industriales
etsii

Escuela Técnica Superior de Ingeniería Industrial

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Escuela Técnica Superior de Ingeniería Industrial

Optimización topológica de estructuras de aeronaves utilizando computación paralela

TRABAJO FIN DE MÁSTER

MÁSTER UNIVERSITARIO EN INGENIERÍA INDUSTRIAL

Autor: **María Teresa Zapata García**

Director: David Herrero Pérez

Cartagena, octubre 2020

Universidad Politécnica de Cartagena

RESUMEN

La computación paralela es en la actualidad el paradigma dominante para realizar simulaciones complejas en mecánica de sólidos, ante las dificultades para continuar aumentando las prestaciones de la computación en serie y el auge de los procesadores multinúcleo. En este contexto, el presente proyecto pretende abordar mediante un enfoque de computación paralela un problema real: la optimización topológica de estructuras de aeronaves, como son un perfil aerodinámico y alas con diferentes configuraciones.

En este proyecto, se realiza un estudio de las principales arquitecturas y estándares existentes dentro de la computación paralela y se analizan las principales herramientas disponibles para la resolución del problema en paralelo. Se estudia también el método de optimización topológica basado en densidades, así como la forma de paralelizar el proceso de optimización topológica.

Se realizan modelos correspondientes a un perfil aerodinámico y tres alas con distintas configuraciones, pertenecientes a un avión comercial, un avión de combate y un avión ligero. Se lleva a cabo la optimización topológica para cada uno de estos modelos, utilizando el clúster de computación científica del grupo MC3 de la UPCT. Así, se consigue la distribución óptima de material para cada uno de los elementos modelados, que constituye un diseño conceptual del elemento y podrá servir como base para diseños futuros. Además, se consigue demostrar que la resolución en paralelo es la única herramienta que posibilita resolver los problemas de optimización planteados, que de otro modo resultarían inabordables debido a su gran tamaño y los elevados tiempos computacionales que requieren.

ABSTRACT

Parallel computing is currently the dominant paradigm for performing complex simulations in solid mechanics, mainly due to the difficulties in continuing to increase the performance of serial computing and the spread of multicore processors. In this context, the present project aims to use a parallel computing approach to address a real problem: the topological optimization of aircraft structures, such as an aerodynamic profile and wings with different configurations.

Within this project, a study of the main architectures and standards within parallel computing is carried out and the available parallel resolution tools are analysed. The topological optimization method based on densities is also studied, as well as how to parallelise the topological optimization process.

An aerodynamic profile and three wings with different configurations (belonging to a commercial aircraft, a fighter plane and a light aircraft) are modelled. Topological optimization is carried out for each of these models, using the scientific computing cluster of the MC3 group at the UPCT. Thus, the optimal distribution of material is achieved for each of the modelled elements, which constitutes a conceptual design of the element and may serve as a basis for future designs. Besides, it has been demonstrated that parallel resolution is the only tool that allows to solve the optimization problems presented, which would otherwise be unaffordable due to their large size and the high computing times they require.

Contenido

Índice de figuras

Índice de tablas

1. INTRODUCCIÓN

En la actualidad, las técnicas de computación de altas prestaciones (High Performance Computing, HPC) son el paradigma dominante para abordar simulaciones complejas en mecánica de sólidos, principalmente debido a que la tecnología disponible para incrementar las prestaciones de la computación en serie ha alcanzado unos límites que son difícilmente mejorables, o el esfuerzo para lograrlo lo hace inviable [1]. Para llegar a comprender este cambio de modelo, es necesario analizar la evolución de la arquitectura de los procesadores, así como la evolución en la manera de abordar los problemas de análisis y optimización en mecánica de sólidos.

En 1965, Gordon Moore, cofundador de Intel, predijo que el número de transistores de un microprocesador se doblaría cada 18 meses, lo que se conocería en adelante como la Ley de Moore. Este incremento del número de transistores va unido al consiguiente incremento de la velocidad de procesamiento. La Ley de Moore ha descrito con gran fidelidad el ritmo de mejora en el rendimiento de los procesadores desde su formulación. Aunque en numerosas ocasiones se ha pronosticado el fin de la vigencia de esta ley, la realidad es que sigue siendo válida, si bien en los últimos años se ha reducido el ritmo de mejora de los procesadores de un solo núcleo y los incrementos en el rendimiento se deben a la inclusión de múltiples núcleos dentro de un procesador.

Son tres los principales factores que explican la ralentización de las mejoras en el rendimiento de los procesadores de un solo núcleo. En primer lugar, existe una limitación de la frecuencia a la que pueden funcionar los procesadores, debido a que a altas frecuencias surgen problemas de consumo de potencia y disipación de calor. Este problema, conocido como “power wall”, ha supuesto que el aumento en la frecuencia de los procesadores se haya estancado, estabilizándose en torno a los 4 GHz [2]. Por otro lado, existe una disminución en los beneficios que se obtienen con la paralelización a nivel de instrucción ILP (Instruction Level Parallelism), ya que llega un momento que el solapamiento de las etapas en las que se divide una instrucción requiere una elevada cantidad de recursos para seguir obteniendo beneficios con este método. Esta limitación se conoce como “ILP wall”. Por último, ocurre que el ritmo de mejora en la velocidad de los procesadores excede al ritmo de mejora en la velocidad de la memoria, lo que se conoce como “memory wall” [3].

Estos factores han llevado a que los esfuerzos de los fabricantes ya no se concentren en fabricar procesadores más rápidos, sino en aumentar el rendimiento mediante la fabricación de procesadores multinúcleo [4]. La computación en paralelo, que permite dividir las tareas para que las realicen varios procesadores simultáneamente, cobra relevancia con la propagación del uso de procesadores multinúcleo.

En la Figura 1.1 se muestra la evolución a lo largo del tiempo en algunas de las características de los procesadores antes mencionadas, como son el número de transistores, el rendimiento con un solo hilo, la frecuencia, la potencia y el número de núcleos. Se observa el repunte en el número de núcleos cuando las demás características estabilizan su crecimiento.

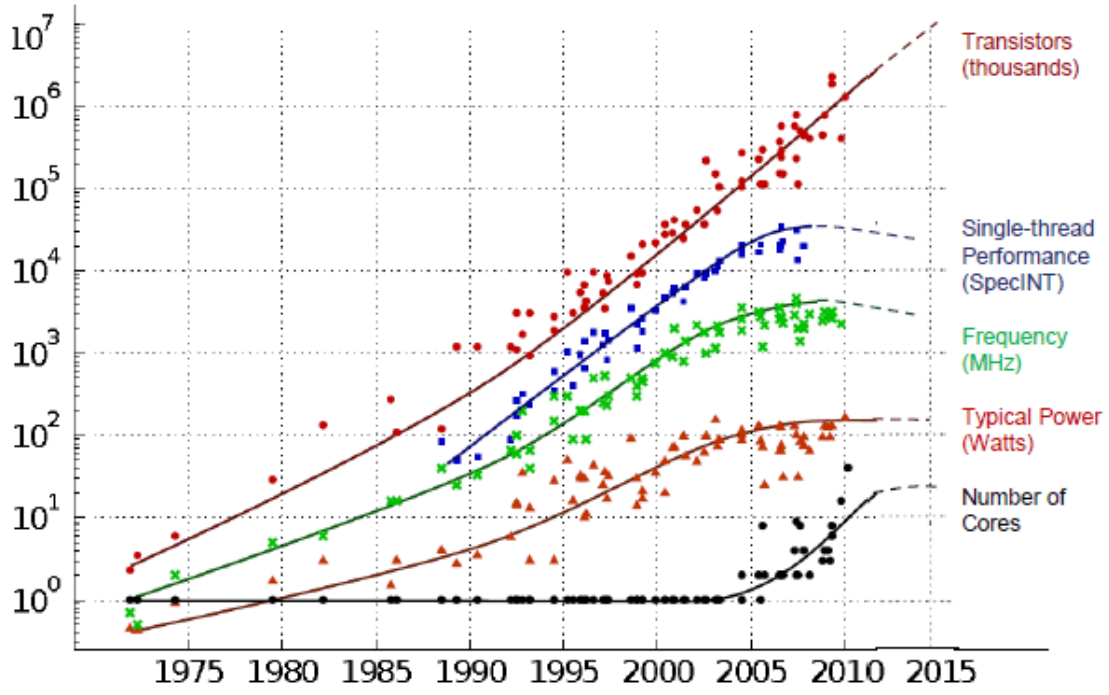

Figura 1.1. Evolución temporal de las características de los procesadores. Recuperado de [5].

No obstante, desde que Moore formuló su ley hasta la primera década de los años 2000, se produjo una extraordinaria mejora del rendimiento de los procesadores de un solo núcleo. Esto puede llevar a pensar que ya no deberían existir problemas de recursos computacionales a la hora de abordar problemas reales de ingeniería, como pueden ser el análisis o la optimización estructural, con un enfoque en serie.

Sin embargo, el crecimiento de la fiabilidad de los modelos (con su inherente complejidad) que se consideran adecuados para reproducir la realidad ha sido también exponencial. Son muchas las variables que intervienen en el hecho de que un análisis sea más complejo y tenga mayor coste computacional. En el caso de modelos de análisis con elementos finitos, la complejidad aumenta con el número de grados de libertad, el refinamiento de la malla, la complejidad de la topología, etc. Para ilustrar el crecimiento exponencial de la fiabilidad y complejidad de los modelos, se puede tomar el ejemplo del modelo de análisis por elementos finitos del ala de un avión presentado por Lecina y Petiau en 1987 [6], con 3500 grados de libertad. En 2001, el modelo presentado por Jegley et al. [7] para modelar el fallo último del ala de una aeronave empleaba 428 000 grados de libertad. Por lo expuesto anteriormente, los problemas de tiempos computacionales y consumos de memoria excesivos siguen estando plenamente vigentes [1].

Una vez vista la nueva tendencia a fabricar procesadores multinúcleo en lugar de aumentar la velocidad de cada núcleo y unida a la complejidad creciente de los problemas a abordar, se comprende plenamente el interés de emplear la computación paralela. Con esta se pretende aprovechar al máximo los recursos que brindan los nuevos procesadores multinúcleo para así poder reducir los tiempos computacionales o abordar problemas que con el modelo de computación en serie resultaban inabordables.

En este marco, el presente proyecto trata de aprovechar las técnicas de computación paralela y aplicarlas a un problema real de ingeniería. Se pretende realizar la optimización topológica de un

perfil aerodinámico y del ala de una aeronave minimizando la compliance como función objetivo, cuyo cálculo tiene un elevado coste computacional al requerir del método de los elementos finitos.

Los objetivos que se persiguen con la realización de este Trabajo de Fin de Máster son los siguientes:

1. Adquirir conocimientos sobre las técnicas de computación en paralelo y los diferentes modelos existentes.
2. Evaluar la aplicabilidad de las técnicas de computación en paralelo a problemas de mecánica de sólidos y las herramientas disponibles.
3. Estudiar el método de optimización topológica basado en densidades y su idoneidad para el problema de estudio.
4. Realizar modelos 2D y 3D de estructuras de aeronaves.
5. Realizar una optimización topológica de los elementos modelados.
6. Evaluar parámetros computacionales y realizar comparativas para simulaciones con diferente número de núcleos.

Esta memoria se ha estructurado en seis capítulos. En el segundo de estos se expondrán los principios y los principales modelos de computación paralela. En el tercer capítulo, se estudia cómo aplicar la computación en paralelo a las simulaciones en mecánica de sólidos, abordando el problema de elasticidad lineal y los métodos de resolución. A continuación, se estudiará el problema de la optimización topológica, centrándose en el método de optimización SIMP (Solid Isotropic Material with Penalization). En el quinto capítulo se expondrán los modelos utilizados y los resultados experimentales de la optimización topológica de las estructuras de una aeronave, obtenidos con el clúster de computación científica del grupo MC3 de la UPCT. Además de la optimización topológica, se pretenden obtener también comparativas en los tiempos de computación y la aceleración al ejecutar las simulaciones con distinto número de procesadores. En el último capítulo se exponen las conclusiones de este estudio.

2. COMPUTACIÓN EN PARALELO

2.1. Introducción

En los últimos años, la imposibilidad de continuar el ritmo de mejora en el rendimiento de los procesadores ha llevado a los fabricantes a incrementar el número de núcleos de cada microprocesador para mejorar sus prestaciones. En esta nueva situación se ha producido el auge de las herramientas de procesamiento en paralelo, que aprovechan el diseño de los nuevos procesadores para optimizar los recursos computacionales en la resolución de problemas.

En este capítulo se va a realizar una introducción a la computación en paralelo, exponiendo las principales arquitecturas y estándares existentes. Frente a la computación en serie que se ha empleado tradicionalmente, la computación en paralelo permite la utilización simultánea de múltiples recursos de computación para resolver un problema.

Las computadoras paralelas pueden clasificarse según el nivel de paralelismo que admite su hardware. Por un lado, se tienen los equipos con procesadores multinúcleo y multiprocesador, que tienen múltiples elementos de procesamiento dentro de una sola máquina, y por otro lado los clústeres de computación y grids, que utilizan varios equipos para trabajar en una misma tarea. En el desarrollo del presente trabajo se va a utilizar el clúster de computación del grupo MC3 de la UPCT.

Un clúster de computación está formado por una serie de ordenadores interconectados por una red local de alta velocidad que actúan de forma conjunta para realizar una tarea. Es importante señalar su gran escalabilidad, ya que es sencillo añadir nuevos nodos con distinto número de procesadores y memoria, y su flexibilidad, que permite que cada uno de los ordenadores que componen el clúster tenga diferente sistema operativo.

Para explotar la paralelización existen diferentes enfoques, que se suelen clasificar jerárquicamente en:

- Paralelización a nivel de instrucción (en inglés, 'Instruction Level Parallelism' o 'ILP'): Con este enfoque se pretende mejorar el rendimiento mediante el solapamiento de las etapas en las que se divide una instrucción en un procesador o unidad funcional. Hay dos principales planteamientos de la paralelización a nivel de instrucción, el llamado 'pipelining'(en el cual se ejecutan simultáneamente diversas etapas que componen las instrucciones) y 'multiple issue' (en el que se inician varias instrucciones específicas simultáneamente (como ALUs) permitidas por el propio diseño de la arquitectura paralela del procesador) [8].
- Paralelización a nivel de datos ('data parallelism'): En este enfoque los datos se distribuyen entre los diferentes nodos de computación paralela, que realizan simultáneamente una misma tarea sobre los datos que les han sido asignados.
- Paralelización a nivel de tareas (en inglés, 'task parallelism', 'function parallelism' o 'control parallelism'): El problema a resolver se divide en partes más pequeñas, cuyo código se distribuye entre los diferentes procesadores, que resuelven estas partes en paralelo.

  La paralelización a nivel de tareas se caracteriza por realizar simultáneamente distintas tareas sobre datos que pueden ser los mismos o diferentes, es decir, dos procesadores diferentes pueden trabajar a la vez con los mismos datos [2].

La paralelización a nivel de tareas ofrece una mayor flexibilidad, mientras que con la paralelización a nivel de datos se alcanzan mayores aceleraciones, gracias a la utilización de hardware específico de tipo SIMD (Single Instruction Multiple Data), como GPUs o unidades de procesamiento específico.

El enfoque que se va a utilizar para las simulaciones en el presente trabajo es la paralelización a nivel de tareas.

Dentro de la computación en paralelo, utilizando paralelización a nivel de tareas, existen tres principales arquitecturas según la utilización de la memoria: memoria compartida, memoria distribuida y arquitectura híbrida entre memoria compartida y distribuida.

En el modelo de memoria compartida (en inglés Shared Memory Model, SMM) múltiples procesadores pueden operar de forma independiente, realizando cada una de las partes en las que ha sido dividida la tarea y compartiendo los mismos recursos de memoria. De este modo, los cambios realizados por uno de los procesadores en una dirección de la memoria son visibles por el resto de los procesadores. En la Figura 2.1 se muestra un esquema de esta arquitectura [9].

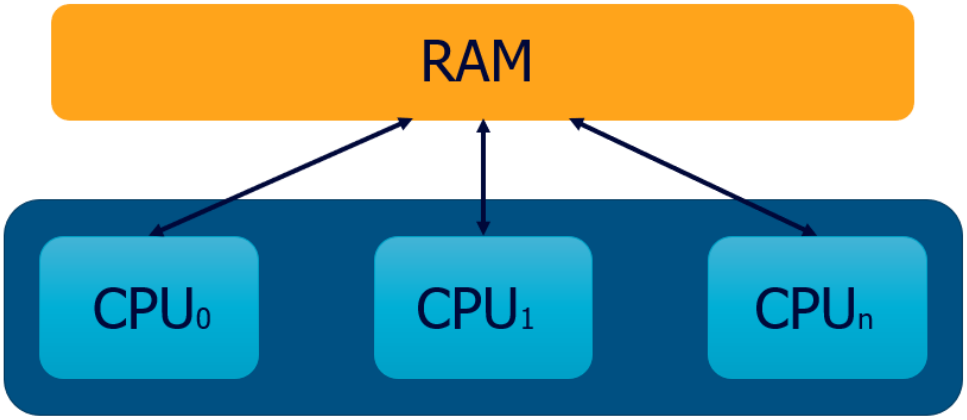

Figura 2.1. Arquitectura de memoria compartida

Se han realizado diversos esfuerzos para estandarizar el modelo de memoria compartida, entre los cuales destaca OpenMP. OpenMP (Open Multi-Processing) fue creado por un grupo de fabricantes, en la segunda mitad de los años 90 y la primera versión se publicó en 1997. El último estándar publicado en noviembre de 2018 es el OpenMP 5.0. Actualmente la organización sin ánimo de lucro OpenMP Architecture Review Board (OpenMP ARB), formada por grandes fabricantes de hardware y software (como AMD, Intel, Nvidia, Texas Instruments, etc.), se encarga del mantenimiento y el lanzamiento de nuevas versiones de OpenMP.

OpenMP es una especificación que define una serie de directivas de compilación, bibliotecas y variables de entorno que tienen por objetivo implementar sistemas de computación en memoria compartida. Las implementaciones de OpenMP se basan en estas especificaciones y se realizan para los lenguajes de programación Fortran y C/C++ y distintos sistemas operativos (Windows, Linux, MacOS, etc.). Además, admiten distintos compiladores como son GCC, IBM XL, Visual C++ y Intel Parallel Studio, entre otros [10].

La estructura OpenMP se basa en el modelo de hilos. Como se muestra en la Figura 2.2, en estos programas se puede distinguir entre regiones secuenciales y regiones paralelas. En las regiones secuenciales se ejecuta un único hilo (maestro) mientras que, al llegar a la región paralela, el programa se subdivide en un número determinado de tareas (hilos) que son ejecutados de forma paralela hasta que vuelven a concurrir. Cada hilo tiene sus datos locales, pero también se beneficia de la vista de la memoria global a través de la cual se comunican todos los hilos [11].

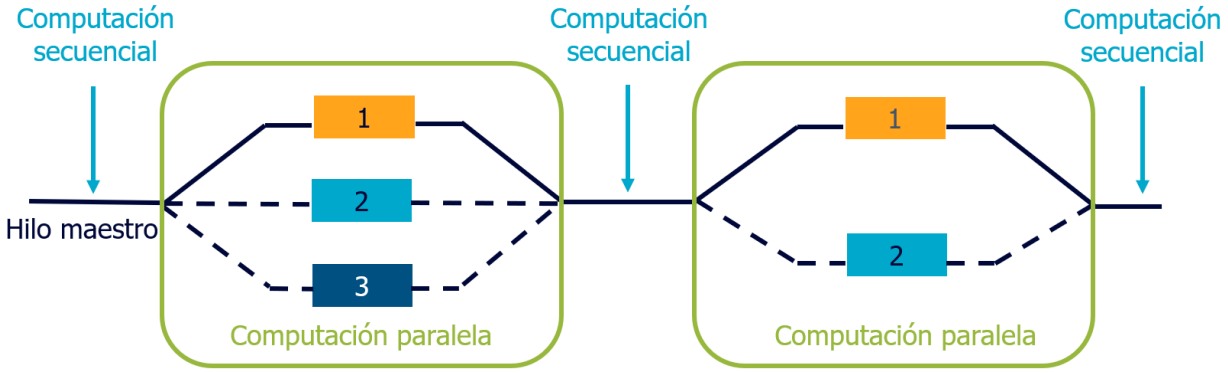

Figura 2.2. Programación en hilos

El estándar OpenMP, y en general la computación paralela con memoria compartida, permite una programación simple y un fácil acceso de todos los procesadores a los datos almacenados en la memoria global. Además, el hecho de que todos los procesadores compartan la misma memoria (ver Figura 2.1), hace que el consumo de memoria no se vea significativamente incrementado respecto a una implementación en serie. No obstante, sus principales desventajas son su baja escalabilidad y que el acceso a la memoria compartida requiere de mecanismos de sincronización entre hilos, lo que provoca periodos de inactividad (que afectan seriamente al rendimiento) en los procesos como consecuencia de estos mecanismos de sincronización.

El segundo tipo de arquitectura para la computación en paralelo es el de memoria distribuida. En este apartado se exponen brevemente los fundamentos de este modelo, que se describirá con más detalle en el siguiente apartado, ya que es el que se va a utilizar para las simulaciones realizadas dentro de este trabajo.

En el modelo de memoria distribuida (también llamado modelo de paso de mensajes) cada nodo consta de un procesador con su propia memoria local, que opera independientemente del resto [9]. Los nodos se conectan entre sí mediante una red de comunicación, como se observa en la Figura 2.3.

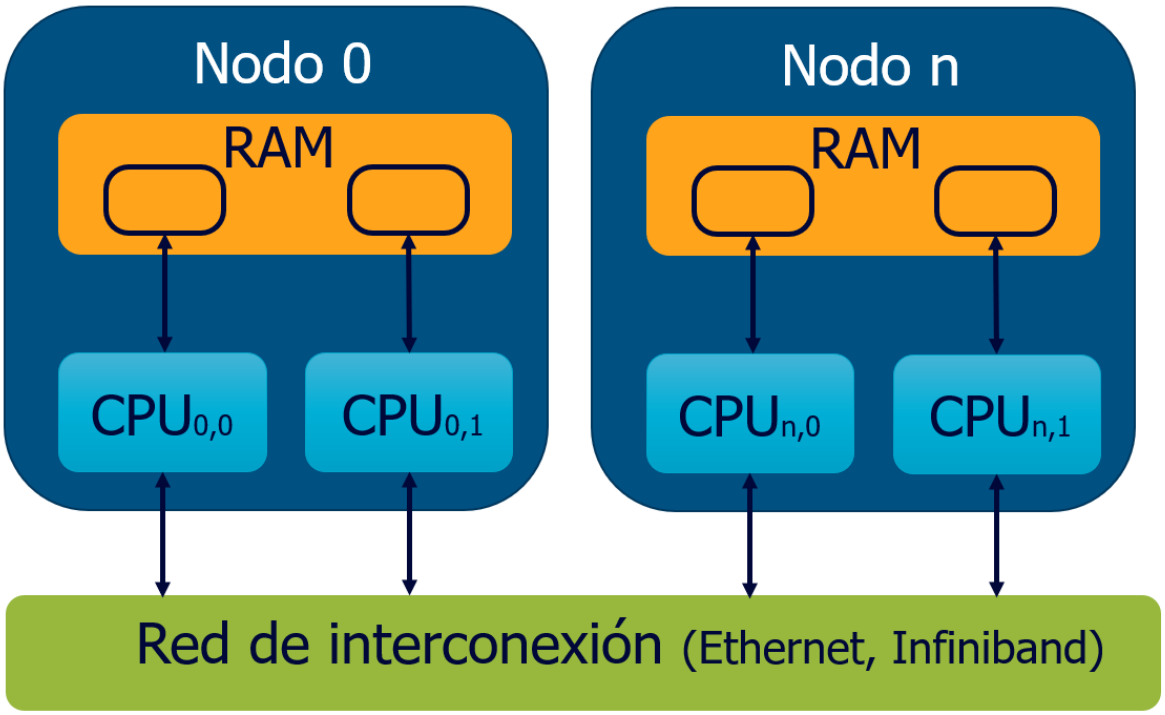

Figura 2.3. Arquitectura hardware de memoria distribuida

La principal ventaja de la computación paralela con memoria distribuida es su escalabilidad, aunque presenta una mayor complejidad en la programación y las tareas de comunicación entre los nodos pueden suponer una pérdida en el rendimiento del sistema, debido al tiempo requerido para intercambiar datos a través de la red de interconexión. No obstante, las aceleraciones que se consiguen utilizando memoria distribuida suelen ser mayores a las que se logran utilizando memoria compartida, puesto que los procesadores no tienen que compartir recursos de memoria y no son necesarias las operaciones de sincronización entre hilos. Como desventaja, cabe mencionar que la cantidad de memoria necesaria suele ser bastante superior al enfoque utilizando memoria compartida.

Actualmente, MPI (Message Passing Interface) es el estándar de paso de mensajes más extendido. Existen diversas implementaciones de este estándar, como OpenMPI y MPICH.

Por último, existen arquitecturas híbridas que combinan los modelos de memoria compartida y distribuida, como se recoge en la Figura 2.4. Dentro de un mismo nodo los procesadores comparten la memoria, pero entre procesadores se utiliza el modelo de paso de mensajes. Así se pretende aprovechar al máximo las ventajas de ambos modelos [9].

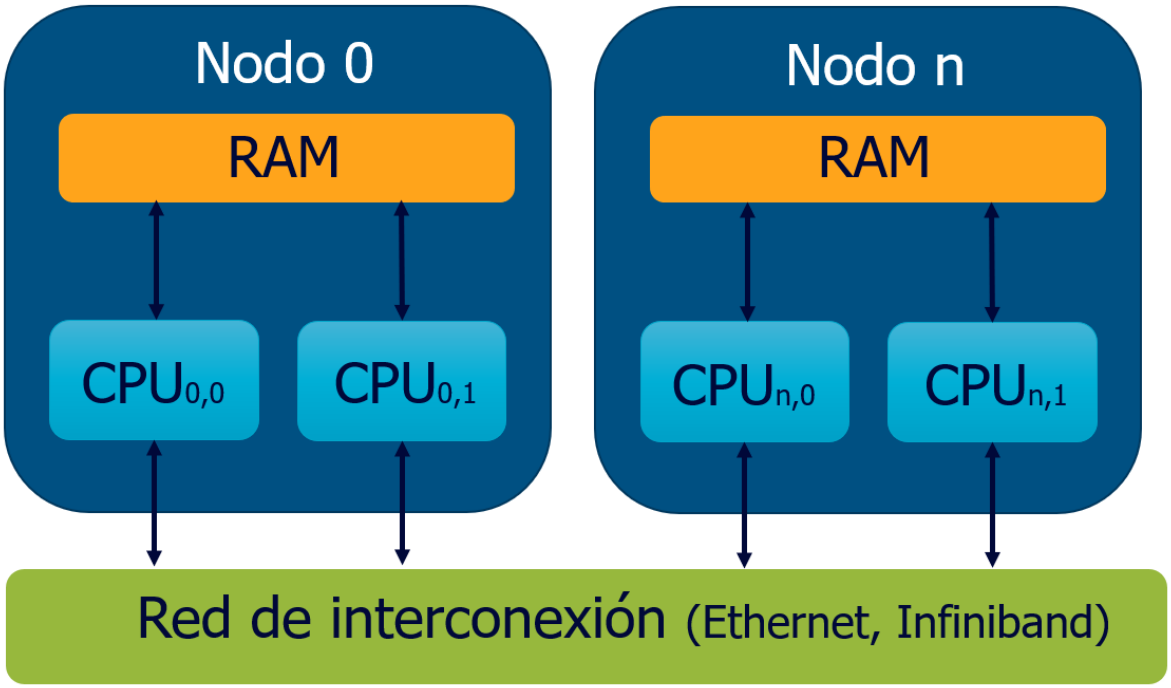

Figura 2.4. Arquitectura híbrida

2.2. MPI

MPI (Message Passing Interface) es el estándar más extendido para el modelo de paso de mensajes. La primera versión (MPI-1) fue creada en el MPI Forum en 1994 con unas 40 organizaciones participantes. Posteriormente aparecieron las versiones MPI-2 en 1997 y MPI-3 en 2012. La última especificación MPI-4 data de 2019 y se estima que en 2020 esté finalizada [12].

MPI es una especificación para programación de paso de mensajes, que proporciona una librería de funciones para C, C++ o Fortran que son empleadas en los programas para comunicar datos entre procesos [13].

La especificación MPI proporciona un sistema de intercambio de mensajes estándar y portable, que permite desarrollar programas para computación en paralelo con arquitectura de memoria distribuida.

Entre las principales características de MPI se encuentran las siguientes [14]:

- Estandarización.
- Portabilidad: multiprocesadores, clúster de computación, etc.
- Amplia funcionalidad.
- Existencia de implementaciones libres (MPICH, OpenMPI, LAM-MPI…)

Cualquier programa MPI puede implementarse con solo 6 funciones, aunque la librería incluye muchas más funciones para aspectos avanzados. Todas estas funciones comienzan por “MPI_” y permiten:

- Iniciar, gestionar y finalizar procesos MPI.
- Comunicar datos entre dos procesos.

- Realizar operaciones de comunicación entre grupos de procesos.
- Crear tipos arbitrarios de datos.

La estructura de un programa MPI en C/C++, que se muestra en la Figura 2.5, incluye [9]:

- Un encabezado donde de añade la biblioteca mpi.h (con #include “mpi.h”).
- Declaración de variables.
- Una inicialización del ambiente en paralelo.
- El desarrollo principal del código, donde se incluyen las llamadas de comunicación MPI (llamadas de comunicación entre dos procesadores o colectivas).
- Cierre de las comunicaciones MPI.

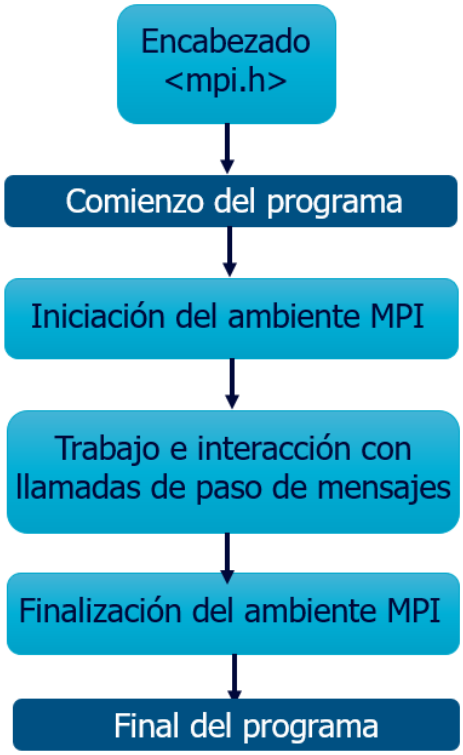

Figura 2.5. Estructura de programación MPI

El estándar MPI tiene múltiples implementaciones, entre las que destacan OpenMPI y MPICH.

OpenMPI es una especificación de código libre, que surge como una evolución de las especificaciones anteriores (LAM/MPI, FT-MPI, etc.) e implementa completamente los estándares MPI-1 y MPI-2. Proporciona una solución de alto rendimiento para gran variedad de dispositivos.

MPICH es otra implementación de MPI de código abierto, desarrollada en el año 2001. Se trata de un estándar portable cuyo uso está muy extendido en arquitecturas como los clústeres de computación. Esta es la implementación que se va a usar en el desarrollo del presente proyecto.

3. COMPUTACIÓN EN PARALELO DE LAS SIMULACIONES EN MECÁNICA DE SÓLIDOS

3.1. Introducción

En este capítulo se exponen las diferentes herramientas utilizadas para abordar en paralelo un problema de mecánica de sólidos.

En primer lugar, se exponen los fundamentos del problema de la elasticidad lineal, que en el presente proyecto se aborda mediante el Método de los Elementos Finitos. A continuación, se trata la necesidad de particionar el problema para asignar recursos computacionales a cada una de las partes. Esto se hace mediante el sistema de particionamiento de grafos jerárquico implementado en el software METIS.

Finalmente se aborda el sistema de resolución en paralelo. Para que este sea eficiente, es necesario almacenar de forma distribuida la matriz de coeficientes, los vectores y las operaciones que se realicen con estos. Se incide en el solver utilizado para las simulaciones: un gradiente conjugado distribuido (PCG) precondicionado con un multigrid algebraico clásico (AMG clásico de Ruge-Stuben [15]). Se detallan los fundamentos de estos métodos de resolución y de la librería Hypre, utilizada para implementar el solver y el precondicionador.

3.2. Problema de elasticidad lineal

El problema elástico tiene por objetivo encontrar los desplazamientos y las tensiones en un sólido deformable elástico, partiendo de la forma original del sólido, las fuerzas exteriores que actúan sobre el mismo y los desplazamientos impuestos en algunos puntos de su superficie. El problema elástico lineal es un tipo particular de problema elástico, en el que se asumen una serie de hipótesis simplificadoras. Para que el problema sea considerado de elasticidad lineal se asume que los desplazamientos y las deformaciones que se producen en el sólido son pequeños y que existe linealidad en la ley de comportamiento del material (relación entre tensiones y deformaciones) [16].

En este apartado se introduce brevemente el problema de la elasticidad lineal para centrarnos en los siguientes apartados en la forma de paralelizar este problema.

Para obtener los desplazamientos y las tensiones en el sólido, el problema de elasticidad lineal se sirve de las ecuaciones de equilibrio, las ecuaciones de compatibilidad y la ley de comportamiento del material, a las que se añaden las condiciones de contorno.

Las ecuaciones de equilibrio externo relacionan las fuerzas exteriores que actúan sobre el material con las componentes del tensor de tensiones. Estas ecuaciones son:

$$F_x + \left(\frac{\partial \sigma_x}{\partial x}\right) + \left(\frac{\partial \tau_{xy}}{\partial y}\right) + \left(\frac{\partial \tau_{xz}}{\partial z}\right) = 0 \quad (3.\ 1)$$

$$F_y + \left(\frac{\partial \tau_{xy}}{\partial x}\right) + \left(\frac{\partial \sigma_y}{\partial y}\right) + \left(\frac{\partial \tau_{yz}}{\partial z}\right) = 0 \quad (3.\ 2)$$

$$F_z + \left(\frac{\partial \tau_{xz}}{\partial x}\right) + \left(\frac{\partial \tau_{yz}}{\partial y}\right) + \left(\frac{\partial \sigma_z}{\partial z}\right) = 0 \quad (3.\ 3)$$

Siendo F_x, F_x y F_x las componentes de la fuerza de volumen actuante sobre el sólido (conocidas) y σ_x, σ_x, σ_x, τ_{xy}, τ_{xz}, τ_{yz} las componentes del tensor de tensiones simétrico.

El segundo grupo de ecuaciones necesarias para la resolución del problema son las ecuaciones de comportamiento o constitutivas, que relacionan las tensiones en el material con las deformaciones que sufre y se presentan a continuación:

$$\sigma_x = \lambda(\varepsilon_x + \varepsilon_y + \varepsilon_z) + 2G\varepsilon_x \tag{3. 4}$$

$$\sigma_y = \lambda(\varepsilon_x + \varepsilon_y + \varepsilon_z) + 2G\varepsilon_y \tag{3. 5}$$

$$\sigma_z = \lambda(\varepsilon_x + \varepsilon_y + \varepsilon_z) + 2G\varepsilon_z \tag{3. 6}$$

$$\tau_{xy} = G\,\gamma_{xy} \tag{3. 7}$$

$$\tau_{xz} = G\,\gamma_{xz} \tag{3. 8}$$

$$\tau_{yz} = G\,\gamma_{yz} \tag{3. 9}$$

Donde ε_i representa la deformación lineal que se produce en una determinada dirección, G es el módulo de elasticidad transversal y $\gamma_{ij} = 2\varepsilon_{ij}$ es la deformación angular. Las ecuaciones (3.4) a (3.9) se conocen como ecuaciones de Lamé. Alternativamente a estas, se pueden encontrar las ecuaciones de comportamiento mediante expresiones que dan las deformaciones en función de las tensiones, lo que se conoce como Ley de Hooke. Además, tanto en las ecuaciones de Lamé como en las de Hooke, se puede incluir un término que tiene en cuenta las deformaciones provocadas por los cambios de temperatura en el material.

Para completar el planteamiento del problema de elasticidad lineal se exponen las ecuaciones de compatibilidad, que relacionan las deformaciones que se producen en el sólido con los desplazamientos (cuyas tres componentes son u_x, u_y y u_z):

$$\varepsilon_x = \frac{\partial u_x}{\partial x} \tag{3. 10}$$

$$\varepsilon_y = \frac{\partial u_y}{\partial y} \tag{3. 11}$$

$$\varepsilon_z = \frac{\partial u_z}{\partial z} \tag{3. 12}$$

$$\varepsilon_{xy} = \frac{1}{2}\left(\frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x}\right) \tag{3. 13}$$

$$\varepsilon_{xz} = \frac{1}{2}\left(\frac{\partial u_x}{\partial z} + \frac{\partial u_z}{\partial x}\right) \tag{3. 14}$$

$$\varepsilon_{yz} = \frac{1}{2}\left(\frac{\partial u_y}{\partial z} + \frac{\partial u_z}{\partial y}\right) \tag{3. 15}$$

De este modo se tienen 15 ecuaciones (tres ecuaciones de equilibrio, seis de comportamiento y seis de compatibilidad) que relacionan las variables del problema como se muestra en la Figura 3.1 y

permiten resolver las 15 incógnitas del problema (las seis componentes del tensor de tensiones, las tres componentes del vector de desplazamientos y las seis componentes del tensor de deformaciones).

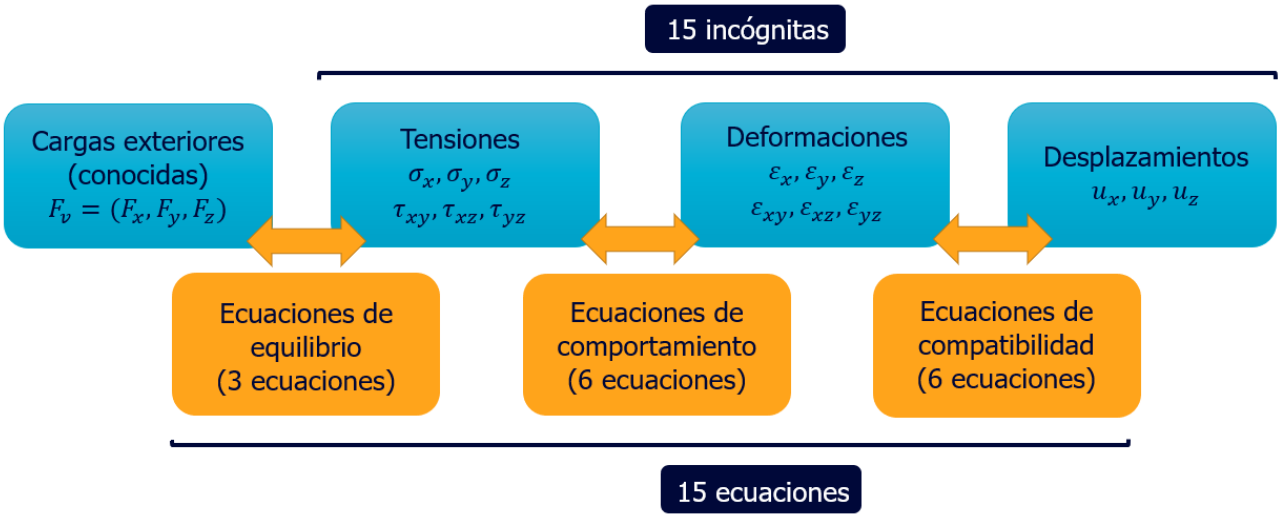

Figura 3.1. Planteamiento del problema de elasticidad lineal

La complejidad del problema de elasticidad lineal radica en la dificultad para resolver las ecuaciones en derivadas parciales (EDPs) de forma analítica, por lo que se suelen utilizar métodos aproximados. Uno de los métodos aproximados más extendidos, y el método que se utilizará en este trabajo, es el Método de los Elementos Finitos (MEF).

El Método de los Elementos Finitos se basa en discretizar el sólido mediante una malla, que conecta un determinado número de nodos en los que se calcularán las incógnitas del problema. El MEF convierte el sistema de ecuaciones en derivadas parciales en un problema matricial de la siguiente forma:

$$K\,u = F \tag{3. 16}$$

siendo “u” el vector de desplazamientos del sólido, “F” el vector de cargas y “K” la matriz de rigidez.

Tras obtener los valores de las incógnitas en los nodos, se obtienen los valores en el resto del dominio mediante funciones de interpolación.

3.3. Particionamiento jerárquico de grafos utilizando METIS

Para paralelizar el problema de elasticidad lineal, que se resolverá mediante el método de los elementos finitos, es necesario particionar los dominios para poder asignar recursos computacionales a cada uno de ellos. Con este fin, en el presente proyecto se utilizará el software METIS, que implementa algoritmos de particionamiento jerárquico de grafos y permite dividir las mallas de elementos finitos y distribuirlas entre los distintos procesadores de la computadora paralela.

METIS es un software desarrollado por la Universidad de Minnesota, que se puede descargar libremente y que además está incluido en muchas distribuciones de sistemas operativos basados en Unix (como Linux y FreeBSD). METIS se basa en el paradigma del particionamiento de grafos multinivel, lo que le permite realizar de manera rápida particiones de grafos irregulares, mallas no estructuradas y matrices dispersas con alta calidad y reduciendo el espacio de almacenamiento requerido.

METIS proporciona una serie de programas independientes que realizan la partición de grafos, la partición de mallas, reducen el espacio de almacenamiento y convierten una malla en grafo. Además, también proporciona una interfaz de programación de aplicaciones (API) que puede usarse para implementar las citadas funcionalidades en nuestros propios programas en C/C++ o Fortran.

El software METIS utiliza el método de partición de grafos multinivel. Este método, cuyo esquema explicativo se recoge en la Figura 3.2, consta de tres fases: la reducción del grafo, la partición inicial y la proyección. En la fase de reducción del grafo, se parte del grafo inicial (G0) y se van obteniendo sucesivamente grafos más pequeños. Esto se consigue anexionando vértices adyacentes en el grafo anterior. El proceso continúa hasta que el grafo queda reducido a unos cientos de vértices.

Posteriormente, en la fase de partición inicial, el grafo más pequeño al que se ha llegado (G3) se divide usando algoritmos como el desarrollado por Kernighan-Lin. Este proceso es muy rápido ya que se realiza sobre un grafo muy reducido.

Finalmente, en la fase de proyección el grafo sobre el que se ha realizado la partición se proyecta hasta que alcanza el tamaño del grafo inicial (G0). Se desagrupan los vértices que fueron anexionados en la fase de reducción y se refina la partición tras cada iteración.

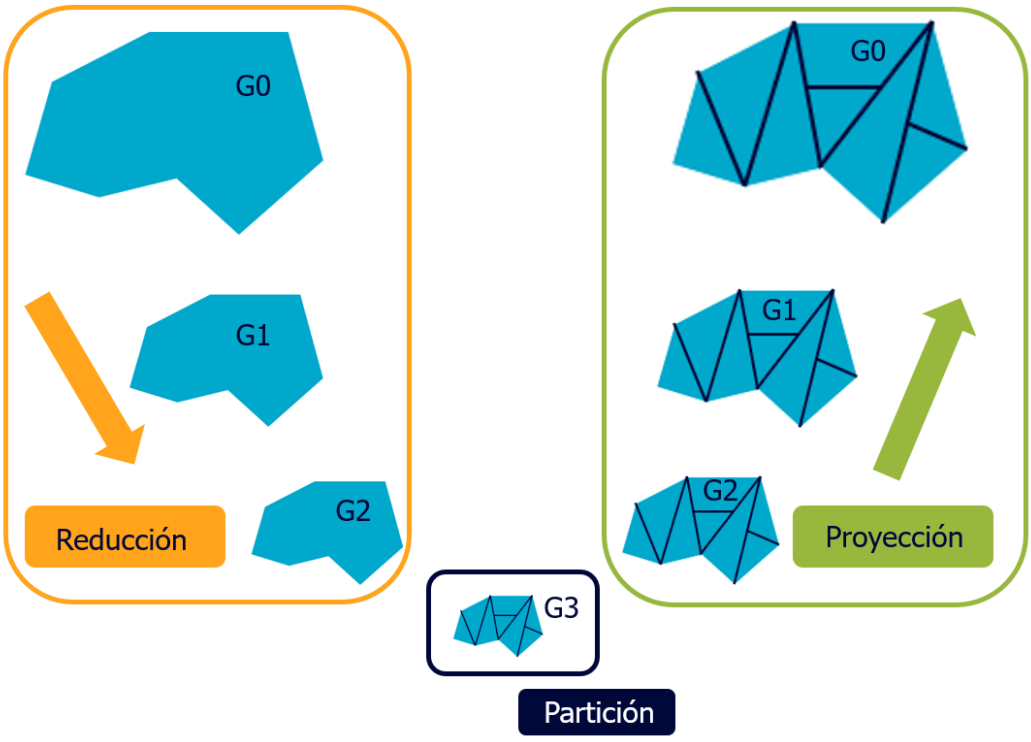

Figura 3.2. Fases de la partición de grafos multinivel.

Como el dato de entrada es la malla de elementos finitos de un determinado problema de elasticidad lineal, METIS proporciona rutinas para transformar esta malla en un grafo (de "n" vértices). Tras realizarse el proceso de partición multinivel, se obtiene como dato de salida un vector de "n" filas. La fila i-ésima contiene el número de la partición a la que pertenece el vértice número "i" del grafo.

El objetivo de este proceso de particionamiento es doble. Por un lado, se pretende repartir equilibradamente las tareas a realizar en un determinado programa entre los distintos procesadores. En este sentido, el software METIS permite tener en cuenta distintas variables como el coste computacional y las necesidades de memoria a la hora de hacer las particiones y distribuirlas entre los procesadores, del mismo modo que asegura que la carga de trabajo de los procesadores esté distribuida de forma equitativa durante todas las fases del programa.

Por otro lado, se pretende minimizar las comunicaciones entre procesadores que se producen cuando un mismo elemento de la malla es asignado a distintos procesadores. Este último objetivo se puede conseguir minimizando el número de fronteras de los dominios que pertenecen a más de una partición (lo que se conoce como "edge-cut") o minimizando el volumen total de las comunicaciones [17].

3.4. Sistema de resolución en paralelo

En este apartado se presenta el método empleado para resolver en paralelo las ecuaciones lineales del problema de elasticidad lineal aproximado mediante el método de elementos finitos.

El solver que se va a utilizar para las simulaciones de este proyecto es un gradiente conjugado distribuido (PCG) precondicionado con un multigrid algebraico clásico (AMG clásico de Ruge-Stuben). En concreto, se va a utilizar la implementación de la librería Hypre.

La librería Hypre proporciona precondicionadores de alto rendimiento y solvers para la resolución de grandes sistemas dispersos de ecuaciones lineales utilizando computación paralela. Hypre contiene diversos tipos de algoritmos precondicionadores escalables, centrados en la resolución de grandes sistemas dispersos de ecuaciones lineales, y además permite implementar los métodos iterativos más comunes (como el método de Krylov y el del gradiente conjugado). La combinación de estos métodos iterativos con los precondicionadores permite obtener soluciones de alta eficiencia. La librería Hypre está escrita en C (excepto la interfaz para elementos finitos que está escrita en C++) y también proporciona una interfaz para usuarios de Fortran [18].

Para facilitar su uso, Hypre proporciona diferentes interfaces (llamadas "conceptual interfaces") según el tipo de problema a abordar. Estas interfaces proporcionan ayuda al usuario para traducir los datos de un determinado tipo de problema y construir la matriz de datos requerida por los solvers de la librería Hypre. Por este motivo, la correcta elección de la interfaz para cada problema es esencial. Las cuatro interfaces que proporciona Hypre son:

- Interfaz estructurada (Struct): Esta interfaz es la apropiada para problemas con patrones de computación (stencil-based computation) en los que los cálculos dependen del patrón que se utilice, como una malla regular.
- Interfaz semiestructurada (SStruct): Es la adecuada para aplicaciones en las que las operaciones dependen mayoritariamente de patrones de computación (stencil-based computation). Las redes son mayoritariamente estructuradas, pero presentan algunas partes que no dependen de dicho patrón.

- Interfaz basada en elementos finitos (FEI): Esta interfaz está pensada para su uso en problemas de elementos finitos con mallas estructuradas o no estructuradas. Es la única interfaz escrita en C++.
- Interfaz lineal algebraica (IJ): Se suele utilizar cuando ninguna de las tres interfaces anteriores es conveniente para un determinado problema. Requiere más esfuerzo por parte del usuario porque no proporciona funcionalidades para realizar el ensamblaje de las matrices de coeficientes y vectores necesarios en la definición del problema, ya que esta interfaz solamente proporciona solvers y precondicionadores de tipo más general que las anteriores.

La Tabla 3.1 muestra los diferentes solvers y precondicionadores disponibles para cada interfaz de Hypre, resaltando la interfaz y las herramientas que se utilizarán en este proyecto.

Solvers	System Interfaces			
	Struct	SStruct	FEI	IJ
Jacobi	X	X		
SMG	X	X		
PFMG	X	X		
Split		X		
SysPFMG		X		
FAC		X		
Maxwell		X		
BoomerAMG		X	X	X
AMS		X	X	X
ADS		X	X	X
MLI		X	X	X
ParaSails		X	X	X
Euclid		X	X	X
PILUT		X	X	X
PCG	X	X	X	X
GMRES	X	X	X	X
FlexGMRES	X	X	X	X
LGMRES	X	X		X
BiCGSTAB	X	X	X	X
Hybrid	X	X	X	X
LOBPCG	X	X		X

Tabla 3.1. Solvers en Hypre según la interfaz. Recuperado de [18].

Una vez que se tiene una visión general de la librería Hypre, en los siguientes apartados se expone en detalle el método del gradiente conjugado, utilizado como solver, y el multigrid algebraico que se usa como precondicionador. Se detalla también el formato de datos utilizado para almacenar la matriz de coeficientes de forma distribuida.

3.4.1. Gradiente conjugado distribuido

El método del gradiente conjugado es, junto con sus variantes, el método iterativo más utilizado para resolver grandes sistemas lineales de ecuaciones cuya matriz de coeficientes es simétrica y semidefinida positiva. El método del gradiente conjugado permite resolver sistemas de la forma

$$Ax = b \tag{3. 17}$$

donde "x" es un vector desconocido, "b" es un vector conocido y "A" es una matriz cuadrada simétrica definida positiva. Este sistema tiene la misma forma que el proporcionado por el método

de los elementos finitos (Ecuación 3.16) por lo que, particularizando para nuestro problema, el vector "x" corresponderá con el vector de desplazamientos "u", "A" con la matriz de rigidez "K" y el vector conocido "b" con el vector de fuerzas exteriores "F" [19].

El método del gradiente conjugado es un caso particular del método del gradiente descendente. Por este motivo, se van a exponer en primer lugar los fundamentos del método gradiente descendente, que serán de gran utilidad para comprender el algoritmo del método del gradiente conjugado.

Los métodos de gradiente descendente fueron desarrollados para minimizar una función cuadrática de la forma

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c \tag{3. 18}$$

Donde "A" es una matriz simétrica de dimensiones n x n, "x" y "b" son vectores y "c" es un escalar. En la Figura 3.3 se representa la forma de la función f (x) para el caso en el que n=2 (aunque podría tratarse de un problema de dimensiones mayores).

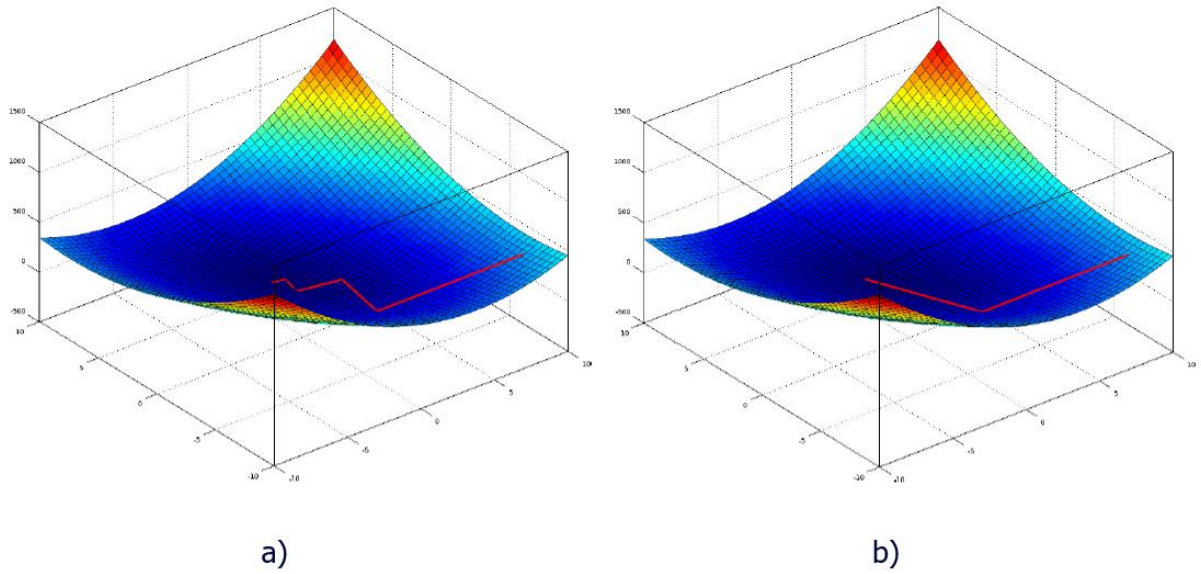

Figura 3.3. Representación de la función f(x). a) Recorrido del método del gradiente descendente para alcanzar el mínimo de f(x). b) Recorrido del método del gradiente conjugado para alcanzar el mínimo de f(x) Recuperado de [20].

Para alcanzar el mínimo de f(x) partiendo de un punto cualquiera x_0, se calcula la dirección en la cual f(x) decrece más rápido. Esta dirección corresponde a la inversa del gradiente (que es un campo vectorial con la propiedad de que, en un punto dado, el vector de campo correspondiente apunta en la dirección del máximo crecimiento de f). Haciendo el gradiente de la expresión (3.18):

$$\nabla f(x) = \frac{1}{2}A^T x + \frac{1}{2}\,Ax - b \tag{3. 19}$$

Como la matriz "A" es simétrica, la expresión (3.19) queda:

$$\nabla f(x) = Ax - b \tag{3. 20}$$

Igualando esta expresión a cero se recupera el sistema de la expresión (3.17). De este modo, la solución del sistema $Ax = b$ es un mínimo de f (lo cual se puede demostrar fácilmente siendo la

matriz A simétrica y definida positiva). Todo lo anterior implica que la solución de $Ax = b$ se puede calcular buscando el punto "x" que minimice f(x) [21].

De este modo, con el método del gradiente descendente se comienza a iterar en un punto arbitrario y siguiendo la línea de máximo descenso del paraboloide (que es la dirección contraria al gradiente) se obtiene una sucesión de puntos x_1, x_2, …, x_n hasta que se llega a un punto x_n lo suficientemente cercano a la solución.

Se define el residuo de cada iteración como:

$$r_i = b - A\,x_i = -\nabla f(x_i) \tag{3. 21}$$

En cada iteración se avanza en la dirección del gradiente, que según la expresión (3.21) se corresponde con el residuo y se obtiene un nuevo valor del vector x:

$$x_{i+1} = x_i + \alpha_i\, r_i \tag{3. 22}$$

Donde α_i, llamada tasa de aprendizaje determina cuánto se avanzará en la dirección de descenso:

$$\alpha_i = \frac{(r_i)^T\, r_i}{(r_i)^T\, A\, r_i} \tag{3. 23}$$

Este proceso se repite de forma iterativa hasta que el valor del residuo queda por debajo de una tolerancia previamente establecida (ε) o se alcanza un número de iteraciones n_{max}:

Algoritmo del Método del Gradiente Descendente
Input $A, x_0, b, \varepsilon, n_{max}$
$i \leftarrow 0\,; tol \leftarrow 2\varepsilon$
While $i < n_{max}\ y\ tol > \varepsilon$ **do**
$r_i \leftarrow b - A\,x_i$
$\alpha_i \leftarrow \frac{(r_i)^T\, r_i}{(r_i)^T\, A\, r_i}$
$x_{i+1} \leftarrow x_i + \alpha_i\, r_i$
$tol \leftarrow \|r_i\|$
$i \leftarrow i + 1$
End while

El método del gradiente conjugado es una modificación del gradiente descendente, que acelera la convergencia evitando que se repitan las direcciones de búsqueda, como se puede observar en la Figura 3.3 b). En el método del gradiente conjugado las direcciones de avance para minimizar f(x) pasan a ser direcciones conjugadas ortogonales (subespacio de Krylov) que se construyen usando los residuos como vectores de base. Estas direcciones conjugadas para cada iteración d_i tienen la forma:

$$d_i = u_i + \sum_{k=0}^{i-1} \beta_{ik} * d_k \tag{3. 24}$$

Como en el método de los gradientes conjugados se usan los residuos como vectores de base, $u_i = r_i$ y la expresión anterior queda:

$$d_{i+1} = r_{i+1} + \beta_{i+1}\, d_i, \tag{3. 25}$$

siendo:

$$\beta_{i+1} = \frac{r_{i+1}^T\, r_{i+1}}{r_i^T\, r_i}. \tag{3. 26}$$

De este modo, el método del gradiente conjugado cumple las siguientes condiciones [21]:

- Cada dirección de descenso es conjugada, respecto a la matriz A, con todas las direcciones de descenso calculadas anteriormente:

$$(d_i)^T A\, d_j = 0\ ;\ \ 0 \le j < i \tag{3. 27}$$

- Los residuos en los puntos generados en el método del gradiente conjugado son ortogonales a las direcciones de descenso en las iteraciones anteriores:

$$r_i \cdot d_j = 0;\ \ 0 \le j < i \tag{3. 28}$$

El algoritmo iterativo se ejecuta hasta que el residuo es menor que la tolerancia establecida o se alcanza el número máximo de iteraciones:

Algoritmo del Método del Gradiente Conjugado
Input $A, x_0, b, \varepsilon, n_{max}$
$i \leftarrow 0\ ; r_0 \leftarrow b - A\, x_0\ ; tol\ \leftarrow \|r_0\|$
While $i < n_{max}\ y\ tol > \varepsilon$ **do**
$\alpha_i \leftarrow \frac{(r_i)^T\, r_i}{(d_i)^T\, A\, d_i}$
$x_{i+1} \leftarrow x_i + \alpha_i\, d_i$
$r_{i+1} \leftarrow r_i - \alpha_i A\, d_i$
$\beta_{i+1} \leftarrow \frac{r_{i+1}^T\, r_{i+1}}{r_i^T\, r_i}$
$d_{i+1} \leftarrow r_{i+1} + \beta_{i+1}\, d_i$
$tol\ \leftarrow \|r_{i+1}\|$
$i \leftarrow i + 1$
End while

En nuestro caso, se utilizará el método del gradiente conjugado conjuntamente con un precondicionador (lo que se conoce como método del gradiente conjugado precondicionado PCG). La idea del precondicionamiento consiste en buscar una semilla inicial x_0 que acelere la búsqueda iterativa que realiza el solver. Este cálculo también se puede realizar en cada paso de la búsqueda iterativa para acelerar dicha búsqueda de forma recursiva. Para que estos precondicionadores sean eficientes, el coste computacional del cálculo de x_0 debe ser inferior a las iteraciones del gradiente

conjugado que se ahorran. El precondicionador que se adopta en este trabajo es un multigrid algebraico aplicado en cada iteración del gradiente conjugado.

3.4.2. Precondicionamiento utilizando multigrid algebraico

Los métodos multigrid han adquirido popularidad en el campo de la computación científica por su escalabilidad, ya que estos métodos son capaces de resolver un sistema lineal disperso con "n" incógnitas mediante "O(n)" operaciones.

Los métodos multigrid son métodos iterativos, cuyo esquema de funcionamiento se muestra en la Figura 3.4. Se parte de una malla de elementos finitos y se van aplicando operadores de restricción, de modo que la malla se va haciendo más gruesa. En cada paso, se realiza también un suavizado para reducir el error. Se continúa haciendo la malla más gruesa hasta que se alcanza un tamaño adecuado para que el sistema se pueda resolver por métodos directos. Una vez obtenida la solución del sistema en la malla más gruesa y el error en esta, se emplean operadores de interpolación para recorrer el camino inverso y obtener la solución para la malla más fina (malla inicial). Este proceso se conoce como ciclo en V [22].

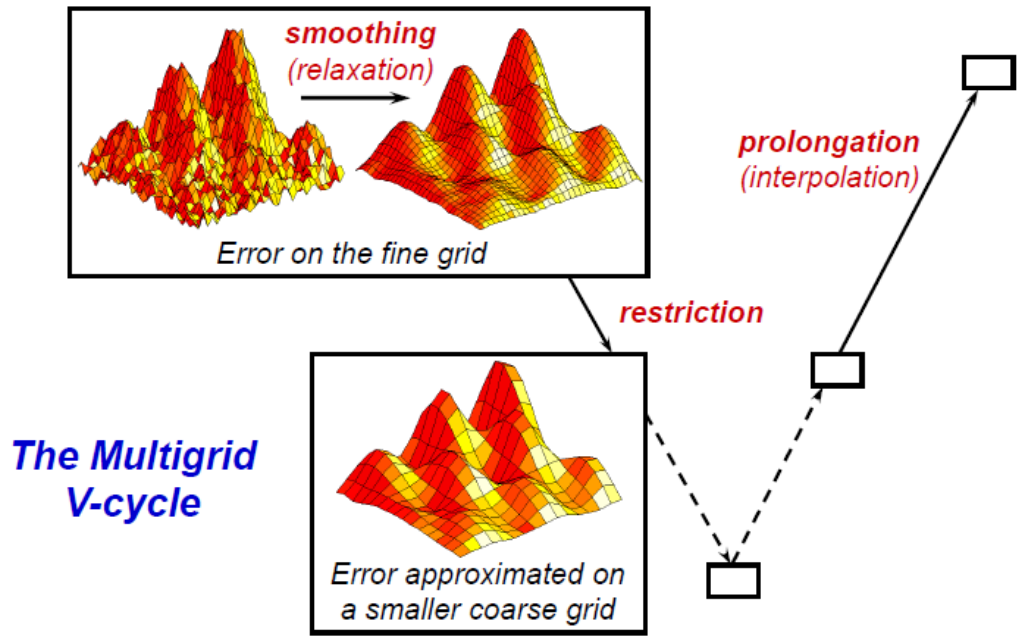

Figura 3.4. Ciclo en V de un método multigrid. Recuperado de [23].

Un método multigrid se ejecuta en dos fases: la fase de configuración y la fase de resolución. En la fase de configuración se definen los operadores de restricción e interpolación que se utilizarán posteriormente. En la fase de resolución se llevan a cabo las iteraciones del ciclo en V hasta que se obtiene la convergencia deseada.

El multigrid algebraico (AMG) es un caso particular de método multigrid que no requiere computación basada en patrones (stencil-based computation). Permite trabajar con cualquier tipo de malla ya que los procesos de restricción e interpolación se realizan completamente sobre la matriz. Esta flexibilidad que proporcionan los algoritmos AMG provoca también que el algoritmo sea complejo. En los algoritmos de multigrid algebraico el tiempo necesario para la fase de configuración no es trivial y puede llegar a igualar al tiempo necesario en la fase de resolución [24].

Aunque estos métodos pueden usarse como solvers para sistemas de ecuaciones lineales, el enfoque más robusto suele ser utilizarlos como precondicionadores en métodos de resolución iterativos, como el método del gradiente conjugado. En las simulaciones realizadas en este proyecto, se utiliza el multigrid algebraico como precondicionador para el método del gradiente conjugado. Como se muestra en la Figura 3.5, en cada iteración del algoritmo del gradiente conjugado, se realiza un ciclo en V del AMG.

La solución obtenida en una iteración del método del gradiente conjugado entra al ciclo en V del multigrid algebraico. Mediante operadores de restricción se reduce la malla quedando un sistema de ecuaciones reducido que se puede resolver mediante métodos directos. La solución de este sistema se prolonga hasta obtener x', que es la solución del sistema proporcionada por el multigrid algebraico. El vector x' se aproxima más a la solución real que el vector x obtenido con el método del gradiente conjugado. Por este motivo, para la siguiente iteración del gradiente conjugado, se introduce x=x', acelerando así la convergencia (y por lo tanto reduciendo el número de iteraciones del gradiente conjugado). Esto es lo que se conoce como Método del Gradiente Conjugado Precondicionado (PCG) con multigrid algebraico.

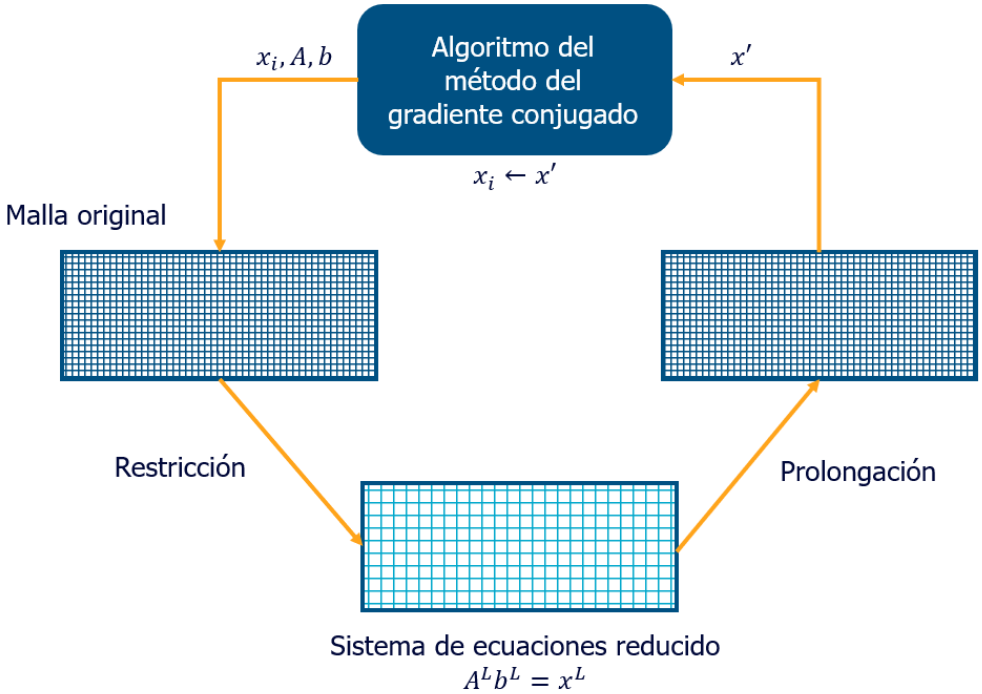

Figura 3.5. Funcionamiento del método del gradiente conjugado con multigrid algebraico como precondicionador.

El multigrid algebraico que se utilizará en las simulaciones de este proyecto es el llamado BoomerAMG, proporcionado por la librería Hypre. Se trata de una implementación paralela del método multigrid algebraico, que permite abordar problemas con mallas estructuradas y no estructuradas al operar sobre la matriz de coeficientes. BoomerAMG permite al usuario elegir entre diferentes técnicas de relajación y suavizado.

3.4.3. Formato de datos distribuido

Además del método de resolución, para asegurar la eficiencia de la resolución en paralelo del problema, también es de gran importancia almacenar la matriz de coeficientes y los vectores de forma distribuida, así como las operaciones que se realicen con estos.

Por este motivo, es relevante el formato de datos en el que se van a almacenar los sistemas lineales de ecuaciones (que, como se ha comentado anteriormente, son sistemas dispersos cuyas matrices de coeficientes tienen un número importante de ceros).

Para almacenar los sistemas de ecuaciones de forma distribuida (repartidos entre los diferentes procesadores) las interfaces de Hypre utilizan el formato que denominan ParCSR. Antes de describir cómo se almacena una matriz en el formato ParCSR, se va a dar una breve explicación sobre el formato de datos CSR (Compressed Sparse Row). El formato CSR permite economizar los recursos de memoria, utilizando tres vectores para almacenar una matriz de dimensiones mxn. Estos tres vectores son [25]:

- Vector "Data": De dimensión n_z, contiene los elementos no nulos de la matriz ordenados por filas.
- Vector "Column_index": De dimensión n_z, contiene el número de columna de la matriz "A" en la que se sitúa cada coeficiente.
- Vector "Row_index": De dimensión m. Contiene el número de coeficientes no nulos de cada fila de la matriz de coeficientes.

Una matriz "A" expresada en formato ParCSR está dividida en "p" partes A_k con k=1, …, p, donde cada una de las partes A_k está almacenada en uno de los procesadores que se usan para resolver el problema de forma distribuida. Cada A_k se divide a su vez en dos matrices D_k y O_k. D_k es una matriz cuadrada de dimensiones $n_k \, x \, n_k$, siendo n_k el número de filas almacenadas en cada procesador "k". O_k es la matriz que contiene los coeficientes de A_k que pertenecen a columnas que están almacenadas en otros procesadores distintos del procesador "k". La matriz O_k es, en general, una matriz muy dispersa (con muchos coeficientes nulos). Además, se define un vector "COL_MAP_O_k", que contiene los índices de las columnas (de la matriz A_k) a las que pertenecen los coeficientes no nulos de O_k [26].

Para ilustrar cómo se expresa una matriz en el formato ParCSR, se incluye un ejemplo en la Figura 3.6. En este caso, la matriz se distribuye en tres procesadores. Con un recuadro se muestran las matrices $D_1, D_2 \, y \, D_3$, almacenadas localmente en los procesadores 1, 2 y 3. El resto de los coeficientes no nulos de cada partición se incluyen en las matrices O_1, O_2, O_3. Los vectores "COL_MAP_O_k" son en este caso: COL_MAP_O_1= (5,6,8), COL_MAP_O_2= (1,2,4,9) y COL_MAP_O_3= (3,4,5,8).

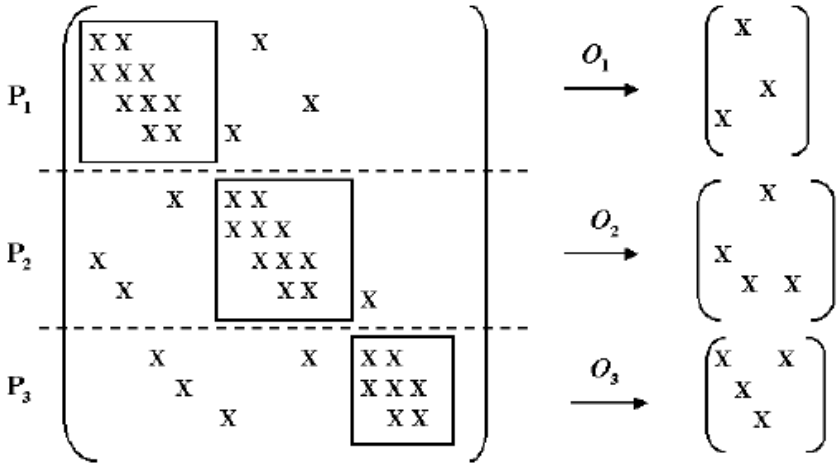

Figura 3.6. Ejemplo de matriz en formato ParCSR, distribuida en tres procesadores. Recuperado de [26].

Una vez vista la información de la matriz que se almacena en cada procesador, es necesario establecer cómo se realizan las comunicaciones de datos entre procesadores. Para esto se genera un paquete de comunicaciones ("communication package") en cada procesador. Este paquete de comunicaciones contiene la información necesaria para realizar la multiplicación de una matriz "A" por un vector "x". Es importante destacar que cada procesador tiene almacenadas localmente un número n_k de filas de la matriz A y las filas correspondientes del vector x.

Por esto, al realizar la multiplicación $A_k x = D_k x_k + O_k \tilde{x}_k$, la operación $D_k x_k$ se puede llevar a cabo sin ninguna comunicación con otros procesadores ya que las filas de x_k están almacenadas localmente en el procesador "k". Como las filas del vector $\tilde{x}_k$ están almacenadas en otros procesadores, se requieren comunicaciones entre procesadores para llevar a cabo la operación.

Ya que inicialmente cada procesador solo conoce los datos de su partición, se realiza un proceso de búsqueda secuencial en el que, mediante comunicaciones punto a punto, cada procesador conoce cuáles son los procesadores de los que debe recibir y a los que debe enviar datos. Las comunicaciones se realizan con las herramientas que proporciona el estándar MPI.

4. OPTIMIZACIÓN TOPOLÓGICA

4.1. Introducción

En este capítulo se detallan las herramientas utilizadas para realizar la optimización topológica de diferentes alas de aeronaves, cuyos resultados se presentarán en el Capítulo 5.

Las técnicas de optimización topológica tienen por objetivo encontrar la distribución óptima de material dentro de un dominio de diseño, de forma que se minimice una función de coste, con una serie de restricciones. Mediante la optimización topológica se consiguen diseños conceptuales de alto rendimiento. Estas técnicas se usan dentro de la industria en las primeras fases del diseño, obteniendo estructuras ligeras para una amplia variedad de aplicaciones.

Los métodos de optimización topológica se pueden clasificar en tres grandes categorías, dependiendo en la forma en la que se introduce el dominio (o frontera) de la estructura a optimizar:

- Métodos Eulerianos: Estos métodos obtienen el diseño conceptual de una estructura a partir de una representación implícita de su frontera.
- Métodos Lagrangianos: Son métodos que utilizan una representación explícita de la frontera de la estructura a optimizar. Necesitan como entrada una malla computacional o un modelo CAD.
- Métodos basados en densidades: Trabajan con una malla fija de elementos finitos a partir de la cual obtienen la distribución óptima de material que minimiza una función objetivo. Entre los métodos basados en densidades más populares se encuentran el método de la homogeneización y el método SIMP (Solid Isotropic Material with Penalization). Este último es el que se va a usar en las optimizaciones de alas de aeronaves del presente proyecto. El método SIMP es el más implementado por software comerciales de optimización topológica, probablemente debido a la relativa simplicidad de su implementación (respecto a otras técnicas de optimización topológica) a partir de un software de análisis estructural mediante elementos finitos.

A la hora de resolver un problema de optimización topológica, la resolución de la malla utilizada cobra una gran relevancia. Se requiere una resolución de la malla lo suficientemente alta para que el error de discretización obtenido al evaluar la función objetivo sea aceptable. La resolución de la malla también permite captar los suficientes detalles de la geometría de la estructura a diseñar.

Por este motivo, se tiene un problema con un gran coste computacional, que se resolverá mediante un enfoque en paralelo para hacer que sea abordable y aprovechar al máximo los recursos computacionales disponibles. A continuación, se describe en profundidad el método SIMP y cómo se paraleliza para aprovechar las ventajas de la computación paralela en problemas de optimización topológica y, en concreto, en las optimizaciones de estructuras de aeronaves del presente proyecto.

4.2. Método Solid Isotropic Material with Penalization (SIMP)

El método SIMP (Solid Isotropic Material with Penalization) es uno de los métodos de optimización topológica más extendidos y el que se usa en las optimizaciones de este proyecto. A continuación, se describe el método SIMP, para centrarnos en cómo se paraleliza en el siguiente apartado.

El problema de optimización topológica se puede plantear del siguiente modo:

$$\min \quad f(\rho_e, u) \tag{4. 1}$$

$$\text{Sujeto a:} \quad \mathrm{K}(\rho_e)\mathrm{u} = \mathrm{f} \tag{4. 2}$$

$$V(\rho_e) \leq V^* \tag{4. 3}$$

$$0 \leq \rho_e(x) \leq 1\,, x \in \mathcal{D} \tag{4. 4}$$

donde $f(\rho_e, u)$ es la función objetivo (compliance) a minimizar, ρ_e es el vector de densidades de los elementos de la malla, u es la respuesta del sistema (campo de desplazamientos), K es la matriz de rigidez global, f es el vector fuerza y x es el vector de elementos finitos del modelo, que pertenece al dominio de diseño $\mathcal{D}$. Se pone como condición que el volumen de material $V(\rho_e)$ sea menor que un volumen objetivo V^* y que las densidades elementales $\rho_e(x)$ (incógnitas que definen el diseño) estén entre cero y la unidad.

La resolución del problema a menudo lleva a obtener grandes áreas con densidades intermedias, que en la práctica son imposibles de fabricar. Para solucionar este problema y obtener distribuciones de material lleno/vacío se emplean técnicas de penalización. El método SIMP usa una función de interpolación entre lleno y vacío, que determina la matriz de rigidez de cada elemento (K_e) de la siguiente forma:

$$K_e = K_{min} + \rho_e^p\,(K_0 - K_{min}) \tag{4. 5}$$

donde K_0 es la matriz de rigidez del material lleno y K_{min} la del material vacío. El factor de penalización se denomina p y es mayor que la unidad. Para obtener diseños de tipo lleno/vacío se suele tomar $p \geq 3$.

Por otro lado, es conveniente normalizar el campo de densidades para evitar dificultades numéricas y de modelado, como la dependencia de malla. Existen diversas técnicas para normalizar el campo de densidades. En este proyecto se va a usar un filtro de Helmholtz. Este tipo de solución tiene la ventaja de que el filtro se calcula como la resolución de una EDP, lo que facilita su resolución en paralelo utilizando un enfoque similar al que se utiliza para resolver el problema elástico, en lugar de realizar operaciones entre elementos vecinos en una región determinada. El problema consiste en la modificación de las densidades con las que se penaliza la rigidez de los elementos, con la finalidad de obtener un campo de densidades con ciertas propiedades de continuidad de varios órdenes.

El filtro de densidades se define como la solución de una ecuación en derivadas parciales de Helmholtz en el dominio $\Omega \subset \mathbb{R}^n$:

$$-r^2\nabla^2\tilde{\rho} + \tilde{\rho} = \rho\,;\ \tilde{\rho} \in \Omega \tag{4. 6}$$

$$\frac{\partial\tilde{\rho}}{\partial n} = 0\,;\ \tilde{\rho} \in \Omega \tag{4. 7}$$

siendo ρ el campo de densidades en los nodos y r una longitud que se toma como radio de la integral de convolución para calcular el filtro de densidades.

Para las optimizaciones realizadas en el presente proyecto se fija como objetivo minimizar la "compliance" de la estructura, esto es, maximizar la rigidez. Por esto, se define la función objetivo de la siguiente forma:

$$compliance = f = \mathrm{f}^T u \tag{4. 8}$$

siendo f el vector de cargas y u el vector de desplazamientos.

Planteando el sistema de ecuaciones del problema de elasticidad lineal:

$$Ku = \mathrm{f} \tag{4. 9}$$

La sensibilidad de la función objetivo de la expresión (4.8) con respecto a la variable de diseño ρ_e es:

$$\mathrm{f}_\rho = \frac{\partial f}{\partial(\rho_e)} = -u^{*T}\frac{\partial \mathrm{K}}{\partial(\rho_e)}u = -u^{*T}p(\rho_e)^{p-1}(K_0 - K_{min})u \tag{4. 10}$$

Donde u^* es la solución del problema adjunto:

$$Ku^* = \frac{\partial f}{\partial u} \tag{4. 11}$$

Siendo f la función objetivo, de modo que $\frac{\partial f}{\partial u} = \mathrm{f}$ para maximizar la rigidez estructural. Así, de las expresiones (4.9) y (4.11) se deduce que $u = u^*$.

El vector de densidades (parámetro de diseño) se actualiza siguiendo los criterios de optimización propuestos por Bendsoe [27]:

$$\rho_{e\ k+1} = \begin{cases} \max\big((1-\zeta)\rho_{ek}, 0\big) & si \quad \rho_{ek}B_{ek}^{\eta} \le \max\big((1-\zeta)\rho_{ek}, 0\big) \\ \min\big((1+\zeta)\rho_{ek}, 1\big) & si \quad \min\big((1-\zeta)\rho_{ek}, 0\big) \le \rho_{ek}B_{ek}^{\eta} \\ \big(\rho_{ek}B_{ek}^{\eta}\big)^q & en\ cualquier\ otro\ caso \end{cases} \tag{4. 12}$$

donde ζ es un parámetro positivo que marca el avance, η es un coeficiente de amortiguación, q es un factor de penalización para conseguir que la distribución de material sea de tipo lleno/vacío y el factor B_{ek} es la condición de Karush-Kuhn-Tucker (KKT) para que la solución sea óptima:

$$B_{ek} = -\frac{\partial f(\rho_e)}{\partial \rho_e}\left(\lambda\frac{\partial V(\rho_e)}{\partial \rho_e}\right)^{-1} = 1 \tag{4. 13}$$

λ es el coeficiente de Lagrange que se calcula mediante el método de la bisección.

El algoritmo de la expresión (4.12) se ejecuta hasta que se alcanza un número prefijado de iteraciones o cuando el cambio en el vector de densidades o en la función objetivo están por debajo de un valor determinado.

4.3. Paralelización del método basado en densidades SIMP

Como se ha comentado anteriormente, las optimizaciones planteadas en el presente proyecto tienen una complejidad computacional elevada, por lo que se hace necesario abordarlas con un enfoque en paralelo. En este apartado se pretende dar una visión global de cómo se integran las diferentes herramientas y métodos de resolución expuestos en el Capítulo 3, con el fin de realizar las optimizaciones topológicas mediante el método SIMP en paralelo.

En la Figura 4.1 se recoge un esquema del proceso completo de optimización topológica, desde que se introduce el modelo y las condiciones de contorno hasta la convergencia del método basado en densidades SIMP.

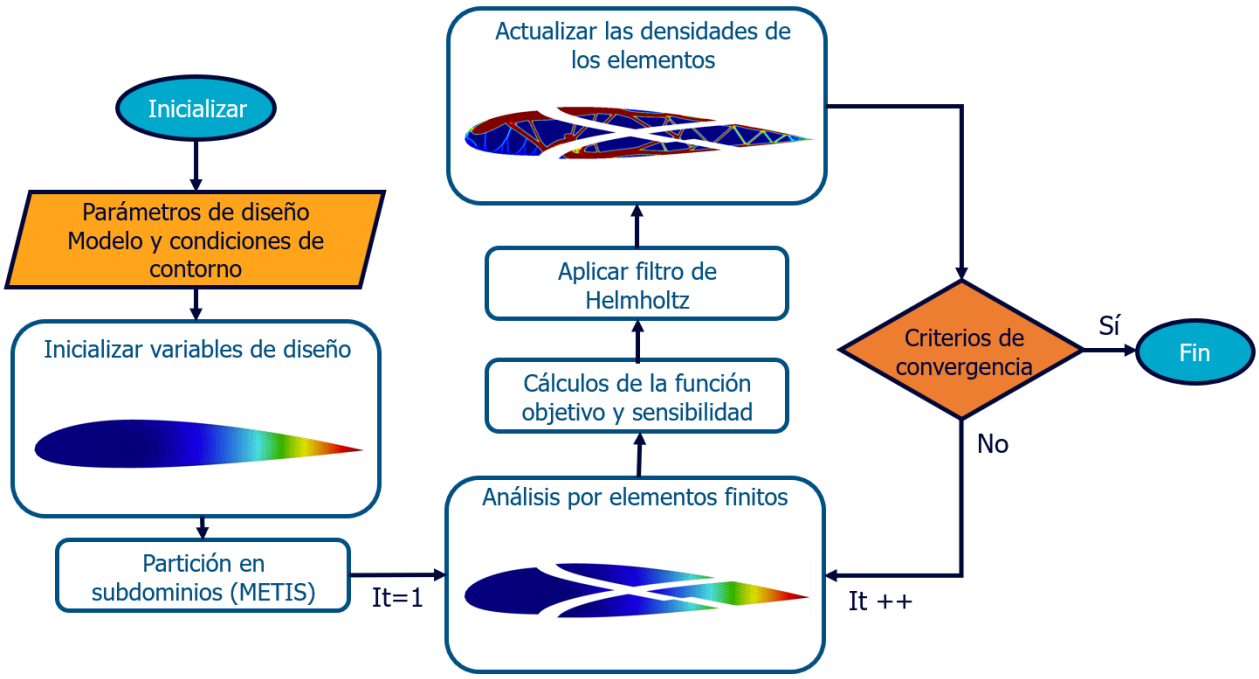

Figura 4.1. Proceso de optimización topológica.

Como se puede observar en el esquema, al iniciar el proceso se introducen como datos de entrada el modelo del espacio de diseño y las condiciones de contorno, así como los parámetros de diseño (cargas aplicadas, volumen máximo de material, radio de filtro de densidades, etc.). Una vez inicializadas las variables de diseño, se lleva a cabo la partición del problema en subdominios sin solapamiento, para asignar cada subdominio a uno de los procesadores que trabajan en paralelo. Esta partición, que se mantendrá durante todo el proceso, se lleva a cabo mediante el software METIS. Este software realiza la partición de grafos multinivel, tal y como se ha expuesto en el apartado 3.3.

Una vez que se tiene la partición, comienza el proceso iterativo que se realiza en paralelo para cada uno de los subdominios. Dentro de este proceso iterativo, se resuelve por elementos finitos el sistema de ecuaciones del problema de elasticidad lineal (ecuación (3.16)). Para resolverlo de forma distribuida, se almacena la matriz de coeficientes y los vectores de forma distribuida, empleando el formato ParCSR. La resolución del sistema se realiza de forma distribuida empleando el método del gradiente conjugado distribuido precondicionado con un multigrid algebraico clásico (cuyos fundamentos han sido expuestos en los apartados 3.4.1 y 3.4.2 respectivamente).

Lo mismo ocurre a la hora de solucionar la ecuación en derivadas parciales de Helmholtz que se emplea como filtro de densidades: se almacenan los datos de forma distribuida en formato ParCSR y como solver se utiliza el gradiente conjugado distribuido (PCG) precondicionado con un multigrid algebraico clásico.

Una vez hecho esto, se calcula de forma distribuida la función objetivo, las sensibilidades y se actualiza el valor de las variables de diseño (vector de densidades). Estos cálculos no requieren comunicaciones, ya que solo se requiere la información de los elementos del propio subdominio y se puede hacer con la información de cada procesador.

Cuando se alcanzan los criterios de convergencia del método previamente definidos, finaliza el proceso iterativo y se obtiene la distribución de material lleno/vacío que maximiza la rigidez, dentro del espacio de diseño establecido. Se obtiene así el diseño conceptual buscado, que servirá como base para el diseño final de la estructura a fabricar.

5. RESULTADOS EXPERIMENTALES

En este capítulo se van a presentar las distintas optimizaciones realizadas y sus resultados. Dentro de las estructuras de aeronaves, se escogen diferentes tipos de alas para las que se busca obtener la distribución óptima de material.

En primer lugar, se presentan los modelos a optimizar. Posteriormente, se muestran imágenes de los diseños conceptuales obtenidos y, en los casos en lo que es posible, los tiempos computacionales y aceleraciones obtenidos ejecutando el problema en diferente número de procesadores.

Todas las optimizaciones que se presentan se han lanzado en el clúster de computación científica del grupo MC3 de la UPCT. Se han utilizado dos nodos, cada uno de ellos equipado con dos procesadores Intel(R) Xeon(R) CPU E5-2687W v4 @3.00GHz. Estos procesadores contienen 12 núcleos cada uno (24 haciendo hyperthreading). Cada nodo dispone de 256Gb de memoria RAM y los nodos se conectan entre sí utilizando una red de 10Gbits.

5.1. Modelos de estructuras de aeronaves a optimizar

El objetivo de este proyecto es aplicar las técnicas de computación en paralelo anteriormente expuestas para obtener diseños conceptuales de estructuras de aeronaves. Dentro de las estructuras de aeronaves, se decide dedicar este trabajo a las optimizaciones de alas de aviones, escogiendo distintos tipos de alas. Por este motivo, antes de presentar los modelos realizados para las optimizaciones, conviene introducir brevemente las partes que conforman la estructura del ala de un avión.

Las alas de un avión son estructuras concebidas para producir sustentación cuando se mueve a gran velocidad a través del aire. El diseño del ala depende de factores como el tamaño, el peso, la velocidad deseada para el vuelo y la aceleración de subida requerida [28].

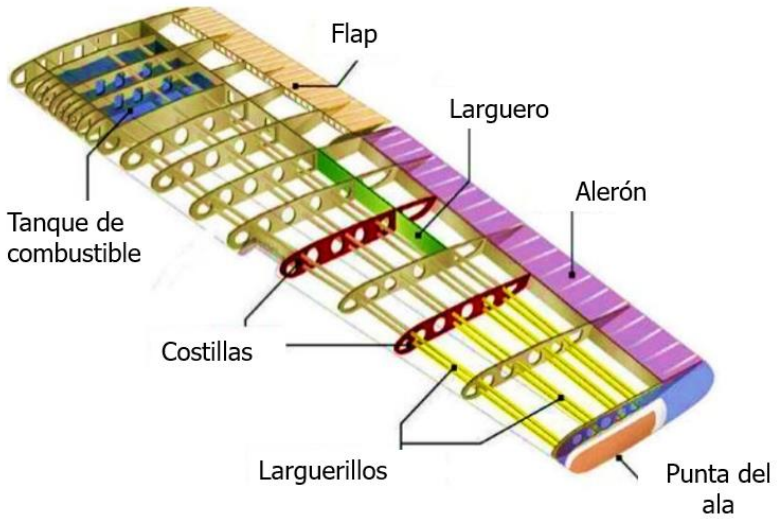

Figura 5.1. Estructura interna del ala de un avión. Recuperado de [29]

Las principales partes de la estructura de un ala, que se muestran en la Figura 5.1, son las siguientes [28]:

- Costillas: Suelen ser perfiles fabricados a partir de una única chapa doblada con los contornos exteriores del perfil del ala. Suelen estar reforzadas con perfiles rigidizadores para prevenir la flexión lateral. Se les practican orificios (aligeramientos) para reducir su peso y poder introducir el cableado. Las costillas son las encargadas de mantener la geometría del ala, transferir las cargas aerodinámicas externas a los largueros, actuar de soporte para los larguerillos y redistribuir la cizalladura en los lugares de cargas concentradas. En la Figura 5.2 se muestra en detalle la estructura de una costilla, colocada en un ala con dos largueros.

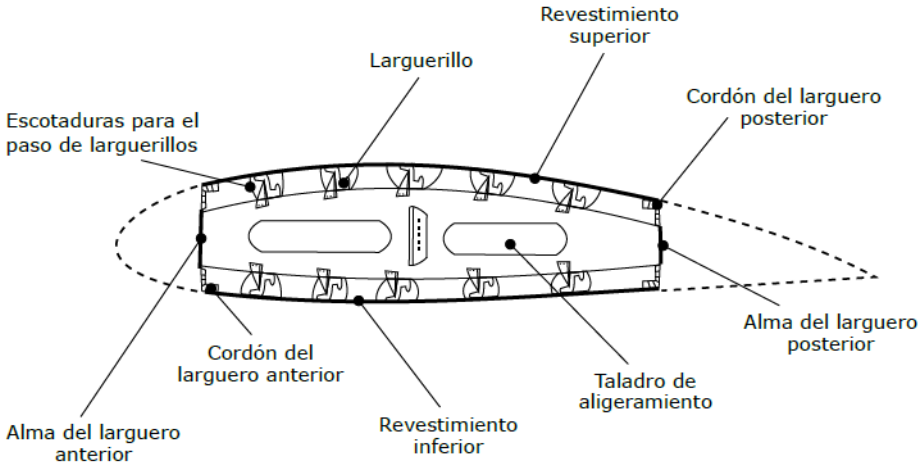

Figura 5.2. Vista lateral de una costilla colocada en un ala con dos largueros. Recuperado de [30]

- Largueros: Son elementos longitudinales que recorren el ala de la raíz a la punta. Suelen ser perfiles en doble T, aunque también se pueden colocar perfiles con otras configuraciones. Son los elementos estructurales más importantes, ya que soportan los esfuerzos de flexión (compresión en la parte superior del ala y tracción en la parte inferior) debidos a las cargas aerodinámicas. Existen alas de un larguero, dos largueros (las más comunes en la aviación comercial) e incluso multilarguero. Los largueros se unen al revestimiento tanto por la parte inferior como por la superior.
- Larguerillos: Son perfiles que unen costillas y revestimientos. Proporcionan momento de inercia fuera del plano a los revestimientos, que de otra forma tendrían una estabilidad mínima.
- Revestimientos: Se corresponden con la superficie exterior del ala, por lo que contribuyen a su forma aerodinámica. Además, son los encargados de recibir prácticamente la totalidad de las cargas que actúan sobre el ala y transferirlas a las costillas y los largueros. En la Figura 5.3 se recogen distintas imágenes de la fabricación del ala en las que se puede apreciar la forma de los revestimientos y cómo estos se colocan respecto a las costillas.

La zona de la estructura del ala compuesta por los largueros, los larguerillos y el revestimiento recibe el nombre de cajón de torsión y en su interior suelen encontrarse los depósitos de combustible. Las costillas pueden servir para separar varios depósitos de combustible [30].

Figura 5.3. Fabricación del ala de una aeronave. Recuperado de [31].

Aunque los elementos que componen la estructura interna del cajón del ala son comunes a todas las aeronaves, existen distintas particularidades en la configuración alar según el tipo de aeronave. De este modo se consiguen las mejores características para cada tipo de vuelo. Por este motivo, existen diversas configuraciones de ala: rectangular, en delta, en flecha, en flecha invertida, elíptica, trapezoidal, variable…

Este proyecto se centra en optimizar las distribuciones de material de tres alas de entre los distintos tipos citados: un ala en flecha correspondiente al avión comercial Airbus A320, un ala elíptica correspondiente a un avión de combate histórico y un ala rectangular típica de un avión ligero. También se optimiza el perfil de una costilla (en 2D). Estos modelos se describen en los siguientes apartados.

5.1.1. Modelo de la costilla del ala de un avión

Antes de lanzar las optimizaciones para las alas de distintas aeronaves, se opta por optimizar una estructura más sencilla, con menor número de elementos en la malla y de grados de libertad (y que por tanto tendrá menor coste computacional). Esta estructura es el perfil aerodinámico correspondiente a una costilla de las que componen el ala, en dos dimensiones.

Existen muchos tipos de perfiles sustentadores estandarizados, aunque en la actualidad, gracias al empleo de grandes computadores, también se crean perfiles específicos para cada necesidad. Dentro de los perfiles estandarizados, existen diferentes familias de perfiles, entre las que se pueden citar las siguientes: la serie NACA, la serie TsAGI, la serie Eppler, la serie Joukowsky y la serie Clark Y. Entre estos, los más utilizados y estudiados son los perfiles NACA, que mediante su designación numérica aportan información de las características del perfil. Estos perfiles fueron desarrollados en Estados Unidos por la National Advisory Committee for Aeronautics (NACA).

Para comprender cómo funciona la designación de los perfiles NACA se van a exponer brevemente los principales parámetros de un perfil aerodinámico. En la Figura 5.4 se presentan las principales partes de un perfil aerodinámico y parámetros que lo definen [32]:

- Intradós: Parte inferior del perfil alar.
- Extradós: Parte exterior del perfil alar.
- Línea de cuerda: Es una línea recta que une el borde de ataque (la primera parte que tiene contacto con el aire) con el borde de fuga (punto por el que las corrientes abandonan el perfil).
- Cuerda: Corresponde a la longitud de la línea de cuerda. Es uno de los principales parámetros que se utilizan para definir un perfil aerodinámico.
- Línea de curvatura media: Une el borde de ataque con el de salida de forma que siempre resulte equidistante al extradós y al intradós.
- Curvatura máxima: Distancia máxima entre la línea de curvatura media y la línea de cuerda.
- Espesor máximo: Corresponde a la distancia entre el intradós y el extradós en la zona donde el espesor vertical es máximo.
- Radio del borde de ataque: Es una medida del afilamiento del borde de ataque. Puede variar desde 0 para perfiles afilados hasta un 2% de la cuerda para perfiles más bien achatados.

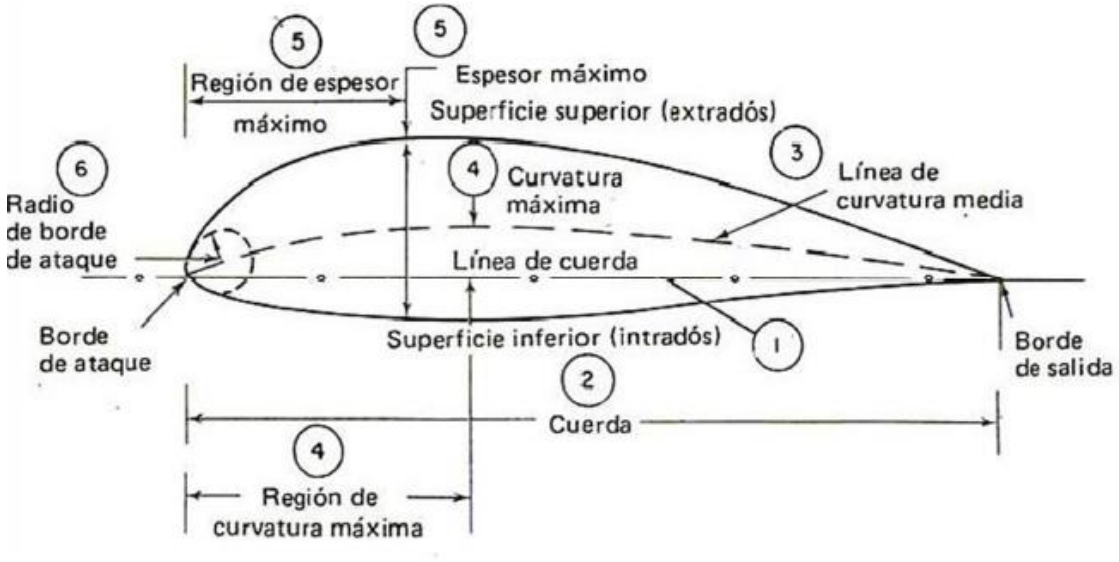

Figura 5.4. Parámetros y regiones de un perfil aerodinámico. Recuperado de [32].

La nomenclatura NACA cuenta con 7 series de perfiles para definir distintos tipos de perfiles aerodinámicos: “four-digit series”, “five-digit series”, modificaciones de las anteriores, “1-series”, “6-series”, “7-series” y “8-series”.

Dentro de este proyecto, se decide modelar un perfil de la serie de cuatro dígitos: el perfil NACA 2315. Se elige este perfil como una simplificación de los perfiles aerodinámicos utilizados en el avión Airbus A320 (cuya ala se modelará posteriormente) que se basan en el perfil NACA 23015 (aunque realizando algunas modificaciones).

Los perfiles NACA de la serie de cuatro dígitos se designan mediante un código de cuatro cifras donde:

- El primer dígito corresponde a la curvatura máxima, expresada en tanto por cien de la cuerda.
- El segundo dígito describe la distancia de máxima curvatura desde el borde de ataque, expresada en tanto por diez de la cuerda.
- Los dos dígitos finales describen el máximo espesor del perfil, en tanto por cien de la cuerda.

De este modo, en el caso del perfil NACA 2315 la designación indica que la máxima curvatura es del 2% y está localizada a una distancia de "0,3*valor de la cuerda" del borde de ataque. El máximo espesor del perfil es del 15% de la cuerda. En el caso de que la designación de un perfil comience con las cifras 00, significa que estamos ante un perfil simétrico.

En la literatura se pueden encontrar las ecuaciones que definen la forma de los perfiles NACA de cuatro dígitos. Estas ecuaciones son las que se implementan en el modelo CAD, representado con el software Gmsh, donde posteriormente se genera también la malla. Las ecuaciones que se usan para definir las coordenadas de la curva superior (x_u, y_u) e inferior (x_L, y_L) del perfil NACA de 4 dígitos son [33]:

$$x_u = x - y_t \, sin\,\theta \tag{5. 1}$$

$$y_u = y_c + y_t \, cos\,\theta \tag{5. 2}$$

$$x_L = x + y_t \, sin\,\theta \tag{5. 3}$$

$$y_L = y_c - y_t \, cos\,\theta \tag{5. 4}$$

donde y_t es la mitad del espesor vertical de un perfil NACA simétrico para un valor dado de la coordenada horizontal "x" y se calcula como:

$$y_t = 5t\,(0{,}2969\,\sqrt{x} - 0{,}1260\,x - 0{,}3516\,x^2 + 0{,}2843\,x^3 - 0{,}1015\,x^4) \tag{5. 5}$$

siendo "t" el valor de los dos últimos dígitos que designan el perfil dividido entre 100, y siendo y_c la curvatura media calculada como:

$$y_c = \begin{cases} \dfrac{m}{p^2}\left(2p\left(\dfrac{x}{c}\right) - \left(\dfrac{x}{c}\right)^2\right), & 0 \le x \le pc \\ \dfrac{m}{(1-p)^2}\left((1-2p) + 2p\left(\dfrac{x}{c}\right) - \left(\dfrac{x}{c}\right)^2\right), & pc \le x \le c \end{cases} \tag{5. 6}$$

donde "m" el primer dígito de la designación del perfil dividido entre 100 y "p" el segundo dígito de la designación del perfil dividido entre 10.

El ángulo θ se define como:

$$\theta = arctan\left(\frac{dy_c}{dx}\right) \tag{5. 7}$$

Se parametriza el archivo ".geo" del modelo CAD en 2D del software Gmsh para que se pueda definir un perfil NACA con cualquier designación de cuatro dígitos.

La forma obtenida para el perfil NACA 2315 implementado en 2 dimensiones se muestra en la Figura 5.5. Se define una longitud de cuerda de 2,2 m y un total de 100 puntos para generar el perfil.

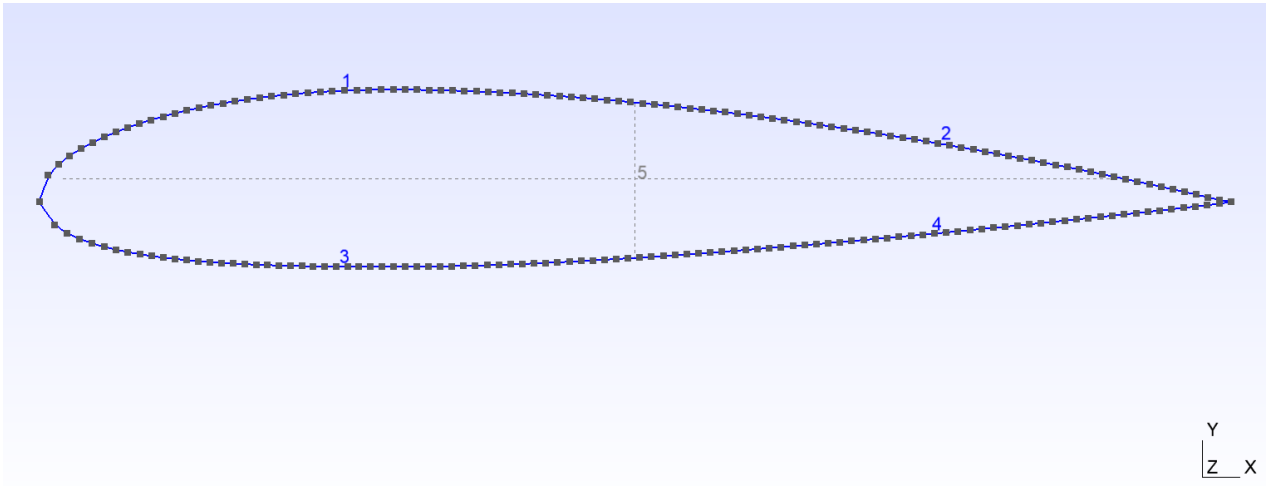

Figura 5.5. Modelo del perfil aerodinámico NACA 2315.

Se opta por mallar la superficie con elementos de orden 1 cuadrangulares y una malla estructurada, aunque también se parametriza el modelo para que permita utilizar elementos triangulares en una malla estructurada o no estructurada. La malla empleada en las optimizaciones se muestra en la Figura 5.6.

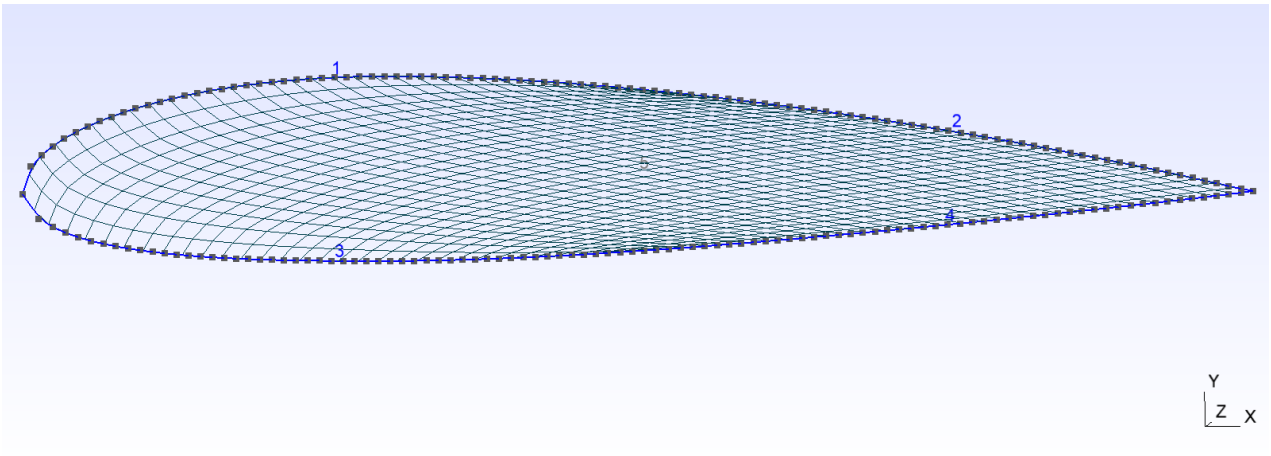

Figura 5.6. Malla estructurada del perfil aerodinámico NACA 2315.

Para completar la descripción de los datos de entrada para la optimización del perfil NACA 2315 es importante exponer las condiciones de contorno impuestas.

Durante el vuelo de una aeronave, en los perfiles aerodinámicos de las alas aparece una diferencia de presiones que el aire ejerce sobre el intradós y el extradós, relacionada con la forma del perfil aerodinámico y el ángulo de ataque del aire. Al ser la presión del aire más elevada en el intradós del perfil, se genera sobre el ala un campo de fuerzas como el que se puede ver en la Figura 5.7. Esta fuerza inducida se puede descomponer en una fuerza vertical de sustentación y una fuerza horizontal de arrastre o resistencia. La sustentación vertical es la responsable de que el avión ascienda y se mantenga en el aire a una altura constante (cuando equilibra al peso). La fuerza de arrastre horizontal que se genera tiene que ser superada por la fuerza de tracción de los motores para que se produzca el avance de la aeronave. La Figura 5.8 ilustra este equilibrio entre las diferentes fuerzas que actúan sobre un avión.

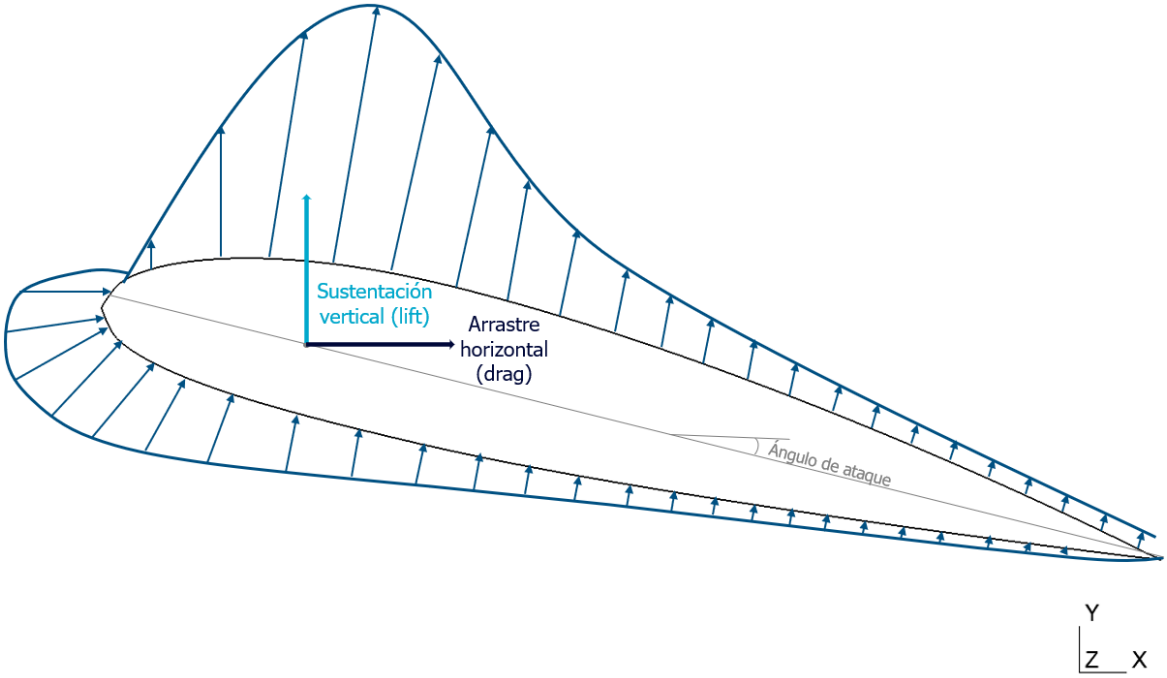

Figura 5.7. Distribución de la fuerza aerodinámica sobre un perfil sustentador. Componente vertical (lift) y horizontal (drag).

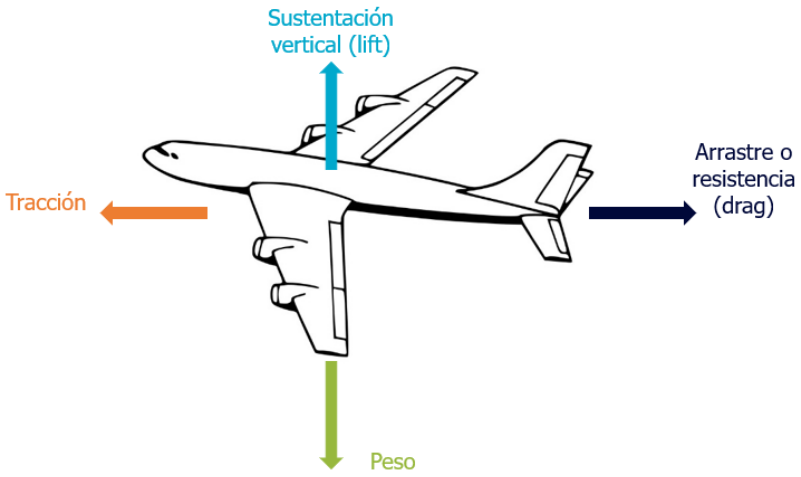

Figura 5.8. Equilibrio de fuerzas en una aeronave.

En nuestro modelo se va a realizar una simplificación importante respecto a las fuerzas actuantes. En lugar del perfil de presión real (como el de la Figura 5.7), se va a considerar una fuerza lineal de sustentación positiva en el en el eje Y (ver Figura 5.9) de valor 1 en el extradós y 0,3 en el intradós. Se introduce también una fuerza horizontal de resistencia de valor 0,1 en el extradós y en el intradós.

En la Figura 5.9 se incluyen también las restricciones de movimiento introducidas. En los cuatro puntos del perfil marcados en color naranja se restringen todos los movimientos. Estas restricciones corresponden a los puntos en los que colocamos los largueros del ala para poder optimizar la distribución de material en la costilla con un perfil NACA 2315.

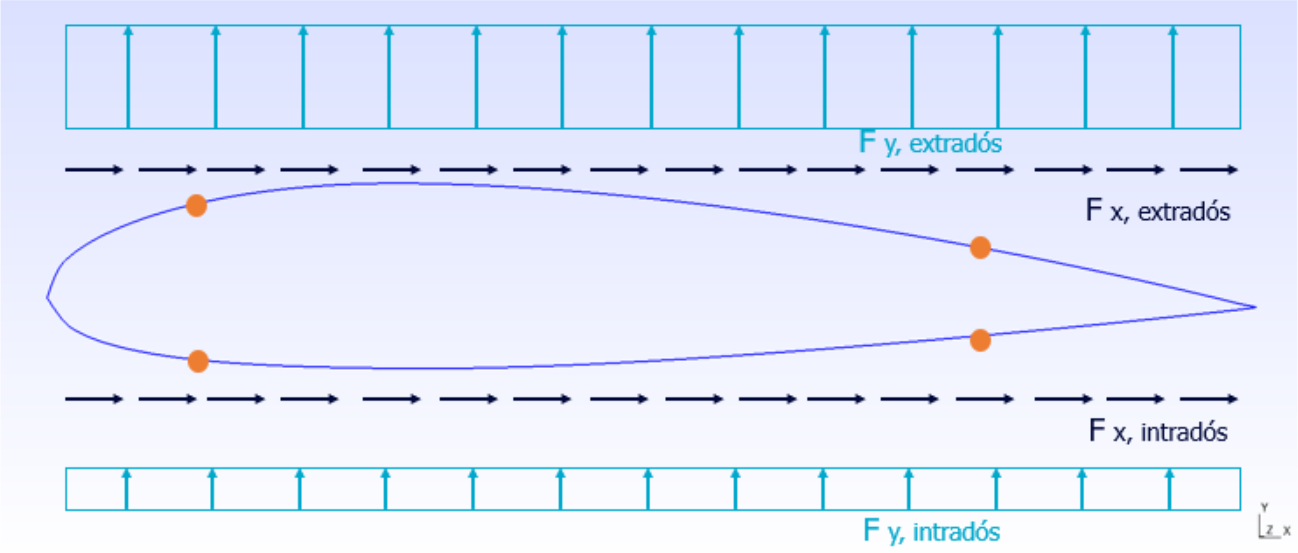

Figura 5.9. Condiciones de contorno impuestas para el perfil aerodinámico NACA 2315.

5.1.2. Modelo del ala de un avión comercial: Airbus A320

En la actualidad, prácticamente el total de los aviones comerciales cuentan con alas en flecha. Esta configuración alar, que comenzó a utilizarse únicamente en aviones de combate, puede verse hoy en día en la gran mayoría de aviones a reacción. Las alas en flecha son aquellas que presentan un cierto ángulo con la perpendicular al fuselaje, esto es, que tienen forma de flecha apuntando hacia el morro, como se puede observar en la Figura 5.10. Aunque es mucho menos común, también existen aviones con los extremos de las alas dirigidos hacia delante (ala en flecha invertida).

Esta configuración presenta ventajas frente a las alas rectangulares cuando se realizan vuelos a altas velocidades (vuelos supersónicos).

Figura 5.10. a) Esquema de un avión con configuración de ala en flecha. b) Airbus A320

Dentro de las numerosas aeronaves que presentan esta configuración alar, se decide modelar y optimizar el ala del avión Airbus A320. El Airbus A320 es un avión comercial de reacción de corto

a medio alcance, desarrollado por Airbus SAS. Es uno de los aviones comerciales más vendidos, con 4983 unidades activas en la actualidad [34].

Dentro del manual de mantenimiento y características del A320 facilitado por la compañía Airbus [35] se encuentran las principales dimensiones de esta aeronave, que se recogen de la Figura 5.11 a la Figura 5.14 y se utilizan como referencia para realizar el diseño del ala.

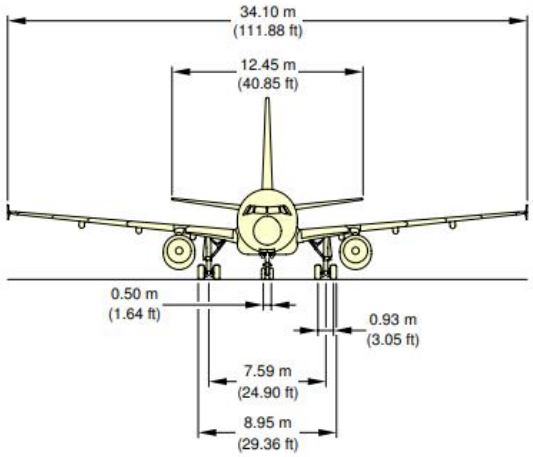

Figura 5.11. Dimensiones del Airbus A320 (I). Recuperado de [35].

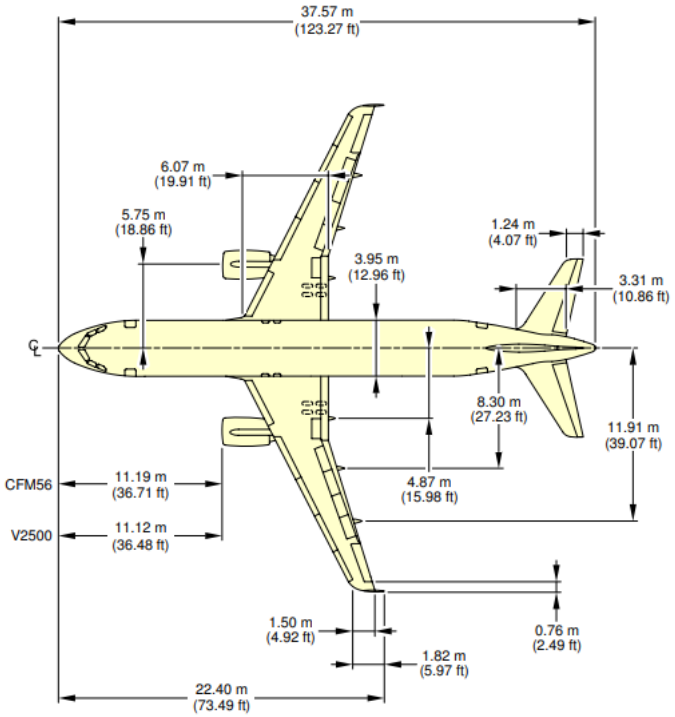

Figura 5.12. Dimensiones del Airbus A320 (II). Recuperado de [35].

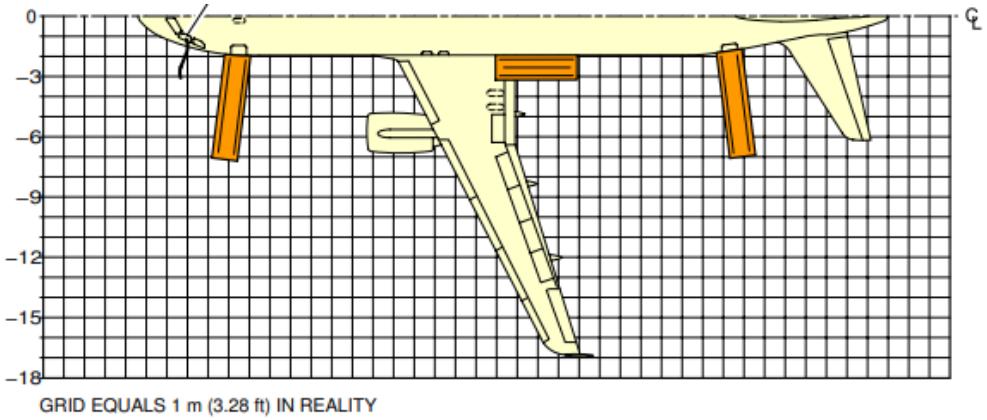

Figura 5.13. Dimensiones del ala del Airbus A320. Recuperado de [35].

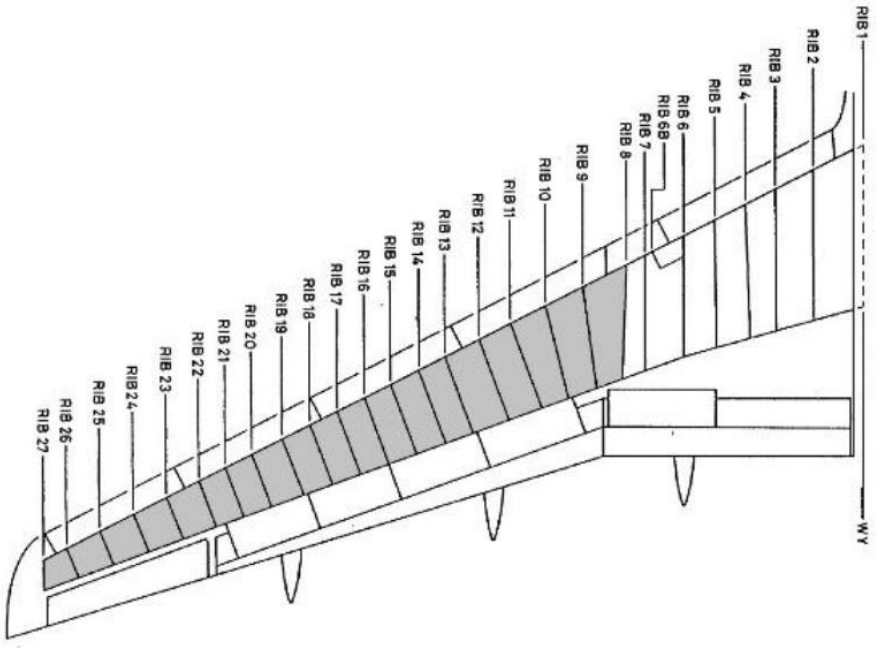

Figura 5.14. Costillas del ala de un Airbus A320. Recuperado de [29].

Tomando como referencia las figuras expuestas, se modela con el software Gmsh el ala en tres dimensiones que se presenta en la Figura 5.15. Se opta por diseñar el cajón del ala con 27 costillas, sin colocar los largueros, para posteriormente mallar todo el volumen que encierran el conjunto de las costillas. El modelo tiene una longitud del ala de 15,60 m. La longitud de cuerda en la raíz del ala es de 6 m (costilla 1) y de 1,40 m en el extremo (costilla 27). A partir de la octava costilla, cambia la inclinación del ala respecto al fuselaje.

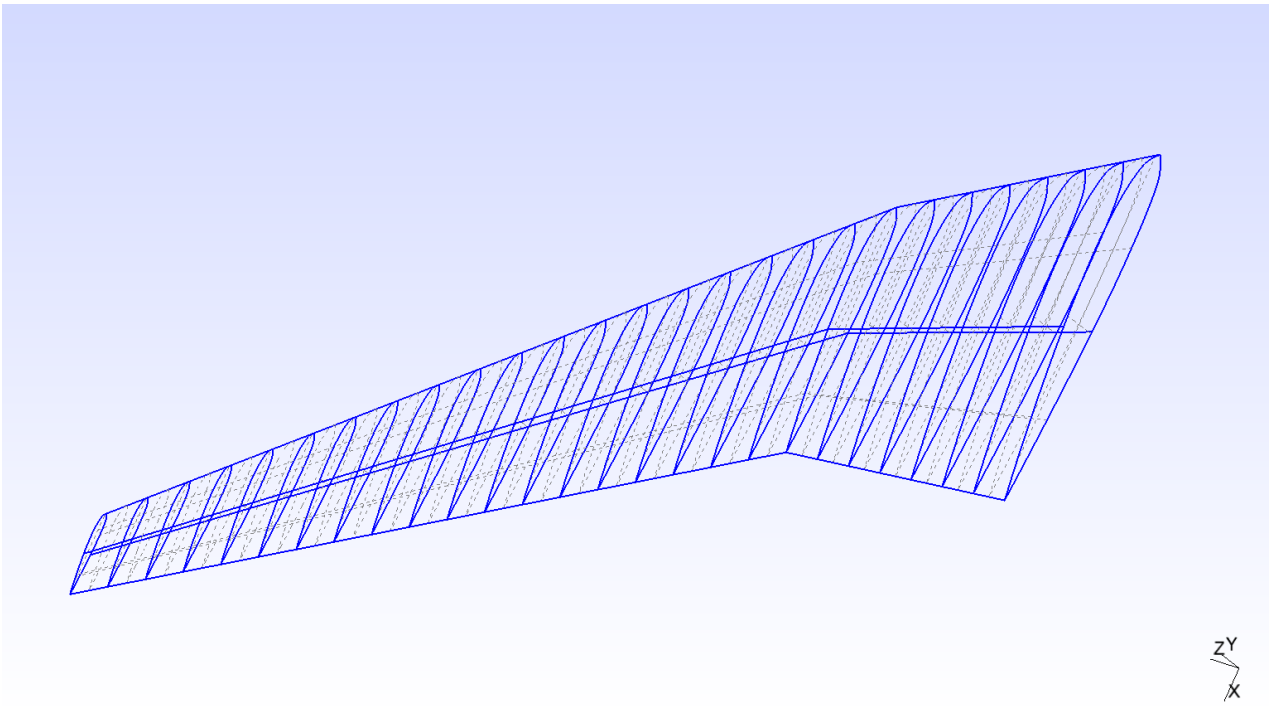

Figura 5.15. Modelo del ala de un avión Airbus A320.

Las superficies de los perfiles NACA y la envolvente, así como el volumen que encierran, se mallan con elementos estructurados, cuadrangulares en el caso de las superficies y hexaédricos para la malla del volumen, como se muestra en la Figura 5.16.

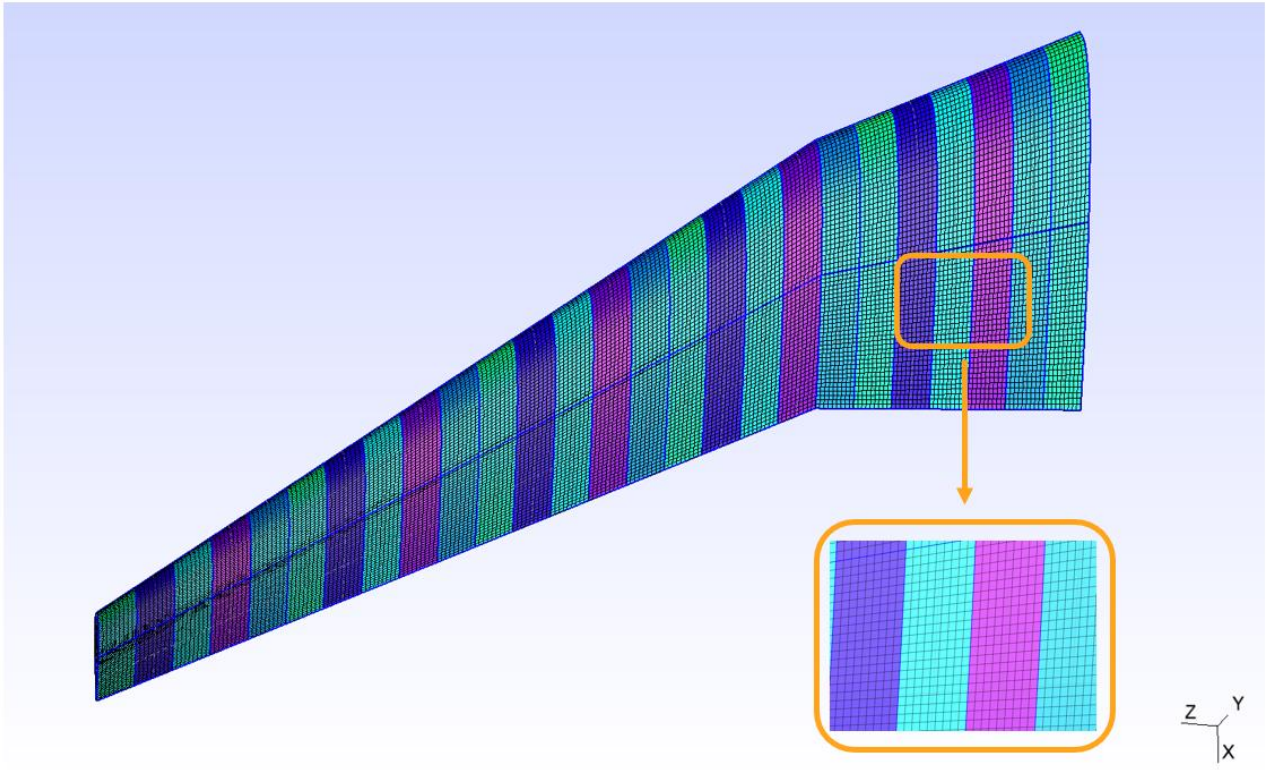

Figura 5.16. Malla del modelo del ala del Airbus A320. Detalle de los elementos de la malla.

Como se ha comentado en el apartado anterior, en este caso también se va a realizar una simplificación de las fuerzas que sufre el ala durante el vuelo.

La distribución de la carga de sustentación no es uniforme en toda la longitud del ala, sino que la sustentación es nula en la punta y máxima en la raíz, siguiendo un perfil de tipo elíptico (ver Figura 5.17). Para obtener la distribución exacta de presiones sobre el ala, sería necesario realizar un análisis por CFD, lo que excede el alcance de este proyecto.

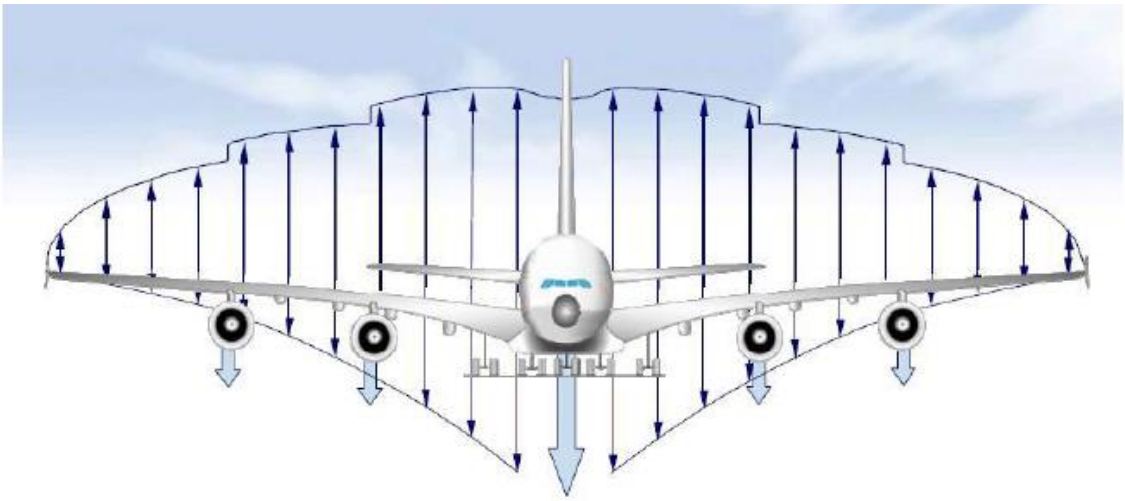

Figura 5.17. Distribución de la fuerza de sustentación en las alas de un avión. Recuperado de [29].

En nuestro caso, por simplicidad, se introducen cargas superficiales uniformes sobre el ala, tanto para la sustentación vertical como para la resistencia horizontal del aire, en la superficie superior y en la inferior del revestimiento. La sustentación vertical tiene un valor de 1 en la parte superior del ala y 0,3 en la parte inferior. En el caso de la resistencia horizontal al avance, se toma un valor de 0,1 en la dirección positiva del eje X tanto en la parte superior del ala como en la parte inferior. Las cargas introducidas en la parte superior del ala se muestran en la Figura 5.18.

En cuanto a las restricciones de movimiento, se restringen todos los desplazamientos en la superficie que corresponde con la costilla de la raíz, es decir, la que va unida al fuselaje.

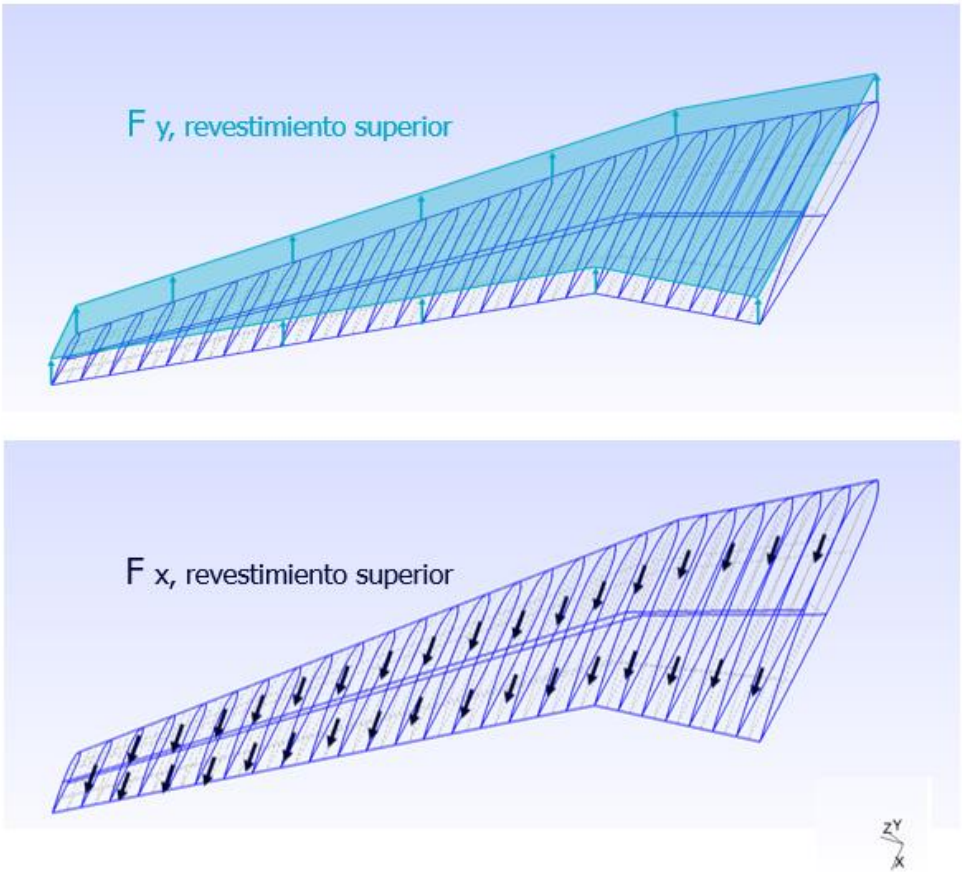

Figura 5.18. Cargas superficiales introducidas en el modelo del ala del Airbus A320.

5.1.3. Modelo de ala elíptica de un avión de combate histórico

Las alas con forma elíptica tienen una configuración más eficiente que las alas en flecha, ya que la forma elíptica permite minimizar la resistencia inducida (fuerza horizontal que se opone al avance y depende de la cantidad de sustentación generada). No obstante, este modelo de alas presenta como principal desventaja la dificultad para su fabricación [36]. Por este motivo, este tipo de alas no se ha generalizado en los aviones actuales y se encuentra principalmente en aviones de combate de la Segunda Guerra Mundial. Una de las aeronaves más populares con este tipo de alas es el Submarine Splitfire, un caza monoplaza británico, cuya configuración alar se puede ver en la Figura 5.19.

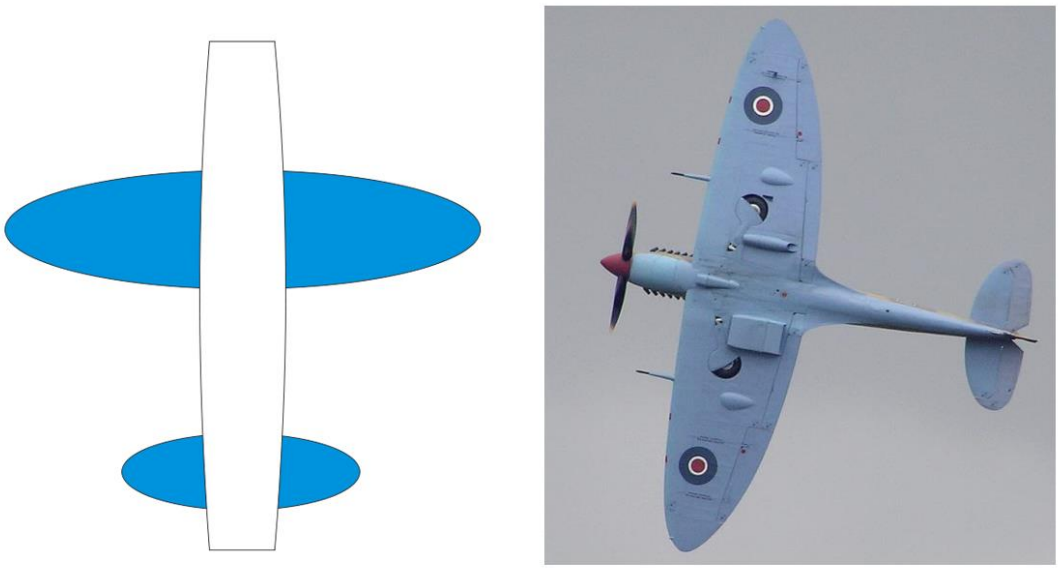

Figura 5.19. a) Configuración de un ala elíptica. b) Submarine Splitfire.

En la bibliografía se pueden encontrar las dimensiones de esta icónica aeronave. En base a lo encontrado en el documento [37], se modela el ala como una semielipse con semieje mayor de longitud 5,6 m y semieje menor de longitud 1,25 m. En la Figura 5.20 se presenta un esquema con las dimensiones principales y en la Figura 5.21 se muestra el modelo realizado con el software Gmsh. En el modelo se incluyen como costillas perfiles sustentadores del tipo NACA 2315, separados entre sí 0,2 m.

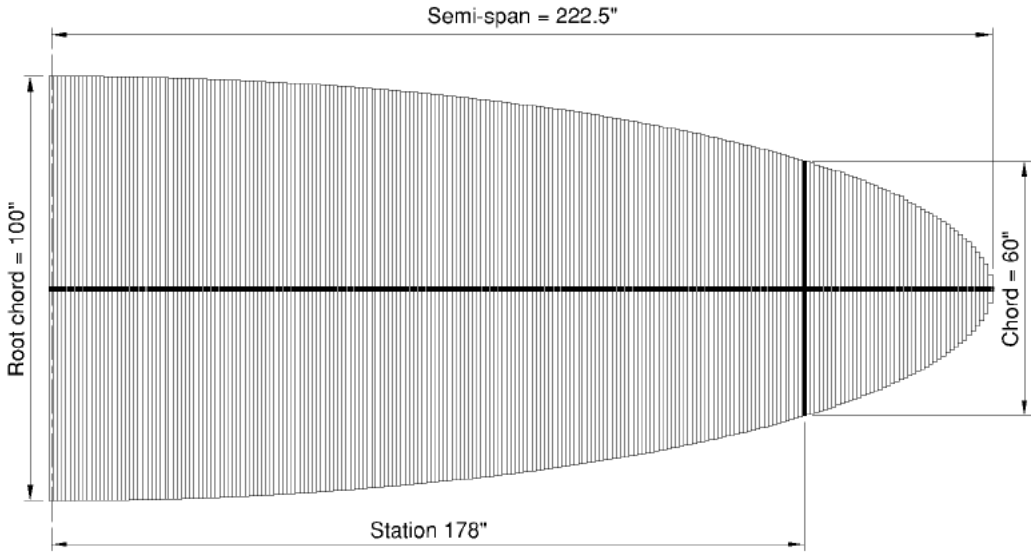

Figura 5.20. Dimensiones principales del ala elíptica. Recuperado de [37].

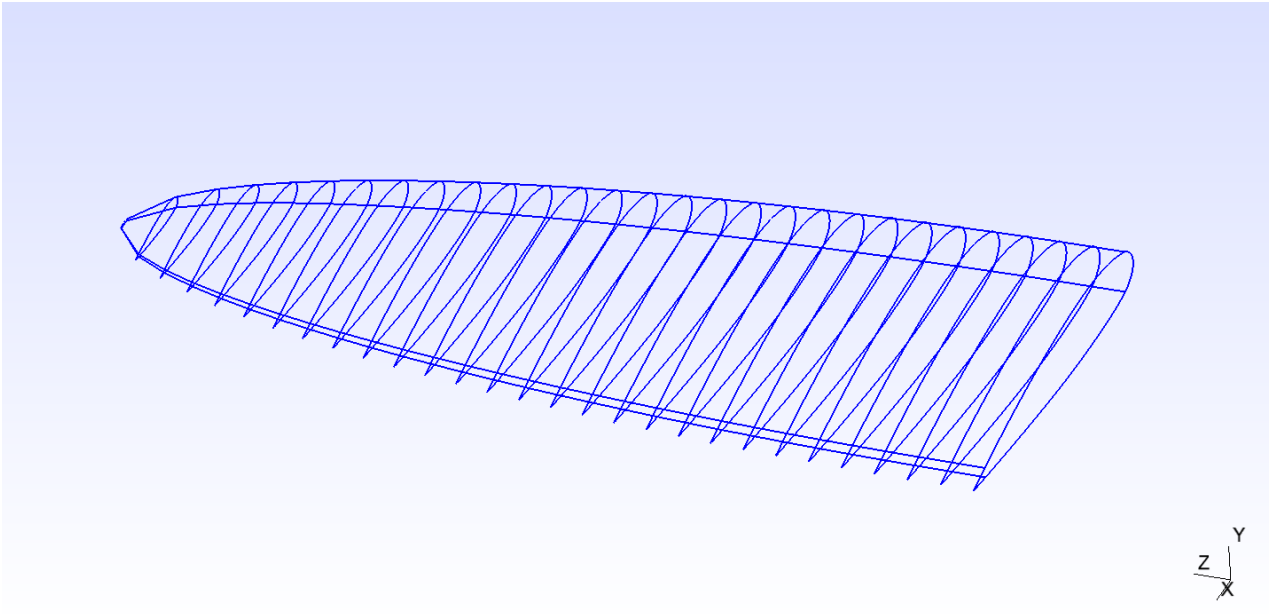

Figura 5.21. Modelo de ala elíptica.

Para realizar los cálculos mediante el método de elementos finitos, se genera una malla sobre las superficies de los perfiles NACA y la superficie de la envolvente. También se genera una malla en el volumen que encierran estas superficies. Se utiliza una malla estructurada, con elementos cuadrangulares sobre las superficies y elementos hexaédricos en la malla en 3D, como se puede observar en la Figura 5.22.

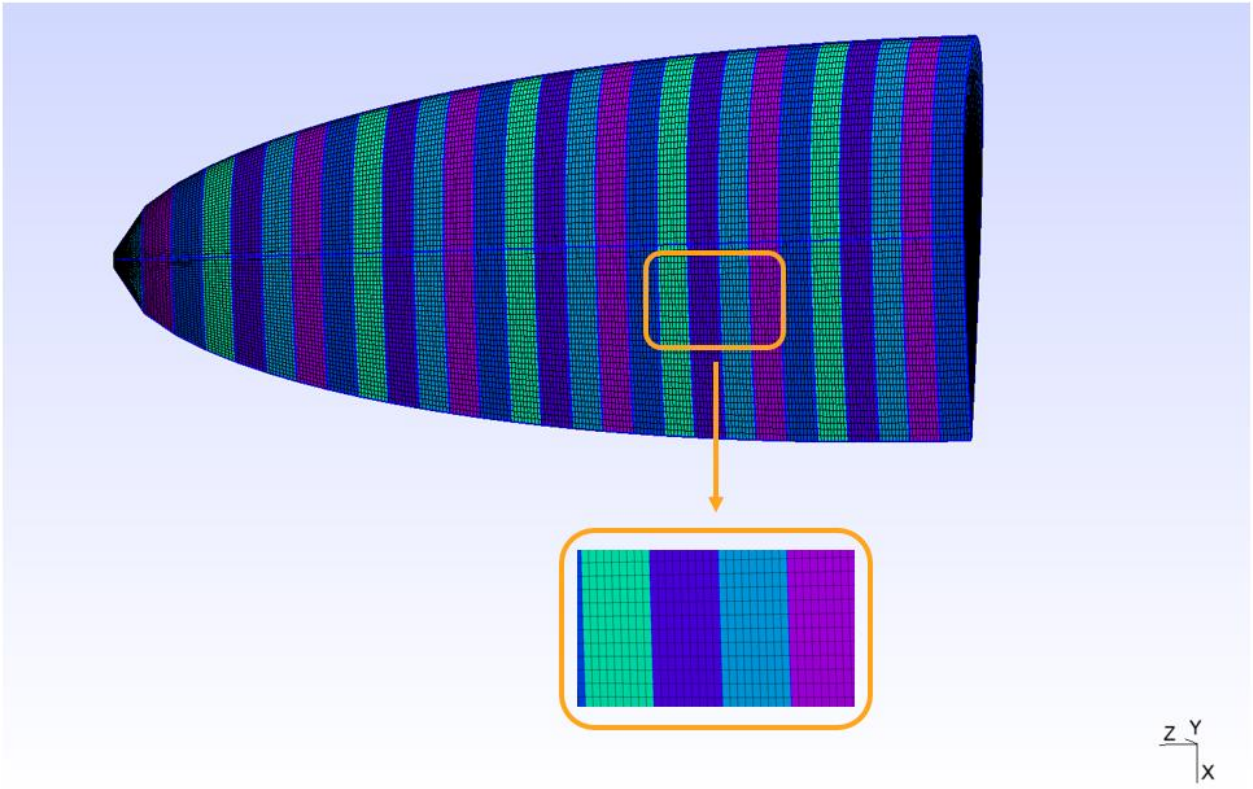

Figura 5.22. Malla 3D del modelo de ala elíptica. Detalle de los elementos de la malla.

Al igual que en el modelo del ala del avión Airbus A320, en este caso también se realiza una simplificación importante a la hora de introducir las cargas en el modelo. Como sustentación vertical en la dirección del eje Y, se incluye una fuerza superficial de valor 1 en la parte superior del revestimiento y una fuerza superficial de valor 0,3 en la parte inferior del revestimiento.

Como resistencia horizontal al avance, se incluye tanto en el revestimiento superior como en el inferior una fuerza superficial de valor 0,12 en la dirección del eje X.

Para completar las condiciones de contorno, se restringen todos los movimientos en la superficie de la costilla correspondiente a la raíz del ala ya que, al ir colocada en el avión, estaría fijada al fuselaje.

5.1.4. Modelo del ala de un avión ligero

Para completar los modelos a optimizar, se incluye el ala rectangular típica de los aviones ligeros o avionetas. Las alas rectangulares de sección transversal constante son las más fáciles de fabricar y económicas. Sin embargo, su eficiencia aerodinámica es baja, por lo que se suele usar en pequeñas aeronaves privadas, que vuelan a bajas velocidades (normalmente propulsadas por hélices).

Para el modelo de ala recta que se va a optimizar dentro de este proyecto, se toma como referencia el ala de la avioneta Piper PA-32 Cherokee Six, una aeronave ligera de tren fijo y seis o siete plazas, producidas en Estados Unidos entre 1965 y 2007. En la Figura 5.23 se recoge el esquema de una aeronave con ala rectangular, que se corresponde con el ala del modelo citado.

Figura 5.23. a) Configuración de un ala recta de cuerda constante. b) Avioneta Piper PA-32 Cherokee Six.

En la Figura 5.24 se observa el modelo del ala rectangular realizado con el software Gmsh. Como perfiles sustentadores, por simplicidad se decide colocar perfiles de tipo NACA 2315 con longitud de cuerda de 1,5 m constante a lo largo de toda el ala. Estos perfiles sustentadores se colocan cada 0,5 m, siendo la longitud total del ala de 5 m.

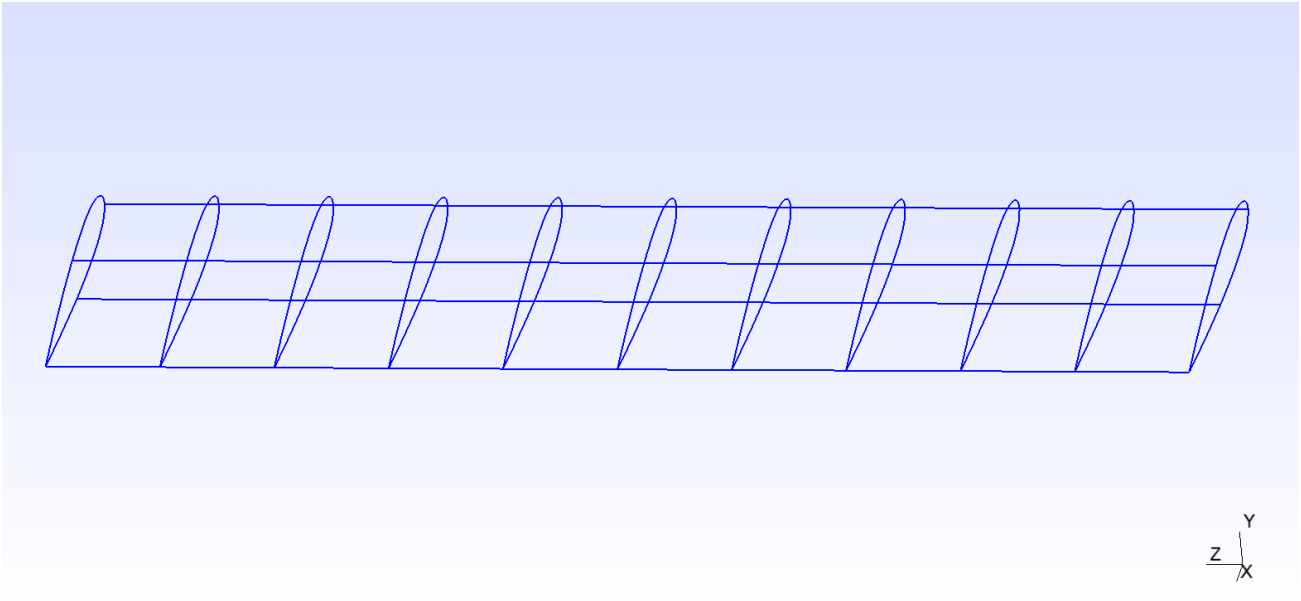

Figura 5.24. Modelo de ala rectangular.

En la Figura 5.25 se muestra la malla estructurada que se genera en Gmsh sobre las superficies y también sobre el volumen que encierra el ala.

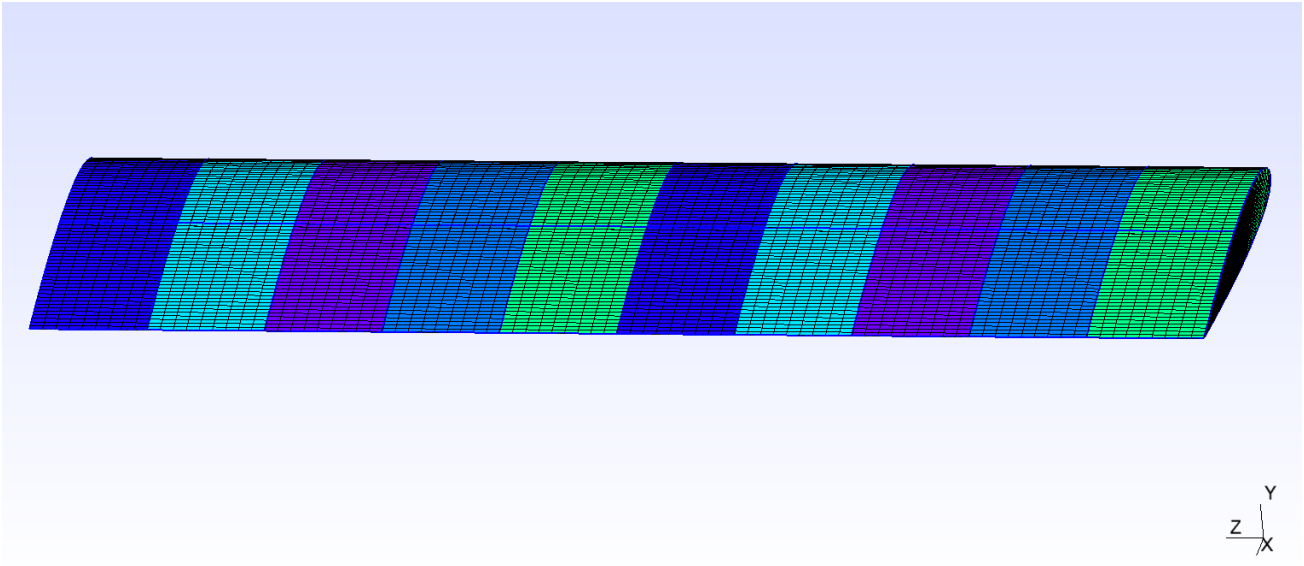

Figura 5.25. Malla 3D del ala rectangular de un avión ligero.

En cuanto a las cargas, se realiza la simplificación detallada para los modelos anteriores. En la dirección del eje Y, se incluye una fuerza superficial de valor 1 en la parte superior del revestimiento y una fuerza superficial de valor 0,6 en la parte inferior del revestimiento, que corresponde a la sustentación vertical en el ala. Como resistencia horizontal al avance, se incluye en el revestimiento superior una fuerza superficial de valor 0,12 en la dirección del eje X y en el revestimiento inferior una fuerza del mismo tipo de valor 0,20.

Además, se restringen todos los movimientos en la superficie de la costilla colocada en la raíz del ala.

5.2. Resultados de la optimización topológica

En este apartado se van a presentar los resultados de las optimizaciones topológicas de los modelos anteriormente expuestos. En todos los casos se van a presentar una serie de imágenes con la distribución óptima de material (lleno/vacío) obtenida mediante el método de las densidades paralelizado. Estas distribuciones óptimas de material constituyen un diseño preliminar, que puede servir de base para un futuro diseño modificado de cada uno de los elementos modelados.

Además, en los casos en los que sea factible y los tiempos computacionales no resulten demasiado elevados, se realizarán las optimizaciones con distinto número de procesadores trabajando en paralelo. De este modo se pretende obtener curvas de escalabilidad y aceleración, que permitan conocer el número óptimo de procesadores. En la Tabla 5.1 se presenta un resumen de los experimentos realizados dentro del presente proyecto.

Nº experimento	Modelo a optimizar	Nº procesadores	Tipo malla	Nº grados de libertad	Resultados
1.1	Perfil aerodinámico NACA 2315	1,6,12,24,36,48	Estructurada con elementos cuadrangulares	526338	Tabla 5.2, Tabla 5.3 Figura 5.28 Figura 5.29
1.2	Perfil aerodinámico NACA 2315	48	Estructurada con elementos cuadrangulares	2101250	Tabla 5.2 Figura 5.26 Figura 5.27
2.1	Ala del avión Airbus A320	48	Estructurada con elementos hexaédricos	4654971	Tabla 5.4
2.2	Ala del avión Airbus A320	48	Estructurada con elementos hexaédricos	36587859	Tabla 5.4 Figura 5.30 a Figura 5.33
3.1	Ala elíptica	48	Estructurada con elementos hexaédricos	8105133	Tabla 5.5 Figura 5.34 a Figura 5.39
4.1	Ala rectangular	48	Estructurada con elementos hexaédricos	5726883	Tabla 5.6 Figura 5.40 a Figura 5.43

Tabla 5.1. Relación de los experimentos realizados.

Para todas las optimizaciones se tienen como datos de entrada la malla de elementos finitos de cada modelo, así como las condiciones de contorno que se han detallado para cada uno de ellos. Además, en todos los casos se introducen una serie de parámetros requeridos por el algoritmo de optimización que maximiza la rigidez estructural. Estos parámetros son:

- Número máximo de iteraciones del método del gradiente conjugado (5000).

- Parámetros relativos al multigrid algebraico empleado como precondicionador: número de pasos de pre y post-suavizado.
- Parámetros de material: Se da un valor de módulo de Young y coeficiente de Poisson, requeridos por el algoritmo para los cálculos. Estos no corresponden a valores del material definitivo para el modelo, sino que sería necesario un análisis para la selección del material tras obtener el diseño conceptual.
- Fracción de volumen. Este parámetro se fija como condición a la optimización. Se busca una distribución óptima de material, de forma que el volumen esté por debajo de un cierto valor. La fracción de volumen (0,4) supone que se va a reducir el volumen de material un 60% respecto al volumen del dominio introducido.

5.2.1. Diseño conceptual del perfil de la costilla del ala. Curva de aceleración

Tal y como se recoge en la Tabla 5.1, se realizan dos experimentos diferentes para la optimización del perfil aerodinámico NACA 2315. El primero de ellos, con 526338 grados de libertad, permite que la optimización se realice en tiempos computacionales razonables con distinto número de procesadores trabajando en paralelo. Así, se lanzará este experimento con 1,6,12,24,36 y 48 procesadores en paralelo, obteniendo curvas de tiempos computacionales y aceleración en función del número de procesadores.

Adicionalmente, se realiza la optimización del perfil NACA 2315 con 2101250 grados de libertad. En este caso, los tiempos computacionales son mayores y no es factible realizar la optimización con un número bajo de procesadores en paralelo. Se calcula con 48 procesadores y se obtienen imágenes de la distribución óptima de material con mayor resolución que en el caso de 526338 grados de libertad. En la Tabla 5.2 se recogen los tiempos computacionales empleados para cada una de las optimizaciones realizadas, así como el número de iteraciones en el que se ha detenido el proceso. En todos los casos se fija una fracción de volumen de 0,4.

Nº experimento	**Nº elementos de la malla**	**Nº grados de libertad**	**Nº procesadores**	**Iteraciones**	**Tiempo (s)**
1.1	262144	526338	1	499	20539,4
1.1	262144	526338	6	499	3908,77
1.1	262144	526338	12	499	2244,93
1.1	262144	526338	24	499	1538,33
1.1	262144	526338	36	499	1669,75
1.1	262144	526338	48	499	1352,42
1.2	1048576	2101250	48	499	6560,81

Tabla 5.2. Iteraciones y tiempos computacionales obtenidos en la optimización del perfil NACA 2315.

Comenzando por la optimización realizada con 2101250 grados de libertad (“Experimento 1.2”) se van a presentar los resultados de la distribución de material sólido/vacío obtenida mediante el método basado en densidades paralelizado. En la Figura 5.26 se puede ver en color rojo las zonas donde se distribuye el material (densidad de valor 1) y en azul se muestran las zonas del dominio que no tendrían material (densidad cercana a 0). En la Figura 5.27 se muestra únicamente en rojo la parte que tendría material.

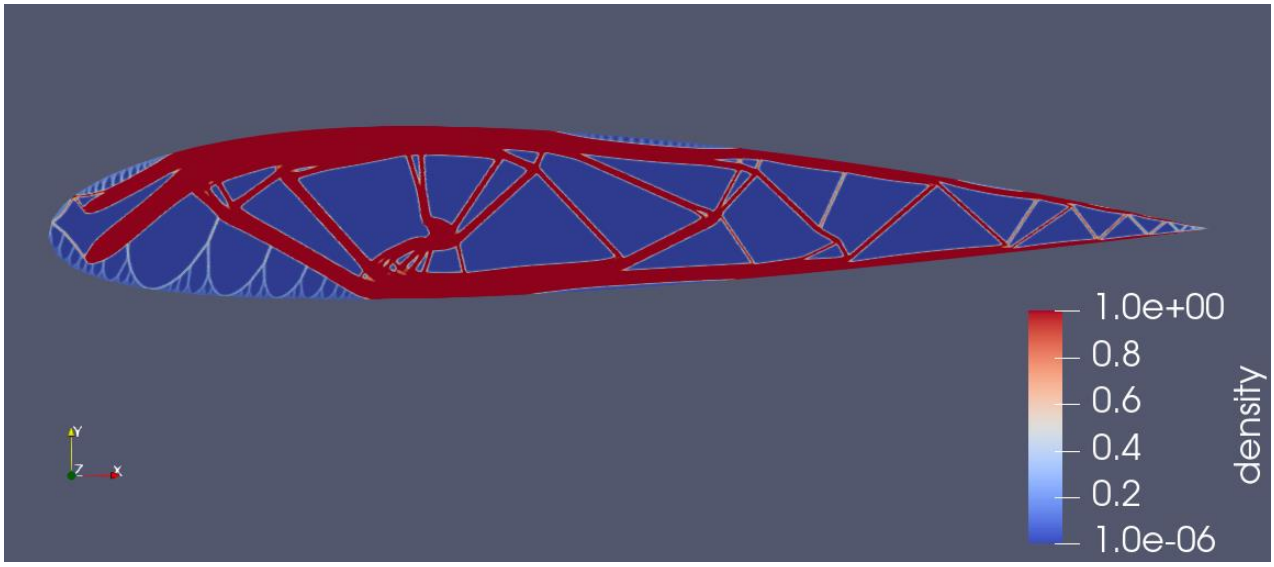

Figura 5.26. Distribución óptima de material para el perfil NACA 2315 (I).

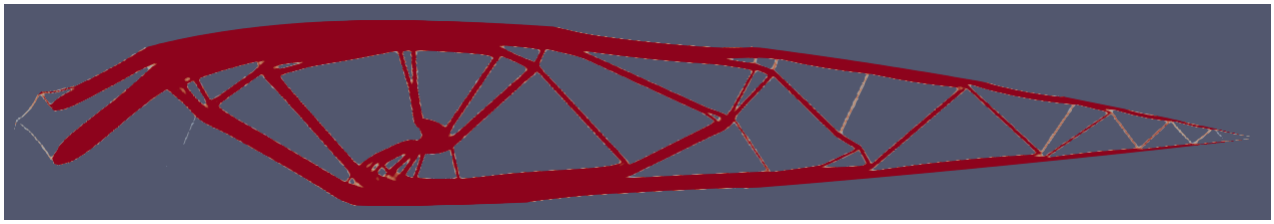

Figura 5.27. Distribución óptima de material para el perfil NACA 2315 (II). Zonas sólidas.

En estas imágenes se observa que, en la parte cercana al borde de salida, la distribución de material se asemeja a una cercha clásica tipo Warren. A medida que nos acercamos al borde de ataque, aumenta el espesor de material que se sitúa en la superficie del intradós y el extradós, ya que se reciben esfuerzos mayores. En la parte intermedia del perfil, aparecen configuraciones de complejidad mayor que una cercha clásica. En el borde de ataque, se observa que el material se sitúa en las proximidades del extradós, quedando vacío el intradós.

Es importante recalcar que la distribución de material constituye un diseño conceptual, enmarcado en las fases preliminares del desarrollo del producto. Este diseño presenta formas que resultarían complejas para la fabricación. Por esto, debe servir al diseñador como guía para desarrollar un diseño final, que será el que se fabrique.

En cuanto a la primera optimización (“Experimento 1.1”) con 526338 grados de libertad, no se va a presentar el diseño conceptual obtenido, ya que se obtienen los mismos resultados que para el caso con 2101250 grados de libertad, pero con una menor resolución. Sin embargo, se tienen tiempos computacionales considerablemente menores: en el caso de 48 procesadores en paralelo, se calcula en un tiempo de 1352,42 segundos (22,54 minutos) para el caso de 526338 grados de libertad y se necesitan 6560,81 segundos (1,82 horas) para el caso de 2101250 grados de libertad. Esta reducción

en los tiempos computacionales se aprovecha para realizar el experimento con distinto número de procesadores y comprobar cómo el tiempo requerido disminuye con el incremento de procesadores en paralelo (ver Figura 5.28).

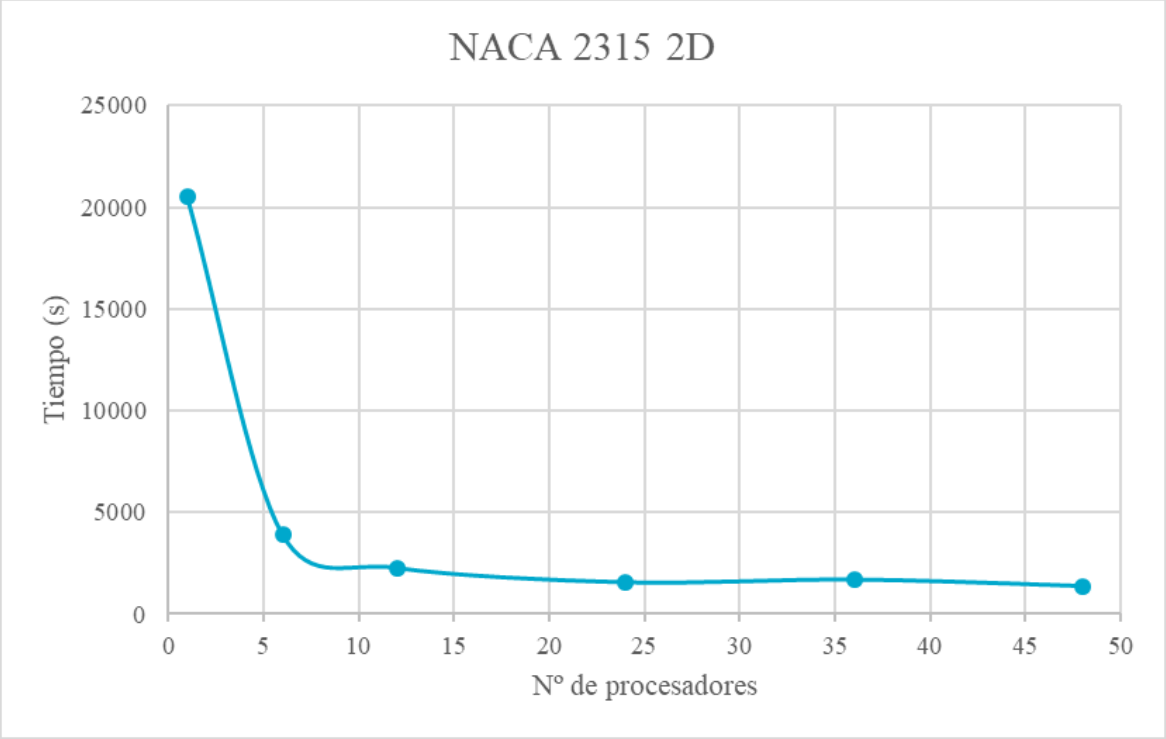

Figura 5.28. Tiempos computacionales frente al número de procesadores en la optimización del perfil NACA 2315.

Además, se calcula la aceleración (speed-up) obtenida con los distintos números de procesadores en paralelo. La aceleración se define como la relación entre el tiempo requerido para los cálculos con un procesador y el tiempo requerido para los cálculos con "n" procesadores en paralelo:

$$speed\ up = \frac{t_{1\ procesador}}{t_{n\ procesadores}} \tag{5. 8}$$

En la Tabla 5.3 se muestran los resultados de speed-up para cada optimización y en la Figura 5.29 se representa el valor de la aceleración frente al número de procesadores.

Nº experimento	**Nº procesadores**	**Tiempo (s)**	**Speed-up**
1.1	1	20539,4	1
1.1	6	3908,77	5,2547
1.1	12	2244,93	9,1492
1.1	24	1538,33	13,3518
1.1	36	1669,75	12,3009

Nº experimento	Nº procesadores	Tiempo (s)	Speed-up
1.1	48	1352,42	15,1871

Tabla 5.3. Resultados de aceleración para las optimizaciones del perfil NACA 2315.

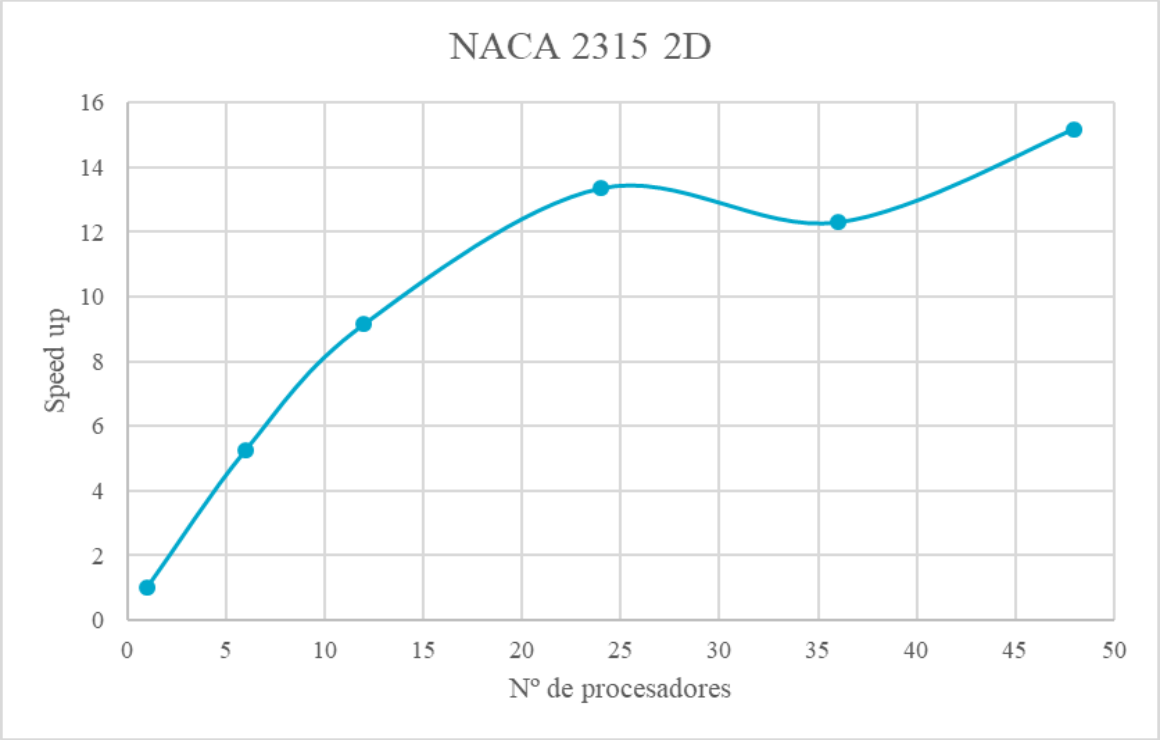

Figura 5.29. Speed-up frente al número de procesadores en la optimización del perfil NACA 2315.

A la vista de estos resultados se puede analizar la mejora que supone la incorporación de un determinado número de procesadores trabajando en paralelo. En la Figura 5.28 se observa una disminución muy importante de forma lineal entre el tiempo computacional requerido usando un único procesador y el requerido con 6 procesadores en paralelo. Al seguir aumentando el número de procesadores, los tiempos computacionales siguen reduciéndose, pero a un ritmo mucho más lento.

La Figura 5.29 permite comprobar cómo la aceleración va aumentando con el número de procesadores. Hasta 6 procesadores aumenta de forma más rápida y a partir de 12 procesadores trabajando en paralelo, la aceleración aumenta más lento al incluir nuevos procesadores. Para 36 procesadores se observa una disminución en la aceleración (esto es, un aumento en el tiempo computacional). Esto se debe a que, para utilizar más de 24 procesadores, es necesario incluir un nuevo nodo y las comunicaciones por red necesarias entre ambos nodos hacen que se degrade la aceleración del sistema. No obstante, en el caso en el que se utilizan 48 procesadores en paralelo sí se observa una mejora en el rendimiento frente al caso en el que se utilizan 24 procesadores.

5.2.2. Diseño conceptual del ala de un avión comercial Airbus A320

Para el caso del ala del avión Airbus A320, se realiza una optimización topológica con 4654971 grados de libertad. A la vista de que la resolución de la distribución de material resultante no era la deseada, se lanzó otra optimización con 36587859 grados de libertad, buscando obtener una mayor resolución en el diseño conceptual. En la Tabla 5.4 se recogen los tiempos computacionales obtenidos para cada uno de estos experimentos.

Nº experimento	Nº elementos de la malla	Nº grados de libertad	Nº procesadores	Iteraciones	Tiempo (s)
2.1	1497600	4654971	48	281	62179,8
2.2	11980800	36587859	48	131	286380

Tabla 5.4. Iteraciones y tiempos computacionales obtenidos en la optimización del ala del Airbus A320.

Estas optimizaciones se han realizado únicamente con 48 procesadores trabajando en paralelo. El gran tamaño de ambos problemas hace que se tengan tiempos computacionales muy elevados, de 62179,8 segundos (17,27 horas) para el experimento 2.1 y 286380 segundos (79,55 horas o 3,3 días) para el caso 2.2. Por este motivo, no resulta factible realizar las optimizaciones con diferente número de procesadores para obtener las curvas de aceleración y se opta por calcular ambos problemas con 48 procesadores en paralelo.

A continuación, se presenta la distribución de material sólido/vacío obtenida para la optimización topológica realizada con 36587859 grados de libertad (Figura 5.30 a Figura 5.33).

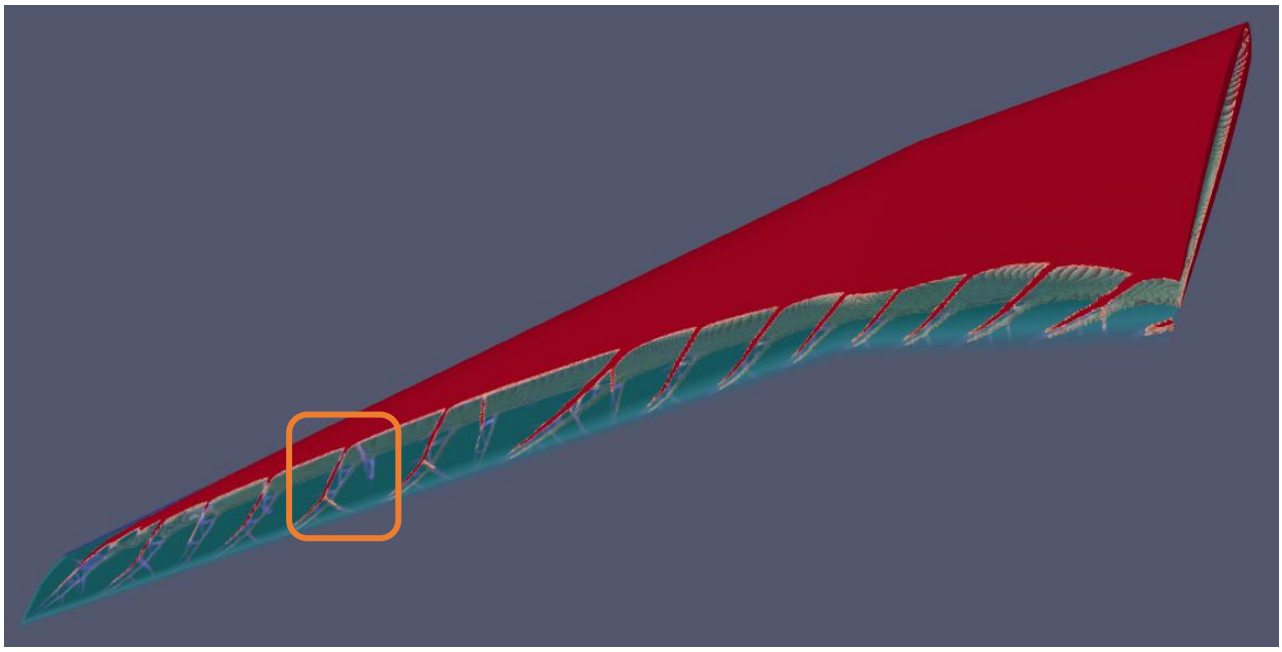

Figura 5.30. Distribución óptima de material para el ala del Airbus A320 (I).

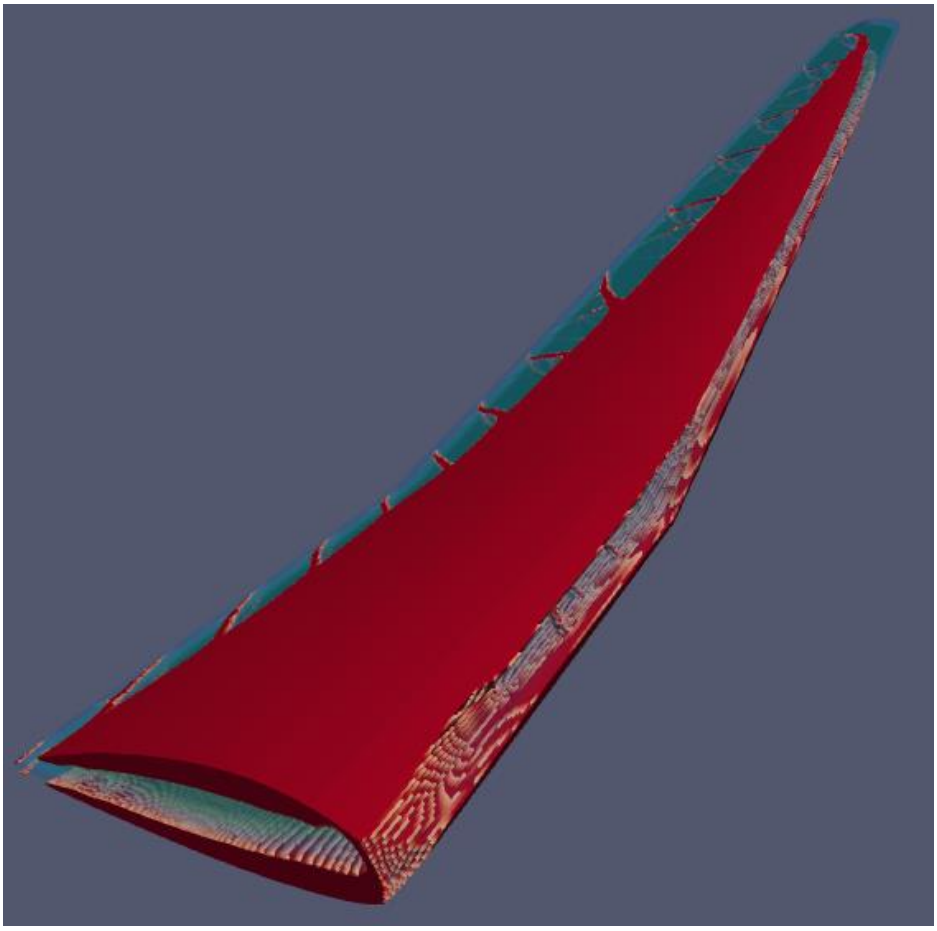

Figura 5.31. Distribución óptima de material para el ala del Airbus A320 (II).

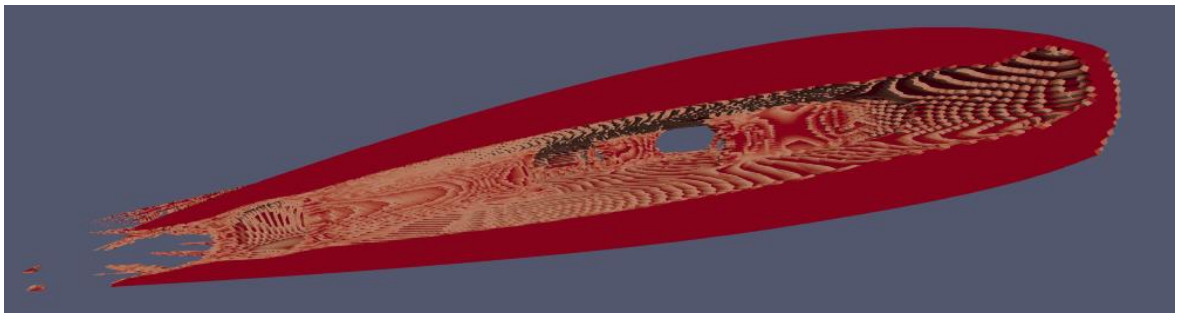

Figura 5.32. Distribución óptima de material para el ala del Airbus A320 (III). Vista desde la raíz del ala.

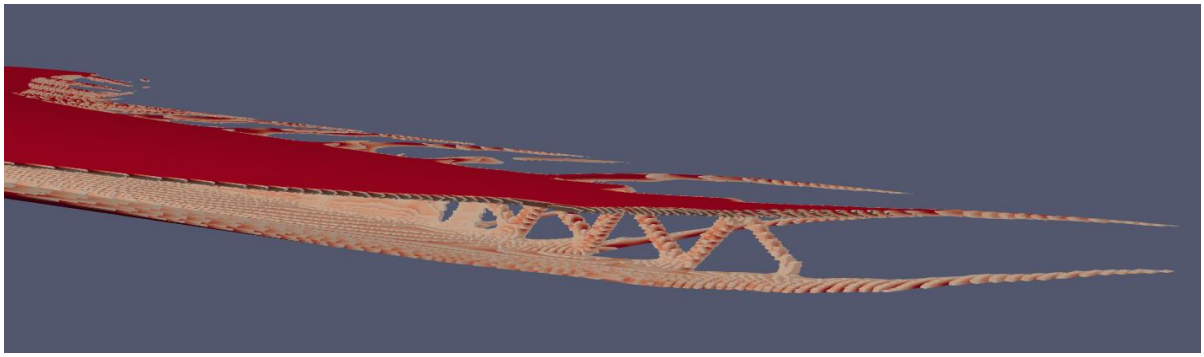

Figura 5.33. Distribución óptima de material para el ala del Airbus A320 (IV). Detalle de cercha. Vista desde la punta del ala.

En la Figura 5.30 se tiene una vista general del ala optimizada, donde se observa que en el diseño conceptual obtenido las costillas no son paralelas entre sí ni tampoco paralelas al fuselaje, sino que la orientación de las costillas va variando a lo largo del ala. De la misma manera, varía a lo largo del ala la separación entre costillas, que están más próximas entre sí en la parte cercana a la raíz del ala y más alejadas en la punta, donde los esfuerzos son menores.

También se observa cómo se distribuye el material en la superficie de recubrimiento, que se concentra en la zona cercana a la raíz, dejando la zona cercana a la punta al descubierto, lo que sugiere que se podría disminuir la sección en el extremo del ala.

Se aprecia además cómo el ala y las costillas no son macizas, sino que las costillas tienen en su plano forma de cercha (ver zona recuadrada en Figura 5.30). Aparecen estructuras de tipo cercha en la zona que une el revestimiento superior e inferior (ver Figura 5.33).

Para obtener imágenes de mayor resolución sería necesaria una malla con un número muy grande de elementos finitos, con lo que el tiempo de resolución del problema se incrementaría muy por encima de los 3,3 días requeridos en este caso. En todas las imágenes se observan formas que serían muy complejas de fabricar, por lo que sería necesaria una segunda etapa de diseño para adaptar este diseño conceptual a las posibilidades de la fabricación en taller.

5.2.3. Diseño conceptual del ala elíptica de un avión de combate

En este apartado se exponen los resultados obtenidos para la optimización topológica realizada sobre el ala elíptica inspirada en la del caza británico Submarine Splitfire.

A diferencia de las optimizaciones expuestas anteriormente, en este caso la fracción de volumen objetivo de la optimización no es 0,4. Se lanzan dos optimizaciones diferentes, variando la fracción de volumen: en la primera de ellas es de 0,5 (esto es, en el diseño conceptual obtenido el material ocupará un 50% del volumen del modelo introducido) y en la segunda la fracción de volumen es de 0,3 (el material ocupará un 30% del volumen de la malla introducida). En la Tabla 5.5 se presentan los tiempos computacionales requeridos e iteraciones realizadas para cada caso.

Nº experimento	**Nº elementos de la malla**	**Nº grados de libertad**	**Nº procesadores**	**Fracción de volumen**	**Iteraciones**	**Tiempo (s)**
3.1.1	15024832	8105133	48	0,5	287	29683,4
3.1.2	15024832	8105133	48	0,3	299	40359,4

Tabla 5.5. Iteraciones y tiempos computacionales obtenidos en la optimización del ala elíptica.

Se observa que para el caso donde la fracción de volumen es 0,3 (donde se reduce más el volumen de material respecto al modelo introducido) el tiempo computacional requerido es mayor. Al igual que en el caso anterior, debido a que los tiempos computacionales son elevados, no se van a obtener gráficas de aceleración con distinto número de procesadores en paralelo.

A continuación, se incluyen una serie de imágenes con la distribución óptima de material obtenida, tanto para el caso donde el volumen final es un 50% del original del modelo, como para el caso en el que se fija un 30% del volumen original.

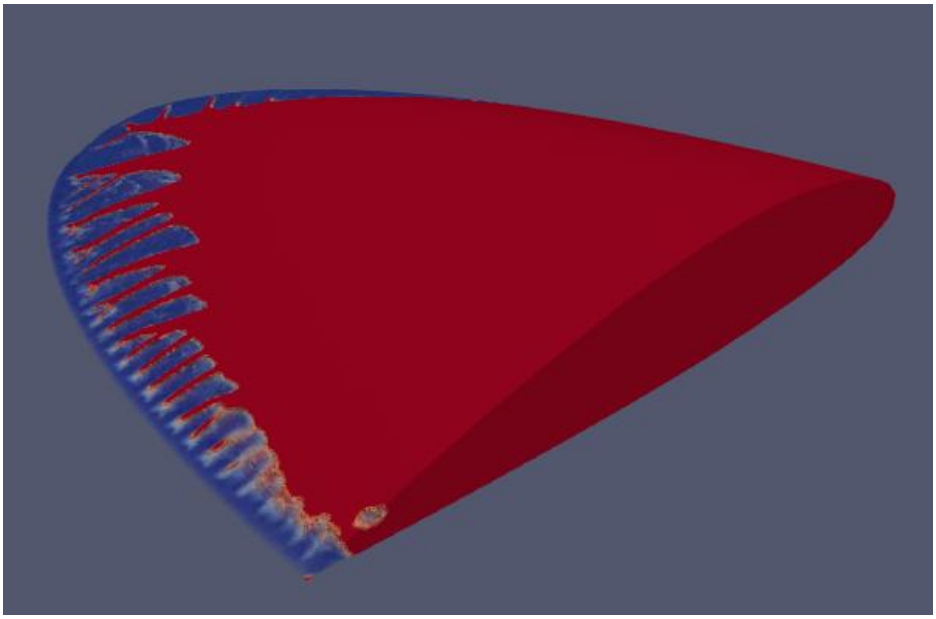

Figura 5.34. Distribución óptima de material para el ala elíptica, con fracción de volumen 0,5(I).

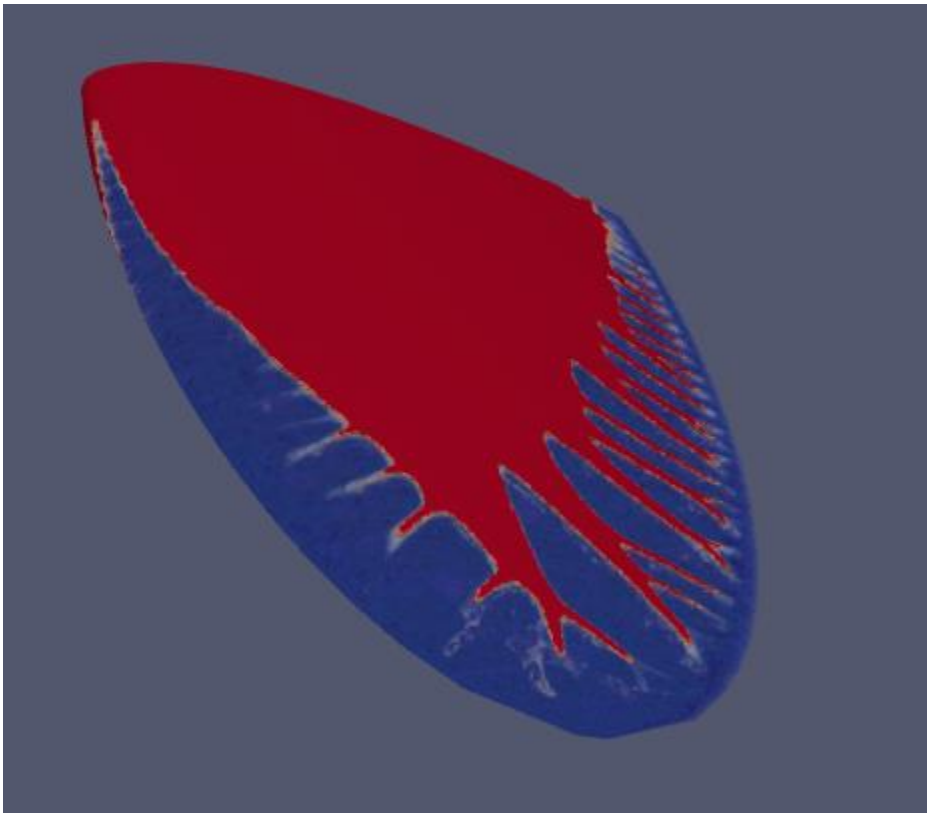

Figura 5.35. Distribución óptima de material para el ala elíptica, con fracción de volumen 0,5(II).

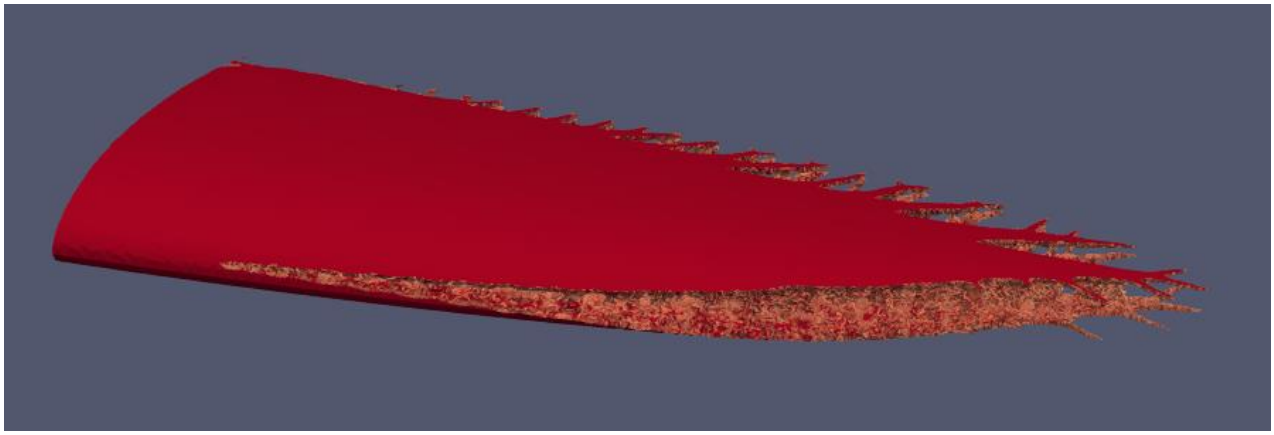

Figura 5.36. Distribución óptima de material para el ala elíptica, con fracción de volumen 0,5(III).

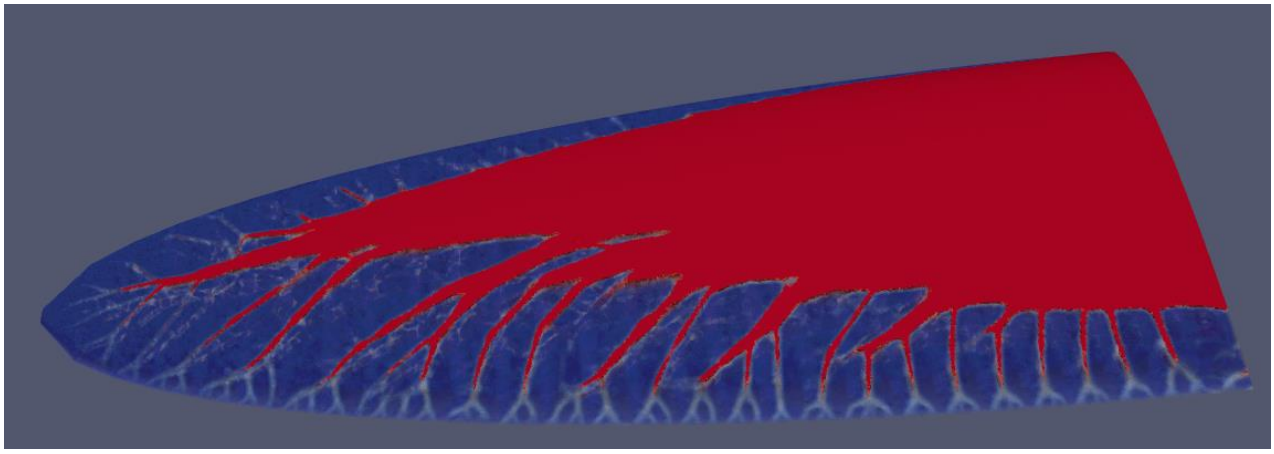

Figura 5.37. Distribución óptima de material para el ala elíptica, con fracción de volumen 0,3(I).

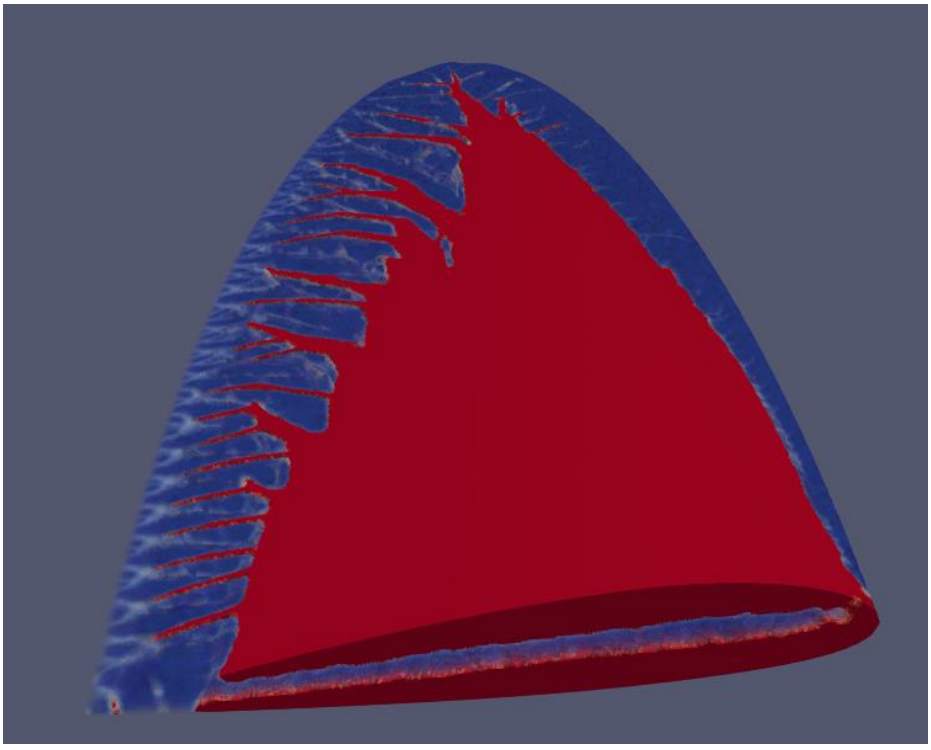

Figura 5.38. Distribución óptima de material para el ala elíptica, con fracción de volumen 0,3(II).

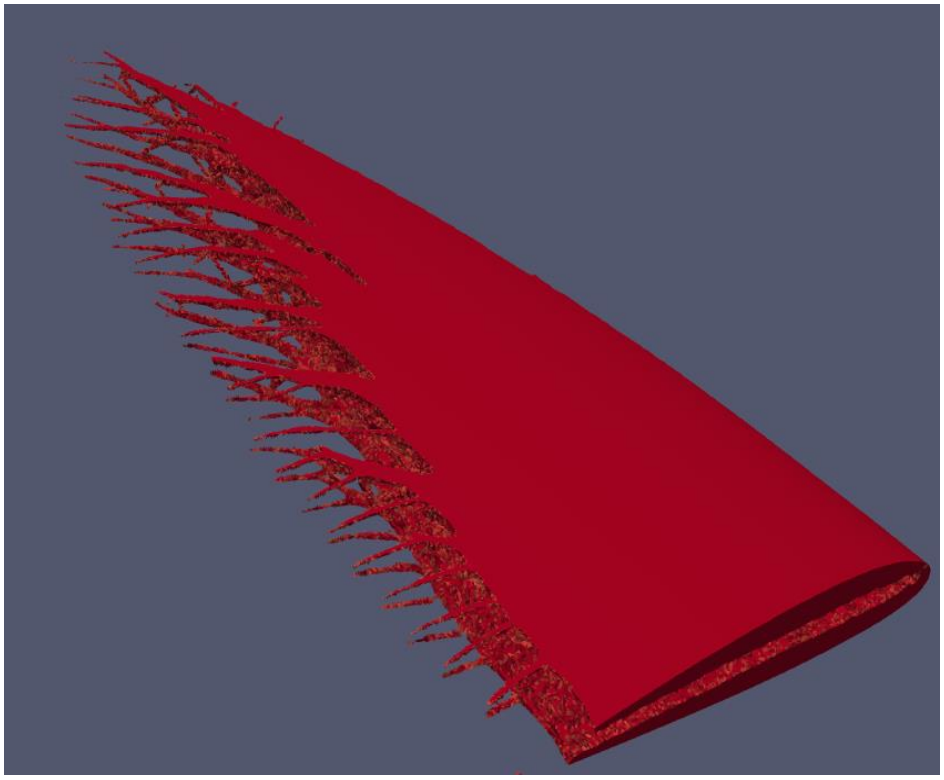

Figura 5.39. Distribución óptima de material para el ala elíptica, con fracción de volumen 0,3(III).

De la Figura 5.34 a la Figura 5.36 se recoge la distribución de material obtenida con fracción de volumen de valor 0,5. Se observa que la sección de la raíz del ala está llena, mientras que a medida que nos alejamos de la raíz empiezan a aparecer secciones huecas. A diferencia del modelo realizado en Gmsh, la distribución de material obtenida indica que las costillas no serían paralelas al fuselaje y equidistantes entre sí. La Figura 5.34 sugiere que, en el modelo optimizado, las costillas más cercanas a la raíz serían paralelas al fuselaje y posteriormente irían formando un cierto ángulo hasta resultar casi perpendiculares al fuselaje. También se observa como el ancho del ala va disminuyendo hacia la punta, siendo en el extremo considerablemente menor al del ala elíptica.

De la Figura 5.37 a la Figura 5.39 se muestra la distribución de material obtenida con fracción de volumen de valor 0,3 (esto es, reduciendo el volumen un 70% respecto al modelo original). Se obtienen formas muy similares a las de fracción de volumen 0,5. Al reducirse aún más la cantidad de material final del ala, se observa que la sección del ala en la raíz aparece hueca. También cabe destacar que gran parte del material se concentra en el borde de ataque del ala, quedando en el borde de salida ramificaciones finas de material.

5.2.4. Diseño conceptual del ala rectangular de un avión ligero

En este apartado se muestran los resultados de la optimización topológica del ala rectangular de cuerda constante de una avioneta.

En esta ocasión, se decide que la fracción de volumen sea de 0,5 (esto es, reducir el volumen un 50% respecto al original). Se resuelve el problema con 5726883 grados de libertad. En la Tabla 5.6 se recoge el número de iteraciones realizadas y el tiempo computacional requerido para resolver la optimización topológica del ala rectangular.

Nº experimento	Nº elementos de la malla	Nº grados de libertad	Nº procesadores	Fracción de volumen	Iteraciones	Tiempo (s)
4.1	1858560	5726883	48	0,5	151	23262,7

Tabla 5.6. Iteraciones y tiempos computacionales obtenidos en la optimización del ala rectangular.

En este caso también nos encontramos ante un problema de gran tamaño que requiere un tiempo computacional elevado (23262,7 segundos o 6,46 horas), incluso abordándolo con 48 procesadores en paralelo. Por este motivo, no resulta factible realizar la optimización con un número de procesadores más pequeño para obtener la curva de aceleración.

De la Figura 5.40 a la Figura 5.43 se presentan varias perspectivas de la distribución óptima de material obtenida. En dichas figuras aparecen rasgos comunes a los encontrados para las optimizaciones de las alas elíptica y en flecha. Se observa cómo las costillas aparecen distribuidas formando un cierto ángulo con el fuselaje, que varía a lo largo del ala, frente a la configuración tradicional de costillas paralelas al fuselaje.

En la Figura 5.41 y la Figura 5.43 se puede observar cómo en las costillas más cercanas a la punta del ala, aparecen estructuras que se asemejan a cerchas. Además, el material está distribuido de forma que aparece un larguero central sólido y otro larguero cercano al borde de ataque de los perfiles (a la izquierda en la Figura 5.43) que no es completamente macizo, sino que podría asemejarse a una cercha en algunas zonas.

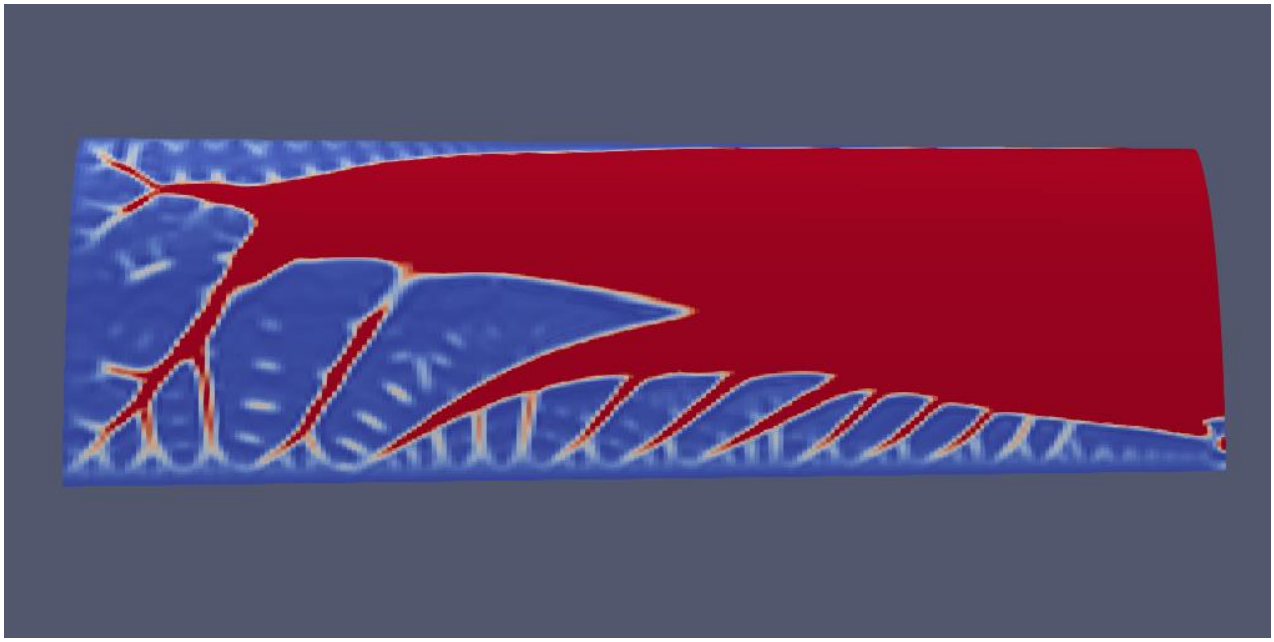

Figura 5.40. Distribución óptima de material para el ala rectangular (I).

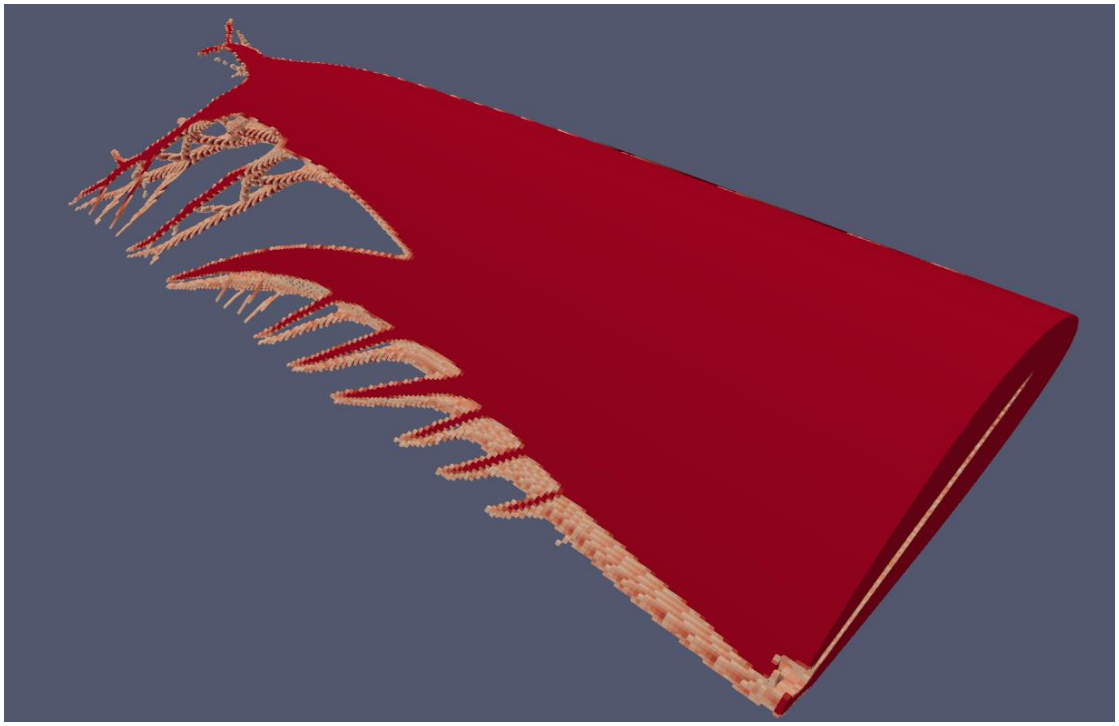

Figura 5.41. Distribución óptima de material para el ala rectangular (II).

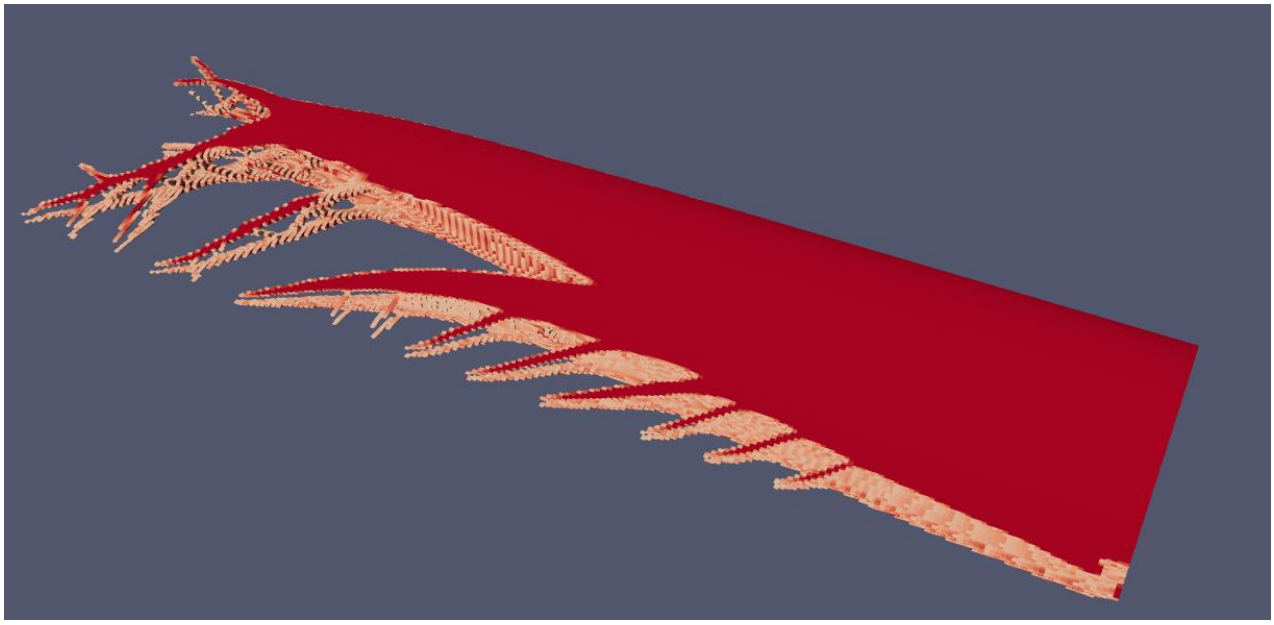

Figura 5.42. Distribución óptima de material para el ala rectangular (III).

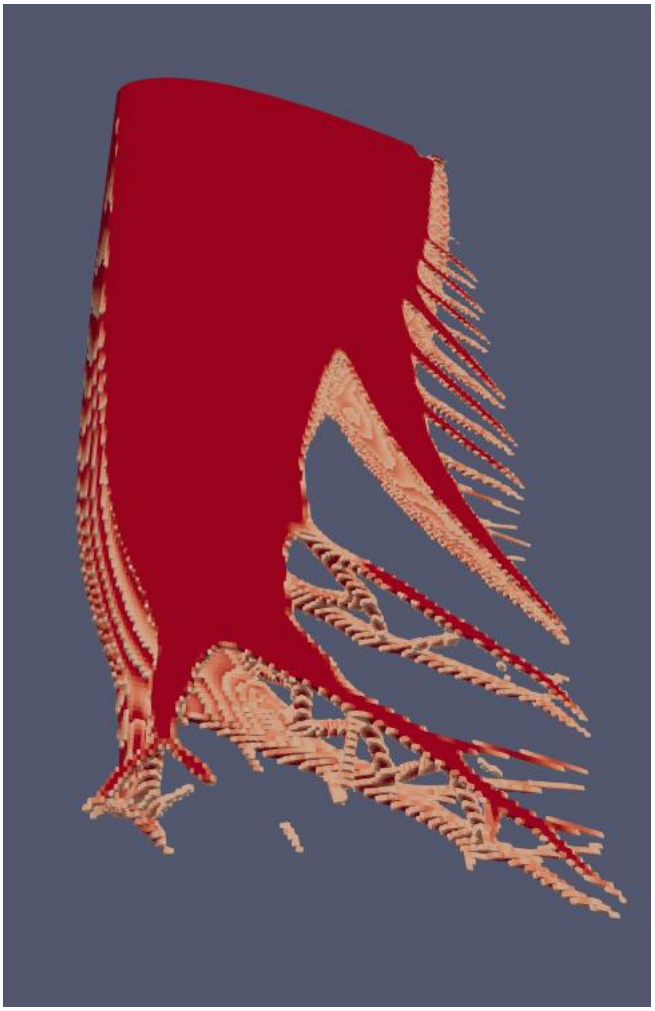

Figura 5.43. Distribución óptima de material para el ala rectangular (IV).

6. CONCLUSIONES Y TRABAJOS FUTUROS

En el desarrollo de este trabajo se ha llevado a cabo un estudio de diferentes técnicas y métodos de resolución que permiten abordar con un enfoque de computación en paralelo el problema de optimización topológica. Además, se han aplicado estas técnicas a un problema real de ingeniería: la optimización topológica de estructuras de aeronaves (en particular, la optimización de un perfil sustentador en 2D y de los modelos en 3D de las alas de un avión comercial, de un avión de combate histórico y un avión ligero). Tras realizar estas optimizaciones con el clúster de computación del grupo MC3 de la UPCT, se han obtenido una serie de conclusiones que se exponen a continuación:

- Los problemas de optimización topológica de estructuras reales requieren tiempos computacionales muy elevados para su resolución, ya que son problemas de gran tamaño y con un elevado número de grados de libertad, que además se deben resolver de forma recursiva durante la optimización, lo que hace que sean inabordables mediante la computación en serie y únicamente sea posible su resolución mediante un enfoque de computación en paralelo. Esto se ha comprobado mediante la optimización de los modelos en tres dimensiones de los tres tipos de alas (en flecha, elíptica y rectangular). Se han obtenido tiempos computacionales del orden de días (siendo 3,3 días el tiempo requerido en el caso del ala en flecha), si bien todas las optimizaciones de las alas se han realizado con 48 procesadores trabajando en paralelo. Esto da una idea de los tiempos computacionales que se necesitarían para abordar el problema con un único procesador en serie, demostrando que la computación en paralelo es la única alternativa que hace factible abordar problemas de optimización con una elevada fidelidad en ingeniería.
- Debido a los elevados tiempos computacionales requeridos para las optimizaciones realizadas, solo ha sido posible obtener la curva de tiempos computacionales y aceleraciones para el modelo del perfil sustentador NACA en 2D. Al optimizar el perfil NACA usando distinto número de procesadores en paralelo (entre 1 y 48), se ha comprobado cómo al aumentar el número de procesadores se reduce el tiempo requerido para llegar a la solución y se incrementa la aceleración. No obstante, la aceleración tiene un incremento considerablemente más rápido (y lineal) hasta llegar a 6 procesadores en paralelo. A partir de 6 procesadores, la aceleración continúa aumentando al incluir más procesadores, pero lo hace a un ritmo más lento. Incluso se observa una degradación de la aceleración al pasar de 24 a 36 procesadores en paralelo, explicada por la necesidad de incluir un nuevo nodo de computación. Al realizarse tareas de comunicación entre los nodos, se incrementan los tiempos computacionales.
- Los experimentos realizados también demuestran que las técnicas de resolución en paralelo utilizadas (particionamiento de grafos jerárquico y resolución del problema de elasticidad lineal mediante gradiente conjugado precondicionado con multigrid algebraico) y el método de optimización SIMP son aplicables a problemas reales de ingeniería y permiten obtener diseños conceptuales.
- En cuanto a las distribuciones de material obtenidas tras realizar las optimizaciones, cabe destacar que estas constituyen diseños conceptuales. Ya que aparecen formas con alta complejidad constructiva, sería necesaria una segunda fase para obtener el diseño propuesto para la fabricación. No obstante, en los diseños conceptuales de los tres tipos de alas, se observan una serie de características comunes que muestran las variaciones del diseño obtenido tras el proceso de optimización respecto a los diseños tradicionales. En todos los casos, los diseños conceptuales sugieren una colocación de las costillas formando un ángulo con el fuselaje que es

variable a lo largo del ala. Además, estas costillas no aparecen equidistantes entre sí y no son macizas, sino que se puede observar que en su plano tienen formas similares a las de una cercha clásica. Al adentrarnos en las cavidades del ala, también se observa cómo se forman largueros que unen el revestimiento superior e inferior con formas similares a una cercha. En cuanto a la forma del ala, se aprecia que, en el caso de las del avión histórico de combate y la avioneta, las distribuciones de material obtenidas presentan claras diferencias con las formas elíptica y rectangular introducidas en el modelo. En el caso del ala en flecha, se observa una mayor coincidencia entre la forma tradicional del ala y el diseño conceptual obtenido. No obstante, en todos los casos, se tienen diseños distintos a los diseños tradicionales y que en muchos casos recuerdan a las formas de las alas de algunas aves.

Una vez concluido esto, se proponen una serie de trabajos futuros que permitirían complementar y mejorar los resultados del presente proyecto:

- Como se ha comentado al exponer los modelos del perfil NACA y de las alas, en este proyecto se ha realizado una simplificación importante de las cargas que actúan sobre las alas del avión. Se plantea como trabajo futuro la obtención del perfil real de carga sobre el ala, para así poder realizar la optimización con las cargas reales actuantes bajo unas determinadas condiciones de vuelo.
- Ya que los diseños obtenidos mediante la optimización topológica son diseños conceptuales que presentan formas complejas de fabricar, se plantea continuar con el proceso de diseño y obtener, siguiendo el diseño conceptual, un segundo modelo orientado a la fabricación. Con este fin, sería necesario estudiar en profundidad el diseño conceptual obtenido, para lo que se podrían emplear técnicas de fabricación aditiva que permitan visualizar con exactitud la distribución del material. Posteriormente, sería necesario evaluar las diferentes técnicas de fabricación de las que se dispone, a fin de obtener un diseño basado en el conceptual cuya fabricación sea viable.
- Una vez que se haya pasado del diseño conceptual a otro adaptado para la fabricación, se plantea realizar una selección del material definitivo. Para esto, sería necesario analizar el modelo con las cargas que tenga que soportar y el material escogido, contemplando los coeficientes de seguridad que proponga la normativa aeronáutica vigente.
- Por último, se propone realizar las optimizaciones con equipos más potentes, que permitan realizar experimentos con un mayor número de grados de libertad y así poder obtener una mayor resolución en las distribuciones óptimas de material.

BIBLIOGRAFÍA

[1] S. Venkataraman y R. Haftka, «Structural optimizacion complexity: What has Moore's Law done for us?,» *Struct. Multidiscip. Optim,* nº 28, pp. 375-387, 2004.

[2] J. Martínez-Frutos, P. J. Martínez-Castejón y D. Herrero-Pérez, «Fine-grained GPU implementation of assembly-free iterative solver for finite elements problems,» *Computers and Structures,* nº 157, pp. 9-18, 2015.

[3] W. A. Wulf y S. A. McKee, «Hitting the Memory Wall: Implications of the Obvious,» 1994.

[4] K. Asanovic, R. Bodik, J. Demmel, T. Keaveny, K. Keutzer, J. Kubiatowicz y D. Wessel, «A View of the Parallel Computing Landscape,» *Communications of the ACM,* nº 52, pp. 56-67, 2009.

[5] K. S. Tiwari, «ResearchGate,» [En línea]. Available: https://www.researchgate.net/figure/Moores-Law-The-number-of-transistors-and-power-consumption-is-constantly-increasing_fig1_266083753. [Último acceso: 30 Marzo 2020].

[6] A. Lecina y C. Petiau, «Advances in optimal design with composite materials. Computer Aided Optimal Design: Structural and Mechanical Systems,» de *NATO ASI Series, Vol F27*, Springer, 1987, pp. 943-953.

[7] D. Jegley, H. Bush y A. Lovejoy, «Structural Response and Failure of a Full Scale Stiched Graphite Epoxy Wing,» de *AIAA Paper No. 2001-1334 Proceedings of the 42nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, Seatlle, Washington, 2001.

[8] P. S. Pacheco, «An Introduction to Parallel Programming,» 2011.

[9] F. Acosta, O. Segura y A. Ospina, «Guía y fundamentos de la programación en paralelo,» 2012. [En línea]. Available: https://revistas.upb.edu.co/index.php/telecomunicaciones/article/view/3306/2907. [Último acceso: Abril 2020].

[10] OpenMP ARB, [En línea]. Available: https://www.openmp.org/about/openmp-faq/. [Último acceso: 1 Abril 2020].

[11] R. Rivas, «Programación con OpenMP. Universidad Central de Venezuela,» 2011. [En línea]. Available: http://www.sc-camp.org/2011/_pdf/OpenMP%20-%20sccamp%202011.pdf. [Último acceso: 1 Abril 2020].

[12] «MPI forum,» [En línea]. Available: https://www.mpi-forum.org/. [Último acceso: 3 Abril 2020].

[13] «Introducción a MPI,» [En línea]. Available: http://informatica.uv.es/iiguia/ALP/materiales2005/2_2_introMPI.htm. [Último acceso: 1 Abril 2020].

[14] Anónimo, «Computación distribuida,» [En línea]. Available: http://bibing.us.es/proyectos/abreproy/11374/fichero/MEMORIA%252F02-COMPUTACION_DISTRIBUIDA.pdf. [Último acceso: 5 Abril 2020].

[15] J. Ruge y K. Stüben, «Algebraic Multrigrid,» de *Multi-grid Methods*, University City Science Center, Philadelphia, Pennsylvania, R.Ewing, 1987, pp. 73-130.

[16] P. Martí, *Apuntes de Elasticidad y Resistencia de Materiales. Introducción a la Teoría de la Elasticidad.*

[17] G. Karypis, «METIS. A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes and Computing Fill-Reducing Orderings of Sparse Matrices,» 2013. [En línea]. Available: http://glaros.dtc.umn.edu/gkhome/fetch/sw/metis/manual.pdf. [Último acceso: 10 Abril 2020].

[18] Lawrence Livermore National Laboratory, «Hypre User's Manual,» 2017. [En línea]. Available: https://computing.llnl.gov/sites/default/files/public/hypre-2.11.2_usr_manual.pdf. [Último acceso: 15 Abril 2020].

[19] J. R. Shewchuk, *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*, School of Computer Science. Carnegie Mellon University, 1994.

[20] M. Vargas-Félix, «Gradiente conjugado,» 2015. [En línea]. Available: http://personal.cimat.mx:8181/~miguelvargas/Course%20notes/Gradiente%20conjugado.pdf. [Último acceso: 10 Abril 2020].

[21] U. Kindelán, *Resolución de sistemas lineales de ecuaciones: Método del gradiente conjugado*, Escuela Técnica Superior de Ingenieros de Minas. Universidad Politécnica de Madrid.

[22] A. H. Baker, R. D. Falgout, T. V. Kolev y U. Meier Yang, «Scaling Hypre's Multigrid Solvers to 100,000 Cores,» 2012. [En línea]. Available: https://www.researchgate.net/publication/278660782_Scaling_Hypre's_Multigrid_Solvers_to_100000_Cores. [Último acceso: 15 Abril 2020].

[23] R. D. Falgout, «Scalable Multigrid Methods. Lawrence Livermore National Laboratory,» 2011. [En línea]. Available: http://www.int.washington.edu/talks/WorkShops/int_11_2a/People/Falgout_R/Falgout.pdf. [Último acceso: 15 Abril 2020].

[24] A. H. Baker, M. Schulz y U. M. Yang, «On the Performance of an Algebraic Multigrid Solver on Multicore Clusters,» 2011. [En línea]. Available: https://computing.llnl.gov/sites/default/files/public/Baker-2011-VECPAR.pdf. [Último acceso: 15 Abril 2020].

[25] D. Ginestar, «Matrices dispersas. Deparatamento de Matemática Aplicada. Universidad Politécnica de Valencia,» 2019. [En línea]. Available: http://personales.upv.es/dginesta/docencia/posgrado/sparse.pdf. [Último acceso: 15 Abril 2020].

[26] R. D. Falgout, J. E. Jones y U. M. Yang, «Pursuing Scalability for hypre's Conceptual Interfaces,» Lawrence Livermore National Laboratory, 2004.

[27] M. Bendsoe, «Optimization of structural topology shape and material,» New York, Springer, 1995.

[28] J. Martínez Palacios, M. E. Martinez Lomas, R. López Carreras y M. G. Del Rio Cidoncha, «Diseño automatizado, mediante técnicas de sistemas expertos en Catia V5, aplicado a componentes estructurales aeronáuticos,» de *XVIII Congreso Internacional de Ingeniería Gráfica*, Barcelona, 2006.

[29] A. Juste Ruiz, *Diseño de un ala de avión mediante técnicas numéricas (FEM),* Universidad Carlos III de Madrid, 2016.

[30] L. M. García-Cuevas Gonzalez, M. Carreres Talens y A. O. Tiseira Izaguirre, *Arquitectura general de aeronaves,* Departamento de Máquinas y Motores Térmicos. Universitat Politécnica de Valencia.

[31] S. Esteban Roncero, *Estucturas preliminares,* Departamento de Ingeniería Aeroespacial y Mecánica de Fluidos. Universidad de Sevilla.

[32] Á. Rodríguez Ortiz, *Diseño simplificado de la sección transversal del ala de una aeronave comercial,* Universidad Carlos III de Madrid. Proyecto Fin de Carrera, 2014.

[33] P. Marzocca, *The NACA airfoil series,* Clarkson University, 2016.

[34] «Production summary : Airbus A320,» [En línea]. Available: https://www.airfleets.net/exploit/production-a320.htm. [Último acceso: 10 Septiembre 2020].

[35] Airbus, «Aircraft Characteristics,» [En línea]. Available: https://www.airbus.com/aircraft/support-services/airport-operations-and-technical-data/aircraft-characteristics.html. [Último acceso: 10 Septiembre 2020].

[36] «9 Types of Aircraft Wings in Depth,» [En línea]. Available: https://www.aircraftcompare.com/blog/types-of-aircraft-wings/. [Último acceso: 10 Septiembre 2020].

[37] D. Ignjatovic, «Decoding the Splitfire. Part 2,» 2017.