%%
function Fobj=objective_fun(flecha_trial)

p=5000;
l=10;
x=[l/3 0.5*l 2*l/3];

% read experimental test
ei_obj=42e9*1000e-5;
flecha_obj=((p*x)/(24*ei_obj)).*(x.^3-2*l*x.^2+l^3);

Fobj=norm(flecha_obj-flecha_trial);
end

%% RUN ANSYS FUNCTION
function err=runansys(e40)

% run ansys
p=5000;
l=10;
x=[l/3 0.5*l 2*l/3];

ei_trial=e40*1000e-5;
flecha_trial=((p*x)/(24*ei_trial)).*(x.^3-2*l*x.^2+l^3);

% allocate fe results and error
Fobj=objective_fun(flecha_trial);

% error
err=Fobj;
disp(err)

% para guardar puntos iniciales
load trials_analit.mat trials

trials.X(end+1,1)=e40;
trials.Fval(end+1,:)=flecha_trial;

save trials_analit.mat trials

%
end
```

A.3 surrogate1var.m

Código MATLAB de cálculo FEM de 1 variable de estudio del caso práctico. Ha de cambiarse la función de optimización en función del método que se desee emplear.

```
clear all; clc; close all;
tic
% Find minimum of function using surrogate model
nvars=1;
fun=@(x) optim_obj(x);

% [e40]
lb=[35e9];
ub=[45e9];
MFE=50;
rng default % For reproducibility

% para empezar
trials.X=[];
trials.Fval=[];
save trials_fe.mat trials

% evaluo funcion objetivo
% load trials_analit.mat trials
% trials.Fval=objective_fun(trials.Fval);

% defino puntos iniciales
options.InitialPoints=trials;

% surrogacion
options = optimoptions
('surrogateopt','MaxFunctionEvaluations',MFE,'PlotFcn','surrogateoptplot')

[x,fval,exitflag,output,trials] = surrogateopt(fun,lb,ub,options); % estamos
resolviendo con SURROGACIÓN.

% check fit: 199446645305.976   29633103012.4187   39398755154.6921   49602923354.5523
      13511232.3174059
err=optim_obj(x);
toc
%%
function err=optim_obj(x)
err=runansys(x(1));
end

%% RUN ANSYS FUNCTION
function err=runansys(e40)

% read experimental test
PruebaCarga_1var;

%
nSpan=1;
nPost=1;
Precision=1e-3;

% write input file file
fileID = fopen('input_data_1var.txt','w');
fprintf(fileID,"e40=%d\n",e40);
fclose(fileID);

% run ansys
ansys_root='C:\Program Files\ANSYS Inc\ANSYS Student\v232\ansys\bin\winx64\ansys232';
ansys_inp='FEM_prueba_viga_1var.txt';
ansys_exe=sprintf('"%s" -b -i %s -o report.out -np 1',ansys_root,ansys_inp);
```

```
dos(ansys_exe);

% allocate fe results and error
Fobj=0;

analysis_type=sprintf('estatico%i_1var',1);

% post
filename=sprintf('%s.txt',analysis_type);
AnsysResult = readtable(filename);

ResultadoFe=AnsysResult.UZ;

Phi=ResultadoCarga_1var-ResultadoFe;
Fobj=sqrt(Phi'*Phi);

% error
%err=norm(squeeze(ResultadoCarga(:)-ResultadoFe(:)))/norm(squeeze(ResultadoCarga(:)));
err=Fobj;
disp(err)
%

% para guardar puntos iniciales
load trials_fe.mat trials

trials.X(end+1,1)=e40;
trials.Fval(end+1,:)=ResultadoFe;

save trials_fe.mat trials

%
end
```

A.4 FEM_prueba_viga_1var.txt

Código de FEM ANSYS de 1 variable de estudio del caso práctico. Para el caso de 3 variables ha de añadirse dichas variables al código junto con sus respectivas propiedades y localización correspondiente en la viga de estudio.

```
/CLEAR

/input,input_data_1var,txt
/PREP7

ET,1,BEAM188                    ! tipo de elemento

SECTYPE,1,BEAM,RECT             ! tipo de seccion. seccion viga y rectangular
```

```
SECDATA,0.025,0.025                    ! dimensiones de la seccion bxh
SECCONTROL,,,,0                        ! masa por ud de lgtud

MP,EX,1,e40                            ! modulo elastico. se ponen 1 por q es el material numero 1.
MP,NUXY,1,0.25                         ! coeficiente de Poisson
MP,DENS,1,2500                         ! densidad del material [kg/m^3]

N,1,,,,
N,101,1,,,
FILL,1,101

E,1,2
EGEN,100,1,ALL

! fuerza a aplicar

SFBEAM,ALL,1,PRES,5000                 ! PRESION EN LA DIRECCION NEGATIVA DE Z. SFBEAM PORQUE ES UNA VIGA
D,1,UX,,,,,UY,UZ,ROTX,ROTZ,            ! apoyo simple (el triangulito).
D,101,,,,,,UY,UZ,ROTX,ROTZ,             ! apoyo con rueditas (carrito).

! Solución de análisis estático

/SOL
ANTYPE,0
SOLVE
FINISH

/POST1
npt=node(0.5,0,0)                      ! 0.5 * L. L en este caso es 1
                                       ! cada modo de vibracion va en un archivo diferente
*do,iFreq,1,1                          ! BUCLE : *do,nombre de la variable,valor inicial,valor final
analysis_type='estatico%iFreq%_1var'   ! estatico%iFreq% nos proporciona que se generara un fichero con nombre estatico1, estatico2, estatico3,etc. dependiendo
                                       ! de como de largo sea el bucle, *do,iFreq,i=1,j=1,2,...,n.
set,1,iFreq                            ! se que hay que leer los resultados de
*get,frequency,MODE,iFreq,FREQ         ! *GET, Par, Entity, ENTNUM, Item1

*cfopen,%analysis_type%,txt
```

```
*vwrite
('FREQ NODE X Y Z UX UY UZ ROTX ROTY ROTZ')

*vwrite,frequency,npt,nx(npt),ny(npt),nz(npt),ux(npt),uy(npt),uz(npt),rotx(npt),roty(npt),rotz(npt)
(f11.4,f8.0,f11.4,f11.4,f11.4,e,e,e,e,e,e)                    ! formatos decimales (f11) con 4 decimales(.4), enteros (f8) con 0 decimales (.0) y exponenciales (e).

*cfclose

*enddo

!  Solución Análisis Modal
! /SOLU
! ANTYPE,2
! MODOPT,LANB,10
! MXPAND,10,,,YES
! allsel
! solve
! finish                    ! Initiate modal solution



! /POST1
! npt=node(lgtud/2,0,0)

! *do,iFreq,1,5
! analysis_type='modal%iFreq%'
! set,1,iFreq
! *get,frequency,MODE,iFreq,FREQ

! *cfopen,%analysis_type%,txt
! *vwrite
! ('FREQ NODE X Y Z UX UY UZ ROTX ROTY ROTZ')

! *vwrite,frequency,npt,nx(npt),ny(npt),nz(npt),ux(npt),uy(npt),uz(npt),rotx(npt),roty(npt),rotz(npt)
! (f11.4,f8.0,f11.4,f11.4,f11.4,e,e,e,e,e,e)

! *cfclose
```

```
! *enddo
```

A.5 bridge_e9_aro_surrogate.m & prueba_carga_aro_surrogate.m

Código FEM en ANSYS correspondiente al análisis completo de las tres pruebas de carga.

A.5.1 bridge_e9_aro_surrogate.m

```
%bridge_e9_aro_surrogate

!*---------------------------
! read external data
!*---------------------------
/inp,input_data,txt

!*---------------------------
! DATA
!*---------------------------
NumberOfSpans=13
NumberOfModes=20

*dim,LengthOfSpan,array,NumberOfSpans
LengthOfSpan(1)=44, 44, 44, 44, 44, 44, 44, 44, 44, 44, 44, 36, 54

*dim,CoordOfSpan,array,NumberOfSpans+1
CoordOfSpan(1)=0, 44 ,88, 132, 176, 220, 264, 308, 352, 396, 440, 484, 520, 574

NumberOfNodesPost=15
*dim,NodesPost,array,NumberOfNodesPost
NodesPost(1)=22, 66, 110, 154, 198, 242, 286, 330, 374, 418, 462, 502, 547, 533.5, 560.5

ElementLength=1

! masa muerta
W1=8.80     ! ANCHO TABLERO
TP=0.0365   ! ESPESOR PAVIMENTO
```

```
DP=2500      ! DENSIDAD PAVIMENTO

/PREP7
/ESHAPE,0

!------------------------
! ELEMENT
!------------------------

ET,1,BEAM188,,,
KEYOPT, 1, 12, 1

ET,2,MPC184,1,,
KEYOPT, 2, 2, 1

ET,3,LINK11

!------------------------
! MATERIAL PROPERTIES
!------------------------

MP,EX,1,e210
MP,NUXY,1,0.3
MP,DENS,1,7850

MP,EX,2,e30
MP,NUXY,2,0.2
MP,DENS,2,2500

MP,EX,3,e40
MP,NUXY,3,0.2
MP,DENS,3,2500

MP,EX,4,e50
MP,NUXY,4,0.2
MP,DENS,4,2500

!------------------------
```

```
! REAL CONSTANTS
!------------------------

R,3,Kx

!------------------------
! SECTIONS
!------------------------

*do,iSec,1,16
SECTYPE, iSec, BEAM, MESH
SECREAD,'SEC%iSec%','SECT', ,'MESH'
SECOFFSET, origin
SECCONTROL, , , , DP*W1*TP! ADDED MASS PER UNIT LENGTH.
*enddo

!-----------------------
!
! GEOMETRY
!
!-----------------------

! keypoints
*do,iSpan,1,NumberOfSpans
k,100+iSpan,CoordOfSpan(iSpan),,
k,200+iSpan,CoordOfSpan(iSpan)+10.5,,
k,400+iSpan,CoordOfSpan(iSpan+1)-10.5,,
*enddo

! curva vano 13
local, 11, 1, CoordOfSpan(NumberOfSpans),105,,-90
csys,11

kmodif,200+iSpan,105,5.7,,
k,300+iSpan,105,11.5,,
kmodif,400+iSpan,105,24.5,,
k,100+NumberOfSpans+1,105,31,

csys,0
```

```
!---------------------- span 1
iSpan=1

kmodif,200+iSpan,CoordOfSpan(iSpan)+10.85,,
k,300+iSpan,CoordOfSpan(iSpan)+10.85+14,,

l,100+iSpan,200+iSpan
latt, , , 1, , , , 1
allsel,all
lsel, inve

l,200+iSpan,300+iSpan
latt, , , 1, , , , 2
allsel,all
lsel, inve

l,300+iSpan,400+iSpan
latt, , , 1, , , , 1
allsel,all
lsel, inve

! final section
sectype,200+iSpan,TAPER
secdata,3, CoordOfSpan(iSpan+1)-10.5, 0
secdata,4, CoordOfSpan(iSpan+1), 0

l,400+iSpan,100+iSpan+1
latt, , , 1, , , , 200+iSpan
allsel,all
lsel, inve

!---------------------- span 2-11

! lines: loop over spans
*do,iSpan,2,11

! initial section
```

```
SECTYPE,100+iSpan,TAPER
SECDATA,5, CoordOfSpan(iSpan), 0 ! STARTING LOCATION OF TAPERED BEAM
SECDATA,6, CoordOfSpan(iSpan)+10.5, 0 ! ENDING LOCATION OF TAPERED BEAM

l,100+iSpan,200+iSpan
latt, , , 1, , , , 100+iSpan
allsel,all
lsel, inve

! intermidiate section
l,200+iSpan,400+iSpan
latt, , , 1, , , , 7
allsel,all
lsel, inve

! final section
SECTYPE,200+iSpan,TAPER
SECDATA,6, CoordOfSpan(iSpan+1)-10.5, 0 ! STARTING LOCATION OF TAPERED BEAM
SECDATA,5, CoordOfSpan(iSpan+1), 0 ! ENDING LOCATION OF TAPERED BEAM

l,400+iSpan,100+iSpan+1
latt, , , 1, , , , 200+iSpan
allsel,all
lsel, inve
*enddo

!---------------------- span 12
iSpan=12

k,300+iSpan,CoordOfSpan(iSpan)+21,,

! initial section
SECTYPE,100+iSpan,TAPER
SECDATA,5, CoordOfSpan(iSpan), 0 ! STARTING LOCATION OF TAPERED BEAM
SECDATA,6, CoordOfSpan(iSpan)+10.5, 0 ! ENDING LOCATION OF TAPERED BEAM

l,100+iSpan,200+iSpan
latt, , , 1, , , , 100+iSpan
allsel,all
```

```
lsel, inve

! intermidiate section
l,200+iSpan,300+iSpan
latt, , , 1, , , , 7
allsel,all
lsel, inve

! intermidiate section
l,300+iSpan,400+iSpan
latt, , , 1, , , , 8
allsel,all
lsel, inve

! final section
SECTYPE,200+iSpan,TAPER
SECDATA,8, CoordOfSpan(iSpan+1)-10.5, 0 ! STARTING LOCATION OF TAPERED BEAM
SECDATA,9, CoordOfSpan(iSpan+1), 0 ! ENDING LOCATION OF TAPERED BEAM

l,400+iSpan,100+iSpan+1
latt, , , 1, , , , 200+iSpan
allsel,all
lsel, inve

!---------------------- span 13
iSpan=13
csys,11
!SELTOL, 10

! initial section
!SECTYPE,100+iSpan,TAPER
!SECDATA,10, kx(100+iSpan), ky(100+iSpan) ! STARTING LOCATION OF TAPERED BEAM
!SECDATA,11, kx(200+iSpan), ky(200+iSpan) ! ENDING LOCATION OF TAPERED BEAM

l,100+iSpan,200+iSpan
!latt, , , 1, , , , 100+iSpan
latt, , , 1, , , , 15
allsel,all
```

```
lsel, inve

! intermidiate section
!SECTYPE,300+iSpan,TAPER
!SECDATA,12, kx(200+iSpan), ky(200+iSpan) ! STARTING LOCATION OF TAPERED BEAM
!SECDATA,13, kx(300+iSpan), ky(300+iSpan) ! ENDING LOCATION OF TAPERED BEAM

l,200+iSpan,300+iSpan
!latt, , , 1, , , , 300+iSpan
latt, , , 1, , , , 16
allsel,all
lsel, inve

! intermidiate section
l,300+iSpan,400+iSpan
latt, , , 1, , , , 14
allsel,all
lsel, inve

! final section
!SECTYPE,200+iSpan,TAPER
!SECDATA,13, kx(400+iSpan), ky(400+iSpan) ! STARTING LOCATION OF TAPERED BEAM
!SECDATA,12, kx(100+iSpan+1), ky(100+iSpan+1) ! ENDING LOCATION OF TAPERED BEAM

l,400+iSpan,100+iSpan+1
!latt, , , 1, , , , 200+iSpan
latt, , , 1, , , , 16
allsel,all
lsel, inve

!seltol
csys,0


!-----------------------
!
! MESH
!
!-----------------------
```

```
allsel,all
LESIZE,ALL,ElementLength
LMESH, ALL

! group elements
CM, rail_element,element
CM, rail_node,node

! supports
TYPE,2

*DO,iSpan,1,NumberOfSpans+1

*if,iSpan,eq,NumberOfSpans+1,then
csys,11
NPT=NODE(105,31,0)
N,1000+iSpan,105-1.8,31,-1.8
N,2000+iSpan,105+1.8,31,-1.8
csys,0
*else
NPT=NODE(CoordOfSpan(iSpan),0,0)
N,1000+iSpan,CoordOfSpan(iSpan),-1.8,-1.8
N,2000+iSpan,CoordOfSpan(iSpan),1.8,-1.8
*endif

!MPC
E,NPT,1000+iSpan
E,NPT,2000+iSpan
*ENDDO

! end link
csys,11
NPT=NODE(105,31,0)
N,3000,105,31.01,
csys,0

TYPE,3
```

```
REAL,3

E,NPT,3000

!-----------------------
!
! BOUNDARY CONDITIONS
!
!-----------------------

D, 1001, , , , 1001+NumberOfSpans, 1, UZ, , , , ,
D, 2001, , , , 2001+NumberOfSpans, 1, UY, UZ, , , ,
D, 3000, all

!-----------------------
!
! SOLUTION
!
!-----------------------
/inp,prueba_carga_aro_surrogate,ans
```

A.5.2 prueba_carga_aro_surrogate.m

```
!-----------------------
!
! load cases
!
!-----------------------

!
lsnum=0

! loop over load type
*do,iPrueba,1,1 !!!!!!!!!!!!!!
analysis_type='prueba%iPrueba%'
```

```
! loop over load cases
*do,lSpan,kSpan,kSpan !1,NumberOfSpans !!!!!!!!!!!!!

! delete previous load cases
allsel
fdele, all, all
sfedele, all, all, all

*if,analysis_type,eq,'prueba1',then

lsnum=lsnum+1


lsel, s, loc, x, CoordOfSpan(lSpan), CoordOfSpan(lSpan+1), , 1
*if,lSpan,le,11,then
sfbeam, all, 1, pres, 59.1e3
*elseif,lSpan,eq,12,then
sfbeam, all, 1, pres, 57.8e3
*elseif,lSpan,eq,13,then
sfbeam, all, 1, pres, 57.8e3
*endif


allsel
lswrite, lsnum


*elseif,analysis_type,eq,'prueba2',and,lSpan,lt,NumberOfSpans,then

lsnum=lsnum+1

lsel, s, loc, x, CoordOfSpan(lSpan), CoordOfSpan(lSpan+1), , 1
*if,lSpan,le,11,then
sfbeam, all, 1, pres, 46.4e3
*elseif,lSpan,eq,12,then
sfbeam, all, 1, pres, 42.5e3
*elseif,lSpan,eq,13,then
sfbeam, all, 1, pres, 47.2e3
*endif
```

```
lsel, s, loc, x, CoordOfSpan(lSpan+1), CoordOfSpan(lSpan+2), , 1
*if,lSpan+1,le,11,then
sfbeam, all, 1, pres, 47.3e3
*elseif,lSpan+1,eq,12,then
sfbeam, all, 1, pres, 43.3e3
*elseif,lSpan+1,eq,13,then
sfbeam, all, 1, pres, 48.1e3
*endif


allsel
lswrite, lsnum

*elseif,analysis_type,eq,'prueba3',and,lSpan,eq,3,then

lsnum=lsnum+1

lSpan=3
lsel, s, loc, x, CoordOfSpan(lSpan), CoordOfSpan(lSpan+1), , 1
sfbeam, all, 1, pres, 23.6e3
f, all, mx, 69.6e3*ElementLength ! carga sobreestimada en nodos extremos


allsel
lswrite, lsnum


*endif
*enddo
*enddo

!----------------------
!
! solution
!
!----------------------
/solu

allsel
```

```
! seltol, 10   ! for taper curved beams
lssolve, 1, lsnum,
finish

!-----------------------
!
! export mode shapes
!
!-----------------------
/post1
! seltol

!
lsnum=0

! loop over load type
*do,iPrueba,1,1 !!!!!!!!!!!!!!!
analysis_type='prueba%iPrueba%'

! loop over load cases
*do,lSpan,kSpan,kSpan !1,NumberOfSpans !!!!!!!!!!!!!!!!!!!
flag=0

*if,analysis_type,eq,'prueba1',then
flag=1
lsnum=lsnum+1



*elseif,analysis_type,eq,'prueba2',and,lSpan,lt,NumberOfSpans,then
flag=1
lsnum=lsnum+1

*elseif,analysis_type,eq,'prueba3',and,lSpan,eq,3,then
flag=1
lsnum=lsnum+1


*endif
```

```
! write ouput file
*if,flag,eq,1,then
set,lsnum

! open file and write header
*cfopen,%analysis_type%_%lSpan%,txt
*vwrite
('FREQ NODE X Y Z UX UY UZ ROTX ROTY ROTZ')


! loop over observation points
*do, j, 1, NumberOfNodesPost
npt=node(NodesPost(J),0,0)


*vwrite,0,npt,nx(npt),ny(npt),nz(npt),ux(npt),uy(npt),uz(npt),rotx(npt),roty(npt),rotz(npt)
(f11.4,f8.0,f11.4,f11.4,f11.4,e,e,e,e,e,e)

*enddo
*cfclose
*endif
!
*enddo
*enddo
```

A.6 cruzados_surrogate_aro.m

Cálculo de funciones objetivo cruzadas en cada caso de carga. Para el caso de funciones objetivo cruzadas correspondiente a la segunda prueba de carga, ha de sustituirse ‘iPrueba=1’ por ‘iPrueba=2’.

```
clear all; clc; close all;
format long
global file_name1
file_name1=sprintf('cruzados_aro_surrogate_1.mat');
vector1=casos_cruzados;
save (file_name1,'vector1')

load (file_name1)
figure (1)
imagesc(vector1)
```

```
c=colorbar('Direction','reverse');
c.Label.String = 'F_{obj}';
title('F_{obj} casos cruzados PRUEBA 1','FontSize',15,'Color','r')
xlabel('wSpan','FontSize',12)
ylabel('xSpan','FontSize',12)
%% RUN ANSYS FUNCTION
function cruces=casos_cruzados(prop)

global xSpan wSpan

% read experimental test
PruebaCarga;

nSpan=13;
nPost=15;
iPrueba=1;
load resultados.mat

for xSpan=1:nSpan
    %
            xref=[210e9 30e9 40e9 50e9];
            prop=resultados(xSpan,2:5);
            prop=prop.*xref;
            % write input file file
            fileID = fopen('input_data.txt','w');
            fprintf(fileID,"E210=%d\n",prop(1));
            fprintf(fileID,"E30=%d\n",prop(2));
            fprintf(fileID,"E40=%d\n",prop(3));
            fprintf(fileID,"E50=%d\n",prop(4));
            fprintf(fileID,"Kx=%d\n",1e8);
            fprintf(fileID,"xSpan=%d\n",xSpan);
            fclose(fileID);

            % delete ansys files
            delete file.*

            % run ansys
            ansys_root='C:\Program Files\ANSYS Inc\ANSYS
Student\v232\ansys\bin\winx64\ansys232';
            ansys_inp='bridge_e9_aro_surrogate_cruzados.ans';
            ansys_exe=sprintf('"%s" -b -i %s -o report.out',ansys_root,ansys_inp);
            dos(ansys_exe);

            analysis_type=sprintf('prueba%i_%i',iPrueba,xSpan);

            ResultadoFe=zeros(3,nSpan,nPost,2); % prueba*nSpan*nPost*2

            lCase=0; % number of load cases

            %
            lCase=lCase+iPrueba^-1;

            % post
            filename=sprintf('%s.txt',analysis_type);
            AnsysResult = readtable(filename);

            % para componer giro en vano 13
            TH=zeros(15,1);
            TH(13)=asin((AnsysResult.X(13)-520)/105);
            TH(14)=asin((AnsysResult.X(14)-520)/105);
```

```
            TH(15)=asin((AnsysResult.X(15)-520)/105);

            ResultadoFe(iPrueba,xSpan,:,1)=AnsysResult.UZ-
2.225*(AnsysResult.ROTX.*cos(TH)+AnsysResult.ROTY.*sin(TH));

ResultadoFe(iPrueba,xSpan,:,2)=AnsysResult.UZ+2.225*(AnsysResult.ROTX.*cos(TH)+AnsysRes
ult.ROTY.*sin(TH));

            ResultadoFe=round(ResultadoFe,3);

    for wSpan=1:nSpan % 1:nSpan


            % allocate fe results and error
            Fobj=0;

            Phi=[squeeze(ResultadoCarga(iPrueba,wSpan,:,1)-
ResultadoFe(iPrueba,xSpan,:,1));squeeze(ResultadoCarga(iPrueba,wSpan,:,2)-
ResultadoFe(iPrueba,xSpan,:,2))];
            % Phi=[squeeze(ResultadoCarga(iPrueba,lSpan,:,1)-
ResultadoCarga(iPrueba,lSpan,:,1)+Precision*0.3);squeeze(ResultadoCarga(iPrueba,lSpan,:
,2)-ResultadoCarga(iPrueba,lSpan,:,2)+Precision*0.3)];

            Fobj=Fobj+sqrt(Phi'*Phi)*iPrueba^-1;
            Fobj=Fobj/(2*nPost)^.5/lCase; % dividido por 2 por npost*2

            cruces(xSpan,wSpan)=Fobj;
    end
end

disp(cruces)
% delete temp files
delete prueba1_*.txt prueba2_*.txt prueba3_*.txt

%
% fprintf('e210 = %0.4e --- e30 = %0.4e --- e40 = %0.4e --- e50 = %0.4e --- kx = %0.4e
--- fobj = %d \n',e210,e30,e40,e50,kx,Fobj)

end
```

REFERENCIAS

[1] H. Rongrong y X. Yong. *Review on the new development of vibration-based damage identification for civil engineering structures: 2010–2019*. En: Journal of Sound and Vibration 491, pág. 115741 (2021).

[2] C.R. Farrar, S.W. Doebling y D.A. Nix. *Vibration-based structural damage identification*. En: Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 359.1778, págs. 131 – 149 (2001).

[3] E. Figueiredo y J. Brownjohn. *Three decades of statistical pattern recognition paradigm for SHM of bridges*. En: Structural Health Monitoring 21.6, págs. 3018 – 3054 (2022).

[4] N. Bakhary, H. Hao y A. Deeks. *Structure damage detection using neural network with multi-stage substructuring*. En: Advances in Structural Engineering 13.1, págs. 95 – 110 (2010).

[5] A. Rytter, *Vibrational Based Inspection of Civil Engineering Structures*, Ph.D. dissertation, Aalborg University (1993).

[6] Doebling, S. W., Farrar, C. R., Prime, M. B., & Shevitz, D. W. *Damage identification and health monitoring of structural and mechanical systems from changes in their vibration characteristics*: A literature review (1996).

[7] Hakim, S. J. S., & Abdul Razak, H. *Modal parameters based structural damage detection using artificial neural networks* - a review. Smart Structures and Systems, 14(2), 159-189 (2014).

[8] Chakraborty, S., Adhikari, S., & Ganguli, R. *The role of surrogate models in the development of digital twins of dynamic systems*. Applied Mathematical Modelling, 90, 662-681 (2021).

[9] Tuegel, E. J., Ingraffea, A. R., Eason, T. G., & Spottswood, S. M. *Reengineering aircraft structural life prediction using a digital twin*. International Journal of Aerospace Engineering (2011).

[10] Davis, S. E., Cremaschi, S., & Eden, M. R. *Efficient surrogate model development: Optimum model form based on input function characteristics*. Computer Aided Chemical Engineering, 40, 457-462 (2017).

[11] Yang, X., Guo, X., Ouyang, H., & Li, D. *A Kriging model based finite element model updating method for damage detection*. Applied Sciences, 7(10), 1039 (2017).

[12] Corlu, C. G., Akcay, A., & Xie, W. *Stochastic simulation under input uncertainty*: A review. Operations Research Perspectives, 7, 100162 (2020).

[13] Gutmann, H.-M. *A radial basis function method for global optimization*. Journal of Global Optimization, 19(2), 201-227 (2001).

[14] Regis, R., & Shoemaker, C. A. *A stochastic radial basis function method for the global optimization of expensive functions*. INFORMS Journal on Computing, 19(4), 497-509 (2007).