XXV CONGRESO NACIONAL DE INGENIERÍA MECÁNICA
Santander, 10-13 junio 2025
UC Universidad de Cantabria
IFToMM

Técnicas aplicadas en optimización topológica para la obtención de mecanismos compliant distribuidos

Alejandro Tobío[1], Ángel Suescun[1,2]

[1] *CEIT-Basque Research and Technology Alliance (BRTA), atobiopena@ceit.es*

[2] *Universidad de Navarra, Tecnun, asuescun@ceit.es*

En el diseño de mecanismos compliant, uno de los mayores retos es el control del efecto bisagra (hinge-effect). Los mecanismos compliant transmiten fuerza/movimiento sin uniones articuladas entre piezas, ya que son composiciones monolíticas. La salida del mecanismo depende directamente de la deformación del mismo, y esta a su vez de la geometría y las cargas aplicadas. Por lo tanto, afinar la geometría de un compliant es esencial para ajustar su comportamiento, sin poner en riesgo su integridad estructural durante el funcionamiento.

Los mecanismos compliant con mayor efecto bisagra poseen una geometría caracterizada por partes rígidas unidas entre sí por pequeñas zonas muy esbeltas de material que actúan como bisagra. Esta configuración alcanza grandes desplazamientos con precisión, pero concentra tensiones en las zonas de bisagra. Por otra parte, los mecanismos con menor efecto bisagra (compliant distribuido) poseen una geometría con transiciones más suaves, evitando puntos de concentración de tensión en el mecanismo y posibles fallos mecánicos.

Los mecanismos compliant suelen tener formas complejas que no son intuitivas, por lo que la optimización topológica es una herramienta muy útil para el diseño de estos mecanismos. Los algoritmos de optimización topológica tienden a generar diseños más afines a los basados en bisagras. Problemas ya conocidos en el campo de la optimización topológica, como el efecto ajedrez, favorecen este tipo de soluciones. A pesar de que existen técnicas documentadas que ayudan a evitar estos efectos, como son los filtros de sensibilidades de la función objetivo, no son suficientes para generar soluciones sin zonas locales de concentración de tensiones.

Se ha desarrollado un código Python de optimización topológica que utiliza el programa ABAQUS como entorno gráfico y solver de elementos finitos para la generación de mecanismos compliant. La herramienta creada es versátil y potente, el entorno gráfico facilita el diseño de volúmenes de partida complejos y el solver permite simular piezas 2D y 3D con mallados finos para un mayor detalle de la solución. Los diseños optimizados se pueden simular en ABAQUS con condiciones reales o exportar a cualquier programa CAD para su post proceso y preparación para fabricación.

Se presenta una metodología que potencia las soluciones compliant distribuidas. La estrategia desarrollada utiliza por una parte la función objetivo y sus sensibilidades para encontrar la mejor distribución de material posible, pero minimizando con técnicas inspiradas en tratamiento de imagen el efecto bisagra de dicha distribución.

El código desarrollado se ha mejorado y validado con ejemplos bibliográficos hasta obtener resultados satisfactorios en el diseño de mecanismos compliant distribuidos. Se presentan las técnicas y mejoras que se han implementado, ejemplos de resultados actuales que produce la herramienta y una comparativa entre un caso con y sin mejoras aplicadas.

1. Introducción

Un mecanismo compliant es un mecanismo monolítico que transmite fuerza y movimiento a través de la deformación elástica de su cuerpo. Al contrario que los mecanismos tradicionales, estos no disponen de piezas articuladas entre sí, sino que dependen exclusivamente de la flexibilidad relativa de la distribución de material. Tienden a ser diseños poco intuitivos, ya que requieren de un equilibrio delicado entre rigidez y flexibilidad para su correcto funcionamiento.

Una herramienta con mucho recorrido utilizada en el diseño de mecanismos compliant es la optimización topológica. Es un método matemático que tiene como finalidad encontrar la mejor distribución de material en un volumen de trabajo dado, mejorando el rendimiento bajo ciertas restricciones de diseño. En esencia, los métodos de optimización topológica buscan mejorar (minimizando o maximizando) una función objetivo que representa el comportamiento del sistema sometido a las condiciones de funcionamiento esperadas. El proceso de optimización es muy sensible a los parámetros iniciales y a los que afectan a cada iteración. Es fundamental tener un algoritmo de decisión robusto que facilite la convergencia de la solución de forma controlada, sin llegar a ser demasiado lento y caro computacionalmente.

Hay diferentes enfoques en optimización topológica sobre cómo utilizar la función objetivo para progresar en la optimización. Los principales métodos son los basados en sensibilidades y los estocásticos. El primero evalúa la derivada de la función objetivo respecto a la variable de diseño (sensibilidad del sistema), que representa la importancia que tiene cada elemento de la discretización en maximizar o minimizar la función objetivo planteada. Esta evaluación es la que guía la optimización, iteración tras iteración, hasta obtener el mejor reparto de material posible, según las condiciones iniciales dadas. Aunque los métodos por sensibilidades son los más comunes, pueden converger a soluciones locales sin capacidad para distinguirlas de una posible solución global. El segundo método aplica técnicas estocásticas con algún criterio dependiente de la cantidad de iteraciones realizadas y de la función objetivo, para que los efectos estocásticos disminuyan a medida que la optimización avanza hacia una posible solución estable. A diferencia de los métodos basados en sensibilidades, éstos tienen mayor facilidad para alcanzar soluciones globales, pero a costa de requerir muchas más iteraciones, lo que se traduce en un mayor esfuerzo computacional.

Los modelos más exitosos basados en sensibilidades son el modelo SIMP (*Solid isotropic Material with Penalization*) y el ESO/BESO (*Bi-directional Evolutionary Structural Optimization*). La principal diferencia radica en como tratan la variable de diseño, que en problemas estructurales equivale a la densidad de material en cada elemento de la discretización aplicada. El primero trabaja con un gradiente continuo de densidades, a medida que avanza el proceso iterativo, se penalizan las densidades intermedias para ir poco a poco definiendo las fronteras entre lo que será la pieza final y el vacío (ausencia de material). El segundo modelo trabaja con densidad discreta (sólido o vacío), se basa en un enfoque evolutivo, en donde se añade o retira material según una tasa evolutiva, que representa el máximo cambio de material permitido por iteración. De esta manera, el conjunto evoluciona paulatinamente hasta alcanzar un reparto de material óptimo. En ambos modelos es esencial afinar tanto la penalización como la tasa evolutiva para obtener resultados aceptables en un número de iteraciones razonables.

En problemas de optimización con un gran número de variables de diseño pueden existir múltiples configuraciones que cumplan de forma similar la funcionalidad y requisitos impuestos. Otras características de diseño no consideradas en la función objetivo pueden ser relevantes a la hora de diferenciar resultados entre sí. En la optimización topológica de mecanismos compliant es común el efecto bisagra, que consiste en la acumulación de grandes deformaciones en zonas muy reducidas. Un efecto pronunciado puede conducir a problemas de fatiga o fallo estructural. Para mejorar la vida útil del mecanismo, es recomendable generar diseños que eviten grandes deformaciones locales, esto se logra suavizando la transición entre espesores de material. Aquellos diseños que cumplen este principio se conocen como mecanismos compliant distribuidos (Figura 1).

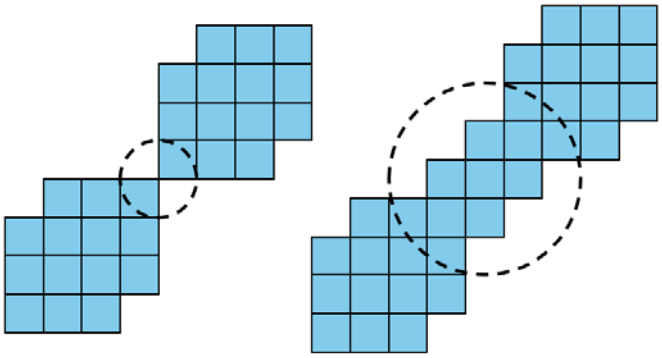

Figura 1: Comparación entre zona con alto efecto bisagra y zona con bajo efecto bisagra.

Por lo tanto, en este trabajo se presenta una metodología para la generación de mecanismos compliant que minimiza el efecto bisagra, sin perjudicar la funcionalidad del mecanismo generado. La estrategia desarrollada emplea la función objetivo y sus sensibilidades para alcanzar una distribución de material óptima, aplicando correcciones en el diseño en cada iteración con técnicas inspiradas en convolución y tratamiento de imagen, potenciando así la obtención de soluciones de compliant distribuidos.

2. Estado del arte

Desde su introducción a finales del siglo XX, la optimización topológica ha acaparado la atención de multitud de científicos e investigadores que han desarrollado diferentes enfoques para la resolución de problemas de optimización. El problema genérico de optimización topológica se trata de obtener la distribución de la variable de diseño que minimiza (o maximiza) la función objetivo que mejor representa al problema, mientras se imponen ciertas restricciones de diseño. A partir de esta estructura básica se han desarrollado diferentes enfoques matemáticos que tratan de resolver eficazmente el problema expuesto para el mayor número de casuísticas posibles.

Como se expone en [1], los enfoques más conocidos/utilizados son: por densidad, evolutivos, derivados topológicos, *level set*, *phase field*, lagrangianos. Los autores hacen una revisión de las características principales de cada enfoque y concluye con una lista de problemáticas recurrentes en el campo de la optimización topológica. Destacan como la comunidad que trabaja en este campo se divide y genera vías de investigación en diferentes direcciones. No existe una corriente clara de trabajo óptima, la tendencia es plantear estrategias alternativas de lo ya existente sin pararse a revisar los resultados y métodos que mejor funcionan. También plantean varios desafíos a los que la comunidad debería de enfrentarse para mejorar el estado de la optimización topológica en su conjunto.

Este artículo parte de ciertas publicaciones que sirven como iniciación al mundo de la optimización topológica y su aplicación a los mecanismos compliant. La referencia [2] ilustra cómo funciona un código de optimización en Python enlazado con el solver de elementos finitos ABAQUS. Las referencias [3] [4] [5] exponen modelos de optimización topológica ESO/BESO adaptados para resolver mecanismos compliant. Dejan patente las características particulares de estos modelos, la resolución de 2 casos combinados (in y out) que representan al compliant, la estrategia evolutiva de material, el uso de muelles artificiales para mejorar la convergencia de las soluciones, como se computa la función objetivo, las sensibilidades y su filtrado. Incluyen varios resultados replicables, que son usados como referencia, para la evaluación del código desarrollado en este artículo.

Un aspecto distintivo de la optimización enfocada a mecanismos compliant es la aplicación de muelles artificiales en los casos de simulación [6]. Estos muelles aplicados en las zonas de entrada y salida del mecanismo ayudan a afianzar una distribución de material más uniforme, mitigando el efecto bisagra numérico, es decir, que 2 elementos sólidos estén conectados por un solo nodo. La referencia [6] también expone que los mecanismos compliant consisten en distribuciones de material poco densas, por lo que es más coherente iniciar un proceso de optimización desde el estado vacío que desde el estado completamente sólido. Este simple cambio reduce considerablemente el tiempo de ejecución de una simulación, al alcanzar en menos iteraciones la restricción de volumen considerada.

El estado del arte deja patente la importancia de mitigar el efecto bisagra extremo en los diseños de mecanismos compliant. Este efecto es complicado de evaluar, ya que un efecto bisagra muy acusado permite un mayor rango de movimiento del mecanismo, pero también lo hace más propenso al fallo estructural. Como contrapartida, eliminar el efecto bisagra refuerza el compliant, pero reduce enormemente su capacidad de movimiento. Por lo tanto, controlar este efecto para la realización de diseños funcionales es crucial. En [7] sustituyen en post proceso zonas de elementos unidos por un único nodo por una bisagra compliant previamente diseñada para minimizar tensiones excesivas. En [8] se implementa una restricción de volumen adaptativo para evitar diseños con bisagras muy predominantes y potenciar resultados distribuidos. No hay una solución única y mejor a esta casuística, es un problema inherente a los compliant que está muy presente en multitud de resultados de artículos del estado del arte.

3. Metodología

Con el objetivo de desarrollar una herramienta de optimización topológica robusta y accesible, capaz de resolver mecanismos compliant, se opta por programar una herramienta en Python enlazada con el programa ABAQUS, que se usa como solver de elementos finitos. Las razones principales son el bagaje previo con este lenguaje y su compatibilidad con ABAQUS, la potencia de cálculo de ABAQUS y el aprovechamiento de su entorno gráfico de pre y post procesamiento. Esta situación ha permitido crear una herramienta potente y versátil, con capacidad para resolver sistemas tridimensionales con suficiente detalle, reduciendo al mínimo el conocimiento necesario de un posible usuario de la herramienta sobre optimización topológica y las variables que entran en juego, necesitando exclusivamente capacidades básicas de uso de programas de elementos finitos. En este apartado se exponen las

técnicas que se han aplicado durante el desarrollo de la herramienta para obtener mejores resultados relativos a mecanismos compliant distribuidos.

La referencia [2] se utiliza como entorno de partida para el código desarrollado mientras que la referencia [4] sirve como modelo de optimización base a partir del cual iterar y mejorar hasta conseguir los resultados propuestos. Una vez programado el modelo y comprobado que el bucle de optimización funciona correctamente en el entorno Python/ABAQUS, se desarrollan mejoras que permiten obtener resultados de mecanismos compliant distribuidos, con el mínimo efecto bisagra posible y en un rango de volumen dado. El problema de optimización resuelto se indica en (1) [4], donde se recoge la función objetivo a maximizar, la restricción de volumen y las ecuaciones de equilibrio de los dos casos de carga (Figura 2). En la ecuación, p es la carga aplicada, u el campo del desplazamiento, k_s la rigidez del muelle ficticio aplicado en la entrada y salida del mecanismo y K la matriz global de rigidez del sistema. Los índices 1 y 2 corresponden con los casos de carga de entrada y salida del mecanismo, respectivamente. En esencia, la función objetivo utilizada es un desarrollo de lo que se conoce como ventaja mecánica, la relación entre la fuerza a la salida frente a la entrada del mecanismo.

$$\text{Maximizar } \frac{1}{p_1}\frac{u_{21}}{\frac{u_{11}}{p_1}+\frac{u_{22}}{p_2}+\frac{1}{k_s}} \qquad (1)$$

$$\text{Sujeto a } \begin{cases} V \leq V^{max} \\ \boldsymbol{Ku_1} = \boldsymbol{f_1} \\ \boldsymbol{Ku_2} = \boldsymbol{f_2} \end{cases}$$

$$\text{y siendo } u_{ij} = \boldsymbol{u_j^T K u_i}/p_i$$

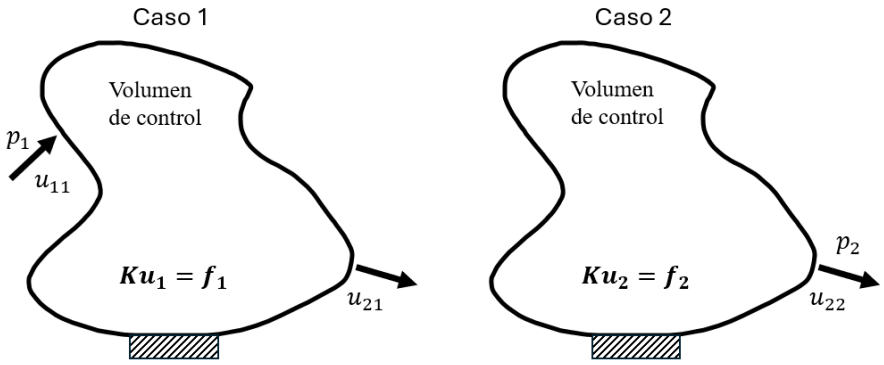

Figura 2: Casos de carga a resolver en cada iteración de la optimización.

Una de las primeras mejoras fue la de adaptar el código de carácter ESO a BESO. La diferencia entre ambos enfoques es que el primero solo considera el hecho de añadir (o retirar) material de forma paulatina al volumen de trabajo, mientras que el segundo enfoque permite al mismo tiempo tanto retirar como introducir material nuevo, lo que lo hace más versátil a la hora de encontrar una solución óptima.

Para los métodos de optimización evolutivos, uno de los parámetros fundamentales es la tasa evolutiva, que decide cuánto material se permite cambiar de estado (sólido a vacío o viceversa) en cada iteración. Se recomienda un 2% máximo de variación de material por iteración para asegurar una buena convergencia de la solución. Dado que la tasa de cambio de material no es tan crítica al principio de la optimización como al final, algunos artículos aplican tasas evolutivas variables de diversas índoles [9]. En este trabajo se implementa una tasa evolutiva variable que empieza en un valor máximo establecido y disminuye linealmente hasta alcanzar el valor mínimo permitido cuando se alcanza el volumen objetivo. Esto consigue que la optimización se acerque a la solución en menos iteraciones, sin sacrificar la precisión necesaria en los estados finales de la optimización. Como añadido, se escala el efecto de la tasa evolutiva con el volumen objetivo, para que optimizaciones con un menor volumen objetivo dispongan de la misma cantidad de iteraciones para alcanzar los requisitos de volumen que otra con un mayor volumen objetivo. Esto refuerza aún más la búsqueda paulatina de la mejor distribución de material.

Se aplica una hibridación al modelo para mejorar la convergencia de las soluciones. El método BESO se basa en dos estados discretos (sólido o vacío) mientras que el método SIMP es continuo, permitiendo densidades intermedias entre ambos estados. Este hecho ayuda a que la convergencia de la solución sea más controlada. Se ajusta el código para que se permitan ciertas densidades intermedias, aportando un efecto pseudo continuo a la variable de diseño durante la optimización. En las primeras iteraciones predominan las densidades intermedias, facilitando la búsqueda de la forma general de la solución. En las iteraciones más avanzadas predominan las densidades extremas, afianzando así los bordes de la solución con el vacío.

Para el algoritmo de distribución de material se ha optado por crear uno basado en el orden de las sensibilidades y en la función logística. Se ha observado que, para aproximar una distribución de material a una solución válida, no es necesario tener en cuenta la magnitud de la sensibilidad del elemento, basta con tener en cuenta el orden de los elementos según su sensibilidad. De esta manera se diferencia en cada iteración entre los elementos que deben ser sólidos de los que deben ser vacíos en base al ranking de elementos por sensibilidad. A los elementos que tienen que ser vacíos se les asigna una densidad artificial mínima que simbolice el cero numérico, se aplica por defecto 10^{-3}. A los elementos que deben ser sólidos, se le asigna una densidad artificial discreta dentro del rango $[10^{-2}, 1]$. Al disponer del orden de importancia de los elementos más sensibles a la función objetivo, se impone una mayor densidad a los elementos con mayor sensibilidad. Este reparto de densidades sólidas intermedias se hace en base a la función logística. En la Figura 3 se incluye un esquema del reparto de material y en (2) la ecuación genérica de una función logística en donde x es el rango de valores en donde se evalúa la función.

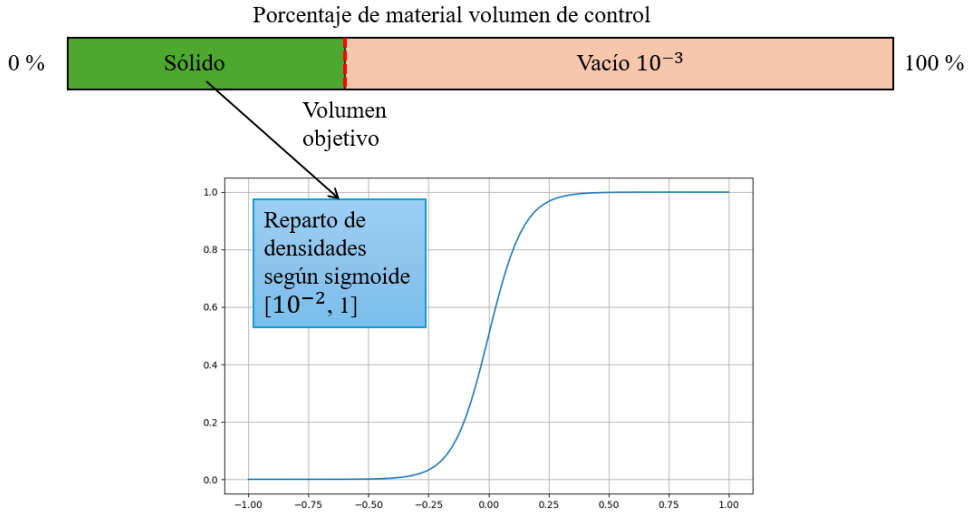

Figura 3: Esquema reparto de material.

$$\frac{1}{1+e^{-x}} \tag{2}$$

La función logística consiste en una función matemática monótonamente creciente con una forma característica de S. La transición entre los dos valores extremos de la función puede ser más o menos abrupta. Se adapta este comportamiento a las posibles densidades sólidas que se manejan en el código, es decir, los valores de la función variarán desde 10^{-2} hasta 1. La transición de la curva es dependiente de la relación entre la iteración actual y el límite preestablecido de iteraciones, de esta manera se consigue un reparto de densidades intermedias para los elementos sólidos en las primeras iteraciones y una tendencia hacia las densidades sólidas extremas consideradas (10^{-2} o 1) en las iteraciones finales. En la Figura 4 se muestra un ejemplo de cómo varía la función logística con un límite de 100 iteraciones, donde los elementos considerados sólidos según su sensibilidad se mapean a lo largo de esta curva, obteniendo valores proporcionales dentro del rango $[10^{-2}, 1]$.

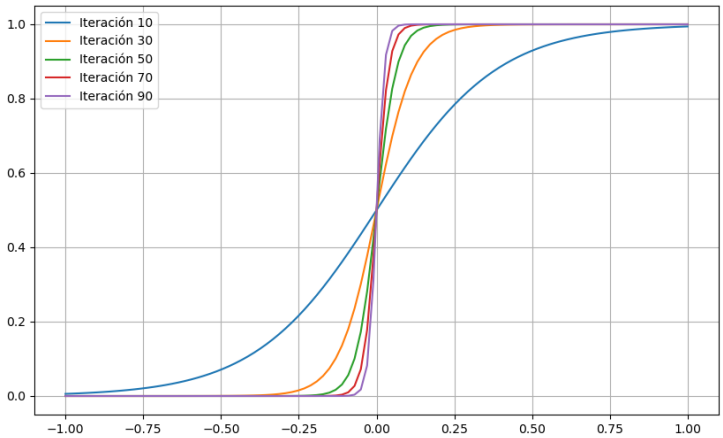

Figura 4: Variación de la función logística, con 100 iteraciones como límite.

Este algoritmo tiende a 3 estados de densidad, vacío (densidad 10^{-3}), material sólido blando (densidad 10^{-2}) y material sólido completo (densidad 1). Si se utiliza exclusivamente la variable volumen objetivo para indicar material sólido, por ejemplo, un 20% del volumen total, en todos los casos la optimización terminaría con un 10% de material sólido completo, 10% de material sólido blando y el 80% restante vacío. Por lo tanto, se divide la variable volumen objetivo en dos, mínimo y máximo, para controlar también los porcentajes de sólido completo y sólido blando en la simulación. En la práctica, esto se consigue jugando con la posición relativa del centro de la función logística respecto a uno de sus extremos, como se puede observar en la Figura 5.

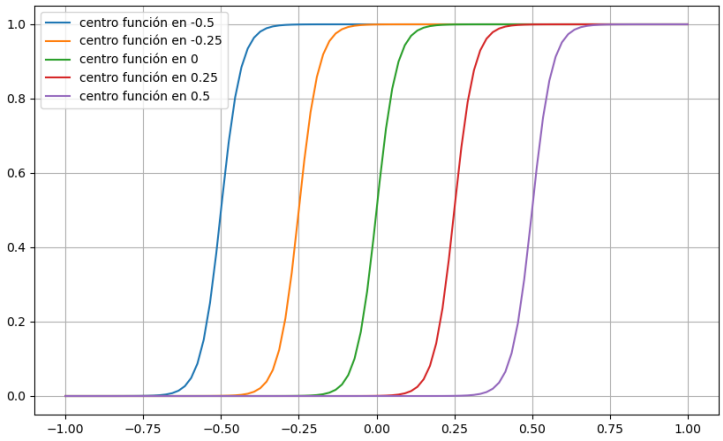

Figura 5: Variación de la función logística con diferentes posiciones para una misma iteración.

De esta manera se establece una configuración en la que, mediante dos parámetros de volumen objetivo (mínimo y máximo), se formaliza una distribución de tres densidades: una estructura sólida completa que equivale al volumen mínimo requerido, una zona de material sólido blando alrededor de esta estructura indicando las áreas potenciales de crecimiento de la pieza, y el resto vacío. En la Figura 6 se ilustra un ejemplo donde se aprecian las 3 fases mencionadas, material sólido completo en gris, material blando en verde, algún elemento con diferentes colores indicando densidades intermedias correspondientes a la transición de la función logística y el vacío oculto, para resaltar las fronteras de la pieza resultante.

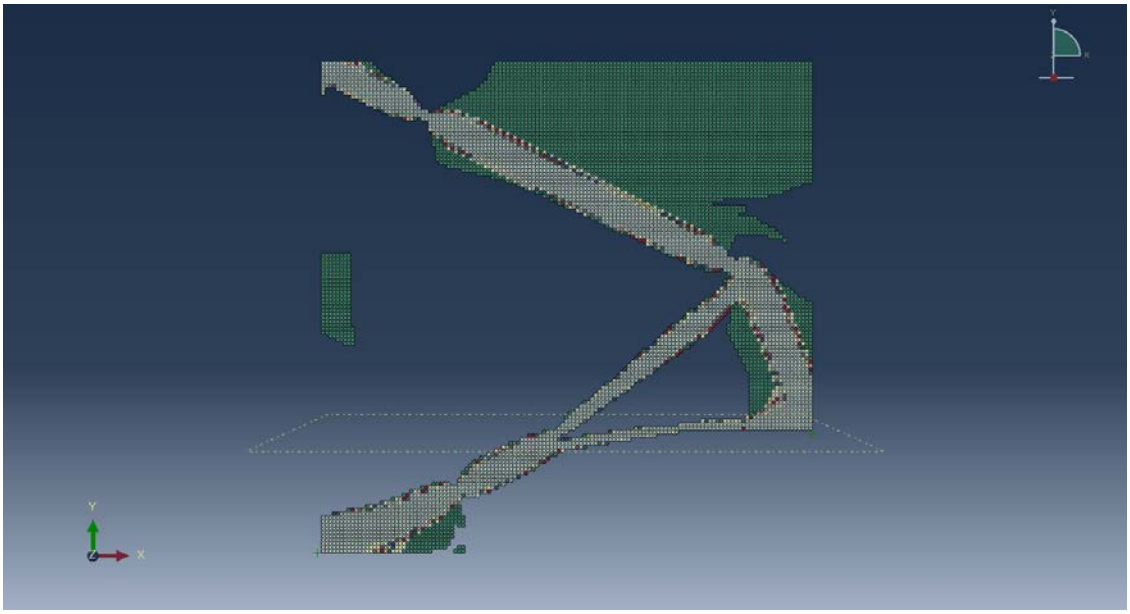

Figura 6: Ejemplo de distribución de material en una iteración concreta con el algoritmo desarrollado.

Inspirándose en la restricción de conectividad de la referencia [10], se crea un control de la continuidad de los elementos sólidos que solo se aplica cuando se alcanza la convergencia. La idea es evitar que el resultado final contenga islas de material sólido desconectadas de todas las condiciones de contorno. Esto se consigue realizando un registro de los elementos sólidos adyacentes a las condiciones de contorno y a su vez los elementos sólidos adyacentes a estos, así hasta hacer un barrido de toda la malla. Los elementos sólidos no registrados se transforman en vacío al no estar conectados a ninguna condición de contorno. Todos los elementos registrados se pasan a sólido

completo para evitar cualquier elemento con densidad intermedia. Este procedimiento asegura que los resultados finales sean una única pieza continua.

Aunque con la configuración descrita se consiguen resultados aceptables, es más interesante aplicar algún tipo de algoritmo de decisión durante el proceso de optimización que aproveche la distribución del material blando alrededor del sólido completo. Como se observa en la Figura 6, el material blando se adhiere al sólido completo con coherencia hasta cierto punto. Existe un límite en donde más material no aporta al desempeño del mecanismo y por lo tanto la optimización lo distribuye en zonas sin interés mecánico, como el gran cúmulo de material en la esquina superior derecha, o directamente en islas de material rodeado de vacío.

Como se comenta en [8], cuando se utiliza una restricción de volumen fija, es difícil acertar de antemano cual será el porcentaje de volumen que obtendrá un mejor resultado. Por norma general, una manera de relajar las restricciones de diseño es incluyéndolas en la función objetivo para que su efecto influya en la optimización. Sin embargo, esta no es una opción viable para la restricción de volumen ya que es necesario tener un control directo sobre la misma en cada iteración, al ser la variable que define el avance de la optimización. Para la herramienta desarrollada, se ha decidido aplicar un enfoque más práctico, en donde siempre que el usuario escoja un volumen objetivo coherente, el programa sea capaz de generar una pieza funcional, cercana al volumen requerido, pero sin estar el resultado limitado por este, priorizando que la distribución de material no muestre un efecto bisagra muy agudo. Por lo tanto, inspirándose en el concepto de convolución en tratamiento de imagen, se aplica un filtro de suavizado a la densidad de cada elemento excepto a los que ya han alcanzado densidad sólida completa, para mantener siempre el volumen mínimo requerido en la optimización y potenciar la distribución que los elementos sólidos están tratando de alcanzar.

El kernel del suavizado, de igual manera que el kernel para el filtro de sensibilidades, tiene en cuenta cuantos y cuáles son los elementos adyacentes de cada elemento analizado. Se ha trabajado con un radio de filtro de sensibilidades de 1.25 y con un radio de filtro de densidades de 1.5 veces la longitud característica del elemento, lo que se traduce en considerar que los elementos adyacentes son todos los que tienen una arista/cara en contacto, o todos los que rodean al principal, respectivamente. El kernel del filtro de sensibilidades tiene en cuenta la distancia de los centros de los elementos, por lo que da un mayor peso al elemento central y menos a los más alejados. El filtro de suavizado de densidades da un mismo peso a todos los elementos por igual, por lo que, si un elemento tiene densidad intermedia o vacía, pero está rodeado de material sólido completo, el elemento aumentará su densidad para que tenga un mayor efecto en la siguiente iteración. Por el contrario, elementos de baja densidad rodeados de vacío disminuirán su densidad, hasta que acaben siendo vacío. En la Figura 7 se muestra un ejemplo de cómo sería aplicar el filtro de suavizado a un elemento bidimensional, se aprecia que la tendencia de este efecto es a redondear (suavizar) posiciones comprometidas de grandes diferencias de densidad, mejorando a la larga el resultado final de la optimización.

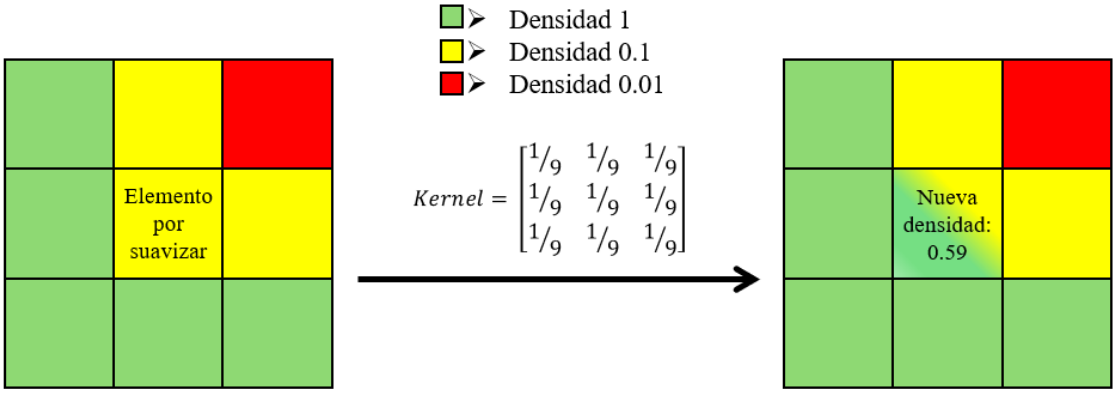

Figura 7: Ejemplo ilustrativo del suavizado de la densidad de un elemento en un mallado 2D.

El suavizado se aplica posterior al reparto de densidades por sensibilidades, siempre y cuando se haya alcanzado el volumen objetivo mínimo de material y se haya superado la mitad de las iteraciones máximas. Estas dos condiciones limitan la aplicación del filtro de suavizado a las fases finales de la optimización, cuando la forma general del mecanismo está definida y los cambios de densidades en cada iteración ocurren en la frontera de la pieza generada, que es justo la zona que interesa suavizar para mejorar el diseño final. Este procedimiento, junto con todas las mejoras implementadas previamente, facilitan la obtención de mecanismos compliant con un efecto bisagra controlado, sin que la condición de volumen limite en exceso la libertad a la hora de distribuir el material de forma óptima.

4. Resultados

En este apartado se muestran los resultados que produce la herramienta para alguno de los ejemplos más estudiados en la bibliografía. En todos los casos se alcanza una solución estable en el rango de volumen elegido, con una buena convergencia por debajo de las 100 iteraciones. Todos los ejemplos poseen una malla muy fina, de decenas de miles de elementos. Esto se hace por dos razones, por una parte, para mostrar la capacidad de cálculo de la herramienta, que es capaz de abordar grandes mallas en tiempos razonables y simulaciones tridimensionales con gran detalle. Por otra parte, una malla más fina permite apreciar mejor el efecto de suavizado que tiene la metodología desarrollada en los resultados. Las especificaciones del equipo para la obtención de los resultados son: procesador Intel i7-7700 de 3.6 GHz, 4 núcleos, 4 procesadores lógicos y 32 GB de RAM.

Para uno de los casos resueltos, se incluye una comparación entre el resultado obtenido con y sin las mejoras desarrolladas aplicadas en la optimización. En concreto, las mejoras omitidas son la tasa evolutiva variable, la fracción de volumen mínima y máxima, la hibridación del modelo SIMP y BESO, la aplicación de la función logística a las sensibilidades y el filtro de suavizado.

En las gráficas de volumen de cada caso (Figura 9, Figura 11 y Figura 13 izquierda) se muestran los límites mínimo y máximo del volumen objetivo y dos curvas que corresponden con la evolución del volumen de la pieza. La fracción volumétrica por densidad tiene en cuenta la densidad de cada elemento mientras que la fracción volumétrica completa asume que todos los elementos son sólidos excepto los vacíos. En dichas gráficas se pueden apreciar las diferentes etapas de la optimización, como en las primeras iteraciones se produce un rápido crecimiento de material para la pieza. Posteriormente la distribución de densidad de material se estabiliza en los límites de volumen establecidos, sólido completo hasta volumen objetivo mínimo y el resto de las densidades intermedias hasta volumen objetivo máximo. Finalmente, tras cumplir los criterios para el suavizado, se aplica dicha técnica para mejorar el diseño en las iteraciones finales y así obtener resultados de mecanismos compliant con el menor efecto bisagra posible. En la última iteración se aplica el control de continuidad y todos los elementos con densidad intermedia se transforman a sólido completo para terminar de definir la pieza resultante.

En la Figura 8 y Figura 9 se muestra el ejemplo conocido como *inverter mechanism* [3] [5] [6], este mecanismo invierte el movimiento: un desplazamiento negativo en el eje X en la entrada (flecha input) implica un desplazamiento positivo en el eje X en la salida (flecha output). Se imponen los límites mínimo y máximo de volumen de 0.1 y 0.3, respectivamente, respecto al volumen de partida. El mecanismo se fija en las zonas de amarre rojas. Se ha utilizado una malla de 20,000 elementos con simetría en el plano XZ, 40,000 elementos considerando el conjunto completo. La solución converge a las 66 iteraciones, después de 40 minutos de optimización. En la gráfica de la función objetivo se aprecia el efecto que tiene la aplicación del filtro de suavizado (iteración 50), especialmente en una situación en donde la función objetivo oscila en los tramos finales de la optimización, una vez alcanzados los límites del volumen objetivo.

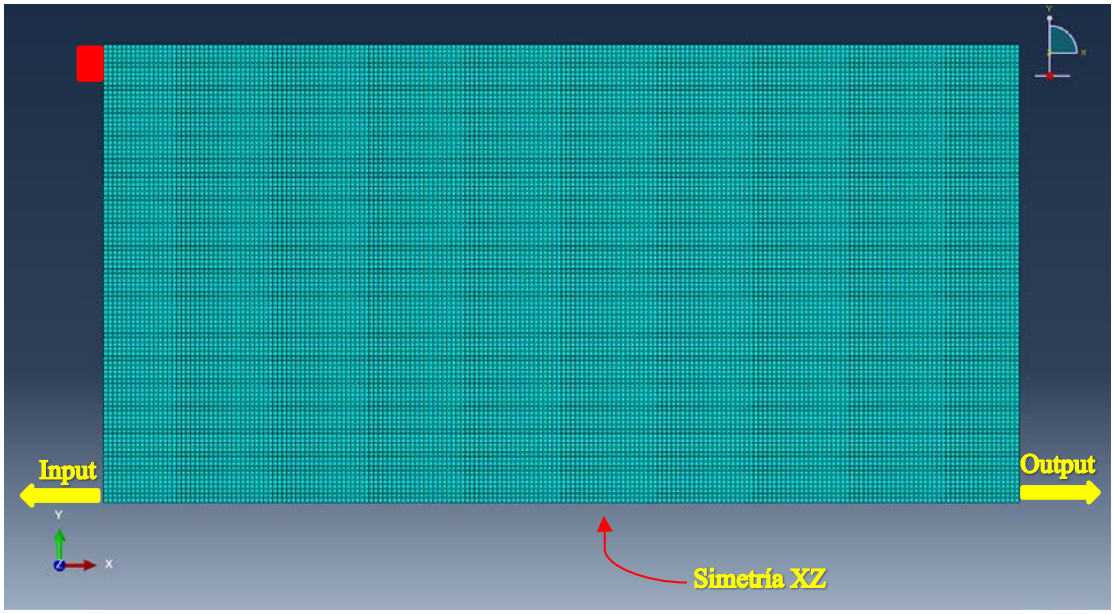

Figura 8: Volumen de partida para el *Inverter mechanism*.

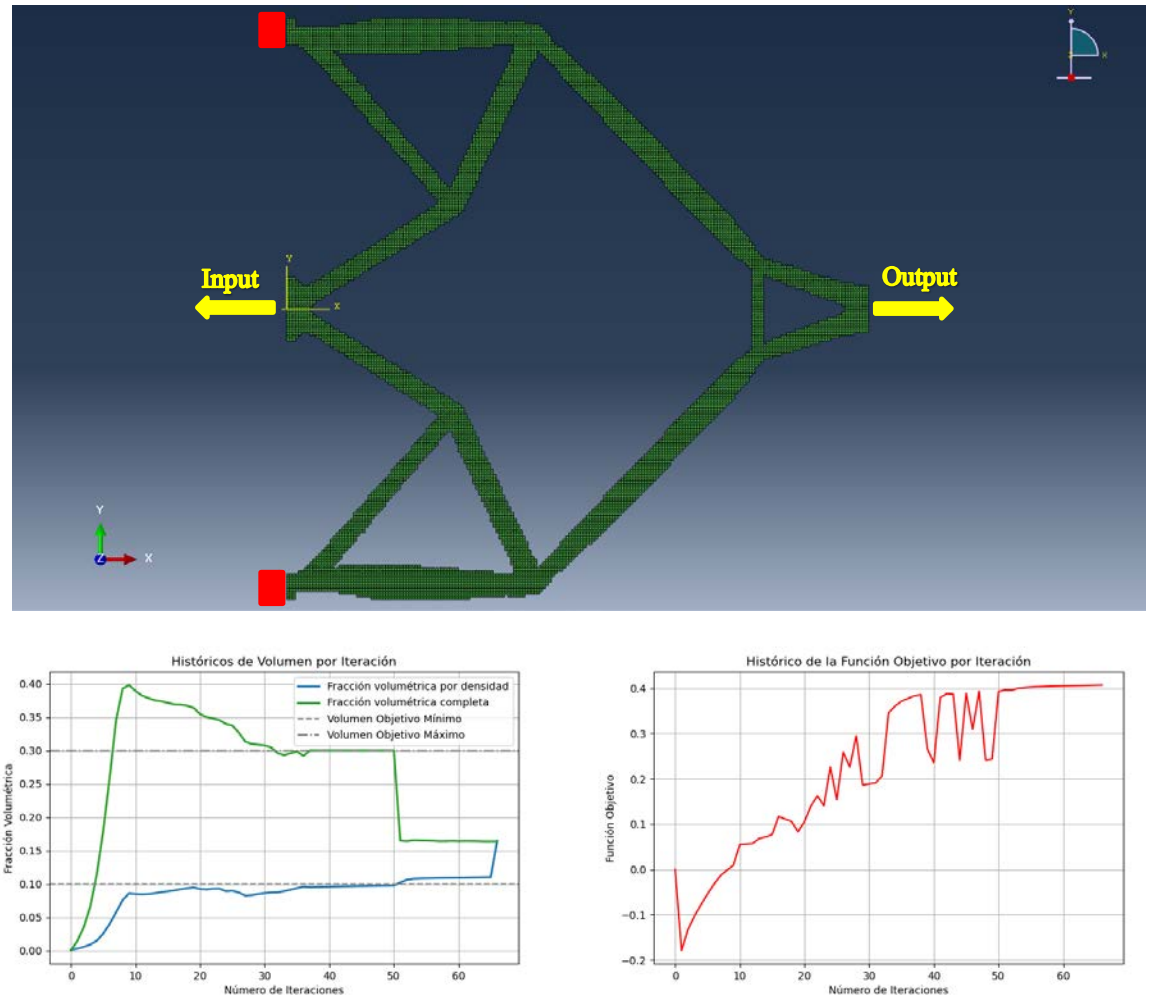

Figura 9: Resultado *Inverter mechanism* (arriba), fracción volumétrica (izquierda), función objetivo (derecha).

En la Figura 10 y Figura 11 se muestra el ejemplo conocido como *Elevation mechanism* [4] [5], una pieza que transforma un movimiento de separación de sus patas en el eje X en un desplazamiento en el eje Z. Se imponen los límites mínimo y máximo de volumen de 0.05 y 0.15, respectivamente, respecto al volumen de partida. El mecanismo se fija en las zonas de amarre rojas. Se ha utilizado una malla de 75,000 elementos con simetría en el plano XZ e YZ, 300,000 elementos considerando el conjunto completo. La solución converge a las 63 iteraciones, después de 278 minutos de optimización. Este ejemplo sirve para ilustrar la capacidad de procesamiento que se consigue con la herramienta. Es factible optimizar problemas tridimensionales con una buena resolución de mallado, lo que permite aspirar a resolver casos más complejos que los habitualmente analizados en la bibliografía.

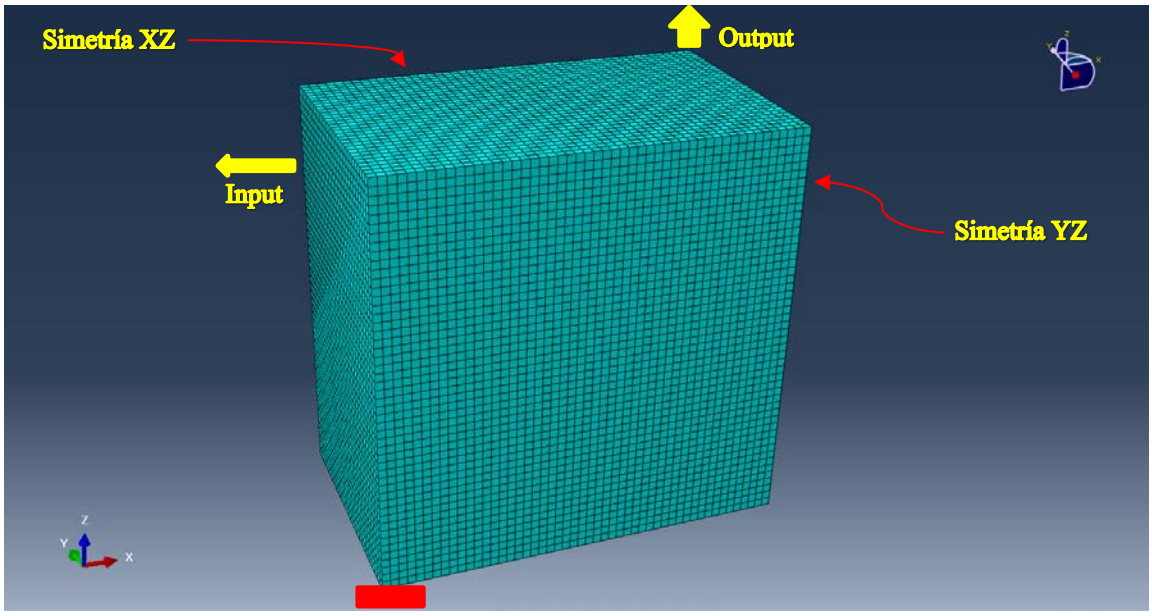

Figura 10: Volumen de partida para el *Elevation mechanism*.

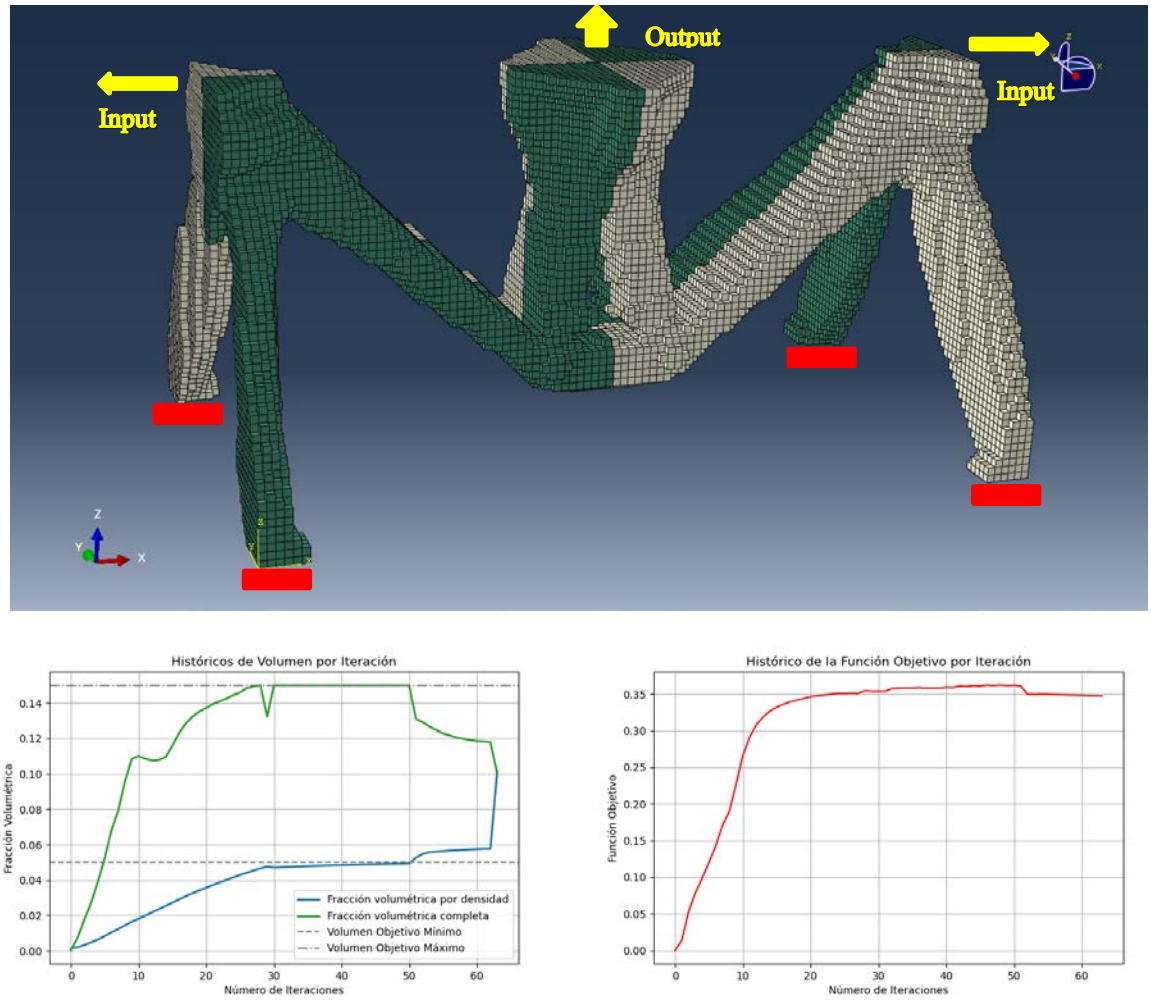

Figura 11: Resultado *Elevation mechanism* (arriba), fracción volumétrica (izquierda), función objetivo (derecha).

En la Figura 12 y Figura 13 se muestra el ejemplo conocido como *Crunching mechanism* [3] [6], una pieza que transforma un movimiento de apriete en el eje Y en un desplazamiento en el eje X. Se imponen los límites mínimo y máximo de volumen de 0.25 y 0.5, respectivamente, respecto al volumen de partida. El mecanismo se fija en las zonas de amarre rojas. Se ha utilizado una malla de 16,000 elementos con simetría en el plano XZ, 32,000

elementos considerando el conjunto completo. La solución converge a las 61 iteraciones, después de 33 minutos de optimización. Este caso es interesante porque dispone de 4 zonas bisagra notables, necesarias para el movimiento del mecanismo. Se observa como la estrategia de optimización desarrollada consigue que dichas zonas tengan el material lo más distribuido posible, con transiciones suaves entre las zonas esbeltas y el resto de material que conectan.

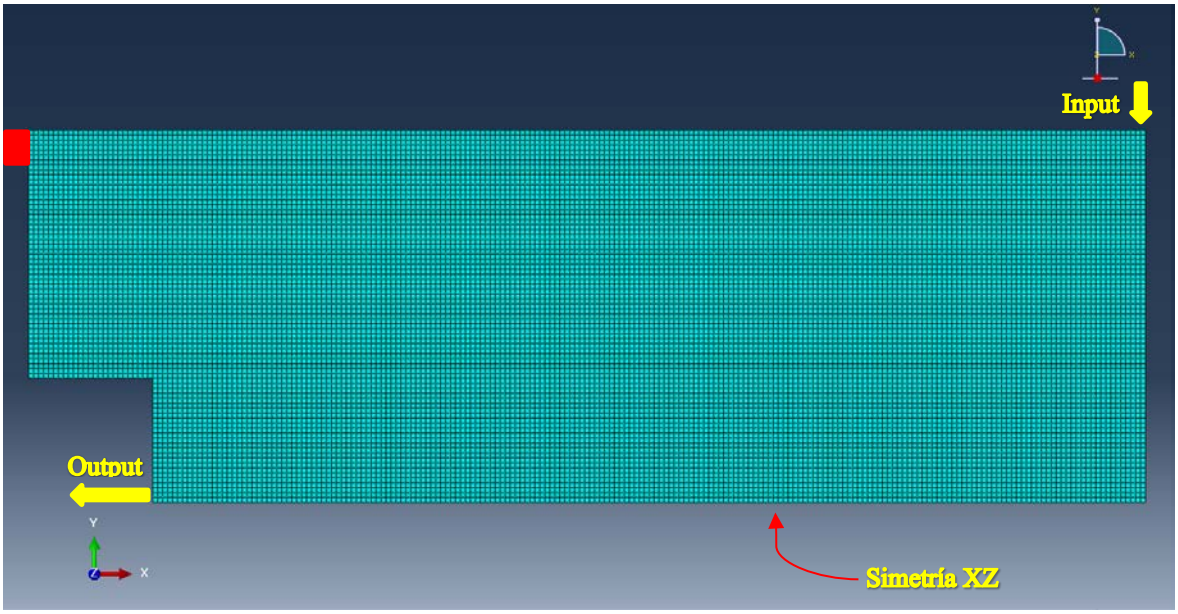

Figura 12: Volumen de partida para el *Crunching mechanism*.

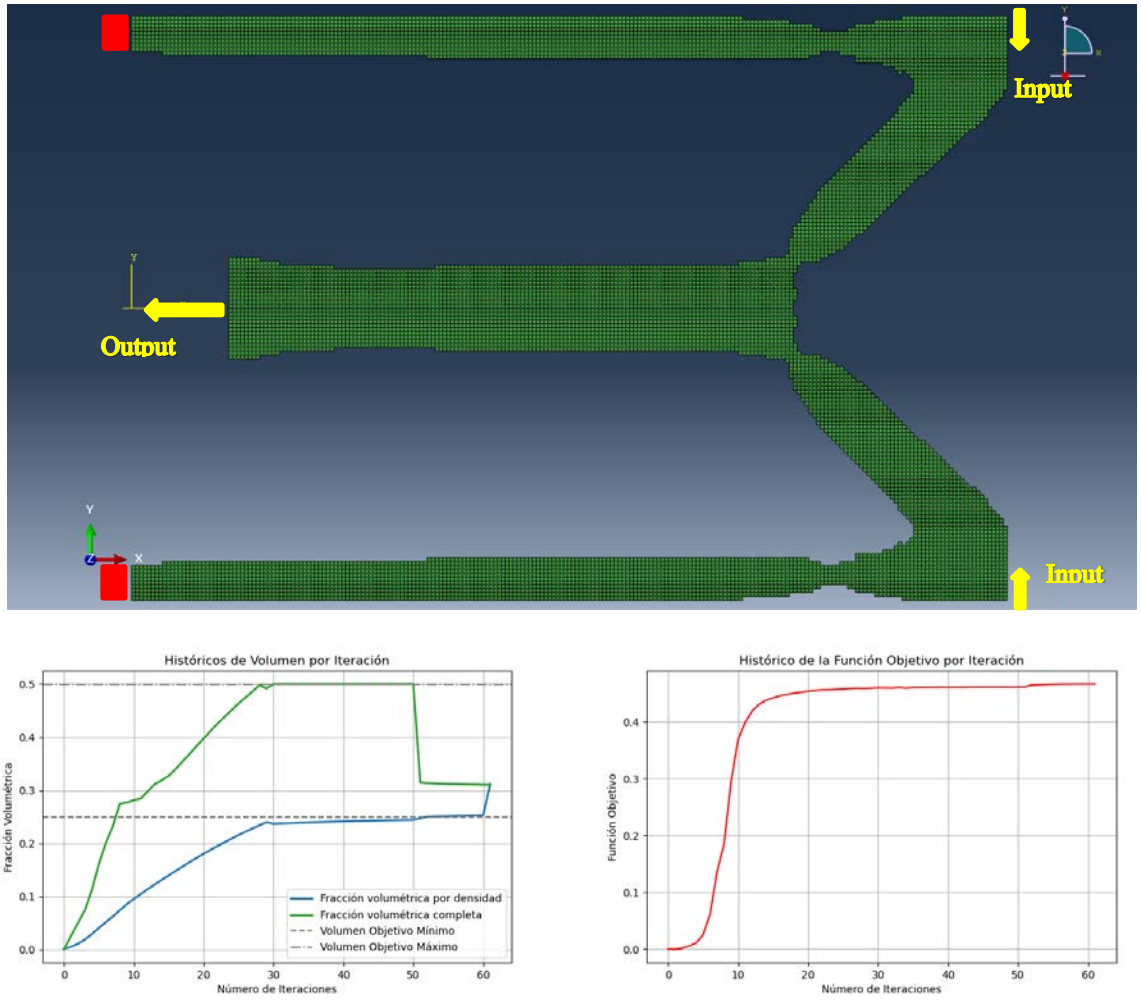

Figura 13: Resultado *Crunching mechanism* (arriba), fracción volumétrica (izquierda), función objetivo (derecha).

En la Figura 14 se muestra el resultado del *Crunching mechanism* si no se implementasen las mejoras desarrolladas en este trabajo, para un volumen objetivo de 0.3 veces el volumen de partida. Las zonas bisagra características de este mecanismo muestran una transición más abrupta y unos espesores inferiores, llegando a conectar zonas de material por un único elemento sólido. Un diseño de estas características tiende a concentrar tensiones y es más propenso a fallar por fatiga. En ambos casos, con y sin mejoras, la función objetivo tiende al mismo valor, indicando que las mejoras implementadas no empeoran el comportamiento mecánico del mecanismo generado.

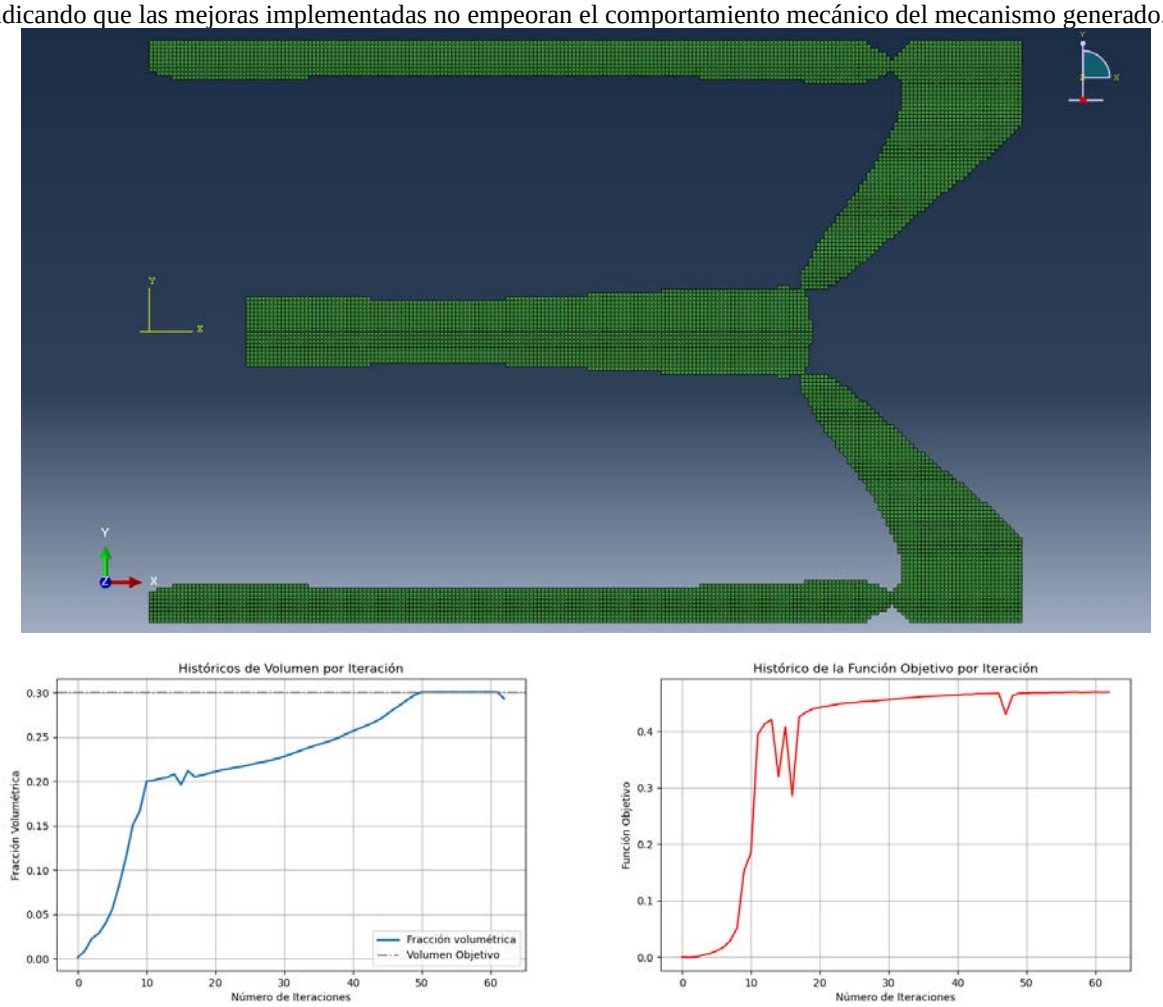

Figura 14: Resultado *Crunching mechanism* sin mejoras implementadas (arriba), fracción volumétrica (izquierda), función objetivo (derecha).

En la Figura 15 y Figura 16 se muestran las distribuciones de tensión del *Crunching mechanism* con y sin las mejoras desarrolladas aplicadas, respectivamente. La escala está en MPa, la pieza tiene unas dimensiones generales de 90 mm de largo por 60 mm de alto, con una profundidad de 1 mm (sólido simulado como estructura tipo shell). Las mallas obtenidas en la optimización son refinadas con mayor detalle para que las zonas bisagra tengan suficientes elementos representativos y no falseen la distribución de tensiones. En ambos casos se aplica la misma carga de apriete en las esquinas superior e inferior derecha de la pieza para obtener el movimiento en el eje X del brazo central de la pieza. Los resultados muestran que minimizar el efecto bisagra reduce los picos máximos de tensión. Para este caso en concreto, la tensión máxima en el diseño mejorado es 3.1 veces menor que la del diseño sin mejoras aplicadas.

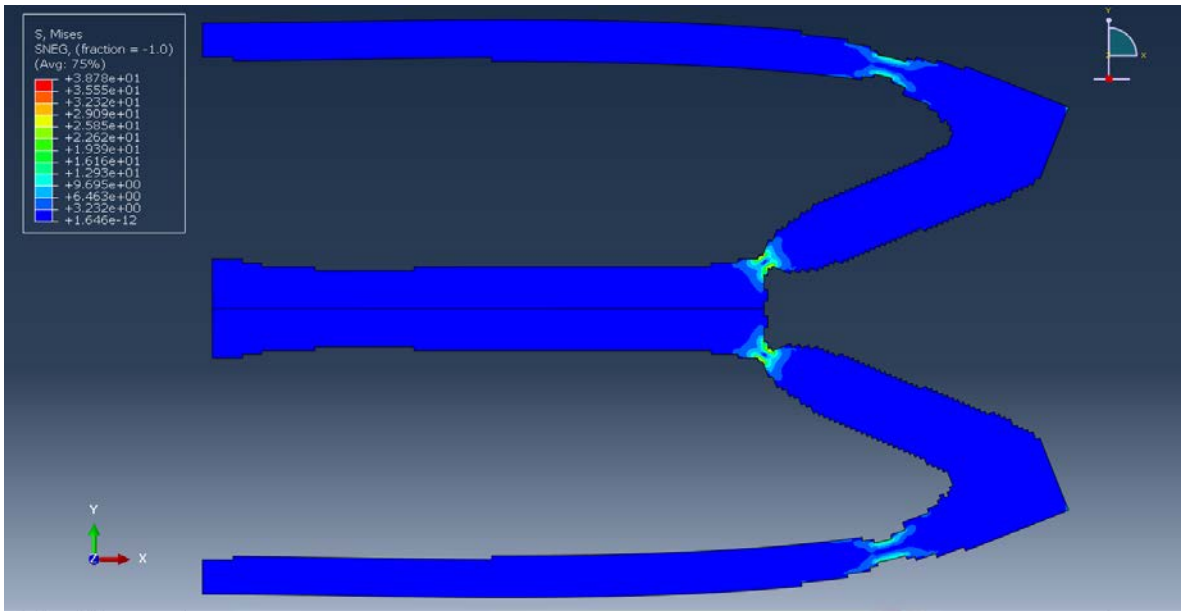

Figura 15: Distribución de la tensión para el *Crunching mechanism*, resultado con mejoras implementadas.

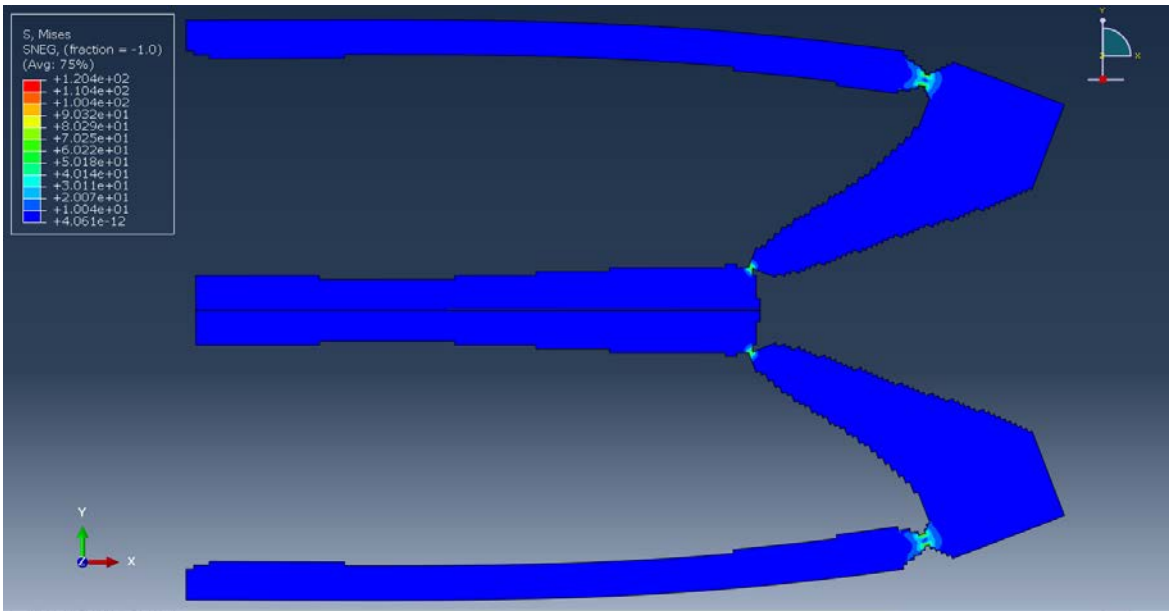

Figura 16: Distribución de la tensión para el *Crunching mechanism*, resultado sin mejoras implementadas.

5. Conclusiones y líneas futuras

Se ha diseñado e implementado una herramienta Python-ABAQUS de optimización topológica orientada al diseño de mecanismos compliant. Este entorno de trabajo facilita la construcción de casos de optimización más elaborados (volúmenes de control complejos, mallados con alto detalle), que los vistos en el estado del arte, además de la posibilidad de exportar el resultado para su post proceso en CAD o realizar simulaciones mecánicas directamente sobre los diseños obtenidos para su validación.

Se ha desarrollado una metodología que potencia las soluciones compliant distribuidas, evitando un efecto bisagra acentuado, dentro de un rango de volumen objetivo dado. Estas soluciones distribuidas mantienen el rendimiento mecánico esperado del mecanismo y reducen la tensión en las zonas bisagra de los diseños generados. La herramienta muestra una muy buena convergencia de los resultados, necesitando menos de 100 iteraciones para alcanzar diseños aceptables.

Este trabajo ha abordado algunos de los desafíos que se plantean en [1]. Se ha demostrado la eficiencia y potencia de la herramienta, capaz de manejar mallados de decenas de miles de elementos en tiempos de cálculo razonables, permitiendo resolver sistemas tridimensionales con mucho detalle. Sobre los parámetros de ajuste y a la facilidad de uso, la herramienta se ha creado de tal forma que la participación del usuario sea mínima. Los parámetros generales de optimización han sido previamente calibrados, un usuario experimentado puede jugar con ellos o dejar por defecto los que mejor han funcionado en la calibración realizada.

Como líneas futuras de trabajo, una vez probada la eficacia de la herramienta, se plantea, por un lado, generar nuevos resultados, especialmente casos tridimensionales y de aplicaciones concretas reales. Por otro lado, se busca mejorar el algoritmo de decisión de material para encontrar un mejor equilibrio entre relajar la restricción de volumen para encontrar la mejor cantidad de material sólido necesario sin que se pierda efectividad en la convergencia hacia una solución de compliant distribuido. Generalizar la herramienta para que sea capaz de resolver mecanismos compliant que combinen varios casos de carga, no solo una entrada y una salida.

6. Referencias

[1] Sigmund, Ole, and Kurt Maute. "Topology optimization approaches: A comparative review." *Structural and multidisciplinary optimization* **48.6**: 1031-1055 (2013)

[2] Zuo, Zhi Hao, and Yi Min Xie. "A simple and compact Python code for complex 3D topology optimization." *Advances in Engineering Software* **85**: 1-11 (2015)

[3] Ansola, Rubén, et al. "A simple evolutionary topology optimization procedure for compliant mechanism design." *Finite Elements in Analysis and Design* **44.1-2**: 53-62 (2007)

[4] Ansola, Rubén, et al. "3D compliant mechanisms synthesis by a finite element addition procedure." *Finite Elements in Analysis and Design* **46.9**: 760-769 (2010)

[5] Li, Yan, et al. "Evolutionary topology optimization of hinge-free compliant mechanisms." *International Journal of Mechanical Sciences* **86**: 69-75 (2014)

[6] Liu, Chih-Hsing, Guo-Feng Huang, and Ta-Lun Chen. "An evolutionary soft-add topology optimization method for synthesis of compliant mechanisms with maximum output displacement." *Journal of Mechanisms and Robotics* **9.5**: 054502 (2017)

[7] Pedersen, Claus BW, Thomas Buhl, and Ole Sigmund. "Topology synthesis of large-displacement compliant mechanisms." *International Journal for numerical methods in engineering* **50.12**: 2683-2705 (2001)

[8] Seltmann, Stephanie, and Alexander Hasse. "Topology optimization of compliant mechanisms with distributed compliance (hinge-free compliant mechanisms) by using stiffness and adaptive volume constraints instead of stress constraints." *Mechanism and Machine Theory* **180**: 105133 (2023)

[9] Lin, Haidong, et al. "An ANSYS APDL code for topology optimization of structures with multi-constraints using the BESO method with dynamic evolution rate (DER-BESO)." *Structural and Multidisciplinary Optimization* **62.4**: 2229-2254 (2020)

[10] Munk, David J., Gareth A. Vio, and Grant P. Steven. "A bi-directional evolutionary structural optimisation algorithm with an added connectivity constraint." *Finite Elements in Analysis and Design* **131**: 25-42 (2017)