



UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA  
Instituto Universitario de Sistemas Inteligentes  
y Aplicaciones Numéricas en la Ingeniería



# **Optimización del diseño con algoritmos genéticos y metamodelos en la minimización del peso en piezas obtenidas mediante fabricación aditiva**

---

Trabajo Fin de Máster

Autor:

**Rubén Paz Hernández**

Tutores:

**Gabriel Winter Althaus**

**Mario Monzón Verona**

Las Palmas de Gran Canaria, Noviembre 2012.

## Contenido

1. Introducción .....	5
1.1. Fabricación aditiva .....	5
1.2. Estructuras celulares .....	7
2. Objetivos .....	10
3. Estado del arte .....	12
4. Nuevas propuestas de optimización adaptadas al problema .....	21
4.1. Búsqueda del diseño óptimo .....	21
4.2. Metamodelo de aproximación .....	22
4.3. Bucle iterativo de refinamiento del metamodelo .....	23
4.4. Algoritmo para la generación de nuevos puntos en el diseño de experimentos y refinamiento del metamodelo de aproximación .....	24
5. Comparación entre metamodelos de aproximación .....	27
5.1. Método de la distancia inversa ponderada .....	30
5.2. Interpolación spline .....	31
5.3. Superficies de respuesta mediante ajuste por mínimos cuadrados con ecuaciones polinómicas .....	32
5.4. Interpolación del vecino más cercano .....	34
5.5. Interpolación lineal basada en triangulación de Delaunay .....	35
5.6. Estudio comparativo .....	37
5.6.1. Ejemplo de referencia .....	37
5.6.2. Resultados del estudio .....	41
6. Desarrollo del algoritmo genético de optimización .....	48
6.1. Algoritmos genéticos .....	48
6.2. Implementación del algoritmo genético .....	50
6.2.1. Problema de referencia .....	50
6.2.2. Pseudocódigo inicial .....	51
6.2.3. Ajuste de los parámetros del algoritmo genético .....	52
7. Implementación del programa. Pruebas y mejoras .....	59

7.1.	Programa inicial.....	59
7.1.1.	Resultados obtenidos .....	61
7.1.2.	Óptimos obtenidos con AG a partir del metamodelo .....	63
7.1.3.	Adición de nuevos puntos antes del AG para mejorar el metamodelo....	64
7.2.	Segundo programa (AG de aproximación a frontera) .....	70
7.2.1.	Resultados obtenidos para el problema con 4 variables de diseño .....	75
7.2.2.	Resultados obtenidos para el problema con 5 variables de diseño .....	79
7.2.3.	Resultados obtenidos para el problema con 3 variables de diseño (evaluaciones con elementos finitos) .....	83
7.2.4.	Resultados obtenidos para el problema con 4 variables de diseño (evaluaciones con elementos finitos) .....	87
7.2.5.	Reducción del número de puntos evaluados .....	88
7.3.	Tercer programa .....	91
7.3.1.	Resultados obtenidos para el problema con 4 variables de diseño .....	93
7.3.2.	Resultados obtenidos para el problema con 3 variables de diseño (evaluaciones con elementos finitos) .....	93
7.4.	Cuarto programa .....	96
7.4.1.	Resultados obtenidos para el problema con 4 variables de diseño.....	102
7.5.	Quinto programa.....	103
7.5.1.	Resultados obtenidos para el problema con 3 variables de diseño (evaluaciones con elementos finitos) .....	104
7.5.2.	Resultados obtenidos para el problema con 4 variables de diseño (evaluaciones con elementos finitos) .....	108
7.5.3.	Resultados obtenidos para el problema con 5 variables de diseño (evaluaciones con elementos finitos) .....	110
7.5.4.	Resultados obtenidos para otro problema de ejemplo (5 variables, evaluaciones con elementos finitos) .....	112
7.6.	Sexto programa.....	119
7.6.1.	Resultados obtenidos para el otro problema de ejemplo (5 variables, evaluaciones con elementos finitos) .....	121
7.6.2.	Resultados obtenidos para el problema de referencia (3 variables, evaluaciones con elementos finitos) .....	124

7.6.3	Resultados obtenidos para el problema de referencia (4 variables, evaluaciones con elementos finitos) .....	125
7.6.4.	Resultados obtenidos para el problema de referencia (5 variables, evaluaciones con elementos finitos) .....	127
8.	Metodología final .....	129
8.1.	Ventajas de la metodología .....	132
8.2.	Inconvenientes de la metodología .....	133
8.3.	Otras limitaciones.....	134
9.	Líneas futuras.....	135
9.1.	Automatización del flujo de trabajo entre motor de cálculo y motor inteligente 135	
9.2.	Metodología para problemas de optimización con mayor número de variables de diseño.....	136
9.3.	Búsqueda exhaustiva de las herramientas comerciales más apropiadas para el diseño y cálculo .....	142
9.4.	Estudio de las tipologías celulares más apropiadas .....	142
9.5.	Análisis de las herramientas de mallado más adaptables para la discretización de diferentes geometrías durante la optimización .....	142
9.6.	Análisis de las posibles restricciones asociadas al proceso de fabricación ....	143
10.	Referencias.....	144

## 1. Introducción

La aparición de nuevas tecnologías de producción como la fabricación aditiva ha modificado la concepción que desde un punto de vista mecánico tenían las piezas diseñadas hasta entonces. Dichas tecnologías permiten obtener geometrías muy complejas e imposibles de conseguir por métodos tradicionales, incluyendo la fabricación asistida por ordenador (FAO) basada en tecnologías CNC.

### 1.1. Fabricación aditiva

La fabricación aditiva es un concepto aceptado para todos aquellos procedimientos que inicialmente se crearon únicamente para el desarrollo de prototipos de una manera rápida (“Rapid Prototyping”) pero que con el tiempo han evolucionado hacia lo que hoy en día se denomina “fabricación rápida” (“Rapid Manufacturing”), donde la pieza producida va más allá del simple prototipo y se convierte incluso en pieza de uso final. Las continuas mejoras en la rapidez de los equipos de fabricación aditiva permiten la elaboración de series de piezas a coste cada vez más competitivo y sin necesidad del uso de útiles previos tales como matrices, moldes, electrodos EDM, etc.

A pesar de que dentro del concepto “Rapid Manufacturing” se engloba gran diversidad de procedimientos de producción, existe un cierto acuerdo en denominar como “fabricación aditiva” a todas aquellas cuya forma básica de construcción de la pieza consiste en añadir material (principalmente capa a capa) a partir de la información digital que se obtiene desde un archivo CAD sólido 3D (normalmente en formato STL).

En el sector de las piezas plásticas, las tecnologías de fabricación aditiva tales como las de deposición de material fundido (FDM) o las de sinterizado láser (SLS), entre otras, serían las sustitutas naturales de tecnologías tradicionales como inyección, soplado, rotomoldeo, etc. Sobre todo se muestran muy competitivas como alternativa a la inyección, donde sólo la inversión previa en un molde hace que según el número de piezas a fabricar la rentabilidad pueda hacer interesante el uso de la fabricación aditiva. En el caso de un número de piezas considerablemente elevado, la inyección sería indudablemente más económica sobre todo por el corto tiempo de ciclo y menor

coste de material, salvo que la complejidad de la pieza haga inviable la fabricación por este método (por ejemplo piezas con huecos interiores y contrasalidas imposibles de desmoldar incluso con moldes complicados y costosos).

Por otro lado, las industrias cada vez tienden más a la idea de “fabricación flexible”, un concepto que implica una mayor flexibilidad de los procesos de fabricación para adaptarse a los cambios del mercado, lo cual supone facilitar el lanzamiento de un nuevo producto, aumento de variedad de piezas a fabricar, y sobre todo menor coste de fabricación para pequeños lotes de piezas.

Por tanto, para estos casos en los que se requiere un lote de pocas piezas y en los que existe gran variedad en el producto final, las tecnologías de fabricación aditiva toman ventaja frente a otros procesos de fabricación como la inyección simplemente por la gran flexibilidad que presentan dichas tecnologías sin la necesidad de grandes inversiones.

Dependiendo del proceso de fabricación, el diseño de las piezas varía para adaptarse a las limitaciones de cada tecnología, por lo que la geometría final estará muy influenciada por esto. La fabricación aditiva aventaja al resto de tecnologías en este aspecto ya que existe mayor libertad de diseño por el mero hecho de que existen menos restricciones geométricas asociadas al proceso de fabricación. En el caso de piezas obtenidas mediante mecanizado, extrusión, soplado, rotomoldeo o inyección, existen múltiples restricciones que se han de tener en cuenta para el diseño, como por ejemplo la presencia de huecos interiores en inyección, el espesor de pared en soplado, etc. Empleando técnicas de fabricación aditiva, estas limitaciones son aminoradas en gran medida únicamente por las características del proceso de fabricación.

Otro aspecto destacable de la fabricación aditiva es que no requiere una planificación del proceso de fabricación tan cuidadosa y detallada como ocurre con otras tecnologías. No es necesario analizar la geometría de la pieza para determinar el orden de las operaciones, las herramientas de cada operación, los amarres, etc. Sólo se necesita tener una idea de ciertos detalles dimensionales de la pieza y un mínimo de conocimiento del funcionamiento y software de la máquina de fabricación aditiva.

Estos aspectos anteriores han permitido automatizar el proceso de fabricación de manera que desde un modelo 3D CAD (“Computer Aided Design”) de la pieza en cuestión, ésta puede ser fabricada directamente a partir de la información geométrica del modelo (normalmente formato STL). Por ello, a este tipo de metodología también se le conoce en inglés como “Automated Fabrication” (“Autofab”) (1).

## 1.2. Estructuras celulares

La gran libertad de diseño que implica la fabricación aditiva también permite crear estructuras huecas, con geometrías prácticamente libres. Esta opción, normalmente imposible en otros procesos de fabricación, facilitará la optimización del diseño de piezas mecánicas, como podría ser la minimización del peso (cumpliendo con unos requerimientos resistentes mínimos) mediante estructuras huecas repetidas en el interior de la pieza, consiguiendo por tanto una mayor eficiencia (2) en múltiples aplicaciones. Este tipo de estructuras repetidas en el espacio se conocen como estructuras aligeradas o celulares (“scaffolds” en inglés) (3) (4) y tienen un amplísimo campo de aplicación en la industria en general, tanto en el sector de piezas de plástico como metálicas.

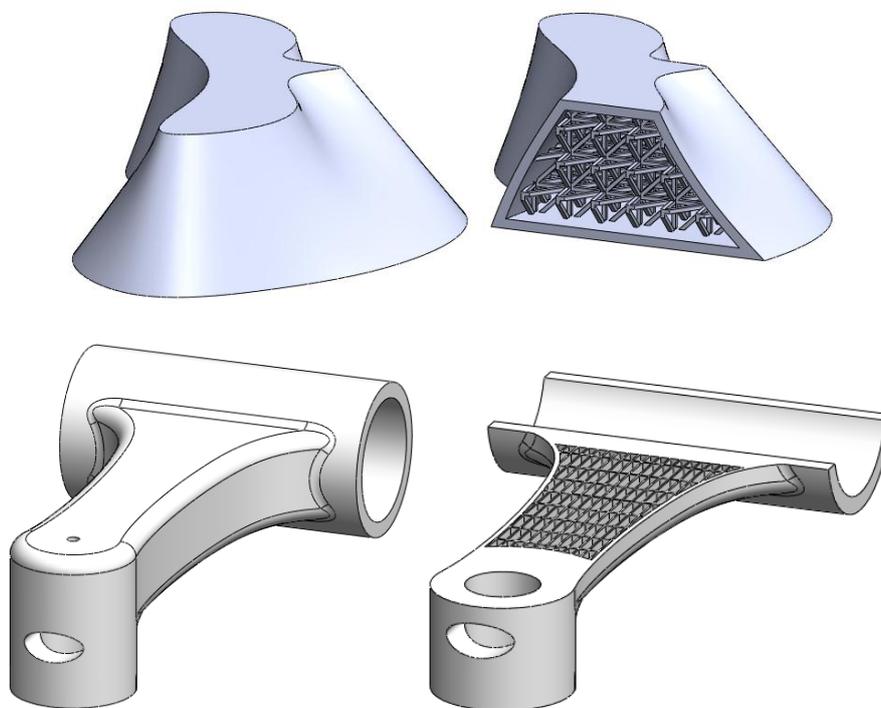


Ilustración 1. Geometrías aligeradas mediante estructuras celulares.

Igualmente, este concepto de estructuras aligeradas se puede aplicar para fabricar piezas que posteriormente se rellenarán con materiales más resistentes, mejorando las propiedades mecánicas de la pieza. En el caso particular de máquinas de fabricación aditiva multimaterial, la pieza reforzada se puede obtener directamente mediante la colocación de diferentes materiales de manera simultánea (1). Esta opción

está ya disponible en algunas tecnologías multimaterial que trabajan con materiales plásticos, mientras que en el caso de metales aún se encuentra en fases de desarrollo.

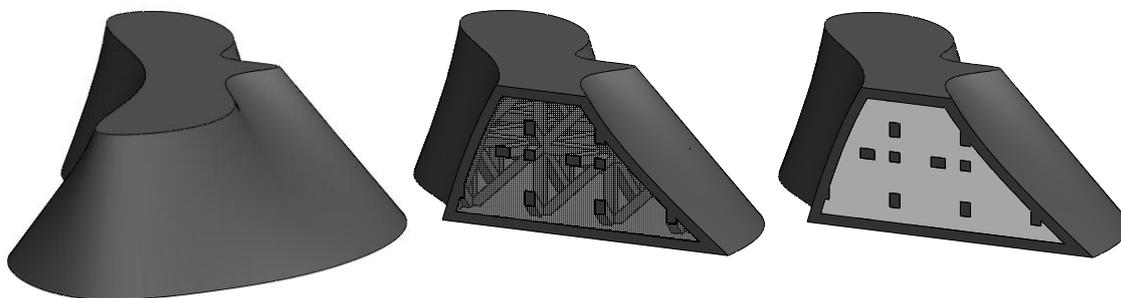


Ilustración 2. Geometría reforzada con dos materiales mediante estructuras celulares.

Teniendo en cuenta estas capacidades de la fabricación aditiva, se ha trabajado en este tema con la finalidad de establecer una metodología que permita al usuario aligerar una pieza con una geometría inicialmente maciza mediante estructuras celulares repetidas en el interior de la misma. Análogamente se busca una metodología para reforzar una geometría mediante la combinación de diferentes materiales empleando también estructuras celulares en el interior.

La metodología debe cumplir con los siguientes requisitos de partida:

- Obtención del diseño de la pieza (con estructura interna celular pero manteniendo el contorno o “piel” del diseño inicial) mediante programas CAD disponibles en el mercado y de fácil implementación en la industria.
- Evaluación del diseño mediante simulaciones con el MEF (“Método de los Elementos Finitos”) empleando igualmente programas comerciales disponibles en el mercado y a ser posible que esté incluido en el propio programa CAD.
- La metodología debe guiar al operario hasta un diseño “óptimo” (o bastante bueno) de la estructura celular, con el objetivo de minimizar el peso pero cumpliendo con las requerimientos resistentes mínimos.
- El proceso debe ser lo más simple, automatizado y eficiente posible para que sirva de aplicación a cualquier industria o usuario de tecnologías de fabricación aditiva.

En el proyecto final de carrera titulado “Análisis teórico y experimental de nuevas estructuras aligeradas en la conformación de piezas plásticas obtenidas a partir de fabricación aditiva” (5) se llevó a cabo una primera propuesta de metodología con este enfoque.

En este Trabajo Fin de Máster se pretende continuar con esta línea de investigación para mejorar la metodología inicial, centrándose en los aspectos de optimización.

## 2. Objetivos

En el trabajo antecesor a este documento (5), se analizaron las herramientas disponibles en el mercado para la generación de los diseños CAD y se eligió un software de diseño y cálculo por elementos finitos en base principalmente a las capacidades de diseño, buscando sobre todo potencialidad para la parametrización de la geometría y regeneración automática de la misma. También se tuvieron en cuenta otros aspectos como la disponibilidad y extensión en el mercado. Una vez elegido el software (SolidWorks), se evaluaron las geometrías más factibles en términos de automatización del diseño de la pieza y tiempo de diseño necesario.

Por otro lado se hizo un estudio de las herramientas de optimización disponibles en dicho software. A partir de la parametrización de ciertas cotas del modelo (referentes a las estructuras celulares) y haciendo uso de un paquete de cálculo incorporado en el propio programa mediante el MEF, es posible definir unas variables de diseño que permiten cambiar la geometría de manera automatizada y simular el comportamiento mecánico de la nueva geometría. Además, también dispone de una aplicación para resolver un problema de optimización, definiendo previamente las variables de diseño, restricciones y objetivos. El método de optimización implementado en el software consiste en un diseño de experimentos inicial en el que se evalúan una serie de diseños, seguido de la obtención de unas superficies de respuesta a partir de los resultados anteriores y, finalmente, la búsqueda del diseño óptimo empleando las estimaciones que se obtienen de las superficies de respuesta anteriores, sin tener que realizar el cálculo por elementos finitos. La estrategia de muestreo inicial se basa en el diseño de experimentos de Box-Behnken (6), que consiste en añadir, para cada grupo de “m” variables posibles, todas las combinaciones de estas variables con el valor máximo y mínimo correspondiente, manteniendo el resto de variables en su valor central. Además de estos puntos se incluye el punto central. Por tanto se evalúan  $2^m$  diseños por cada una de los grupos posibles, por lo que se tendría un total de puntos iniciales definido por  $2^m \cdot C_{n^{\text{variables}}}^m + 1$ , siendo  $C_{n^{\text{variables}}}^m$  la combinatoria del número de variables tomadas de “m” en “m”. Dependiendo del número de variables de diseño, “m” toma diferentes valores, siendo “m=2” para 5 o menos variables de diseño.

Sin embargo, esta herramienta de optimización no es del todo útil ya que en algunas ocasiones el diseño óptimo estimado mediante las superficies de respuesta resulta ser un diseño no factible por incumplir alguna restricción. Esto ocurre porque las superficies de respuesta son una mera aproximación, por lo que un pequeño error

de estimación puede conducir a una estimación de diseño óptimo no factible. En este caso el programa informa de que ha fallado la optimización y no muestra información alguna. Esto implica que no se consiga con seguridad un resultado “óptimo” con esta metodología. Además, cualquier problema de mallado o regeneración de la geometría durante el cálculo de los diferentes diseños detiene el proceso de optimización. No obstante, esto ha mejorado en las últimas versiones del programa.

Para tratar de solventar los problemas anteriores y garantizar la obtención de un diseño “óptimo”, se propuso una metodología de optimización similar a la anterior. Ésta se basa en un diseño de experimentos factorial de 3 niveles completo (6) (más exhaustivo para tener más puntos para el ajuste), una posterior obtención de las superficies de respuesta mediante ecuaciones de orden 2 ajustadas por mínimos cuadrados en el Statgraphics y seguidamente, una estimación final de los 10 mejores diseños mediante un programa en Matlab que realiza un barrido incremental sobre todo el dominio en busca de estos 10 mejores diseños estimados (las estimaciones se realizan con las ecuaciones de ajuste obtenidas anteriormente). Finalmente, se simulan estos 10 diseños y se escoge el mejor de ellos. El hecho de simular los 10 mejores diseños estimados garantiza en cierta medida obtener un diseño factible entre esos 10 mejores, ya que existe un cierto error en las estimaciones.

Partiendo de esta experiencia y de trabajos posteriores (7) basados en este método, el objetivo de este trabajo fin de máster consiste en mejorar dicha metodología para la optimización del diseño de estructuras celulares en la minimización del peso de piezas obtenidas mediante fabricación aditiva, centrándose exclusivamente en el método de optimización (dadas las características del máster) y dejando para líneas futuras otros asuntos como la automatización del proceso, el tipo de estructuras celulares más apropiadas, validación experimental del método, etc.

### 3. Estado del arte

Entre las múltiples referencias relacionadas con esta línea de investigación, cabe destacar sin duda alguna los trabajos realizados por D. Rosen (“Georgia Institute of Technology”). Inicialmente propuso un método de modelado paramétrico de estructuras de barras para fabricación aditiva orientado a piezas de dimensiones considerables (8). Posteriormente, a partir del modelado paramétrico anterior, realizó otros trabajos de gran interés centrados en la optimización de la estructura de barras para completar una metodología que permita obtener piezas más ligeras con estructuras repetidas interiormente pero con buenas propiedades mecánicas (9) (10).

La metodología planteada por este autor consiste en primer lugar en obtener el fichero de triangulación (formato STL) que define el contorno de la pieza. Posteriormente se divide el contorno en diferentes superficies que servirán para aproximar la geometría mediante superficies de Bezier en Matlab. Estas superficies o parches permitirán delimitar las estructuras de barras que se generarán de manera parametrizada (también mediante un programa en Matlab) y orientando las barras en la dirección de las cargas en la medida de lo posible. Finalmente se obtiene la información geométrica de cada uno de los vértices de las barras que se han unido a cada parche, se reagrupan correctamente y se introduce esta información geométrica en el ANSYS para finalmente optimizar el diámetro de las barras (minimizando el peso pero conservando las propiedades mecánicas exigidas). En esta fase cabe destacar que se agrupan las barras en función de las tensiones que soportan tras un cálculo inicial, de manera que todas las barras de un mismo grupo tendrán el mismo diámetro, limitando por tanto el número de variables de diseño. Finalmente la geometría óptima se exporta en STL para proceder a su fabricación.

Esta metodología presenta múltiples ventajas que la convierten en una herramienta muy interesante. En primer lugar, es posible definir una estructura de barras de manera “automatizada” y además orientando (en cierta medida) las barras en la dirección de las cargas. Esto implica un uso eficiente del material ya que se consigue que gran parte de ellas trabajen a compresión/tracción, aumentando la rigidez relativa al peso. Por otro lado, el hecho de trabajar con estructuras de barras definidas mediante la información geométrica de los vértices facilita enormemente las simulaciones y optimización del diámetro de las barras en ANSYS, ya que es posible realizar un mallado con elementos tipo “beam”, es decir, elementos lineales (de 1D), siendo los nodos de los elementos los propios vértices de las barras. Esto evita realizar un mallado de elementos volumétricos (como tetraedros, hexaedros, etc.), lo cual

reduce el coste computacional drásticamente. La optimización se simplifica ya que básicamente consiste en variar la sección que define cada elemento (es decir, cada barra), por lo que el mallado no varía de un diseño a otro, reduciendo muchísimo el tiempo de cómputo. También hay que darle un gran valor a la capacidad para definir una geometría con estructura interna formada por centenas de miles de barras, lo cual es inviable en programas CAD 3D mediante operaciones booleanas. Algunos programas permiten crear estructuras celulares a partir de un modelo macizo, como es el caso del software “netfabb”, pero sin capacidad de simulación/optimización.

Sin embargo, este método también presenta algunos aspectos negativos. En primer lugar, la metodología planteada requiere de bastante trabajo del diseñador, ya que debe exportar la geometría en STL, dividir el contorno en diferentes superficies, obtener en Matlab el fichero que define la geometría, introducirlo en ANSYS, realizar un primer cálculo, reagrupar las barras según las tensiones, y finalmente definir las variables de diseño y aplicar la optimización hasta obtener la geometría óptima. A pesar de que puede ser un método muy potente, también es cierto que es un proceso complejo y de difícil automatización, sobre todo por el flujo de trabajo entre diferentes programas (CAD-Matlab-ANSYS). Por otro lado, también cabe destacar que esta metodología no sería viable para el caso de querer reforzar una pieza combinando diferentes materiales a modo de estructuras celulares y relleno en el interior de la pieza. Todas las ventajas que conlleva el mallado con elementos de 1D no serían aplicables en este caso, ya que habría que mallar el espacio comprendido entre las barras y simular el contacto entre ambos materiales. Otro factor de menor importancia es que con esta metodología, la longitud de las barras será siempre la misma durante el proceso de optimización. A pesar de que en el modelado paramétrico de la estructura se puede definir la longitud de las barras, esta dimensión no entra en juego en la optimización final, por lo que la única forma de tener en cuenta esta variable en la optimización sería repitiendo todo el proceso para diferentes longitudes de barra, lo cual sería inviable.

Teniendo en cuenta que se pretende establecer una metodología lo más simple y eficiente posible y que también sea aplicable a la combinación de diferentes materiales, se descartó esta línea de trabajo, ya que la parte novedosa radica sobre todo en el proceso para generar la información geométrica de una pieza con estructuras celulares. Además el presente trabajo pretende centrarse en el proceso de optimización y no tanto en el procedimiento de generación de la geometría celular.

Una metodología con cierta similitud a la anterior (aunque desvinculada de la fabricación aditiva y de las estructuras celulares), es la desarrollada en el artículo “Multiphysics CAD-Based Design Optimization” (11). La propuesta de optimización

comienza con una optimización topológica inicial. Esto consiste en simular la geometría de partida y localizar las zonas con menores cargas. En estas zonas es donde el diseñador debe eliminar material para reducir peso. Una vez localizada esta zona, se rediseña la pieza eliminando material y parametrizando las dimensiones. Posteriormente, se realiza el mallado en un programa de elementos finitos y se optimiza la posición de los nodos en las zonas donde se pretende eliminar material. Esto se conoce como optimización no paramétrica, ya que no se modifica la geometría del modelo en cada iteración, sólo se recolocan algunos nodos de la malla, reduciendo el tiempo de cálculo. Finalmente, al concluir el proceso de optimización se genera la geometría CAD a partir de la información de la malla. Para realizar estas tareas se emplean programas comerciales CAD/MEF como el Unigraphics NX3 y el HYI-3D, que interactúan a través de un postprocesador desarrollado por los autores (Prep3D).

La limitación de esta metodología radica en que se requiere el tratamiento del fichero de la malla y esta tarea se complica sobre todo en el caso de tener que modificar miles de estructuras celulares o cambios importantes en las dimensiones. Además no permite una optimización topológica sino optimización de tamaño. En otras palabras, si se trata de una pieza con una estructura interior de 4 barras, la optimización variará la geometría de las barras pero no el número de éstas. A diferencia del método anterior, en este caso sí sería posible modificar la longitud de las barras (siempre y cuando se mantenga el número de éstas). Por contrapartida, se requiere una breve modificación de la malla en cada diseño, lo cual no sucedía en la metodología anterior.

Centrándose más en los aspectos de optimización se han encontrado varias referencias de interés. En el artículo “Structural Optimization Tool using Genetic Algorithms and Ansys” (12) se aplica un algoritmo genético (AG por sus siglas, o “GA” en inglés) para optimizar el diseño de diferentes piezas. La evaluación de la función objetivo de los diferentes individuos en el algoritmo genético se lleva a cabo empleando los resultados de distintas simulaciones mediante elementos finitos a través del programa ANSYS. El código del algoritmo genético se relaciona con el ANSYS de dos modos: o empleando lenguaje de programación del propio ANSYS (“Ansys Parametric Design Language”, APDL); o bien mediante comandos UPF (“User Programmable Features”) en lenguaje Fortran, el cual es compatible con este software. Empleando esta metodología el autor consigue optimizar el diseño de piezas con un gran número de variables de diseño.

Este planteamiento implica la regeneración de la geometría, mallado y cálculo para cada uno de los individuos que se evalúan en las múltiples generaciones del algoritmo genético. Según el autor, en uno de los casos prácticos se realizaron 75000 simulaciones en ANSYS. Evidentemente esto se consiguió mediante estaciones de

trabajo en paralelo (“clusters”) que permiten reducir drásticamente el tiempo de cálculo. Por tanto, esta metodología será viable únicamente para aquellos usuarios que cuenten con estaciones de trabajo de este tipo con suficiente capacidad como para abordar estos problemas. Además, de experiencias anteriores se ha observado que en el ANSYS no existen operaciones o herramientas que permitan generar estructuras celulares en el interior de una pieza de manera sencilla y parametrizable, a modo de patrón repetido en el espacio. Por otro lado, la libertad de diseño que se deriva de las tecnologías de fabricación aditiva “obliga” en cierta medida al uso de un software CAD con más potencial para generar geometrías complejas, por lo que en este sentido el ANSYS se ve desfavorecido a pesar de tratarse de uno de los programas de cálculo más potentes en el mercado.

Sin embargo, esta metodología de aplicación directa de algoritmos genéticos podría ser rentable en tiempo de cómputo cuando se trate de problemas en los que las simulaciones sean relativamente rápidas o en problemas con un gran número de variables de diseño en los que otros métodos de optimización dejan de ser una opción por el elevado número de simulaciones requeridas (crecimiento exponencial del número de simulaciones con el número de variables de diseño). Finalmente, también hay que valorar positivamente los algoritmos genéticos como herramienta de optimización, destacando frente a otros métodos deterministas en el hecho de que se pueden aplicar sin necesitar una ecuación de evaluación ni su derivada.

Otros autores han realizado trabajos similares al anterior. Por ejemplo, en el artículo “CAD-based Evolutionary Design Optimization with CATIA V5” (13) los autores proponen un método de optimización basado también en algoritmos genéticos. Para ello, se define la geometría en CATIA correctamente parametrizada y se transfiere al ANSYS para hacer las simulaciones correspondientes por elementos finitos. La interacción entre estos dos programas comerciales y el programa que incorpora el algoritmo genético se realiza mediante un programa en C++ realizado por los autores (DynOPS). Este programa se encarga de leer la información genética de cada uno de los individuos que genera el algoritmo genético, para posteriormente pasar dicha información al CATIA, regenerar la geometría atendiendo al genotipo, recuperar el fichero con la información geométrica para pasarla al ANSYS y finalmente esperar por las simulaciones. Una vez leídos los resultados los envía al algoritmo genético para que evalúe la aptitud del individuo. Para comparar la metodología con otros métodos, se usó una de las herramientas de optimización incorporadas en el CATIA. Entre los métodos de optimización disponibles en este software (gradiente conjugado, recocido simulado y diseño de experimentos), se comparó con el recocido simulado (“Simulated Annealing”) (14) por ser el más parecido a los algoritmos genéticos (método estocástico), concluyendo que la metodología planteada es más versátil.

La aplicación de esta metodología requiere el flujo de trabajo entre 4 programas diferentes (programa del algoritmo genético, CATIA, ANSYS y DynOPS), dificultando el proceso. Además, el hecho de simular todos los individuos de las generaciones del algoritmo genético implica el uso de varias estaciones de trabajo en paralelo para poder abordar el problema. Incluso bajo estas condiciones la optimización puede tardar varios días.

También existen múltiples referencias de métodos de optimización basados en métodos de aproximación para estimar los resultados sin realizar tantas simulaciones. En el artículo “Structural Design Examples Using Metamodel-Based Approximation Model” (15), los autores emplean un metamodelo de aproximación basado en el método Kriging para minimizar el peso de diferentes piezas. Este modelo de aproximación (Kriging) permite estimar los resultados a partir de un número menor de simulaciones, por lo que esta metodología es muy interesante en el caso de que el cálculo de cada nuevo diseño implique un elevado tiempo de cómputo. En este artículo concreto, la optimización se lleva a cabo en el ANSYS, que incorpora internamente el método Kriging.

Esta propuesta reduce considerablemente el número de simulaciones para llegar a la solución óptima en comparación con la optimización directa mediante algoritmos genéticos. Sin embargo, el ANSYS no dispone de herramientas CAD potentes que permitan la regeneración automatizada de estructuras celulares.

En aplicaciones biomédicas también se encuentran ejemplos de estos métodos de optimización. Por ejemplo en el artículo “Multiobjective design optimisation of coronary stents” (16), se optimizan las dimensiones de stents coronarios mediante un diseño de experimentos inicial de 15 individuos, una posterior construcción del metamodelo de aproximación mediante el Kriging, y un algoritmo genético de optimización multiobjetivo (NSGA-II) que evalúa la aptitud de los individuos mediante los resultados estimados con el interpolador Kriging. Tras finalizar el algoritmo genético, se escogen 5 individuos distribuidos a lo largo de la frontera de Pareto y se simulan. Estos últimos puntos evaluados permiten actualizar de nuevo el metamodelo de aproximación, mejorando su aproximación. Este proceso se repite en varias iteraciones (mejorando paulatinamente el metamodelo de aproximación) hasta cumplir el criterio de parada y obtener el diseño óptimo final.

Igualmente, en otros muchos sectores como el aeronáutico se han aplicado técnicas de optimización similares a la anterior. Así por ejemplo, en el artículo “Multiobjective Optimization Using Approximation Model-Based Genetic Algorithms” (17) se emplea un diseño de experimentos inicial, seguido de la construcción de un

modelo de aproximación basado en el método Kriging, y una optimización final mediante algoritmos genéticos multiobjetivo, con el fin de optimizar las dimensiones de un avión supersónico a través de cálculos CFD (“Computational Fluid Dynamics”) por elementos finitos. Como peculiaridad, cabe destacar que el diseño de experimentos (DDE por sus siglas, o “DOE” en inglés) empleado fue el “Latin Hypercube”. Éste consiste en elegir un número de puntos de muestreo iniciales y dividir cada dimensión del dominio en tantas partes iguales como puntos elegidos, obteniendo una rejilla de celdas. Posteriormente se van asignando los puntos de muestreo iniciales de manera aleatoria pero recordando la fila y columna (en el caso de 2D) de todos los puntos anteriores para evitar que existan dos puntos en la misma fila o columna. Otro diseño de experimentos que mejora la exploración del “Latin Hypercube” es el ortogonal. Éste se diferencia del anterior en que el dominio se divide también en tantos subdominios iguales como puntos de muestreo se pretendan incluir, y la asignación de éstos se realiza añadiendo un punto en cada subdominio y comprobando que el muestreo en todo el dominio cumple con las condiciones del DDE “Latin Hypercube”. De este modo se consigue un muestreo igual al anterior pero con una densidad de muestreo idéntica en cada subdominio.

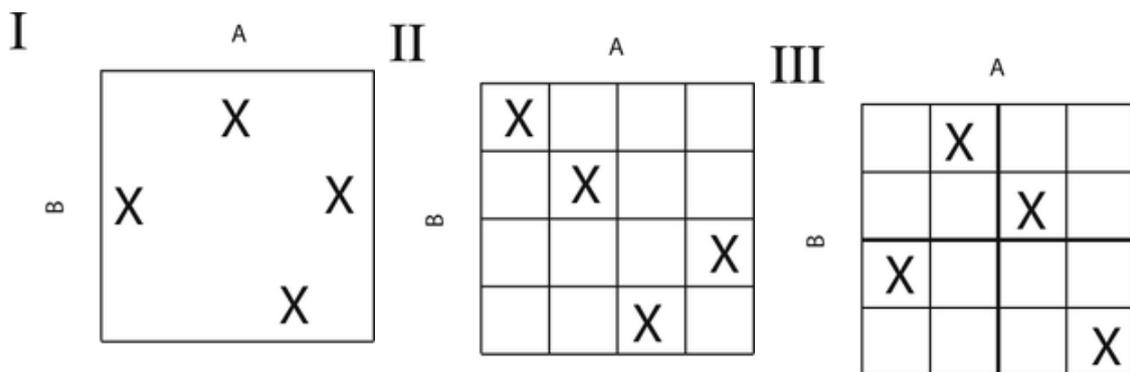


Ilustración 3. De izquierda a derecha: DDE “random”, “Latin Hypercube” y “Ortogonal”.

Este planteamiento es muy interesante ya que combina las ventajas de los metamodelos de aproximación con la optimización de un algoritmo genético.

También se llevó a cabo una búsqueda de las herramientas de optimización disponibles en los programas comerciales de cálculo por elementos finitos. Anteriormente ya se ha hecho mención a los 3 tipos de optimización incorporados en Catia y la posibilidad de diseño de experimentos de Box-Behnken y optimización mediante superficies de respuesta en el SolidWorks. Entre ellos cabe destacar principalmente el ANSYS. Este programa cuenta con múltiples opciones para la optimización, principalmente por estar relacionado con el modeFRONTIER.

El modeFRONTIER es una plataforma de integración para optimización multiobjetivo, problemas multidisciplinares y toma de decisiones en un proceso automatizado. Es una de las herramientas de optimización más potentes del mercado, empleada en diferentes sectores y por múltiples marcas de reconocimiento mundial (Siemens, Sony, Ford, Volvo, Audi, Toyota, Pirelli, etc.). Permite interactuar con diferentes programas CAD/FEM (ANSYS, Catia, SolidWorks, Abaqus, Pro/ENGINEER, etc.), de manera que los nuevos diseños propuestos en el algoritmo de optimización se evalúan automáticamente en los programas CAD/FEM con los que interactúa. Cuenta internamente con distintas técnicas de diseño de experimentos (de exploración, factoriales completos y reducidos, ortogonales, etc.), metamodelos de aproximación (interpolaciones polinomiales, vecino más cercano, superficies paramétricas, distribuciones gaussianas, redes neuronales), distintas técnicas/algoritmos de optimización (basados en gradiente, algoritmos genéticos, recocido simulado, estrategias de evolución, basados en superficies de respuesta, etc.) y otras aplicaciones.

El ANSYS dispone también de muchas de estas herramientas. Cuenta con diferentes métodos de optimización como el MOGA (“Multi-Objective Genetic Algorithm”), el NLPQL (“Non-linear Programming by Quadratic Lagrangian”, basado en gradiente mediante el método de cuasi-Newton) o el “Screening” para un diseño preliminar (método no iterativo de muestreo mediante un generador cuasi-aleatorio basado en el algoritmo de Hammersley (18)), así como múltiples tipos de diseños de experimentos (central compuesto, Box-Behnken, etc.) (6), y metamodelos de aproximación (superficies de respuesta con polinomios ajustados de orden 2, Kriging, redes neuronales, “Sparse Grid”, etc.).

7	Design of Experiments	
8	Design of Experiments Type	Central Composite Design
9	Design Type	Central Composite Design Optimal Space-Filling Design Box-Behnken Design Custom Custom + Sampling Sparse Grid Initialization
7	Meta Model	
8	Response Surface Type	Standard Response Surface - Full 2-nd Ord...
9	Refinement	Standard Response Surface - Full 2-nd Order Polynomials
10	Refinement Type	Kriging Non Parametric Regression
11	Verification Points	Neural Network
12	Generate Verification Points	Sparse Grid

Ilustración 4. Diseños de experimentos y metamodelos disponibles en ANSYS.

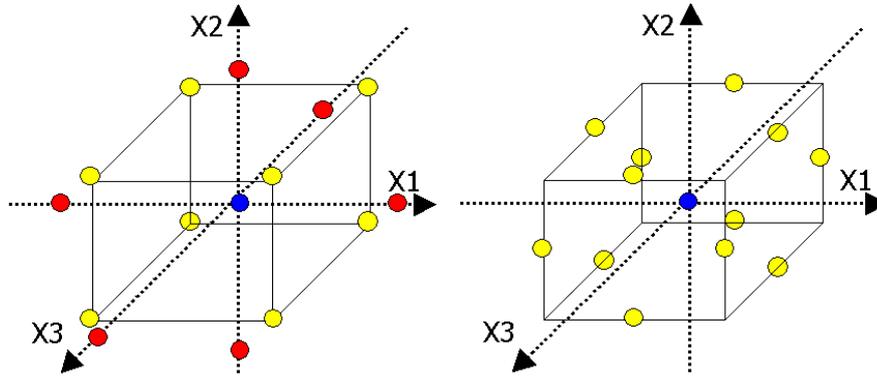


Ilustración 5. Diseño de experimentos central compuesto (izquierda) y de Box-Behnken (derecha) en un dominio 3D.

Entre los diseños de experimentos disponibles, cabe destacar el central compuesto y el de Box-Behnken por ser los más empleados. El diseño central compuesto evalúa todas las combinaciones posibles de valor máximo y mínimo de cada variable (es decir, muestreo factorial de 2 niveles), que equivale a evaluar todas las esquinas del dominio (puntos en amarillo de la imagen izquierda anterior). Además también añade dos puntos más por cada variable, desplazados levemente por encima del valor máximo y por debajo del valor mínimo, manteniendo constantes el resto de variables. Estos puntos gráficamente se corresponden a los 2 puntos en cada eje central del dominio (puntos en rojo en la imagen izquierda anterior). Sin embargo, el diseño de experimentos de Box-Behnken incluye, para cada una de las parejas de variables posibles, todas las combinaciones con los valores máximo y mínimo de estas 2 variables, manteniendo el resto en su valor central. Gráficamente esto se interpreta como los puntos medios de todas las aristas que delimitan el dominio (puntos en amarillo en la imagen derecha anterior).

Además de estas opciones, el ANSYS también dispone de algunos métodos de refinamiento automático del metamodelo de aproximación, como es el caso del Kriging o el "Sparse Grid". En el Kriging, el propio modelo matemático interpolador (basado en un modelo estadístico) determina el punto donde se produce un mayor error entre el valor real y el valor estimado, por lo que el refinamiento consiste en ir agregando nuevos puntos en aquellas zonas de mayor error de interpolación. El "Sparse Grid" (19) es un tipo de refinamiento jerárquico basado en funciones bases a trozos multilineales. Para ello, es necesario llevar a cabo un diseño de experimentos también del tipo "Sparse Grid". El algoritmo de refinamiento se construye a partir de un tensor tipo "sparse". En problemas de múltiples variables de diseño con influencia dispar en los resultados, el refinamiento mediante una rejilla regular en todo el dominio implicaría un elevado número de puntos de refinamiento que sería inviable. Por ello, el "Sparse Grid" emplea un algoritmo adaptativo dimensionalmente para detectar la

dimensión o variable con mayor error de aproximación e ir añadiendo nuevos puntos en esa dirección.

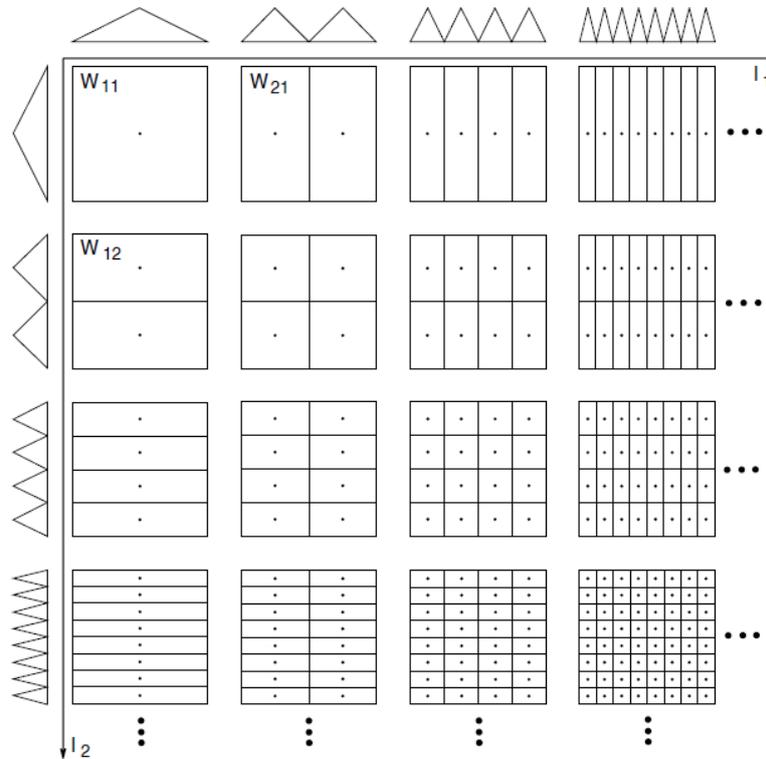


Ilustración 6. Esquema de refinamiento "Sparse Grid".

Por último, también hay que destacar otro método de optimización más novedoso y totalmente diferente de los planteamientos nombrados hasta el momento. Se trata de las bases reducidas ("reduced basis") (20). Se caracterizan por emplear un número reducido de las funciones base empleadas internamente en el cálculo por elemento finitos, reduciendo por tanto el coste computacional de las simulaciones. La elección de estas bases se lleva a cabo de tal manera que se minimiza el error en los resultados. Además, esta estimación del error permite determinar las zonas con peor ajuste para calcular nuevos puntos que mejoren paulatinamente la precisión de los mismos.

A pesar de que este planteamiento puede ser muy potente, su aplicabilidad depende de que se pueda expresar la ecuación de gobierno del problema de una forma determinada ( $a(\cdot, \cdot; u) = \sum_{q=1}^{Q_a} a^q(\cdot, \cdot) \theta_a^q(u)$ ). Por otro lado, se necesita información de las bases empleadas en el cálculo por elementos finitos. Esta información no está disponible en los programas comerciales MEF, por lo que esta metodología es de excesiva dificultad y por tanto poco viable para esta aplicación concreta.

## 4. Nuevas propuestas de optimización adaptadas al problema

Tras analizar anteriormente los diferentes métodos de optimización y sus ventajas e inconvenientes para el problema en cuestión, se propusieron nuevas ideas para mejorar la metodología inicial desarrollada en trabajos previos (5) (7).

En primer lugar, se optó por abordar el problema mediante la optimización basada en diseño de experimentos, metamodelo de aproximación y optimización final. Esta metodología permite reducir el número de simulaciones necesarias para llegar al diseño óptimo, disminuyendo por tanto el tiempo de diseño y los costes asociados. A pesar de que existen muchas alternativas de optimización basadas en este concepto (como se observó anteriormente), en este trabajo se analizaron nuevas técnicas con la finalidad de mejorar la eficiencia del proceso de optimización para este problema específico de minimización del peso con estructuras celulares en fabricación aditiva.

Por tanto, el trabajo fin de máster tratará de desarrollar y mejorar los pasos de diseño de experimentos, obtención del metamodelo de aproximación y búsqueda del diseño óptimo a partir del metamodelo de aproximación.

### 4.1. Búsqueda del diseño óptimo

El primer aspecto que se podría mejorar con respecto a la metodología de trabajos previos (5) (7) es el algoritmo de búsqueda del diseño óptimo. Anteriormente, esta búsqueda se realizaba mediante un barrido por todo el dominio con un cierto paso de muestreo en cada variable de diseño, evaluando para cada caso las ecuaciones ajustadas de las diferentes restricciones del problema y de la masa (superficies de respuesta). Durante este bucle se iban almacenando las combinaciones de variables que cumplían todas las restricciones para posteriormente ordenarlas atendiendo a la masa. Una vez hecho esto, se escogían los 10 mejores diseños estimados de esta lista ordenada (es decir, los 10 primeros casos de la lista) y se calculaban por elementos finitos para finalmente escoger el mejor de ellos.

Este método puede ser práctico en el caso de dominios pequeños que permitan usar frecuencias de muestreo muy pequeñas sin suponer esto un coste computacional excesivo. Sin embargo, el hecho de trabajar con una frecuencia de muestreo para cada

variable implica que la búsqueda del óptimo sea poco precisa, sobre todo en aquellas variables en las que un ligero incremento suponga un gran cambio en las respuestas del sistema. En el caso de usar frecuencias de muestreo muy pequeñas para tratar de acercarse lo más posible al punto óptimo, el número de iteraciones del bucle de búsqueda aumentaría exponencialmente, por lo que sería inviable. Igualmente, a medida que aumenta el número de variables de diseño, el método de búsqueda anterior es cada vez más ineficiente.

Para solucionar esto, se plantea el uso de algoritmos genéticos que permitirán obtener el punto óptimo con mayor rapidez a partir de las ecuaciones de ajuste. También se podrían plantear otros métodos deterministas como los basados en el gradiente, pero hay que decir que estos últimos requieren de una expresión continua y diferenciable para su aplicación, e incluso en algunos problemas pueden converger hacia óptimos locales. Sin embargo, los algoritmos genéticos no requieren de continuidad, diferenciable o convexidad en la función a optimizar y además tienen mayor capacidad de exploración, pudiendo “saltar” durante la evolución de un óptimo local a un óptimo global.

## 4.2. Metamodelo de aproximación

Otro de los aspectos mejorables de la metodología de partida consiste en reducir la interacción entre diferentes programas, es decir, tratar de unificar todo el proceso empleando el menor número de programas posible. Por ello, se pretende crear un programa que además de incorporar el código del algoritmo genético de optimización, también incluya el metamodelo de aproximación de las diferentes respuestas del sistema.

En la metodología de partida, el ajuste se realizaba por mínimos cuadrados mediante una ecuación de orden 2 (con todas las interacciones entre variables incluidas). Uno de los objetivos de este trabajo fin de máster es evaluar diferentes metamodelos de aproximación y escoger el más apropiado teniendo la capacidad de ajuste en relación al número de puntos necesarios para su entrenamiento, así como otros aspectos como su simplicidad. En el siguiente capítulo se detalla con mayor profundidad los diferentes métodos evaluados a modo comparativo.

A pesar de que estas propuestas podrían suponer una mejora notable en la metodología de optimización, el cuello de botella se encuentra en otras fases del proceso. El tiempo de diseño requerido para un caso práctico con cierta complejidad no

se vería reducido significativamente con estas nuevas aportaciones, ya que lo que ralentiza enormemente la metodología es el número de simulaciones necesarias para obtener un metamodelo lo suficientemente representativo de la realidad. Por tanto, la parte más importante de la metodología es el diseño de experimentos y sobre todo el algoritmo de refinamiento del metamodelo de aproximación.

### 4.3. Bucle iterativo de refinamiento del metamodelo

Para tener un cierto control sobre la bondad del metamodelo de aproximación, se plantea un proceso iterativo en el que se vayan definiendo y simulando nuevos puntos que permitirán ir mejorando paulatinamente el ajuste del metamodelo hasta que el error de las estimaciones sea inferior a un cierto valor.

Por tanto, se tratará de implementar un algoritmo con un bucle en el que en cada iteración se añadan los resultados de los nuevos diseños evaluados para actualizar el metamodelo. Con esta técnica, el ajuste irá mejorando con respecto a los resultados simulados. Al finalizar cada iteración, se estimarán los resultados del siguiente nuevo punto de diseño y se ejecutará el cálculo por elementos finitos de dicho punto. Mientras las diferencias entre las estimaciones del metamodelo de aproximación y los resultados simulados sean considerables, el bucle se repetirá mejorando progresivamente el ajuste. Cuando los errores entre las estimaciones y los resultados simulados sean inferiores a un cierto valor, entonces se dará por bueno el metamodelo de aproximación. No obstante, para aprovechar este último punto simulado, se introducirá en la nube de puntos y se actualizará de nuevo el metamodelo mejorando su ajuste.

Finalmente, con un metamodelo de aproximación “fiable” se ejecutaría el algoritmo genético de optimización final para realizar la búsqueda del diseño con menor masa que cumpla las restricciones establecidas. Tras encontrar el diseño óptimo según el algoritmo genético y basándose en el metamodelo de aproximación, es necesario corroborar que dicho diseño cumple las restricciones impuestas en el problema, ya que evidentemente el metamodelo es una aproximación de la realidad. Por tanto, se calcula dicho diseño en el programa de elementos finitos y se comprueba si realmente cumple las restricciones. En el caso de que no cumpla todos los requisitos, entonces se reconstruiría de nuevo el metamodelo de aproximación añadiendo este último punto y mejorando por tanto el ajuste. Posteriormente se realizaría de nuevo la búsqueda del diseño óptimo mediante el algoritmo genético. Este proceso se repite en

un bucle hasta que el diseño óptimo estimado cumpla con las restricciones del problema.

#### 4.4. Algoritmo para la generación de nuevos puntos en el diseño de experimentos y refinamiento del metamodelo de aproximación.

Por último, faltaría por definir un diseño de experimentos y un criterio para ir añadiendo los nuevos puntos en el refinamiento del metamodelo. Tanto los puntos para el diseño de experimentos inicial como para el refinamiento del metamodelo deben ser elegidos de manera que exploren las zonas de interés con la mayor eficiencia posible. Una correcta localización de los puntos de muestreo no solo reducirá el número de puntos necesarios, sino que también aumentará la precisión del metamodelo de aproximación, ya que se construye a partir de éstos.

Como se comentó anteriormente, algunos programas como el ANSYS ya disponen de ciertas herramientas para tal fin, como el Kriging o el "Sparse Grid". La principal diferencia entre estos 2 métodos es que en el Kriging se mejora la superficie de respuesta en aquellas zonas con peor ajuste, mientras que la metodología "Sparse grid" mejora el ajuste en todo el espacio de trabajo en la dirección de la dimensión de mayor error de ajuste. A pesar de que estos métodos podrían ser muy interesantes, se pretende implementar un diseño de experimentos más adaptado y eficiente a este problema concreto y que trate de añadir nuevos puntos en aquellas zonas de interés.

Hay que tener presente que la finalidad de esta línea de investigación es desarrollar una metodología para dimensionar eficientemente las estructuras celulares repetidas en el interior de una pieza fabricada mediante "Additive Manufacturing", tanto en el caso de pretender aligerar como en el caso de reforzar mediante la combinación de diferentes materiales. Por esta razón, el objetivo siempre será minimizar la masa pero manteniendo las propiedades mecánicas (resistencia y rigidez) dentro de unos valores exigidos. Se trata por tanto de un problema monoobjetivo con una serie de restricciones que están siempre en conflicto con el objetivo.

En este caso de optimización en los que intervienen únicamente variables geométricas asociadas a la definición de la estructura celular, existe una relación directa entre dichas variables y las respuestas del sistema (el objetivo y las restricciones). Así por ejemplo, en una pieza aligerada mediante una estructura celular a modo de celdas huecas repetidas en el interior de la misma, un aumento de una

variable geométrica asociada al tamaño de las celdas huecas implica siempre una disminución de masa y un aumento de tensiones y desplazamientos al someterse a una carga dada. Es decir, un incremento de esta variable siempre conllevará una menor masa pero mayores desplazamientos y tensiones, por lo que existe una relación monótona entre las variables y las respuestas.

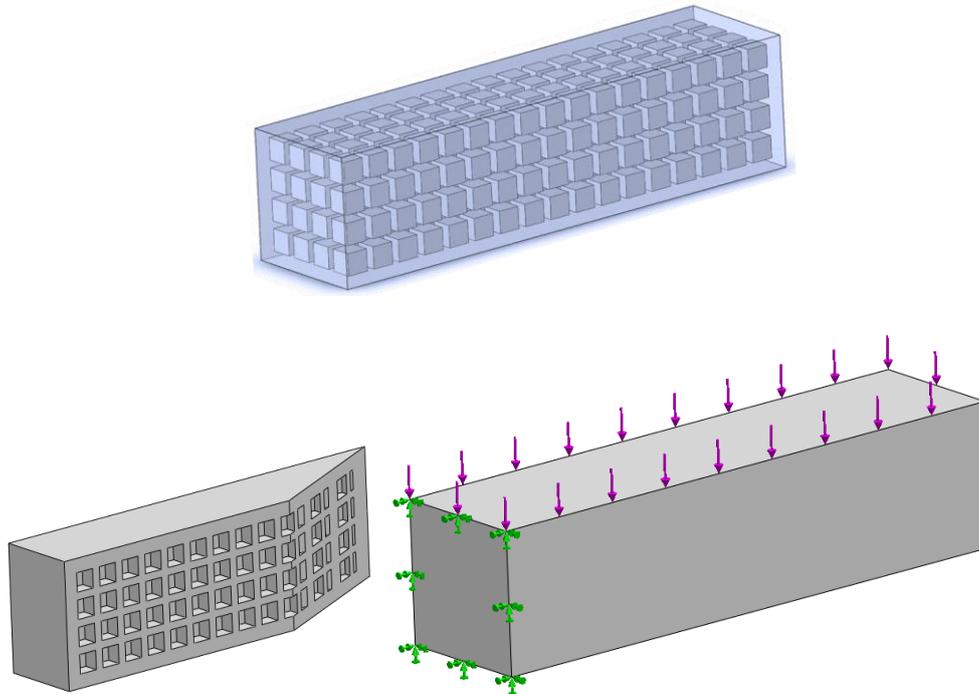


Ilustración 7. Ejemplo de geometría con estructura celular en el interior (huecos cúbicos). Geometría de las celdas interiores (arriba), sección de la pieza (izquierda) y cargas aplicadas (derecha).

En este contexto es lógico deducir que las soluciones óptimas (diseños factibles con menor peso) estarán siempre en los límites entre los valores aceptables de tensiones y/o desplazamientos y los valores inaceptables, es decir, en la frontera entre el espacio factible (diseños que cumplen todas las restricciones) y no factible (diseños que no cumplen todas las restricciones).

Por tanto, la estrategia más rápida para llegar a la solución óptima consiste en localizar esta frontera. Para ello se pretende programar un diseño de experimentos que, a partir de un muestreo inicial, vaya añadiendo nuevos puntos cada vez más próximos a ésta. Esta estrategia podría desencadenar en soluciones relativamente buenas sin tener que aplicar técnicas de optimización ni metamodelos de aproximación.

Teniendo en cuenta que la solución óptima se encontrará en esta frontera, interesa construir un metamodelo de aproximación bastante preciso en dicha zona y

más basto en el resto del dominio. Para ello simplemente se debe aumentar la densidad de muestreo en los alrededores de la frontera.

Aplicando este concepto de aproximación a la frontera tanto en la fase del diseño de experimentos como en el refinamiento del metamodelo, se puede conseguir una mejora sustancial frente a la metodología de partida (5) (7) e incluso frente a otras técnicas de optimización existentes.

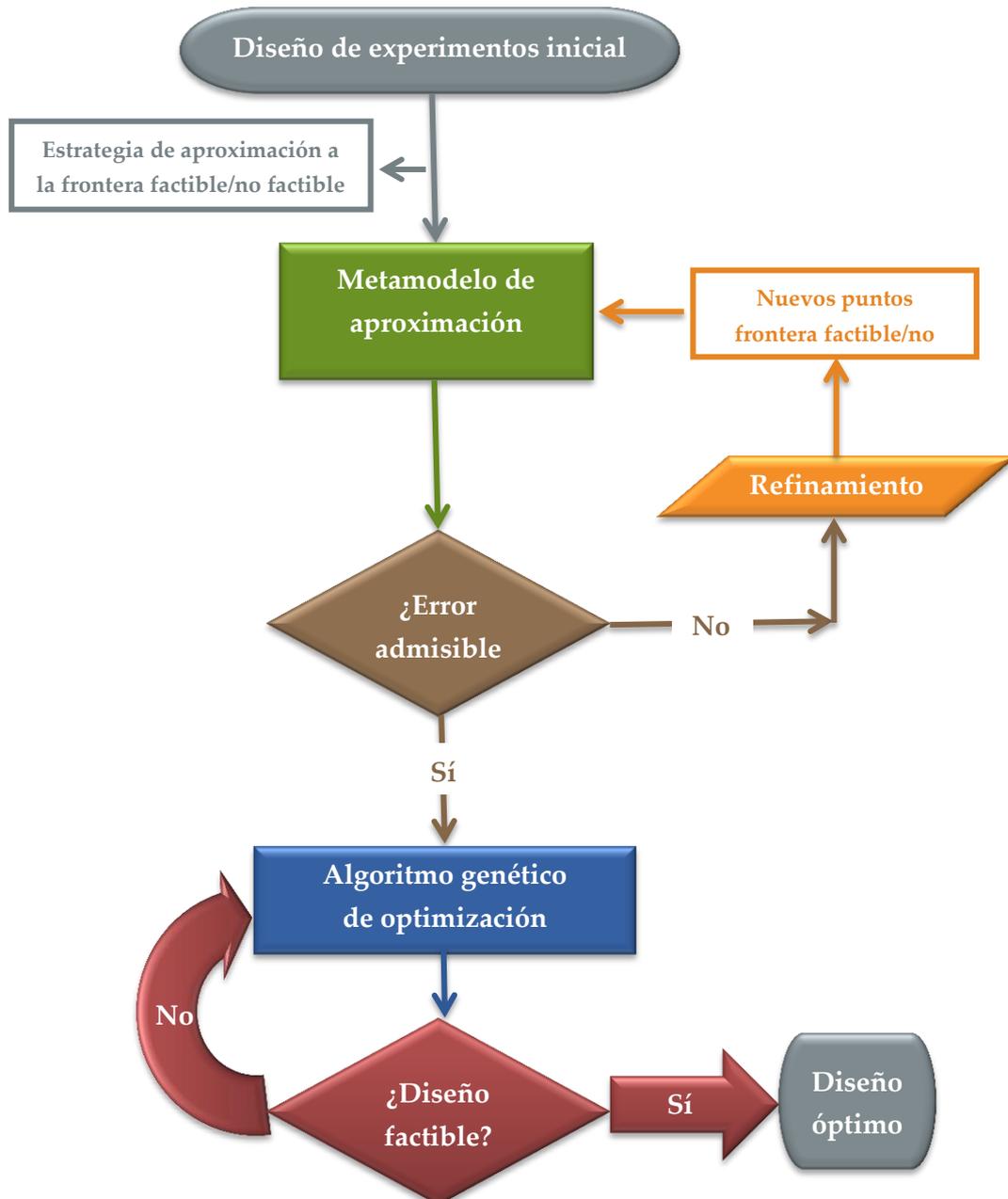


Ilustración 8. Diagrama de flujo de la metodología de optimización.

## 5. Comparación entre metamodelos de aproximación

El primer aspecto analizado para desarrollar la metodología propuesta consistió en un estudio comparativo entre distintos tipos de metamodelos de aproximación (estimadores de resultados). A pesar de que existen múltiples tipos de metamodelos posibles, la primera duda surge en torno a la elección entre dos grandes conjuntos bien diferenciados. Por un lado se encuentran los metamodelos basados en métodos de interpolación, y por otro lado los basados en métodos de regresión (o superficies de respuesta).

La principal diferencia funcional entre estos dos tipos de metamodelos es que en el caso de modelos de regresión, la función empleada para realizar las estimaciones está ajustada para minimizar el error cuadrático medio de las estimaciones en los puntos dato frente a los valores reales de esos datos, mientras que en los modelos de interpolación la ecuación está ajustada de manera que las estimaciones en los puntos que son dato coincidan con los valores reales.

Normalmente los modelos de regresión se emplean cuando se conoce la forma de la ecuación de la respuesta que se pretende estimar. En estos casos se ajustan los parámetros de dicha ecuación mediante un método iterativo para minimizar el error cuadrático medio (ajuste por mínimos cuadrados). Con ello se consigue una relación (variabilidad explicada) entre cada variable independiente y la respuesta. Sin embargo en el caso de desconocer la forma de la ecuación de respuesta (logarítmica, polinómica, etc.) será necesario asumir un tipo de expresión y ajustar los parámetros de la misma. Evidentemente el tipo de expresión empleado influirá en la calidad del ajuste. En los casos en los que no se conoce el tipo de correlación entre las respuestas y las variables independientes, la práctica más habitual consiste en emplear una ecuación polinómica de orden 2 con todos los términos posibles, incluyendo las interacciones entre variables (5) (7). No obstante, podría plantearse usar una ecuación polinómica de mayor número de términos y por tanto de mayor flexibilidad y adaptabilidad de la superficie de respuesta a la nube de puntos. Esto se conseguiría aumentando el orden de la ecuación polinómica de ajuste, pero también podría suponer posibles oscilaciones en la respuesta que ocasionarían un aumento del error en la estimación. Es por esta razón por la que normalmente no se trabajan con ecuaciones polinómicas de orden mayor que 2. Sin embargo, en este trabajo fin de máster se pretende realizar una comparación entre las estimaciones realizadas por una ecuación de ajuste de orden 2, y una ecuación de ajuste de orden 3, en ambos casos con todos los términos cruzados posibles

(interacción entre factores) para ver si existe diferencia y valorar la opción más adecuada.

Por otro lado, los modelos de interpolación se emplean cuando se pretende estimar un resultado a partir de una serie de puntos con valores conocidos (puntos de muestreo). La gran ventaja de los métodos de interpolación frente a los de regresión es que en los puntos empleados para construir el modelo las estimaciones son exactas, es decir, de error nulo. Esto implica que si se tienen dos puntos conocidos con bastante proximidad entre ellos, la estimación de un nuevo punto entre ambos será muy próxima al valor real. En definitiva, los métodos de interpolación tiene la ventaja de que el hecho de aumentar la densidad de muestreo en una zona puede conllevar a un metamodelo con mucha precisión en dicha zona. Sin embargo en el caso de superficies de respuesta mediante modelos de regresión, el incremento de muestreo en una zona no conseguirá tanta precisión como lo haría un interpolador, ya que los parámetros de la ecuación de ajuste se determinan minimizando el error cuadrático medio en todos los puntos dato  $y$ , además, están muy limitados por la forma predeterminada de la ecuación de ajuste. A pesar de ello existen algunas técnicas para mejorar la precisión de las superficies de respuesta como la reducción paulatina del espacio de trabajo para que la superficie de respuesta se adapte únicamente a los puntos de dicha zona (21).

Aparte de los modelos de regresión y de interpolación, se puede diferenciar otro conjunto de metamodelos de aproximación, los basados en redes neuronales. Consisten en un método de aprendizaje y estimación inspirado en el funcionamiento del sistema nervioso de los animales. Así por ejemplo, un cerebro aprende mediante la reorganización de las conexiones sinápticas entre las neuronas que lo componen. Del mismo modo, las redes neuronales (aplicadas al aprendizaje de modelos) tienen un gran número de procesadores interconectados que simulan de manera simplificada la funcionalidad de las neuronas biológicas. La reorganización de las conexiones sinápticas se modela mediante un mecanismo de pesos, que son ajustados durante la fase de aprendizaje. Además de los pesos y las conexiones, cada neurona tiene asociada una función matemática denominada función de transferencia, la cual genera la señal de salida de la neurona a partir de las señales de entrada.

En el artículo “An efficient algorithm for Kriging approximation and optimization with large-scale sampling data” (22), el autor realiza una breve introducción destacando las ventajas e inconvenientes de estos tres tipos de metamodelos de aproximación. De las superficies de respuesta obtenidas mediante ajuste por mínimos cuadrados (regresión), destaca su efectividad en problemas con un bajo número de variables de diseño. Sin embargo, están muy limitadas por la forma de la ecuación de ajuste, de modo que si se desconoce la expresión de comportamiento del

problema será muy difícil elegir el tipo de ecuación más apropiada. De las redes neuronales destaca su capacidad para resolver problemas más complejos con menor error que las superficies de respuesta. Por contrapartida las redes neuronales requieren un muestreo muy denso para la fase de aprendizaje y muy costoso computacionalmente, lo cual lo convierte en un método aplicable sólo para problemas muy complejos con un gran número de variables de diseño. Por ello se ha descartado esta opción como metamodelo de aproximación para el problema en cuestión. Por último completa la tercera tipología de metamodelos con el Kriging, destacando su capacidad para resolver problemas tan complejos como las redes neuronales, mejorando el ajuste a medida que aumenta el número de puntos de muestreo. Sin embargo, el aumento del número de puntos supone un enorme coste computacional (mayor que en el caso de superficies de respuesta y que incluso de las redes neuronales) ya que para cada punto de estimación se deben calcular los coeficientes de peso de cada uno de los puntos de muestreo, mediante la resolución de un sistema de ecuaciones. Por tanto, cuanto mayor sea el número de puntos, más grande será el sistema de ecuaciones. Por ello el autor propone un nuevo método para el cálculo de los coeficientes de peso del método Kriging en problemas en los que se requiera un gran número de puntos de muestreo.

El método Kriging es el modelo de interpolación más comúnmente empleado en los problemas de optimización, tanto aplicado directamente como con novedosas modificaciones propuestas (15) (16) (17) (22), de ahí su disponibilidad en el propio ANSYS. Este método asume que la distancia y la dirección entre puntos de muestreo es una expresión de la correlación espacial entre los puntos y que por tanto dicha información puede emplearse para explicar la variabilidad en la respuesta (23) (24). El algoritmo ajusta una función matemática a un determinado número de puntos o a aquéllos que se encuentren dentro de un radio de búsqueda. Está considerado como uno de los mejores métodos de interpolación ya que provee estimaciones insesgadas (de sesgo nulo, es decir, error nulo entre valor esperado de la estimación y el real) y de varianza mínima. Es un método muy elaborado y con fundamento estadístico, lo cual lo convierte en el interpolador más exacto estadísticamente hablando y probablemente el más apropiado cuando la intensidad de muestreo es considerable. Sin embargo el algoritmo normalmente requiere de cierto conocimiento estadístico previo y es bastante complejo. Además, cuando el distanciamiento entre los puntos de muestreo es muy grande el Kriging no funciona correctamente y deja de ser una opción competitiva frente a otro tipo de metamodelos de aproximación. Dado que la finalidad de este trabajo es definir una metodología para optimizar las geometrías celulares con el menor número de simulaciones posible (intensidad de muestreo baja), se ha desechado

esta opción para esta aplicación concreta, valorando más otros metamodelos más simples y con menos densidad de muestreo.

Los metamodelos evaluados en este estudio comparativo fueron los siguientes:

- Método de la distancia inversa ponderada (exponente 2 y 3)
- Interpolación spline
- Superficies de respuesta mediante ajuste por mínimos cuadrados con ecuaciones polinómicas (de orden 2 y 3)
- Interpolación del vecino más cercano
- Interpolación lineal basada en triangulación de Delaunay

A continuación se explica brevemente el fundamento de cada uno de ellos.

### 5.1. Método de la distancia inversa ponderada

Este método de aproximación asume que cada punto del conjunto de datos de muestreo tiene una influencia local que disminuye con la distancia. Por lo tanto los valores de los puntos cercanos al punto donde se pretende estimar el resultado tienen mayor importancia o peso en el valor que será asignado al mismo. Normalmente la búsqueda de los puntos a tener en cuenta en la estimación se hace considerando un número de puntos o un radio de influencia para no tener en cuenta todos los datos del dominio y despreciar la influencia de los más alejados. La expresión para estimar el resultado en un punto del dominio viene dada por:

$$u_e = \frac{\sum_{i=1}^{n^{\text{ptos}}} \frac{u_i}{d_i^p}}{\sum_{i=1}^{n^{\text{ptos}}} \frac{1}{d_i^p}}$$

Siendo “ $u_e$ ” el valor estimado en el punto de interés, “ $n^{\text{ptos}}$ ” el número de puntos de muestreo que están dentro del radio de influencia y que por tanto influirán en la estimación, “ $u_i$ ” el resultado para el punto de muestreo “ $i$ ” (dentro del radio de influencia), “ $d_i$ ” la distancia desde el punto de interés hasta el punto “ $i$ ” y “ $p$ ” el parámetro del exponente que toma un valor entero entre 1 y 3 (normalmente  $p=2$ ). Cuanto mayor sea este parámetro, más rápidamente pierde peso un punto al alejarse del punto de interés y por tanto más exacta es la estimación. En el estudio comparativo se probó este método con  $p=2$  y  $p=3$ .

El método de la distancia inversa se comporta como exacto en los puntos de muestreo cuando no se utiliza ningún factor de suavizado.

Entre los inconvenientes de este método de interpolación cabe destacar que genera una superficie con una apariencia de “ojos de buey” alrededor de los puntos de muestreo y que, en el caso de que el número de puntos de muestreo sea elevado, requiere un alto coste computacional para evaluar las distancias del punto de interés a los puntos de muestreo (25). Sin embargo, en algunas aplicaciones puede ser una buena opción dada su simplicidad a la hora de programar el algoritmo. A modo comparativo, algunos autores (26) recomiendan este método frente al Kriging cuando la base de datos disponible sea pequeña (los variogramas no son conocidos) o cuando la distancia entre los puntos de muestreo sea muy grande.

## 5.2. Interpolación spline

Este interpolador consiste en un conjunto de polinomios cúbicos que describen la tendencia de una serie de puntos. Se asemeja a un metamodelo de regresión pero en este caso la interpolación se lleva a cabo localmente y no en todo el dominio. Otra diferencia es que la “superficie” obtenida con la interpolación spline contiene a todos los puntos de muestreo, mientras que en los métodos de regresión esto no ocurre.

Es muy usado en aplicaciones CAD para interpolar superficies a partir de unos nodos de control. El modelo matemático ajusta las combinaciones lineales de splines (dando lugar a las B-splines en 3D) en cada zona del dominio para minimizar la curvatura de la respuesta a partir de los puntos de muestreo. Además de minimizar la curvatura, la superficie interpolada debe pasar por todos los puntos de muestreo y tener la misma derivada en los extremos de cada tramo para que la superficie sea suave. La forma de la superficie final va a depender de un parámetro de tensión que hace que el comportamiento de la superficie interpolada tienda a asemejarse a una membrana más o menos tensa que pasa por los puntos de muestreo. En general produce resultados muy buenos, siendo apropiado para superficies que varían de forma gradual. Sin embargo no se recomienda para extrapolar valores ni cuando las respuestas del sistema sufren cambios muy abruptos.

La ventaja fundamental del método de interpolación mediante splines cúbicas con respecto a otros como el de la distancia inversa ponderada, el vecino más cercano o la interpolación lineal es que, con estos últimos, los valores interpolados nunca pueden

ser ni mayores ni menores que los valores de los puntos utilizados para interpolar, por lo que resulta imposible interpolar correctamente máximos y mínimos (25).

Sin embargo, su implementación computacional es bastante compleja y sobre todo su refinamiento está muy limitado al hecho de que se requiere una distribución de datos a modo de rejilla para reconstruir el modelo de interpolación, de manera que no se puede aplicar un refinamiento local en las zonas de interés. Por tanto, si se pretende refinar el metamodelo de aproximación en una zona del dominio, será necesario evaluar todos los puntos de la nueva rejilla, por lo que el coste computacional sería inviable en múltiples aplicaciones. Este hándicap ha conllevado al desarrollo de las T-splines, que permiten el refinamiento localizado a través de la subdivisión de cada cuadrante de la rejilla (27) (28).

### 5.3. Superficies de respuesta mediante ajuste por mínimos cuadrados con ecuaciones polinómicas

Este metamodelo de aproximación consiste en ajustar los coeficientes de una ecuación polinómica para adaptarse a una serie de datos y poder explicar la respuesta del sistema en función de las variables de entrada, es decir, las variables de diseño. Para ajustar los coeficientes a los datos se lleva a cabo un ajuste por mínimos cuadrados, que consiste en minimizar el error cuadrático medio entre los valores reales de los puntos y los valores estimados con la superficie de respuesta.

Como se ha comentado anteriormente, cuando se desconoce la expresión de la respuesta del sistema se emplean ecuaciones polinómicas de orden 1 ó 2 (29). En el caso de emplear una ecuación de orden 1, la expresión sería la siguiente:

$$u_e = c_0 + \sum_{i=1}^n c_i \cdot var_i$$

Siendo “ $u_e$ ” el valor estimado en el punto de interés, “ $c_0$ ” el coeficiente del término independiente del polinomio, “ $n$ ” el número de variables de diseño (o factores), “ $c_i$ ” el coeficiente del polinomio para “ $var_i$ ” y “ $var_i$ ” el valor de la variable  $i$ -ésima. El valor de los coeficientes “ $c_0 \sim c_n$ ” se obtiene minimizando (derivando e igualando a 0) la siguiente expresión:

$$\text{Min} \frac{\sum_{i=1}^{n^{\text{ptos}}} (u_i - u_e)^2}{n^{\text{ptos}}} = \text{Min} \frac{\sum_{i=1}^{n^{\text{ptos}}} (u_i - (c_0 + \sum_{i=1}^n c_i \cdot var_i))^2}{n^{\text{ptos}}}$$

Siendo “ $u_i$ ” el valor de la respuesta en el punto “ $i$ ” y “ $n^{\text{ptos}}$ ” el número de puntos de muestreo.

Para modelos de orden 2, la ecuación sería:

$$u_e = c_0 + \sum_{i=1}^n c_i \cdot var_i + \sum_{i=1}^n \sum_{\substack{j=1 \\ j \geq i}}^n c_{ij} \cdot var_i \cdot var_j$$

Siendo “ $c_{ij}$ ” los coeficientes del polinomio para los términos de orden 2 y “ $var_i$ ” y “ $var_j$ ” las interacciones entre las variables “ $i$ ” y “ $j$ ”. El valor de los coeficientes “ $c_0 \sim c_n$ ” y “ $c_{11} \sim c_{nn}$ ” se obtienen minimizando la siguiente expresión:

$$\begin{aligned} & \text{Min } \frac{\sum_{i=1}^{n^{\text{ptos}}} (u_i - u_e)^2}{n^{\text{ptos}}} = \\ & = \text{Min } \frac{\sum_{i=1}^{n^{\text{ptos}}} \left( u_i - \left( c_0 + \sum_{i=1}^n c_i \cdot var_i + \sum_{i=1}^n \sum_{\substack{j=1 \\ j \geq i}}^n c_{ij} \cdot var_i \cdot var_j \right) \right)^2}{n^{\text{ptos}}} \end{aligned}$$

En este trabajo se probará también con ecuaciones polinómicas de orden 3. Para este caso la expresión tendría la siguiente forma:

$$u_e = c_0 + \sum_{i=1}^n c_i \cdot var_i + \sum_{i=1}^n \sum_{\substack{j=1 \\ j \geq i}}^n c_{ij} \cdot var_i \cdot var_j + \sum_{i=1}^n \sum_{\substack{j=1 \\ j \geq i}}^n \sum_{\substack{k=1 \\ k \geq j}}^n c_{ijk} \cdot var_i \cdot var_j \cdot var_k$$

Siendo “ $c_{ijk}$ ” los coeficientes del polinomio para los términos de orden 3 y “ $var_i$ ”, “ $var_j$ ” y “ $var_k$ ” las interacciones entre las variables “ $i$ ”, “ $j$ ” y “ $k$ ”. El valor de los coeficientes “ $c_0 \sim c_n$ ”, “ $c_{11} \sim c_{nn}$ ” y “ $c_{111} \sim c_{nnn}$ ” se obtienen minimizando la siguiente expresión:

$$\begin{aligned} & \text{Min } \frac{\sum_{i=1}^{n^{\text{ptos}}} (u_i - u_e)^2}{n^{\text{ptos}}} = \\ & = \text{Min } \frac{\sum_{i=1}^{n^{\text{ptos}}} \left( u_i - \left( c_0 + \sum_{i=1}^n c_i \cdot var_i + \sum_{i=1}^n \sum_{\substack{j=1 \\ j \geq i}}^n c_{ij} \cdot var_i \cdot var_j + \sum_{i=1}^n \sum_{\substack{j=1 \\ j \geq i}}^n \sum_{\substack{k=1 \\ k \geq j}}^n c_{ijk} \cdot var_i \cdot var_j \cdot var_k \right) \right)^2}{n^{\text{ptos}}} \end{aligned}$$

La principal ventaja de este método de aproximación es que tiene buenas propiedades de extrapolación, es simple y la ecuación de ajuste es continua y derivable. Sin embargo tiene el inconveniente de ser una mera aproximación (no da

resultados exactos en los puntos de muestreo) y está muy limitado por la forma de la ecuación.

#### 5.4. Interpolación del vecino más cercano

El método del vecino más cercano está basado en el diagrama de Voronoi. Este diagrama se obtiene mediante la partición del dominio en “q” polígonos N-dimensionales de tal manera que cada polígono contiene a un solo punto del muestreo inicial y todos los puntos interiores del mismo están más próximos al punto central asociado que a cualquier otro punto de muestreo (25). De este modo, la interpolación consiste básicamente en asignar a todos los puntos contenidos en un “polígono” el valor del punto de muestreo asociado al mismo, es decir, el punto de muestreo más cercano.

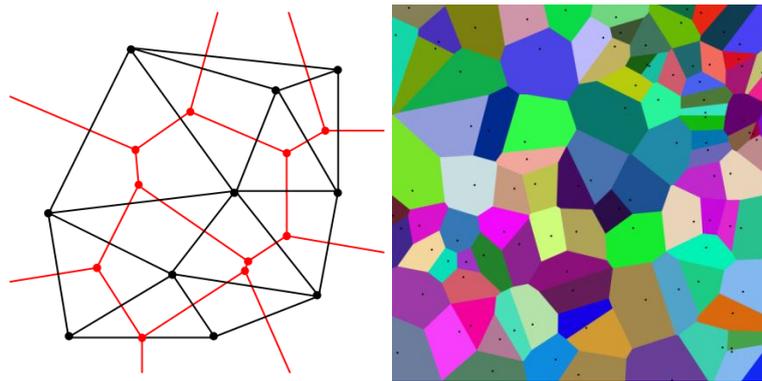


Ilustración 9. Diagrama de Voronoi en 2D (izquierda) y mapa de colores de la interpolación del vecino más cercano (derecha).

La principal ventaja de este método es que es muy simple y que permite estimar resultados fuera de la envolvente convexa. Sin embargo, los resultados interpolados son constantes en cada polígono N-dimensional, por lo que es un método poco preciso y discontinuo.

Otra alternativa más elaborada también basada en el diagrama de Voronoi es el la interpolación del vecino natural. Este método estima el resultado de un punto como la suma ponderada de los vecinos naturales. La determinación de los vecinos naturales se realiza incluyendo el punto a estimar en el diagrama de Voronoi, de modo que las aristas del nuevo polígono determinan los vecinos naturales (con aristas adyacentes). La interpolación del resultado en dicho punto se calcula mediante la suma ponderada

de éstos, siendo el peso de cada vecino igual al área que se solapa del nuevo polígono sobre el polígono del vecino antes de incluir el nuevo punto, en tanto por uno. Con ello se consigue una interpolación continua, más suave y derivable en todo el dominio excepto en los puntos dato (25). Sin embargo sólo es aplicable en el dominio comprendido en el interior de la envolvente convexa de los datos, mientras que el interpolador del vecino más cercano no tiene esta limitación.

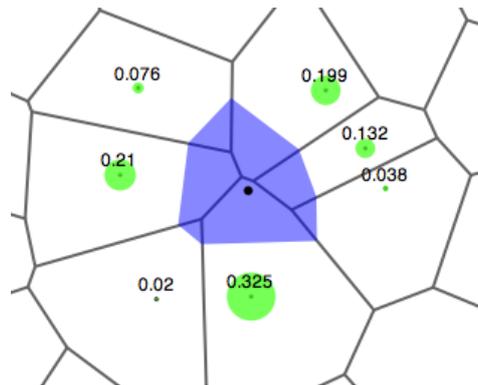


Ilustración 10. Interpolación del vecino natural.

### 5.5. Interpolación lineal basada en triangulación de Delaunay

La triangulación de Delaunay consiste en dividir el dominio en una red de símplex N-dimensionales (triángulos en 2D, tetraedros en 3D, etc.) que cumplan la condición de Delaunay. Esta condición se cumple si la geometría que circunscribe cada símplex de la red no contiene ningún vértice de otro símplex. Para una mejor comprensión, la condición de Delaunay se cumpliría en 2D si la circunferencia que circunscribe cada triángulo de la red no contiene ningún vértice de otro triángulo. Si se unen los centros de cada una de las circunferencias asociadas a triángulos con aristas compartidas, se obtiene el diagrama de Voronoi, por lo que la triangulación de Delaunay y el diagrama de Voronoi están relacionados.

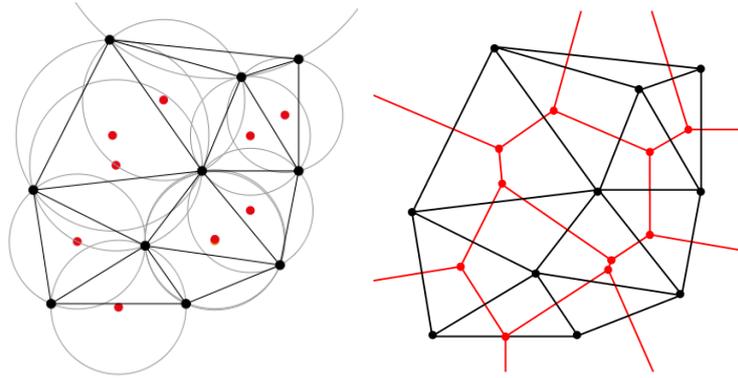


Ilustración 11. Triangulación de Delaunay (izquierda) y diagrama de Voronoi resultante (derecha).

Una vez realizada la triangulación de Delaunay, el primer paso para llevar a cabo la interpolación lineal consiste en identificar en qué simplex se encuentra el punto que se pretende estimar. Posteriormente, el valor interpolado de éste se obtiene a partir de la suma ponderada de los valores de cada uno de los vértices del simplex (interpolación lineal). El peso de cada vértice se determina mediante las coordenadas baricéntricas del punto a estimar. Así por ejemplo en el caso de un problema en 2D, el valor “p” estimado vendría dado por:

$$p = \sum_{i=1}^3 w_i \cdot p_i$$

Siendo “w<sub>i</sub>” y “p<sub>i</sub>” el peso y valor respectivamente para el punto “i”. El peso “w<sub>i</sub>” se determina como el cociente entre el área asociada al vértice “i” y el área total del triángulo (equivale a la coordenada baricéntrica):

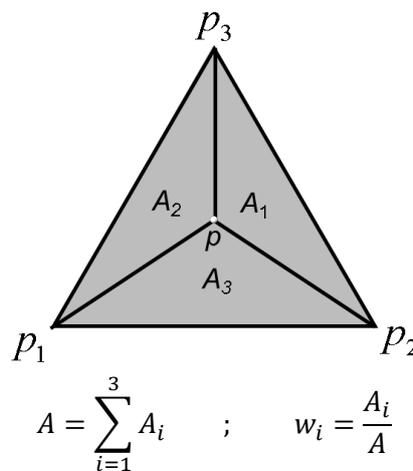


Ilustración 12. Determinación del peso de cada vértice para la interpolación lineal en 2D.

Análogamente, para un caso n-dimensional las expresiones vendrían dadas por:

$$p = \sum_{i=1}^{n+1} w_i \cdot p_i \quad ; \quad A = \sum_{i=1}^{n+1} A_i \quad ; \quad w_i = \frac{A_i}{A} \quad ; \quad (n = \text{dimensión ó } n^{\text{a}} \text{ variables})$$

La principal ventaja de este método es que su algoritmo es bastante rápido y que la respuesta contiene a los puntos de muestreo. Sin embargo sólo es aplicable al dominio comprendido dentro de la envolvente convexa y la superficie resultante contiene discontinuidades en la derivada en cada arista de los simplex en los que se divide el dominio.

## 5.6. Estudio comparativo

El estudio comparativo entre los metamodelos anteriores se llevó a cabo sobre un ejemplo ilustrativo (7) que se define a continuación y que servirá de referencia para diferentes pruebas realizadas en este trabajo fin de máster.

### 5.6.1. Ejemplo de referencia

Se trata de la geometría de una pala de un microaerogenerador de aproximadamente 1 m de longitud, aligerada con estructuras celulares en el interior y orientada a fabricación aditiva. El objetivo de este problema consiste optimizar el diseño de las estructuras celulares para minimizar el peso de la pala del microaerogenerador manteniendo la deflexión máxima inferior a 15mm para una carga total de 50N a flexión distribuida en la superficie de sobrepresión. El material asignado para las simulaciones de elementos finitos fue el ABS P400, que se corresponde con el material empleado para fabricación mediante FDM (“Fused Deposition Modeling”).



Ilustración 13. Ensayo a flexión de pala real de aerogenerador (izquierda) y simulación a modo de ejemplo para la geometría de referencia (derecha).

En esta geometría la estructura celular interior está definida por una serie de huecos con forma de paralelepípedos, concretamente prismas con base cuadrada, aunque en otras pruebas posteriores se trabajó con huecos cúbicos o incluso con huecos con forma de prisma con base rectangular.



Ilustración 14. Vistas seccionadas de la geometría con estructuras celulares en el interior.

Las variables de diseño para este caso concreto son 4: la dimensión del lado de la base del prisma cuadrado hueco ( $L$ ), el espesor exterior ( $e$ ) de la piel que define el contorno de la pieza, el espesor de pared entre huecos ( $e_h$ ) y la altura de los prismas cadrados ( $h$ ).

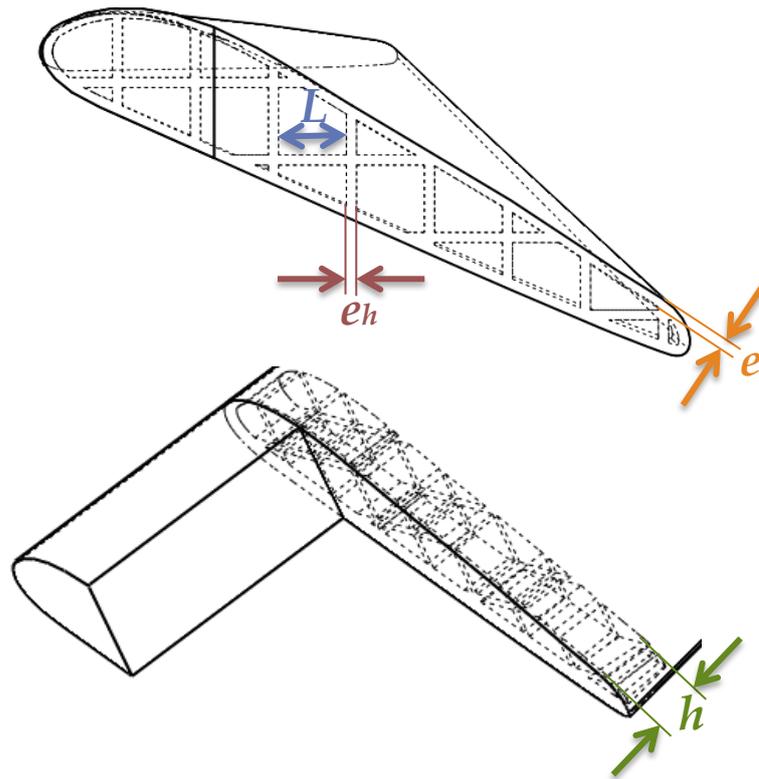


Ilustración 15. Variables de diseño del problema de referencia (4 variables).

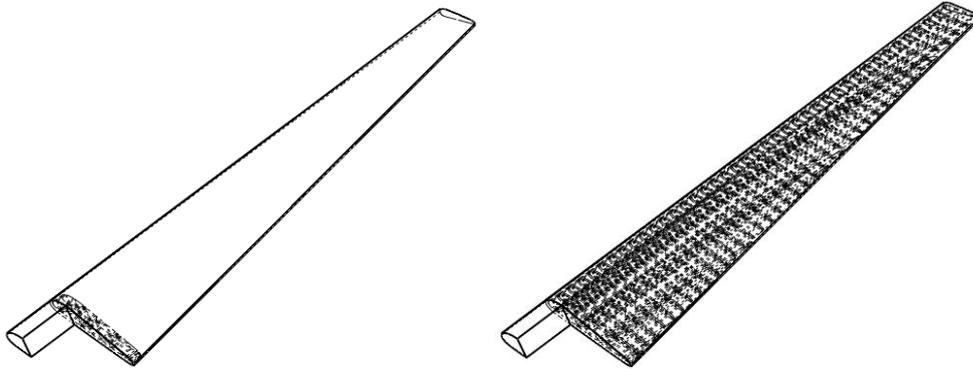


Ilustración 16. Reproducción de las geometrías celulares en el interior de la pieza (patrón repetido en el espacio).

El intervalo de trabajo o espacio de búsqueda definido para cada una de las variables fue el siguiente:

- $L \rightarrow 20\sim 60\text{mm}$
- $e \rightarrow 3\sim 8\text{mm}$
- $e_h \rightarrow 3\sim 8\text{mm}$
- $h \rightarrow 20\sim 60\text{mm}$

Estos intervalos se mantuvieron durante todas las pruebas realizadas con esta geometría en particular. El mallado empleado fue un mallado con refinamiento automático que disminuye el tamaño del elemento en las zonas de mayor curvatura. Los elementos empleados fueron tetraedros con funciones base parabólicas (con nodos en los vértices del tetraedro y en el centro de las aristas). El número total de nodos es de aproximadamente 120000.

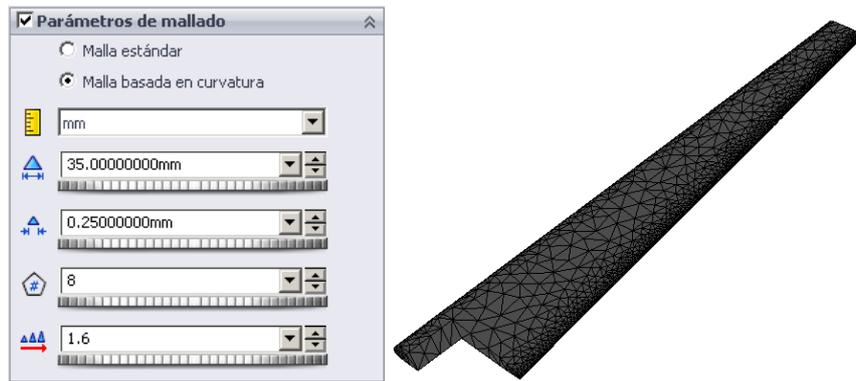


Ilustración 17. Mallado adaptativo basado en curvatura.

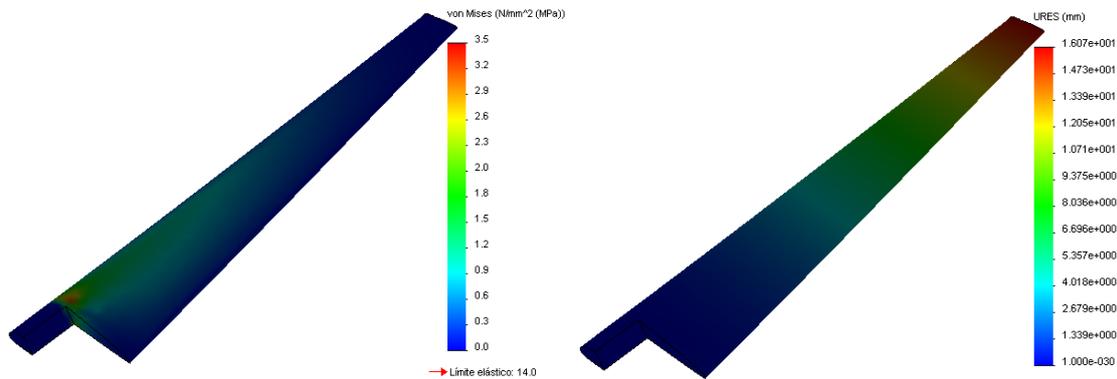


Ilustración 18. Gráfico de tensiones de Von Mises (izquierda) y campo de desplazamientos (derecha).

Para obtener directamente los valores de interés de cada diseño (masa y desplazamiento máximo), se emplearon 2 sensores que almacenan automáticamente estos valores.



Ilustración 19. Sensores de masa y desplazamiento máximo del modelo.

### 5.6.2. Resultados del estudio

Para realizar una comparación entre los metamodelos anteriores se llevó a cabo un diseño de experimentos inicial (con el ejemplo de referencia definido en el apartado anterior) y posteriormente, con esta nube de puntos, se elaboraron los modelos de interpolación en el lenguaje de programación Matlab, que ya incluye algunas funciones internas de interpolación. Finalmente se estimaron los resultados con los distintos métodos en una serie de puntos aleatorios previamente simulados por elementos finitos para evaluar el error cometido en la interpolación.

El diseño de experimentos inicial fue factorial de 3 niveles completo ( $3^k$ ), de modo que se requieren  $3^{n^{\text{variables}}}$  puntos. Para este caso de 4 variables, el número total de diseños evaluados en el diseño de experimentos inicial asciende a  $3^4$ , es decir, 81 diseños diferentes. Estos diseños surgen de todas las combinaciones de los valores extremos y el valor central de cada variable:

- $L \rightarrow 20, 40, 60\text{mm}$
- $e \rightarrow 3, 5.5, 8\text{mm}$
- $e_h \rightarrow 3, 5.5, 8\text{mm}$
- $h \rightarrow 20, 40, 60\text{mm}$

Una vez simulados estos 81 casos con elementos finitos, se estimó el resultado de 10 nuevos diseños o puntos aleatorios del dominio empleando los diferentes metamodelos de aproximación entrenados a partir de los 81 puntos iniciales. Para interpretar los resultados se calculó el error medio absoluto (EMA) y el error porcentual medio absoluto (EPMA) para cada metamodelo, tanto para la masa como para el desplazamiento:

$$EMA = \frac{\sum_{i=1}^{n^{\text{ptos}}} |\text{valor}_{\text{simulado}} - \text{valor}_{\text{estimado}}|}{n^{\text{ptos}}}$$

$$EPMA = \frac{\sum_{i=1}^{n^{\text{ptos}}} |\text{valor}_{\text{simulado}} - \text{valor}_{\text{estimado}}| \cdot \frac{100}{\text{valor}_{\text{simulado}}}}{n^{\text{ptos}}}$$

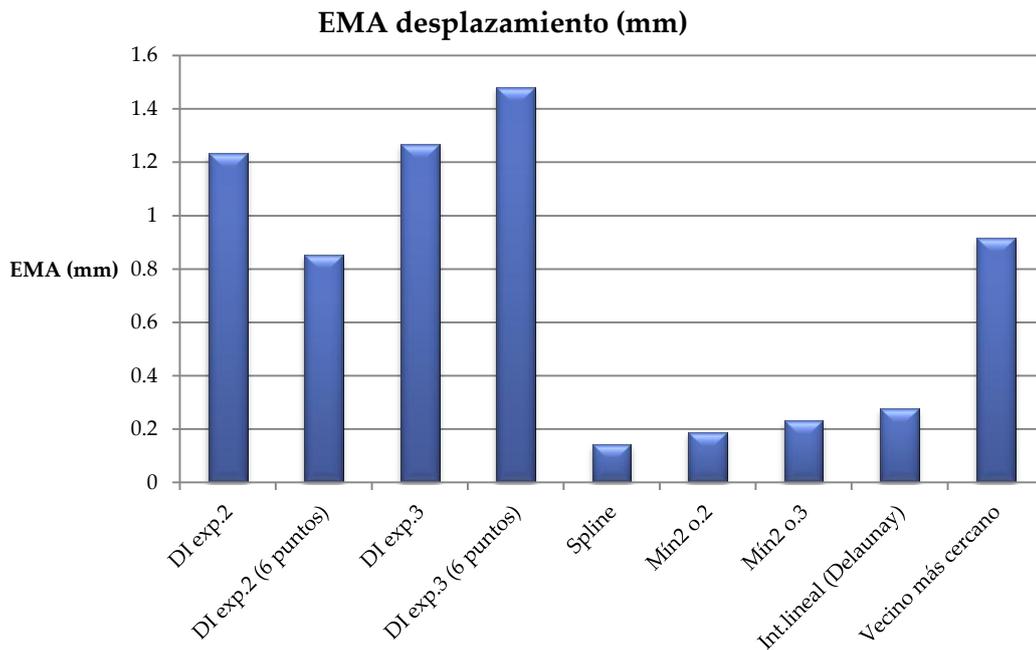
Los valores obtenidos para cada caso se aprecian en la siguiente tabla:

	EMA desplazamiento (mm)	EPMA desplazamiento (%)	EMA masa (g)	EPMA masa (%)
DI exp.2	1.233	8.214	213.057	12.903
DI exp.2 (6 puntos)	0.852	5.780	91.291	5.314
DI exp.3	1.269	8.450	211.353	12.708
DI exp.3 (6 puntos)	1.482	9.765	224.261	13.280

Spline	0.143	0.965	7.347	0.421
Mín2 o.2	0.187	1.271	11.773	0.670
Mín2 o.3	0.230	1.605	14.505	0.850
Int.lineal (Delaunay)	0.275	1.921	13.973	0.746
Vecino más cercano	0.916	6.536	155.015	8.208

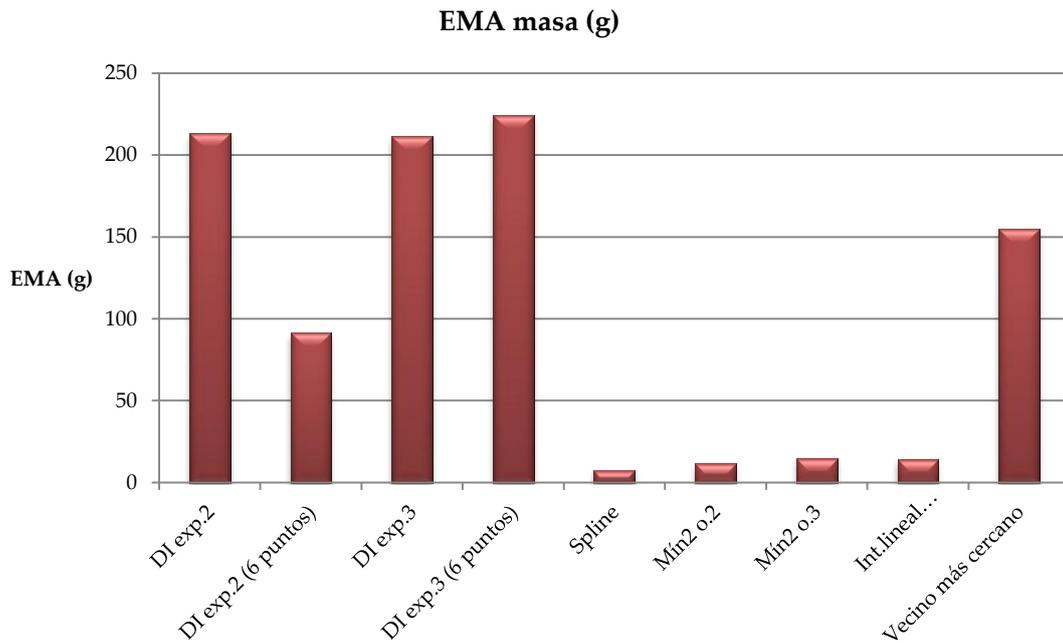
Tabla 1. Resultados de error de los diferentes metamodelos evaluados.

Estos resultados se recogen en las siguientes gráficas para facilitar su interpretación:



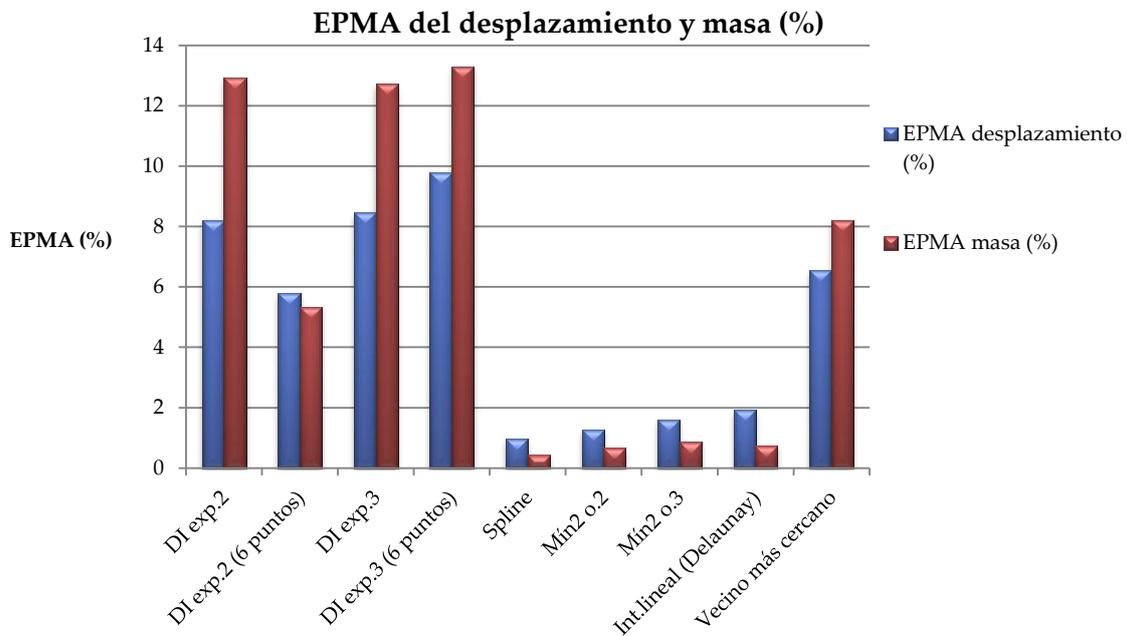
Gráfica 1. Error medio absoluto (EMA) de desplazamiento de los diferentes metamodelos.

Se observa que la spline, mínimos cuadrados de orden 2, mínimos cuadrados de orden 3 y la interpolación lineal de Delaunay estiman los resultados de desplazamiento con un error medio absoluto (EMA) significativamente inferior al resto de metamodelos. Entre estos 4 metamodelos, la mayor precisión en la estimación se consigue con la spline, seguida por los mínimos cuadrados de orden 2, mínimos cuadrados de orden 3 e interpolación lineal de Delaunay respectivamente. También cabe destacar que el hecho de incluir sólo 6 puntos para la estimación mediante la distancia inversa ponderada produce mejoras en el caso de usar exponente 2 y perjudica en el caso de exponente 3.



Gráfica 2. Error medio absoluto (EMA) de masa de los diferentes metamodelos.

Las conclusiones que se pueden extraer de la gráfica de error medio absoluta para la masa son prácticamente las mismas que en el caso del desplazamiento. Únicamente cabría destacar que en este caso la interpolación lineal de Delaunay mejora levemente con respecto a mínimos cuadrados de orden 2.



Gráfica 3. Error porcentual medio absoluto (EPMA) de desplazamiento y masa.

De esta última gráfica se puede concluir nuevamente que la spline, mínimos cuadrados (de orden 2 y 3) y la interpolación lineal son los metamodelos que proporciona unas estimaciones más precisas. También cabe destacar que estos 4 metamodelos consiguen una mayor precisión en las estimaciones de la masa que en los desplazamientos, mientras que por norma general en el resto de metamodelos sucede lo contrario.

A pesar de que la spline se muestra como la mejor opción en cuanto a precisión de los resultados, su aplicación está muy limitada por el hecho de que requiere de una distribución tipo rejilla de los datos para construir el modelo de aproximación. Esto dificultaría la idea de añadir nuevos puntos próximos a la frontera entre espacio factible y no factible para acercarse al diseño óptimo, tanto en la fase de diseño de experimentos como en el refinamiento del metamodelo de aproximación.

La única alternativa para poder aplicar un refinamiento localizado empleando splines sería mediante las T-splines. Como se comentó anteriormente, esta nueva modalidad permite un refinamiento localizado mediante la subdivisión de los cuadrantes de la rejilla de datos. Sin embargo, si se pretende añadir un nuevo punto en un lugar concreto del espacio de trabajo y emplear dicho punto para actualizar el metamodelo, entonces será necesario añadir todos los puntos que se requieren para la descomposición recursiva del espacio ("quadtree") hasta aproximarse al punto deseado.

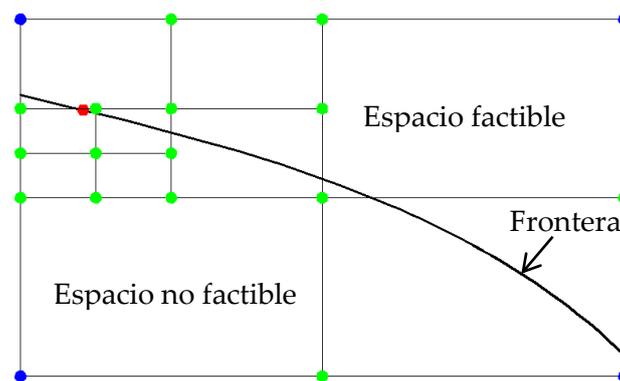


Ilustración 20. Refinamiento de la T-spline ("quadtree").

Suponiendo que con los cuatro puntos iniciales (azul) se estima que el diseño "óptimo" está localizado donde se encuentra el punto rojo, entonces convendría simular dicho punto por elementos finitos e incluirlo en la nube de puntos para actualizar el metamodelo y estimar de nuevo (con este metamodelo más preciso) la localización de un "óptimo" más cercano al verdadero. Sin embargo, dado que las T-

spline requieren una distribución de datos tipo “quadtree” (28), no es posible incluir directamente el punto rojo para rehacer el metamodelo, sino que se tendrían que añadir todos los puntos necesarios para ir subdividiendo el espacio (puntos en verde). Esto complicaría bastante el refinamiento del metamodelo y supondría un muestro muy intenso. Además, en el caso de problemas de más de 3 dimensiones (como el ejemplo de referencia de 4 variables de diseño) estos problemas se acentúan.

Por estas razones se descartó la opción de usar splines como metamodelo de aproximación.

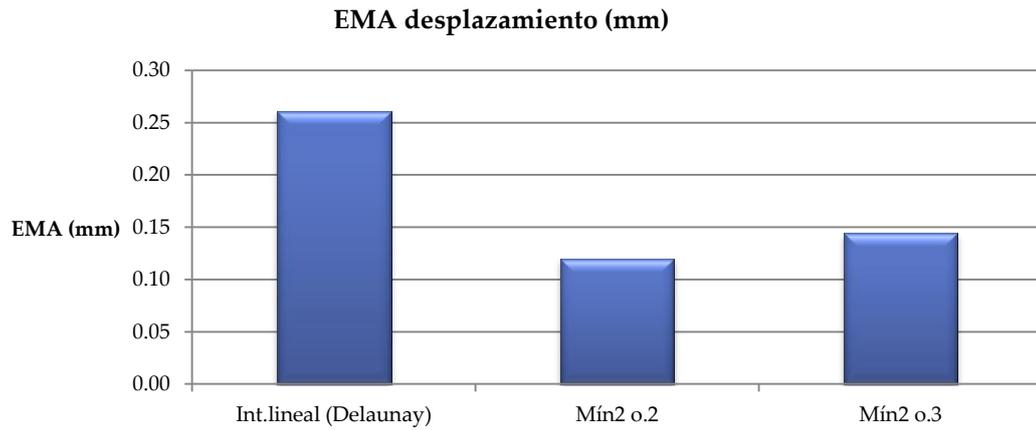
Posteriormente se realizó otra comparación entre mínimos cuadrados de orden 2, mínimos cuadrados de orden 3 e interpolación lineal (basada en la triangulación de Delaunay) en una situación más real del problema. Para ello se empleó el mismo ejemplo de referencia pero con un muestreo inicial de 33 puntos siguiendo un diseño de experimentos planteado durante las primeras fases de desarrollo (se explicará en capítulos posteriores). Igualmente, los puntos a estimar no fueron tomados aleatoriamente sino siguiendo esa metodología propuesta (aproximación a la frontera factible-no factible), con un total de 16 puntos.

Los resultados se resumen en la siguiente tabla:

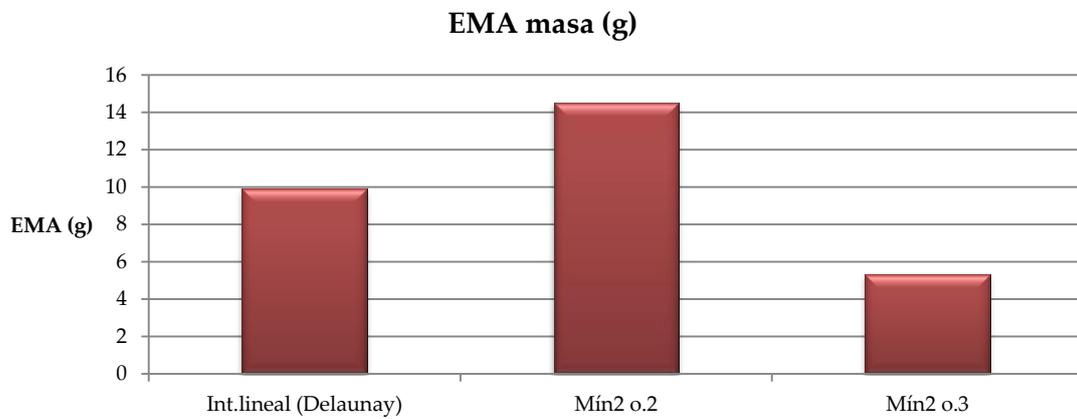
	EMA desplazamiento (mm)	EPMA desplazamiento (%)	EMA masa (g)	EPMA masa (%)
Int.lineal (Delaunay)	0.261	1.753	9.909	0.526
Mín2 o.2	0.120	0.795	14.509	0.803
Mín2 o.3	0.144	0.964	5.328	0.290

Tabla 2. Resultados de error de los 3 metamodelos evaluados.

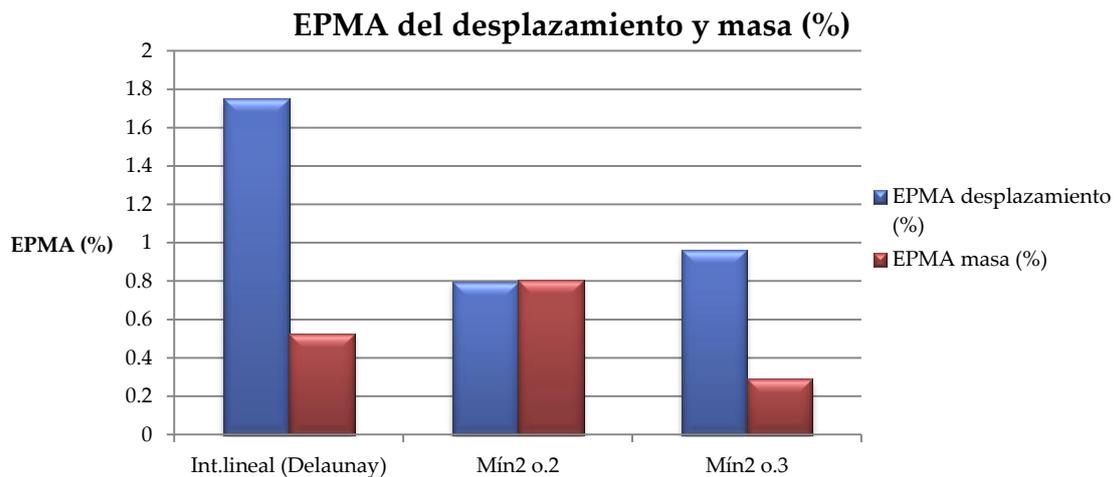
Estos resultados están representados en las siguientes gráficas. En ellas se puede observar que los mínimos cuadrados tanto de orden 2 como de orden 3 tienen unos resultados similares. La ecuación de orden 3 se comporta mejor para la estimación de la masa, pero para los desplazamientos la ecuación de orden 2 da resultados más precisos. La interpolación lineal estima buenos resultados de masa, sólo superada por mínimos cuadrados de orden 3, pero comete más error en el desplazamiento que los dos casos de ajuste por mínimos cuadrados.



Gráfica 4. Error medio absoluto (EMA) de desplazamiento de los 3 metamodelos.



Gráfica 5. Error medio absoluto (EMA) de masa de los 3 metamodelos.



Gráfica 6. Error porcentual medio absoluto (EPMA) de desplazamiento y masa de los 3 metamodelos.

A pesar de que estos 3 metamodelos ofrecen unos resultados aparentemente similares, se optó por emplear interpolación lineal basada en la triangulación de Delaunay. Esta decisión se debe a que en los modelos de regresión existe una mayor dificultad para estimar los resultados con precisión cuando se intensifica el muestreo en una zona concreta. Los mínimos cuadrados ajustan los parámetros de la ecuación de ajuste para minimizar el error cuadrático medio en los puntos de muestreo, pero no garantizan que la superficie de respuesta incluya dichos puntos por la “rigidez” de la ecuación de ajuste. Además, la superficie de respuesta se puede distorsionar por otras zonas cuando se intensifica el muestreo en una parte concreta del dominio por esta “rigidez” de la ecuación.

Sin embargo en la interpolación lineal a medida que se intensifica el muestreo en una zona los resultados son cada vez más precisos, ya que cuanto más cerca esté el punto a estimar de un punto de muestreo, menor será el error en la estimación.

## 6. Desarrollo del algoritmo genético de optimización

### 6.1. Algoritmos genéticos

Los algoritmos evolutivos son métodos de búsqueda basados en la evolución de los seres vivos y empleados normalmente para la resolución de problemas de optimización. Están inspirados en cierto modo en la teoría de la evolución de Darwin, con una competición entre seres vivos por los recursos limitados del entorno en el que conviven, dando lugar a que sobreviven o permanecen más en el tiempo (con más probabilidad de generar descendencia) los seres más adaptados a su entorno natural.

En cada iteración de un algoritmo evolutivo hay determinadas operaciones que se realizan o no bajo la consideración de probabilidades, por lo que se trata de un método heurístico de búsqueda (no determinista).

Entre los algoritmos evolutivos se distinguen las estrategias evolutivas, la programación evolutiva, la programación genética y los algoritmos genéticos. Estos últimos constituyen el paradigma más completo de los algoritmos evolutivos ya que resumen todas las ideas fundamentales de la evolución natural.

A groso modo, un algoritmo genético (AG) es un algoritmo inspirado en una mayor probabilidad de supervivencia y de creación de descendientes de los seres más adaptados al entorno, es decir, de los mejores en su evaluación de la función objetivo. Consta de operadores de selección, cruce y mutación que van dando lugar a las sucesivas generaciones de individuos, convergiendo éstas paulatinamente hasta el individuo óptimo con el paso de las generaciones. También es habitual aplicar elitismo (mantener el mejor individuo en todas las generaciones) para mejorar la eficiencia.

Su principal virtud es que la convergencia del método no está vinculada a requisitos de continuidad, diferenciabilidad, etc. y que no se atasca en mínimos o máximos locales, como ocurre en otros métodos de optimización deterministas. Además también cabe destacar su simplicidad, robustez, flexibilidad y su habilidad de adaptación en el proceso de búsqueda (30).

Sin embargo no se trata de un método sustitutivo que se considere mejor que otros (“No free lunch”), sino que normalmente se combina con métodos de optimización deterministas que son más efectivos en la optimización local, aprovechando por tanto la capacidad de los algoritmos genéticos para buscar el óptimo global y finalmente se refina el resultado con optimización determinista.

La secuencia de pasos de un algoritmo genético es la siguiente:

- 1) Creación de la población inicial de individuos (aleatoriamente).
- 2) Evaluación de la función objetivo de cada individuo de la población.
- 3) Mientras no se cumplan las condiciones de parada:
  - a. Seleccionar probabilísticamente los mejores individuos creando una población auxiliar.
  - b. Cruzar probabilísticamente algunos individuos de esta población auxiliar y almacenar sus hijos en la nueva población. Almacenar también, en la nueva población, los individuos de la población auxiliar que no fueron cruzados.
  - c. Mutación probabilística de algunos individuos de la nueva población.
  - d. Reparación de los individuos que se salen del dominio.
  - e. Evaluación de la función objetivo de cada individuo de la nueva población.
  - f. Elitismo (reemplazar el peor individuo por el mejor hasta el momento).
  - g. Evaluación de las condiciones de parada.

Para garantizar la convergencia de un algoritmo genético es fundamental encontrar un equilibrio entre la explotación (o presión de selección) y la exploración (o diversidad de la población) evitando así la convergencia prematura. Si en las primeras generaciones de un algoritmo genético se encuentra un individuo más apto que el resto, una presión de selección elevada propiciará una reproducción tan abundante de este individuo que la diversidad de la población se verá mermada y el algoritmo se atascará en un mínimo local. Sin embargo una excesiva exploración frente a una baja presión de selección supondría que la convergencia sea muy lenta, por lo que se necesitarían más generaciones y por tanto más tiempo de cálculo para alcanzar el óptimo.

Este equilibrio se establece ajustando los distintos parámetros que intervienen en el proceso, como el tamaño de la población, el tipo de selección y cruce o las probabilidades de cruce y mutación. A mayor tamaño de población, probabilidad de cruce o probabilidad de mutación, mayor será entonces la diversidad y por ende menor la presión de selección. Igualmente, a menor tamaño de población, probabilidad de cruce o mutación, menor será la diversidad, aumentando por tanto la presión de selección.

## 6.2. Implementación del algoritmo genético

El siguiente paso en el desarrollo de la metodología fue crear el código de un algoritmo genético en Matlab. Para comprobar el correcto funcionamiento del mismo se evaluó con un problema típico de optimización sin incluir metamodelo de aproximación, diseño de experimentos inicial y refinamiento.

### 6.2.1. Problema de referencia

Se decidió usar el ejemplo de referencia anterior para seguir en la misma línea de trabajo, pero esta vez evitando las simulaciones por elementos finitos con la finalidad de agilizar la ejecución de las diferentes pruebas y depuración de errores en el código. Para prescindir de las simulaciones, la evaluación de la función objetivo de cada individuo se calculó empleando una ecuación polinómica de orden 2 obtenida mediante ajuste por mínimos cuadrados a partir de una nube de 49 puntos simulados previamente con elementos finitos. La distribución de esta nube de puntos se corresponde con la primera propuesta de diseño de experimentos planteada en este trabajo, que se comentará en el capítulo siguiente.

Las ecuaciones obtenidas en el ajuste por mínimos cuadrados para evaluar el desplazamiento (mm) y la masa (g) de un punto "p" fueron las siguientes:

$$\begin{aligned} \text{desplazamiento} = & 24.1317 + 0.0632925 \cdot p(1) - 3.11677 \cdot p(2) - 1.19797 \cdot p(3) + 0.0789463 \cdot p(4) \\ & + 0.0000104292 \cdot p(1)^2 - 0.0112888 \cdot p(1) \cdot p(2) + 0.00461183 \cdot p(1) \cdot p(3) \\ & + 0.000164433 \cdot p(1) \cdot p(4) + 0.209078 \cdot p(2)^2 + 0.0681854 \cdot p(2) \cdot p(3) - 0.00248343 \\ & \cdot p(2) \cdot p(4) + 0.035974 \cdot p(3)^2 + 0.000917187 \cdot p(3) \cdot p(4) - 0.000846009 \cdot p(4)^2 \end{aligned}$$

$$\begin{aligned} \text{masa} = & 1135.34 - 28.2223 \cdot p(1) + 225.325 \cdot p(2) + 264.993 \cdot p(3) - 13.6714 \cdot p(4) + 0.245105 \cdot p(1)^2 \\ & + 1.2674 \cdot p(1) \cdot p(2) - 0.854859 \cdot p(1) \cdot p(3) - 0.028917 \cdot p(1) \cdot p(4) - 6.06156 \\ & \cdot p(2)^2 - 11.0138 \cdot p(2) \cdot p(3) + 0.620936 \cdot p(2) \cdot p(4) - 8.72012 \cdot p(3)^2 - 0.403534 \cdot p(3) \\ & \cdot p(4) + 0.129436 \cdot p(4)^2 \end{aligned}$$

$$p(j) = \text{valor de la variable "j"} \quad (j=1,2,3,4)$$

Una vez conocidas las ecuaciones para el problema de referencia, se calculó el diseño óptimo mediante una función de optimización disponible en el Matlab (basada en el gradiente):

**Óptimo teórico: 44.458 4.386 3.000 60.000**

**Masa: 1600.809g**

**Desplazamiento: 15.000mm**

Como se aprecia, el diseño óptimo tiene una masa de 1600.809g (será el valor de referencia que se tendrá para las diferentes pruebas), con un desplazamiento justo al límite de la restricción (15mm, como cabía esperar). Estos valores se consiguen con un individuo definido por los cromosomas [44.458 4.386 3.000 60.000], que físicamente se interpretan como " $L=44.458mm$ ", " $e=4.386mm$ ", " $e_i=3.000mm$ " y " $h=60.000mm$ ". Por tanto, las soluciones del algoritmo genético deben aproximarse a este óptimo "teórico".

### 6.2.2. Pseudocódigo inicial

El pseudocódigo de partida se resume en los siguientes puntos (en **negrita** se incluyen los parámetros que se fueron modificando en las múltiples pruebas para ajustar correctamente el algoritmo):

- 1) Población inicial aleatoria de 100 individuos
- 2) Cálculo de la función objetivo (F) de cada individuo con las ecuaciones anteriores:  
 $FA = \text{Factor de Amplificación para la penalización} = 10^{99}$   
Si desplazamiento > 15  $\Rightarrow$  **Penalización** = (desplazamiento - 15)<sup>2</sup> · FA  
 $F = \text{masa} + \text{Penalización}$
- 3) Ordenación de la población según la función objetivo de cada individuo y almacenamiento del "mejor".
- 4) Mientras número de generaciones < **número máximo de generaciones**:
  - a) Actualización del número de generaciones.
  - b) Selección por torneo 2 a 2: se escogen 2 individuos aleatoriamente y se almacena el mejor de ellos en la población auxiliar (100 individuos).
  - c) Selección de los individuos que cruzan con un 80% de **probabilidad de cruce** (se escogen 80 individuos aleatoriamente de los 100 que componen la población auxiliar para aplicar cruce).
  - d) Cruce antitético de 2 individuos aleatorios (padre y madre) de los escogidos para el cruce, con valor de  $\alpha$  aleatorio entre (-0.5~1.5) y posterior almacenamiento de los hijos en la nueva población.  
 $Hijo1 = \alpha \cdot \text{padre} + (1-\alpha) \cdot \text{madre}$   
 $Hijo2 = (1-\alpha) \cdot \text{padre} + \alpha \cdot \text{madre}$
  - e) Almacenamiento del resto de individuos de la población auxiliar que no cruzan (100-80=20 individuos) en la nueva población.
  - f) Mutación de cada individuo de la nueva población con una **probabilidad de mutación** del 10%. Selección aleatoria de la variable mutada (o cromosoma mutado de la cadena de cromosomas), y modificación de la misma con un incremento aleatorio comprendido entre  $\pm(0\sim 10\%)$  del intervalo de trabajo de la variable mutada, siendo en este caso el 10% la **amplitud de la mutación**.
  - g) Reparación de los individuos de la nueva población que se hayan salido del dominio establecido.

- h) Cálculo de la función objetivo (F) de cada individuo:  
**FA**=Factor de Amplificación para la penalización=10<sup>99</sup>  
Si desplazamiento>15 ⇒ **Penalización**=(desplazamiento-15)<sup>2</sup>·**FA**  
**F**=masa+**Penalización**
- i) Ordenación de la nueva población según función objetivo.
- j) Sustitución del peor individuo de la población por el “mejor” de la generación anterior (elitismo).
- k) Reordenación de la nueva población según función objetivo y almacenamiento del “mejor”.

Los parámetros empleados para ajustar el algoritmo genético se resumen en los siguientes:

- **FA**=Factor de Amplificación para la penalización
- **Penalización**=tipo de penalización empleada por incumplimiento de las restricciones
- **Número máximo de generaciones**
- **Probabilidad de cruce**
- **Probabilidad de mutación**
- **Amplitud de mutación**

### 6.2.3. Ajuste de los parámetros del algoritmo genético

Tras crear el código en Matlab del algoritmo genético para el problema de referencia y realizar la depuración del mismo, se procedió a ajustar los parámetros anteriores con la finalidad de garantizar la convergencia y correcto funcionamiento del algoritmo. Como se comentó con anterioridad, es importante encontrar un equilibrio entre presión de selección y diversidad de población. Una configuración más inclinada hacia la diversidad de población implica un mayor número de generaciones y tiempo de cómputo para alcanzar el óptimo, lo cual podría ser inviable en algunas aplicaciones. Sin embargo, en el contexto de este trabajo hay que recordar que se emplearán metamodelos de aproximación para evaluar la función objetivo de cada individuo y así evitar las costosas simulaciones por elementos finitos. Esto implica que se puede optar por una configuración del algoritmo genético con mayor tendencia a la diversidad aunque suponga aumentar el número de generaciones, ya que la evaluación de la función objetivo no será excesivamente costosa gracias a los metamodelos de aproximación.

### Prueba 1

La primera prueba se realizó con la configuración inicial y un total de 50 generaciones.

- **FA**=10<sup>99</sup>
- **Penalización**=(desplazamiento-15)<sup>2</sup>-FA
- **Número máximo de generaciones**=50
- **Probabilidad de cruce**=0.8 (80%)
- **Probabilidad de mutación**=0.1 (10%)
- **Amplitud de mutación**=10%

El diseño óptimo al que llega el algoritmo genético en diferentes ejecuciones con esta configuración se resume en la siguiente tabla:

Ejecución	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
1	45.090	4.400	3.000	60.000	15.000	1600.913
2	45.458	4.491	3.000	51.212	15.000	1611.606
3	47.641	4.456	3.000	60.000	15.000	1603.359
4	42.445	4.338	3.013	60.000	15.000	1602.418
5	52.863	4.465	3.503	60.000	15.000	1638.913
6	43.152	4.450	3.004	42.269	14.995	1625.448
7	42.937	4.259	3.000	26.669	15.000	1655.245
8	46.243	4.438	3.000	59.107	14.999	1602.706
9	45.203	4.465	3.038	53.108	14.999	1610.918
10	44.808	43.951	3.000	60.000	14.998	1601.102

Tabla 3. Resultados obtenidos en la prueba 1.

Se observa que los resultados se aproximan bastante a la masa óptima aunque dependiendo de la ejecución, el resultado varía considerablemente, por lo que sería conveniente ajustar los parámetros para garantizar que el algoritmo converja de manera más exacta al óptimo “teórico”.

### Prueba 2

La siguiente prueba se realizó con un factor de amplificación (FA) variable con el número de generaciones. La finalidad de esta prueba es tratar de penalizar menos el incumplimiento de las restricciones en las primeras generaciones del algoritmo,

dotando de mayor flexibilidad y libertad en la evolución de la población en las primeras fases de evolución. El factor de amplificación en este caso aumenta exponencialmente con el número de generación ( $10^{n^{\circ}}$  de generación).

- **FA**= $10^{n^{\circ}}$  de generación
- **Penalización**=(desplazamiento-15)<sup>2</sup>·FA
- **Número máximo de generaciones**=50
- **Probabilidad de cruce**=0.8 (80%)
- **Probabilidad de mutación**=0.1 (10%)
- **Amplitud de mutación**=10%

El hecho de tener una penalización tan débil en las primeras generaciones implica que el mejor individuo de las primeras generaciones tenga un valor de su función objetivo casi sin penalización. Por tanto, tal y como está programado el algoritmo genético, el mejor individuo final será uno que probablemente no cumpla las restricciones por obtenerse en las primeras generaciones donde la penalización era muy baja. Para evitar esto habrá que recalcular de nuevo la función objetivo del mejor individuo de la población anterior y así tener todos los individuos con una función aptitud penalizada bajo el mismo criterio.

Los resultados obtenidos se resumen en la siguiente tabla:

Ejecución	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
1	49.150	4.489	3.001	60.000	14.999	1606.521
2	41.254	4.398	3.001	51.645	14.998	1613.452
3	50.866	4.567	3.000	56.498	14.997	1615.630
4	45.946	4.522	3.010	47.089	14.983	1620.511
5	43.013	4.295	3.553	52.059	15.000	1636.363
6	48.268	4.540	3.000	52.941	15.000	1612.978
7	47.030	4.431	3.158	57.988	14.999	1611.888
8	46.593	4.499	3.000	53.992	14.997	1609.430
9	44.477	4.387	3.000	59.998	15.000	1600.811
10	44.633	4.485	3.002	43.897	14.998	1622.055

Tabla 4. Resultados obtenidos en la prueba 2.

Como se aprecia la solución óptima sigue variando considerablemente dependiendo de la ejecución.

### Prueba 3

La tercera prueba se llevó a cabo bajo las mismas condiciones anteriores pero con un total de 100 generaciones en vez de 50 generaciones.

- $FA=10^n$  de generación
- **Penalización**=(desplazamiento-15)<sup>2</sup>-FA
- **Número máximo de generaciones**=100
- **Probabilidad de cruce**=0.8 (80%)
- **Probabilidad de mutación**=0.1 (10%)
- **Amplitud de mutación**=10%

Ejecución	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
1	44.391	4.385	3.000	60.000	15.000	1600.811
2	42.327	4.388	3.000	55.778	15.000	1606.627
3	42.462	4.322	3.155	58.788	14.995	1610.976
4	40.856	4.305	3.000	60.000	15.000	1604.078
5	44.451	4.386	3.000	60.000	15.000	1600.808
6	47.189	4.507	3.000	54.308	15.000	1609.358
7	50.069	4.565	3.000	54.775	14.998	1615.215
8	43.614	4.367	3.000	60.000	15.000	1600.996
9	43.537	4.410	3.000	56.313	15.000	1605.114
10	41.644	4.386	3.002	54.257	15.000	1609.370

Tabla 5. Resultados obtenidos en la prueba 3.

Se aprecia a simple vista que los resultados mejoran al aumentar el número de generaciones, como cabía esperar. Sin embargo el algoritmo a veces converge a una solución algo distanciada del óptimo “teórico” en términos de masa. También se podría afirmar que existen distintas combinaciones de variables que ofrecen resultados muy similares como ocurre por ejemplo con la ejecución número 2 y la número 4. A pesar de que los resultados obtenidos son muy similares, los cromosomas varían considerablemente entre ambos.

### Prueba 4

Se emplearon de nuevo 50 generaciones pero con una probabilidad de mutación que varía progresivamente desde 0 en la primera generación hasta un 60% en la última

generación, de modo que en las últimas generaciones habrá bastante mutación y esto ayudará a que el algoritmo no se estanque fácilmente. La amplitud de mutación también aumenta progresivamente hasta un 50% en la última generación, por lo que la mutación puede ser más agresiva en las últimas generaciones. La penalización en este caso se obtiene como la diferencia en valor absoluto (antes era la diferencia al cuadrado) para penalizar más a los individuos que están al límite de cumplir las restricciones. Por ejemplo un individuo con desplazamiento de 15.001mm tendrá una penalización de  $0.001^2$  (es decir, 0.000001) con su amplificación correspondiente. Si esta amplificación no es elevada, entonces el individuo apenas será penalizado. En el caso de emplear la diferencia en valor absoluto, la penalización del individuo será de 0.001 (1000 veces mayor) con su amplificación, por lo que se aplicará una mayor penalización. El factor de amplificación permanece igual que en el caso anterior ( $10^{n^{\text{de}}}$  generación) y, por último, destacar que se aplicará elitismo a partir de la generación número 11.

- **FA**= $10^{n^{\text{de}}}$  generación
- **Penalización**= $|\text{desplazamiento}-15| \cdot \text{FA}$
- **Número máximo de generaciones**=50
- **Probabilidad de cruce**=0.8 (80%)
- **Probabilidad de mutación**= $0.6 \cdot n^{\text{de}} \text{generación} / N^{\text{o}} \text{máx. generaciones}$  (0~60%)
- **Amplitud de mutación**= $0.5 \cdot n^{\text{de}} \text{generación} / N^{\text{o}} \text{máx. generaciones}$  (0~50%)

Ejecución	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
1	43.792	4.372	3.000	60.000	14.999	1601.032
2	46.344	4.429	3.001	60.000	14.997	1602.082
3	48.537	4.476	3.000	60.000	14.999	1605.160
4	44.228	4.383	3.000	60.000	14.997	1601.240
5	44.233	4.382	3.000	60.000	14.999	1600.959
6	44.015	4.377	3.000	60.000	14.999	1600.986
7	43.133	4.361	3.000	60.000	14.993	1602.186
8	43.155	4.357	3.000	59.998	15.000	1601.289
9	45.102	4.402	3.000	60.000	14.997	1601.312
10	44.466	4.386	3.000	60.000	15.000	1600.844

Tabla 6. Resultados obtenidos en la prueba 4.

Los resultados obtenidos con esta configuración son bastante buenos y muy similares entre ellos, lo cual es síntoma de que el algoritmo genético está bien ajustado.

A pesar de que esta configuración podría ser válida, se probaron más combinaciones para contrastar los resultados.

### Prueba 5

Esta prueba se llevó a cabo con 50 generaciones, probabilidad de cruce del 50%, probabilidad de mutación del 60% y amplitud de mutación de un 50%. La penalización se realizó mediante la diferencia al cuadrado y con factor de amplificación de  $10^{99}$ .

- **FA**= $10^{99}$
- **Penalización**=(desplazamiento-15)<sup>2</sup>·FA
- **Número máximo de generaciones**=50
- **Probabilidad de cruce**=0.5 (50%)
- **Probabilidad de mutación**=0.6 (60%)
- **Amplitud de mutación**=0.5 (50%)

Ejecución	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
1	45.343	4.411	3.000	60.000	14.991	1602.252
2	43.766	4.372	3.000	60.000	14.998	1601.269
3	42.116	4.347	3.000	59.277	14.994	1603.779
4	44.791	4.394	3.000	60.000	14.998	1601.047
5	44.195	4.393	3.000	59.046	15.000	1601.878
6	47.444	4.454	3.000	60.000	14.997	1603.489
7	44.477	4.399	3.000	59.103	14.999	1601.869
8	43.959	4.375	3.000	60.000	15.000	1600.872
9	44.342	4.386	3.000	60.000	14.995	1601.428
10	43.446	4.364	3.000	60.000	15.000	1601.100

Tabla 7. Resultados obtenidos en la prueba 5.

Nuevamente los resultados son bastante buenos e incluso se podría decir que algo más estables que en el caso anterior (la peor ejecución de esta prueba obtiene una masa de 1603.779g, mientras que en la prueba anterior el peor caso fue de 1605.16g). Una alta probabilidad de mutación ayuda a la exploración local y evita que el algoritmo converja a resultados alejados del óptimo. Por ello estas dos últimas pruebas han dado mejores resultados que las anteriores.

### Prueba 6

Esta última prueba se realizó en las mismas condiciones anteriores exceptuando una probabilidad de mutación mayor (80%) y menor amplitud de la mutación (10%).

- $FA=10^{99}$
- **Penalización**=(desplazamiento-15)<sup>2</sup>-FA
- **Número máximo de generaciones**=50
- **Probabilidad de cruce**=0.5 (50%)
- **Probabilidad de mutación**=0.8 (80%)
- **Amplitud de mutación**=0.1 (10%)

Ejecución	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
1	42.884	4.356	3.000	60.000	14.992	1602.543
2	43.361	4.362	3.000	60.000	15.000	1601.121
3	43.731	4.370	3.000	60.000	15.000	1600.992
4	45.602	4.412	3.000	60.000	14.999	1601.292
5	44.106	4.379	3.000	60.000	14.999	1600.988
6	44.267	4.383	3.000	60.000	14.998	1601.085
7	41.008	4.312	3.000	60.000	14.995	1604.502
8	43.364	4.363	3.000	60.000	14.998	1601.451
9	46.270	4.426	3.000	60.000	15.000	1601.707
10	45.720	4.418	3.000	60.000	14.993	1602.154

Tabla 8. Resultados obtenidos en la prueba 6.

Los resultados siguen siendo buenos. Esto demuestra que existen múltiples configuraciones que podrían ser válidas para llegar al óptimo. A pesar de que aparentemente con 50 generaciones podría ser suficiente (en la última ejecución el óptimo fue obtenido en la generación número 44), para garantizar la convergencia en el programa final se emplearán más generaciones, ya que el coste computacional será despreciable frente a las simulaciones por elementos finitos.

## 7. Implementación del programa. Pruebas y mejoras.

En este capítulo se muestran las pruebas más significativas que se han llevado a cabo para encauzar el trabajo hasta llegar a la metodología final de optimización. A pesar de que se han probado un número importante de programas diferentes, sólo se mencionan aquéllos que han sido de mayor importancia para elegir la metodología final, excluyendo múltiples pruebas correspondientes sobre todo a los primeros intentos.

Como se comentó en capítulos anteriores, los mejores diseños estarán justo en el límite entre el espacio factible y no factible, por lo que la estrategia para optimizar el diseño con el menor número de simulaciones posibles será añadiendo nuevos puntos en esta frontera. Por ello, todos los programas desarrollados en esta fase de pruebas se basan en esta estrategia.

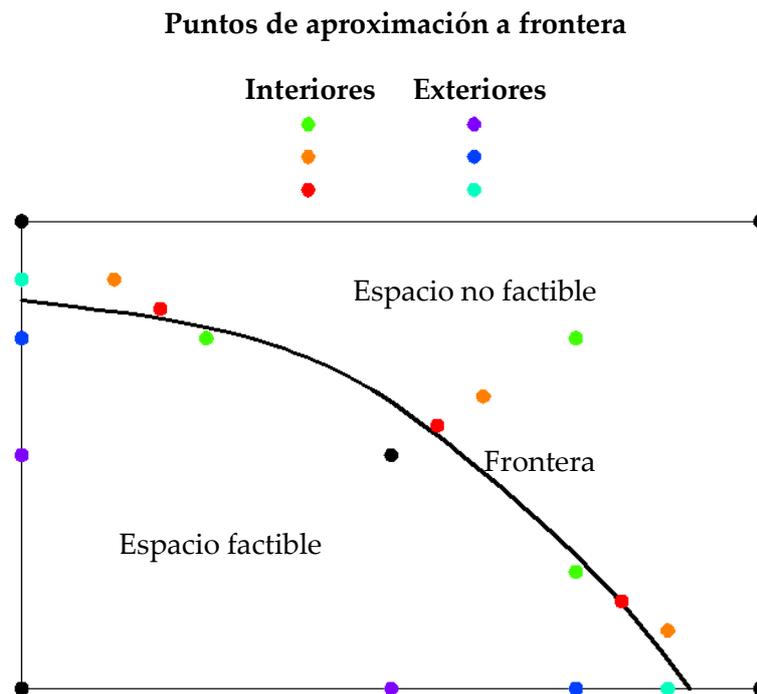
### 7.1. Programa inicial

La primera idea desarrollada consistió en aplicar un diseño de experimentos inicial, seguido de un algoritmo de adición de puntos para aproximación a la frontera y refinamiento del metamodelo, y por último la aplicación de un AG de optimización final con adición del mejor individuo de cada generación para mejorar el metamodelo a medida que evoluciona el algoritmo genético.

Como se comentó en capítulos anteriores, el hecho de emplear interpolación lineal basada en la triangulación de Delaunay implica que sólo se pueden estimar resultados dentro de la envolvente convexa de los puntos de muestreo. Por esta razón el muestro inicial elegido es el factorial de 2 niveles ( $2^n$ ), que equivale a evaluar todas las esquinas del dominio y así formar una envolvente convexa de todo el espacio de búsqueda. También se añade en el muestreo inicial el punto que se encuentra en el centro del dominio. Este punto ayudará a realizar una mejor triangulación en la interpolación y además servirá de referencia para ir añadiendo nuevos puntos en la fase de aproximación a la frontera. Por tanto, en la primera fase se evaluarán " $2^{n+1}$ " puntos, siendo ésta la estrategia de muestreo inicial empleada en todas las pruebas realizadas en este trabajo.

Tras realizar el muestreo inicial, se lleva a cabo la aproximación a la frontera mediante adición de puntos. Esta aproximación se realiza tanto por el interior como

por las aristas del dominio (exterior). Para la aproximación interior, simplemente se compara el punto central del muestreo inicial con el resto de puntos (esquinas del dominio) y se van añadiendo puntos intermedios sólo si uno de los dos puntos comparados cumple las restricciones y el otro no. La aproximación por el exterior (aristas del dominio) se realiza de manera análoga, pero en este caso comparando dos esquinas contiguas, de modo que se añadirá un nuevo punto entre ambas si una de ellas cumple todas las restricciones y la otra no. Al finalizar la adición de puntos de aproximación a la frontera por el interior y exterior, se evalúa el error cometido del metamodelo de aproximación de las diferentes restricciones y de la masa con respecto a las simulaciones. Mientras la desviación media de algunas de las restricciones o de la masa sea superior al valor establecido por el usuario en pantalla al ejecutar el programa, el bucle de adición de puntos para aproximación a la frontera se repite en un bucle hasta conseguir que el error medio de cada respuesta estimada sea inferior a los valores asignados, garantizando así que se obtiene un metamodelo con un cierto nivel de aproximación. En la siguiente imagen se representa esquemáticamente la estrategia de aproximación a la frontera.



**Ilustración 21. Adición de puntos intermedios entre factible/no factible para aproximación a frontera.**

Los puntos en negro se corresponden con el muestreo inicial ( $2^{n+1}$ ). Los puntos en verde y violeta están vinculados al primer punto interior y exterior respectivamente.

En cada nueva iteración se añaden nuevos puntos entre el último punto añadido y el más cercano que se encuentre en el la región opuesta del espacio (es decir, si el punto de colocación está en la zona factible, se asocia con uno del espacio no factible) y que se haya añadido en iteraciones anteriores a la última.

Finalmente se aplica un algoritmo genético para buscar el óptimo. Este algoritmo está basado en la configuración de la prueba 5 del capítulo anterior, pero con la diferencia de que la evaluación de la función aptitud se realiza a través del metamodelo de aproximación. Además, para mejorar el ajuste del metamodelo se “simula” el mejor individuo de cada generación (si no ha sido simulado previamente) y se incluye en la nube de datos para reconstruir el metamodelo.

El pseudocódigo de este programa sería el siguiente:

- 1) Muestreo inicial  $2^{n+1}$  (esquinas y punto central).
- 2) Aproximación a la frontera con la adición de nuevos puntos hasta que el metamodelo estime los resultados con una desviación media inferior al valor definido por el usuario para cada restricción y para la masa:
  - a) Puntos interiores: se añaden nuevos puntos entre el punto central y las esquinas que se encuentren en el espacio opuesto (factible/no factible). En las sucesivas iteraciones se añade un nuevo punto entre el último punto añadido y el punto más cercano del espacio opuesto de los creados en iteraciones anteriores al último punto añadido.
  - b) Puntos exteriores: se añaden nuevos puntos entre esquinas del dominio contiguas que se encuentren en espacios opuestos (factible/no factible). En las sucesivas iteraciones se añade un nuevo punto entre el último punto añadido y el punto más cercano del espacio opuesto de los creados en iteraciones anteriores al último punto añadido.
- 3) Algoritmo genético de optimización. La función aptitud se evalúa mediante interpolación lineal basada en triangulación de Delaunay. El mejor individuo de cada generación se simula (sólo si no ha sido evaluado anteriormente) para aumentar el número de puntos de entrenamiento del metamodelo. Por tanto el ajuste del metamodelo va mejorando a medida que evoluciona el AG.
- 4) Búsqueda del mejor individuo simulado (óptimo).

### 7.1.1. Resultados obtenidos

Tras ejecutar el programa se observa que en el muestreo inicial se evalúan 17 puntos ( $2^4+1$ ), como era esperado, mientras que en cada iteración de adición de puntos para la aproximación a la frontera se evalúan 6 puntos interiores y 10 exteriores, es decir, 16 puntos en cada iteración. Este bucle se repite tantas veces como sea necesario

hasta que los errores de estimación de las diferentes respuestas sean inferiores a los valores establecidos. En este caso se eligió una desviación máxima de estimación del desplazamiento de 0.1mm y de 2g para la masa. Con este margen de error de estimación, el número de simulaciones realizadas (puntos evaluados) en el muestreo inicial y aproximación a la frontera es de 65, por lo que el bucle de aproximación a la frontera se repitió 3 veces ( $17+3 \cdot 16=65$ ). Sin embargo, para desviaciones mayores como por ejemplo 0.5mm de desplazamiento y 10g de masa, el número de puntos necesarios disminuye a 49 puntos, ejecutándose el bucle en 2 ocasiones ( $17+2 \cdot 16=49$ ) (estos 49 puntos fueron los empleados para ajustar las ecuaciones polinómicas que sustituyen el cálculo por elementos finitos).

El resto de puntos añadidos en el programa se corresponden al AG final. Este valor dependerá del número máximo de generaciones así como de los aspectos probabilísticos intrínsecos al AG. Para las pruebas realizadas con este programa inicial se emplearon 100 generaciones y 500 generaciones.

En la siguiente tabla se recogen los diseños óptimos obtenidos en las 10 ejecuciones realizadas con el programa así como el número total de puntos simulados y el número de generaciones empleadas en el AG.

Ejecución	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Nº generaciones AG	Nº puntos simulados	Desplazamiento (mm)	Masa (g)
1	52.725	4.482	3.571	57.350	100	73	14.993	1645.371
2	60.000	4.721	3.000	60.000	100	68	14.985	1663.543
3	60.000	4.722	3.000	60.000	100	72	14.983	1663.879
4	60.000	4.714	3.000	60.000	100	74	14.997	1661.791
5	60.000	4.714	3.000	60.000	100	76	14.998	1661.752
6	60.000	4.713	3.000	60.000	500	89	15.000	1661.436
7	60.000	4.713	3.000	60.000	500	85	15.000	1661.443
8	59.106	4.714	3.000	58.620	500	96	14.995	1657.076
9	60.000	4.713	3.000	60.000	500	78	14.999	1661.504
10	60.000	4.713	3.000	60.000	500	86	14.999	1661.505

Tabla 9. Resultados obtenidos en 10 ejecuciones del programa inicial.

Se observa que tanto para 100 generaciones como para 500 generaciones el diseño óptimo de las diferentes ejecuciones no consigue acercarse a los 1600.809g de referencia (óptimo teórico), siendo lo frecuente obtener una masa de aproximadamente 1661g (masa máxima=2808.1g; masa mínima=1218.1g). No obstante, hay que decir que en alguna ocasión (poco frecuente) de otras muchas ejecuciones realizadas, el programa consigue un óptimo más cercano al teórico que en estos casos anteriores, por

lo que esto depende del carácter aleatorio del algoritmo genético. También se observa que el número total de puntos simulados aumenta cuando se incrementa el número de generaciones del algoritmo genético.

### 7.1.2. Óptimos obtenidos con AG a partir del metamodelo

Tras comprobar que el programa no consigue unos resultados del todo alentadores, se decidió ejecutar un algoritmo genético sobre el metamodelo de aproximación construido con los 65 puntos iniciales (muestreo y aproximación a la frontera). Con esta prueba simplemente se quiere valorar cuál es el óptimo según el metamodelo de aproximación. Los resultados se resumen en la siguiente tabla:

Ejecución	L (mm)	e (mm)	e <sub>s</sub> (mm)	h (mm)	Nº generaciones AG	Función aptitud
1	60.000	4.721	3.000	60.000	100	1663.159
2	60.000	4.722	3.000	60.000	100	1663.304
3	60.000	4.721	3.000	60.000	100	1663.143
4	60.000	4.721	3.000	60.000	500	1663.117
5	60.000	4.721	3.000	60.000	1000	1663.151

Tabla 10. Diseños óptimos según el metamodelo de aproximación entrenado con 65 puntos.

El óptimo obtenido es prácticamente idéntico en las diferentes ejecuciones, incluso aumentando el número de generaciones del AG (100, 500 y 1000). Como se observa, los valores coinciden prácticamente con los resultados anteriores, por lo que la simulación del mejor individuo de cada generación no aporta mejoras en el refinamiento del metamodelo. Sin embargo en algunas ejecuciones el programa consigue llegar a un diseño con menor masa (como por ejemplo la ejecución 1 con 1645.371g) debido a que la simulación del mejor individuo en las primeras generaciones ayuda a mejorar el ajuste del metamodelo. Así por ejemplo, si en la primera generación (individuos totalmente aleatorios) resulta que el mejor individuo según el metamodelo está localizado en las proximidades del óptimo teórico o de otro punto relativamente bueno, entonces el programa simulará dicho punto, actualizará el metamodelo, y en la evolución del AG final obtendrá un diseño con una masa menor de 1663g. Por esta razón el programa converge en alguna ocasión poco frecuente a diseños significativamente mejores.

A pesar de que la idea inicial era tratar de mejorar el ajuste del metamodelo con la adición de puntos próximos a la frontera y con la simulación del mejor individuo de

cada generación, se concluye tras estas primeras pruebas que la simulación de los mejores individuos de cada población no implica mejoras sustanciales en el metamodelo y por tanto la metodología planteada no converge hacia el óptimo.

### 7.1.3. Adición de nuevos puntos antes del AG para mejorar el metamodelo

Para tratar de mejorar este primer programa se pensó en añadir más puntos antes de aplicar el AG y mejorar así el ajuste del metamodelo con la intención de que las evaluaciones de la función objetivo sean más certeras. Para ello se creó otra subrutina en la que se fueron incorporando manualmente nuevos puntos a los 65 ya disponibles (muestreo inicial y aproximación a la frontera) para posteriormente ejecutar el algoritmo genético y ver hasta qué valor de masa converge.

La primera prueba consistió en añadir a la nube de datos un nuevo punto próximo al óptimo teórico para ver si repercute en el resultado final del algoritmo genético. En este ejemplo el punto añadido fue el [44.500 4.500 3.000 60.000], mientras que la cadena de cromosomas del óptimo teórico es [44.458 4.386 3.000 60.000]. El desplazamiento del nuevo punto es de 14.807mm y la masa de 1627g. Ejecutando el algoritmo genético con esta nueva incorporación se observa que el óptimo obtenido es el punto [45.189 4.407 3.104 59.955] con una función aptitud de 1610.408. Como se aprecia, el algoritmo genético converge hacia un punto más próximo al óptimo teórico y con un valor mejorado de la función aptitud, todo ello con tan solo un nuevo punto en la nube de datos.

Posteriormente se realizó otra prueba sustituyendo el punto anterior por otro algo más alejado del óptimo teórico, concretamente el [44.500 4.000 3.000 60.000], con un desplazamiento de 15.701mm y una masa de 1509.8g. El óptimo al que llega el algoritmo genético en este caso es [50.643 4.547 3.000 60.000], con valor de la función aptitud de 1634.339. Al estar el nuevo punto añadido algo más alejado del óptimo teórico, el resultado al que llega el AG empeora con respecto al caso anterior, pero evidentemente supone una mejora significativa.

Otra prueba que se realizó fue añadir el diseño [50.000 6.000 5.000 35.800] de 12.537mm de desplazamiento y 2099.6g de masa a la nube de 65 puntos. En este caso, el diseño incorporado está bastante alejado del óptimo teórico. Con estos datos para construir el metamodelo, el AG converge hacia la solución [60.000 4.722 3.000 60.000], de 1663.399 de función aptitud. Como se aprecia, el nuevo punto añadido no aporta mejoras en el metamodelo y por ello el AG converge hacia la misma solución que en el

caso de emplear los 65 puntos. Por ello es muy importante que los nuevos puntos añadidos para el refinamiento del metamodelo se localicen del modo más eficiente posible.

La siguiente idea propuesta para añadir nuevos puntos antes de aplicar el AG final fue simplemente mediante la combinación de los diferentes puntos de la última iteración de aproximación a la frontera por el exterior. Se planteó añadir un punto medio entre cada pareja de puntos que estén en la misma cara del contorno (en un caso tridimensional). En problemas de “n” dimensiones, esto se interpreta como añadir un nuevo punto entre dos puntos de la aproximación a la frontera por el exterior (última iteración) sólo si estos dos puntos contienen un cromosoma idéntico entre sí (una variable con el mismo valor) y además el valor de dicha variable coincide con el valor máximo o mínimo del intervalo definido para dicho cromosoma. En definitiva, volviendo al caso de 3D para facilitar la interpretación, equivale a añadir puntos entre medio de “esquinas de la frontera entre espacio factible y no factible” que pertenezcan a la misma cara del contorno (no son exactamente las esquinas de dicha frontera, si no los puntos añadidos en la última iteración de aproximación a éstas).

Por tanto, se seleccionaron los puntos añadidos en la última iteración de aproximación a la frontera por el exterior (puntos 56~65 de la prueba anterior) y se probó a añadir un punto medio entre dos de ellos que comparten una misma “cara” del contorno. Así por ejemplo, en la primera prueba se probó con los dos últimos puntos (64 y 65), que cumplen esta condición. Sin embargo, el punto 57 y el 65 no cumplen porque no tienen ninguna variable en común con valor máximo o mínimo. En la siguiente tabla se pueden apreciar estas observaciones anteriores:

Punto	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
56	20.000	3.000	4.875	20.000	14.964	1971.300
57	20.000	3.625	3.000	20.000	14.962	1831.000
58	20.000	3.000	6.125	60.000	14.864	1929.700
59	20.000	3.625	3.000	60.000	15.294	1716.800
60	35.000	3.000	8.000	20.000	14.706	1991.300
61	35.000	3.000	8.000	60.000	15.382	1763.500
62	60.000	4.875	3.000	20.000	14.247	1831.000
63	60.000	4.875	3.000	60.000	14.718	1701.600
<b>64</b>	<b>60.000</b>	<b>3.625</b>	<b>8.000</b>	<b>20.000</b>	<b>15.275</b>	<b>1893.400</b>
<b>65</b>	<b>60.000</b>	<b>3.625</b>	<b>8.000</b>	<b>60.000</b>	<b>16.053</b>	<b>1652.300</b>

Tabla 11. Puntos añadidos en la aproximación a la frontera en la última iteración. Combinación de los puntos 64 y 65.

El punto añadido en medio de estos dos puntos (64 y 65) se corresponde con la cadena de cromosomas [60.000 3.625 8.000 40.000], con un valor de desplazamiento de 16.003mm y una masa de 1721.1g. Añadiendo este punto en la nube de datos para construir el metamodelo de aproximación y ejecutando el algoritmo genético, se obtiene un diseño óptimo definido por [60.00 4.722 3.000 60.000], con una función aptitud de 1663.298. La adición del nuevo punto no ha conseguido mejorar la solución óptima alcanzada por el algoritmo genético.

Esta misma prueba se realizó con múltiples combinaciones de los puntos de la tabla anterior (cumpliendo la condición de compartir contorno). Sin embargo, no se consiguió ninguna mejora hasta combinar los puntos 59 y 63. En este caso, el nuevo punto añadido sería el [40.000 4.250 3.000 60.000], con desplazamiento de 15.061mm y masa de 1597.7g.

Punto	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
56	20.000	3.000	4.875	20.000	14.964	1971.300
57	20.000	3.625	3.000	20.000	14.962	1831.000
58	20.000	3.000	6.125	60.000	14.864	1929.700
<b>59</b>	<b>20.000</b>	<b>3.625</b>	<b>3.000</b>	<b>60.000</b>	<b>15.294</b>	<b>1716.800</b>
60	35.000	3.000	8.000	20.000	14.706	1991.300
61	35.000	3.000	8.000	60.000	15.382	1763.500
62	60.000	4.875	3.000	20.000	14.247	1831.000
<b>63</b>	<b>60.000</b>	<b>4.875</b>	<b>3.000</b>	<b>60.000</b>	<b>14.718</b>	<b>1701.600</b>
64	60.000	3.625	8.000	20.000	15.275	1893.400
65	60.000	3.625	8.000	60.000	16.053	1652.300

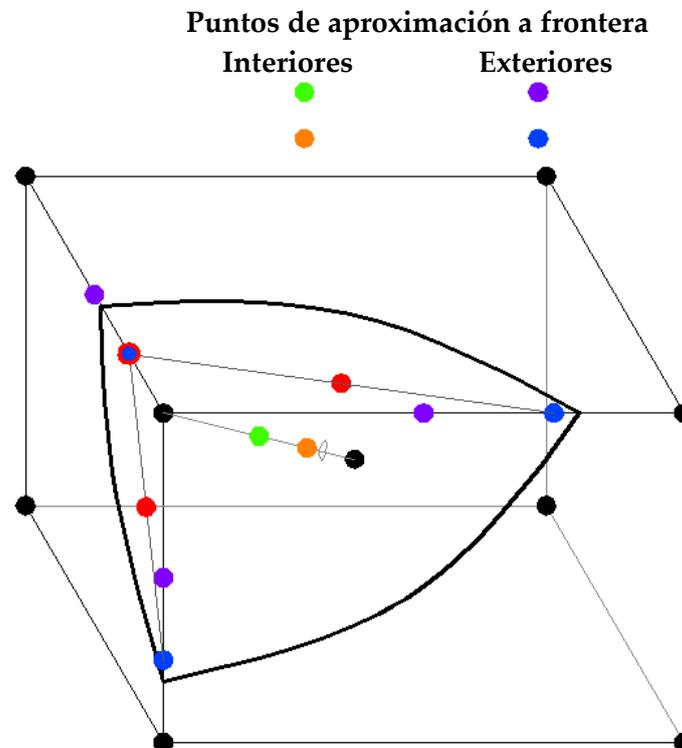
Tabla 12. Puntos añadidos en la aproximación a la frontera en la última iteración. Combinación de los puntos 29 y 63.

Ejecutando el algoritmo genético con el metamodelo construido con los 65 puntos y el nuevo punto añadido, el óptimo obtenido es el [41.662 4.327 3.000 60.000], con un valor de la función aptitud de 1612.328g. Como se observa, el óptimo obtenido se aproxima significativamente al óptimo teórico, por lo que en este caso esta estrategia de adición de nuevos puntos da buenos resultados. Sin embargo, el hecho de añadir todos los puntos que cumplan esta condición supondría aumentar el muestreo hasta un total de 96 puntos antes de aplicar el AG final, lo cual es un valor muy elevado para este problema de solo 4 variables. En ese caso de 96 puntos, el óptimo obtenido por el AG es [40.996 4.309 3.000 60.000], con un valor de la función aptitud de 1609.730. A pesar de que el resultado es bueno, el número de puntos evaluados es inaceptable, por

lo que se decide tratar de encontrar de antemano qué combinación de esquinas conducirá a un punto más eficiente para mejorar el metamodelo.

Observando las esquinas combinadas (punto 59 y 63) se aprecia que una de ellas es prácticamente la esquina de menor masa (punto 63, con desplazamiento de 14.718mm y masa de 1701.600g), sólo superada por el punto 65 (1652.300g). Sin embargo el desplazamiento del punto 65 es significativamente mayor (16.053mm), por lo que en igualdad de condiciones sería probable que la esquina asociada al punto 63 sea la mejor.

En base a esto, un planteamiento lógico sería buscar la esquina de la frontera entre espacio factible y no factible con menor masa y combinar ésta con las esquinas contiguas. Es decir, si todas las esquinas de la frontera están prácticamente al límite entre espacio factible y no factible, la esquina de menor masa será la mejor entre éstas. Sin embargo, para aplicar esta idea apropiadamente es necesario que todos los puntos de aproximación a la frontera de la última iteración estén en unas condiciones comparables. Es decir, si un punto está muy próximo a la frontera y otro punto está algo más alejado hacia la zona no factible, se puede cometer un error en la elección de la mejor esquina simplemente por esta diferencia de aproximación a la frontera entre ambas esquinas.



**Ilustración 22.** Esquina de menor masa errónea por no aproximarse suficientemente a la frontera.

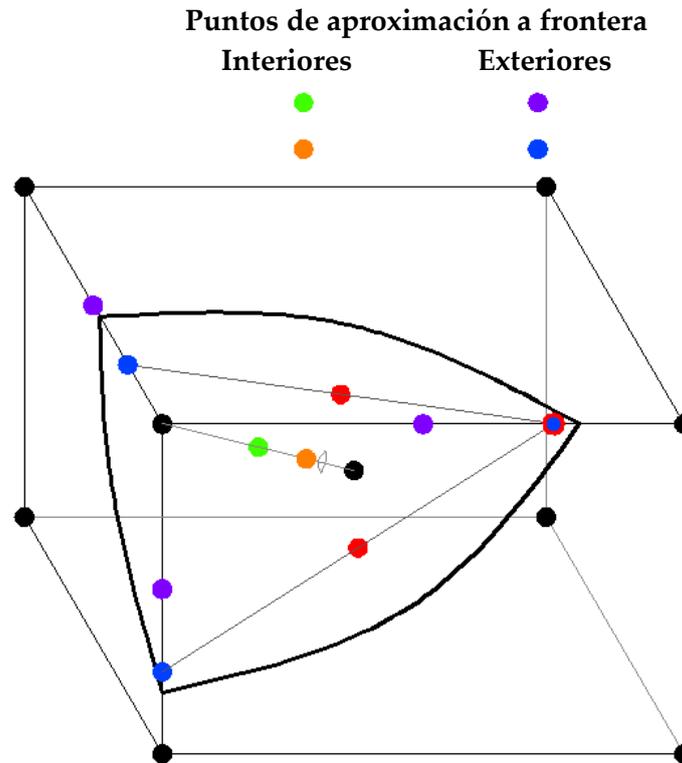


Ilustración 23. Esquina “real” de menor masa.

Gráficamente esto se interpreta a través de los puntos azules con contorno rojo de las imágenes anteriores. En el primer caso la esquina elegida sería el punto azul con contorno rojo, que realmente no es la mejor esquina sino que se escoge erróneamente, lo cual conlleva a añadir los dos puntos rojos al combinarse con el resto de esquinas contiguas. Sin embargo en el segundo caso se indica la mejor esquina real (azul con contorno rojo) y se muestran (en rojo) los puntos añadidos con esta esquina de referencia.

Para evitar una diferencia excesiva entre unos puntos y otros en la aproximación a la frontera y así poder aplicar esta nueva idea, se decidió cambiar la estrategia de adición de puntos en la aproximación a la frontera. La nueva estrategia consiste en realizar una estimación lineal a partir de los valores de las restricciones de los dos puntos que se empleaban anteriormente para calcular el punto medio, con la finalidad de añadir el nuevo punto justo en la frontera según las estimaciones de la interpolación lineal. Es decir, si se va a añadir un nuevo punto (“p”) entre dos puntos de referencia de los cuales el primero (“p<sub>1</sub>”) cumple la restricción de desplazamiento con 14.5mm (el error cometido es de  $|15-14.5|=0.5\text{mm}$ , “e<sub>1</sub>”) y el segundo (“p<sub>2</sub>”) no cumple la restricción con un valor de 26mm (el error cometido es de  $|15-26|=11\text{mm}$ ,

“ $e_2$ ”), la interpolación lineal añadirá el nuevo punto mucho más próximo al primero de ellos. Esto se realiza mediante una media ponderada de los dos puntos en función de la proximidad al valor límite de la restricción.

$$p = p_1 \cdot \frac{e_2}{e_1 + e_2} + p_2 \cdot \frac{e_1}{e_1 + e_2}$$

Por otro lado hay que decir que con esta nueva propuesta se está intensificando aún más el muestreo por los contornos del dominio. Este aspecto puede ser muy positivo sobre todo teniendo en cuenta que en la mayoría de problemas existe una disparidad importante entre la influencia de las diferentes variables de diseño sobre las respuestas del sistema, por lo que es muy frecuente que el diseño óptimo se encuentre en un valor extremo de alguna o varias variables de diseño. Por tanto es muy importante intensificar la búsqueda en el contorno del dominio.

También se plantea otra nueva modificación al programa original para mejorar el ajuste del metamodelo. Anteriormente se simulaba el mejor individuo de cada generación del algoritmo genético final para ir mejorando la precisión del metamodelo y alcanzar el óptimo teórico a medida que evoluciona el algoritmo. Sin embargo esta estrategia no da buenos resultados, por lo que se descarta la simulación del mejor individuo de cada población y se emplea otro tipo de refinamiento. Como se observó en los resultados anteriores, el óptimo alcanzado al finalizar el programa sólo conseguía buenos resultados cuando se simulaba un individuo poco evolucionado en las primeras generaciones. Esto es síntoma de que el metamodelo de aproximación requiere de la adición de nuevos puntos para mejorar su ajuste. Estos nuevos puntos se deben añadir en la medida de lo posible en la frontera entre espacio factible y no factible, que es donde se encontrará el diseño óptimo. A pesar de que esto es justamente lo que se realiza en la fase de muestreo inicial y aproximación a la frontera por las aristas e interior, se plantea aplicar un algoritmo genético de “aproximación a la frontera” que consistirá básicamente en ejecutar un AG (evaluando la aptitud de los individuos con el metamodelo de aproximación) en repetidas ocasiones (tantas veces como variables de diseño) de modo que se simule el óptimo de cada ejecución para refinar el metamodelo y se penalice a los individuos de las siguientes ejecuciones de este AG por cercanía a los puntos simulados previamente en esta fase. Con esto se conseguirá añadir nuevos puntos próximos a la frontera y con una cierta expansión (causada por la penalización) que se puede interpretar como “exploración” de la zona, mejorando el ajuste.

Posteriormente se aplicará el AG de optimización final. Este algoritmo empleará el metamodelo para evaluar la aptitud de los individuos y, con la evolución del mismo,

se obtendrá el diseño óptimo estimado, que será simulado. Finalmente se escoge como óptimo el mejor diseño de todos los simulados.

## 7.2. Segundo programa (AG de aproximación a frontera)

Las ideas propuestas en el apartado anterior se implementaron en una subrutina de Matlab para valorar su eficiencia. Los nuevos cambios se resumen en:

- 1) Adición de nuevos puntos en la aproximación a la frontera (exterior e interior) mediante interpolación de las restricciones de los puntos combinados (anteriormente se añadía siempre en el punto medio entre los dos puntos de referencia). En el caso de que existan varias restricciones en el problema y que por tanto se pueda estimar el nuevo punto añadido atendiendo a los valores de unas restricciones u otras, se calculará la posición de los diferentes puntos posibles y finalmente se escogerá el más próximo al punto de referencia factible. La siguiente imagen facilita la interpretación. Suponiendo que se tiene una esquina del dominio que cumple las dos restricciones del problema (punto negro a la izquierda), y otro punto a la derecha que no cumple ninguna restricción, se podría estimar la frontera de la primera restricción (punto rojo) y de la segunda restricción (punto azul). Según la estimación, los puntos a la derecha del punto rojo (sobre la arista) no cumplirán la primera restricción, por lo que el nuevo punto añadido será el rojo y no el azul. Es decir, siempre se escogerá el punto más próximo a la zona factible.

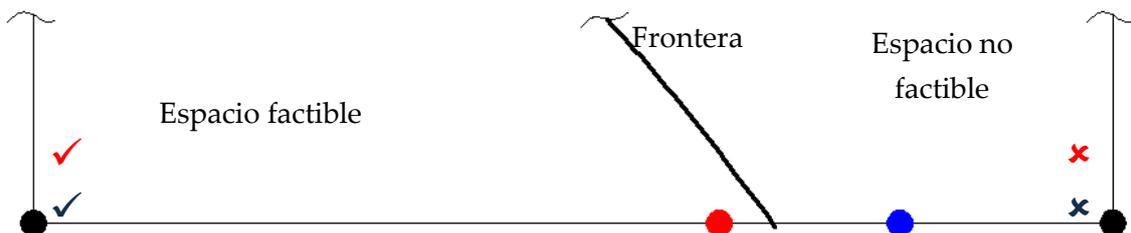


Ilustración 24. Selección del nuevo punto de aproximación a la frontera con varias restricciones posibles para su localización.

- 2) Adición de puntos intermedios entre la mejor esquina de la frontera y las restantes contiguas. La mejor esquina de la frontera se busca en los puntos añadidos en la última iteración de la fase anterior, y se combina con el resto de

- puntos exteriores de dicha iteración que sean contiguos (misma cara del contorno).
- 3) Incorporación de un algoritmo genético de aproximación en el que se ejecuta el AG varias veces, simulando los diseños óptimos obtenidos y penalizando a los individuos que se aproximen a éstos en ejecuciones posteriores del AG de aproximación. En un principio se decidió ejecutar este algoritmo al menos “n” veces, es decir, tantas veces como variables de diseño existentes en el problema. No obstante, para garantizar que el metamodelo de ajuste sea lo suficientemente exacto al finalizar esta fase de la metodología, se propuso establecer otro criterio de parada que consiste en repetir el AG de aproximación a la frontera hasta que las respuestas estimadas por el metamodelo para el punto óptimo al que converge tengan un error inferior a lo establecido por el usuario. Por ello, una vez repetido el AG de aproximación “n” veces se evalúan los errores de estimación de las diferentes restricciones y en el caso de que sean superiores a los valores establecidos por el usuario se repite de nuevo el AG.

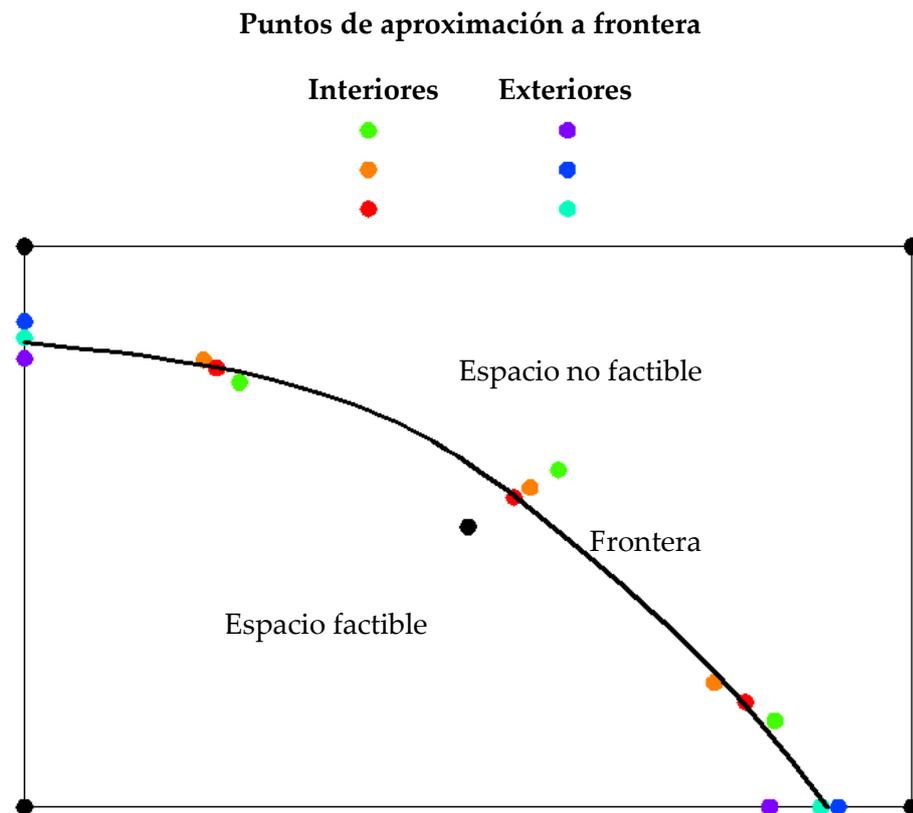


Ilustración 25. Adición de puntos (aproximación frontera) mediante interpolación lineal.

Evidentemente con la interpolación lineal en la adición de puntos de aproximación a la frontera se consigue una mayor precisión en la localización de los mismos que en el programa inicial, como se aprecia en la imagen anterior.

El pseudocódigo para este programa sería:

- 1) Muestreo inicial  $2^{n+1}$  (esquinas y punto central).
- 2) Aproximación a la frontera con la adición de nuevos puntos hasta que el metamodelo estime los resultados con una desviación media inferior al valor definido por el usuario:
  - a) Puntos interiores: se añaden nuevos puntos entre el punto central y las esquinas que se encuentren en el espacio opuesto mediante una media ponderada según el valor de las restricciones de los puntos. En las sucesivas iteraciones se añade un nuevo punto (con la misma estrategia) entre el último punto añadido y el punto más cercano del espacio opuesto de los creados en iteraciones anteriores al último punto añadido.
  - b) Puntos exteriores: se añaden nuevos puntos entre esquinas del dominio contiguas que se encuentren en espacios opuestos mediante una media ponderada según el valor de las restricciones. En las sucesivas iteraciones se añade un nuevo punto (también interpolando las restricciones) entre el último punto añadido y el punto más cercano del espacio opuesto y creado en iteraciones anteriores al último punto añadido.
- 3) Adición de puntos intermedios entre la mejor esquina de la frontera y las restantes contiguas (puntos obtenidos en la última iteración de aproximación a frontera por el exterior).
- 4) Algoritmo genético de aproximación a la frontera. Se repite reiteradamente siempre que se haya ejecutado menos de "n" veces (número de variables de diseño) o si el error de la estimación de los resultados del punto óptimo al que converge el algoritmo difiere más de lo permitido por el usuario. El mejor individuo alcanzado por el algoritmo se simula y se incluye en la nube de datos para refinar el metamodelo y trabajar con mayor precisión en las siguientes ejecuciones. Los individuos próximos a puntos evaluados en esta fase se penalizan para favorecer la exploración.
- 5) Algoritmo genético de optimización final. Con la nube de puntos elaborada en las etapas anteriores se reconstruye el metamodelo de aproximación para evaluar la aptitud de los individuos en el algoritmo. Finalmente, éste converge hacia un óptimo que se simula. En el caso de que no cumpla las restricciones, se actualiza el metamodelo con el último punto evaluado y se ejecuta de nuevo el algoritmo en un proceso iterativo hasta conseguir un diseño factible.
- 6) Búsqueda del mejor individuo simulado (óptimo).

Dado que la elección de la mejor esquina de la frontera depende del error máximo permitido en la fase de aproximación a la frontera, se decidió establecer un

valor relativamente bajo que garantice en cierta medida que el programa selecciona la esquina correcta para añadir los nuevos puntos entre ésta y el resto contiguas. El valor elegido fue un 1% de error, de modo que en cada iteración se calcula el error absoluto cometido por cada punto con respecto a la restricción “crítica” (la más desfavorable en cada caso) y finalmente se calcula el valor medio del error de todos los puntos de la última iteración, en tanto por ciento. Si este valor supera el 1%, se repetirá de nuevo el bucle.

Igualmente se tomó este valor como condición de parada del AG de aproximación a la frontera. En este caso, se calcula el error absoluto del punto óptimo al que llega el algoritmo con respecto a los valores de las diferentes restricciones obtenidos en la simulación. Finalmente se determina el error medio y si es superior al 1% se repite el AG de aproximación.

Por otro lado también hay que hacer una breve mención a la evaluación de la función aptitud en el AG de aproximación. Evidentemente la aptitud de cada individuo de la población vendrá dada por la suma del valor de masa estimado por el metamodelo de aproximación y la penalización correspondiente si el individuo no cumple alguna restricción según los valores obtenidos del metamodelo correspondiente. Sin embargo, en este AG también se penalizará a los individuos que estén muy próximos a los puntos óptimos obtenidos en iteraciones anteriores del propio algoritmo. Es decir, para cada individuo de la población se calculará la distancia a los puntos simulados en iteraciones anteriores y si está a una distancia menor de un cierto valor, se sumará un valor proporcional a la cercanía (amplificado con un factor) aumentando la función aptitud y penalizando al individuo (se minimiza) para que el algoritmo evolucione hacia individuos más alejados de los puntos ya añadidos, obligando por tanto a explorar en la frontera factible/no factible. Por ello se ha denominado “algoritmo genético de aproximación a la frontera”.

Así por ejemplo, suponiendo que ya se han ejecutado dos iteraciones de este algoritmo y se ha añadido por tanto el punto óptimo de cada caso (puntos en verde y rojo respectivamente de la siguiente imagen), en la tercera iteración los individuos próximos a estos dos puntos se penalizarán por cercanía. En el caso del punto azul, se calculará la distancia del mismo a cada uno de los puntos simulados previamente, es decir, la distancia al punto verde y al punto rojo. Si la distancia (“ $d_{\text{punto1-punto2}}$ ”) es inferior a un valor dado, entonces se penaliza. En otras palabras, si el punto en cuestión está dentro del radio de influencia de uno de los puntos añadidos anteriormente (es decir, dentro del “nicho” en la terminología de los AG multiobjetivo con penalización por compartición de recursos), se sumará un término a la aptitud que empeorará la “calidad” del individuo.

En el caso del punto azul de la imagen izquierda se penalizará por cercanía al punto rojo, siendo la penalización:

$$P_{cercanía} = \frac{1}{d_{\text{azul-rojo}}} \cdot 10^{99}$$

En el caso de que el punto azul estuviera dentro de otros nichos, se acumularía la penalización de todos los casos en “ $P_{cercanía}$ ”. La función aptitud (“ $F$ ”) sería por tanto la suma del valor de la masa, la penalización por incumplimiento de las restricciones (“ $P_{restricciones}$ ”) y la penalización por cercanía (“ $P_{cercanía}$ ”):

$$F = \text{masa} + P_{restricciones} + P_{cercanía}$$

Para determinar el valor del radio de influencia o distancia mínima (“ $d_{\min}$ ”) a partir de la que se penaliza a un individuo por cercanía a un punto evaluado, primero se calcula el radio equivalente del dominio (“ $R_{equivalente}$ ”) con la siguiente ecuación:

$$R_{equivalente} = \frac{\sqrt{\sum_{i=1}^n (\text{Límite superior}_i - \text{Límite inferior}_i)^2}}{2}$$

Siendo “ $n$ ” el número de variables de diseño, y “Límite superior $_i$ ” y “Límite inferior $_i$ ” los valores extremos de la variable “ $i$ ”.

Posteriormente se establece el número de nichos (“ $q$ ”) en el que se divide el dominio y con esto se determina la distancia mínima:

$$d_{\min} = \frac{R_{equivalente}}{\sqrt[q]{q}}$$

Todos los puntos a una distancia menor que (“ $d_{\min}$ ”) de un punto simulado anteriormente serán penalizados.

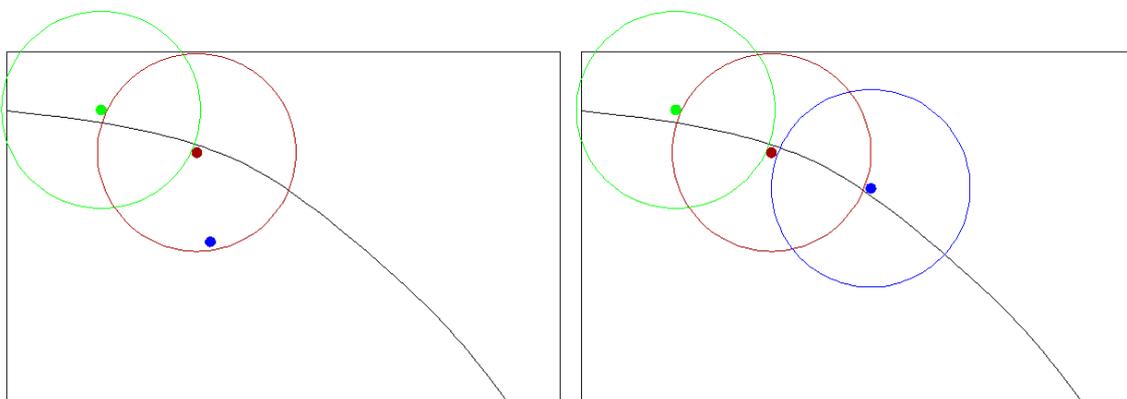


Ilustración 26. Punto azul penalizado por cercanía (izquierda) y punto final óptimo al finalizar el AG (punto azul en la imagen derecha).

La evolución del algoritmo genético guiará hacia un individuo óptimo lo más cercano posible al de la iteración anterior pero sin entrar en el radio de influencia y próximo a la frontera entre espacio factible y no factible, como se observa en la imagen derecha anterior. Esto favorece la exploración de la frontera.

A mayor número de nichos (“q”), menor será el radio de influencia y por tanto la exploración se realizará de modo más localizado. Sin embargo si el número de nichos es muy bajo el radio de influencia será mayor y por tanto la exploración se expandirá más (menos concentrada). Se realizaron algunas pruebas modificando el número de nichos, pero no se apreció una diferencia significativa en los resultados. Finalmente se decidió emplear 20 nichos (“q=20”) para realizar la exploración más localizada en las zona de mejor comportamiento según el metamodelo.

Por otro lado, el AG de aproximación a la frontera se configuró según las prueba número 5 del capítulo anterior, con la diferencia de que se estableció un número mayor de generaciones (100) para favorecer la convergencia hacia al óptimo. Sin embargo el AG de optimización final se configuró atendiendo a la prueba 6 del capítulo anterior, es decir, con mayor probabilidad de mutación (80%) pero con menor rango de mutación de la variable ( $\pm 10\%$ ). En este caso el número de generaciones se fijó en 200 para garantizar la convergencia a pesar de la alta probabilidad de mutación.

### 7.2.1. Resultados obtenidos para el problema con 4 variables de diseño

En la siguiente tabla se muestra el resultado óptimo obtenido en 10 ejecuciones diferentes:

Ejecución	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Nº puntos simulados	Desplazamiento (mm)	Masa (g)
1	42.211	4.350	3.000	59.008	64	14.998	1603.458
2	46.248	4.430	3.000	60.000	61	14.993	1602.619
3	42.643	4.351	3.000	60.000	61	14.990	1602.928
4	41.771	4.330	3.000	60.000	62	14.993	1603.623
5	46.339	4.432	3.000	60.000	61	14.993	1602.675
6	40.209	4.294	3.000	59.767	61	14.999	1605.684
7	40.469	4.301	3.000	59.794	61	14.997	1605.389
8	44.308	4.386	3.000	60.000	61	14.995	1601.509
9	41.625	4.330	3.000	60.000	61	14.988	1604.473
10	40.862	4.311	3.000	59.725	62	14.997	1604.788

Tabla 13. Resultados obtenidos en 10 ejecuciones del segundo programa (cuatro variables).

Como se aprecia el mejor diseño en todas las ejecuciones se aproxima bastante al óptimo teórico, sobre todo en términos de masa. Sin embargo la cadena de cromosomas varía ligeramente, lo cual es un indicativo de que podrían existir distintos diseños con unos resultados muy similares en términos de masa y desplazamiento. También se puede observar que el número total de simulaciones es de 61, aunque puede aumentar dependiendo de la rapidez con la que se cumpla el criterio de parada del AG de aproximación o el AG final.

En la siguiente tabla se recogen los resultados de los diferentes puntos simulados en la última ejecución anterior:

Punto	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
1	20.000	3.000	3.000	20.000	16.088	1712.307
2	20.000	3.000	3.000	60.000	16.483	1582.601
3	20.000	3.000	8.000	20.000	13.653	2266.619
4	20.000	3.000	8.000	60.000	14.230	2056.206
5	20.000	8.000	3.000	20.000	11.649	2529.173
6	20.000	8.000	3.000	60.000	11.547	2523.654
7	20.000	8.000	8.000	20.000	10.918	2808.140
8	20.000	8.000	8.000	60.000	10.999	2721.914
9	60.000	3.000	3.000	20.000	17.984	1394.122
10	60.000	3.000	3.000	60.000	18.641	1218.149
11	60.000	3.000	8.000	20.000	16.471	1777.462
12	60.000	3.000	8.000	60.000	17.311	1520.782
13	60.000	8.000	3.000	20.000	11.287	2464.468
14	60.000	8.000	3.000	60.000	11.448	2412.682
15	60.000	8.000	8.000	20.000	11.478	2572.463
16	60.000	8.000	8.000	60.000	11.822	2439.970
17	40.000	5.500	5.500	40.000	12.678	2067.614
18	26.383	3.798	3.798	26.383	14.723	1823.852
19	27.794	3.974	3.974	52.206	14.690	1769.255
20	48.752	4.406	4.406	31.248	14.554	1775.247
21	47.787	4.527	4.527	47.787	14.500	1749.807
22	52.245	3.969	7.031	27.755	14.684	1870.282
23	50.023	4.247	6.753	50.023	14.550	1804.238
24	20.000	3.000	5.234	20.000	14.778	2013.885
25	20.000	4.226	3.000	20.000	14.033	1940.628
26	20.000	3.000	6.291	60.000	14.798	1943.407
27	20.000	4.502	3.000	60.000	13.902	1897.116
28	39.125	3.000	8.000	20.000	14.996	1934.883
29	29.992	3.000	8.000	60.000	14.997	1848.964
30	60.000	5.228	3.000	20.000	13.709	1908.447

31	60.000	5.531	3.000	60.000	13.694	1860.653
32	60.000	4.473	8.000	20.000	13.914	2043.129
33	60.000	5.105	8.000	60.000	13.726	1944.777
34	25.086	3.636	3.636	25.086	14.979	1801.048
35	26.447	3.806	3.806	53.553	14.972	1736.786
36	50.216	4.223	4.223	29.784	14.935	1726.005
37	49.261	4.342	4.342	49.261	14.917	1687.809
38	53.616	3.798	7.202	26.384	14.971	1852.055
39	51.650	4.044	6.956	51.650	14.939	1759.449
40	20.000	3.000	4.855	20.000	14.975	1968.865
41	20.000	3.649	3.000	20.000	14.922	1835.457
42	20.000	3.000	5.896	60.000	14.959	1910.065
43	20.000	3.863	3.000	60.000	14.885	1766.605
44	39.184	3.000	8.000	20.000	15.000	1934.138
45	30.033	3.000	8.000	60.000	15.000	1848.223
46	60.000	4.555	3.000	20.000	14.781	1759.439
47	60.000	4.863	3.000	60.000	14.740	1698.528
48	60.000	3.847	8.000	20.000	14.889	1933.478
49	60.000	4.357	8.000	60.000	14.788	1800.254
50	40.000	4.256	3.000	40.000	15.189	1608.060
51	40.000	3.931	4.448	60.000	15.024	1666.581
52	40.000	4.363	3.000	60.000	14.869	1623.368
53	45.016	3.931	5.500	60.000	14.950	1702.756
54	60.000	4.709	3.000	40.000	15.102	1675.442
55	60.000	4.355	5.500	40.000	15.032	1779.832
56	60.000	4.610	5.500	60.000	14.569	1797.322
57	40.673	4.315	3.000	60.000	14.976	1607.657
58	43.446	4.461	3.000	45.646	15.000	1619.246
59	52.423	4.526	3.000	34.473	15.079	1643.651
60	30.066	4.090	3.000	49.833	15.099	1651.715
61	41.207	4.329	3.000	58.042	15.017	1603.312
62	40.862	4.311	3.000	59.725	14.997	1604.788

Tabla 14. Puntos simulados en el programa.

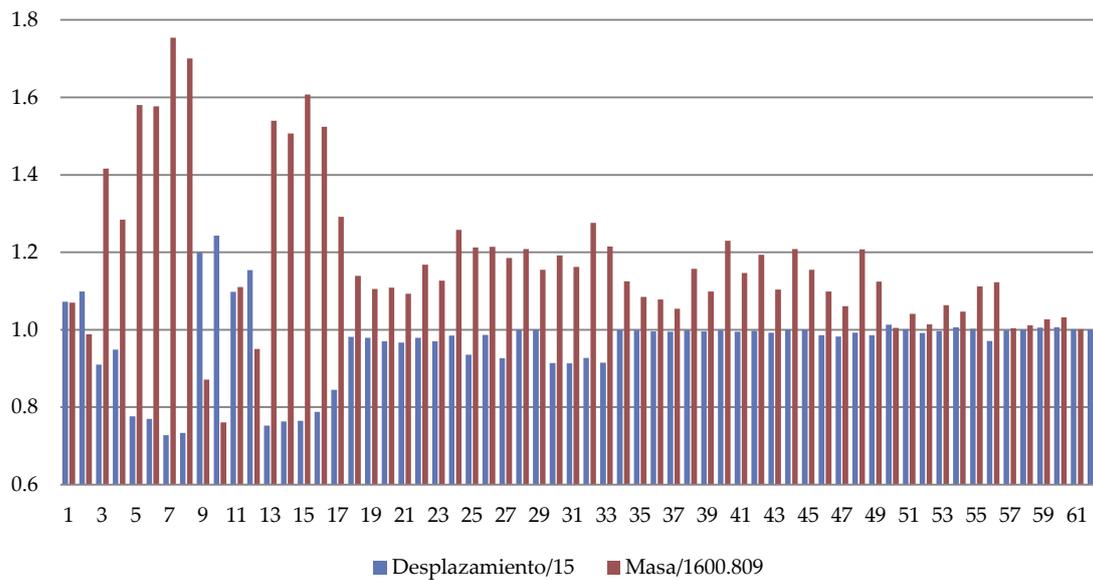
Los primeros 17 puntos de la tabla (en negro), se corresponden con las esquinas del dominio y el punto central ( $2^n+1$ ). Los siguientes 5 puntos (18~23, en naranja) se corresponden con la primera iteración en la adición de puntos de aproximación a la frontera por el interior. Los 10 puntos siguientes (24~33, en verde), se corresponden a los puntos de aproximación la frontera por las aristas. Esta etapa se repite de nuevo hasta el punto 49. Los siguientes 7 puntos en violeta (50~56) están asociados a los puntos añadidos entre la mejor esquina de la frontera (punto 47) y las contiguas. Posteriormente se encuentran 4 puntos (57~60, en azul) vinculados al AG de aproximación a la frontera y, finalmente, los últimos 2 puntos (61~62, en rojo) asociados al AG de optimización final.

Para ver la evolución de los diseños simulados se representó gráficamente el valor de masa y desplazamiento (de cada punto o diseño) relativo a la masa y desplazamiento respectivamente del diseño óptimo teórico (1600.809g y 15.000mm):

$$\text{Desplazamiento relativo} = \frac{\text{Desplazamiento}}{15}$$

$$\text{Masa relativa} = \frac{\text{Masa}}{1600.809}$$

### Desplazamiento y masa relativos al diseño óptimo teórico



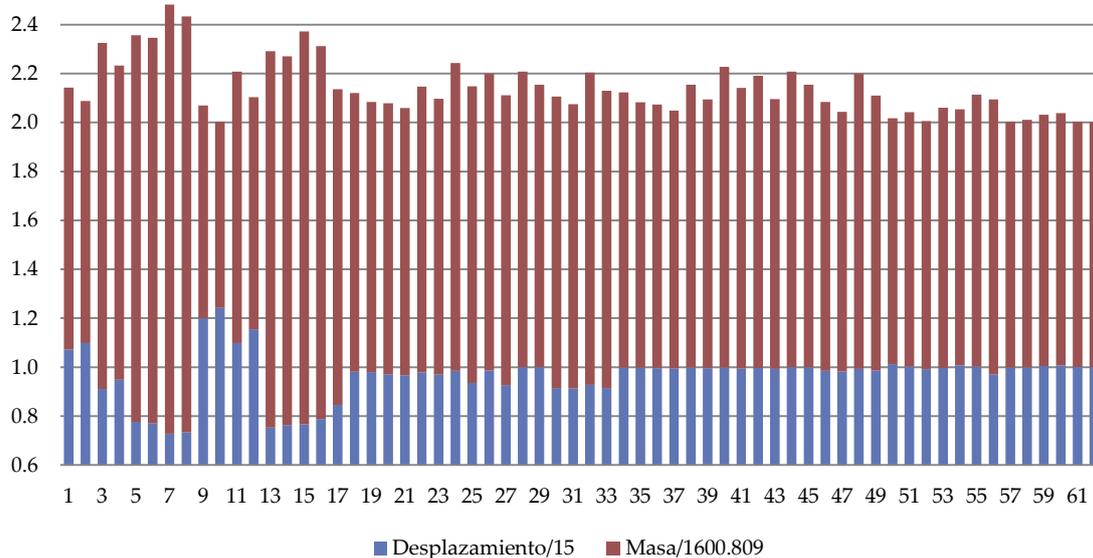
Gráfica 7. Desplazamiento y masa de los puntos simulados relativos a los valores obtenidos en el diseño óptimo (4 variables).

En la gráfica anterior se aprecia como a medida que mejoran los diseños (puntos evaluados), la masa y desplazamiento relativos se aproximan a 1. También se aprecia que en algunos casos la masa relativa es inferior a 1 (menor por tanto que la del diseño óptimo), pero el desplazamiento relativo correspondiente a esos casos es superior a 1, lo que indica un diseño no factible (desplazamiento mayor de 15mm).

En la siguiente gráfica se representa el valor acumulado del desplazamiento relativo y masa relativa para cada uno de los diseños evaluados. En términos globales interesa obtener un valor lo más reducido posible de esta suma ya que esto supondría un diseño con poca masa y poco desplazamiento. Observando dicha gráfica cabe destacar por ejemplo que el diseño número 10 (que se corresponde con la cadena de cromosomas [60.000 3.000 3.000 60.000], esquina del dominio) consigue un valor casi

idéntico que el óptimo al que llega el programa (punto 62). Sin embargo en el primer caso el desplazamiento relativo es mucho mayor y la masa relativa bastante menor. En un problema de optimización multiobjetivo en el que se pretenda minimizar la masa y el desplazamiento, estos dos puntos serían igual de buenos, formando parte ambos puntos del frente más próximo a la frontera de Pareto. El diseño 62 sería más óptimo en términos de desplazamiento que el número 10, pero en términos de masa este último sería mejor. En una situación muy similar se encuentran otros puntos como el 52, 57 ó 61, por lo que todos ellos serían igual de óptimos en un problema multiobjetivo.

**Acumulación del desplazamiento~masa relativos al diseño óptimo**



Gráfica 8. Valor acumulado de desplazamiento y masa relativa de los diseños evaluados (4 variables).

### 7.2.2. Resultados obtenidos para el problema con 5 variables de diseño

Dado que el programa parece que converge hacia el óptimo se probó con otro problema basado en el mismo ejemplo de referencia pero incluyendo una variable de diseño más. La finalidad de esta prueba es ver nuevamente si el programa guía el diseño hasta un punto cercano al óptimo teórico.

En el caso anterior las celdas huecas en el interior de la pieza tenían forma de prismas cuadrados, ya que los dos lados de la base eran siempre iguales. Sin embargo en este caso los lados pueden ser diferentes dando lugar a huecos con forma de prismas rectangulares. Por tanto las variables de diseño de este problema serían: la

dimensión del lado vertical de la base del prisma rectangular hueco ( $L_1 = 20\sim 60\text{mm}$ ), la dimensión del lado horizontal de la base del prisma rectangular hueco ( $L_2 = 20\sim 60\text{mm}$ ), el espesor exterior ( $e = 3\sim 8\text{mm}$ ) de la piel que define el contorno de la pieza, el espesor de pared entre huecos ( $e_h = 3\sim 8\text{mm}$ ) y la altura de los prismas rectangulares ( $h = 20\sim 60\text{mm}$ ).

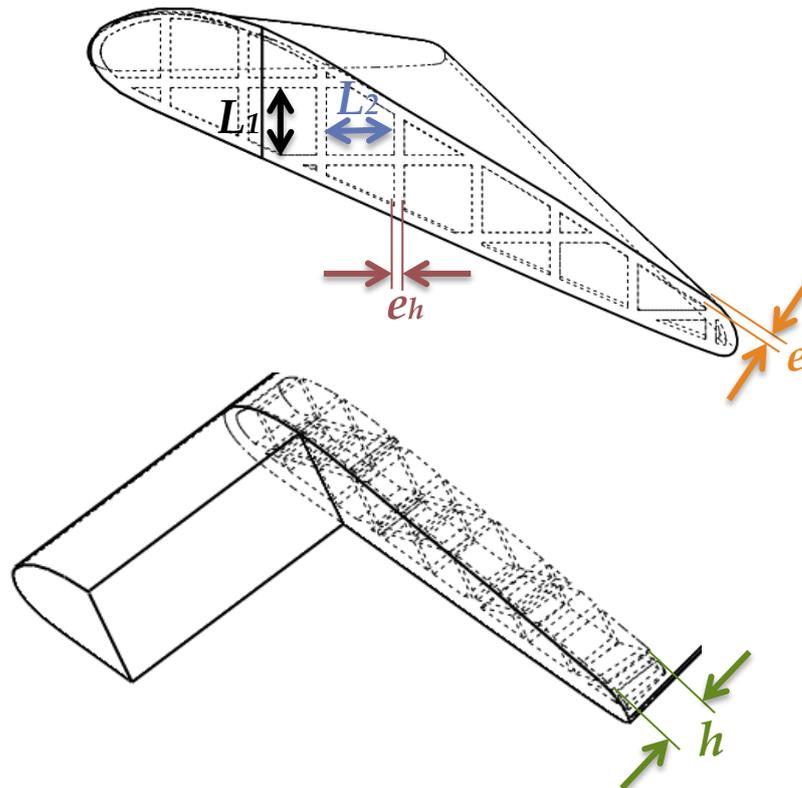


Ilustración 27. Variables de diseño del problema de referencia (5 variables).

Para evitar el cálculo por elementos finitos de los diseños evaluados en estas primeras fases de pruebas, se sustituyeron las simulaciones por unas ecuaciones polinómicas de orden 2 ajustadas por mínimos cuadrados a los desplazamientos y masa de 243 puntos simulados previamente (diseño factorial de 3 niveles,  $3^5=243$ ). Las ecuaciones obtenidas fueron:

$$\begin{aligned} \text{desplazamiento} = & 22.7562 + 0.0690042p(1) + 0.0232506p(2) - 2.72272p(3) - 0.892653p(4) \\ & + 0.0226944p(5) - 0.000441598p(1)^2 + 0.000234441p(1)p(2) - 0.00664508p(1)p(3) \\ & + 0.00124339p(1)p(4) + 0.0000954352p(1)p(5) + 0.000176765p(2)^2 \\ & - 0.00710247p(2)p(3) + 0.001481p(2)p(4) + 0.0000717502p(2)p(5) + 0.181117p(3)^2 \\ & + 0.0722127p(3)p(4) - 0.00277144p(3)p(5) + 0.0156755p(4)^2 + 0.000558176p(4)p(5) \\ & - 0.00010755p(5)^2 \end{aligned}$$

$$\begin{aligned} \text{masa} = & 1164.64 - 12.487p(1) - 9.69244p(2) + 222.976p(3) + 194.915p(4) - 9.68803p(5) \\ & + 0.101916p(1)^2 - 0.016941p(1)p(2) + 0.701231p(1)p(3) - 0.420172p(1)p(4) \\ & - 0.0164958p(1)p(5) + 0.0731809p(2)^2 + 0.640947p(2)p(3) - 0.381789p(2)p(4) \\ & - 0.0147488p(2)p(5) - 6.09746p(3)^2 - 10.8345p(3)p(4) + 0.630854p(3)p(5) \\ & - 2.96703p(4)^2 - 0.332141p(4)p(5) + 0.0745826p(5)^2 \end{aligned}$$

$$p(j) = \text{valor de la variable "j"} \quad (j=1,2,3,4,5)$$

El óptimo “teórico” obtenido mediante una función de optimización del Matlab basada en el método del gradiente fue:

**Óptimo teórico: 51.363 40.794 4.509 3.000 54.840**

**Masa: 1659.573g**

**Desplazamiento: 15.000mm**

En la siguiente tabla se muestran los diseños óptimos obtenidos en 10 ejecuciones del programa:

Ejecución	L1 (mm)	L2 (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Nº puntos simulados	Desplazamiento (mm)	Masa (g)
1	58.076	39.276	4.546	3.000	60.000	125	14.997	1663.433
2	53.937	41.310	4.519	3.000	51.789	126	15.000	1660.502
3	58.564	41.600	4.586	3.000	60.000	126	14.997	1663.670
4	53.058	42.203	4.557	3.000	57.758	129	14.998	1660.663
5	52.020	41.057	4.531	3.000	57.817	126	15.000	1660.201
6	58.076	39.276	4.546	3.000	60.000	125	14.997	1663.433
7	53.937	41.310	4.519	3.000	51.789	126	15.000	1660.502
8	58.564	41.600	4.586	3.000	60.000	126	14.997	1663.670
9	53.058	42.203	4.557	3.000	57.758	129	14.998	1660.663
10	52.020	41.057	4.531	3.000	57.817	126	15.000	1660.201

Tabla 15. Resultados obtenidos en 10 ejecuciones del segundo programa (5 variables).

Se aprecia que el programa converge en todas las ocasiones a una solución similar en términos de masa, siendo ésta muy próxima a la del diseño óptimo teórico.

En la siguiente gráfica se muestra el desplazamiento y masa relativos al diseño óptimo teórico. Dado que el número de puntos evaluados (126) es demasiado elevado para su representación a modo de diagrama de barras, se representó en forma de línea. Nuevamente se observa que a medida que avanza el programa los diseños evaluados van disminuyendo su masa relativa y ajustando el desplazamiento hasta el valor máximo permitido hasta converger ambos valores hacia la unidad (diseño óptimo).

Igualmente se representó en forma de líneas la acumulación del desplazamiento relativo y la masa relativa. A medida que evoluciona el programa los individuos evaluados van disminuyendo su valor relativo acumulado hasta llegar a un valor de 2 unidades. No obstante existen algunos diseños iniciales con valores próximos a 2 que serían tan buenos como el diseño óptimo final si se tratara de un problema multiobjetivo, pero que en este caso no son factibles por no cumplir la restricción de desplazamiento inferior a 15mm.

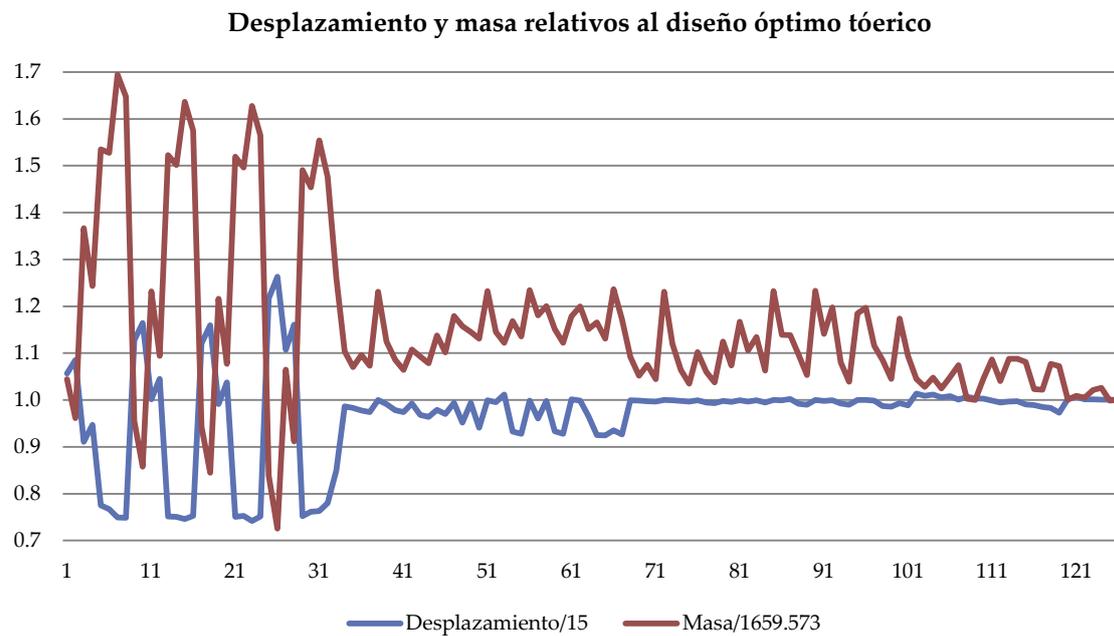


Tabla 16. Desplazamiento y masa de los puntos simulados relativos a los valores obtenidos en el diseño óptimo (5 variables).

Acumulación del desplazamiento~masa relativos al diseño óptimo

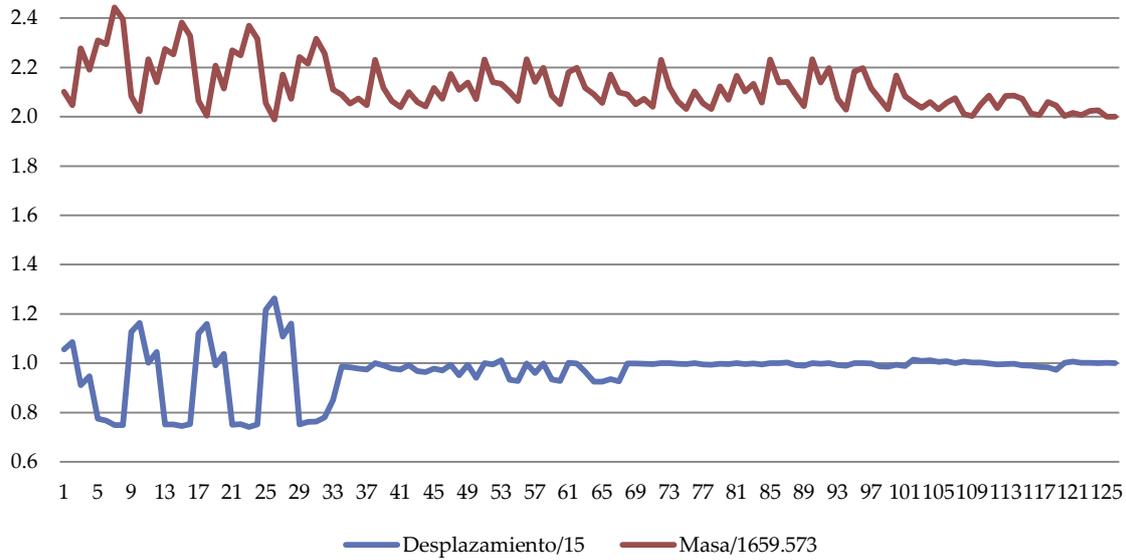
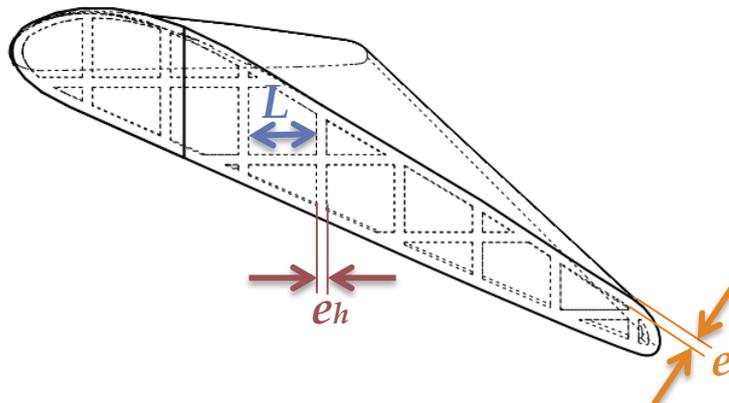


Tabla 17. Valor acumulado de desplazamiento y masa relativos de los diseños evaluados (5 variables).

7.2.3. Resultados obtenidos para el problema con 3 variables de diseño (evaluaciones con elementos finitos)

La siguiente prueba realizada con esta versión del programa se llevó a cabo sobre el problema de referencia pero con 3 variables de diseño, correspondientes al lado de los huecos en forma de cubos ( $L = 20\sim 60\text{mm}$ ), el espesor exterior de la pieza ( $e = 3\sim 8\text{mm}$ ) y el espesor entre huecos ( $e_h = 3\sim 8\text{mm}$ ).



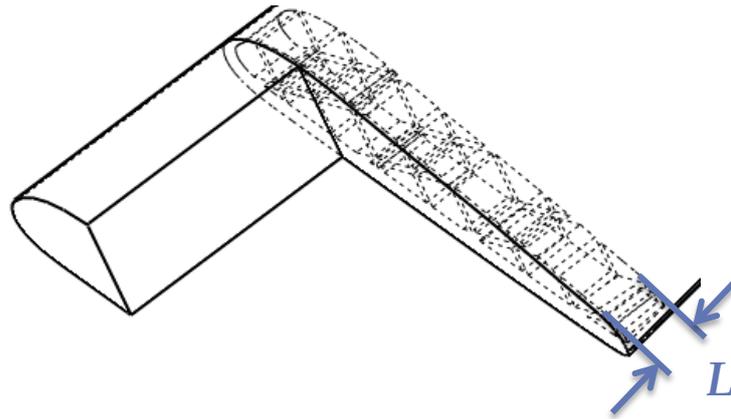


Ilustración 28. Variables de diseño del problema de referencia (3 variables).

En los casos anteriores se han sustituido las simulaciones por elementos finitos por unas ecuaciones “representativas” simplemente para agilizar las pruebas de las diferentes estrategias programadas en Matlab. Sin embargo en este caso se pretende realizar una prueba más real llevando a cabo el cálculo por elementos finitos de los diseños que el programa estime oportunos. Con la geometría definida y parametrizada en base a las variables de diseño, los parámetros de mallado establecidos, y los sensores de almacenamiento de los resultados de interés (desplazamiento máximo y masa) también definidos, se ejecuta el programa. Éste interactúa con el usuario a través de la ventana de comandos del Matlab, solicitando en primer lugar la información inicial para contextualizar el problema, como el número de variables de diseño, el límite inferior y superior de cada variable, o el número de restricciones del problema y el valor límite de cada una. Con estos datos iniciales el programa empieza a pedir en pantalla el resultado de diferentes diseños, incluyendo evidentemente la información de la cadena de cromosomas para que el usuario modifique la geometría de la pieza. Estos valores se introducen en el diseño parametrizado y se ejecuta el análisis correspondiente por elementos finitos. Una vez finalizado el cálculo se introducen en la ventana de comandos del Matlab los valores medidos por los sensores del programa de cálculo por elementos finitos. Estos valores introducidos por el usuario son leídos y almacenados por el programa para su posterior uso en el algoritmo implementado. Este proceso se repite para cada uno de los puntos que el programa pretenda evaluar hasta mostrar en pantalla el diseño óptimo y los resultados correspondientes calculados con elementos finitos.

Tras ejecutar el programa e introducir todos los resultados pedidos en pantalla, el diseño óptimo obtenido fue (tras 40 simulaciones):

**DISEÑO ÓPTIMO: 39.875 4.091 3.000**  
**Desplazamiento: 14.992mm**  
**Masa: 1632.55g**

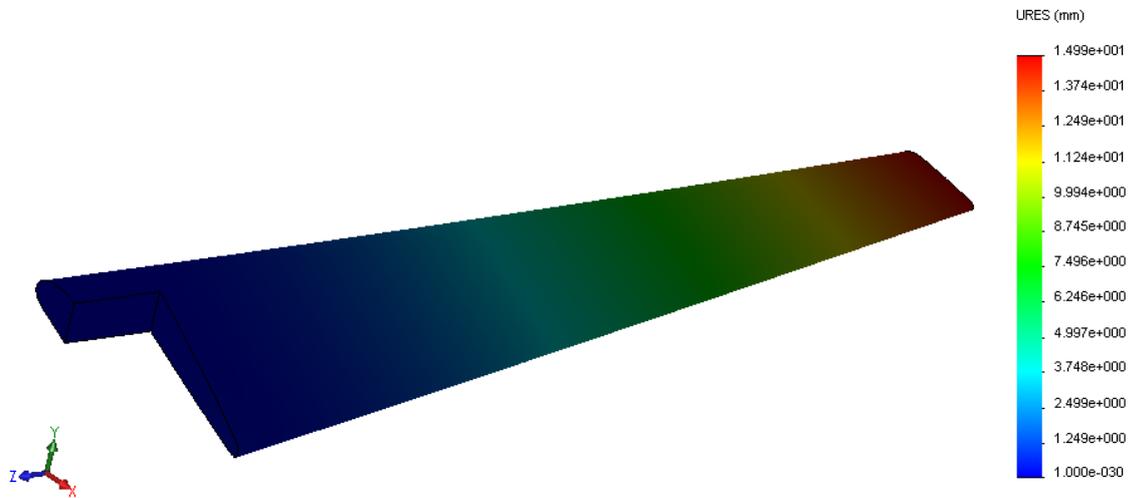


Ilustración 29. Campo de desplazamientos del diseño óptimo.

Para comparar este resultado obtenido se ejecutó el diseño de optimización disponible en el SolidWorks (diseño de experimentos de Box-Behnken y superficies de respuesta), obteniendo el siguiente resultado (tras 14 simulaciones):

**DISEÑO ÓPTIMO (SolidWorks): 47.532 4.496 3.193**  
**Desplazamiento: 14.652mm**  
**Masa: 1690.07g**

		Actual	Inicial	Óptimo
lado		47.531738	40.000000	47.531738
espesor		4.49628mm	6mm	4.49628mm
espesorentrehuecos		3.193176	6.000000	3.193176
Desplazamiento1	< 15mm	14.65185mm		14.65185mm
Masa1	Minimizar	1690.07 g		1690.07 g

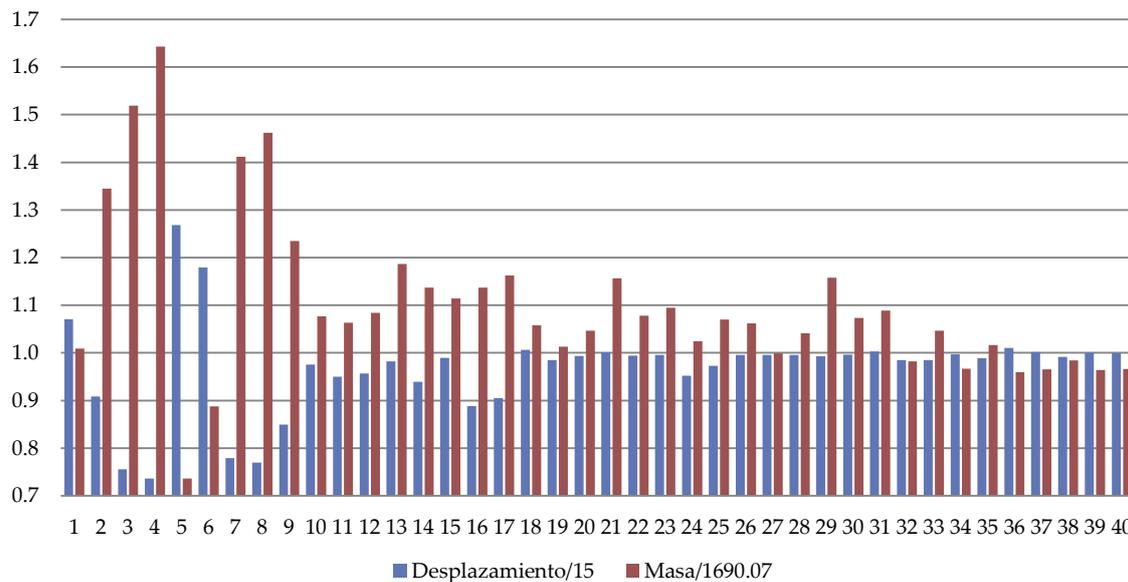
Ilustración 30. Resultado optimización SolidWorks para el problema de referencia (3 variables).

El resultado obtenido por el programa desarrollado mejora el resultado obtenido mediante la optimización del SolidWorks (1632.55g frente a 1690.07g). También se aprecia que el diseño óptimo según el SolidWorks no está “explotado” al máximo ya que el desplazamiento obtenido (14.652mm) no está al límite de la restricción (15mm). Sin embargo, el programa propuesto requiere de un total de 40 diseños evaluados, mientras que la optimización del SolidWorks sólo requiere 14

evaluaciones (13 del diseño de experimentos y la evaluación del óptimo según la superficie de respuesta).

En las siguientes gráficas se muestra el desplazamiento y masa relativos de los puntos evaluados en la ejecución del programa. Dado que no se tiene un valor certero del diseño más optimizado, se tomó como referencia 15mm para el desplazamiento (valor límite) y 1690.07g para la masa (peso del diseño óptimo obtenido en la optimización del SolidWorks).

**Desplazamiento y masa relativos**

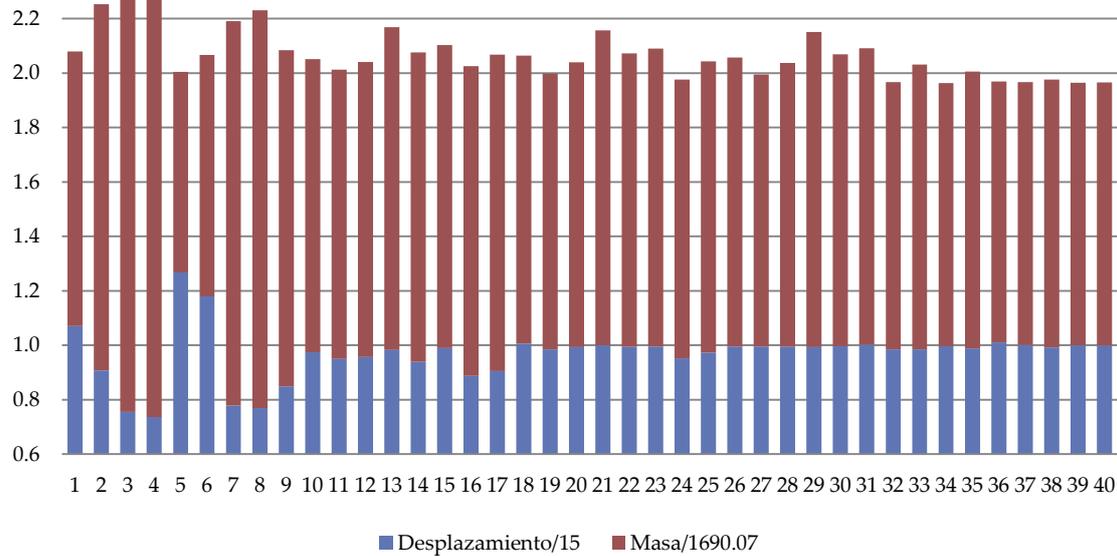


**Gráfica 9. Desplazamiento y masa relativos de los puntos simulados (5 variables y cálculo con MEF).**

A medida que evoluciona el programa los resultados de desplazamiento se van aproximando a la unidad (valor máximo factible), mientras que la masa relativa va disminuyendo e incluso toma valores inferiores a la unidad en los últimos puntos evaluados, lo cual indica que se trata de diseños con menor masa que la del diseño óptimo del SolidWorks.

En la siguiente gráfica se muestra el valor acumulado de desplazamiento relativo y masa relativa de los puntos evaluados. Los menores valores se obtienen para los últimos puntos, siendo éstos por tanto los más eficientes desde el punto de vista de rigidez/peso.

Acumulación del desplazamiento~masa relativos



Gráfica 10. Valor acumulado de desplazamiento y masa relativos de los diseños evaluados (5 variables y cálculo con MEF).

#### 7.2.4. Resultados obtenidos para el problema con 4 variables de diseño (evaluaciones con elementos finitos)

La misma prueba anterior se repitió con el problema de referencia con 4 variables de diseño, (evaluaciones mediante elementos finitos). El óptimo al que llega el programa tras 79 simulaciones fue:

**DISEÑO ÓPTIMO: 37.195 4.110 3.000 60.000**  
**Desplazamiento: 14.879mm**  
**Masa: 1621.47g**

Tras observar el resultado cabe destacar que el hecho de incluir una variable de diseño más (la altura de los huecos con forma de prismas cuadrados) ha permitido reducir ligeramente la masa frente al óptimo obtenido en el caso anterior (de 1632.55g con 3 variables de diseño a 1621.47g con una variable más).

Ejecutando la optimización del SolidWorks para este problema con 4 variables de diseño, se obtiene, tras 26 simulaciones:

**DISEÑO ÓPTIMO (SolidWorks): 46.830 4.479 3.13 59.347**  
**Desplazamiento: 14.709mm**  
**Masa: 1678.64g**

		Actual	Inicial	Óptimo
lado		46.829834	40.000000	46.829834
espesor		4.47873mm	5.5mm	4.47873mm
espesorentrehuecos		3.312653	5.500000	3.312653
h		59.34692mm	40mm	59.34692mm
Desplazamiento1	< 15mm	14.70907mm	12.74209mm	14.70907mm
Masa1	Minimizar	1678.64 g	2087.14 g	1678.64 g

Ilustración 31. Resultado optimización SolidWorks para el problema de referencia (4 variables).

Nuevamente el resultado al que llega el programa implementado supera al obtenido por la optimización del SolidWorks ya que reduce la masa de 1678.64g a 1621.47g cumpliendo la restricción de desplazamiento. Sin embargo se requieren más simulaciones (79 frente a 26) por lo que se decidió analizar qué puntos de los calculados son menos influyentes en la convergencia del método y en qué fase del programa se podría mejorar la eficiencia del programa sin repercutir en la calidad de la solución final.

### 7.2.5. Reducción del número de puntos evaluados

Tras comprobar que la metodología alcanza resultados relativamente buenos pero con un elevado número de puntos de muestreo, se plantearon nuevas ideas para reducir el número de evaluaciones y conseguir un programa más eficiente. Para ello se trabajó inicialmente sobre el problema de referencia con cuatro variables (sin evaluaciones mediante elementos finitos para agilizar las pruebas).

La primera modificación planteada fue realizar la adición de puntos de aproximación a la frontera de manera reiterada en cada arista o dirección interior. Anteriormente esta fase se realizaba en un bucle en el que cada iteración conllevaba la adición de un nuevo punto en todos los frentes del interior y exterior, repitiendo este proceso siempre que el error medio absoluto (en términos porcentuales) de los puntos añadidos fuera superior al 1%. Por tanto, todas las aristas del contorno con un tramo en zona factible y otro en zona no factible tendrán el mismo número de puntos asociados a esta etapa de la metodología aunque en una de ellas la precisión sea mucho mayor.

Sin embargo en esta nueva propuesta se añaden nuevos puntos en una misma dirección hasta que la desviación con respecto al límite de la frontera sea inferior al 1%.

De este modo, en una arista en la que se identifique más rápidamente el límite entre espacio factible y no factible se añadirán menos puntos que en el caso de otra arista en la que la estimación de la frontera factible/no factible sea menos precisa, ya que se necesitarían más iteraciones para aproximarse a la frontera a una distancia tal que el error cometido en la restricción crítica sea menor al error máximo permitido.

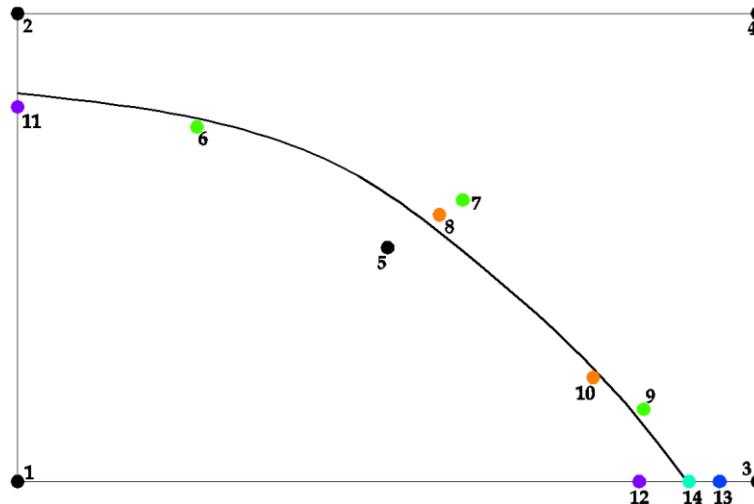


Ilustración 32. Nueva estrategia de adición de puntos de aproximación a la frontera.

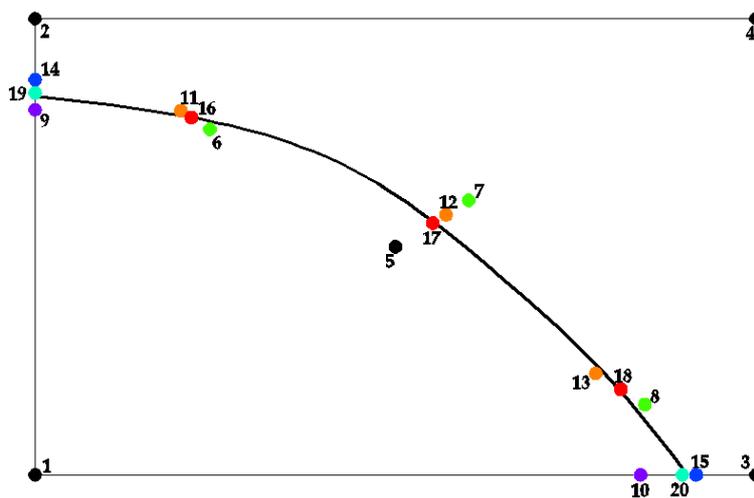


Ilustración 33. Estrategia anterior de adición de puntos de aproximación a la frontera.

Otro aspecto que se modificó para disminuir el número de simulaciones fue añadir un único punto en cada dirección durante la adición de puntos en la aproximación a la frontera por el interior. La precisión en este caso no es tan importante como en la aproximación por el exterior ya que no repercute en ninguna decisión del algoritmo. Además, el muestreo por las aristas es más importante ya que normalmente las soluciones óptimas se encuentran en el contorno.

Con estas nuevas modificaciones se ejecutó el programa para el problema de referencia con 4 variables (sin cálculo por elementos finitos) obteniendo los siguientes resultados:

Ejecución	L (mm)	e (mm)	e <sub>n</sub> (mm)	h (mm)	Nº puntos simulados	Desplazamiento (mm)	Masa (g)
1	42.462	4.345	3.000	59.836	56	14.999	1602.176
2	40.972	4.309	3.000	60.000	56	14.997	1604.237
3	42.896	4.383	3.000	59.099	58	14.967	1606.800

Tabla 18. Resultados obtenidos en 3 ejecuciones del programa modificado con un solo punto en cada dirección de aproximación a la frontera por el interior y adición de puntos hasta cumplir el error en cada arista implicada en la aproximación por las aristas.

Los resultados obtenidos son buenos en los 3 casos por estar muy próximos a la masa del diseño óptimo teórico (1600.809g). El diseño óptimo es similar al alcanzado con la versión anterior del programa pero se reduce el número de puntos simulados de 61 a 56.

En base a la lista de puntos evaluados en la última ejecución anterior, se decidió aplicar el algoritmo genético de optimización a partir de esta nube de puntos modificada. El objetivo perseguido en este caso es eliminar diferentes puntos de dicha muestra para ver su influencia en el resultado óptimo al que converge el AG y así tratar de reducir en la medida de lo posible el número de puntos de muestreo necesarios para la metodología.

Dado que anteriormente se eliminaron algunos puntos de aproximación a la frontera por el interior y el óptimo alcanzado no empeoró con respecto a pruebas anteriores, se propuso prescindir totalmente de estos puntos. Por otro lado, también se decidió excluir algunos de los puntos añadidos al combinar la mejor esquina de la frontera con el resto de esquinas contiguas. Concretamente se decidió mantener aquellos puntos que surgieron a partir del valor medio entre la esquina de referencia y las “n-1” (“n”=nº de variables de diseño) esquinas contiguas con menor masa, eliminando el resto de combinaciones. Además, también se prescindió de los puntos

correspondientes al AG de aproximación a la frontera. Eliminando esta lista de puntos y aplicando el algoritmo genético de optimización, el punto óptimo obtenido fue:

L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
41.238	4.315	3.000	60.000	14.998	1603.736

Tabla 19. Resultado óptimo del AG sin los puntos de aproximación a la frontera por el interior y con solo las combinaciones de la mejor esquina de la frontera con las “n-1” mejores contiguas.

Se aprecia que el AG llega a una solución muy cercana al óptimo teórico en términos de masa con sólo 42 simulaciones. Sin embargo también hay que decir que en este problema concreto uno de los puntos añadidos entre la mejor esquina de la frontera y el resto contiguas está muy próximo al óptimo teórico. Por ello, se eliminó de la nube de datos y se ejecutó de nuevo el AG, obteniendo el siguiente resultado:

L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
50.474	4.687	3.000	50.698	14.852	1641.985

Tabla 20. Resultado óptimo del AG con las mismas condiciones anteriores pero sin incluir el mejor punto de la combinación entre la mejor esquina de la frontera y el resto de esquinas contiguas.

Se observa que el resultado empeora casi 40g. Sin embargo, si se incluye este último punto en la nube de datos y se ejecuta de nuevo el AG, el óptimo obtenido es:

L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
46.102	4.519	3.000	47.622	14.995	1617.616

Tabla 21. Resultado óptimo del AG al incluir el óptimo obtenido anteriormente.

Dado que el resultado mejora significativamente con respecto al anterior, se decidió mantener el AG de aproximación a la frontera ya que ayudará a mejorar el ajuste del metamodelo y a explorar el dominio próximo a la frontera.

### 7.3. Tercer programa

La nueva versión del programa se creó a partir del código del algoritmo anterior pero incluyendo las nuevas modificaciones (exclusión de la aproximación a la

frontera por el interior, nueva estrategia de aproximación a la frontera por las aristas una a una y combinación de la mejor esquina de la frontera con las “n-1” mejores contiguas). También se decidió realizar la estimación del segundo punto de aproximación a la frontera por las aristas empleando una ecuación de orden 2 que se define a partir de las dos esquinas de referencia y del último punto añadido. Con esto se pretende estimar con mayor precisión el punto donde la arista pasa de zona factible a zona no factible y así tratar de reducir el número de puntos simulados en esta fase de aproximación a la frontera. A pesar de que se podría ir aumentando el orden de la ecuación de ajuste a medida que se simulan más puntos sobre la arista, se descartó esta opción porque al incrementar el orden del polinomio también pueden producirse oscilaciones que conllevarían a cometer un gran error de predicción. Por ello el resto de puntos se calculan mediante interpolación lineal.

El pseudocódigo sería:

- 1) Muestreo inicial  $2^{n+1}$  (esquinas y punto central).
- 2) Aproximación a la frontera con la adición de nuevos puntos entre esquinas del dominio contiguas que se encuentren en espacios opuestos (factible/no factible). Las coordenadas del punto añadido se determinan mediante interpolación lineal de las restricciones (media ponderada) de los dos puntos más cercanos a la frontera. Sin embargo, la estimación de las coordenadas del segundo punto añadido entre las esquinas se realizará mediante el ajuste de una ecuación polinómica de orden 2 determinando los 3 coeficientes de la ecuación mediante los dos esquinas y el primer punto añadido previamente. La aproximación se realiza arista por arista hasta tener una desviación frente al valor límite de la restricción crítica menor del 1%.
- 3) Adición de puntos intermedios entre la mejor esquina de la frontera y las “n-1” mejores contiguas.
- 4) Algoritmo genético de aproximación a la frontera. Se repite reiteradamente siempre que se haya ejecutado menos de “n” veces o si el error medio de la estimación de los resultados de las restricciones del punto óptimo al que converge el algoritmo difiere más del 1%. El mejor individuo alcanzado por el algoritmo se simula y se incluye en la nube de datos para refinar el metamodelo y trabajar con mayor precisión en las siguientes ejecuciones. Los individuos próximos a puntos evaluados en esta fase se penalizan para favorecer la exploración.
- 5) Algoritmo genético de optimización final. Con la nube de puntos elaborada en las etapas anteriores se reconstruye el metamodelo de aproximación para evaluar la aptitud de los individuos. Finalmente éste converge hacia un óptimo que se simula. En el caso de que no cumpla las restricciones, se actualiza el metamodelo con el último punto evaluado y se ejecuta de nuevo el algoritmo en un proceso iterativo hasta conseguir un diseño factible.
- 6) Búsqueda del mejor individuo simulado (óptimo).

### 7.3.1. Resultados obtenidos para el problema con 4 variables de diseño

En la siguiente tabla se muestra el diseño óptimo alcanzado en 10 ejecuciones diferentes y el número de simulaciones realizadas en cada caso:

Ejecución	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Nº puntos simulados	Desplazamiento (mm)	Masa (g)
1	42.804	4.357	3.000	60.000	43	14.986	1603.384
2	42.251	4.347	3.000	60.000	43	14.982	1604.474
3	40.930	4.313	3.000	60.000	43	14.990	1605.271
4	41.073	4.313	3.000	59.896	44	14.997	1604.143
5	40.928	4.309	3.000	60.000	44	14.997	1604.392
6	50.749	4.531	3.000	60.000	43	14.986	1612.621
7	41.101	4.312	3.000	60.000	43	14.998	1603.866
8	40.724	4.310	3.000	60.000	45	14.986	1606.188
9	41.117	4.316	3.000	60.000	44	14.991	1604.853
10	39.331	4.287	3.000	60.000	43	14.971	1611.309

Tabla 22. Resultados óptimos del tercer programa en el problema de referencia (4 variables).

Se aprecia que el diseño alcanzado en las diferentes ejecuciones está próximo al óptimo teórico en términos de masa. Los resultados de la optimización con este programa son similares a los obtenidos con versiones anteriores pero con una mejora significativa en el número de puntos evaluados (de 61 puntos en versiones anteriores a 43 puntos con las modificaciones implementadas en el código).

### 7.3.2. Resultados obtenidos para el problema con 3 variables de diseño (evaluaciones con elementos finitos)

Dado que el programa implementado parece funcionar correctamente, se ejecutó para el caso de 3 variables de diseño con evaluaciones mediante cálculo por elementos finitos. El diseño óptimo al que llega la metodología es el siguiente:

**DISEÑO ÓPTIMO: 60.000 4.540 3.000**  
**Desplazamiento: 14.975mm**  
**Masa: 1634.85g**

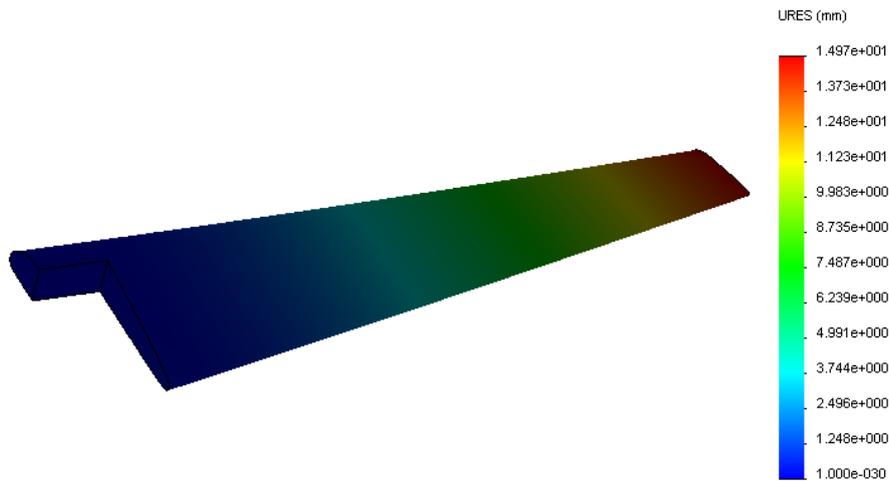


Ilustración 34. Campo de desplazamientos del diseño óptimo obtenido.

El óptimo conseguido se asemeja bastante en masa al obtenido con la versión anterior del programa (1634.85g frente a 1632.55g en la versión anterior) pero con 11 evaluaciones menos (29 puntos frente a 40 puntos anteriormente), mejorando significativamente la eficiencia de la metodología con las modificaciones introducidas. También cabe destacar que el diseño óptimo obtenido en este caso difiere sustancialmente del anterior, por lo que existen diferentes combinaciones de variables que proporcionan resultados similares de desplazamiento y masa.

Por otro lado, la masa del diseño óptimo lograda por el paquete de optimización del SolidWorks era de 1690.07g, mientras que con la metodología actual se obtiene un diseño de 1634.85g. Sin embargo, el número de simulaciones necesarias en la optimización del SolidWorks es de sólo 14, mientras que el programa desarrollado requiere 29 simulaciones.

En la siguiente tabla se muestran los 29 puntos evaluados durante el proceso de optimización:

Punto	L (mm)	e (mm)	e <sub>h</sub> (mm)	Desplazamiento (mm)	Masa (g)
1	20.000	3.000	3.000	16.061	1705.09
2	20.000	3.000	8.000	13.625	2273.12
3	20.000	8.000	3.000	11.338	2567.12
4	20.000	8.000	8.000	11.046	2777.22
5	60.000	3.000	3.000	19.025	1244.24
6	60.000	3.000	8.000	17.693	1499.71
7	60.000	8.000	3.000	11.684	2386.56

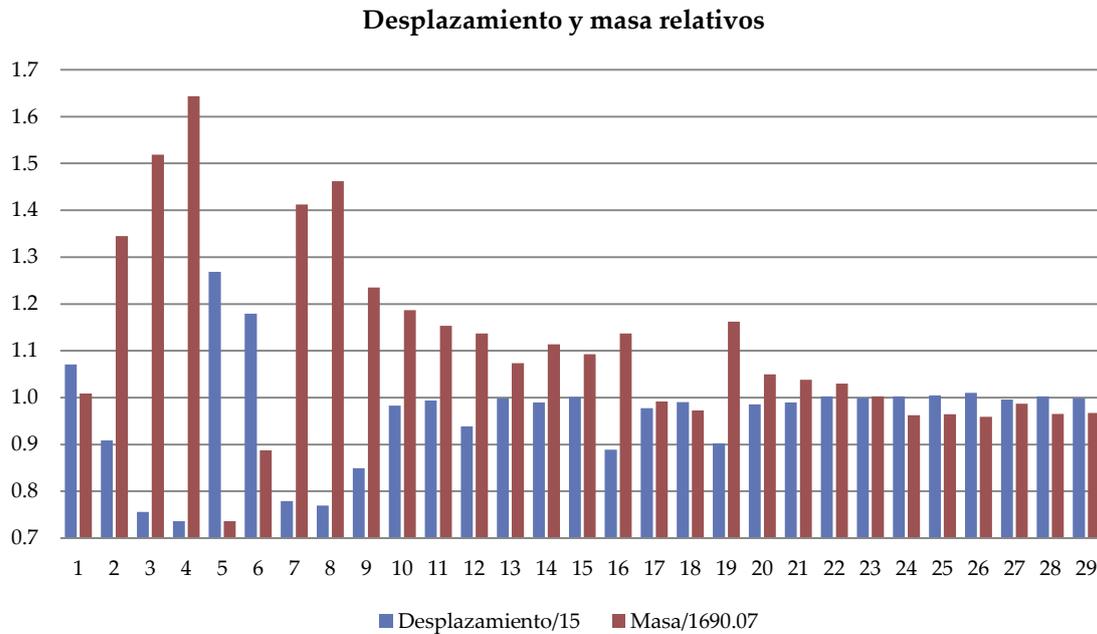
8	60.000	8.000	8.000	11.543	2470.68
9	40.000	5.500	5.500	12.741	2087.14
10	20.000	3.000	5.178	14.745	2005.20
11	20.000	3.000	4.700	14.905	1948.70
12	20.000	4.123	3.000	14.074	1921.50
13	20.000	3.562	3.000	14.975	1814.40
14	33.520	3.000	8.000	14.842	1882.70
15	35.157	3.000	8.000	15.015	1847.10
16	60.000	5.741	3.000	13.331	1921.20
17	60.000	4.709	3.000	14.664	1676.10
18	60.000	4.577	3.000	14.851	1643.70
19	60.000	5.190	8.000	13.536	1964.30
20	60.000	4.272	8.000	14.786	1774.50
21	60.000	4.178	8.000	14.847	1754.60
22	60.000	4.115	8.000	15.045	1741.10
23	60.000	4.346	5.500	14.987	1694.90
24	40.000	4.070	3.000	15.040	1626.80
25	60.000	4.521	3.000	15.070	1630.30
26	48.904	4.285	3.000	15.154	1620.90
27	31.330	3.858	3.000	14.943	1668.40
28	56.705	4.479	3.000	15.043	1631.60
29	60.000	4.540	3.000	14.975	1634.85

Tabla 23. Puntos evaluados por elementos finitos durante la aplicación de la metodología.

Los primeros 9 puntos se corresponden al diseño de experimentos factorial de 2 niveles (esquinas del dominio) y al punto central. Del 10 al 22 son los puntos de aproximación a la frontera factible/no factible por las aristas del dominio. Dentro de estos 12 puntos se pueden diferenciar 5 grupos (en diferentes colores) asociados a cada una de las 5 aristas del contorno que están divididas en zona factible y no factible. Posteriormente se selecciona la esquina de la frontera con menor masa. Las esquinas de la frontera se corresponden con el último punto de cada grupo, de modo que se debe elegir entre el punto 11, 13, 15, 18 y 22. En este caso el punto 18 es el de menor masa, por lo que se combinará con las “n-1” (2) mejores esquinas de la frontera contiguas, que daría lugar a los puntos 23 y 24. Como se aprecia, sólo las esquinas de la frontera asociadas a los puntos 13 y 22 son contiguas a la mejor esquina (punto 18), por lo que se combinará el punto 18 con el 13 y el 22. De este modo, el punto 23 surge de la media entre el punto 18 y el 22, y el punto 24 de la media entre el 18 y el 13. Posteriormente los puntos 25, 26 y 27 se corresponden a los añadidos durante el AG de aproximación a

la frontera, y por último, los puntos 28 y 29 se corresponden a los simulados en el AG de optimización final.

En la siguiente gráfica se representan los valores relativos de desplazamiento y masa de cada uno de los 29 diseños evaluados (tomando como referencia 15mm para el desplazamiento y 1690.07g para la masa –masa del diseño óptimo del SolidWorks-).



Gráfica 11. Desplazamiento relativo y masa relativa de los puntos calculados durante el proceso de optimización.

A pesar de que la metodología funciona correctamente, el número de puntos de simulados es demasiado elevado con respecto a la optimización del SolidWorks, por lo que se realizaron nuevas modificaciones para disminuir el número de puntos de muestreo.

#### 7.4. Cuarto programa

Observando los puntos evaluados en la prueba anterior se optó por tratar de disminuir el número de simulaciones en la fase de adición de puntos para la aproximación a la frontera. Esta decisión se tomó debido a que en esta parte del algoritmo fueron necesarias un total de 13 evaluaciones de los 29 puntos totales, siendo

la fase con mayor intensidad de muestreo. A pesar de que sería conveniente reducir el número de simulaciones en esta fase, se considera muy interesante esta estrategia de evaluar puntos próximos a la frontera factible/no factible en las aristas del contorno del dominio, ya que normalmente las soluciones óptimas están en el contorno. Por ejemplo en la prueba anterior el punto óptimo alcanzado (29) está justamente en la arista donde se encontró la mejor esquina de la frontera (punto 18), y muy próximos entre sí. Es por ello por lo que se consideró oportuno seguir realizando esta etapa del método pero tratando de minimizar el número de puntos de muestreo requeridos.

Tras analizar los resultados de las múltiples pruebas realizadas con el problema de referencia (con 3, 4 y 5 variables de diseño) se encontró una cierta relación entre la mejor esquina de la frontera y la arista correspondiente. Tanto para el caso de 3 variables de diseño como para 4 y 5 variables, se observó que la mejor esquina de la frontera entre espacio factible y no factible se encuentra en una arista que contiene al punto de menor masa de los implicados en las aristas con tramo factible y no factible. En otras palabras, si se agrupan todas las esquinas del dominio que forman parte de una arista “cortada” por la frontera entre espacio factible y no factible, la mejor esquina de la frontera estará en una arista común al punto de menor masa de los anteriores. Por ello se plantea identificar este punto y realizar la aproximación a la frontera sólo por las aristas que parten de esta esquina. Posteriormente se combina la mejor esquina de la frontera con el resto contiguas, aunque en algunos casos puede que sólo se tenga una esquina de la frontera y por tanto no se añada ningún punto.

Esta explicación anterior se entiende mejor con los dos ejemplos siguientes a modo de imagen esquemática. En primer lugar se identifican las aristas cortadas por la frontera entre espacio factible y no factible (imagen superior izquierda) y las esquinas de dichas aristas (aristas y esquinas implicadas, en rojo). Posteriormente se escoge la esquina de menor masa entre las seleccionadas anteriormente (imagen superior derecha, esquina marcada en rojo). Tras identificar la esquina del dominio, se realiza la aproximación a la frontera por las aristas comunes a esta esquina, añadiendo nuevos puntos (imagen inferior izquierda, puntos en violeta). Finalmente (imagen inferior derecha) se realiza la adición de nuevos puntos (en naranja) entre la esquina de la frontera de menor masa (punto violeta marcado en verde) y el resto de esquinas de la frontera evaluadas previamente (que sean contiguas, es decir, que compartan cara del contorno). Los nuevos puntos se ubican en el punto medio de las dos referencias.

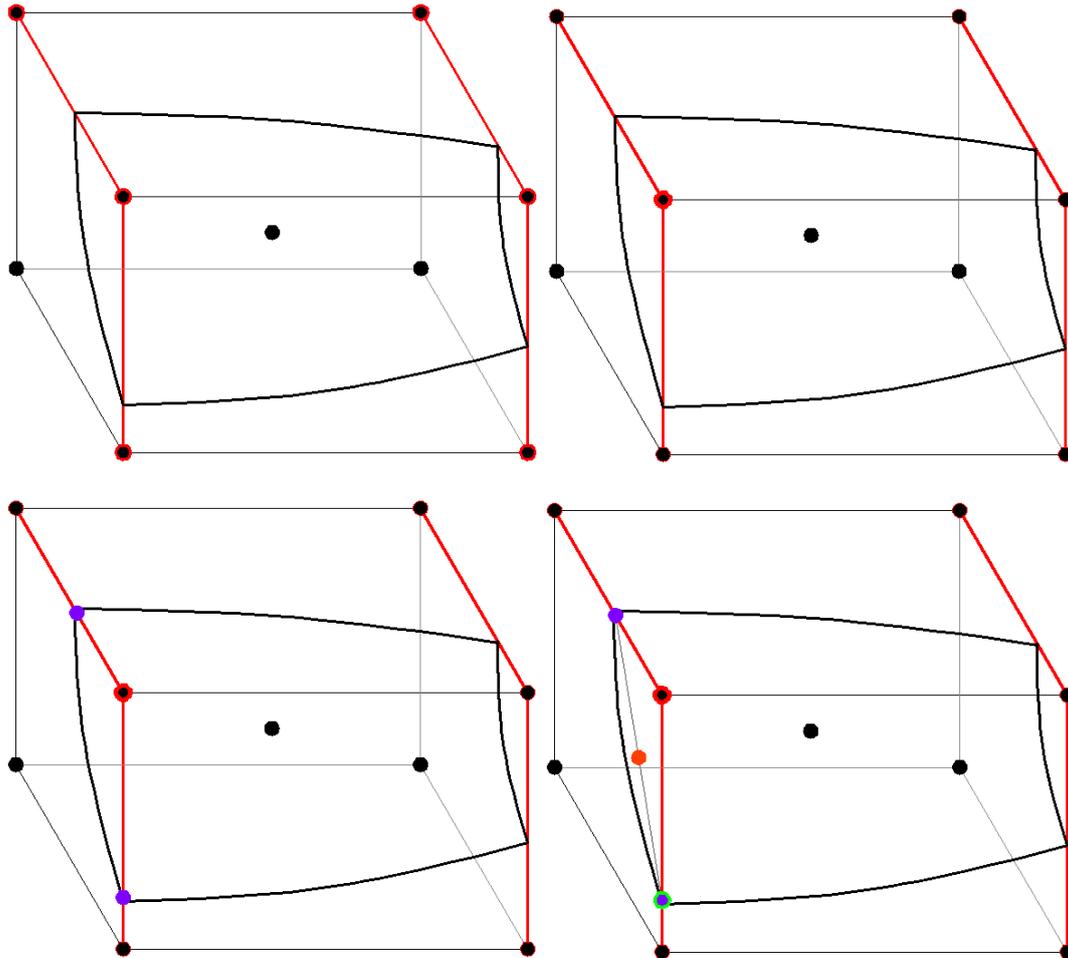


Ilustración 35. Nueva estrategia de adición de puntos de aproximación a la frontera y adición de nuevos puntos (en este caso 1) entre la mejor esquina de la frontera y el resto de esquinas contiguas evaluadas.

Dependiendo del caso, la mejor esquina de la frontera tendrá una serie de esquinas evaluadas contiguas, por lo que se añadirán nuevos puntos (punto naranja del ejemplo anterior) o no dependiendo de la situación. Así por ejemplo, en el caso anterior la mejor esquina de la frontera sólo tiene una esquina evaluada y contigua, por lo que se añadirá un punto más entre ambas esquinas de la frontera. Sin embargo en otras situaciones como en la de las siguientes imágenes se pueden encontrar varias esquinas de la frontera evaluadas que sean contiguas a la esquina de referencia, por lo que se añadirán tantos puntos como esquinas contiguas evaluadas.

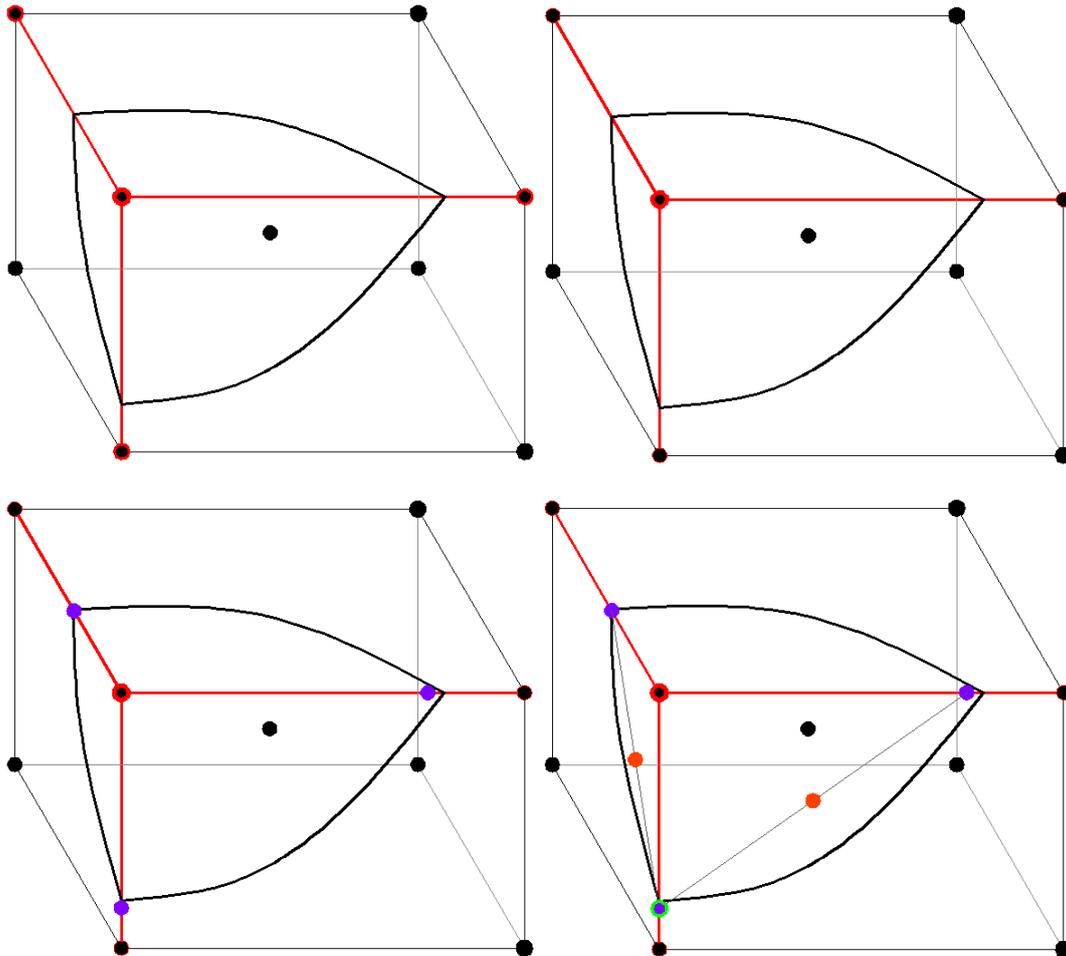


Ilustración 36. Ejemplo similar al anterior. En este caso existen 2 esquinas de la frontera evaluadas y contiguas a la mejor esquina, por lo que se añaden 2 nuevos puntos (en naranja).

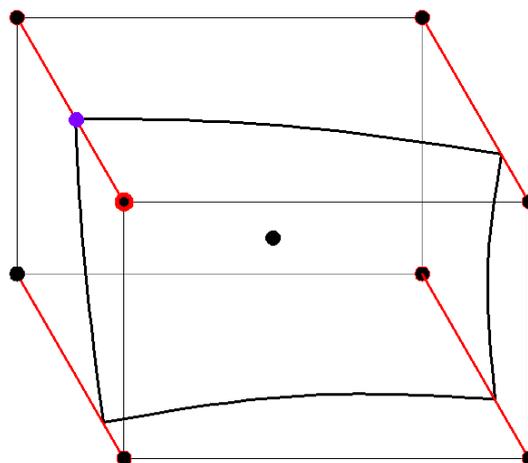
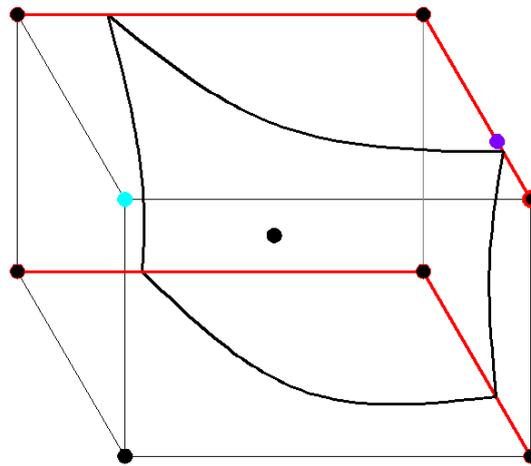


Ilustración 37. Otro ejemplo. En este caso sólo se realiza la aproximación a la frontera por una arista, por lo que no se añaden nuevos puntos de la combinación entre esquinas contiguas de la frontera.

También se observó que la esquina del dominio empleada como esquina de referencia para realizar la aproximación a la frontera por las aristas coincide en todos los casos con la esquina del dominio de menor masa. Por ello, para identificar las aristas de aproximación a la frontera se podría pensar en localizar directamente la esquina del dominio de menor masa. Sin embargo las aristas que parten de la esquina de menor masa (punto azul en la siguiente imagen) no estarán siempre “cortadas” por la frontera entre espacio factible y no factible, por lo que este planteamiento no siempre funcionaría correctamente.



**Ilustración 38.** La esquina del dominio de menor masa (azul) no forma parte de una arista “cortada” por la frontera. Por ello la esquina de referencia (marcada en rojo) no coincide con la de menor masa del dominio.

Por otro lado, dado que el número de puntos evaluados con las nuevas modificaciones es relativamente bajo para crear un metamodelo con buena aproximación, se decidió cambiar el criterio de parada del AG de aproximación a la frontera. Se evaluarán al menos “n+1” puntos, uno más que en la versión anterior para mejorar el ajuste del metamodelo.

Otro cambio importante en el AG de aproximación a la frontera es que los individuos de las poblaciones generadas para determinar el punto “n+1” y los siguientes puntos añadidos en esta fase no se penalizarán por cercanía a puntos simulados previamente. Con esta modificación se pretende por un lado evaluar al menos un punto obtenido sin penalizar a los individuos justo antes de aplicar el AG final, y por otro lado que el cálculo del error de estimación se realice sobre puntos en los alrededores del punto óptimo y no alejados de éste por la penalización por cercanía ya que interesa un buen ajuste del metamodelo en las zonas próximas al óptimo,

siendo menos importante en el resto del dominio. Una vez que se hayan añadido “n+1” puntos en esta parte de la metodología, se comprueba el error cometido en las estimaciones del último punto añadido y en caso de ser inferior al 1% se finaliza este AG de aproximación y se continúa con el AG final. Anteriormente el error cometido en las estimaciones se calculaba teniendo en cuenta únicamente el valor de las restricciones (valores estimados frente a valores simulados). Sin embargo en el nuevo programa se tendrá en cuenta también el error cometido en la estimación de la masa, por lo que se exigirá un metamodelo con mayor precisión pero en puntos próximos al óptimo ya que no se penaliza por cercanía.

En cuanto al criterio de parada del AG de optimización, en la versión de programa anterior se detenía el proceso si éste convergía a un punto que resultaba ser factible tras las simulaciones. Sin embargo en este caso se decidió repetir el AG hasta que el punto simulado mejore al mejor diseño que se tenía antes de aplicar el AG final de optimización. De este modo lo que se pretende es garantizar que el AG de optimización mejore al diseño óptimo obtenido en las fases previas de la metodología.

Además también se modificó el código de los dos AG del programa (aproximación a frontera y optimización final) para evitar que en el caso de converger hacia un punto ya simulado anteriormente el programa vuelva a simularlo.

Por último, tras unas primeras pruebas del nuevo programa se observó que el primer punto simulado en el AG de aproximación a la frontera era prácticamente idéntico al mejor de los diseños evaluados antes de aplicar este AG de aproximación a la frontera. Como el objetivo de esta fase es explorar por la frontera, el hecho de simular dos puntos casi idénticos no aporta información relevante para mejorar el metamodelo y mucho menos para explorar. Por ello se decidió penalizar a los individuos próximos al mejor punto simulado hasta el momento para evitar que el AG converja a un punto muy cercano. Esta penalización sólo se aplica en la primera iteración del AG de aproximación a la frontera, de modo que en el resto de iteraciones se penalizará la cercanía a puntos añadidos en esta fase hasta que se llegue a la iteración “n+1” en la que ya no se aplicará penalización de este tipo.

El pseudocódigo correspondiente al nuevo programa sería el siguiente:

- 1) Muestreo inicial  $2^{n+1}$  (esquinas y punto central).
- 2) Aproximación a la frontera con la adición de nuevos puntos. En este nuevo programa se identifican las aristas “cortadas” por la frontera entre espacio factible y no factible y de las esquinas del dominio involucradas en dichas arista se toma como referencia la de menor masa. Posteriormente se añaden nuevos puntos en las aristas que parten de esta esquina y que están “cortadas” por la frontera. Las

- coordenadas del punto añadido se determinan mediante interpolación lineal de las restricciones (media ponderada) de los dos puntos más cercanos a la frontera. Sin embargo, la estimación de las coordenadas del segundo punto añadido entre las esquinas se realizará mediante el ajuste de una ecuación polinómica de orden 2 determinando los 3 coeficientes de la ecuación mediante las dos esquinas y el primer punto añadido. Este proceso se realiza arista por arista hasta tener una desviación menor del 1% frente al valor límite de la restricción crítica.
- 3) Adición de puntos intermedios entre la mejor esquina de la frontera y las esquinas contiguas evaluadas. (Esta parte de la metodología no aportará nuevos puntos en los casos en los que sólo se haya realizado la aproximación a la frontera por una única arista).
  - 4) Algoritmo genético de aproximación a la frontera. Se repite reiteradamente siempre que se haya ejecutado menos de “n+1” veces o si el error medio de la estimación de los resultados de las restricciones y masa del punto óptimo al que converge el algoritmo difiere más de lo permitido (1%). El mejor individuo alcanzado por el algoritmo se simula si no se ha simulado previamente y se incluye en la nube de datos para refinar el metamodelo y trabajar con mayor precisión en las siguientes ejecuciones. Los individuos próximos a puntos evaluados en esta fase se penalizan para favorecer la exploración en las “n” primeras iteraciones. Además, en la primera iteración se penaliza la cercanía al mejor diseño evaluado antes de empezar el bucle del AG de aproximación a la frontera. El punto “n+1” y posteriores se obtienen sin penalizar por cercanía para intensificar de nuevo el muestro en las zonas de mejores diseños según el metamodelo de aproximación.
  - 5) Algoritmo genético de optimización final. Con la nube de puntos elaborada en las etapas anteriores se reconstruye el metamodelo de aproximación para evaluar la aptitud de los individuos en el algoritmo. Finalmente, éste converge hacia un óptimo que se simula si no se ha simulado previamente. En el caso de que no mejore al diseño óptimo alcanzado en las fases previas al AG de optimización final, se actualiza el metamodelo con el último punto evaluado y se ejecuta de nuevo el algoritmo en un proceso iterativo hasta conseguir un diseño mejor que el logrado en etapas anteriores de la metodología.
  - 6) Búsqueda del mejor individuo simulado (óptimo), que será justamente el último punto evaluado en el AG final.

#### 7.4.1. Resultados obtenidos para el problema con 4 variables de diseño

Se realizaron 10 ejecuciones del nuevo programa con el problema de referencia (con 4 variables de diseño). En la siguiente tabla se muestra el diseño óptimo obtenido en cada caso y el número de puntos evaluados:

Ejecución	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Nº puntos simulados	Desplazamiento (mm)	Masa (g)
1	41.878	4.239	3.636	54.146	26	14.992	1639.538
2	42.184	4.304	3.348	56.664	26	14.973	1625.382
3	43.634	4.362	3.249	55.491	26	14.993	1618.408
4	42.937	4.335	3.322	54.748	26	14.993	1623.183
5	43.510	4.368	3.023	60.000	26	14.984	1604.163
6	46.018	4.423	3.000	59.818	26	15.000	1601.636
7	38.620	4.257	3.000	60.000	26	14.995	1610.092
8	48.140	4.390	3.538	56.843	26	14.990	1632.351
9	42.802	4.315	3.282	57.918	26	14.984	1618.832
10	42.311	4.340	3.000	60.000	26	14.997	1602.379

Tabla 24. Resultados óptimos del cuarto programa en el problema de referencia (4 variables).

Como se aprecia, los resultados obtenidos varían mucho dependiendo de la ejecución. En algunos casos el método llega a una solución muy próxima al óptimo teórico en términos de masa (1600.809g), mientras que en otras ejecuciones como la número 1 el resultado empeora considerablemente. Esta variabilidad se debe al carácter aleatorio del AG de aproximación y al AG final. No obstante, el número de puntos simulados fue en todos los casos de 26, que igualaría al número de puntos requeridos por la optimización del SolidWorks para un problema de 4 variables.

Tras comprobar que el programa no llega con facilidad a un resultado próximo al óptimo teórico, se optó por emplear ecuaciones polinómicas de orden 2 ajustadas por mínimos cuadrados como metamodelo de aproximación en lugar de la interpolación lineal basada en triangulación de Delaunay.

### 7.5. Quinto programa

A pesar de que en un principio se optó por emplear interpolación lineal como metamodelo de aproximación, en esta versión del programa se recurrió a ecuaciones polinómicas de orden 2 ajustadas por mínimos cuadrados. En capítulos anteriores se observó que las estimaciones con este modelo de regresión eran más precisas que las obtenidas con interpolación lineal. Sin embargo se apostó por la interpolación lineal por el hecho de que se trata de un método interpolador y no de un modelo de ajuste, de modo que al concentrar el muestreo en una zona las estimaciones del interpolador serán más precisas que las del modelo de regresión.

Desde el punto de vista de la programación del nuevo metamodelo, fue necesario crear una función externa en Matlab que define la forma de la ecuación polinómica en función del número de variables de diseño, automatizando así el proceso. Por tanto, dependiendo del número de variables esta función auxiliar definirá la forma de la expresión polinómica incluyendo todos los términos (término independiente, todos los términos de orden 1 y todos los términos de orden 2 con todas las combinaciones posibles).

Una vez implementado el programa únicamente con el modelo de regresión como metamodelo de aproximación, se realizó una primera prueba con el problema de referencia con 4 variables de diseño y sin simulaciones por elementos finitos. Evidentemente el resultado obtenido por el programa fue muy próximo al óptimo teórico ya que las ecuaciones empleadas para sustituir los elementos finitos fueron obtenidas mediante ajuste por mínimos cuadrados con ecuaciones polinómicas de orden 2, lo cual favorece al metamodelo de aproximación y por tanto no se pueden sacar conclusiones al respecto. Por ello, para las pruebas del nuevo programa se aplicó la metodología evaluando los diferentes diseños con cálculo por elementos finitos.

### 7.5.1. Resultados obtenidos para el problema con 3 variables de diseño (evaluaciones con elementos finitos)

La primera prueba realizada con el nuevo programa se llevó a cabo sobre el problema de referencia con 3 variables de diseño. Durante el proceso de optimización se detuvo el programa ya que se observaba que éste no convergía hacia una solución, sino que cada vez se obtenían peores resultados en las sucesivas simulaciones. En la siguiente tabla se muestran los diseños evaluados hasta detener el programa en el punto 23:

Punto	L (mm)	e (mm)	e <sub>h</sub> (mm)	Desplazamiento (mm)	Masa (g)
1	20.000	3.000	3.000	16.061	1705.09
2	20.000	3.000	8.000	13.625	2273.12
3	20.000	8.000	3.000	11.338	2567.12
4	20.000	8.000	8.000	11.046	2777.22
5	60.000	3.000	3.000	19.025	1244.24
6	60.000	3.000	8.000	17.693	1499.71
7	60.000	8.000	3.000	11.684	2386.56
8	60.000	8.000	8.000	11.543	2470.68
9	40.000	5.500	5.500	12.741	2087.14

10	60.000	5.741	3.000	13.331	1921.24
11	60.000	4.709	3.000	14.664	1676.11
12	60.000	4.577	3.000	14.851	1643.69
13	48.203	4.242	3.000	15.204	1614.08
14	36.829	4.043	3.000	14.894	1649.15
15	24.882	3.587	3.000	15.408	1709.95
16	21.710	3.000	7.389	14.110	2163.08
17	49.201	4.392	3.000	14.988	1645.59
18	49.821	4.405	3.000	15.043	1644.66
19	48.684	4.383	3.000	14.892	1645.65
20	51.304	4.434	3.000	15.048	1644.90
21	50.258	4.413	3.000	15.043	1644.45
22	60.000	4.091	7.999	15.043	1735.98
23	50.856	4.430	3.000	-----	-----

Tabla 25. Puntos evaluados durante la ejecución de la quinta versión del programa (3 variables).

Tras visualizar los resultados de los diferentes puntos evaluados se observó que la inclusión de los nuevos puntos simulados en el AG final (puntos en rojo) propició una distorsión en el metamodelo de aproximación. Al añadir nuevos puntos muy cercanos entre sí y actualizar las superficies de respuesta, éstas se distorsionan para minimizar el error cuadrático, ocasionando que el método no converja hacia una solución que mejore el diseño óptimo que se tenía antes de aplicar el AG de optimización final y por tanto evitando que se cumpla la condición de parada del AG final. Esto demuestra la limitación de los modelos de regresión, que no garantizan unos resultados muy precisos sobre todo cuando se desconoce la ecuación de comportamiento del problema ya que están muy supeditados a la forma de la ecuación de ajuste. Sin embargo los puntos evaluados en la fase del AG de aproximación a la frontera (en azul) parecen ser buenos.

En base a estas conclusiones anteriores, se decidió modificar el programa para emplear interpolación lineal como metamodelo de aproximación en el AG final, y las ecuaciones de ajuste por mínimos cuadrados como metamodelo en el AG de aproximación a la frontera. Con esto se evitará el problema anterior de distorsión de las ecuaciones de ajuste al añadir nuevos puntos en el AG final.

Con esta modificación en el programa, se ejecutó de nuevo la optimización obteniendo los siguientes resultados:

Punto	L (mm)	e (mm)	e <sub>h</sub> (mm)	Desplazamiento (mm)	Masa (g)
1	20.000	3.000	3.000	16.061	1705.09
2	20.000	3.000	8.000	13.625	2273.12
3	20.000	8.000	3.000	11.338	2567.12
4	20.000	8.000	8.000	11.046	2777.22
5	60.000	3.000	3.000	19.025	1244.24
6	60.000	3.000	8.000	17.693	1499.71
7	60.000	8.000	3.000	11.684	2386.56
8	60.000	8.000	8.000	11.543	2470.68
9	40.000	5.500	5.500	12.741	2087.14
10	60.000	5.741	3.000	13.331	1921.24
11	60.000	4.709	3.000	14.664	1676.11
12	60.000	4.577	3.000	14.851	1643.69
13	47.523	4.244	3.150	15.099	1627.36
14	33.975	3.873	3.000	14.977	1639.45
15	22.533	3.490	3.000	15.406	1737.96
16	60.000	7.982	5.026	11.635	2420.42
17	46.961	4.318	3.000	15.020	1638.28
18	60.000	4.521	3.000	15.072	1630.25
19	56.280	4.455	3.070	15.169	1631.90
20	60.000	4.540	3.000	14.975	1634.69

Tabla 26. Puntos evaluados durante la ejecución de la quinta versión del programa (interpolación lineal en el AG final y ajuste por mínimos cuadrados en el AG de aproximación a la frontera) con 3 variables.

El diseño óptimo obtenido es prácticamente idéntico al alcanzado con la tercera versión del programa, tanto en términos de masa como en desplazamiento y variables de diseño. Sin embargo, en la tercera versión se necesitaron 29 simulaciones mientras que en este caso sólo 20, mejorando sustancialmente la eficiencia del método. En comparación con la optimización del SolidWorks, el programa consigue un diseño 55.38g más ligero (1690.07g del SolidWorks frente a los 1634.69g obtenidos). Sin embargo, el coste computacional del programa desarrollado es algo mayor ya que se requieren 6 simulaciones más (14 puntos en el SolidWorks frente a 20 del programa). Por otro lado también hay que resaltar que con esta metodología se consigue mejorar el diseño óptimo del SolidWorks desde la simulación número 11, por lo que el resto de fases de la metodología no serían necesarias para mejorar el resultado.

A pesar de que los resultados son bastante buenos tanto en lo referente a la optimización como al número de simulaciones requeridas, se realizaron una serie de modificaciones referentes al AG de aproximación a la frontera para tratar de mejorar su

eficiencia. En primer lugar se decidió emplear interpolación lineal basada en triangulación de Delaunay como metamodelo para determinar la ubicación de los puntos posteriores al “n+1”. Por tanto, los primeros “n+1” puntos simulados en esta fase se determinan empleando como metamodelo las ecuaciones ajustadas por mínimos cuadrados, y los puntos posteriores al “n+1” empleando interpolación lineal. Esto se hizo para añadir nuevos puntos a partir del metamodelo que se empleará en el AG final. Con ello se consigue determinar el punto “n+1” de esta fase mediante las ecuaciones de ajuste, y los puntos restantes mediante interpolación lineal.

Por otro lado, a pesar de que el punto añadido “n+1” en esta fase se determina mediante las ecuaciones de ajuste (y sin penalización por cercanía a otros puntos), se decidió calcular el error de estimación de este punto suponiendo que el metamodelo de aproximación empleado fue la interpolación lineal. Como el metamodelo empleado en los siguientes puntos y en el AG final es el basado en interpolación lineal, el error se debe determinar con respecto a este metamodelo. Además con esto se controla en cierta medida las posibles distorsiones que puedan surgir en las ecuaciones de ajuste al añadir los “n+1” puntos simulados en esta fase.

Tras realizar estos cambios en el código, los resultados obtenidos fueron:

Punto	L (mm)	e (mm)	e <sub>h</sub> (mm)	Desplazamiento (mm)	Masa (g)
1	20.000	3.000	3.000	16.061	1705.09
2	20.000	3.000	8.000	13.625	2273.12
3	20.000	8.000	3.000	11.338	2567.12
4	20.000	8.000	8.000	11.046	2777.22
5	60.000	3.000	3.000	19.025	1244.24
6	60.000	3.000	8.000	17.693	1499.71
7	60.000	8.000	3.000	11.684	2386.56
8	60.000	8.000	8.000	11.543	2470.68
9	40.000	5.500	5.500	12.741	2087.14
10	60.000	5.741	3.000	13.331	1921.24
11	60.000	4.709	3.000	14.664	1676.11
12	60.000	4.577	3.000	14.851	1643.69
13	47.523	4.244	3.150	15.107	1627.36
14	36.322	3.954	3.000	14.953	1633.03
15	24.541	3.557	3.000	15.450	1709.60
16	49.707	4.353	3.000	15.073	1632.81
17	37.280	3.967	3.000	15.021	1627.63
18	36.952	3.962	3.000	15.034	1629.29

19	60.000	4.521	3.000	15.010	1630.26
20	36.685	3.959	3.000	15.031	1630.97
21	37.582	3.994	3.017	15.017	1631.06
22	37.456	3.993	3.000	14.975	1632.44

Tabla 27. Puntos evaluados durante la ejecución de la quinta versión del programa con las últimas modificaciones (3 variables).

El diseño óptimo obtenido es muy similar al conseguido anteriormente en términos de masa (1632.44g frente a 1634.69), pero la cadena de cromosomas es bastante diferente, por lo que existen múltiples combinaciones con resultados similares. En este caso se emplearon 22 simulaciones, 2 más que en la prueba anterior. Esta diferencia se debe al carácter estocástico del AG final, que necesitó 6 iteraciones en este caso frente a las 3 de la ejecución anterior. Sin embargo se observa que se requiere un punto menos en el AG de aproximación a la frontera por el hecho de evaluar el error empleando las estimaciones de la interpolación lineal.

#### 7.5.2. Resultados obtenidos para el problema con 4 variables de diseño (evaluaciones con elementos finitos)

Con la última versión del programa se ejecutó la optimización para el problema de referencia con 4 variables de diseño. Los puntos evaluados durante la optimización fueron los siguientes:

Punto	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
1	20.000	3.000	3.000	20.000	16.061	1705.09
2	20.000	3.000	3.000	60.000	16.454	1578.69
3	20.000	3.000	8.000	20.000	13.621	2273.12
4	20.000	3.000	8.000	60.000	14.058	2092.02
5	20.000	8.000	3.000	20.000	11.338	2567.12
6	20.000	8.000	3.000	60.000	11.418	2521.56
7	20.000	8.000	8.000	20.000	11.046	2777.22
8	20.000	8.000	8.000	60.000	11.111	2713.70
9	60.000	3.000	3.000	20.000	18.324	1399.41
10	60.000	3.000	3.000	60.000	19.029	1244.24
11	60.000	3.000	8.000	20.000	16.555	1793.55
12	60.000	3.000	8.000	60.000	17.690	1499.71
13	60.000	8.000	3.000	20.000	11.554	2443.66

14	60.000	8.000	3.000	60.000	11.684	2386.56
15	60.000	8.000	8.000	20.000	11.332	2580.16
16	60.000	8.000	8.000	60.000	11.547	2470.68
17	40.000	5.500	5.500	40.000	12.745	2087.14
18	60.000	5.743	3.000	60.000	13.340	1921.53
19	60.000	4.714	3.000	60.000	14.644	1677.41
20	60.000	4.575	3.000	60.000	14.900	1643.16
21	60.000	4.577	3.000	42.208	14.878	1670.18
22	42.727	4.163	3.000	53.459	15.113	1609.09
23	37.769	4.019	3.000	36.959	14.953	1637.33
24	54.840	4.312	3.000	26.065	15.117	1676.81
25	46.144	4.298	3.000	52.833	15.054	1627.43
26	53.524	4.413	3.000	57.574	15.158	1623.77
27	60.000	4.537	3.000	60.000	15.044	1633.88
28	60.000	4.549	3.000	60.000	14.994	1636.86

Tabla 28. Puntos evaluados durante la ejecución de la quinta versión del programa (4 variables).

El resultado óptimo logrado por este programa difiere notablemente frente al resultado de la segunda versión del programa. En este caso la masa del diseño óptimo es de 1636.86g, aproximadamente 15g superior a la del diseño óptimo de la segunda versión del programa (1621.47g). Sin embargo el número de simulaciones disminuye de 79 a 28, por lo que la nueva versión es más eficiente. Comparando el nuevo programa con la optimización del SolidWorks, se aprecia que el código desarrollado consigue una masa inferior (1638.86g frente a 1678.64g) con tan solo 2 evaluaciones más (28 simulaciones frente a 26 del SolidWorks). Además, desde el punto número 19 ya se mejora el resultado de la optimización del SolidWorks.

A pesar de que los resultados son buenos, el método en este caso no converge hacia una solución del todo óptima ya que no se mejora la masa del diseño óptimo obtenido para el problema de 3 variables de diseño. En el peor de los casos, el óptimo teórico tendría una masa idéntica al caso de 3 variables de diseño. Sin embargo, al ampliar el dominio de búsqueda con la nueva variable, el método no converge exactamente hacia la solución óptima pero sí que alcanza un buen diseño, mejorando la optimización del SolidWorks.

### 7.5.3. Resultados obtenidos para el problema con 5 variables de diseño (evaluaciones con elementos finitos)

El programa implementado también se ejecutó para el problema de referencia con 5 variables de diseño. Los puntos simulados por elementos finitos durante el proceso de optimización fueron los siguientes:

Punto	L1 (mm)	L2 (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
1	20.000	20.000	3.000	3.000	20.000	16.053	1705.09
2	20.000	20.000	3.000	3.000	60.000	16.454	1578.69
3	20.000	20.000	3.000	8.000	20.000	13.629	2273.12
4	20.000	20.000	3.000	8.000	60.000	14.058	2092.02
5	20.000	20.000	8.000	3.000	20.000	11.338	2567.12
6	20.000	20.000	8.000	3.000	60.000	11.418	2521.56
7	20.000	20.000	8.000	8.000	20.000	11.046	2777.22
8	20.000	20.000	8.000	8.000	60.000	11.111	2713.70
9	20.000	60.000	3.000	3.000	20.000	16.970	1568.35
10	20.000	60.000	3.000	3.000	60.000	17.336	1429.26
11	20.000	60.000	3.000	8.000	20.000	14.665	2072.28
12	20.000	60.000	3.000	8.000	60.000	15.373	1844.00
13	20.000	60.000	8.000	3.000	20.000	11.382	2513.01
14	20.000	60.000	8.000	3.000	60.000	11.468	2462.31
15	20.000	60.000	8.000	8.000	20.000	11.207	2693.54
16	20.000	60.000	8.000	8.000	60.000	11.334	2610.61
17	60.000	20.000	3.000	3.000	20.000	16.897	1552.41
18	60.000	20.000	3.000	3.000	60.000	17.383	1411.64
19	60.000	20.000	3.000	8.000	20.000	14.538	2052.83
20	60.000	20.000	3.000	8.000	60.000	15.258	1819.79
21	60.000	20.000	8.000	3.000	20.000	11.476	2503.79
22	60.000	20.000	8.000	3.000	60.000	11.542	2452.20
23	60.000	20.000	8.000	8.000	20.000	11.194	2685.23
24	60.000	20.000	8.000	8.000	60.000	11.321	2600.10
25	60.000	60.000	3.000	3.000	20.000	18.323	1399.41
26	60.000	60.000	3.000	3.000	60.000	19.030	1244.24
27	60.000	60.000	3.000	8.000	20.000	16.555	1793.55
28	60.000	60.000	3.000	8.000	60.000	17.690	1499.71
29	60.000	60.000	8.000	3.000	20.000	11.554	2443.66
30	60.000	60.000	8.000	3.000	60.000	11.684	2386.56
31	60.000	60.000	8.000	8.000	20.000	11.330	2580.16
32	60.000	60.000	8.000	8.000	60.000	11.547	2470.68

33	40.000	40.000	5.500	5.500	40.000	12.742	2087.14
34	60.000	60.000	5.743	3.000	60.000	13.318	1921.62
35	60.000	60.000	4.704	3.000	60.000	14.655	1674.94
36	60.000	60.000	4.570	3.000	60.000	14.908	1641.90
37	60.000	37.683	4.304	3.000	60.000	14.751	1623.35
38	60.000	48.054	4.351	3.000	40.466	14.936	1642.30
39	20.000	47.853	3.796	3.000	60.000	15.372	1634.07
40	20.000	50.169	3.820	3.000	37.202	15.222	1675.71
41	33.693	28.982	3.875	3.000	60.000	15.218	1610.23
42	45.018	40.305	4.269	3.000	60.000	14.844	1628.45
43	59.723	38.985	4.228	3.000	60.000	14.934	1600.64

Tabla 29. Puntos evaluados durante la ejecución de la quinta versión del programa (5 variables).

El diseño óptimo alcanzado tiene una masa de 1600.64g, 36.22g menos que en el caso de 4 variables de diseño. En este caso la nueva variable incluida en la optimización produce mejoras significativas en el diseño óptimo. Dado que no se tiene ninguna referencia comparable para este problema con 5 variables (y evaluaciones con MEF), se realizó la optimización en el SolidWorks, obteniendo el siguiente resultado tras 42 simulaciones:

**DISEÑO ÓPTIMO (SolidWorks): 58.316 30.060 4.291 3.144 59.633**  
**Desplazamiento: 14.574mm**  
**Masa: 1659.70g**

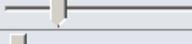
		Actual	Inicial	Óptimo
lado1		58.313599	60.000000	58.313599
lado2		30.060425	60.000000	30.060425
esp		4.291122	8.000000	4.291122
espesorentrehuecos		3.143509	8.000000	3.143509
h		59.633179	60.000000	59.633179
Desplazamiento1	< 15mm	14.57386mm	11.54653mm	14.57386mm
Masa1	Minimizar	1659.7 g	2470.68 g	1659.7 g

Ilustración 39. Resultado de la optimización del SolidWorks (problema de referencia, 5 variables).

Como se aprecia el resultado obtenido mediante la optimización del SolidWorks (Box-Behnken y superficies de respuesta) tiene una masa 59.06g peor que la obtenida con el programa desarrollado. Además en este caso el programa realiza sólo 43 simulaciones, una más que en la optimización del SolidWorks.

#### 7.5.4. Resultados obtenidos para otro problema de ejemplo (5 variables, evaluaciones con elementos finitos)

Dado que todas las pruebas realizadas hasta el momento se han aplicado sobre la misma geometría modificando únicamente el número de variables de diseño, se decidió probar la metodología implementada con otro problema de ejemplo.

La geometría en este caso se corresponde a la potencia o codo de una bicicleta, que debe soportar una fuerza vertical de 1000N sin que las tensiones de Von Mises sobrepasen el límite elástico del material en ningún punto (14MPa al tratarse de ABS) y con un desplazamiento máximo de 6mm. Se pretende minimizar el peso de la pieza cumpliendo estos requisitos mediante la optimización de 5 variables de diseño, correspondientes a las 3 dimensiones de las celdas huecas ( $L_x$ ,  $L_y$ ,  $L_z$ ), el espesor de la "piel exterior" ( $e$ ) y el espesor entre las celdas huecas ( $e_h$ ). Para dificultar el proceso de optimización y poner a prueba la metodología, se definió un dominio bastante amplio, establecido por los siguientes intervalos de las variables:

- $L_x \rightarrow 5\sim 50\text{mm}$
- $L_y \rightarrow 10\sim 40\text{mm}$
- $L_z \rightarrow 20\sim 150\text{mm}$
- $e \rightarrow 1\sim 20\text{mm}$
- $e_h \rightarrow 1\sim 20\text{mm}$

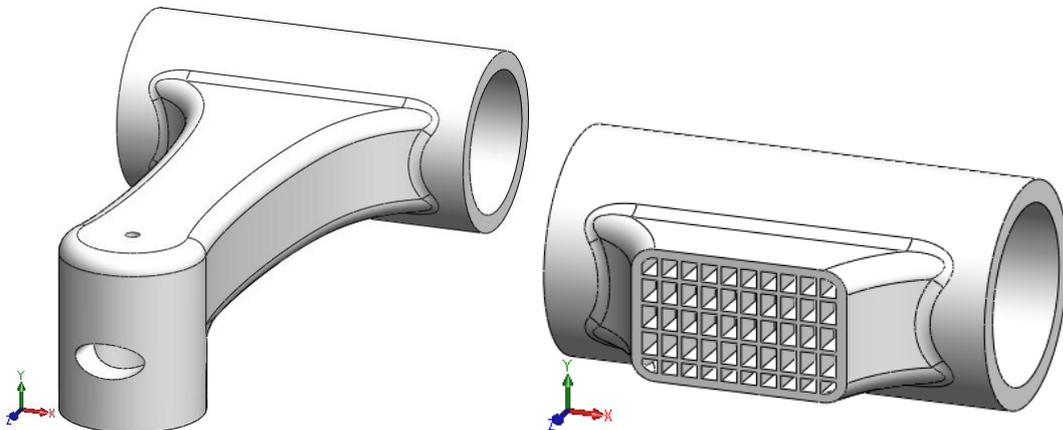


Ilustración 40. Geometría de la pieza a optimizar y sección donde se aprecia la estructura celular.

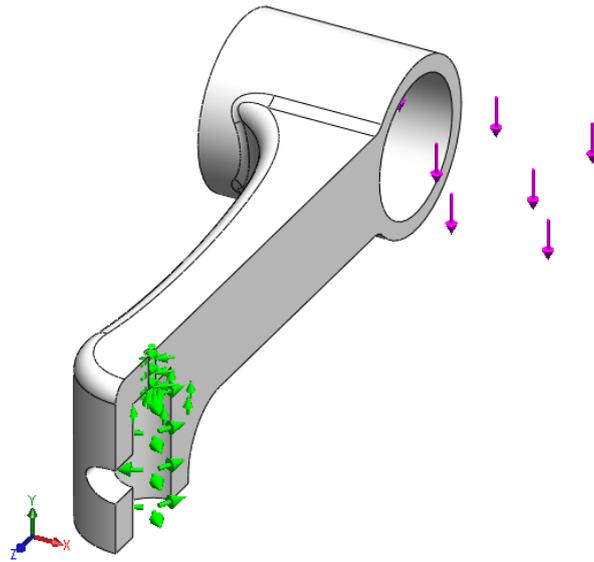


Ilustración 41. Sección de la pieza y condiciones de contorno.

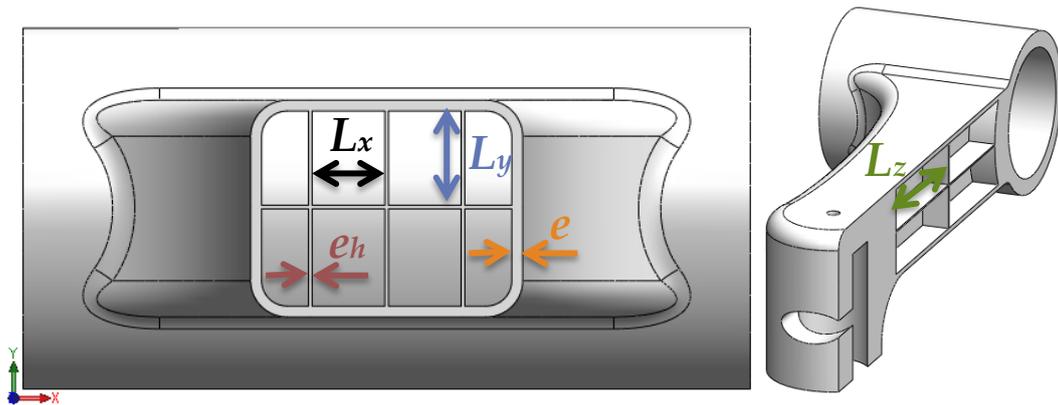


Ilustración 42. Variables de diseño.

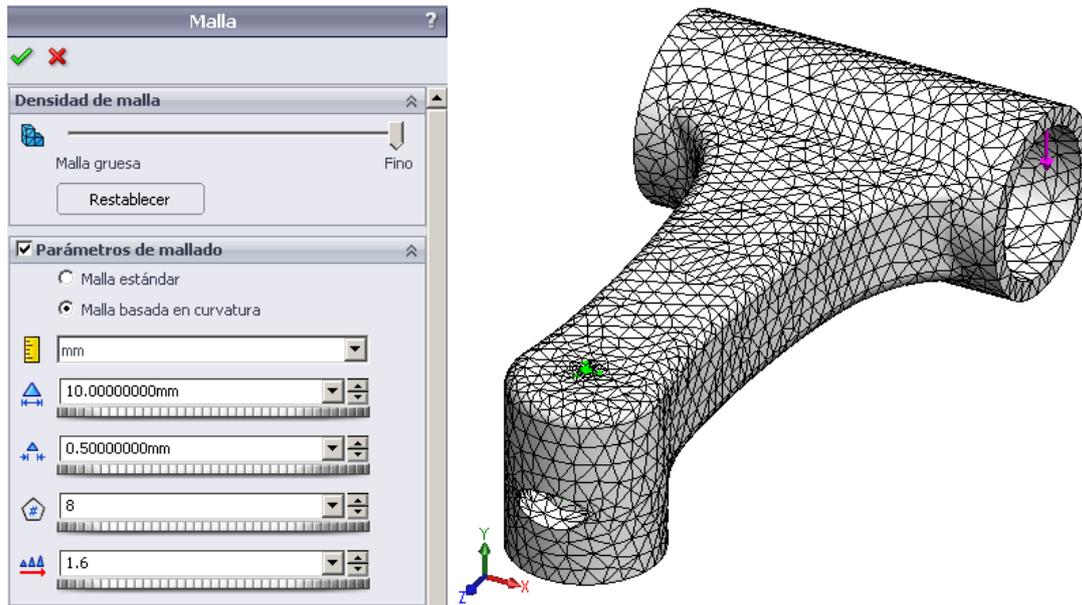


Ilustración 43. Mallado basado en curvatura.

Con el análisis estático totalmente definido (parametrización de la geometría, condiciones de contorno, malla, material y sensores), así como el intervalo de las variables de diseño, las dos restricciones del problema (tensión máxima de Von Mises y desplazamiento máximo) y el objetivo (minimizar la masa), se ejecutó la optimización del SolidWorks obteniendo el siguiente resultado tras 42 simulaciones:

**DISEÑO ÓPTIMO (SolidWorks): 50.000 25.000 85.000 10.500 1.000**  
**Tensión de Von Mises: 10.525N/mm<sup>2</sup>**  
**Desplazamiento: 5.532mm**  
**Masa: 1678.92g**

		Actual	Inicial	Óptimo
ladox		50.000000	27.500000	50.000000
ladoy		25.000000	10.000000	25.000000
ladoz		85.000000	85.000000	85.000000
esp.ext.		10.500000	10.500000	10.500000
esp.ent.huecos		1.000000	10.500000	1.000000
Tensión1	≤ 14 N/mm <sup>2</sup>	10.525 N/mm <sup>2</sup>	10.297 N/mm <sup>2</sup>	10.525 N/mm <sup>2</sup>
Desplazamiento1	≤ 6mm	5.5323mm	4.95434mm	5.5323mm
Masa1	Minimizar	1678.92 g	1878.98 g	1678.92 g

Ilustración 44. Resultado óptimo en la optimización del SolidWorks.

Posteriormente se aplicó la metodología desarrollada en este trabajo. Los puntos simulados durante el proceso de optimización se muestran en la siguiente tabla:

Punto	L <sub>x</sub> (mm)	L <sub>y</sub> (mm)	L <sub>z</sub> (mm)	e (mm)	e <sub>h</sub> (mm)	Tensión Von Mises (MPa)	Desplazamiento (mm)	Masa (g)
1	5.000	10.000	20.000	1.000	1.000	21.88	10.21	1432.97
2	5.000	10.000	20.000	1.000	20.000	10.37	4.80	1972.53
3	5.000	10.000	20.000	20.000	1.000	10.21	4.76	1929.23
4	5.000	10.000	20.000	20.000	20.000	10.30	4.70	1990.82
5	5.000	10.000	150.000	1.000	1.000	22.76	10.54	1410.10
6	5.000	10.000	150.000	1.000	20.000	10.72	4.87	1958.94
7	5.000	10.000	150.000	20.000	1.000	10.37	4.76	1926.77
8	5.000	10.000	150.000	20.000	20.000	9.76	4.71	1985.23
9	5.000	40.000	20.000	1.000	1.000	24.87	11.72	1390.02
10	5.000	40.000	20.000	1.000	20.000	10.46	4.87	1946.39
11	5.000	40.000	20.000	20.000	1.000	10.38	4.76	1922.11
12	5.000	40.000	20.000	20.000	20.000	9.87	4.71	1985.08
13	5.000	40.000	150.000	1.000	1.000	26.29	12.14	1365.43
14	5.000	40.000	150.000	1.000	20.000	10.47	4.96	1918.48
15	5.000	40.000	150.000	20.000	1.000	9.34	4.76	1919.39
16	5.000	40.000	150.000	20.000	20.000	9.93	4.71	1973.96
17	50.000	10.000	20.000	1.000	1.000	36.35	16.13	1330.70
18	50.000	10.000	20.000	1.000	20.000	16.03	5.34	1896.39
19	50.000	10.000	20.000	20.000	1.000	10.29	4.78	1917.83
20	50.000	10.000	20.000	20.000	20.000	10.02	4.71	1987.87
21	50.000	10.000	150.000	1.000	1.000	38.10	16.81	1303.63
22	50.000	10.000	150.000	1.000	20.000	12.71	5.64	1839.84
23	50.000	10.000	150.000	20.000	1.000	10.29	4.78	1915.00
24	50.000	10.000	150.000	20.000	20.000	9.97	4.71	1979.69
25	50.000	40.000	20.000	1.000	1.000	47.96	21.68	1279.95
26	50.000	40.000	20.000	1.000	20.000	19.44	6.36	1786.57
27	50.000	40.000	20.000	20.000	1.000	10.35	4.78	1909.51
28	50.000	40.000	20.000	20.000	20.000	9.89	4.71	1979.22
29	50.000	40.000	150.000	1.000	1.000	52.74	22.59	1250.82
30	50.000	40.000	150.000	1.000	20.000	17.23	6.88	1670.80
31	50.000	40.000	150.000	20.000	1.000	10.00	4.78	1906.38
32	50.000	40.000	150.000	20.000	20.000	9.80	4.72	1962.94
33	27.500	25.000	85.000	10.500	10.500	10.36	5.22	1815.86
34	50.000	40.000	150.000	18.701	1.000	10.37	4.81	1888.61
35	50.000	40.000	150.000	14.687	1.000	10.27	5.01	1802.32
36	50.000	40.000	150.000	13.913	1.000	10.22	5.07	1782.19
37	50.000	40.000	150.000	13.226	1.000	10.37	5.13	1763.32

38	50.000	40.000	150.000	12.619	1.000	10.43	5.20	1745.81
39	50.000	40.000	150.000	12.084	1.000	9.30	5.26	1729.73
40	50.000	40.000	150.000	11.613	1.000	9.39	5.33	1715.07
41	50.000	40.000	150.000	11.201	1.000	9.38	5.39	1701.84
42	50.000	40.000	150.000	10.841	1.000	10.45	5.45	1689.98
43	50.000	40.000	150.000	10.527	1.000	10.45	5.51	1679.36
44	50.000	40.000	150.000	10.253	1.000	10.49	5.56	1669.94
45	50.000	40.000	150.000	10.015	1.000	-----	-----	-----

Tabla 30. Puntos evaluados durante la ejecución del programa (se detuvo el proceso).

El proceso se detuvo porque se detectó que el método no convergía hacia la frontera en la fase de aproximación por las aristas. Esto se debe a que la interpolación lineal empleada estima resultados muy alejados y requiere de muchas iteraciones para acercarse a la frontera por tratarse de una respuesta poco lineal. Se añadieron 10 puntos en esta fase (puntos en naranja en la tabla anterior, desde el 34 hasta el 44) y no se consiguió aproximarse a la frontera con un error inferior al 1% en la restricción crítica. No obstante hay que decir que el último punto evaluado mejora el diseño obtenido con el SolidWorks tan solo con dos simulaciones más.

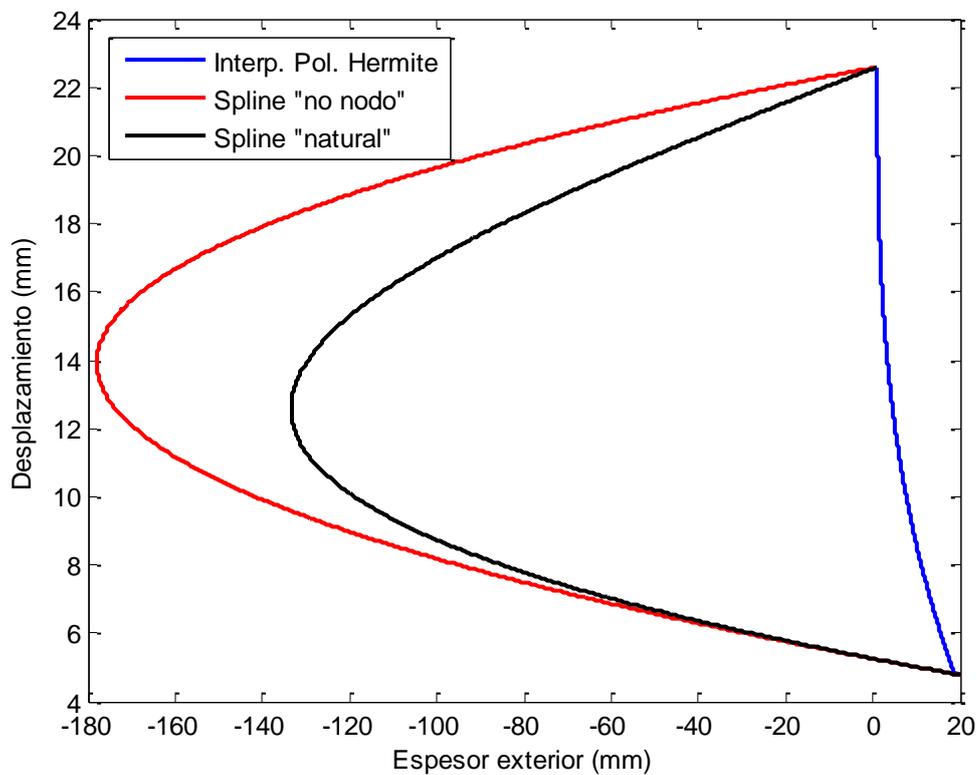
Otra conclusión importante de esta prueba es que a medida que se disminuye el espesor exterior ( $e$ ), los desplazamientos aumentan, como era de esperar. Sin embargo, la tensión máxima de Von Mises tiene un comportamiento aleatorio. Así por ejemplo entre el punto 38 y el punto 39, el espesor exterior disminuye de 12.619mm a 12.084mm, manteniendo el resto de variables constantes. Lógicamente, al disminuir el espesor exterior se pierde momento de inercia y por tanto la tensión máxima de Von Mises debería aumentar. Sin embargo se aprecia que inexplicablemente la tensión máxima de Von Mises del punto 39 es inferior a la del punto 38. Esto se debe a que el modelo elástico empleado es un modelo lineal (módulo de elasticidad constante), de modo que las concentraciones de tensiones en algunas zonas se incrementarán casi hasta el infinito si se refina la malla, por lo que los resultados de tensiones en estas zonas están falseados por una limitación del modelo de cálculo y por tanto los resultados dependerán fuertemente de la densidad de mallado. A modo de ejemplo, si se tiene una viga empotrada con una carga a flexión, las tensiones en la fibra de tracción o compresión de la sección empotrada se incrementarán exponencialmente a medida que se refine el mallado en esa zona. Por ello es importante que el análisis realizado por elementos finitos tenga un comportamiento lo más real posible. Esto se resolvería en este caso aplicando un modelo elástico no lineal.

A pesar de que esta sería la solución correcta para un caso real, se decidió prescindir de las tensiones en las siguientes pruebas de la metodología simplemente para reducir el tiempo de cálculo y centrar el trabajo en los aspectos relevantes de la metodología, como la convergencia tan lenta en la aproximación a la frontera.

Para tratar de resolver este problema, en primer lugar se decidió aumentar el margen de error en la aproximación a la frontera, que era de un 1%. En las primeras versiones del programa este error debía ser pequeño para elegir correctamente la mejor esquina de la frontera. Sin embargo, esto ya no es tan importante ya que sólo se realiza la aproximación a la frontera por las aristas comunes al punto de menor masa de los pertenecientes a aristas “cortadas” por la frontera. Por esta razón se decidió emplear un error de un 5% en lugar del 1%, logrando por tanto que en algunos problemas se necesiten menos simulaciones. Además esto ayudará también a que el AG final cumpla con mayor facilidad el criterio de parada sobre todo en los casos en los que el diseño óptimo real se encuentre sobre una arista del dominio. En esos casos se obtendrán diseños muy buenos en la fase de aproximación a la frontera, de modo que si el error permitido en esta etapa es muy bajo, se obtendrán diseños muy buenos que serán difíciles de mejorar, por lo que el AG final repetirá sucesivamente hasta conseguirlo. Al permitir un mayor error en la aproximación, es probable que el mejor individuo de esta fase no sea tan bueno como antes, por lo que el AG final conseguirá mejorarlo más fácilmente en la mayoría de casos.

A pesar de que esta modificación puede ayudar a reducir el número de simulaciones, el principal problema del programa no se ha resuelto, que es justamente la estimación del punto para aproximarse a la frontera. Como se ha comprobado, la interpolación lineal (o cuadrática al añadir el segundo punto) en problemas complejos como el anterior no converge con facilidad. Por ello se decidió sustituir la interpolación lineal (y cuadrática cuando se tienen 2 puntos) por un modelo interpolador más complejo que mejore la convergencia, como la spline cúbica o la interpolación con polinomios cúbicos de Hermite (a trozos). Ambos métodos están definidos por una ecuación cúbica en cada trozo comprendido entre dos puntos dato. La principal diferencia es que la interpolación spline garantiza continuidad en la primera y segunda derivada en los puntos que son dato. Sin embargo, la interpolación cúbica de Hermite no garantiza continuidad en la primera y segunda derivada y mantiene unos valores fijos de la primera derivada en los puntos dato, calculando estas pendientes a través de las posiciones de los datos. Cuando el comportamiento de los resultados a interpolar es suave, la spline cúbica da mejores resultados, pero en el caso de cambios bruscos la spline puede producir oscilaciones, por lo que la interpolación cúbica de Hermite sería la mejor opción (31).

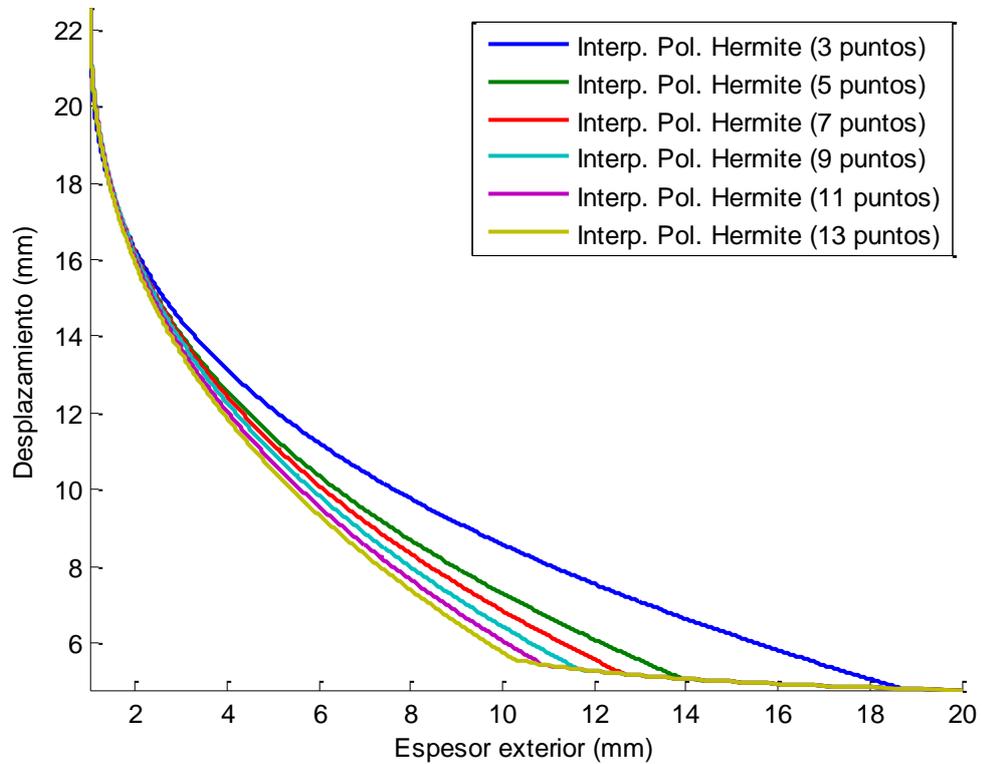
Para comparar estos dos interpoladores se creó una subrutina en Matlab que representa gráficamente el resultado de la interpolación spline (cúbica) y la interpolación con polinomios cúbicos de Hermite a lo largo de la arista de aproximación, empleando para ello los resultados de los puntos 29, 31 y 34 (desplazamiento frente a espesor exterior). Dado que la spline requiere de 2 condiciones adicionales para determinar el valor de todos los coeficientes de las ecuaciones cúbicas de cada tramo, se pueden aplicar diferentes condiciones de frontera para tener el mismo número de incógnitas que de ecuaciones. Entre las diferentes condiciones de frontera, se probaron las 2 posibles para este caso: condición "no nodo", que consiste en exigir que el segundo y penúltimo nodo tengan tercera derivada no nula, y la condición "natural" (o "frontera libre"), que consiste en asumir que la segunda derivada en los extremos es nula.



Gráfica 12. Comparativa entre interpolación cúbica de Hermite y splines cúbicos "no nodo" y "natural".

Se aprecia que las dos versiones de spline producen oscilaciones bruscas en la interpolación, por lo que se optó por emplear la interpolación mediante polinomios cúbicos de Hermite. En la siguiente gráfica se observa cómo va ajustándose la interpolación cúbica de Hermite a medida que se añaden nuevos datos. En este caso se

fueron añadiendo puntos de 2 en 2, desde los 3 primeros correspondientes a las simulaciones anteriores (puntos 29, 31 y 34) hasta los 13 puntos evaluados (puntos 29, 31, 34~44).



Gráfica 13. Evolución de la interpolación de Hermite a medida que se añaden nuevos datos.

Inicialmente con 3 puntos dato la estimación del interpolador estaba bastante alejada del comportamiento que se aprecia con 13 puntos. A medida que se añaden nuevos puntos el comportamiento de las estimaciones de desplazamiento frente al espesor exterior es más próximo a la realidad, que parece asemejarse a una hipérbola.

## 7.6. Sexto programa

A partir de las sugerencias anteriores (aumento del error permitido en la aproximación a la frontera y en el AG de aproximación a un 5%, e interpolación cúbica de Hermite para la ubicación de nuevos puntos en la aproximación a la frontera por las aristas) se modificó el programa, cuyo pseudocódigo sería el siguiente:

- 1) Muestreo inicial  $2^{n+1}$  (esquinas y punto central).
- 2) Aproximación a la frontera con la adición de nuevos puntos. Se identifican las aristas “cortadas” por la frontera entre espacio factible y no factible y de las esquinas del dominio involucradas en dichas arista se selecciona la de menor masa. Posteriormente se añaden nuevos puntos en las aristas que parten de esta esquina y que están “cortadas” por la frontera. Las coordenadas del punto añadido se determinan mediante interpolación cúbica de Hermite a partir de los valores de las restricciones (en el caso de varias restricciones se escoge el punto más próximo a la esquina factible). Este proceso se realiza arista por arista hasta tener una desviación frente al valor límite de la restricción crítica menor del 5%.
- 3) Adición de puntos intermedios entre la mejor esquina de la frontera y las esquinas contiguas evaluadas. (Esta parte de la metodología no aportará nuevos puntos en los casos en los que sólo se haya realizado la aproximación a la frontera por una única arista).
- 4) Algoritmo genético de aproximación a la frontera. Se repite reiteradamente siempre que se haya ejecutado menos de “n+1” veces o si el error medio de la estimación (mediante interpolación lineal) de los resultados de las restricciones y masa del punto óptimo al que converge el algoritmo difiere más de lo permitido (5%). El mejor individuo alcanzado por el algoritmo se simula si no se ha simulado previamente y se incluye en la nube de datos para refinar el metamodelo y trabajar con mayor precisión en las siguientes ejecuciones. Los individuos próximos a puntos evaluados en esta fase se penalizan para favorecer la exploración en las “n” primeras iteraciones. Además, en la primera iteración se penaliza la cercanía al mejor diseño evaluado antes de empezar el bucle del AG de aproximación a la frontera. El punto “n+1” y posteriores se obtienen sin penalizar por cercanía para intensificar de nuevo el muestro en las zonas de mejores diseños según el metamodelo de aproximación. Los primeros “n+1” puntos se obtienen empleando como metamodelo las ecuaciones de ajuste por mínimos cuadrados. Los puntos posteriores se determinan con interpolación lineal (triangulación de Delaunay). El error de estimación de las respuestas del sistema se calcula siempre mediante interpolación lineal.
- 5) Algoritmo genético de optimización final. Con la nube de puntos elaborada en las etapas anteriores se reconstruye el metamodelo de aproximación (interpolación lineal) para evaluar la aptitud de los individuos en el algoritmo. Finalmente, éste converge hacia un óptimo que se simula si no se ha simulado previamente. En el caso de que no mejore al diseño óptimo alcanzado en las fases previas al AG de optimización final, se actualiza el metamodelo con el último punto evaluado y se ejecuta de nuevo el algoritmo en un proceso iterativo hasta conseguir un diseño mejor que el logrado en etapas anteriores de la metodología.
- 6) Búsqueda del mejor individuo simulado (óptimo), que será justamente el último punto evaluado en el AG final.

### 7.6.1. Resultados obtenidos para el otro problema de ejemplo (5 variables, evaluaciones con elementos finitos)

El nuevo programa se probó en primer lugar con el problema más complejo de los planteados (codo de la bicicleta). Dado que los resultados obtenidos para la tensión máxima de Von Mises no son fiables (por las limitaciones comentadas anteriormente), se excluyó esta restricción del problema. Para tener de nuevo un diseño óptimo de referencia, se aplicó la optimización del SolidWorks a este problema, alcanzando exactamente el mismo diseño que en el caso anterior, lo cual indica que la restricción de tensión máxima de Von Mises es menos exigente que el desplazamiento máximo permitido (6mm).

Una vez conocido el diseño óptimo del SolidWorks, se ejecutó el programa desarrollado.

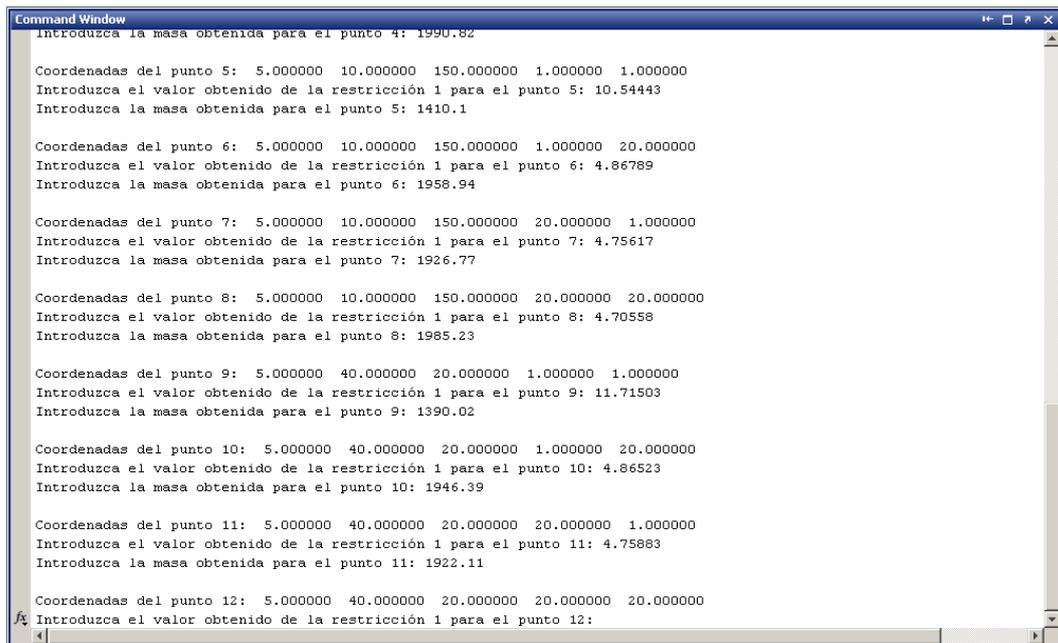


Ilustración 45. Ventana de comandos del Matlab durante la interacción del programa con el usuario.

Los puntos simulados fueron los siguientes:

Punto	L <sub>x</sub> (mm)	L <sub>y</sub> (mm)	L <sub>z</sub> (mm)	e (mm)	e <sub>h</sub> (mm)	Desplazamiento (mm)	Masa (g)
1	5.000	10.000	20.000	1.000	1.000	10.207	1432.97
2	5.000	10.000	20.000	1.000	20.000	4.802	1972.53
3	5.000	10.000	20.000	20.000	1.000	4.759	1929.23

4	5.000	10.000	20.000	20.000	20.000	4.703	1990.82
5	5.000	10.000	150.000	1.000	1.000	10.544	1410.10
6	5.000	10.000	150.000	1.000	20.000	4.868	1958.94
7	5.000	10.000	150.000	20.000	1.000	4.756	1926.77
8	5.000	10.000	150.000	20.000	20.000	4.706	1985.23
9	5.000	40.000	20.000	1.000	1.000	11.715	1390.02
10	5.000	40.000	20.000	1.000	20.000	4.865	1946.39
11	5.000	40.000	20.000	20.000	1.000	4.759	1922.11
12	5.000	40.000	20.000	20.000	20.000	4.707	1985.08
13	5.000	40.000	150.000	1.000	1.000	12.136	1365.43
14	5.000	40.000	150.000	1.000	20.000	4.961	1918.48
15	5.000	40.000	150.000	20.000	1.000	4.760	1919.39
16	5.000	40.000	150.000	20.000	20.000	4.714	1973.96
17	50.000	10.000	20.000	1.000	1.000	16.125	1330.70
18	50.000	10.000	20.000	1.000	20.000	5.343	1896.39
19	50.000	10.000	20.000	20.000	1.000	4.775	1917.83
20	50.000	10.000	20.000	20.000	20.000	4.705	1987.87
21	50.000	10.000	150.000	1.000	1.000	16.811	1303.63
22	50.000	10.000	150.000	1.000	20.000	5.643	1839.84
23	50.000	10.000	150.000	20.000	1.000	4.779	1915.00
24	50.000	10.000	150.000	20.000	20.000	4.710	1979.69
25	50.000	40.000	20.000	1.000	1.000	21.684	1279.95
26	50.000	40.000	20.000	1.000	20.000	6.362	1786.57
27	50.000	40.000	20.000	20.000	1.000	4.782	1909.51
28	50.000	40.000	20.000	20.000	20.000	4.711	1979.22
29	50.000	40.000	150.000	1.000	1.000	22.594	1250.82
30	50.000	40.000	150.000	1.000	20.000	6.875	1670.80
31	50.000	40.000	150.000	20.000	1.000	4.783	1906.38
32	50.000	40.000	150.000	20.000	20.000	4.719	1962.94
33	27.500	25.000	85.000	10.500	10.500	5.218	1815.86
34	50.000	40.000	150.000	18.701	1.000	4.810	1888.61
35	50.000	40.000	150.000	15.512	1.000	4.950	1822.49
36	50.000	40.000	150.000	13.241	1.000	5.132	1763.73
37	50.000	40.000	150.000	11.683	1.000	5.320	1717.28
38	50.000	40.000	150.000	10.630	1.000	5.491	1682.86
39	50.000	40.000	150.000	9.925	1.000	5.638	1658.43
40	50.000	40.000	150.000	9.467	1.000	5.746	1641.95
41	5.251	40.000	20.000	4.965	1.000	6.790	1553.58
42	50.000	10.281	20.000	7.059	1.000	6.325	1596.11
43	5.000	25.629	138.612	6.643	1.101	5.993	1618.30
44	23.871	22.909	57.022	11.144	11.318	5.100	1857.78
45	47.346	10.000	109.333	9.009	1.000	5.750	1651.78

46	5.000	10.000	120.117	1.000	1.000	10.533	1409.58
47	32.324	40.000	99.355	8.474	1.000	6.006	1607.21
48	33.067	38.008	99.161	8.548	1.000	5.993	1609.65

Tabla 31. Puntos evaluados durante la ejecución de la sexta versión del programa (codo bicicleta).

El diseño óptimo logrado tiene una masa de 1609.65g. Con la optimización del SolidWorks la masa del diseño óptimo para este problema es de 1678.92g, casi 70g peor. En términos porcentuales la metodología desarrollada mejora la masa un 4.3% frente a la optimización del SolidWorks. El único aspecto negativo es que el programa implementado empleó un total de 48 simulaciones, mientras que el SolidWorks requiere 42 simulaciones (6 menos). Sin embargo hay que decir que el dominio de búsqueda de este problema se definió de manera exagerada para dificultar la optimización y poner a prueba la metodología. También cabe destacar que tan solo con la fase de aproximación a la frontera por las aristas (puntos en naranja) se consigue mejorar la optimización del SolidWorks. Así por ejemplo el punto 39 ya es mejor que el diseño óptimo del SolidWorks (42 simulaciones).

Por otro lado también hay que resaltar que la interpolación cúbica de Hermite mejora significativamente la velocidad de convergencia en la aproximación a la frontera. Con sólo 7 puntos (del 34 al 40, en naranja) consigue aproximarse hasta 5.746mm de desplazamiento, mientras que en la prueba anterior se llegó a 5.56mm con 11 puntos (del 34 al 44).

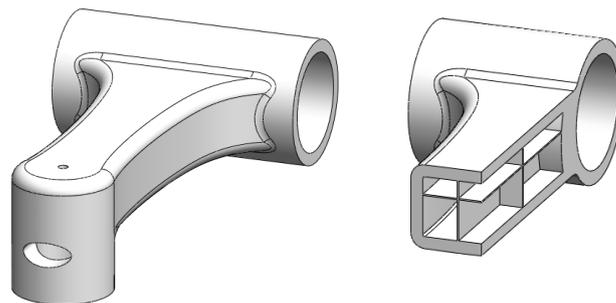


Ilustración 46. Diseño óptimo con estructuras celulares internas.

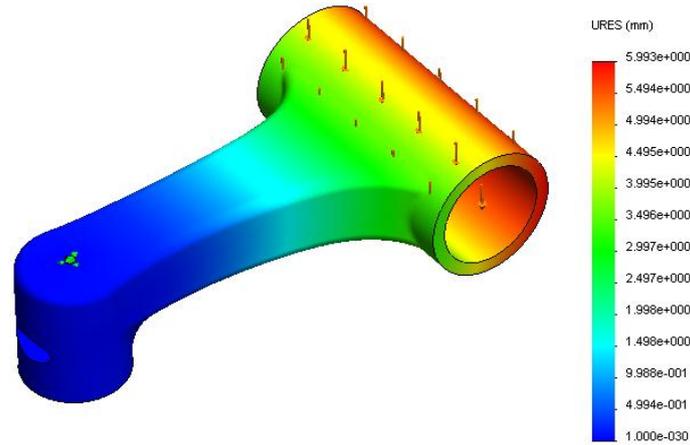


Ilustración 47. Campo de desplazamientos del diseño óptimo (codo bicicleta).

### 7.6.2. Resultados obtenidos para el problema de referencia (3 variables, evaluaciones con elementos finitos)

Posteriormente se aplicó el programa para optimizar la pala del microaerogenerador con de 3 variables de diseño. Los puntos simulados fueron:

Punto	L (mm)	e (mm)	e <sub>h</sub> (mm)	Desplazamiento (mm)	Masa (g)
1	20.000	3.000	3.000	16.061	1705.09
2	20.000	3.000	8.000	13.625	2273.12
3	20.000	8.000	3.000	11.338	2567.12
4	20.000	8.000	8.000	11.046	2777.22
5	60.000	3.000	3.000	19.025	1244.24
6	60.000	3.000	8.000	17.693	1499.71
7	60.000	8.000	3.000	11.684	2386.56
8	60.000	8.000	8.000	11.543	2470.68
9	40.000	5.500	5.500	12.741	2087.14
10	60.000	5.741	3.000	13.331	1921.24
11	60.000	4.521	3.000	15.070	1630.25
12	48.810	4.292	3.000	15.130	1623.41
13	37.455	4.034	3.000	14.904	1641.80
14	26.284	3.639	3.000	15.343	1694.90
15	52.544	4.469	3.000	14.963	1648.14
16	38.061	4.012	3.000	15.042	1630.31
17	37.646	4.016	3.000	14.974	1634.79

Tabla 32. Puntos evaluados durante la ejecución de la versión 6 del programa (problema de referencia con 3 variables).

El resultado obtenido en este caso es similar al de versiones anteriores del programa (1634.79g frente a 1632.44g o 1634.69g en dos pruebas de quinto programa). Sin embargo, el número de simulaciones requeridas disminuyó con respecto a otras versiones anteriores (17 evaluaciones frente a 22 ó 20 de las pruebas del quinto programa). Para este problema el SolidWorks obtiene un diseño óptimo de 1690.07g, un 3.38% peor que con la metodología planteada pero con 3 simulaciones menos (14 puntos evaluados frente a 17 del programa desarrollado).



Ilustración 48. Sección del diseño óptimo de la pala (3 variables).

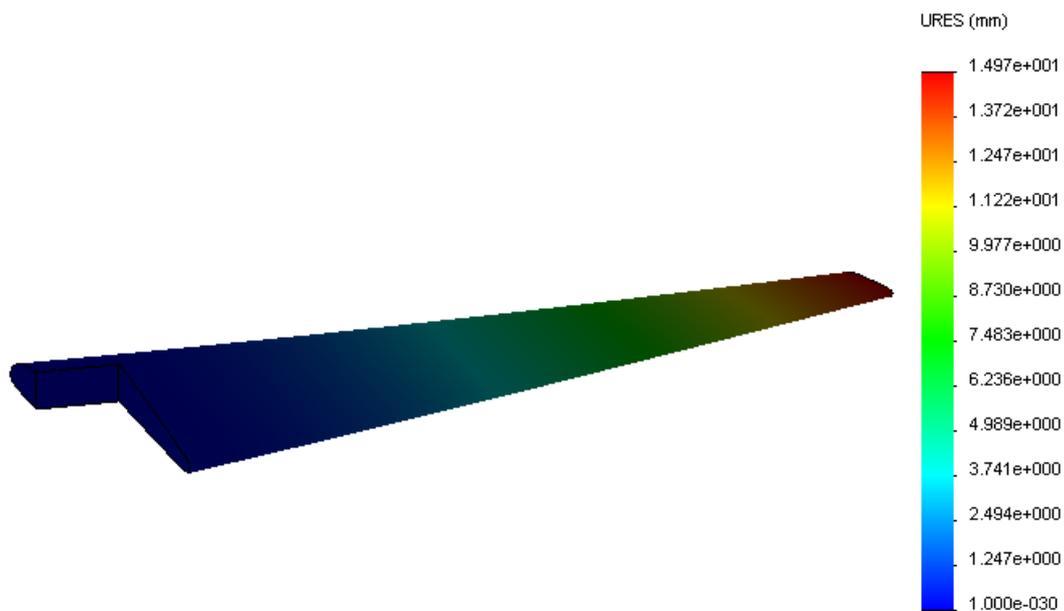


Ilustración 49. Campo de desplazamientos del diseño óptimo de la pala (3 variables).

### 7.6.3 Resultados obtenidos para el problema de referencia (4 variables, evaluaciones con elementos finitos)

Posteriormente se probó también el programa con el caso de 4 variables de diseño. Los puntos evaluados durante la optimización se muestran en la siguiente tabla:

Punto	L (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
1	20.000	3.000	3.000	20.000	16.061	1705.09
2	20.000	3.000	3.000	60.000	16.454	1578.69
3	20.000	3.000	8.000	20.000	13.621	2273.12
4	20.000	3.000	8.000	60.000	14.058	2092.02
5	20.000	8.000	3.000	20.000	11.338	2567.12
6	20.000	8.000	3.000	60.000	11.418	2521.56
7	20.000	8.000	8.000	20.000	11.046	2777.22
8	20.000	8.000	8.000	60.000	11.111	2713.70
9	60.000	3.000	3.000	20.000	18.324	1399.41
10	60.000	3.000	3.000	60.000	19.029	1244.24
11	60.000	3.000	8.000	20.000	16.555	1793.55
12	60.000	3.000	8.000	60.000	17.690	1499.71
13	60.000	8.000	3.000	20.000	11.554	2443.66
14	60.000	8.000	3.000	60.000	11.684	2386.56
15	60.000	8.000	8.000	20.000	11.332	2580.16
16	60.000	8.000	8.000	60.000	11.547	2470.68
17	40.000	5.500	5.500	40.000	12.745	2087.14
18	60.000	5.743	3.000	60.000	13.340	1921.53
19	60.000	4.528	3.000	60.000	15.035	1631.92
20	42.806	4.183	3.000	60.000	15.062	1605.87
21	25.821	3.759	3.000	60.000	15.290	1646.02
22	48.594	4.338	3.000	42.861	15.034	1645.93
23	44.415	4.111	3.000	24.817	15.122	1673.12
24	48.900	4.425	3.000	60.000	14.924	1640.00
25	42.907	4.235	3.050	59.641	15.008	1621.550
26	44.181	4.270	3.000	60.000	15.084	1620.230
27	43.203	4.284	3.000	60.000	14.951	1628.270

Tabla 33. Puntos evaluados durante la ejecución de la versión 6 del programa (problema de referencia con 4 variables).

El resultado óptimo en este caso obtiene una masa menor que en la versión anterior del programa (1628.27g frente a 1636.86g), y con una evaluación menos. Por otro lado, con la optimización del SolidWorks se obtiene una masa de 1678.64g, un 3.09% superior a la obtenida con el programa generado, pero con una simulación menos.



Ilustración 50. Sección del diseño óptimo de la pala (4 variables)

#### 7.6.4. Resultados obtenidos para el problema de referencia (5 variables, evaluaciones con elementos finitos)

Por último se aplicó el programa para el caso de 5 variables de diseño en el problema de referencia. Los puntos simulados fueron:

Punto	L1 (mm)	L2 (mm)	e (mm)	e <sub>h</sub> (mm)	h (mm)	Desplazamiento (mm)	Masa (g)
1	20.000	20.000	3.000	3.000	20.000	16.053	1705.09
2	20.000	20.000	3.000	3.000	60.000	16.454	1578.69
3	20.000	20.000	3.000	8.000	20.000	13.629	2273.12
4	20.000	20.000	3.000	8.000	60.000	14.058	2092.02
5	20.000	20.000	8.000	3.000	20.000	11.338	2567.12
6	20.000	20.000	8.000	3.000	60.000	11.418	2521.56
7	20.000	20.000	8.000	8.000	20.000	11.046	2777.22
8	20.000	20.000	8.000	8.000	60.000	11.111	2713.70
9	20.000	60.000	3.000	3.000	20.000	16.970	1568.35
10	20.000	60.000	3.000	3.000	60.000	17.336	1429.26
11	20.000	60.000	3.000	8.000	20.000	14.665	2072.28
12	20.000	60.000	3.000	8.000	60.000	15.373	1844.00
13	20.000	60.000	8.000	3.000	20.000	11.382	2513.01
14	20.000	60.000	8.000	3.000	60.000	11.468	2462.31
15	20.000	60.000	8.000	8.000	20.000	11.207	2693.54
16	20.000	60.000	8.000	8.000	60.000	11.334	2610.61
17	60.000	20.000	3.000	3.000	20.000	16.897	1552.41
18	60.000	20.000	3.000	3.000	60.000	17.383	1411.64
19	60.000	20.000	3.000	8.000	20.000	14.538	2052.83
20	60.000	20.000	3.000	8.000	60.000	15.258	1819.79
21	60.000	20.000	8.000	3.000	20.000	11.476	2503.79
22	60.000	20.000	8.000	3.000	60.000	11.542	2452.20

23	60.000	20.000	8.000	8.000	20.000	11.194	2685.23
24	60.000	20.000	8.000	8.000	60.000	11.321	2600.10
25	60.000	60.000	3.000	3.000	20.000	18.323	1399.41
26	60.000	60.000	3.000	3.000	60.000	19.030	1244.24
27	60.000	60.000	3.000	8.000	20.000	16.555	1793.55
28	60.000	60.000	3.000	8.000	60.000	17.690	1499.71
29	60.000	60.000	8.000	3.000	20.000	11.554	2443.66
30	60.000	60.000	8.000	3.000	60.000	11.684	2386.56
31	60.000	60.000	8.000	8.000	20.000	11.330	2580.16
32	60.000	60.000	8.000	8.000	60.000	11.547	2470.68
33	40.000	40.000	5.500	5.500	40.000	12.742	2087.14
34	60.000	60.000	5.743	3.000	60.000	13.318	1921.62
35	60.000	60.000	4.514	3.000	60.000	15.009	1628.57
36	60.000	37.373	4.319	3.000	60.000	14.675	1627.97
37	37.240	45.196	4.061	3.000	60.000	15.289	1588.02
38	60.000	42.650	4.155	3.000	38.640	15.052	1608.91
39	20.000	44.123	3.719	3.000	46.227	15.374	1644.22
40	60.000	20.615	4.063	3.000	45.744	14.962	1674.14
41	60.000	38.733	4.195	3.000	60.000	14.961	1593.30
42	60.000	39.901	4.199	3.000	60.000	14.946	1590.44

Tabla 34. Puntos evaluados durante la ejecución de la versión 6 del programa (problema de referencia con 5 variables).

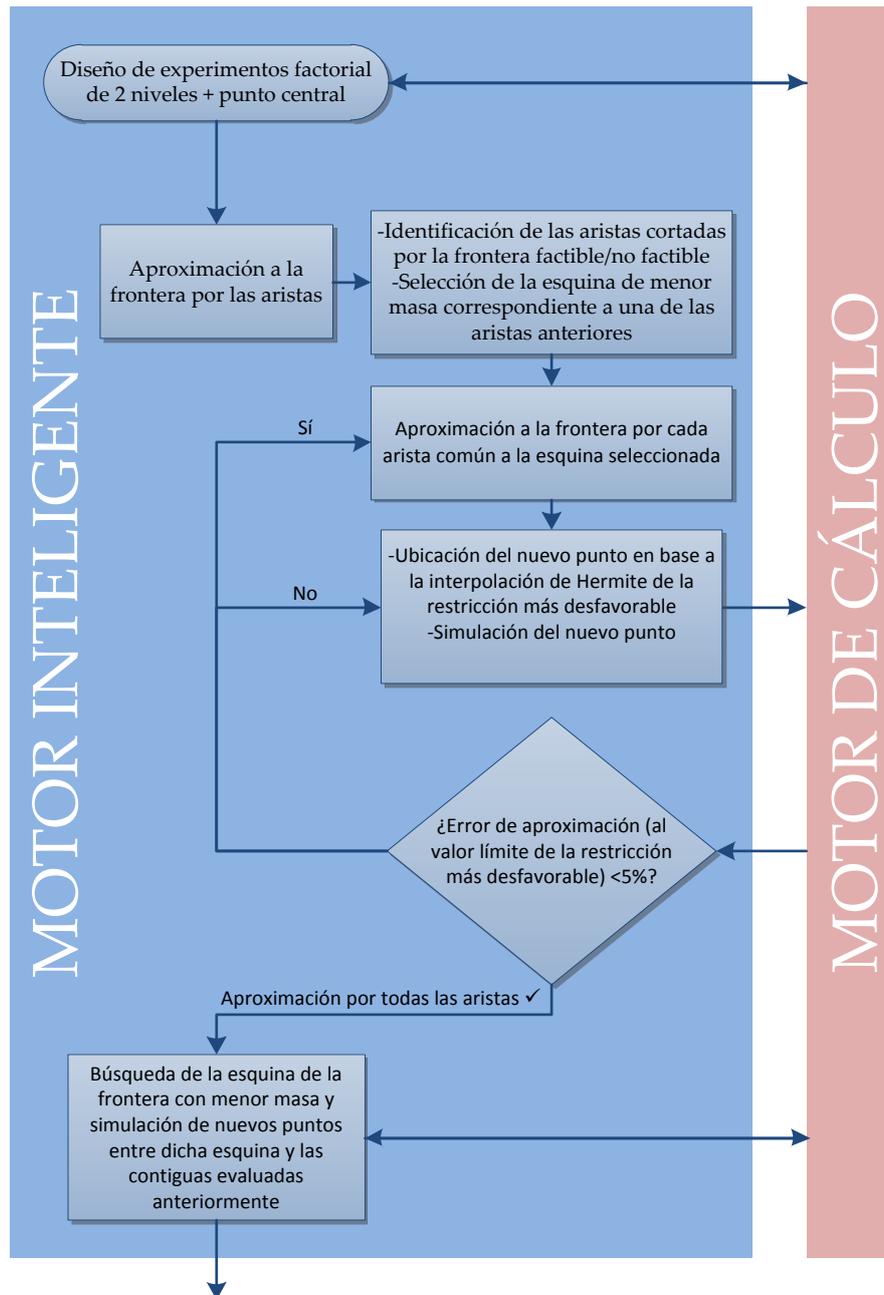
El diseño óptimo alcanzado tiene una masa de 1590.44g, ligeramente inferior a la obtenida con la versión anterior del programa (1600.64g) y con una simulación menos. Este mismo problema resulto con la optimización del SolidWorks (diseño de experimentos de Box-Behnken, obtención de las superficies de respuesta, estimación del óptimo y evaluación del mismo) obtiene una geometría final de 1659.7g. En términos porcentuales, la metodología implementada en este documento mejora el diseño óptimo en este caso concreto un 4.35% en términos de masa empleando el mismo número de simulaciones.

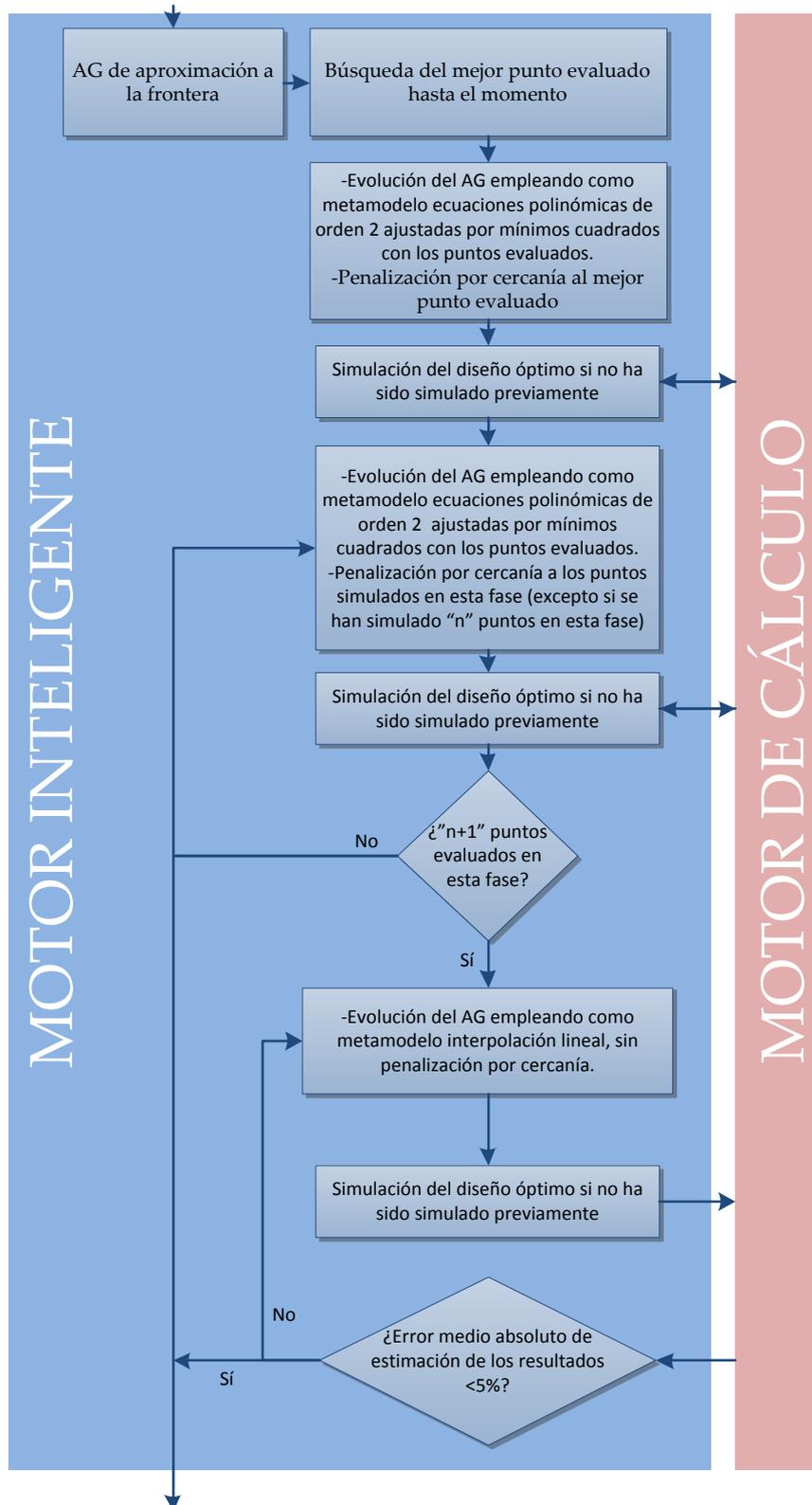


Ilustración 51. Sección del diseño óptimo de la pala (5 variables).

## 8. Metodología final

Tras comprobar que la última versión del programa obtiene buenos resultados se dio por válida la metodología establecida. La estrategia de optimización se basa por tanto en el pseudocódigo descrito para la sexta versión del programa. A continuación se resume la metodología de optimización en un diagrama de flujo:





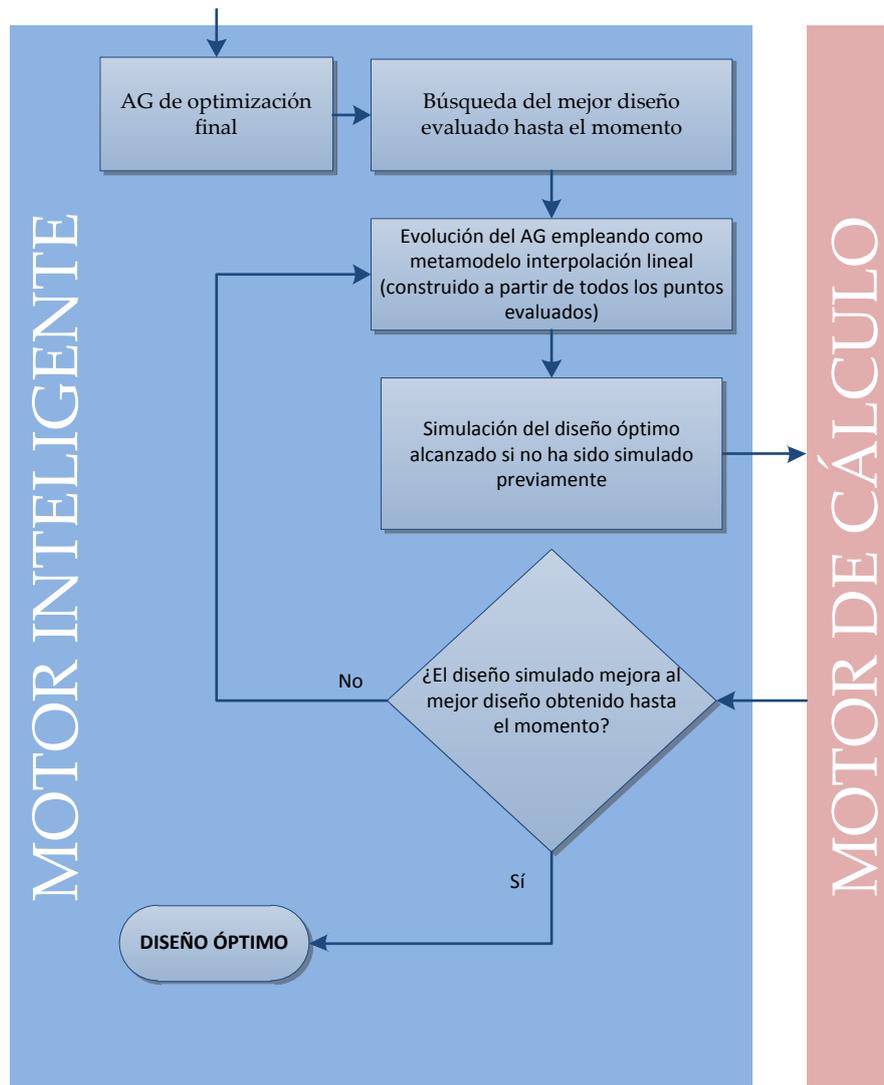


Ilustración 52. Diagrama de flujo de la metodología final de optimización.

El número de puntos de muestreo necesarios para la metodología vendría dado por:

- “ $2^{n+1}$ ” puntos iniciales (“ $n$ ”= $n^{\circ}$  de variables de diseño).
- “1” punto o más en la aproximación a la frontera por las aristas.
- “ $n+1$ ” puntos o más en el AG de aproximación a la frontera.
- “1” punto o más en el AG final.

$$Puntos\ de\ muestreo_{metodología} \geq (2^n + 1) + (1) + (n + 1) + (1) \geq 2^n + n + 4$$

En el caso de la metodología de diseño de experimentos de Box-Behnken y optimización mediante superficies de respuesta, se requieren los puntos del diseño de experimentos más el punto óptimo estimado:

$$Puntos\ de\ muestreo_{O.B.B.+sup.respuesta} = (2^m \cdot C_n^m + 1) + (1) = 2^m \cdot C_n^m + 2$$

Siendo “m” el número de variables agrupadas y “ $C_n^m$ ” la combinatoria de “n” variables tomadas de “m” en “m”.

En la siguiente tabla se muestra el número de puntos necesarios de la metodología desarrollada frente a la optimización con diseño de experimentos de Box-Behnken y superficies de respuesta para diferente número de variables:

Nº variables de diseño ("n")	Nº puntos necesarios		
	Box-Behnken + superficies de respuesta		Metodología desarrollada
3	(m=2)	(12+1)+1=14	≥15
4	(m=2)	(24+1)+1=26	≥24
5	(m=2)	(40+1)+1=42	≥41
6	(m=3)	(48+1)+1=50	≥74
7	(m=3)	(56+1)+1=58	≥139

Tabla 35. Puntos de muestreo necesarios para la metodología planteada y la optimización basada en diseño de experimentos de Box-Behnken y superficies de respuesta.

A medida que aumenta el número de variables de diseño el crecimiento exponencial del muestreo inicial ( $2^n$ ) requerido en la metodología planteada implica un mayor número de puntos de muestreo que para el diseño de experimentos de Box-Behnken. Sin embargo hay que decir que el número de variables de diseño para las estructuras celulares no será superior a 7, siendo lo más habitual un valor entre 3 y 5 variables, por lo que el número de puntos requeridos por la metodología es aceptable.

### 8.1. Ventajas de la metodología

Las principales ventajas de la metodología son las siguientes:

- Permite obtener un diseño bastante optimizado con un bajo número de puntos de muestreo, disminuyendo por tanto el tiempo de diseño. Como se ha comprobado, mejora los resultados obtenidos por el diseño de experimentos de Box-Behnken y estimación del óptimo mediante superficies de respuesta, con un número similar de simulaciones. El diseño de experimentos de Box-Behnken

y superficies de respuesta está considerado el método más eficiente de estimación para problemas con pocas variables de diseño, lo cual añade valor a la metodología planteada (22). Otros métodos de optimización como los basados directamente en algoritmos genéticos requieren de un número mucho mayor de puntos de muestreo (del orden de miles de simulaciones) para converger hacia el diseño óptimo.

- Los bucles iterativos implementados garantizan obtener una solución al finalizar el proceso de optimización. En el caso de la optimización del SolidWorks, si el diseño óptimo estimado no cumple las restricciones por un error en la predicción, entonces la optimización se detendrá y no se obtendrá una solución al problema de optimización.
- A pesar de que la metodología está orientada para la optimización de estructuras celulares en el interior de piezas fabricadas por “Additive Manufacturing”, pueden existir otras aplicaciones en las que sea muy eficiente. Para ello es necesario que se tenga una relación monótona entre cada variable de diseño y las diferentes respuestas (función siempre creciente o decreciente), así como un único objetivo que esté en conflicto con el resto de restricciones. En otras palabras, el incremento de una variable de diseño debe suponer un incremento en todas las restricciones y un decremento en el objetivo, o viceversa.

## 8.2. Inconvenientes de la metodología

Los aspectos negativos de esta metodología son los siguientes:

- Al tratarse de un programa implementado específicamente para resolver la optimización de estructuras celulares en piezas de fabricación aditiva, su aplicación para otros problemas de optimización estará limitada a las condiciones explicadas en el punto anterior, de modo que funcionará correctamente si se dan dichas condiciones.
- A medida que aumenta el número de variables de diseño el crecimiento del número de puntos de muestreo necesarios para la optimización crece exponencialmente. Sin embargo para esta aplicación concreta el número máximo de variables será 7, siendo lo más habitual entre 3 y 5 variables, por lo que para esta aplicación no sería un problema.

### 8.3. Otras limitaciones

A la hora de aplicar la metodología es importante tener en cuenta otras limitaciones referentes exclusivamente al motor de cálculo y por tanto ajenas al propio método de optimización. En el caso concreto de emplear el cálculo por elementos finitos se plantean las siguientes limitaciones:

- El modelado del problema debe aproximarse lo más posible a la realidad, por lo que es de suma importancia establecer correctamente las condiciones de contorno, así como el material o el tipo de estudio (lineal, no lineal, etc.).
- El mallado también debe ser lo suficientemente fino como para obtener unos resultados cercanos a la realidad.
- Para poder automatizar el proceso es necesario que la configuración de malla establecida permita realizar el mallado de las diferentes geometrías sin que se produzcan problemas en la discretización.
- Por último también es importante que las modificaciones en el mallado de un diseño con respecto a otro no tenga influencia en los resultados.

## 9. Líneas futuras

### 9.1. Automatización del flujo de trabajo entre motor de cálculo y motor inteligente

El primer aspecto en el que habría que trabajar en líneas futuras sería en la automatización del proceso de interacción entre el motor de diseño-cálculo y el motor inteligente. Con esto se pretende reducir el tiempo de diseño, convirtiendo esta herramienta en un aspecto muy útil a la hora de lanzar un nuevo producto al mercado. La meta final es que el usuario se encargue únicamente de definir la geometría y su parametrización en el programa CAD, así como los tipos de análisis y resultados a tener en cuenta en la optimización. Una vez definidos los aspectos relativos al motor de cálculo, simplemente ejecutaría el programa de optimización indicando previamente las variables que entran en juego en la optimización así como los intervalos del dominio y las restricciones del problema con sus valores límites y zona factible. Con esta información el programa de optimización empezaría su proceso de muestreo aportando al programa CAD y de cálculo los valores de las variables del diseño que se pretende evaluar. El motor de cálculo lee los valores y ejecuta los análisis definidos previamente y, tras obtener los resultados, transmite la información de interés al programa de optimización. Este proceso se repite para cada uno de los puntos de muestreo que requiera el programa de optimización.

Para conseguir esto y en base a la experiencia adquirida durante este trabajo fin de máster, se recomienda crear un programa a modo de macro bien en el SolidWorks o en CATIA para incorporar la metodología de optimización en el propio software de diseño y cálculo y así poder automatizar el proceso. Se trata de una tarea bastante compleja por los aspectos de programación, sobre todo vinculados a acceder a cierta información del propio software, pero que podría dar lugar a una metodología útil y muy interesante para la optimización de estructuras celulares para piezas de fabricación aditiva. Esta misma opción podría aplicarse en el ANSYS incluso a través de un programa en FORTRAN, pero existen ciertas limitaciones en la generación automatizada de la geometría que descartan este software. Posteriormente se podrían mejorar otros aspectos como el interfaz gráfico de la aplicación para facilitar su uso.

## 9.2. Metodología para problemas de optimización con mayor número de variables de diseño

En el caso de problemas con mayor número de variables, la metodología planteada se ve limitada por el crecimiento exponencial del muestreo inicial y también por el coste computacional de la triangulación de Delaunay cuando se emplean más de 7 variables de diseño. El muestreo factorial de 2 niveles se eligió entre otros aspectos para obtener una envolvente convexa que abarque todo el dominio y por tanto pueda aplicarse la interpolación lineal en cualquier punto. A pesar de que a priori el número de variables de diseño no será superior a 7, se podría abrir otra línea de trabajo para un número mayor de variables de diseño. Una opción sería emplear una metodología similar a la planteada pero realizando un diseño de experimentos con menor intensidad de muestreo (sobre todo a medida que aumenta el número de variables), como el diseño de experimentos de Box-Behnken, emplear ecuaciones polinómicas de orden 2 ajustadas por mínimos cuadrados para la fase de aproximación a la frontera, y finalmente sustituir la interpolación lineal por un metamodelo que pueda ser aplicado en el dominio exterior a la envolvente convexa de los datos, como podrían ser los mínimos cuadrados móviles (32). Este método de aproximación (no es interpolador ya que no da resultados exactos en los puntos de muestreo) estima el resultado en un punto a partir de la suma ponderada de los datos existentes:

$$u(\vec{x}) = \sum_{l=1}^n \Phi_l(\vec{x})u_l$$

Siendo “ $u$ ” el valor estimado para el punto definido por el cromosoma “ $\vec{x}$ ”, “ $n$ ” el número de puntos dato, “ $u_l$ ” el valor de la respuesta para el punto “ $l$ ” y “ $\Phi_l$ ” es la función peso, que viene determinada por:

$$\Phi_l(\vec{x}) = \frac{w(\vec{x} - \vec{x}_l)}{\sum_{i=1}^n w(\vec{x} - \vec{x}_i)}$$

$$w(\vec{x} - \vec{x}_l) = \begin{cases} e^{-\frac{\|\vec{x} - \vec{x}_l\|^2}{R^2}} & \text{si } \|\vec{x} - \vec{x}_l\| \leq R \\ 0 & \text{si } \|\vec{x} - \vec{x}_l\| > R \end{cases}$$

Siendo “ $R$ ” es el radio de influencia ponderado al intervalo de cada variable y “ $\|\vec{x} - \vec{x}_l\|$ ” la distancia ponderada (al intervalo de cada variable) del punto de interés al punto dato (“ $\vec{x}_l$ ”). El radio de influencia ponderado (“ $R$ ”) interesa que varíe dependiendo de la distribución de los datos con respecto al punto de estimación. Un valor recomendado es 0.4 veces la distancia media (ponderada) con respecto a los puntos dato (“ $R = 0.4 \cdot R_{medio}$ ”).

El hecho de calcular las distancias con una “ponderación” se debe a que algunas variables de diseño tendrán un intervalo de trabajo menor, por lo que su influencia tendría menos peso. Por ello, la distancia ponderada (“ $\|\vec{x} - \vec{x}_l\|$ ”) entre el punto de interés y el dato “ $l$ ” se calcula amplificando el intervalo de las variables con menor dominio:

$$\|\vec{x} - \vec{x}_l\| = \sqrt{\sum_{j=1}^{n^{\circ}\text{variables}} \left[ ((x)_j - (x_l)_j) \cdot \frac{\text{Intervalo}_j}{\text{Intervalo mayor}} \right]^2}$$

Siendo “ $n^{\circ}\text{variables}$ ” el número de variables de diseño (o dimensión del problema), “ $(x)_j$ ” el valor de la variable “ $j$ ” para el punto de estimación, “ $(x_l)_j$ ” el valor de la variable “ $j$ ” del punto “ $l$ ”, “ $\text{Intervalo}_j$ ” el intervalo de búsqueda de la variable “ $j$ ”, y por último “ $\text{Intervalo mayor}$ ”, que hace referencia al mayor de los intervalos de trabajo de las diferentes variables de diseño.

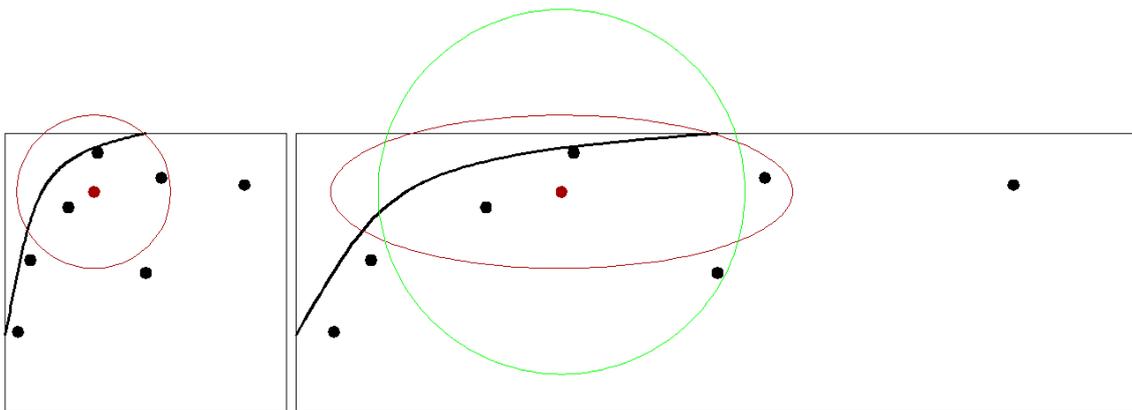
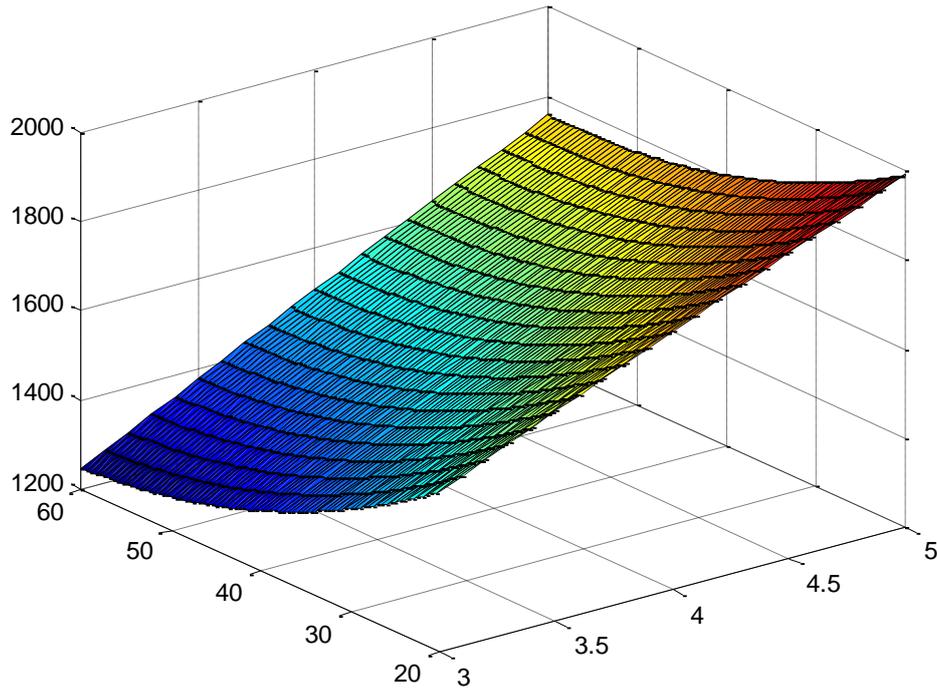
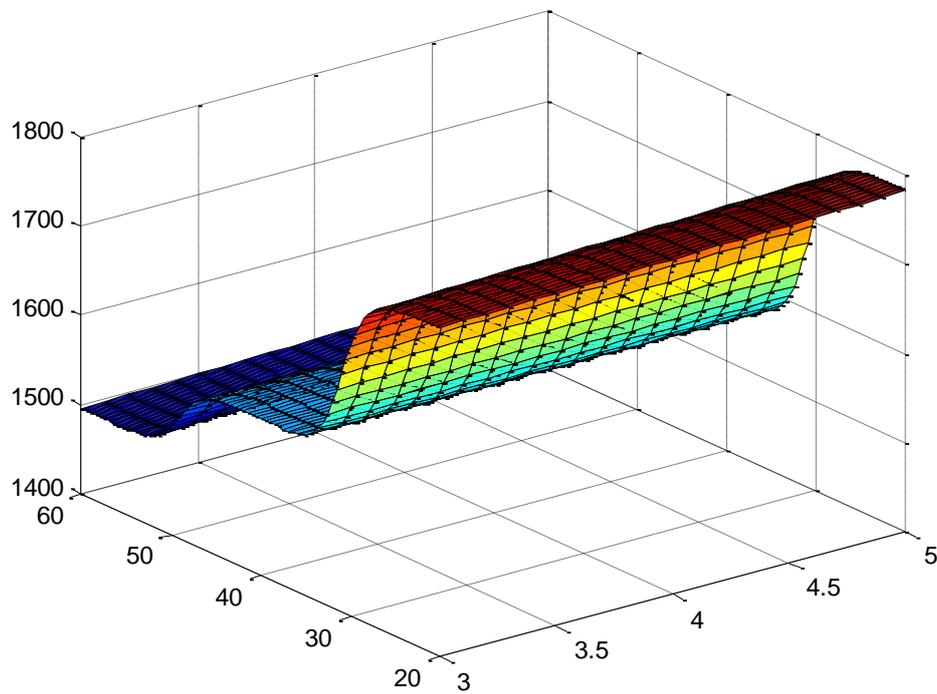


Ilustración 53. Dominio cuadrado (izquierda) y rectangular (derecha). El radio de influencia debe ponderarse según el intervalo de cada variable.

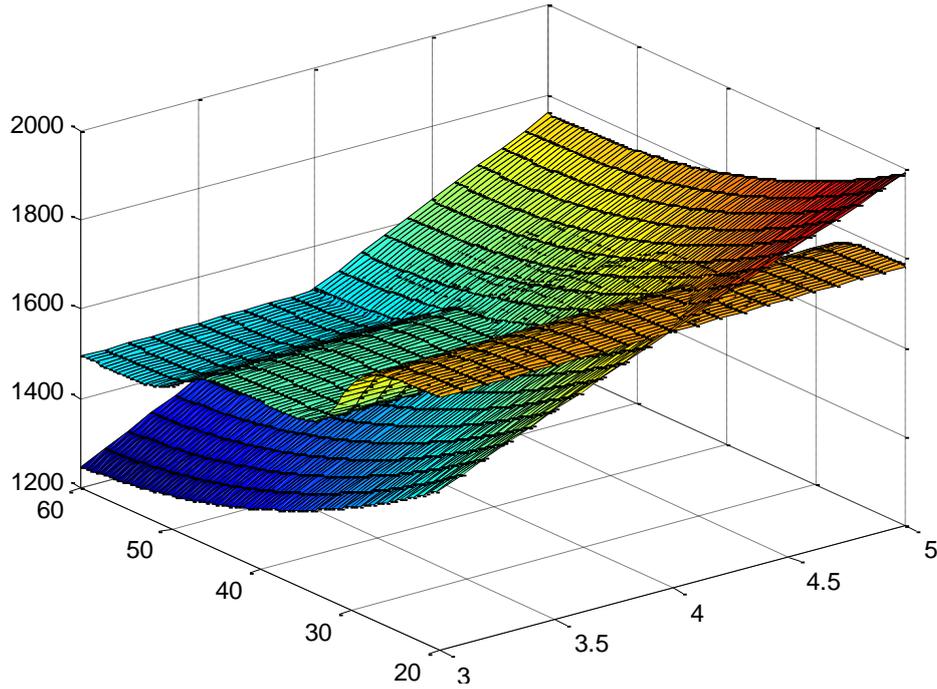
En la imagen derecha anterior se aprecia claramente como el radio de influencia ponderado (en rojo) circunscribe los mismos puntos que en el caso de dominio cuadrado. Sin embargo si no se tiene en cuenta esta ponderación en el cálculo de las distancia, el radio de influencia (en verde) abarcará a otros puntos y restará influencia a la variable de menor intervalo. Esto mismo se aprecia en las siguientes gráficas, donde se muestra la respuesta real en un caso de 2 variables de diseño así como la respuesta estimada sin ponderación y con ponderación.



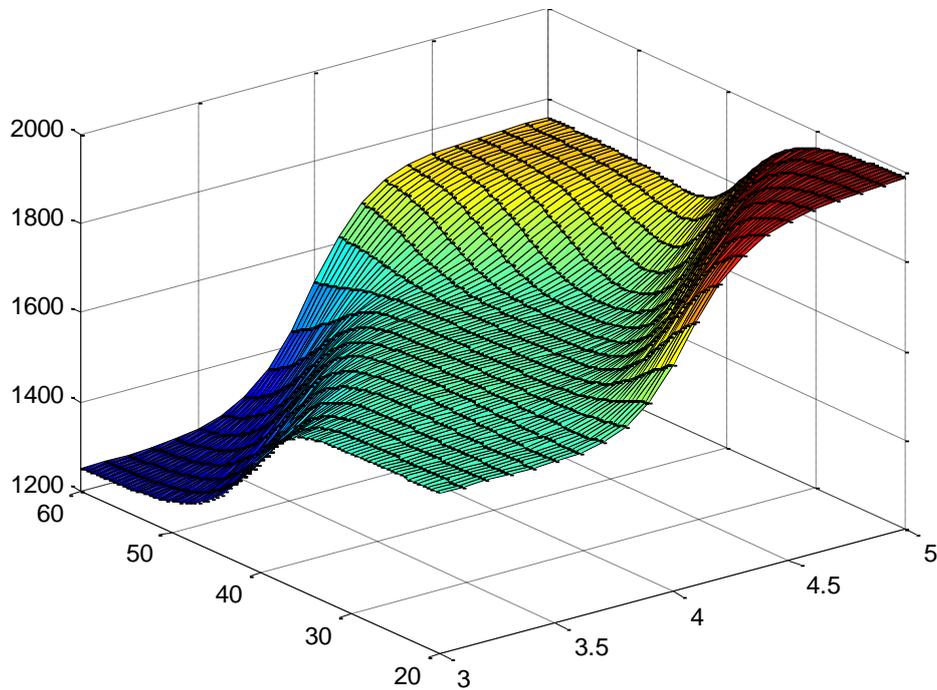
Gráfica 14. Respuesta real del sistema.



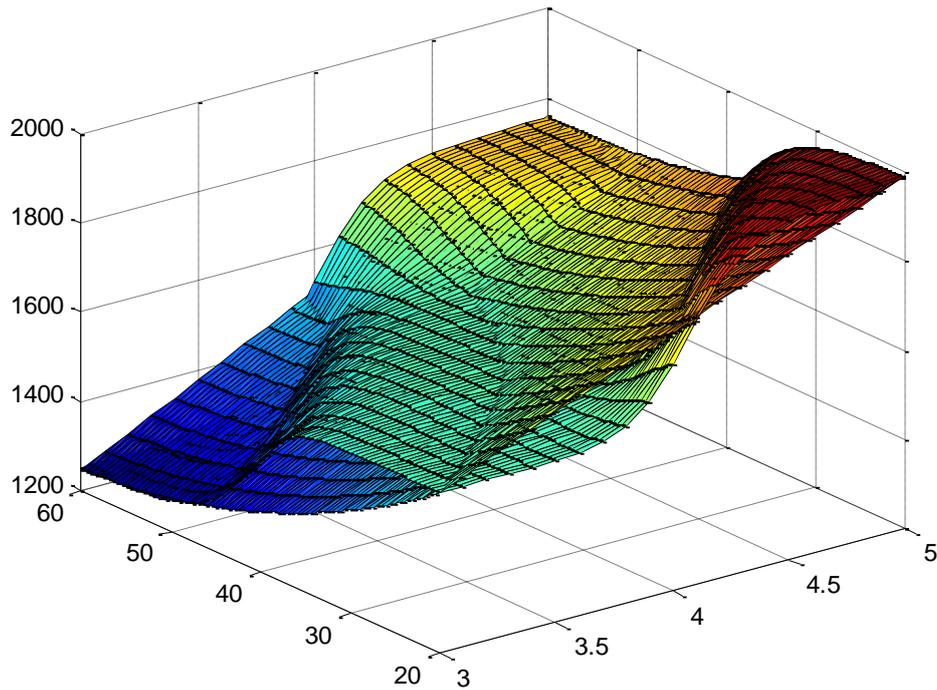
Gráfica 15. Respuesta estimada con mínimos cuadrados móviles sin ponderación en la distancia (5 puntos dato: 4 esquinas y punto central).



Gráfica 16. Respuesta real y respuesta estimada con mínimos cuadrados móviles sin ponderación en la distancia (5 puntos dato: 4 esquinas y punto central).



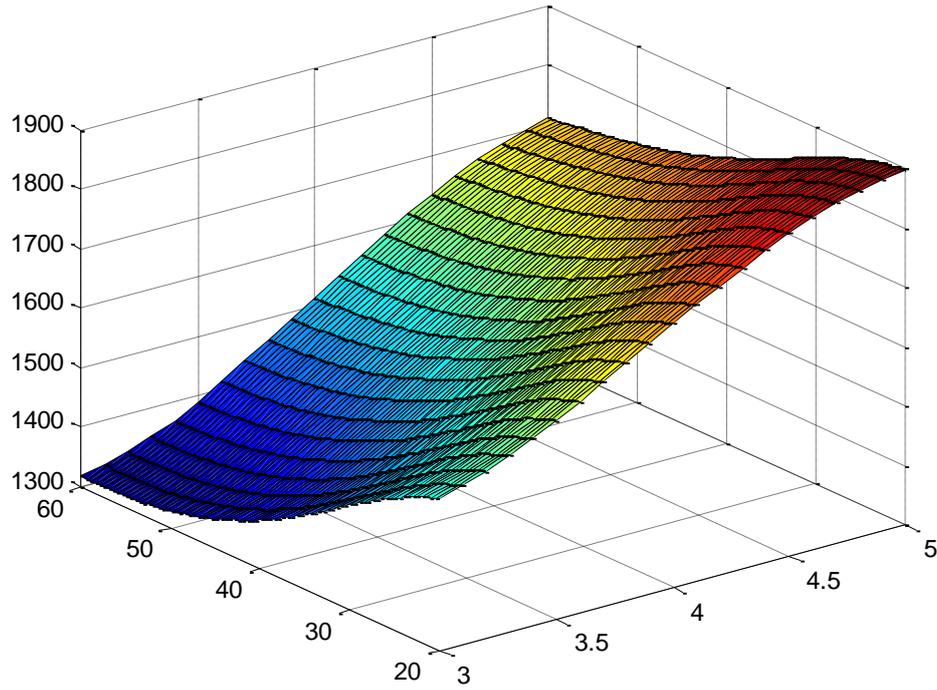
Gráfica 17. Respuesta estimada con mínimos cuadrados móviles con ponderación en la distancia (5 puntos dato: 4 esquinas y punto central).



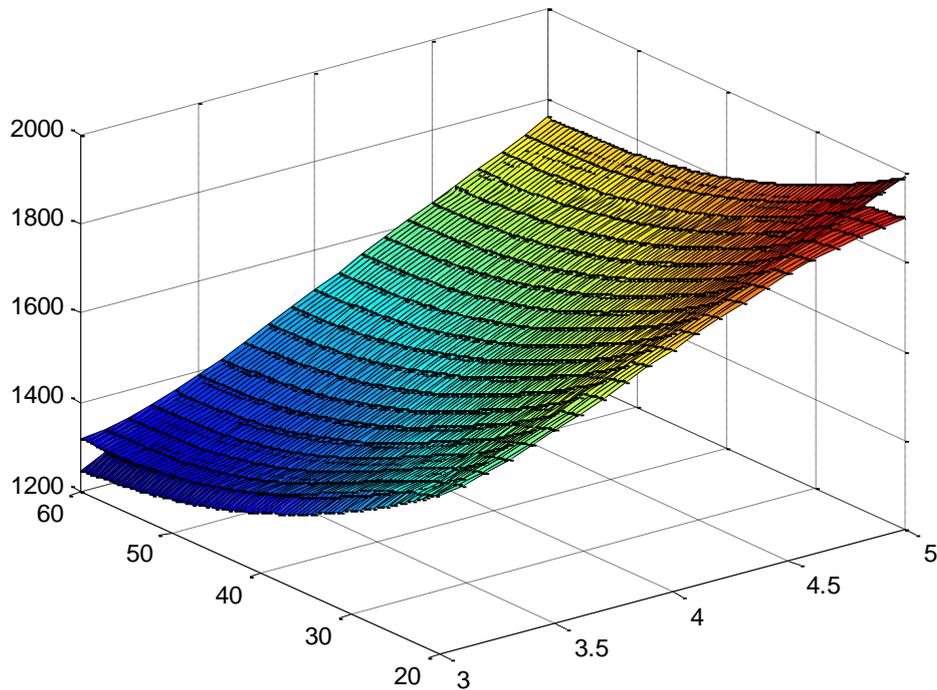
Gráfica 18. Respuesta real y respuesta estimada con mínimos cuadrados móviles con ponderación en la distancia (5 puntos dato: 4 esquinas y punto central).

Se aprecia claramente que es importante realizar la ponderación al intervalo de cada variable en el cálculo de la distancia para obtener una “buena” aproximación. Se aprecia claramente como la respuesta estimada por mínimos cuadrados adopta la forma de una superficie suave con “ondas” asociadas a los puntos de muestreo. La precisión de ésta con respecto a la respuesta real difiere significativamente debido a que sólo se han incluido 5 puntos de muestreo (4 esquinas y punto central).

Al aumentar el número de puntos de muestreo la aproximación por mínimos cuadrados móviles se ajusta más a los valores reales. En las siguientes imágenes se aprecia que con una rejilla de 5x5 (25 puntos) uniforme la respuesta estimada por los mínimos cuadrados móviles se aproxima bastante a la respuesta real. También cabe destacar que la respuesta estimada en las esquinas no coincide exactamente con la respuesta real a pesar de tratarse de puntos dato empleados para generar el metamodelo. Esto se debe a que los mínimos cuadrados móviles es un método de aproximación, no un método interpolador, por lo que los resultados estimados en los puntos dato serán aproximados a los reales pero no coincidirán necesariamente.



Gráfica 19. Respuesta estimada con mínimos cuadrados móviles (rejilla de datos de 5x5=25 puntos).



Gráfica 20. Respuesta real y respuesta estimada con mínimos cuadrados móviles (rejilla de datos de 5x5=25 puntos).

### 9.3. Búsqueda exhaustiva de las herramientas comerciales más apropiadas para el diseño y cálculo

Por otro lado no hay que olvidar que el objetivo de esta línea de investigación consiste en crear una metodología final de trabajo lo más simple posible para que pueda ser empleada fácilmente por usuarios de tecnologías de fabricación aditiva, priorizando la aplicabilidad de la metodología frente a otros aspectos. Esto conllevaría un análisis exhaustivo de los programas de diseño CAD y cálculo por elementos finitos existentes en el mercado para evaluar las posibilidades y limitaciones de cada uno, principalmente en cuanto a libertad en el diseño y automatización de las geometrías celulares. Nuevamente, en base a la experiencia adquirida, se recomienda el SolidWorks o el CATIA, tomando ventaja este último por tener más potencial sobre todo en aspectos de diseño.

### 9.4. Estudio de las tipologías celulares más apropiadas

También sería importante estudiar las tipologías más apropiadas para la generación de las estructuras celulares teniendo en cuenta aspectos como el tiempo de trabajo requerido por el diseñador para generar la geometría y automatizar la regeneración de la misma, el tiempo de cómputo necesario para regenerar la geometría al cambiar una de las variables de diseño, la eficiencia estructural de las geometrías en términos de rigidez relativa al peso, etc. Hasta ahora el mejor rendimiento se ha obtenido con celdas huecas con forma de paralelepípedos. No obstante sería interesante hacer un estudio exhaustivo y comparar los resultados de esta tipología con otras como estructuras de barras con diferentes diseños y tratar de sacar unas conclusiones.

### 9.5. Análisis de las herramientas de mallado más adaptables para la discretización de diferentes geometrías durante la optimización

Uno de los principales problemas de las herramientas de optimización disponibles en el mercado es que cuando se produce un fallo por problemas de mallado, la optimización no se lleva a cabo. A pesar de que este tema diverge del tema

central de este trabajo, sería conveniente analizar los tipos de mallado más apropiados en el software de cálculo por elementos finitos para evitar estos problemas al aplicar el método de optimización.

### **9.6. Análisis de las posibles restricciones asociadas al proceso de fabricación**

Finalmente, también sería conveniente analizar las posibles restricciones asociadas al proceso de fabricación. Para este tipo de aplicaciones la tecnología FDM aparenta un mayor potencial que el resto por su mayor capacidad para fabricación de piezas de mayor escala y por la posibilidad de generar geometrías huecas con piel exterior sin depositar material soporte. No obstante, en el caso de que se requiera material soporte para los voladizos de las celdas interiores, el incremento de peso será despreciable y por tanto se podría aplicar la metodología de optimización obteniendo resultados ligeramente conservadores por la rigidez adicional que aportaría este material y ligeramente benévolos en cuanto a masa. Independientemente de esta consideración, será necesario analizar algunas limitaciones constructivas que puedan afectar al límite de las variables de diseño como por ejemplo el espesor mínimo de pared. Por último, en el caso de piezas multimaterial también sería interesante realizar un estudio para valorar el tipo de contacto más apropiado entre componentes para el cálculo por elementos finitos.

## 10. Referencias

1. **Ian Gibson, David W. Rosen, Brent Stucker.** *Additive Manufacturing Technologies: Rapid Prototyping to Direct Digital Manufacturing.* s.l. : Springer, 2009. pág. 462. ISBN 1441911197, 9781441911193.
2. *Design for Additive Manufacturing of Cellular Structures.* **Chen Chu, Greg Graf and David W. Rosen.** 5, Georgia : s.n., 2008, Computer-Aided Design and Applications, Vol. 5, págs. 686-696.
3. **Bartolo, Paulo Jorge da Silva.** *Virtual and rapid manufacturing. Advanced Research in Virtual and Rapid Prototyping.* s.l. : Taylor & Francis, 2007. pág. 860. ISBN 0415416027, 9780415416023.
4. *Computer-aided Porous Scaffold Design for Tissue Engineering Using Triply Periodic Minimal Surfaces.* **Yoo, Dong-Jin.** 1, Pocheon-Si, Daejin University, Department of Computer Aided Mechanical Design Engineering : Springer, Febrero de 2011, INTERNATIONAL JOURNAL OF PRECISION ENGINEERING AND MANUFACTURING, Vol. 12, págs. 61-71. 487-711.
5. **Rubén Paz Hernández, Mario Monzón Verona, Fernando Ortega García.** Análisis teórico y experimental de nuevas estructuras aligeradas en la conformación de piezas plásticas obtenidas a partir de fabricación aditiva. *Proyecto Final de Carrera.* Las Palmas de Gran Canaria : s.n., 2010.
6. 9/14/2006, STATGRAPHICS – Rev. Diseño de Experimentos – Diseño de Superficie de Respuesta.
7. **Mario Monzón Verona, Rubén Paz Hernández, Fernando Ortega García.** New methodology for lightening parts made from additive manufacturing technologies. [ed.] International Union of Theoretical and Applied Mechanics. *XXIII International Congress of Theoretical and Applied Mechanics (ICTAM2012).* Pekín : s.n., Agosto de 2012. ISBN 978-988-16022-3-7.
8. **Hongqing Wang, David W. Rosen.** PARAMETRIC MODELING METHOD FOR TRUSS STRUCTURES. *ASME Design Engineering Technical Conferences.* Montreal : s.n., 29 de Septiembre de 2002. DETC02/CIE-34495.
9. **Hongqing Wang, Yong Chen, David W. Rosen.** A Hybrid Geometric Modeling Method for Large Scale Conformal Cellular Structures.

10. **Hongqing Wang, David Rosen.** Computer-Aided Design Methods for Additive Fabrication of Truss Structures. *Master Thesis, School of Mechanical Engineering, Georgia Institute Of Technology.* 2001.
11. **A. Vaidya, S. Yang, J. St. Ville, D. T. Nguyen, S. D. Rajan.** Multiphysics CAD-Based Design Optimization. 2005.
12. **Roman GÄatzi, Marion Uebersax, Oliver KÄonig.** Structural Optimization Tool using Genetic Algorithms and Ansys. *CAD-FEM User's Meeting Schweiz 2000.* Schweiz : s.n., 2000.
13. **Oliver KÄonig, Marc Wintermantel.** CAD-based Evolutionary Design Optimization with CATIA V5. *EVEN - Evolutionary Engineering AG.* Zurich : s.n.
14. **Mariano Vázquez Espí, Jaime Cervera Bravo.** Recocido simulado: un nuevo algoritmo para la optimización de estructuras. *Departamento de Estructuras de Edificación. Escuela Técnica Superior de Arquitectura de Madrid.* 1994.
15. **JIN-HWAN LEE, SEOK-CHEOL HWANG, JOON HONG PARK, KWON-HEE LEE.** Structural Design Examples Using Metamodel-Based Approximation Model. *9th WSEAS Int. Conference on APPLIED COMPUTER and APPLIED COMPUTATIONAL SCIENCE.* ISSN: 1790-5117 ISBN: 978-960-474-173-1.
16. *Multiobjective design optimisation of coronary stents.* **Sanjay Pant, Georges Limbert, Nick P. Curzen, Neil W. Bressloff.** s.l. : ELSEVIER, 2011, Biomaterials, Vol. 32, págs. 7755-7773.
17. **Hyoung-Seog Chung, Juan J. Alonso.** Multiobjective Optimization Using Approximation Model-Based Genetic Algorithms. *10th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.* New York : American Institute of Aeronautics and Astronautics Paper, 2004. AIAA 2004-4325.
18. **Tien-Tsin Wong, Wai-Shing Luk, Pheng-Ann Heng.** Sampling with Hammersley and Halton Points.
19. **Hans-Joachim Bungartz, Michael Griebel.** Sparse grids. [ed.] Cambridge University Press. 2004. págs. 1-123. DOI: 10.1017/S0962492904000182.
20. **ROZZA, GIANLUIGI.** AN INTRODUCTION TO REDUCED BASIS METHOD FOR PARAMETRIZED PDEs. *Chair of Modelling and Scientific Computing.* Ecole Polytechnique Fédérale de Lausanne, Switzerland : s.n., 2009.

21. **G. Gary Wang, Zuomin Dong, Peter Aitchison.** Adaptive Response Surface Method – A Global Optimization Scheme for Approximation-based Design Problems.
22. *An efficient algorithm for Kriging approximation and optimization with large-scale sampling data.* **S. Sakata, F. Ashida, M. Zako.** 193, s.l. : ELSEVIER, 2004, Computers methods in applied mechanics and engineering, págs. 385-404.
23. **Fallas, Jorge.** Modelos digitales de elevación: Teoría, métodos de interpolación y aplicaciones. 2007.
24. —. MODELO DE ELEVACIÓN DIGITAL PARA LAS HOJAS CARTOGRÁFICAS TILARÀN Y JUNTAS ESCALA 1:50.000 DEL INSTITUTO GEOGRAFICO NACIONAL, COSTA RICA. *Universidad Nacional, Laboratorio de Teledetección y Sistemas de Información Geográfica.* 2003.
25. **Dressle, Mgr. Miroslav.** Art of Surface Interpolation. KUNŠTÁT : s.n., 2009.
26. **Mario Villatoro, Carlos Henríquez, Freddy Sancho.** COMPARACIÓN DE LOS INTERPOLADORES IDW Y KRIGING EN LA VARIACIÓN ESPACIAL DE pH, Ca, CICE y P DEL SUELO. *Agronomía Costarricense.* 2008. 95-105. ISSN:0377-9424.
27. **Marina Brovka, José María Escobar Sánchez, Rafael Alejandro Montenegro Armas.** Aplicación del análisis isogeométrico a un problema bidimensional de Poisson. *Trabajo Fin de Máster (SIANI, ULPGC).* Las Palmas de Gran Canaria : s.n., 2012.
28. **José Iván López González, José María Escobar Sánchez, Rafael Alejandro Montenegro Armas.** Modelización de sólidos para el análisis isogeométrico. *Trabajo Fin de Máster (SIANI, ULPGC).* Las Palmas de Gran Canaria : s.n., 2012.
29. **Carmen Dolores Fernández Melcón, Montserrat Piñeiro Barcia.** SUPERFICIES DE RESPUESTA. MÉTODOS Y DISEÑOS.
30. **Althaus, Gabriel Winter.** Introducción a los Algoritmos Evolutivos.
31. **Morcillo, Carlos González.** Introducción al dibujo de curvas de aproximación e interpolación por computador. *Splines: Curvas y Superficies.*
32. *Meshless methods: An overview and recent developments.* **T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, P. Krysl.** s.l. : ELSEVIER, 1996, Computer methods in applied mechanics and engineering, Vol. 139, págs. 3-47.