



Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

*Optimización de un procedimiento de vigilancia marítima en un
escenario de recursos limitados*

Grado en Ingeniería Mecánica

ALUMNO: Rubén García Alonso

DIRECTORES: Sergio Borrallo Tirado
Débora Cores Carrera

CURSO ACADÉMICO: 2024-2025

Universida_{de}Vigo



Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

*Optimización de un procedimiento de vigilancia marítima en un
escenario de recursos limitados*

Grado en Ingeniería Mecánica
Intensificación en Tecnología Naval
Cuerpo General

UniversidadeVigo

RESUMEN

La vigilancia marítima es una de las misiones principales de la Armada, contribuyendo a la seguridad nacional, la protección de los intereses marítimos y la preservación y conservación del medio marino y sus recursos. No obstante, estas operaciones de vigilancia se desarrollan en un escenario de recursos limitados, tales como la disponibilidad de plataformas navales, su personal, la capacidad de detección, así como la autonomía de las mismas o su eficacia. Este TFG tiene como objetivo optimizar la asignación de tiempos de patrulla óptimos dependiendo de las condiciones y características tanto de las áreas de vigilancia como de las unidades, así como las prioridades de la autoridad encargada, mediante el diseño y análisis de recursos que permitan maximizar la eficiencia operativa en un escenario de recursos limitados.

El trabajo combina técnicas de optimización matemática y modelos de asignación de recursos basados en programación lineal para mejorar la cobertura de áreas marítimas críticas y la detección de actividades de interés, como el tráfico ilícito, la pesca y la explotación ilegal y las posibles amenazas a la seguridad.

La propuesta presentada en este trabajo busca no solo optimizar la vigilancia marítima actual, sino también servir como herramienta para la planificación futura de cualquier proceso de optimización en el que se conozcan sus variables. Este planteamiento permitirá a la Armada gestionar sus recursos de una manera más sostenible, eficaz y óptima.

PALABRAS CLAVE

Optimización, Programación no lineal, Fmincon, Eficiencia operativa, Modelo matemático

AGRADECIMIENTOS

Quiero expresar mi más profundo agradecimiento a todas las personas que han formado parte de este camino y que, de una u otra manera, han contribuido a la realización de este trabajo y este proyecto de vida.

A mi madre, Maite, por su apoyo incondicional, por ser siempre un ejemplo de esfuerzo y dedicación, y por brindarme la confianza necesaria para superar cada desafío. A mi hermano, Asier, por su compañía y por estar siempre presente, incluso en los momentos más difíciles.

A mis amigos, por su amistad, por compartir conmigo este recorrido y hacer que los momentos de mayor exigencia fueran más llevaderos con su compañía y apoyo.

Quiero agradecer especialmente a mis profesores, Sergio Borrallo Tirado y Débora Cores Carrera, por su orientación, por compartir sus conocimientos y por su disposición a guiarme en este proceso, facilitando el desarrollo del presente trabajo con sus consejos y enseñanzas.

Finalmente a Marta, por su paciencia, comprensión y por estar siempre a mi lado, brindándome ánimo y apoyo en cada etapa de este proyecto.

A todos ellos, mi más sincero agradecimiento por acompañarme en este reto y por hacer posible la culminación de esta etapa tan importante en mi vida.

CONTENIDO

Contenido	1
Índice de Figuras	4
Índice de Tablas.....	6
1 Introducción y objetivos	7
1.1 Contexto y motivación del trabajo	7
1.2 Planteamiento del problema.....	10
1.3 Objetivos	10
1.4 Estructura de la memoria	11
2 Estado del arte	13
2.1 La vigilancia marítima	13
2.1.1 Definición de vigilancia marítima	13
2.1.2 Tecnologías.....	14
2.1.3 Desafíos operativos.....	16
2.1.4 Tendencias futuras	17
2.2 Optimización.....	18
2.2.1 Concepto de la optimización	18
2.2.2 Optimización de función objetivo con restricciones.....	18
2.2.3 Aplicado a la vigilancia marítima.....	19
2.3 El método simplex	20
2.3.1 Definición y fundamentos.....	20
2.3.2 Consideraciones al utilizar el método simplex	21
2.3.2.1 Variables de holgura y exceso	21
2.3.2.2 Variable artificial o método de la M.....	21
2.3.2.3 Variables no negativas.....	22
2.3.3 Aplicaciones del Método Simplex.....	22
2.4 El método fmincon.....	23
2.4.1 Algoritmos subyacentes.....	23
2.4.2 Elección de fmincon	24
2.5 Herramientas computacionales	25
2.5.1 Definición y contexto	25
2.5.2 Herramientas computacionales evaluadas	25
2.5.2.1 GAMS.....	26
2.5.2.2 LINGO.....	27

2.5.2.3 MATLAB	28
3 Desarrollo del TFG.....	30
3.1 Formulación del problema	30
3.1.1 Descripción del problema de optimización en vigilancia marítima	30
3.1.2 Objetivos y restricciones del modelo.....	30
3.1.3 Método matemático seleccionado.....	31
3.2 Estructura del Código.....	35
3.2.1 Interfaz gráfica de bienvenida y solicitud de datos	35
3.2.2 Validación y preprocesamiento de datos	37
3.2.3 Implementación del algoritmo de optimización	38
3.2.4 Representación gráfica y visualización	40
3.2.5 Salida de resultados en consola	41
3.3 Configuración del entorno y requisitos	41
3.4 Modelado del área de patrullaje	42
3.4.1 Definición de las regiones y representación geográfica	42
3.5 Evaluación del algoritmo y posibles mejoras	44
3.5.1 Análisis de la eficiencia del modelo	44
3.5.2 Posibles mejoras y optimización futura.....	44
3.5.3 Interpretación operativa	45
4 Resultados obtenidos.....	47
4.1 Evaluación general del modelo	47
4.1.1 Objetivo y metodología del análisis	47
4.1.2 Definición de métricas de evaluación.....	47
4.2 Análisis 1: Comprobación del modelo	49
4.2.1 Verificación de la correcta asignación de tiempos con pesos uniformes.	49
4.2.2 Evaluación del modelo destacando factores clave como WA o CA	51
4.3 Análisis 2: Escenario estándar	54
4.3.1 Condiciones estándar sin factor meteorológico	55
4.3.2 Condiciones estándar con factor meteorológico.....	56
4.4 Análisis 3: Saturación operativa	57
4.4.1 Área extensa con Tmax reducido	57
4.4.2 Áreas de mucho interés con Tmax reducido.....	59
4.5 Análisis 4: Sensibilidad a variabilidad meteorológica.....	60
4.6 Análisis 5: Robustez ante asignaciones extremas de Tmax.....	61
4.6.1 Tmax insuficiente	61
4.6.2 Tmax excesivo	62

4.7 Análisis 6: Prueba de escalabilidad. Modificación del número de áreas	63
4.7.1 Ejecución con 3 áreas	63
4.7.2 Ejecución con 5 áreas	64
4.7.3 Ejecución con 8 áreas	65
4.7.4 Ejecución con 10 áreas	66
4.8 Análisis 7: Sensibilidad ante parámetros no lógicos.....	67
4.8.1 Datos mal introducidos	67
4.8.2 Área con probabilidades y factor meteorológico cero	68
4.8.3 Área con probabilidades cero	69
4.9 Validación del código y conclusiones.....	70
4.10 Análisis de desempeño.....	72
5 Conclusiones y líneas futuras	74
5.1 Conclusiones	74
5.2 Líneas futuras	75
6 Bibliografía.....	77
Anexo I: Implicaciones Sociales, y/o Económicas, y/o Ambientales	82
Anexo II: Reflexiones Éticas y Sociales	83
Anexo III: Código fuente en Matlab	84
Anexo IV: Datos utilizados para la simulación.....	93
Anexo V: Diccionario de acrónimos	98

ÍNDICE DE FIGURAS

Figura 1-1 Cuadro mando resumen 2023 puertos estatales [2]	7
Figura 1-2 Organismos implicados en la vigilancia marítima en España [6]	8
Figura 1-3 Actividad marítima en embarcaciones rápidas y no identificadas en el STROG [9]	9
Figura 2-1 Espacios de interés para la seguridad marítima nacional [13].....	13
Figura 2-2 Lanzamiento de dron ScanEagle desde buque [20]	14
Figura 2-3 USV Sead 23 en navegación [25]	15
Figura 2-4 Visualización sistema GAIA IA [29]	17
Figura 2-5 Sistema de identificación automática (AIS) [37].....	20
Figura 2-6 Representación gráfica del método simplex [40]	20
Figura 2-7 Interfaz del software GAMS [48]	26
Figura 2-8 Interfaz del software LINGO [51]	27
Figura 2-9 Entorno gráfico de MATLAB [52].....	28
Figura 3-1 Código definiendo fmincon	33
Figura 3-2 Interfaz de bienvenida	35
Figura 3-3 Interfaz de solicitud de datos al usuario	36
Figura 3-4 Código validación tiempo máximo.....	37
Figura 3-5 Código definiendo los datos de entrada.....	37
Figura 3-6 Código definiendo fmincon	39
Figura 3-7 Gráfica de barras mostrando resultados.....	40
Figura 3-8 Salida de datos en consola mostrando resultados	41
Figura 3-9 Código definiendo coordenadas que definen áreas	42
Figura 3-10 Código de la representación geográfica.....	43
Figura 3-11 Código de áreas en el mapa	43
Figura 3-12 Interfaz mostrando áreas a patrullar con tiempo.....	43
Figura 3-13 Comparativa de ejecución secuencial y paralela	45
Figura 4-1 Resultados mostrados en consola	47
Figura 4-2 Gráfico radar con diferentes aspectos de un modelo	49
Figura 4-3 Asignación de tiempos por zonas	50
Figura 4-4 Tiempos ejecución algoritmo	51
Figura 4-5 Análisis destacando WA.....	51
Figura 4-6 Asignación de tiempos diferentes capacidades.....	52
Figura 4-7 Interfaz explicación capacidad	53
Figura 4-8 Asignación de tiempos con factor meteorológico	53
Figura 4-9 Perfil naval BAM Furor P-46 [54]	55

Figura 4-10 Asignación de tiempos en Análisis 2.....	55
Figura 4-11 Asignación de tiempos escenario estándar	56
Figura 4-12 Error Tmax insuficiente	57
Figura 4-13 Asignación tiempos con área extensa	58
Figura 4-14 Asignación de tiempos sin tiempos extras.....	58
Figura 4-15 Asignación de tiempos en saturación operativa.....	59
Figura 4-16 Asignación de tiempos con meteorología cambiante	60
Figura 4-17 Advertencia error Tmax insuficiente	61
Figura 4-18 Asignación de tiempos tras error	62
Figura 4-19 Asignación de tiempos con Tmax excesivo.....	63
Figura 4-20 Asignación de tiempos con 3 áreas.....	64
Figura 4-21 Asignación de tiempos con 5 áreas.....	65
Figura 4-22 Asignación de tiempos con 8 áreas.....	65
Figura 4-23 Error al ejecutar 10 áreas	66
Figura 4-24 Comparación resultados	67
Figura 4-25 Mensaje advertencia	68
Figura 4-26 Tiempos con Z5 ceros.....	69
Figura 4-27 Tiempos con Z5 ceros salvo factor meteorológico.....	70
Figura 4-28 Número de iteraciones por análisis.....	72
Figura 4-29 Tiempos medios de ejecución.....	73

ÍNDICE DE TABLAS

Tabla 2-1 Comparativa fmincon y Método Simplex	24
Tabla 2-2 Comparación de los diferentes algoritmos de optimización [44] [45] [46]	25
Tabla 2-3 Comparativa herramientas computacionales evaluadas	29
Tabla 3-1 Restricciones aplicadas al modelo	31
Tabla 3-2 Formato de entrada de los datos al código	38
Tabla 4-1 Datos de salida del modelo	48
Tabla 4-2 Parámetros de métrica del rendimiento	48
Tabla 1 Datos entrada Análisis 1 – Ensayo 1	93
Tabla 2 Datos entrada Análisis 1 – Ensayo 2	93
Tabla 3: Datos entrada Análisis 1 – Ensayo 3	93
Tabla 4 Datos entrada Análisis 1 – Ensayo 4	94
Tabla 5 Datos entrada Análisis 1 – Ensayo 5	94
Tabla 6 Datos entrada Análisis 1 – Ensayo 6	94
Tabla 7 Datos entrada Análisis 1 – Ensayo 7	95
Tabla 8 Datos entrada Análisis 1 – Ensayo 8	95
Tabla 9 Datos entrada Análisis 2 – Ensayo 1	95
Tabla 10 Datos entrada Análisis 2 – Ensayo 2	96
Tabla 11 Datos entrada Análisis 3 – Ensayo 1	96
Tabla 12 Datos entrada Análisis 3 – Ensayo 2	96
Tabla 13 Datos entrada Análisis 4	97
Tabla 14 Datos entrada Análisis 7 – Ensayo 1	97
Tabla 15 Datos entrada Análisis 7 – Ensayo 2	97

1 INTRODUCCIÓN Y OBJETIVOS

1.1 Contexto y motivación del trabajo

Históricamente y en especial en las últimas décadas, el entorno marítimo ha adquirido una relevancia estratégica fundamental para el comercio, la seguridad y el desarrollo económico de las naciones. En este sentido, España ocupa una posición clave debido a su situación geográfica, lo que la convierte en un punto de conexión entre el Océano Atlántico y el Mar Mediterráneo, así como entre Europa, África y América. La extensión de nuestras aguas jurisdiccionales y la importancia de las costas requieren un sistema de vigilancia marítima eficiente que garantice la seguridad de los intereses económicos y estratégicos.

En este contexto, destaca la importancia del sector marítimo en España el cual es un componente crucial de nuestra economía, con un volumen significativo de mercancías transportadas por mar y una gran densidad de tráfico marítimo atravesando nuestras aguas. Como se muestra en la Figura 1-1, en 2023, los puertos españoles gestionaron el transporte de más de 540 millones de toneladas de mercancías, incluyéndose productos industriales, agrícolas y de consumo. Además, el número de contenedores manejados alcanzó los 5 millones de TEU, lo que resalta así la importancia del comercio internacional[1].

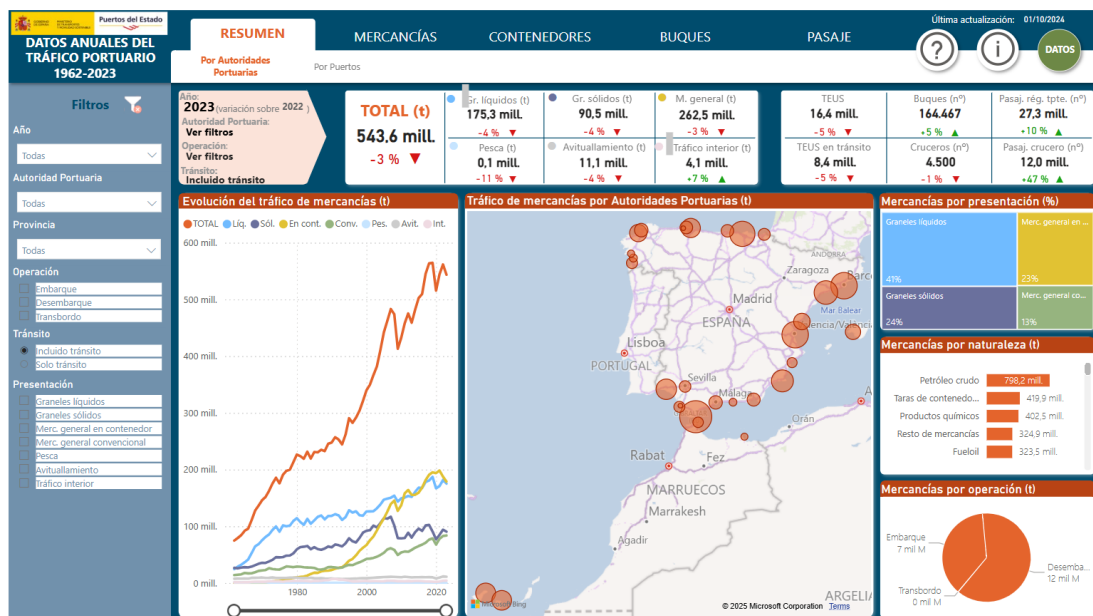


Figura 1-1 Cuadro mando resumen 2023 puertos estatales [2]

En lo referente a buques, aproximadamente 20.000 barcos de diferentes tamaños y propósitos atracaron en puertos españoles durante el mismo año. Estas actividades comerciales marítimas tienen entonces gran impacto en la economía nacional. Solamente el sector portuario español generó en 2023 un impacto económico de 24.304 millones de euros y 250.000 empleos, lo que representa aproximadamente un 2,2% del PIB y el 1,4% del empleo total de la economía nacional [3]. Estos datos reflejan la vitalidad del sector marítimo en España y la conectividad global y comercio exterior del país; cobrando un sentido crucial la vigilancia marítima para garantizar que este sector se desarrolla en las máximas condiciones de seguridad y legalidad.

En la actualidad, los procedimientos de vigilancia marítima hacen frente a múltiples desafíos. Por un lado, existe un aumento de actividades ilícitas como el tráfico de drogas, la inmigración irregular, la pesca ilegal y el contrabando[4]. Por otro lado, los recursos disponibles para llevar a cabo estas operaciones son limitados, lo que obliga a optimizar los medios materiales, tecnológicos y humanos disponibles. En este contexto, la Armada juega un papel fundamental en la seguridad y defensa marítima, colaborando con organismos nacionales e internacionales para garantizar la protección de las aguas españolas y la legislación vigente[5].

La seguridad marítima en España está gestionada por diferentes organismos representados en la Figura 1-2, como son la Armada, la Guardia Civil y la Agencia Tributaria, lo que supone una distribución de responsabilidades en el ámbito marítimo. La Armada, a través del Mando Operativo Marítimo, desempeña un papel esencial en la vigilancia y protección de los mares, pero la dispersión de competencias dificulta una gestión unificada.



Figura 1-2 Organismos implicados en la vigilancia marítima en España [6]

Esta dispersión de competencias en el sector marítimo afecta significativamente a la eficiencia operativa, debido a la fragmentación en el proceso de toma de decisiones, la coordinación y la gestión de recursos. Esta dispersión se origina en la distribución de responsabilidades entre diferentes niveles administrativos (gobierno central, comunidades autónomas y autoridades locales). Uno de los principales problemas derivados de esta fragmentación es la falta de una estrategia unificada que agilice toda la gestión de operaciones portuarias y vigilancia marítima. La dispersión de competencias entre la Guardia Civil, Salvamento Marítimo, Aduanas y la Armada puede generar problemas en la respuesta ante incidentes como el narcotráfico o la piratería. La falta de un mando único o de protocolos de coordinación efectivos ralentizan la actuación y reducen la eficiencia y eficacia en la lucha contra el crimen en la mar.

La falta de un sistema coordinado de vigilancia ha propiciado un aumento documentado de actividades delictivas en el entorno marítimo español, tal como reflejan informes recientes sobre narcotráfico y contrabando. Desde la década de los 2000, se ha producido un aumento significativo en los delitos relacionados con el tráfico de drogas, la piratería y el contrabando. Uno de los casos más reseñables es el uso de narcosubmarinos, utilizados para transportar grandes cantidades de cocaína desde

América del Sur hasta las costas españolas[7]. Este tipo de delitos ha obligado a las autoridades a implementar estrategias más robustas de seguridad marítima, como la Estrategia de Seguridad Nacional Marítima (ENSM) de 2024, cuyo objetivo es proteger las rutas comerciales, las infraestructuras del litoral y el fondo marino y los recursos que los océanos proporcionan. Además la colaboración internacional es crucial para combatir estos delitos, ya que el crimen organizado opera en territorios de frontera y emplea tecnología avanzada para la contradetección. A pesar de estos esfuerzos, el crimen en el sector marítimo sigue presentando un desafío importante para España, requiriéndose una vigilancia constante y eficiente, y una cooperación internacional continua para mantener la seguridad en las aguas.

La implementación de nuevas técnicas de optimización se presenta como una solución clave para maximizar la eficacia de los recursos disponibles. La gestión eficiente de sensores y plataformas, los sistemas de toma de decisiones y la inteligencia artificial han demostrado ser herramientas útiles en este tipo de operaciones. No debemos olvidar, sin embargo, que la aplicación de estas herramientas debe adaptarse a las particularidades del entorno nacional y de los retos específicos que enfrenta la Armada Española en sus operaciones de vigilancia.

El desarrollo de un procedimiento optimizado de vigilancia marítima es una respuesta práctica a la necesidad de garantizar la seguridad en un entorno complejo con recursos limitados. Este enfoque busca mejorar la capacidad de la Armada Española para enfrentar desafíos actuales y futuros, promoviendo la innovación y una gestión eficiente de los recursos. Además, el proyecto se alinea con los esfuerzos globales por la sostenibilidad, reduciendo costos e impactos ambientales, y contribuyendo al fortalecimiento de las capacidades nacionales en el ámbito marítimo.

Se puede predecir que una asignación ineficiente de patrullas marítimas puede generar zonas desprotegidas y aumentar la vulnerabilidad ante delitos como el narcotráfico, la inmigración irregular, la pesca ilegal y la piratería. Esto se debe a factores como la cobertura desigual del territorio marítimo, la actual falta de coordinación entre organismos, un escenario de recursos limitados, unos patrones delictivos cambiantes y un tiempo de respuesta o vigilancia inadecuado. Un claro ejemplo de ello es el caso del Estrecho de Gibraltar, una zona de alta actividad delictiva, como se observa en la Figura 1-3, y con recursos gubernamentales limitados, donde el narcotráfico se ha hecho cada vez más visible.

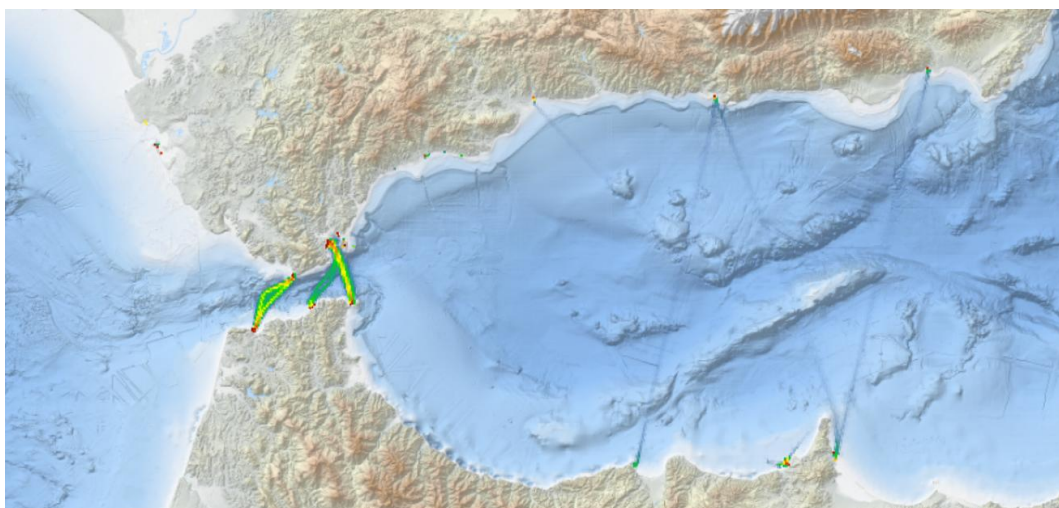


Figura 1-3 Actividad marítima en embarcaciones rápidas y no identificadas en el STROG [9]

Para este trabajo fin de grado, la optimización de los recursos de la vigilancia marítima es un desafío clave para garantizar la cobertura eficiente de grandes extensiones marítimas. Los modelos matemáticos juegan un papel fundamental al permitir una planificación estratégica y la asignación óptima de sensores, drones, plataformas y otros medios de vigilancia. Mediante el uso de técnicas como la programación lineal, los algoritmos genéticos o la teoría de juegos, es posible diseñar esquemas de patrullaje que maximicen la detección minimizando los costos operativos.

Este trabajo se enfoca en optimizar la asignación de tiempos de patrullaje en la vigilancia marítima, resolviendo problemas de cobertura desigual y uso ineficiente de recursos. Mediante un modelo matemático se distribuye el tiempo de patrulla considerando restricciones operativas como autonomía, condiciones meteorológicas e importancia de cada zona.

1.2 Planteamiento del problema

El presente trabajo fin de grado tiene como objetivo desarrollar un algoritmo que permita asignar de manera óptima el tiempo de patrullaje de un patrullero para vigilar un conjunto de áreas de diferente tamaño y características, cuyas dimensiones pueden variar según las necesidades. Cada una de estas zonas presenta una serie de características particulares, entre las que destacan una serie de eventos potenciales cuya probabilidad de ocurrencia está predefinida.

El patrullero cuenta con unas capacidades determinadas para localizar estos eventos, lo que influye directamente en su capacidad de patrullaje en cada área. Además, se asigna una importancia relativa a la localización de cada evento, de acuerdo con su relevancia estratégica, así como una serie de restricciones de larga permanencia de manera continuada en un mismo área, para fomentar una distribución de tiempos más realista, eficiente y homogénea.

Dado un tiempo total de patrullaje disponible, el algoritmo diseñado tiene como objetivo distribuir ese tiempo entre las diferentes áreas de forma que se maximicen las probabilidades de detectar los eventos de interés, optimizando así el uso de los recursos y aumentando la eficiencia en la vigilancia.

La solución propuesta toma en cuenta tanto las características específicas de las áreas como las capacidades del patrullero, con el fin de encontrar la distribución de tiempos que maximice la probabilidad de localizar los eventos según su importancia y probabilidad de ocurrencia.

Este planteamiento permite una gestión eficiente del tiempo de patrullaje, garantizando que se prioricen las áreas y eventos más relevantes, maximizando la probabilidad de éxito en la vigilancia.

1.3 Objetivos

El objetivo principal de este trabajo fin de grado es desarrollar un programa en MATLAB que permita optimizar el proceso de vigilancia marítima, atendiendo a parámetros tanto de la plataforma a emplear como del entorno a patrullar y la importancia de la autoridad responsable. El programa tiene como fin mejorar la eficiencia y efectividad de las operaciones de vigilancia mediante la integración de herramientas matemáticas de optimización.

La solución a desarrollar se presenta como una herramienta adaptable a distintos contextos de ingeniería, destacando su versatilidad y eficiencia en escenarios operativos restringidos. Además, esta herramienta contribuirá a reducir el impacto ambiental, promoviendo el uso eficiente de recursos y adoptando estrategias que respeten las condiciones del entorno, fomentando así prácticas más sostenibles en el ámbito de la vigilancia marítima y otros contextos.

Como objetivos específicos, se pretende, partiendo de un modelo de problema determinado, resolver un desafío de interés para la Armada, de gran interés actual, mediante el diseño de un algoritmo y la programación del código correspondiente. Una vez el código ha sido creado, se analizará su robustez y se evaluará su rendimiento ante diferentes escenarios tipo, con el fin de obtener conclusiones sobre su funcionamiento e implementación en un entorno operativo real.

En términos generales, el objetivo es diseñar una herramienta flexible y escalable, que no solo sea aplicable al problema de vigilancia marítima planteado en este trabajo, sino que también pueda adaptarse a resolver otros problemas de optimización similares, ajustando y definiendo los parámetros de entrada del programa según sea necesario.

1.4 Estructura de la memoria

Tras contextualizar el entorno en el que se desarrolla este trabajo y definir los objetivos que se pretenden alcanzar, se procede a describir la estructura de la memoria, diseñada para presentar de forma clara tanto el planteamiento del problema como el desarrollo y conclusiones extraídas del trabajo. Este documento se divide en cinco capítulos, seguidos de una bibliografía que recopila las fuentes consultadas, un anexo donde se plasma el código íntegro utilizado para su posterior comprobación y/o utilización, y un diccionario de acrónimos utilizados en esta memoria. La organización de esta memoria es la siguiente:

- Capítulo 1: Introducción y objetivos. En este primer capítulo se aborda la importancia de optimizar la vigilancia marítima, concretamente en un escenario de recursos limitados. Se expone el papel fundamental de la Armada y otros organismos en la protección del territorio marítimo y el impacto de las tecnologías en la mejora de la eficiencia operativa. Asimismo, se define el marco en el que se sitúa este trabajo, se describe la problemática de la asignación de recursos y la necesidad de aplicar métodos de optimización. Finalmente, se plantean los objetivos del proyecto y se presenta la estructura de la presente memoria.
- Capítulo 2: Estado del arte. Se lleva a cabo una revisión del estado actual de la vigilancia marítima y de los métodos de optimización empleados en la gestión de recursos. Los conceptos tratados en este capítulo se pueden resumir en los siguientes puntos:
 - Introducción a la vigilancia marítima, definición e importancia de mejorar su eficiencia.
 - Descripción de los procedimientos actuales de patrullaje y los retos asociados a su implementación.
 - Exploración de las técnicas de optimización utilizadas en diferentes ámbitos de la ingeniería y su aplicabilidad en la vigilancia marítima.
 - Análisis del Método Simplex y del uso de fmincon en la optimización de asignación de recursos.
 - Revisión de estudios previos y trabajos relacionados en la optimización del patrullaje marítimo.
- Capítulo 3: Desarrollo del TFG. En este capítulo se detalla el desarrollo del modelo de optimización implementado en MATLAB. Se explican las herramientas utilizadas y los datos empleados para alcanzar los objetivos del trabajo. De igual manera, se justifica la selección del algoritmo fmincon para resolver el problema de asignación de tiempos de patrullaje. Además, se describen los pasos seguidos en la formulación del problema, la estructura del código, los procedimientos de validación y las estrategias adoptadas para mejorar la eficiencia del modelo. Por último, se exponen los criterios de optimización y las restricciones aplicadas al modelo.
- Capítulo 4: Resultados obtenidos. En este capítulo se presentan los resultados obtenidos tras la aplicación del modelo de optimización. Se analizan los valores obtenidos en distintos escenarios y se evalúa la eficacia del modelo en términos de distribución del tiempo de patrullaje y mejora en la eficiencia operativa. Se compara la solución generada con enfoques tradicionales y se determina el impacto de la optimización en la cobertura del patrullaje. Finalmente, se validan los resultados mediante pruebas de robustez y sensibilidad a variaciones en los parámetros de entrada.
- Capítulo 5: Conclusiones y líneas futuras. En el último capítulo se presentan las conclusiones obtenidas tras el desarrollo del trabajo. Se resumen los beneficios de la aplicación del modelo

de optimización en la vigilancia marítima y se identifican las principales limitaciones del estudio. Asimismo, se proponen futuras líneas de investigación para mejorar el modelo, como la integración de técnicas de inteligencia artificial, la incorporación de datos en tiempo real y la optimización multiobjetivo.

- Bibliografía y anexos: Finalmente, se incluye la bibliografía donde se detallan las fuentes consultadas a lo largo del trabajo, junto con los anexos que complementan y respaldan el contenido de la memoria.

2 ESTADO DEL ARTE

2.1 La vigilancia marítima

2.1.1 Definición de vigilancia marítima

La vigilancia marítima comprende el conjunto de actividades, tecnologías, sistemas y procedimientos destinados a monitorear, controlar y garantizar la seguridad y protección de las aguas marítimas. Su propósito principal es detectar amenazas y la prevenir actividades ilícitas en la mar; tales como el tráfico de drogas, la migración irregular, la pesca ilegal, el expolio del patrimonio y la piratería. Además, abarca el control del tráfico marítimo y la protección de recursos marinos[10] dentro de nuestra ZEE.

La eficiencia en la vigilancia marítima depende no solo de la tecnología y los sistemas de monitoreo, sino también de la optimización de los recursos disponibles, como la asignación de patrullas en zonas estratégicas. En este contexto, el presente trabajo busca mejorar la planificación del patrullaje mediante herramientas matemáticas que permitan distribuir el tiempo de vigilancia de manera óptima.

Según la Agencia Europea de Control de Fronteras y Costas (Frontex), la vigilancia marítima desempeña un papel crucial en cuanto a garantizar la seguridad y el control de las fronteras marítimas de la Unión Europea y prevenir actividades ilegales [11], destacando zonas de interés para la seguridad marítima tal como se muestra en la Figura 2-1. Por su parte, la OMI subraya la importancia de la vigilancia marítima para promover la seguridad de la navegación y proteger el medio marino frente a actividades no reguladas [12].

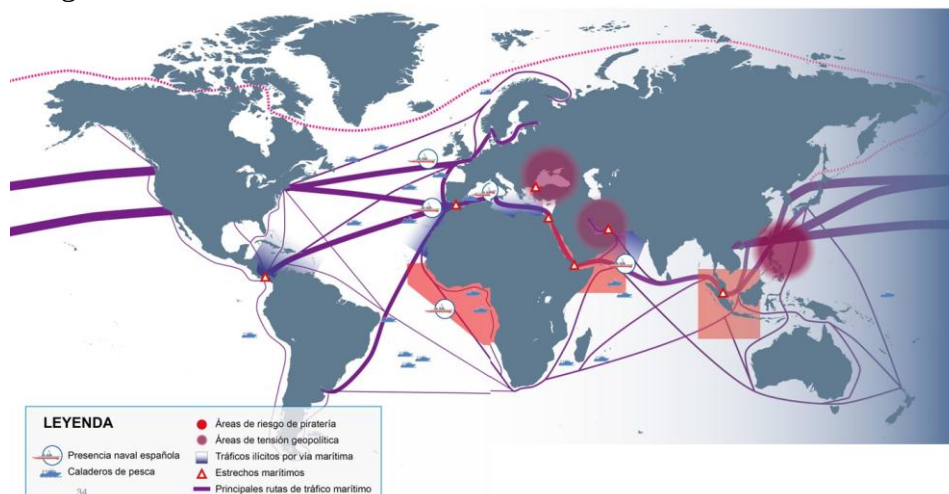


Figura 2-1 Espacios de interés para la seguridad marítima nacional [13]

La vigilancia marítima es esencial para alcanzar una Maritime Domain Awareness (MDA) efectiva, ya que permite recopilar, analizar y utilizar información en tiempo real para fortalecer la seguridad en la mar. A través de diversas tecnologías integradas en las plataformas de los organismos de vigilancia, la información procesada mediante IA y análisis de datos facilita la identificación de patrones sospechosos y la optimización en la asignación de recursos. Esto permite una respuesta más rápida y eficiente ante incidentes[14], generando una clara RMP de las aguas de interés.

Por ejemplo, en la lucha contra el narcotráfico, la MDA permite predecir rutas ilícitas y coordinar patrullas en zonas estratégicas. En la prevención de la migración irregular, permite detectar embarcaciones con alta probabilidad de sospecha. Asimismo, en la protección de infraestructuras críticas, contribuye a anticipar posibles amenazas. De esta manera, la vigilancia marítima se consolida como un pilar fundamental de la Maritime Situational Awareness (MSA), garantizando un control más eficaz del espacio marítimo y reforzando la seguridad.

Una vigilancia marítima eficaz permite una respuesta oportuna ante amenazas de seguridad, el medio marino y la actividad económica, asegurando una MDA sólida. Un ejemplo reciente es el daño al cable submarino entre Finlandia y Estonia en diciembre de 2024, presuntamente causado por un petrolero ruso de la “flota en la sombra” de Moscú[15].

Otro caso, como el sucedido en enero de 2025, evidencia cómo el narcotráfico aprovecha las lagunas en la vigilancia marítima en su favor. En este caso, las autoridades interceptaron 34 kg de cocaína en aguas del Canal de Ibiza. La droga era arrojada al mar por tripulaciones corruptas de barcos mercantes para su posterior recogida por embarcaciones más pequeñas[16].

Además, la crisis de piratería en el Golfo de Adén y la costa de Somalia entre 2008 y 2012 demuestra la importancia contar con un sistema de monitoreo robusto. La ausencia de vigilancia facilitó que grupos armados secuestraran numerosos barcos comerciales, afectando así al comercio internacional y poniendo en riesgo la seguridad de las tripulaciones[17].

2.1.2 Tecnologías

La Armada invierte una gran cantidad de recursos en la integración de tecnologías avanzadas para reforzar la vigilancia marítima y garantizar la protección de las aguas territoriales y el espacio de soberanía[18]. Entre las principales herramientas utilizadas hoy en día destacan los radares de última generación de los que disponen los buques de la Armada, que permiten la detección temprana de actividades sospechosas a gran distancia, mejorando de manera significativa la capacidad de anticipación y respuesta ante posibles amenazas[19]. Además, el uso del Sistema de Identificación Automática (AIS) facilita el monitoreo en tiempo real del tráfico marítimo, asegurando una supervisión continua y precisa.



Figura 2-2 Lanzamiento de dron ScanEagle desde buque [20]

En el ámbito submarino, los sonares son fundamentales para la detección y clasificación de amenazas bajo el agua, tanto en submarinos como en buques de superficie [19].

Asimismo, la Armada está incorporando tecnología de drones para la vigilancia marítima como se puede ver en la Figura 2-2, permitiendo la evaluación y reconocimiento de áreas extensas. El uso de drones proporciona imágenes en tiempo real y complementa la información obtenida a través de otros medios como radares y satélites.

A su vez, la integración de satélites permite la recopilación de datos estratégicos para la vigilancia marítima, pudiendo cubrir áreas mucho más extensas proporcionando información continua que resulta esencial para la detección de actividades ilegales o sospechosas.

Por otra parte, las plataformas de control marítimo integran información proveniente de diferentes fuentes (radares, satélites, drones...) para proporcionar una visión completa y coherente de la situación de las costas. Esto facilita en gran medida lograr una respuesta coordinada y eficiente ante cualquier eventualidad. Cabe destacar la importancia que van adquiriendo la IA y el análisis de Big Data en la vigilancia marítima. La IA por su parte es utilizada para analizar patrones de comportamiento y predecir posibles actividades sospechosas, mientras que el análisis con Big Data permite manejar y procesar grandes volúmenes de información, mejorando así el proceso de toma de decisiones.

Los Sistemas Marítimos No Tripulados (SMNT) representan una de las áreas más dinámicas y prometedoras de la industria naval militar. Estos sistemas abarcan desde pequeñas embarcaciones de superficie hasta sofisticados vehículos submarinos, capaces de llevar a cabo misiones de reconocimiento, vigilancia y operaciones en entornos de alto riesgo para la vida humana. La Armada considera los vehículos no tripulados (UXV) esenciales para llevar a cabo operaciones multidominio, ya que mejoran las capacidades operativas de las unidades tripuladas y reducen riesgos.

Con la incorporación del programa NAIAD en los BAM y en pruebas recientes, se ha validado su utilidad en escenarios complejos[21]. El documento “Armada 2050” traza los pasos para integrar estos sistemas, alineándose así España con la tendencia global. La interoperabilidad y la capacidad de operar en entornos degradados son desafíos críticos, así como la popularización de tecnologías avanzadas[22]. La integración de la IA, la computación cuántica y los sistemas autónomos se posiciona como la clave para mantener la ventaja operativa en un escenario de combate[23].

Por su parte, los SMNT han revolucionado el panorama naval, destacando los vehículos de superficie y submarinos con capacidades avanzadas y autonomía mejorada, permitiéndoles operar en zonas de alto riesgo. La colaboración operativa entre múltiples sistemas no tripulados está potenciando la vigilancia y el control marítimo, reduciendo la intervención humana en tareas con riesgo asociado[24].



Figura 2-3 USV Sead 23 en navegación [25]

La Armada, en busca de aumentar sus capacidades tecnológicas, ha integrado recientemente el Sead 23, mostrado en la Figura 2-3; un USV multifuncional de 7 metros de eslora, con motorización diésel y propulsión mediante waterjet. Puede alcanzar velocidades de 30 nudos con olas de metro y medio de altura, y su autonomía y capacidad de adaptación al medio se sitúa por encima de la media para este tipo de vehículos. Está equipado con radar, sonar, AIS, señores multiparamétricos y armamento; como la estación de armas Lionfish de 12,70 mm y el torpedo Black Scorpion, ambos de la empresa italiana Leonardo[25].

La integración de estas tecnologías en los procedimientos de vigilancia marítima permite una optimización sustancial en la asignación de recursos, mejorando la cobertura de las áreas de patrullaje y reduciendo tiempos de respuesta ante incidentes. Al combinar sensores avanzados con modelos de optimización, es posible distribuir los medios de vigilancia de manera más eficiente, asegurando que las zonas de mayor riesgo reciban mayor atención. Esto no solo incrementa la seguridad marítima, sino que también permite una gestión más sostenible y racional de los recursos disponibles.

2.1.3 Desafíos operativos

La vigilancia marítima enfrenta en la actualidad una serie de desafíos operativos complejos que requieren de una constante atención y de soluciones integrales. Ejemplo de ello son:

- Detección y seguimiento de embarcaciones no identificadas: La identificación de embarcaciones que operan sin AIS o con estos desactivado, conocidas como “dark ships”, es un reto significativo en la actualidad. Estas prácticas son comunes entre actores ilícitos, como narcotraficantes y pescadores ilegales, que buscan eludir la detección [26]. La ausencia de señales electrónicas complica su localización y seguimiento, demandando tecnologías avanzadas y métodos innovadores para su monitoreo efectivo.
- Protección de infraestructura marítima crítica: Las infraestructuras submarinas como cables de comunicación y tuberías son vitales para la economía global y la seguridad nacional. Sin embargo, su protección es limitada y, eventos recientes han demostrado su vulnerabilidad. Recientes incidentes en el Mar Báltico han puesto de manifiesto la necesidad de reforzar la vigilancia y defensa de estas estructuras frente a posibles sabotajes o accidentes[27].
- Tensiones geopolíticas y amenazas emergentes: El panorama geopolítico actual introduce riesgos adicionales para la seguridad marítima. Conflictos regionales y disputas territoriales pueden derivar en ataques a embarcaciones civiles o bloqueos de rutas comerciales estratégicas. Por ejemplo, los ataques en las costas yemeníes del Mar Rojo han obligado a las navieras a modificar sus rutas, evitando el Canal de Suez y optando por rutas más largas y costosas.
- Limitaciones de recursos y mantenimiento: La disponibilidad y el estado operativo de los recursos dedicados a la vigilancia marítima son factores críticos. Incidentes como la avería del patrullero “Río Flumen” de la Guardia Civil, evidencian cómo las limitaciones en mantenimiento y recursos pueden afectar de manera negativa a la eficacia operativa[28].
- Aumento del tráfico marítimo y congestión de rutas: La globalización y el crecimiento de flotas comerciales, pesqueras y recreativas han incrementado la densidad del tráfico marítimo, especialmente en estrechos y zonas de alto tránsito. Esto dificulta la identificación de amenazas y exige mejoras en los sistemas de control, comunicación y vigilancia.
- Ciberseguridad y amenazas electrónicas: La dependencia de sistemas interconectados y automatizados en la vigilancia marítima ha incrementado la vulnerabilidad ante ataques cibernéticos y guerra electrónica. Actores hostiles pueden manipular señales de GPS, interferir con los sistemas de monitoreo o comprometer las comunicaciones de buques y centros de control. La protección de infraestructuras digitales y el fortalecimiento de la resiliencia ante ciberataques se han convertido en una prioridad estratégica.

2.1.4 Tendencias futuras

Las tendencias futuras están marcadas por la incorporación de tecnologías avanzadas que optimizan la detección de amenazas y la gestión de datos. Entre ellas destaca el uso de redes neuronales, el cual ha demostrado un gran potencial para la detección automática de amenazas, permitiendo el análisis en tiempo real de datos provenientes de radares, sensores, drones y satélites.

El módulo GAIA AI utiliza, como muestra la Figura 2-4, tres redes neuronales diseñadas para reconocer patrones en aplicaciones como la vigilancia marítima, terrestre y aérea, posibilitando así una detección temprana, el seguimiento y la clasificación de una amplia variedad de amenazas[29]. Además, sistemas como el desarrollado recientemente por Airbus y la UPM emplean técnicas de aprendizaje profundo para automatizar el análisis de datos sobre la posición y actividad de barcos sin sistemas de identificación automática, contribuyendo así a la lucha contra la pesca ilegal y otras actividades ilícitas[30].

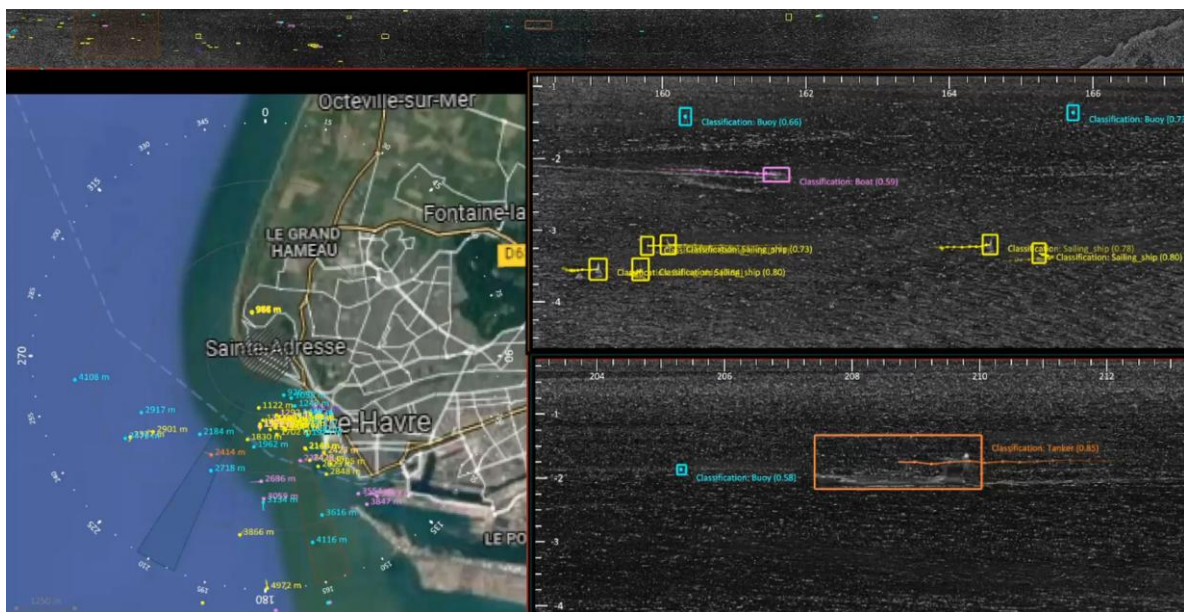


Figura 2-4 Visualización sistema GAIA IA [29]

Por otro lado, la aplicación de la tecnología blockchain en la gestión de datos marítimos está emergiendo como una solución pionera para mejorar la trazabilidad de la información en este ámbito. Blockchain permite la creación de registros inmutables y descentralizados, reduciendo significativamente el riesgo de manipulación o falsificación de datos relacionados con el tráfico marítimo, certificación de embarcaciones y operaciones portuarias[31].

Una de las aplicaciones más contemporáneas es la combinación de satélites de observación con inteligencia artificial. Esto permite un análisis más preciso y automatizado del tráfico marítimo, mejorando la detección de embarcaciones no registradas y optimizando la gestión del espacio marítimo. La integración de sistemas multisensor y algoritmos de inteligencia artificial permite generar alertas tempranas y mejorar la capacidad de respuesta operativa ante eventos no previstos.

Junto con estas innovaciones, otras tecnologías emergentes están contribuyendo al avance de la vigilancia marítima. Prueba de ellos es un nuevo sistema multisensor combinado con IA utilizado para mejorar la seguridad marítima al detectar e identificar objetos que suponen una amenaza de colisión para los buques[32]. Asimismo, el desarrollo de drones autónomos, equipados también con IA, está potenciando la capacidad de vigilancia de grandes extensiones oceánicas, permitiendo patrullajes continuos y coordinados sin necesidad de supervisión constante.

El IoT marítimo está implantándose en el ámbito naval, lo que conduce a una mejor recopilación y análisis de datos en tiempo real, proporcionando información detallada sobre el tráfico marítimo, las condiciones meteorológicas y la presencia de posibles amenazas. Dispositivos IoT como boyas inteligentes, rastreadores GPS y embarcaciones equipadas con sensores, permiten la recopilación y transmisión de datos en tiempo real, mejorando la monitorización del tráfico y la detección de actividades sospechosas.

2.2 Optimización

2.2.1 Concepto de la optimización

La optimización es un concepto clave en una amplia variedad de disciplinas, desde la ingeniería y la economía hasta la biología y las ciencias sociales. Como concepto, la optimización se refiere al proceso de hacer cualquier acción lo más eficientemente posible. En términos operativos, la optimización implica maximizar la eficiencia en la asignación de recursos bajo restricciones técnicas y logísticas. En el contexto empresarial, la optimización puede significar mejorar la eficiencia operativa para aumentar la rentabilidad.

En el ámbito de la ingeniería, la optimización puede implicar el diseño de sistemas y procesos que funcionen con la mayor eficacia y eficiencia posibles. En la biología, la optimización puede observarse en la evolución de las especies, donde los organismos desarrollan características clave que les permiten sobrevivir y reproducirse con mayor éxito en su entorno[33].

Las matemáticas juegan un papel fundamental en cualquier proceso de optimización, especialmente en el desarrollo de algoritmos que pueden resolver problemas más complejos de optimización. Los métodos de programación lineal, no lineal y dinámica son herramientas poderosas utilizadas para encontrar soluciones óptimas en una variedad de problemas. Por ejemplo, los algoritmos de optimización utilizados en la planificación de rutas para minimizar el tiempo y la distancia de los viajes, en la asignación de recursos en proyectos para maximizar la eficiencia, y en la gestión de carteras de inversión para equilibrar el riesgo y el rendimiento[34].

Sin embargo, la optimización no es siempre un método sencillo. A menudo, al igual que ocurrirá en este trabajo fin de grado, los problemas de optimización se dificultan por la presencia de múltiples objetos en conflicto, restricciones e incertidumbres. En tales casos, encontrar una solución óptima puede requerir de técnicas avanzadas como la optimización multiobjetivo, optimización bajo incertidumbre y los algoritmos evolutivos[35].

2.2.2 Optimización de función objetivo con restricciones

Un problema matemático de optimización consiste en encontrar el valor óptimo (mínimo o máximo) de una función objetivo $f(x)$ dentro de un conjunto factible definido por un conjunto de restricciones. Matemáticamente, se define como muestra la ecuación (2-1).

$$\min_{x \in \mathbb{R}^n} f(x) \text{ o } \max_{x \in \mathbb{R}^n} f(x) \quad (2-1)$$

Sujeto a las restricciones de la ecuación (2-2).

$$\begin{aligned} g_i(x) &\leq 0, \quad i=1, \dots, m \\ h_j(x) &= 0, \quad j=1, \dots, p \end{aligned} \quad (2-2)$$

Donde:

- $x = (x_1, x_2, \dots, x_n)$
- $f(x): \mathbb{R}^n \rightarrow \mathbb{R}$ es la función objetivo que se busca optimizar

- $g_i(x)$ son las restricciones de desigualdad, las cuales limitan la región factible en la que se puede buscar solución óptima.
- $h_j(x)$ son las restricciones de igualdad, que también definen el conjunto de soluciones factibles.

Dependiendo de la forma de $f(x)$, $g_i(x)$ y $h_j(x)$, los problemas de optimización pueden clasificarse en diferentes categorías:

- Optimización lineal: Si la función objetivo y/o las restricciones son funciones lineales, tendrían la forma de la ecuación (2-3).

$$f(x) = c^T x, \quad g_i(x) = a_i^T x - b_i, \quad h_j(x) = d_j^T x - e_j \quad (2-3)$$

Se resuelve normalmente mediante el Método Simplex o algoritmos de programación lineal.

- Optimización no lineal: Cuando la función objetivo y/o las restricciones contienen términos no lineales. Pueden resolverse con métodos como el gradiente descendente, Newton-Raphson, o algoritmos de optimización convexa.
- Optimización entera: Si las variables x solo pueden tomar valores enteros, el problema se vuelve más complejo y se resuelve con técnicas de ramificación y acotación (branch and bound) o algoritmos metaheurísticos.
- Optimización convexa: Cuando la función objetivo y las restricciones son convexas, garantizando un único óptimo global. Se utilizan métodos como programación cuadrática o métodos de punto interior.
- Optimización dinámica: Utilizado cuando el problema involucra una secuencia de decisiones a lo largo del tiempo, resolviéndose con propagación dinámica o modelos estocásticos.

2.2.3 Aplicado a la vigilancia marítima

La optimización desempeña un papel fundamental en la vigilancia marítima, mejorando la eficiencia en la detección, monitoreo y respuesta ante amenazas en la mar. Para lograr optimizar este proceso se emplean diferentes métodos matemáticos en los procesos de planificación de rutas de patrullaje, asignación de recursos y el procesamiento de datos de sensores.

Uno de los principales desafíos en la vigilancia marítima es la cobertura efectiva de grandes extensiones de océano dentro de un escenario de recursos limitados. La optimización de rutas de patrullaje, por su parte, permite que embarcaciones, aeronaves, drones y sensores cubran la mayor área posible con la menor cantidad de recursos y tiempo. En este caso, algoritmos como la optimización basada en colonias de hormigas o la programación lineal ayudan a encontrar trayectorias óptimas para minimizar estos costos operativos y maximizar la probabilidad de detección de actividades sospechosas[36].

Una de las áreas donde la optimización juega un papel determinante en la vigilancia marítima es a la hora de asignar recursos. Las fuerzas de seguridad marítima deben distribuir patrullas, personal y equipos de manera eficiente para responder a los incidentes en el menor tiempo posible. Modelos de optimización estocástica permiten considerar la incertidumbre en la ubicación y el comportamiento de embarcaciones sospechosas, ayudando a predecir patrones y mejorar la asignación de vigilancia en zonas de alto riesgo.

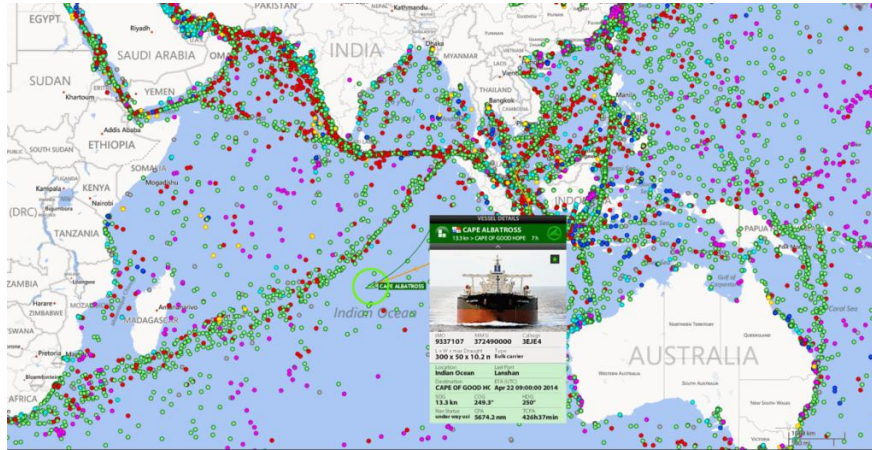


Figura 2-5 Sistema de identificación automática (AIS) [37]

Otro de los aspectos de la vigilancia marítima en los que la optimización es crucial es el análisis de datos. Con el uso de satélites, radar, y sistemas de identificación automática (AIS), como refleja la Figura 2-5, se recopila una gran cantidad de información sobre el tráfico y el entorno marítimo. Técnicas de optimización y aprendizaje automático permiten filtrar y priorizar datos relevantes, ayudando a detectar anomalías y prever actividades ilícitas con mayor precisión y probabilidad. En este ámbito, la combinación de sistemas de radar y sensores con algoritmos de IA ha permitido desarrollar soluciones innovadoras para la vigilancia marítima, mejorando la detección y monitoreo de amenazas[38].

2.3 El método simplex

2.3.1 Definición y fundamentos

El método simplex es una tecnología algebraica y gráfica usada para resolver problemas de programación lineal. Esta metodología, desarrollada por George Dantzig en 1947, se ha convertido en una herramienta fundamental en el ámbito de investigación operativa y en la optimización de recursos en diferentes áreas como la industria, la economía y la ingeniería[39].

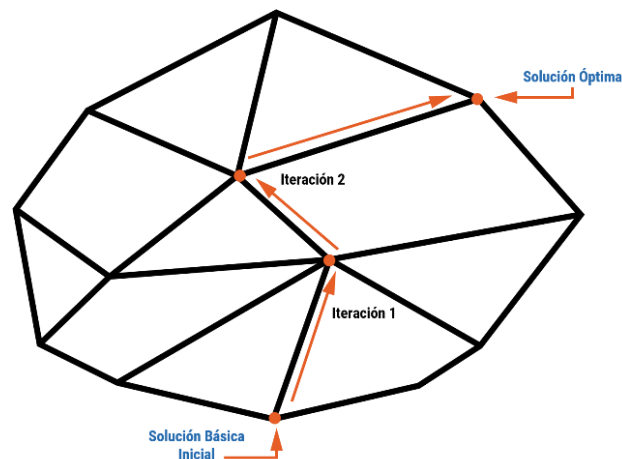


Figura 2-6 Representación gráfica del método simplex [40]

Este es un método iterativo que permite mejorar la solución en cada paso. La razón matemática de esta mejora reside en que el método consiste en avanzar desde el vértice de un poliedro hasta un vértice vecino, como se representa en la Figura 2-6, de manera que aumente o disminuya, según el contexto de la función objetivo, ya sea maximizar o minimizar[40].

La importancia de la teoría de matrices en el método simplex es fundamental, ya que el algoritmo se basa en esta teoría para resolver problemas.

Una matriz puede ser definida como una ordenación rectangular de elementos, ya sean números reales o complejos, que están dispuestos en filas y columnas. La matriz identidad es una matriz cuadrada; es decir, tiene el mismo número de filas y columnas, de orden n , con todos los elementos diagonales iguales a 1 y todos los demás elementos iguales a 0. Se denota así la matriz identidad de orden n .

2.3.2 Consideraciones al utilizar el método simplex

2.3.2.1 Variables de holgura y exceso

Este método trabaja con ecuaciones, y las restricciones iniciales modeladas mediante programación lineal no lo son. Para convertir estas inecuaciones en ecuaciones se utilizan variables de holgura y exceso relacionadas con los recursos de las restricciones. En el tabulado final estas variables representan el “Slack or surplus” en los programas de resolución de investigación de operaciones, y son fundamentales en el análisis de sensibilidad y la creación de las anteriormente mencionadas matrices identidad[39].

Por tanto, las variables de holgura se suman si la restricción es de signo $\leq$ y se restan si la restricción es de signo $\geq$. Ambas del lado izquierdo de la restricción.

Una de las consideraciones más importantes que se debe tener en cuenta en el sistema es que debe ser restrictivo; por lo que el lado derecho de las restricciones no puede albergar variables, tan solo un número mayor o igual a 0. Por ejemplo, como refleja la ecuación (2-4), tenemos que:

$$3X_1 + 4X_2 + 2X_3 + 1X_4 \leq -16 \quad (2-4)$$

Convertimos la desigualdad en igualdad añadiendo una variable de holgura tal y como aparece en la ecuación (2-5).

$$3X_1 + 4X_2 + 2X_3 + 1X_4 + 1S_1 = -16 \quad (2-5)$$

Multiplicamos ambos lados por (-1) para conseguir que el lado derecho cumpla con la condición de ser un número mayor o igual que 0, quedando la ecuación (2-6).

$$-3X_1 - 4X_2 - 2X_3 - 1X_4 - 1S_1 = 16 \quad (2-6)$$

2.3.2.2 Variable artificial o método de la M

Una variable artificial es utilizada para convertir inecuaciones $\geq$ en ecuaciones, o también pueden ser utilizadas cuando existen igualdades en el problema original. Estas variables no deben formar parte de la solución ya que no representan recursos. Su objetivo es la formación de la matriz identidad. Son representadas con la letra A , se suman a las restricciones y su coeficiente resultante se denomina M . Cuando se abordan problemas en los que la optimización que se busca es la maximización se utiliza signo (-), y en problemas en los que se busca minimización se utiliza el signo (+); todo esto buscando que el valor de la solución sea 0[41].

2.3.2.3 Variables no negativas

Todas las variables de este método han de cumplir la condición de no negatividad. Si una variable no tiene restricción de no negatividad, se reemplaza por la diferencia de dos variables positivas. Si X_i es una variable sin restricción de no negatividad, tiene que ser cambiada por $X_i (+) - X_i (-)$, donde $X_i (+) \geq 0$ y $X_i (-) \geq 0$. En la práctica estas variables son poco comunes y son utilizadas principalmente en la programación por metas[39].

2.3.3 Aplicaciones del Método Simplex

Sus aplicaciones abarcan desde la optimización de procesos industriales hasta la gestión de la cadena de suministro y la logística, pasando por la optimización de carteras de inversión en el sector financiero. También es utilizado en la agricultura para maximizar rendimientos y minimizar costos, y en el ámbito sanitario para mejorar la asignación de recursos hospitalarios. La versatilidad y eficiencia del Método Simplex lo convierten en una herramienta imprescindible para mejorar la eficiencia operativa y la toma de decisiones en múltiples campos. El Método Simplex destaca principalmente por su aplicación en:

- **Industria y manufactura:** Las empresas utilizan el Método Simplex para optimizar la producción y la distribución de productos, determinando la combinación óptima de productos que maximiza las ganancias o minimiza los costos, teniendo en cuenta las restricciones de recursos como materia prima, mano de obra y capacidad de producción.
- **Logística:** El Método Simplex es crucial para resolver problemas de ruteo de vehículos, asignación de flota y planificación de redes de transporte. De esta manera permite a las empresas minimizar costos de portes y mejorar la eficiencia en el reparto de productos y servicios.
- **Sector financiero:** En la optimización de carteras de inversión, el Método Simplex es utilizado para maximizar el retorno esperado mientras se minimiza el riesgo, cumpliendo con los límites de inversión y las políticas de riesgo.
- **Agricultura:** Los agricultores deciden en base a este método la mejor combinación de cultivos que maximice el rendimiento y minimice los costos de producción, considerando factores limitantes como la disponibilidad de la tierra, el agua y otros productos consumibles agrícolas.
- **Gestión de proyectos:** Se recurre al Método Simplex principalmente para problemas de programación y asignación de tareas, así como para la planificación de proyectos complejos. En el área de salud por ejemplo, este método ayuda en gran parte la labor de optimizar recursos hospitalarios; como la asignación de camas, personal médico y equipos, mejorando la eficiencia del servicio y reduciendo costes[42].
- **En este trabajo fin de grado** se va a implementar el Método Simplex para optimizar un proceso de vigilancia marítima en un escenario de recursos limitados, maximizando la eficiencia de la vigilancia marítima en cada área atendiendo a sus características y peculiaridades, y minimizando el costo de dicha vigilancia al reducir la cantidad de días en la mar. Los factores limitantes con los que trabajaremos serán parámetros que afectan la operatividad y disponibilidad real de un buque, como pueden ser la autonomía, la disponibilidad de la dotación o la capacidad de nuestra plataforma para localizar los diferentes sucesos que planteemos.

Debido a su eficiencia y robustez en la resolución de problemas de programación lineal, el método Simplex se ha consolidado como una herramienta fundamental en ámbitos como la asignación de recursos, la planificación de rutas, la logística operativa y, en general, en cualquier contexto donde sea necesario optimizar funciones lineales sujetas a un conjunto de restricciones lineales.

2.4 El método fmincon

El método fmincon de MATLAB es una función especial de optimización que se utiliza para encontrar el valor mínimo de una función no lineal sujeta a restricciones. Es parte de Optimization Toolbox y está diseñado para resolver problemas de programación no lineal con restricciones lineales y no lineales, así como límites en las variables[43].

Fmincon busca minimizar una función objetivo definida previamente por el usuario, aceptando un vector o matriz x y devuelve un escalar real f , evaluado en x . El usuario proporciona una estimación inicial x_0 , y fmincon ajusta los valores de x para encontrar el mínimo de la función objetivo, respetando las restricciones impuestas[43].

La sintaxis básica de fmincon es:

$$x = \text{fmincom}(\text{fun}, x_0, A, b, Aeq, beq, lb, ub, \text{nonlcon}, \text{options})$$

Donde:

fun: Función objetivo a minimizar.

x_0 : Punto de inicio para la búsqueda.

A, b : Matrices para restricciones lineales de la forma $A \cdot x \leq b$.

Aeq, beq : Matrices para restricciones lineales de igualdad $Aeq \cdot x = beq$.

lb, ub : Límites inferiores y superiores para las variables.

nonlcon: Función que define restricciones no lineales.

options: Estructura que especifica opciones de optimización, como el algoritmo a utilizar.

fmincon en comparación con el Método Simplex, como se detalla en la Tabla 2-1, puede aplicarse en el contexto de la optimización de la vigilancia marítima modelando el problema como un a optimización no lineal con restricciones, donde se busca minimizar el costo operativo, el tiempo de respuesta o el tiempo dedicado a cada patrullaje, estando sujeto a restricciones como las capacidades del patrullero, la autonomía de la plataforma o el tiempo máximo en la mar de la dotación.

2.4.1 Algoritmos subyacentes

fmincon ofrece varios algoritmos para resolver problemas de optimización, como son:

- Región de confianza reflectante: Este algoritmo es adecuado para problemas grandes con restricciones simples. Utiliza métodos de región de confianza para aproximar la función objetivo en una vecindad alrededor del punto actual, y busca una dirección de descenso que minimice esta aproximación.
- Región de confianza con restricciones: Similar a la región de confianza reflectante, pero manejando restricciones más complejas y es adecuado para problemas de tamaño mediano.
- Programación cuadrática secuencial (SQP): Resuelve una secuencia de subproblemas de programación cuadrática, cada uno de los cuales aproxima el problema original. Es eficaz para problemas no lineales con restricciones no lineales.
- Algoritmo interior-punto: Útil para problemas grandes y dispersos. Utiliza una técnica de barrera para manejar las restricciones y busca soluciones en el interior del conjunto factible.
-

2.4.2 Elección de *fmincon*

El método *fmincon* de MATLAB ha sido seleccionado para el problema de este trabajo debido a la naturaleza no lineal de la función objetivo utilizada en la optimización del patrullaje marítimo. A diferencia del método Simplex, que es adecuado para problemas de programación lineal, *fmincon* permite manejar funciones objetivo y restricciones que presentan términos no lineales.

En este caso, la función objetivo incluye términos cuadráticos que penalizan de forma diferencial la asignación de tiempo extra en cada zona, dependiendo de factores como la criticidad operativa, las condiciones meteorológicas y la capacidad de detección de la unidad de vigilancia.

El carácter no lineal del problema se debe a la relación entre el tiempo extra asignado a cada zona y los coeficientes de ponderación que incluyen factores como la probabilidad de ocurrencia de sucesos de interés, la influencia de las condiciones ambientales y la importancia estratégica de cada región. Estos coeficientes generan interacciones multiplicativas que no pueden ser manejadas por métodos de optimización lineal. Además, el uso de restricciones de igualdad e inecuaciones en el modelo, tales como la limitación del tiempo total de patrullaje y la asignación mínima requerida por zona, refuerzan la necesidad de un método de optimización capaz de gestionar condiciones no lineales de manera eficiente.

En comparación con alternativas más simples como el método Simplex, *fmincon* permite una mayor flexibilidad en la formulación del problema y asegura la convergencia a soluciones óptimas en escenarios donde la función objetivo y las restricciones presentan complejidad estructural.

La elección de este método garantiza que la asignación de tiempos de patrullaje se realice de manera óptima bajo condiciones operativas realistas, maximizando la cobertura de vigilancia y optimizando el uso de los recursos disponibles.

	fmincon	Simplex
Tipo de problema	No lineales con restricciones	Problemas lineales
Algoritmo	Región de confianza, SQP, interior-punto	Simplex
Manejo de restricciones	Soporta restricciones lineales y no lineales, así como límites en variables	Solo restricciones lineales
Ventajas	Flexibilidad para manejar problemas complejos y no lineales	Eficiente para problemas lineales
Desventajas	Requiere buenas estimaciones iniciales	Ineficiencia con grandes problemas

Tabla 2-1 Comparativa *fmincon* y Método Simplex

En base a estas características, *fmincon* se presenta como una herramienta especialmente adecuada para abordar el problema planteado, al permitir la resolución de funciones objetivo no lineales con restricciones tanto lineales como no lineales, ofreciendo además múltiples algoritmos internos que pueden adaptarse a las particularidades del modelo de optimización implementado.

2.5 Herramientas computacionales

2.5.1 Definición y contexto

Las herramientas computacionales juegan un papel crucial en el mundo moderno, especialmente cuando nos referimos a optimización. Las herramientas computacionales disponibles hoy en día han revolucionado enormemente la manera en que abordamos y resolvemos problemas complejos de optimización, abarcando una grandísima cantidad de datos e información, y permitiéndonos encontrar soluciones más rápidas, precisas y eficientes.

2.5.2 Herramientas computacionales evaluadas

Entre las herramientas más destacadas se encuentran software de modelado matemático como MATLAB, GAMS y LINGO. MATLAB es ampliamente conocido en el ámbito de la ingeniería y ciencias para el análisis de datos, simulaciones y procesos de optimización. Su versatilidad y capacidad para manejar grandes volúmenes de datos lo hacen un software idóneo para resolver problemas complejos. GAMS y LINGO, por otro lado, son herramientas que destacan por su gran implementación en la industria por su gran capacidad para resolver problemas de programación lineal, no lineal y entera.

Algoritmo	Características	Ventajas	Desventajas
Búsqueda heurística [44]	Estrategias basadas en reglas empíricas	Rápida ejecución. Útil con información parcial	No garantiza la mejor solución
Búsqueda metaheurística [45]	Combinan varias heurísticas para explorar grandes espacios de búsqueda	Adaptabilidad. Problemas complejos y de gran escala	Computacionalmente costoso
Algoritmos genéticos [45]	Inspirados en evolución biológica	Explorar grandes espacios y adaptarse a distintos problemas	Alto consumo de recursos computacionales
Enjambre de partículas [45]	Comportamiento colectivo de agentes que comparten información	Rápida convergencia	Puede quedar atrapado en óptimos locales

Tabla 2-2 Comparación de los diferentes algoritmos de optimización [44] [45] [46]

Otra herramienta importante en el ámbito de la optimización es el uso de algoritmos de búsqueda heurística y metaheurística. Los algoritmos genéticos, la optimización por enjambre de partículas y los algoritmo de recocido simulado son técnicas que se utilizan para resolver problemas de optimización en los que los métodos tradicionales no son eficaces. Estos algoritmos son especialmente útiles en problemas con múltiples soluciones locales, donde encontrar la solución óptima global puede ser computacionalmente intensivo. Se comparan dichos algoritmos para su aplicación en un problema de optimización en escenario con recursos limitados obteniendo la Tabla 2-2 como resultado.

En la optimización basada en datos, las herramientas de Machine Learning han cobrado gran relevancia. Frameworks como TensorFlow, PyTorch y scikit-learn permiten la implementación de modelos de aprendizaje automático que pueden ser utilizados para la optimización en diferentes áreas, tales como la predicción, clasificación y reconocimiento de patrones. En la optimización de una cadena

de suministro, por ejemplo, los modelos de Machine Learning pueden predecir la demanda futura y optimizar los niveles de inventario, minimizando los costos y mejorando la eficacia operativa.

Por otro lado, las herramientas de simulación como Simulink y Arena también juegan un papel esencial en los procesos de optimización, especialmente en la ingeniería de sistemas. Estas herramientas permiten la creación de modelos de simulación que representan procesos físicos y sistemas complejos. A través de la simulación se pueden identificar cuellos de botella y áreas de mejora, permitiendo así la optimización del diseño y operación del sistema.

Existen también software de análisis y visualización de datos, como Excel, Tableau y Power BI, ampliamente utilizados en la optimización basada en datos. Estas herramientas permiten a los usuarios analizar grandes volúmenes de datos y visualizar resultados de manera comprensible, facilitando de esta manera la toma de decisiones en base a la optimización.

Finalmente, las plataformas computacionales en la nube, como AWS, Google Cloud y Azure, han facilitado enormemente el acceso a recursos computacionales avanzados necesarios para la optimización. Estas plataformas ofrecen servicios de computación de alto rendimiento y almacenamiento masivo de datos, permitiendo a las organizaciones ejecutar modelos de optimización complejos sin necesidad de una infraestructura local costosa.

2.5.2.1 GAMS

El General Algebraic Modeling System (GAMS) es un software de alto nivel diseñado para modelar y resolver problemas de optimización matemática. Fue desarrollado por primera vez en 1978 y se ha convertido en una herramienta esencial en campos como la economía, ingeniería y la investigación operativa. GAMS permite a los usuarios construir modelos algebraicos complejos y resolver problemas lineales, no lineales y de optimización entera mixta.

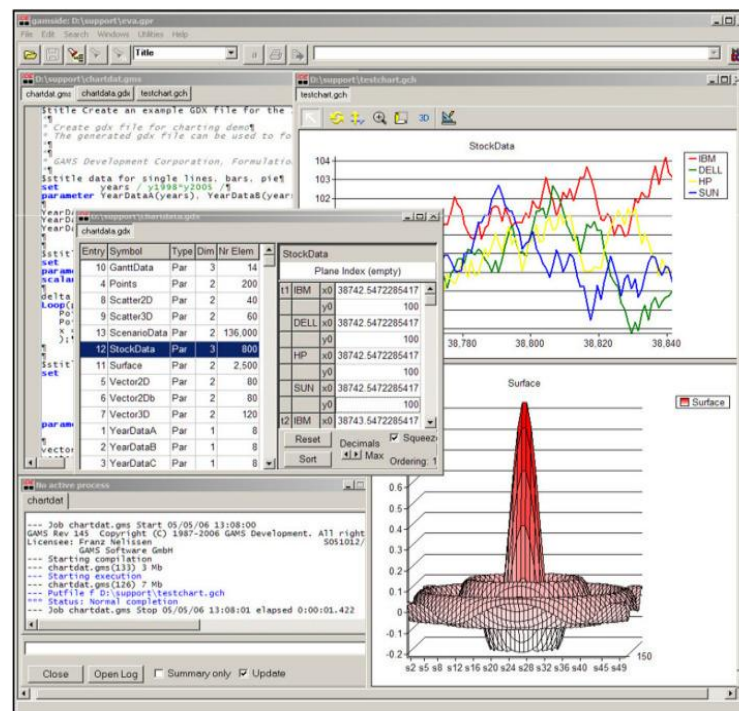


Figura 2-7 Interfaz del software GAMS [48]

GAMS se integra además con Python a través de GAMSPy, lo que permite a los usuarios aprovechar la flexibilidad del lenguaje Python para crear modelos matemáticos complejos. Esta integración facilita la implementación de modelos en entornos de computación en la nube, como AWS y Google Cloud, lo que posibilita la resolución de problemas de optimización a gran escala de manera eficiente[49].

Una de las características más destacadas de este software es su capacidad para trabajar con múltiples solucionadores de terceros, como BARON, CONOPT, CPLEX, Gurobi, Mosek, SNOPT y XPRESS. De esta manera se permite a los usuarios implementar algoritmos híbridos que combinan diferentes solucionadores para encontrar una solución óptima. Además, GAMS ofrece un entorno de desarrollo integrado (IDE) que facilita la creación, edición y solución de modelos[47] como se observa en la Figura 2-7 donde se muestra la interfaz del software.

2.5.2.2 LINGO

LINGO es un software de modelado y optimización que es utilizado para resolver problemas de programación lineal, no lineal, y entera. Desarrollado por LINDO Systems, este software proporciona una herramienta completa para la formulación y resolución de modelos de optimización. La versión más reciente, LINGO 21, incluye mejoras significativas y nuevas características que facilitan la creación y solución de modelos[50].

Una de las principales ventajas de este software es su capacidad para expresar modelos de manera intuitiva y legible, utilizando para ello una sintaxis similar a la que se usaría en papel, como puede verse en la Figura 2-8. Esto hace que los modelos sean más fáciles de construir, entender y mantener. Además LINGO puede manejar grandes volúmenes de datos y extraer información directamente de bases de datos y hojas de cálculo.

LINGO también incluye una serie de solucionadores integrados que seleccionan automáticamente el solucionador adecuado según la formulación del modelo. Esto elimina la necesidad de especificar o cargar solucionadores separados, lo que simplifica el proceso de optimización. Además, LINGO permite la creación de aplicaciones interactivas y la integración con otros programas a través de interfaces DLL y OLE[50].

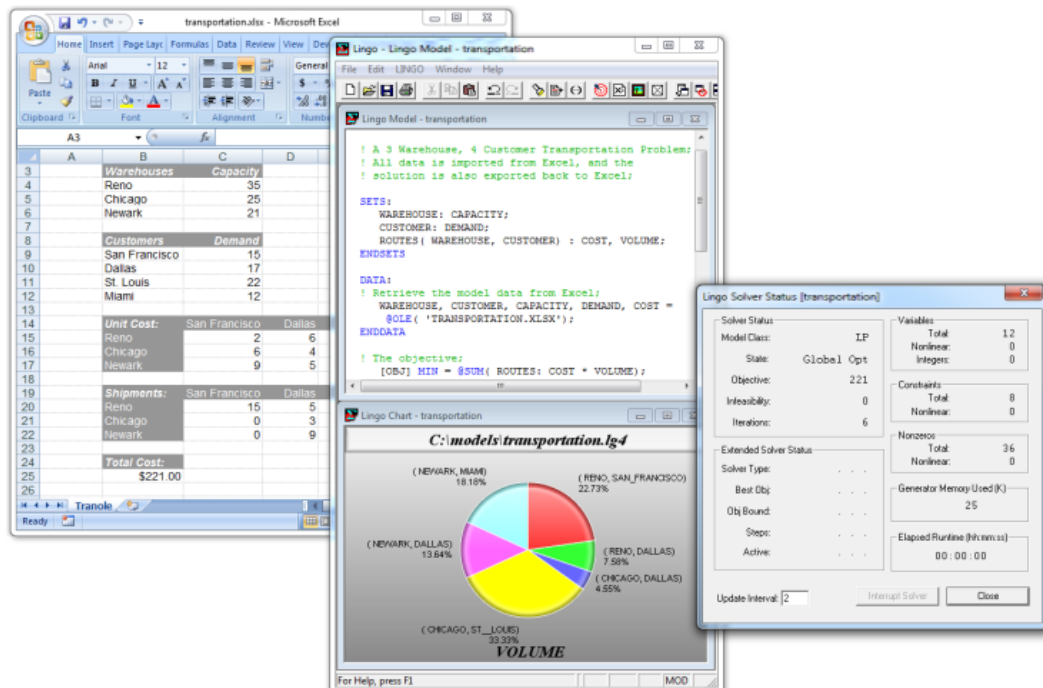


Figura 2-8 Interfaz del software LINGO [51]

2.5.2.3 MATLAB

MATLAB (MATrix LABoratory) es un software de programación y un entorno de desarrollo integrado que fue creado por la compañía MathWorks en 1984. El software está diseñado principalmente para el cálculo numérico, la visualización de datos y el desarrollo de algoritmos. MATLAB ofrece una vasta gama de herramientas y funciones que facilitan la implementación y prueba de modelos matemáticos y algoritmos complejos. Su lenguaje de programación es conocido por ser intuitivo y de alto nivel, lo que permite a los usuarios focalizar sus esfuerzos más en la resolución del problema y menos en la sintaxis del código.

Una de las principales ventajas del software MATLAB es su capacidad para manejar matrices y arrays, que son fundamentales en el análisis y procesamiento de datos. Su entorno de trabajo incluye funciones predefinidas para realizar operaciones matemáticas tanto básicas como avanzadas, gráficos en 2D y 3D, procesamiento de señales, análisis de sistemas y diseño de controladores, entre otros. Estas características hacen de MATLAB una herramienta imprescindible en disciplinas como la ingeniería, la física, la economía o la biología.

MATLAB además cuenta con diversas herramientas adicionales y toolboxes especializados que amplían sus capacidades. El toolbox de Machine Learning, por ejemplo, facilita la creación y entretenimiento de modelos de aprendizaje automático, mientras que el toolbox de Simulink permite la simulación y modelado de sistemas dinámicos. Estas herramientas adicionales proporcionan un entorno versátil y potente para el análisis y la optimización en una amplia gama de aplicaciones.

Por su parte, el entorno gráfico del programa, mostrado en la Figura 2-9, también es una característica a destacar, ya que permite a los usuarios crear visualizaciones de datos personalizados y de alta calidad. Esto es particularmente útil para la presentación de resultados y la comunicación de hallazgos en informes y publicaciones académicas. Los gráficos de MATLAB pueden ser fácilmente exportados a diferentes formatos, facilitando así su integración en documentos y presentaciones.

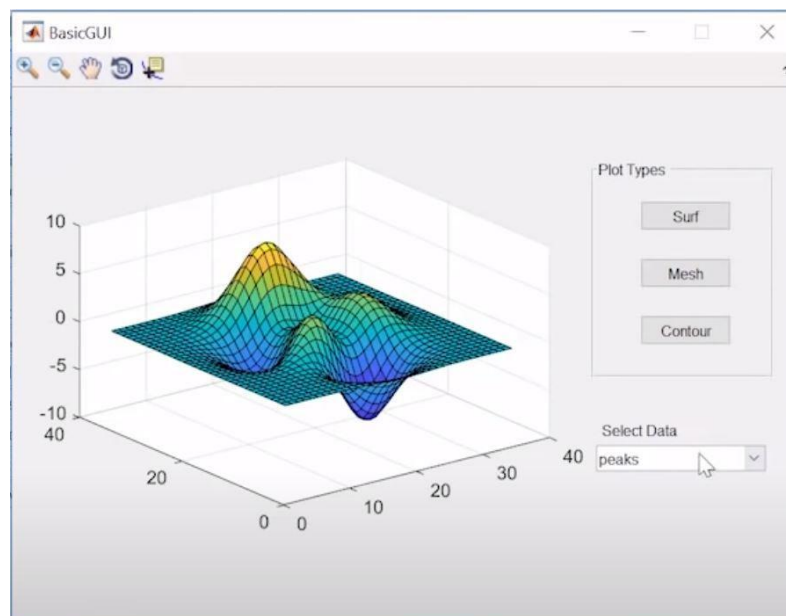


Figura 2-9 Entorno gráfico de MATLAB [52]

MATLAB es una herramienta ampliamente utilizada en la resolución de problemas de optimización debido a su capacidad para manejar grandes volúmenes de datos, realizar cálculos numéricos avanzados y visualizar resultados de manera efectiva. Su flexibilidad permite la implementación de distintos enfoques matemáticos para la optimización, incluyendo programación lineal, no lineal, entera y estocástica.

Una de las principales ventajas de MATLAB en la optimización es su facilidad para formular y resolver problemas con restricciones, gracias a funciones como `fmincon`, `linprog` y `quadprog`, que permiten minimizar o maximizar funciones objetivo bajo diversas condiciones. También es posible emplear algoritmos heurísticos y metaheurísticos, como algoritmos genéticos o recocido simulado, para encontrar soluciones aproximadas en problemas donde los métodos tradicionales pueden no ser viables.

Otra ventaja clave es la integración con otras herramientas y lenguajes de programación, lo que facilita la automatización de procesos y la implementación de soluciones híbridas combinando optimización y aprendizaje automático. Además, su capacidad para representar visualmente los resultados a través de gráficos en 2D y 3D, como se muestra en la Figura 2-9, permite interpretar de manera clara la efectividad de las soluciones obtenidas y realizar ajustes en los modelos según sea necesario.

En el contexto de la vigilancia marítima, MATLAB permite modelar y analizar escenarios de patrullaje, optimizando la asignación de recursos y minimizando costos operativos. Su capacidad de simulación facilita la evaluación de diferentes estrategias de patrullaje antes de su implementación real, mejorando así la planificación y la toma de decisiones en tiempo real. Gracias a estas ventajas, MATLAB se convierte en una herramienta esencial para abordar problemas de optimización en entornos dinámicos y con restricciones operativas complejas, por delante de softwares como LINGO o GAMS, como se aprecia en la comparación entre softwares de la Tabla 2-3.

Aspecto	MATLAB	GAMS	LINGO
Tipo de problemas	Simulaciones, análisis numérico y optimización	Optimización en logística y producción	Modelado de decisiones en ingeniería
Optimización	Lineal, no lineal, heurística e IA	Lineal, entera, mixta y no lineal	Lineal, no lineal y estocástica
Precisión y robustez	Alta	Alta en optimización, menor en simulaciones	Menor precisión en cálculos complejos
Modelado matemático	Ecuaciones diferenciales y sistemas dinámicos	Algebraico sin simulaciones físicas	Modelado algebraico con menos herramientas de simulación
Interfaz y visualización	Avanzada con gráficos interactivos	Básica, basada en texto	Interfaz sencilla para optimización
Manejo de datos	Procesa grandes volúmenes con herramientas avanzadas	Procesa grandes conjuntos, menos en visualización	Moderado, mejor en problemas estructurados

Tabla 2-3 Comparativa herramientas computacionales evaluadas

3 DESARROLLO DEL TFG

3.1 Formulación del problema

3.1.1 Descripción del problema de optimización en vigilancia marítima

La vigilancia marítima es una actividad crítica para la seguridad y defensa de un país, especialmente en zonas estratégicas donde convergen múltiples intereses económicos, políticos y de seguridad. En el caso de la Armada, y de este trabajo fin de grado en particular, el desafío radica en patrullar eficientemente áreas marítimas con recursos limitados, asegurando una cobertura óptima y maximizando la probabilidad de detección de eventos de interés, como actividades ilícitas, tráfico de embarcaciones sospechosas y amenazas potenciales.

El problema de optimización de este trabajo fin de grado se plantea como la asignación del tiempo de patrullaje en diferentes regiones dentro de un área determinada, teniendo en cuenta las características específicas de cada zona y los recursos disponibles. En este contexto, el tiempo total de patrullaje es una variable crítica, ya que viene determinado como la autonomía de los buques, la disponibilidad, la disponibilidad de la dotación y las condiciones meteorológicas. El objetivo es encontrar una distribución óptima del tiempo de patrullaje que maximice la efectividad de la vigilancia, minimizando el riesgo de actividades no detectadas y optimizando el uso de los medios disponibles.

Para resolver este problema, se emplean técnicas de optimización matemática que permiten modelar la distribución de los recursos de manera eficiente. En este proyecto se ha optado por el uso del algoritmo `fmincon` de MATLAB, el cual es adecuado para resolver problemas de optimización con restricciones no lineales.

Este planteamiento permite definir una función objetivo que minimiza la dispersión del tiempo de patrullaje respecto a la distribución óptima, considerando restricciones como la disponibilidad de recursos y las condiciones operativas. Además, el modelo incorpora factores como la influencia del clima en la operatividad de las unidades y la criticidad de cada zona en función de datos históricos de actividad sospechosa.

3.1.2 Objetivos y restricciones del modelo

El objetivo principal del modelo es determinar la distribución óptima del tiempo de patrullaje en un área determinada, asegurando que las zonas de mayor criticidad reciban mayor atención sin superar los recursos disponibles, en este caso el tiempo máximo disponible. Para lograrlo, se define una función objetivo que minimiza la variabilidad en la asignación del tiempo de patrullaje respecto a una distribución teóricamente óptima, teniendo en cuenta la probabilidad de ocurrencia de eventos de interés en cada región.

En términos matemáticos, la función objetivo se expresa como una maximización del tiempo adicional de patrullaje asignado a cada zona más allá del tiempo mínimo requerido, ponderado por la importancia estratégica de esa región. Esta estrategia permite asignar más tiempo a las áreas con mayor riesgo o actividad sospechosa, sin generar desequilibrios excesivos en la vigilancia global del área.

Este modelo cuenta también con un enfoque adaptativo, ya que incorpora variables dinámicas como el impacto de las condiciones meteorológicas, la disponibilidad de recursos y la capacidad operativa de las unidades de patrulla. De esta manera el sistema puede ajustarse en función de cambios en el entorno operativo, optimizando su desempeño en tiempo real.

Para garantizar que la solución obtenida sea factible tanto matemáticamente como tácticamente, se han definido las restricciones mostradas en la Tabla 3-1, junto con su descripción y su implementación en MATLAB.

Restricción	Descripción	Implementación en MATLAB
Tiempo máximo disponible (Tmax)	La suma del tiempo de patrullaje en todas las regiones no puede superar el tiempo total disponible	Se define como restricción de desigualdad en fmincon, asegurando que $\Sigma(\text{tiempo asignado}) \leq T_{\max}$
Tiempo mínimo por región	Cada región debe recibir un tiempo mínimo de patrullaje según su extensión	Se define como restricción de desigualdad que impide valores menores al mínimo requerido
Factor meteorológico	El tiempo de patrullaje varía según las condiciones meteorológicas	Se incorpora un factor corrector multiplicador aplicado al tiempo de vigilancia en cada región
Distribución equitativa	Evita una asignación excesivamente desigual entre regiones	Se introduce una penalización en la función objetivo para minimizar diferencias extremas
Verificación de factibilidad	Garantiza que las soluciones cumplan con Tmax y el tiempo mínimo requerido	Se implementa una validación que revisa si el tiempo total asignado es mayor al permitido y solicita al usuario ajustes si es necesario

Tabla 3-1 Restricciones aplicadas al modelo

3.1.3 Método matemático seleccionado

El modelo de optimización implementado en este trabajo se basa en la resolución de un problema de optimización matemática con restricciones, utilizando para ello el solver fmincon de MATLAB. Este método permite encontrar el mínimo de una función objetivo sujeta a restricciones de igualdad e inequaciones, optimizando la asignación de tiempos de patrullaje entre diferentes regiones.

El modelo se basa en la minimización de una función objetivo $f(x)$ que mide la dispersión del tiempo asignado a cada zona respecto a una distribución ideal. Matemáticamente, el problema se expresa como aparece en la ecuación (3-1).

$$\min_x f(x) \quad (3-1)$$

La función objetivo minimiza la dispersión cuadrática de los tiempos asignados a cada región, ponderando los valores según factores de importancia operativa y condiciones meteorológicas.

El modelo planteado se basa en una función objetivo de tipo cuadrático como se muestra en la ecuación (3-2).

$$\min_x f(x) = \sum_{i=1}^n (x_i \cdot (3 - p_i) \cdot F_m)^2 \quad (3-2)$$

Donde:

- x_i es el tiempo extra de patrullaje asignado a cada región i
- p_i es la ponderación operativa de cada región i
- F_m es el factor meteorológico que afecta a cada región

Esta función objetivo es continua y suave C^2 en todo el dominio de definición y está compuesta exclusivamente por términos cuadráticos positivos, por lo que se puede afirmar que es estrictamente convexa siempre que $(3 - p_i) \cdot F_m \neq 0$, condición que se cumple en la práctica para todos los escenarios planteados.

La ponderación operativa de las regiones i se expresa como muestra la ecuación (3-3).

$$p_i = WA \cdot CA \cdot PA_i + WB \cdot CB \cdot PB_i + WC \cdot CC \cdot PC_i \quad (3-3)$$

Donde:

- A, B, C representan sucesos que ocurren en las áreas a patrullar, como por ejemplo narcotráfico, expolio del patrimonio subacuático o vuelos no autorizados.
- WA representa la importancia operativa asignada por la autoridad responsable para dicho suceso.
- CA representa la capacidad de la que dispone la plataforma patrullera para localizar dicho suceso.
- PA representa la probabilidad de que se produzca dicho suceso en ese área

Por otra parte, el conjunto factible definido por las restricciones impuestas es un conjunto convexo y cerrado. En particular:

1. Restricción de tiempo total, como refleja la ecuación (3-4), define un semiespacio convexo.

$$\sum_{i=1}^n x_i + k \cdot \sum_{i=1}^n S_i \leq T_{max} \quad (3-4)$$

Donde:

- x_i es el tiempo extra asignado a la región i .
- S_i es el tamaño de la región i .
- k es un coeficiente de ajuste.
- T_{max} es el tiempo total de patrullaje disponible.

2. La condición de no negatividad, como muestra la ecuación (3-5), define un ortante positivo, también convexo.

$$x_i \geq 0, \quad \forall i \in \{1, \dots, n\} \quad (3-5)$$

3. Incorporación del factor meteorológico, como muestra la ecuación (3-6), donde F_m es un coeficiente corrector en función de las condiciones meteorológicas de cada zona.

$$T_{total}(i) = (k \cdot S_i + x_i) \cdot F_m \quad (3-6)$$

Como la función objetivo es estrictamente convexa y el conjunto factible es convexo, se garantiza la existencia de una única solución óptima global para el problema planteado. Esta propiedad valida formalmente la elección de un algoritmo basado en métodos deterministas como `fmincon`, el cual está diseñado para resolver problemas de optimización no lineal con restricciones, siendo especialmente eficaz en contextos convexos.

En el código, estas restricciones se implementan en `fmincon` como aparece en la Figura 3-1.

```
% -----
% FUNCIÓN PARA OPTIMIZAR EL PATRULLAJE
% -----
function [x, fval, exitflag, output] = optimizar_patrullaje(datos, n)

    Aeq = ones(1, n);
    beq = datos.Tmax - datos.k * sum(datos.S);
    lb = zeros(n, 1);
    ub = [];
    x0 = ones(n, 1) * 20;
    [x, fval, exitflag, output] = fmincon(@(x) fun(n, x, datos), x0, ...
        [], [], Aeq, beq, lb, ub);
```

Figura 3-1 Código definiendo `fmincon`

A_{eq} y beq definen la restricción de igualdad $A_{eq} \times x = beq$ imponiendo que la suma total de los valores de los tiempos extra debe ser igual al tiempo disponible, descontando el tiempo mínimo; mientras que lb y ub imponen la condición de no negatividad en las variables de optimización y no establece un límite superior de tiempos de patrullaje fijo.

Para la definición de la función objetivo, se analizaron dos enfoques basados en normas de optimización:

- Norma ℓ_1 :

$$\|x\|_1 = \sum_{i=1}^n |x_i| \quad (3-7)$$

Definida en la ecuación (3-7), se consideró esta norma por su capacidad para inducir soluciones dispersas, que son aquellas que cuentan con asignaciones de tiempo concentradas en pocas regiones.

Sin embargo, su no diferenciabilidad en $x = 0$ dificulta la convergencia con métodos basados en gradientes, como los empleados por `fmincon`. Pero lo que más afectaba a la resolución del modelo planteado fue que esta norma podría generar soluciones donde algunas regiones no recibieran ningún tiempo de patrullaje, quedando completamente descartadas (`tiempos_totales = 0`).

- Norma ℓ_2 . Originalmente la forma euclidiana ℓ_2 como muestra la ecuación (3-8):

$$\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2} \quad (3-8)$$

Al incluir la raíz cuadrada, se generan valores más equilibrados y menos sensibles a valores extremos. Por el contrario, en este modelo se ha utilizado la norma ℓ_2^2 (la versión cuadrática de la norma ℓ_2), como indica la ecuación (3-9), ya que este formato penaliza los valores grandes debido al término cuadrático, lo que puede llevar a distribuciones más homogéneas.

$$\|x\|_2^2 = \sum_{i=1}^n x_i^2 \quad (3-9)$$

Además, al no incluir la raíz cuadrada, la función es más fácil de optimizar porque su gradiente es más sencillo y bien definido.

El problema de optimización de la vigilancia marítima presenta restricciones tanto lineales como no lineales, lo que requiere un enfoque capaz de manejar este tipo de condiciones. `fmincon` fue seleccionado debido a sus siguientes ventajas:

- Capacidad para manejar restricciones complejas: A diferencia del método simplex, que solo es adecuado para problemas de programación lineal, `fmincon` permite definir restricciones no lineales, como la influencia del factor meteorológico en la asignación del tiempo de patrullaje.

También permite definir restricciones de igualdad e inecuaciones, necesarias para garantizar que la suma del tiempo total de patrullaje no supere el máximo disponible (T_{max}), mientras que cada zona reciba el mínimo requerido.

- Flexibilidad en la función objetivo: Se optó por un modelo basado en la minimización de la dispersión cuadrática del tiempo de patrullaje, lo que permite una distribución más equitativa entre las regiones de vigilancia.

Métodos alternativos como la programación entera o la optimización heurística no ofrecen la misma precisión al trabajar con funciones cuadráticas de penalización.

- Eficiencia computacional en comparación con métodos heurísticos: Algoritmos como recocido simulado o búsqueda por enjambre de partículas pueden ser más efectivos en problemas altamente no lineales, pero suelen requerir mayor tiempo de convergencia y ajustes manuales de parámetros. `fmincon` usa algoritmos eficientes como SQP (Sequential Quadratic Programming) y Interior-Point, que permiten encontrar soluciones óptimas de manera más rápida para problemas de tamaño moderado.

- Soporte en MATLAB y facilidad de implementación: MATLAB es ampliamente utilizado en ingeniería y defensa, lo que facilita la integración del código con otras herramientas analíticas. `fmincon` cuenta con una interfaz bien documentada y opciones avanzadas de configuración que permiten refinar el proceso de optimización sin necesidad de desarrollar algoritmos personalizados desde cero.

3.2 Estructura del Código

El código desarrollado para la optimización del patrullaje marítimo en un escenario de recursos limitados está estructurado en varios bloques funcionales, cada uno con una finalidad específica. Se implementa en MATLAB, utilizando técnicas de optimización matemática con el solver `fmincon` y herramientas de visualización geoespacial. La ejecución del código sigue una secuencia modular que facilita su interpretación, mantenimiento y mejora futura. La estructura del código se divide en:

3.2.1 Interfaz gráfica de bienvenida y solicitud de datos

La ejecución del programa inicia con la función `interfaz_bienvenida()`, que muestra una pantalla inicial interactiva para proporcionar información introductoria y mejorar la experiencia del usuario.

Se crea una ventana emergente con el comando `figure()` mostrando el título “Optimización de la vigilancia marítima”. Como se muestra en la Figura 3-2, la ventana muestra una imagen con el comando `imshow()`, llamando al archivo `imagen.jpg` que se encuentra en el mismo directorio con el código. Además, se agregan etiquetas de texto donde se incluyen el nombre del autor y el del CUD Uvigo de la Escuela Naval Militar.

Para una experiencia más interactiva, se agrega un botón de inicio con el comando `uicontrol('pushbutton')`, que al ser pulsado cierra esta interfaz de bienvenida y ejecuta la función principal del código `optimización_patrullaje()`. Al pulsar el botón “Comenzar”, se ejecuta: `close(fig)` y `optimización_patrullaje`;



Figura 3-2 Interfaz de bienvenida

Tras cerrar la interfaz de bienvenida, el usuario ingresa los parámetros a través de `interfaz()`, mostrada en la Figura 3-3, utilizando la función `inputdlg()` para obtener los datos necesarios para el problema:

- Probabilidades de detección (PA, PB, PC)
- Peso asignado a cada suceso (WA, WB, WC)
- Capacidad de nuestro patrullero para localizar cada suceso (CA, CB, CC)
- Tamaño de las zonas (S)
- Tiempo total disponible (Tmax)
- Factor meteorológico (factor_meteorologico)
- Parámetro de ajuste (k)

Una vez el usuario introduce los datos, se verifica que el tiempo mínimo requerido $k * \text{sum}(S)$ no supere T_{max} . Si la condición no se cumple, se muestra una interfaz con un mensaje de error `errordlg()` y se reinicia la solicitud de datos.

The screenshot shows a window titled "Ingresa Datos" with the following fields and values:

- Probabilidad PA (ej: 0.25,0.5,0.3,0.4,0.2):** 0.25,0.5,0.3,0.4,0.2
- Probabilidad PB (ej: 0.5,0.5,0.3,0.4,0.3):** 0.5,0.5,0.3,0.4,0.3
- Probabilidad PC (ej: 0.75,0.6,0.6,0.5,0.2):** 0.75,0.6,0.6,0.5,0.2
- Peso WA:** 1
- Peso WB:** 1.5
- Peso WC:** 0.5
- Capacidad CA:** 1.5
- Capacidad CB:** 1
- Capacidad CC:** 2
- Tamaño de zonas S (ej: 100,100,200,150,180):** 100,100,200,150,180
- Nombres de zonas (ej: A,B,C,D,E):** Z1,Z2,Z3,Z4,Z5
- Factor Meteorológico (ej: 1.2,0.8,1.0,1.1,0.9):** 1.2,0.8,1.0,1.1,0.9
- Constante k:** 0.09
- Tiempo total Tmax:** 100

At the bottom right are "OK" and "Cancel" buttons.

Figura 3-3 Interfaz de solicitud de datos al usuario

El código incorpora también datos por defecto que aparecen en los prompt de solicitud de los datos para mostrar al usuario el formato en el que deben ser introducidos los datos; o en caso de que no se introduzca algún dato, el código corra de manera completa.

Junto con la interfaz de solicitud de datos al usuarios, se añade una interfaz en el que se puede consultar qué factor meteorológico corresponde con las condiciones meteorológicas previstas en el área a patrullar. Esto supone una gran ayuda al usuario, que puede elegir qué factor se corresponde más con las condiciones reales, lo que supone mayor precisión y rigurosidad en el cálculo del tiempo a patrullar cada zona.

3.2.2 Validación y preprocesamiento de datos

El código implementa validaciones como se muestra en la Figura 3-4 para asegurar que los datos ingresados por parte del usuario sean coherentes antes de comenzar la optimización, donde se verifica que el tiempo total disponible T_{max} sea suficiente para cubrir el tiempo mínimo requerido.

```
% Validación del tiempo mínimo requerido
tiempo_minimo_total = datos.k * sum(datos.S);
if tiempo_minimo_total > datos.Tmax
    h = errordlg(['El tiempo máximo ingresado es insuficiente.' ...
        ' Ajuste el valor de Tmax.'], 'Error', 'modal');
    uiwait(h); % Espera a que el usuario cierre el mensaje
    continue; % Vuelve a pedir los datos
end
```

Figura 3-4 Código validación tiempo máximo

Esta validación previene errores en la ejecución al asegurar que los valores ingresados son viables antes de proceder con la optimización.

Asimismo, el código convierte los datos ingresados por el usuario desde cadenas de texto a valores numéricos como se muestra en la Figura 3-5 para garantizar que puedan ser utilizados en cálculos matemáticos.

```
datos.PA = str2num(respuesta{1});
datos.PB = str2num(respuesta{2});
datos.PC = str2num(respuesta{3});
datos.WA = str2double(respuesta{4});
datos.WB = str2double(respuesta{5});
datos.WC = str2double(respuesta{6});
datos.CA = str2double(respuesta{7});
datos.CB = str2double(respuesta{8});
datos.CC = str2double(respuesta{9});
datos.S = str2num(respuesta{10});
datos.nombres_S = strsplit(strtrim(respuesta{11}), ',');
datos.factor_meteorologico = str2num(respuesta{12});
datos.k = str2double(respuesta{13});
datos.Tmax = str2double(respuesta{14});
```

Figura 3-5 Código definiendo los datos de entrada

Estas conversiones permiten asegurar que los valores ingresados sean correctamente interpretados y utilizados dentro del código sin generar errores inesperados. Con los parámetros ingresados en cada prompt, el código los introduce a MATLAB como una constante y los opera como vector fila o celda, como se muestra en la Tabla 3-2, donde se expone el formato de cada valor ingresado al modelo por parte del usuario.

Cabe destacar que las matrices que se generan al introducir los datos en la interfaz son $(1 \times n)$, pero MATLAB opera mejor con matrices del tipo $(n \times 1)$, por lo que hay que generar la matriz transpuesta de estos datos al operar. En el software de MATLAB hay varias formas de generar la transpuesta y en este trabajo se ha optado por utilizar el método `A(:)` ya que genera un reordenamiento de los datos a columnas.

Prompt	MATLAB	Formato
Probabilidad sucesos (PA, PB, PC)	datos.PA	Vector fila $(1 \times n)$
Peso (WA, WB, WC)	datos.WA	Escalar
Capacidad (CA, CB, CC)	datos.CA	Escalar
Tamaño de zonas	datos.S	Vector fila $(1 \times n)$
Nombres de zonas	datos.nombres_S	Celda $(1 \times n)$
Factor meteorológico	datos.factor_meteorologico	Vector fila $(1 \times n)$
Constante k	datos.k	Escalar
Tiempo máximo	datos.Tmax	Escalar

Tabla 3-2 Formato de entrada de los datos al código

3.2.3 Implementación del algoritmo de optimización

La implementación del algoritmo de optimización tiene como objetivo la asignación óptima del tiempo de patrullaje en distintas regiones, garantizando un equilibrio entre la eficiencia operativa y las restricciones impuestas. Para ello, se emplea el solver `fmincon` de MATLAB, el cual permite la resolución de problemas de optimización no lineal con restricciones lineales y no lineales.

La correcta implementación del algoritmo de optimización es un aspecto crucial del sistema, ya que de ello depende tanto la obtención de soluciones viables como la eficiencia global del proceso. Para garantizar una convergencia estable hacia soluciones cercanas al óptimo, es fundamental una gestión precisa de las estructuras de datos, un control riguroso de los criterios de parada y una adecuada configuración de los parámetros de entrada.

Además, se debe prestar especial atención a la estabilidad numérica del algoritmo, ya que pequeños errores de redondeo o acumulación pueden comprometer la validez de los resultados. En este sentido, la implementación debe estar orientada a minimizar el coste computacional manteniendo la robustez del sistema, especialmente en escenarios donde las condiciones del entorno o los requisitos operativos imponen restricciones adicionales.

En el código, la optimización sigue la estructura modular y utiliza la función `fmincon`, como se muestra en la Figura 3-6, para minimizar la función objetivo bajo las restricciones establecidas.

```
% -----
% FUNCIÓN PARA OPTIMIZAR EL PATRULLAJE
% -----
function [x, fval, exitflag, output] = optimizar_patrullaje(datos, n)

    Aeq = ones(1, n);
    beq = datos.Tmax - datos.k * sum(datos.S);
    lb = zeros(n, 1);
    ub = [];
    x0 = ones(n, 1) * 20;
    [x, fval, exitflag, output] = fmincon(@(x) fun(n, x, datos), x0, ...
        [], [], Aeq, beq, lb, ub);
```

Figura 3-6 Código definiendo fmincon

Donde:

- Se establece un punto inicial de búsqueda con $x0 = \text{ones}(n, 1) \times 20$. Este es el punto de inicio desde donde fmincon comienza la optimización. Se establece que cada región tenga 20 unidades de tiempo extra.
- Se define la restricción de igualdad $Aeq \times x = beq$, que matemáticamente se expresa como muestra en la ecuación (3-10):

$$\sum_{i=1}^n x_i = Tmax - k \cdot \sum S_i \quad (3-10)$$

Esto se define con $Aeq = \text{ones}(1, n)$ porque se suman todas las x_i . beq representa en este caso el tiempo máximo disponible tras descontar el tiempo mínimo de patrullaje.

- Se establecen cotas $lb \leq x_i \leq ub$ entre las que se puede encontrar el valor de los tiempos extras. Al definirse $lb = \text{zeros}(n, 1)$ y $ub = []$ se establece una cota mínima en cero y una cota máxima no establecida.

Cuando se ejecuta fmincon inicia la búsqueda en $x0 = [20, 20, \dots, 20]$ y evalúa la función objetivo en $x0$. Se realizan iteraciones de optimización usando un algoritmo basado en gradientes para buscar la mejor solución. Se evalúan pequeñas modificaciones de x y verifica cómo cambia $f(x)$; luego se van ajustando los valores de x respetando la restricción de igualdad. Este proceso termina cuando el cambio en x es muy pequeño o se alcanza el número máximo de iteraciones.

La implementación del algoritmo de optimización en MATLAB permite la asignación eficiente de tiempos de patrullaje bajo restricciones operativas. Utilizando fmincon, se garantiza que los recursos disponibles se distribuyan de manera óptima, teniendo en cuenta condiciones meteorológicas, probabilidades de detección y prioridades estratégicas. Este planteamiento basado en optimización matemática asegura una vigilancia marítima más efectiva y cobertura de las zonas de interés.

3.2.4 Representación gráfica y visualización

Para mostrar los resultados de una forma visual y representativa, el programa genera una representación geográfica del patrullaje optimizado, utilizando herramientas de visualización. Para ello se requieren de comandos utilizados para:

- Definición de coordenadas geográficas: Se especifican las regiones a patrullar con sus respectivas posiciones geográficas formando un área.
- Generación del mapa: Se utiliza `worldmap()` y se definen los límites de latitud y longitud para mostrar el mapa a la escala adecuada. Con `geoshow()` se superponen los datos geoespaciales.
- Coloreado de zonas: `patcm()` permite asignar colores a las áreas en función del tiempo de patrullaje, facilitando la interpretación visual del usuario.
- Incorporación de etiquetas: Mediante `textm()` se añaden los nombres de las regiones representadas y los valores de tiempo total de patrullaje de cada región.

Para complementar la representación geográfica, el programa genera un gráfico de barras a color, como muestra la Figura 3-7, representando la distribución de tiempos de patrullaje por región. Para ejecutar esta interfaz se requieren comandos para:

- Generación del gráfico: Se utiliza `bar()` para representar los tiempos asignados en cada región.
- Asignación de colores: `colormap('parula')` asigna un degradado de color a las barras, facilitando la identificación de valores.
- Líneas de referencia: Se incluyen líneas horizontales (`plot()`) para marcar los tiempos mínimos requeridos en cada área.
- Etiquetado: Se incorporan valores numéricos sobre cada barra, mejorando la interpretación visual de los resultados. Además se representa una línea roja horizontal sobre cada barra que representa el valor del tiempo mínimo de cada zona, para una mejor identificación del tiempo extra de cada región.

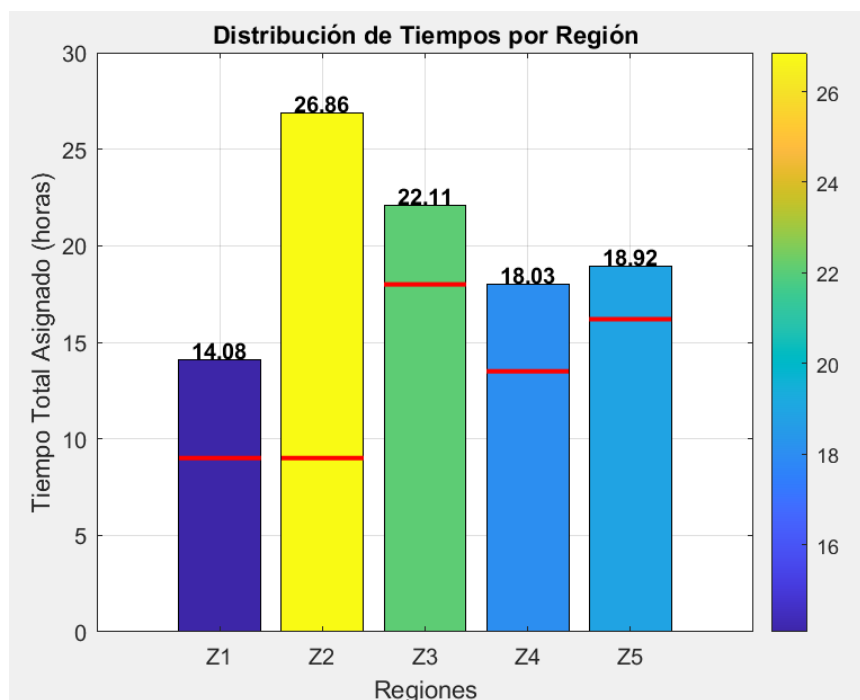


Figura 3-7 Gráfica de barras mostrando resultados

Esta representación facilita la comparación visual de tiempos asignados, permitiendo identificar regiones con mayor necesidad de vigilancia y optimizar la asignación de recursos.

3.2.5 Salida de resultados en consola

El módulo de salida de resultados proporciona una manera clara y estructurada de presentar la información relevante derivada del análisis de optimización. Los resultados se imprimen en la propia consola de MATLAB, como refleja la Figura 3-8, para una rápida y sencilla interpretación. Con `fprintf()` se imprimen:

- Tiempo mínimo requerido para cada zona.
- Tiempo extra asignado tras la optimización.
- Tiempo total final dedicado a cada región.
- Valor de la función objetivo `fval`, indicando la eficiencia de la optimización.
- Indicadores de convergencia `exitflag`, que verifican si el algoritmo alcanzó una solución óptima. El indicador `exitflag()` puede tomar los valores:
 - 1: Convergencia exitosa, se encontró una solución óptima.
 - 0: Se alcanzó el número máximo de iteraciones sin encontrar una solución óptima.
 - -1: Terminación prematura debido a problemas en la configuración o errores numéricos.
 - -2: No se encontró un punto factible que satisfaga las restricciones.
 - -3: El problema no está acotado y no tiene solución óptima finita

```

-----
El valor de la función objetivo (fval) es: 317.5459

El valor de exitflag es: 1

-----
El vector de tiempos EXTRA en las regiones es:
Región: Z1 - Tiempo Extra: 5.08
Región: Z2 - Tiempo Extra: 17.86
Región: Z3 - Tiempo Extra: 4.11
Región: Z4 - Tiempo Extra: 4.53
Región: Z5 - Tiempo Extra: 2.72
-----

-----
Tiempos MÍNIMOS asignados por región:
Región: Z1 - Tiempo Mínimo: 9.00
Región: Z2 - Tiempo Mínimo: 9.00
Región: Z3 - Tiempo Mínimo: 18.00
Región: Z4 - Tiempo Mínimo: 13.50
Región: Z5 - Tiempo Mínimo: 16.20
-----

-----
Tiempo TOTAL asignado por región:
Región: Z1 - Tiempo Total: 14.08
Región: Z2 - Tiempo Total: 26.86
Región: Z3 - Tiempo Total: 22.11
Región: Z4 - Tiempo Total: 18.03
Región: Z5 - Tiempo Total: 18.92
-----
    
```

Figura 3-8 Salida de datos en consola mostrando resultados

3.3 Configuración del entorno y requisitos

Para garantizar la correcta ejecución del código y la optimización eficiente de los recursos computacionales, es fundamental configurar adecuadamente el entorno de desarrollo en MATLAB, abarcando desde la instalación del software hasta la organización de archivos y la gestión de librerías necesarias para la ejecución del modelo.

El código ha sido desarrollado en MATLAB debido a su capacidad para manejar cálculos numéricos avanzados, algoritmos de optimización y visualización de datos de manera eficiente. Se recomienda utilizar una versión reciente del software, preferiblemente MATLAB R2020a o posterior, para garantizar la compatibilidad con las funciones empleadas en la optimización del patrullaje marítimo y la representación gráfica de los resultados.

Además de contar con una versión actualizada de MATLAB, es imprescindible disponer de los siguientes Toolboxes, los cuales proporcionan funcionalidades clave para el desarrollo y ejecución del modelo:

- Optimization Toolbox: Incluye el solver fmincon, utilizado para resolver el problema de optimización del tiempo de patrullaje bajo restricciones.
- Mapping Toolbox: Esencial para la representación geoespacial de los datos, facilitando la visualización de las áreas de patrullaje mediante funciones como worldmap() y geoshow().
- Statistics and Machine Learning Toolbox: Útil para la realización de análisis estadísticos avanzados, con potencial aplicación en futuras mejoras del modelo, como la integración de técnicas de aprendizaje automático para la optimización del patrullaje.

La correcta instalación y configuración de estos elementos garantiza la estabilidad del entorno de trabajo y permite la ejecución eficiente de los algoritmos de optimización, asegurando la fiabilidad de los resultados obtenidos.

3.4 Modelado del área de patrullaje

El modelado del área de patrullaje es un aspecto clave en la optimización de la vigilancia marítima. En el caso de este trabajo fin de grado, se ha seleccionado el Estrecho de Gibraltar, basando la elección en el histórico de datos de actividades delictivas en la zona, incluyendo narcotráfico y tráfico de personas. Esta información permite definir las regiones más críticas y optimizar los recursos disponibles para maximizar la eficiencia del patrullaje. La elección de estas áreas no es aleatoria ciertamente, sino que responde a patrones de actividad delictiva registrados en la zona, lo que permite mejorar la efectividad de las operaciones de vigilancia y control.

3.4.1 Definición de las regiones y representación geográfica

En el código se definen las regiones de patrullaje, como puede verse en la Figura 3-9, mediante la asignación de coordenadas geográficas específicas a cada zona de vigilancia. Cada región se define con cuatro puntos geográficos que forman un área a patrullar.

```
% -----  
% COORDENADAS FIJAS DE LAS REGIONES  
% -----  
regiones = {  
    [36.621, 36.309, 36.043, 36.056; -6.706, -6.309, -6.301, -6.754];  
    [36.264, 36.098, 35.94, 35.996; -6.241, -5.886, -5.883, -6.255];  
    [36.089, 36.004, 35.96, 35.933; -5.862, -5.562, -5.542, -5.843];  
    [36.011, 36.248, 36.047, 35.972; -5.51, -5.124, -4.964, -5.502];  
    [36.333, 36.531, 36.204, 36.049; -5.137, -4.523, -4.312, -4.918]  
};
```

Figura 3-9 Código definiendo coordenadas que definen áreas

Cada una de estas zonas representa un área de vigilancia que es priorizada en función de la probabilidad de actividades ilícitas en la región. Para la representación geográfica de las zonas de patrullaje, el código utiliza worldmap() y geoshow(), como se aprecia en la Figura 3-10; herramientas de MATLAB que permiten la visualización de datos sobre un mapa global. Estas funciones son empleadas para representar las áreas a patrullar con el dato de las horas a dedicar, facilitando así su análisis visual.

```
for i = 1:n
    latitudes = regiones{i}(1, :);
    longitudes = regiones{i}(2, :);
    patchm(latitudes, longitudes, 'FaceColor', cmap(colores(i), :), ...
        'FaceAlpha', 0.7, 'EdgeColor', 'k', 'LineWidth', 1);
```

Figura 3-10 Código de la representación geográfica

A continuación, se trazan las áreas a patrullar mediante `patchm()`, como se muestra en la Figura 3-11, permitiendo visualizar las zonas de vigilancia en el mapa real.

```
figure;
hold on;
worldmap([35.3 36.8], [-6.7 -4.2]);
load coastlines;
geoshow(coastlat, coastlon, 'DisplayType', 'line', 'Color', 'k');
```

Figura 3-11 Código de áreas en el mapa

Se asignan colores a las áreas de patrullaje en función del tiempo total asignado a cada región. Cada área se representa con una tonalidad diferente en base a un gradiente de color, generado mediante la función `colormap('turbo')`. Esto permite que las zonas con mayor tiempo de patrullaje se distingan visualmente de aquellas con menor tiempo asignado.

Para facilitar la interpretación de los datos en el mapa, se colocan etiquetas de texto sobre cada región como se muestra en la Figura 3-12. Estas etiquetas muestran el nombre de la zona y el tiempo total asignado, lo que permite una interpretación de la distribución de recursos.

La correcta representación visual de las regiones de patrulla permite identificar rápidamente las zonas con mayor presencia de vigilancia. Esta herramienta facilita la toma de decisiones estratégicas, ya que ayuda a evaluar si la distribución de los tiempos de patrullaje se alinea con las necesidades reales de seguridad en la zona. Además, la inclusión de etiquetas informativas mejora la legibilidad del mapa, permitiendo a los operadores comprender la distribución de recursos sin necesidad de realizar cálculos adicionales.

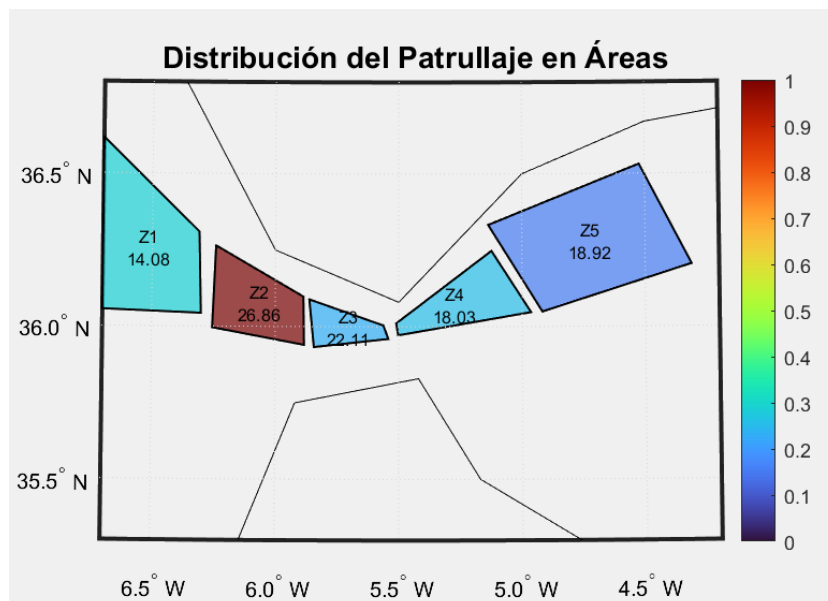


Figura 3-12 Interfaz mostrando áreas a patrullar con tiempo

3.5 Evaluación del algoritmo y posibles mejoras

La evaluación del algoritmo de optimización desarrollado para la distribución del tiempo de patrullaje en vigilancia marítima permite determinar su eficiencia, robustez y viabilidad operativa. Este análisis se basa en la comparación entre los tiempos mínimos requeridos y los tiempos asignados, así como en la verificación del cumplimiento de restricciones impuestas en el modelo.

El código implementa la función de optimización `fmincon` de MATLAB, la cual resuelve problemas de optimización no lineal con restricciones. Se analiza el comportamiento del algoritmo en distintos escenarios, evaluando su sensibilidad a variaciones en los parámetros de entrada, como el factor meteorológico y las probabilidades de detección de eventos. Además, se examina el modelo frente a cambios bruscos en las condiciones operativas y la convergencia del algoritmo en casos de alta complejidad computacional.

A partir de las simulaciones realizadas, se han identificado fortalezas y áreas de mejora en el modelo, lo que permite planear futuras optimizaciones para incrementar rendimientos y aplicabilidad operativa. Se ha observado que el algoritmo responde adecuadamente en escenarios planteados con condiciones meteorológicas estables, pero su desempeño puede degradarse cuando los factores ambientales presentan una alta variabilidad.

3.5.1 Análisis de la eficiencia del modelo

Para medir la eficacia del modelo, se han utilizado los siguientes indicadores:

- **Robustez en la asignación de tiempos:** El algoritmo es capaz de distribuir los tiempos de patrullaje de manera coherente, aunque en casos de alta variabilidad en la criticidad de las zonas, algunas áreas pueden recibir asignaciones que rozan su límite mínimo, lo que podría afectar la cobertura óptima en escenarios dinámicos.
- **Escalabilidad del modelo:** Se ha probado el rendimiento del algoritmo en configuraciones con un mayor número de regiones, observando que, aunque `fmincon` logra resolver el problema, la velocidad de convergencia se ve afectada, lo que sugiere la necesidad de métodos de optimización más eficientes para instancias de gran escala.
- **Capacidades de ajuste a eventos dinámicos:** Aunque el modelo integra el factor meteorológico, la actualización en tiempo real de este parámetro podría mejorar la precisión en la asignación de recursos. Implementaciones futuras podrían considerar la incorporación de un sistema de predicción basado en datos históricos y modelado de series temporales.
- **Convergencia del algoritmo:** Se analiza la cantidad de iteraciones requeridas por `fmincon` para alcanzar una solución factible. En escenarios con restricciones más estrictas, el tiempo de convergencia puede aumentar. Se añade el comando `disp` en el código para mostrar el número de iteraciones que el algoritmo ha realizado para la optimización.

3.5.2 Posibles mejoras y optimización futura

Para incrementar la eficiencia y adaptabilidad del modelo, se proponen las siguientes mejoras:

- **Optimización de la función objetivo:** Se puede modificar la función objetivo para introducir penalizaciones adicionales en los casos en los que se observe una distribución de tiempos excesivamente desigual. Además, se puede evaluar la implementación de un modelo multiobjetivo que permita optimizar simultáneamente la cobertura y la eficiencia de patrullaje.
- **Integración de heurísticas avanzadas:** La implementación de algoritmos como búsqueda por enjambre de partículas o recocido simulado podría mejorar la velocidad de convergencia en escenarios con restricciones severas. Estos enfoques pueden complementar la optimización actual y proporcionar soluciones más eficientes en menor tiempo.

- **Paralelización y mejora en la ejecución:** La paralelización del código en MATLAB puede reducir significativamente los tiempos de ejecución al aprovechar múltiples núcleos de procesamiento. La paralelización es una técnica que permite dividir una tarea computacional en partes más pequeñas que pueden ejecutarse simultáneamente en diferentes unidades de procesamiento, lo que acelera el tiempo total de cálculo como se muestra en la Figura 3-13. En MATLAB, esto se puede lograr mediante herramientas como `parfor` en Parallel Computing Toolbox, permitiendo realizar múltiples evaluaciones de funciones en paralelo en lugar de secuencialmente. Esto resultaría beneficioso especialmente en simulaciones a gran escala donde el costo computacional es elevado y se requiere una optimización eficiente del tiempo de ejecución.

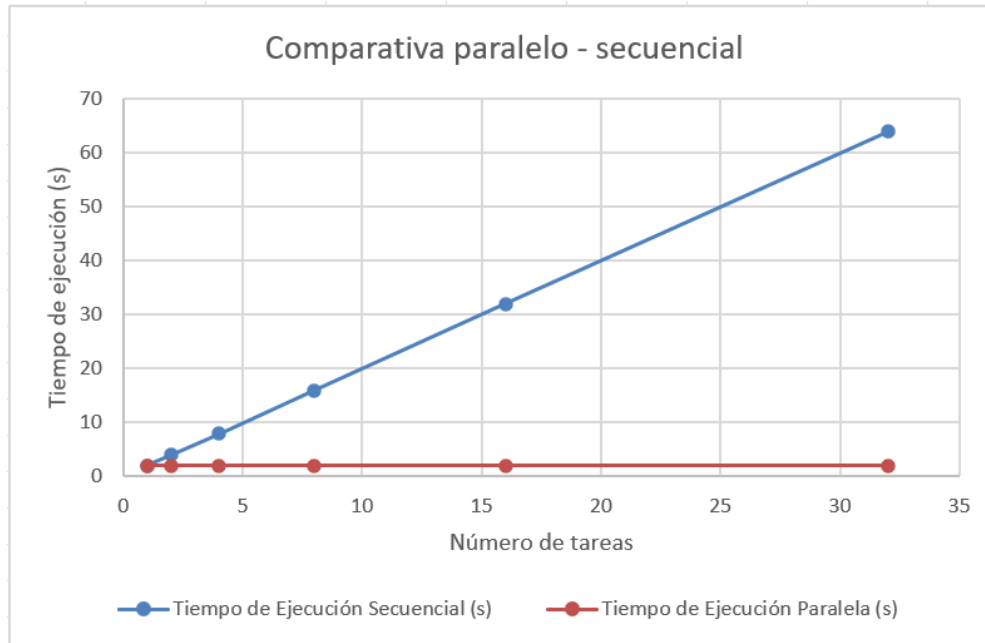


Figura 3-13 Comparativa de ejecución secuencial y paralela

- **Uso de modelos predictivos:** Incorporar técnicas de aprendizaje automático permitiría predecir la demanda de patrullaje en función de datos históricos, ajustando dinámicamente la distribución de tiempos en respuesta a patrones detectados en la actividad marítima.
- **Evaluación de robustez con simulaciones avanzadas:** El modelo debería ser sometido a pruebas en entornos más complejos, que reflejan variaciones significativas en las condiciones operativas, incorporando datos históricos y simulaciones detalladas que representen fluctuaciones en la actividad delictiva y cambios climáticos. Este método permitiría analizar cómo el modelo responde ante eventos inesperados y evaluar su eficacia en escenarios de vigilancia marítima realistas.

3.5.3 Interpretación operativa

La interpretación operativa de los resultados obtenidos permite evaluar la aplicabilidad del modelo en escenarios reales de vigilancia marítima. Los aspectos más relevantes son:

- **Eficiencia en la asignación de recursos:** Se analiza si el modelo distribuye los tiempos de manera equitativa según la criticidad de cada región.
- **Impacto en la cobertura del patrullaje:** Se estudia si la optimización logra una cobertura adecuada de las zonas de mayor interés.

- Impacto en la eficiencia operativa: Se ha observado que la implementación actual permite reducir la dispersión de tiempos asignados sin comprometer la equidad en la vigilancia de las zonas. Sin embargo, la falta de una penalización explícita en la función objetivo para evitar asignaciones excesivamente desequilibradas puede afectar la efectividad del patrullaje en algunos casos.
- Respuesta a cambios en las condiciones operacionales: Se revisa la capacidad del algoritmo para adaptarse a escenarios con recursos más limitados o condiciones meteorológicas adversas.
- Aplicabilidad en la planificación operacional: Se evalúa si el modelo es viable para integrarse en los procesos de decisión de las autoridades marítimas.

4 RESULTADOS OBTENIDOS

4.1 Evaluación general del modelo

4.1.1 Objetivo y metodología del análisis

El objetivo principal de este apartado es evaluar el rendimiento del modelo de optimización de patrullaje marítimo, verificando su eficacia, eficiencia computacional y adaptabilidad ante distintos escenarios.

Para ello, se analizará la distribución del tiempo de patrullaje asignado a cada región y su cumplimiento con los criterios de optimización establecidos. También se medirá la eficiencia computacional del algoritmo mediante el tiempo de ejecución y la cantidad de iteraciones necesarias para alcanzar una solución óptima, como se muestra en la Figura 4-1, lo que permitirá determinar su escalabilidad en escenarios más complejos.

```
Tiempo de ejecución del algoritmo: 1.1653 segundos  
Número de iteraciones ejecutadas: 10
```

Figura 4-1 Resultados mostrados en consola

Se verificará que el modelo cumpla con todas las restricciones impuestas, asegurando que el tiempo total de patrullaje no supere el máximo permitido y que cada región reciba al menos su tiempo mínimo requerido. Además, se evaluará la sensibilidad del algoritmo ante variaciones en los parámetros de entrada, como la criticidad de las zonas, con el fin de comprobar su capacidad de ajuste.

Finalmente se analizará la robustez del modelo verificando la estabilidad de sus soluciones y su capacidad para converger de manera eficiente. Los resultados se visualizarán mediante mapas geoespaciales, gráficos de barras y análisis numéricos que permitirán comparar el desempeño del modelo en distintos escenarios.

4.1.2 Definición de métricas de evaluación

Para evaluar los aspectos anteriormente comentados, se realizarán múltiples simulaciones variando los parámetros de entrada y registrando diferentes métricas de rendimiento del modelo. Se implementará un conjunto de ensayos con configuraciones predefinidas que permitan analizar el comportamiento del modelo en distintos escenarios.

En cada simulación, se recogerán los parámetros de salida recogidos en la Tabla 4-1. Asimismo, estos datos, junto con otros relacionados con el problema y el rendimiento, se almacenan en un Excel en una carpeta generada en el instante de la ejecución del código.

Parámetro	Representa
Tiempo mínimo (Tmin)	Proporcional a área de cada región
Tiempo extra (x)	Tiempo adicional optimizado por la función objetivo
Tiempo total (Ttotal)	Suma de tiempo mínimo y extra de cada región

Tabla 4-1 Datos de salida del modelo

Asimismo, el código incorpora parámetros para la métrica del rendimiento del modelo, como se muestran en la Tabla 4-2, indicando el comportamiento del mismo en cada ensayo.

Parámetro	Representa
Tiempo de ejecución del algoritmo	Medido con tic y toc
Número de iteraciones requeridas	Indica cuántas veces se ha ajustado la solución hasta encontrar la óptima
Convergencia del algoritmo (exitflag)	Indica cómo ha finalizado la optimización
Valor de función objetivo (fval)	Resultado final del cálculo de optimización
Valor de función ponderada (pp)	Prioridad de cada zona en función de riesgo de amenaza y capacidad operativa

Tabla 4-2 Parámetros de métrica del rendimiento

Con la recopilación tanto de los resultados de salida como los parámetros de rendimiento, seremos capaces de evaluar aspectos como:

- **Eficacia operativa:** Determinando si la distribución del tiempo de patrullaje permite optimizar la cobertura de vigilancia y cumplir con los criterios de asignación establecidos. Se analizará si las zonas con mayor criticidad reciben un tiempo de patrullaje en relación con su relevancia operativa y si, realmente, la optimización mejora la eficiencia respecto a una asignación uniforme o basada en heurísticas simples.
- **Eficiencia computacional:** Será medida mediante el tiempo de ejecución del algoritmo y el número de iteraciones necesarias para alcanzar una solución óptima. Se busca garantizar que el modelo proporcione resultados en un tiempo razonable, permitiendo su implementación en escenarios operativos sin generar una carga computacional relevante.
- **Escalabilidad del modelo:** Se analizará si gestionando un mayor número de regiones no se compromete la calidad de la solución ni se aumenta el tiempo de cómputo. Se realizarán pruebas con diferente número de zonas para determinar en qué punto el rendimiento del algoritmo se ve comprometido.
- **Cumplimiento de restricciones:** Se verificará que el modelo respeta los límites de tiempo total disponible y que cada región reciba al menos el tiempo mínimo requerido para una vigilancia efectiva. Se comprobará que la solución optimizada cumpla con las condiciones impuestas sin generar asignaciones inviables.

- Sensibilidad a cambios de parámetros: Analizando la respuesta del modelo ante variaciones de criticidad en las zonas se estudiará la flexibilidad y capacidad de adaptación a diferentes escenarios.
- Robustez y estabilidad del algoritmo: Si el modelo converge de manera eficiente a soluciones óptimas y si estas se mantienen estables ante pequeños cambios en las condiciones iniciales determinará si el modelo es robusto y estable. Se analizará si variaciones nimias generan diferencias significativas en la asignación de tiempos de patrullaje, lo que podría indicar una dependencia excesiva de algún factor.



Figura 4-2 Gráfico radar con diferentes aspectos de un modelo

Como se observa en la Figura 4-2, el objetivo del trabajo fin de grado es desarrollar un modelo de optimización que mantenga un rendimiento equilibrado en todos los aspectos clave de evaluación sin que ninguna métrica decaiga significativamente.

4.2 Análisis 1: Comprobación del modelo

El primer análisis al que se va a someter el código tiene como propósito verificar su correcto funcionamiento y evaluar su capacidad para asignar de manera eficiente los recursos de vigilancia en un escenario de recursos limitados. La fase inicial de las pruebas se ha desarrollado en diferentes etapas, cada una con un propósito diferente pero necesario.

4.2.1 Verificación de la correcta asignación de tiempos con pesos uniformes

En este primer ensayo todos los factores de ponderación se establecen con el mismo peso, excepto en una única área en concreto. El objetivo es comprobar si el modelo detecta correctamente la variación en esa área específica y ajusta la distribución del tiempo en consecuencia.

Este primer ensayo es determinante para validar que el modelo responde de manera coherente a cambios en los parámetros de entrada y que no presenta deficiencias o errores en la asignación de recursos.

Para realizar este primer ensayo se han asignado los mismos valores de entrada a todos los parámetros de entrada salvo la probabilidad de un suceso en cada área, como se refleja en la Tabla 1, Tabla 2, Tabla 3, Tabla 4 y Tabla 5 del Anexo IV: Datos utilizados para la simulación

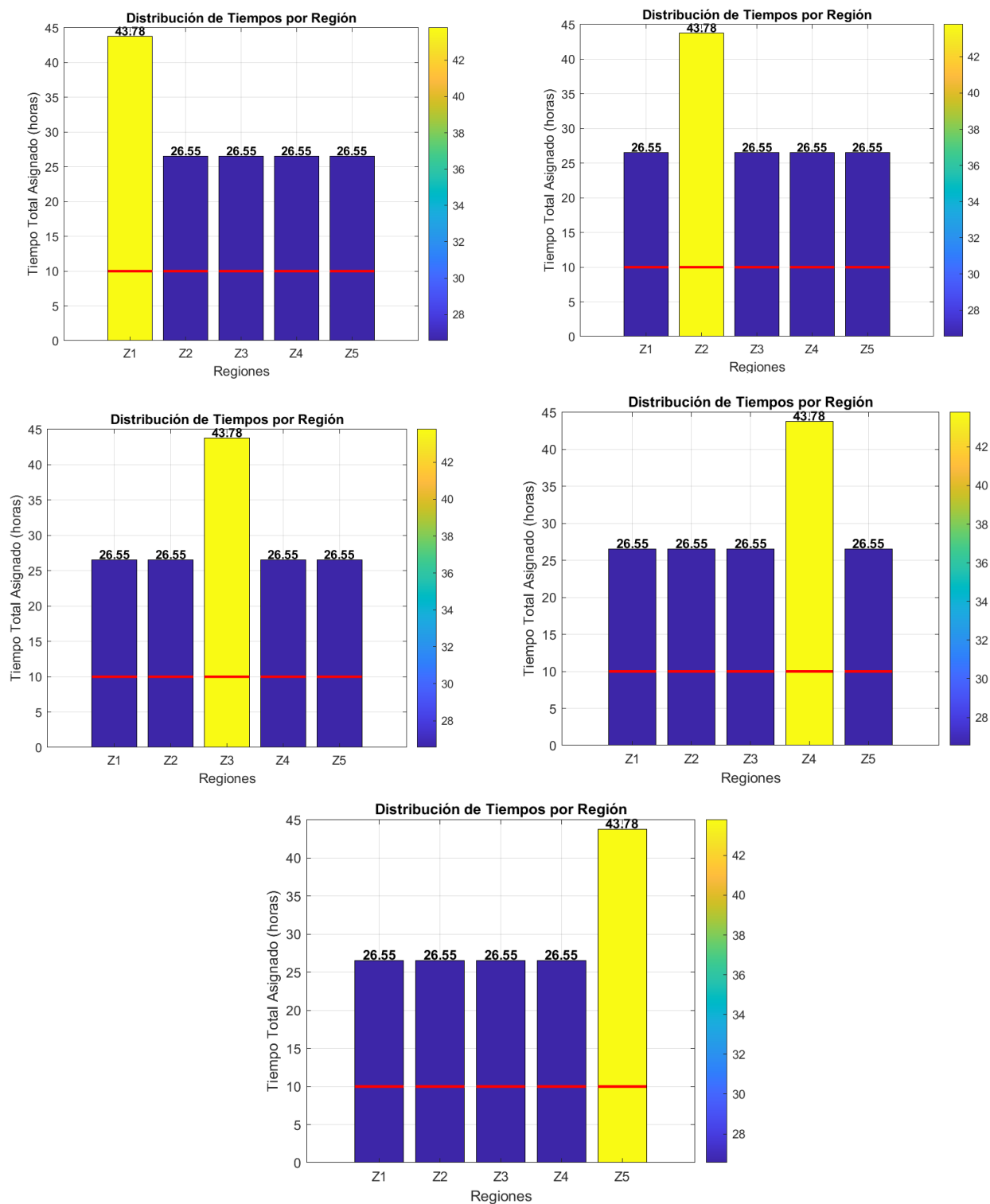


Figura 4-3 Asignación de tiempos por zonas

Como se puede analizar en la Figura 4-3, los resultados obtenidos permiten validar el correcto funcionamiento del código en función de distintos factores de ponderación. Se puede observar una adecuada distribución de los recursos, ya que al modificar el peso de un área específica, el modelo responde asignando un tiempo significativamente mayor. Esto indica que el algoritmo logra detectar correctamente las prioridades de los parámetros de entrada. También se puede apreciar que el modelo está equilibrado ya que las zonas no priorizadas reciben una asignación estable de 26.55 horas en cada ejecución.

En este análisis el tiempo medio de ejecución del algoritmo fue de 0.117 segundos y el número de iteraciones necesarias para alcanzar una solución óptima fue de 4. En la Figura 4-4 se aprecian los tiempos de ejecución de las 5 ejecuciones, cada vez destacando una zona.



Figura 4-4 Tiempos ejecución algoritmo

4.2.2 Evaluación del modelo destacando factores clave como WA o CA

A continuación, se realizan pruebas donde se destacan distintos factores de ponderación, como la importancia asignada por la autoridad responsable o la capacidad del patrullero para localizar determinadas acciones.

En el primer ensayo de este escenario, se incrementa el valor de WA manteniendo el resto de parámetros en 1, como se muestra en la Tabla 6, lo que debería resultar en una mayor asignación de tiempo de patrullaje a aquellas regiones donde la autoridad asigna mayor prioridad.

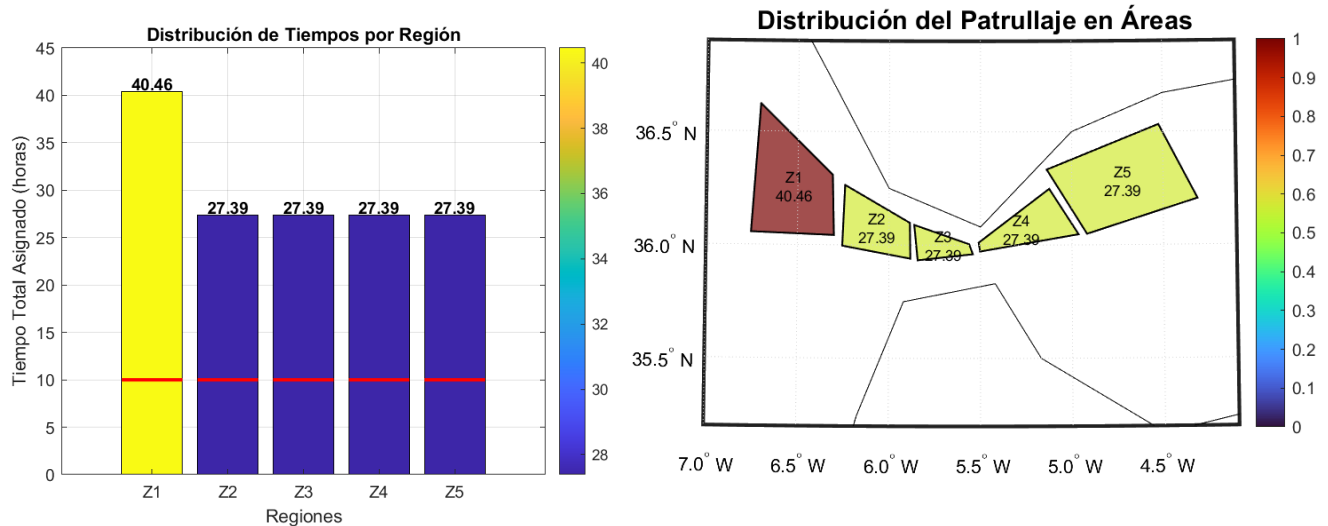


Figura 4-5 Análisis destacando WA

En la Figura 4-5, se observa que la Z1 es la región con mayor asignación de tiempo (40.46 horas), siendo un resultado significativamente superior al resto de áreas (27.39 horas). Esta diferencia notable confirma que al aumentar de peso el factor de la importancia estratégica WA, el algoritmo responde adecuadamente, favoreciendo a las regiones con mayor prioridad según la autoridad responsable.

En el segundo ensayo, se asignan diferentes valores a las capacidades para analizar si el modelo redistribuye los recursos de manera que las áreas donde el patrullero tiene mayor capacidad de acción reciban mayor atención. A todas las áreas se les asigna una probabilidad de sucesos 0.5, como se muestra en la Tabla 7, con las siguientes excepciones:

- Área 1: PA = 1 y CA = 0.4
- Área 2: PB = 1 y CB = 1
- Área 3: PC = 1 y CC = 1.4

De esta manera, se asigna un suceso a cada zona y diferentes capacidades del patrullero para analizar cómo el código asigna los tiempos de vigilancia, obteniendo los resultados mostrados en la Figura 4-6.

Como muestra la Figura 4-6, se observa que a la Z1 se le asigna una cantidad de tiempo menor en comparación con Z2 y Z3, y que a la Z3 se le asigna la mayor cantidad de tiempo. Esto indica que el algoritmo asigna más tiempo de vigilancia a aquellas zonas con una capacidad menor.

Esto se debe a que en la ecuación de la ponderación, el término de la capacidad se encuentra multiplicando al resto de términos, como se muestra en la Ecuación (3-3), resultando en mayor tiempo.

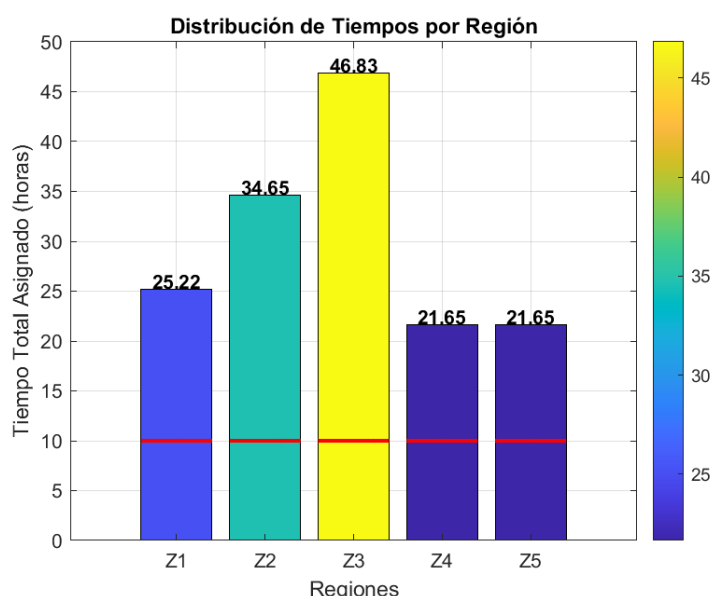


Figura 4-6 Asignación de tiempos diferentes capacidades

Una capacidad mayor que 1, por ejemplo CA = 1.4, indica que el barco es poco capaz de localizar el suceso A, ya que este factor conseguirá un mayor tiempo de vigilancia. Como muestra la Figura 4-7, para lograr una mayor comprensión del usuario con el funcionamiento del código, se añade una interfaz de ayuda al introducir los datos que explica este factor.

En consecuencia, se concluye que el sistema de optimización implementado es sensible a los factores de ponderación definidos, lo que permite una distribución adaptativa del tiempo de patrullaje en función de los criterios establecidos por los responsables.

Esta capacidad es fundamental para diseñar estrategias de patrullaje más eficaces y alineadas con las necesidades específicas de cada región.

Cabe mencionar que la capacidad, al igual que el peso asignado por la autoridad, suponen un valor arbitrario, siendo la diferencia entre distintos valores lo que realmente genera desigualdades a la hora de asignar tiempos. Para partir de una referencia común, se puede pensar que el valor 1 supone un valor neutro para el código, e introducir valores oscilando en un $[-0,5, +0,5]$ para definir capacidades, pesos y probabilidades.

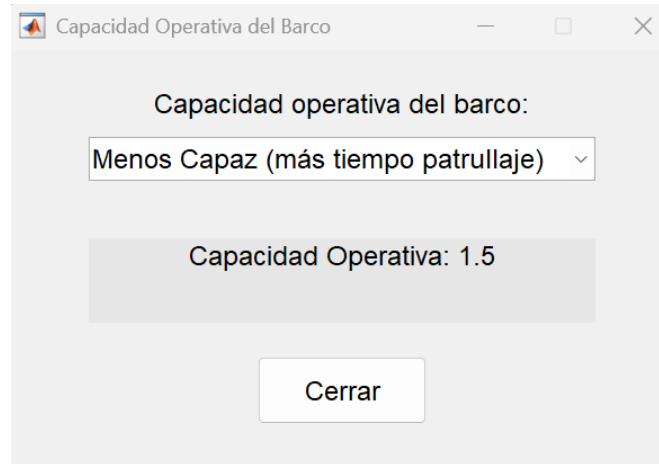


Figura 4-7 Interfaz explicación capacidad

4.2.3 Impacto del factor meteorológico

En el tercer ensayo del primer análisis, se analiza cómo influye el factor meteorológico en los resultados. En este ensayo se igualan todos los parámetros de entrada a 1, eliminando cualquier diferencia entre áreas, y quedando solamente el factor meteorológico distinto en cada área, como se refleja en la Tabla 8. El reparto de factores meteorológicos se asignó de la siguiente manera:

- Zona 1: 0.85
- Zona 2: 0.95
- Zona 3: 1
- Zona 4: 1
- Zona 5: 1.15

El objetivo de este ensayo es someter al código a una situación en la que sólo puede basarse en el factor meteorológico para asignar una prioridad en los tiempos de patrulla; al ser este factor el único que supone diferencias de valor entre áreas.

Con estos parámetros de entrada, el resultado obtenido es el que se muestra en la Figura 4-8, donde se visualiza un reparto diferente de tiempos de vigilancia por cada área.

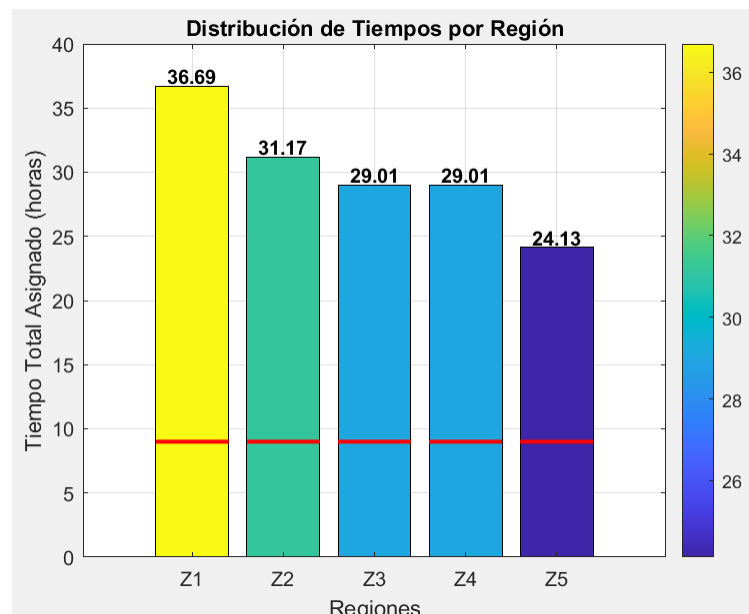


Figura 4-8 Asignación de tiempos con factor meteorológico

Se puede apreciar cómo el modelo asigna un mayor tiempo de vigilancia a la Zona 1, cuyo factor meteorológico es menor; y cómo a la Zona 5 se le asigna un menor tiempo de vigilancia al disponer de un coeficiente meteorológico mayor.

Esto indica que el modelo ha hecho una asignación de tiempos acorde a los parámetros de entrada, los cuales sólo se veían modificados los factores meteorológicos. Con estos resultados se afirma que cuanto mejores condiciones meteorológicas hay en el área a patrullar, mayor coeficiente hay que introducir al programa; y por tanto, menor tiempo de patrulla se le asignará.

Para este último ensayo, el tiempo de ejecución del algoritmo fue de 1.9192 segundos, y el número de iteraciones necesarias para alcanzar la optimización fue 8. Cabe mencionar que esto supone un incremento del 1640% en el tiempo de ejecución con respecto a los ensayos donde sólo se modificaban las probabilidades de las áreas; y el número de iteraciones se ha visto duplicado con respecto a los ensayos realizados anteriormente.

4.3 Análisis 2: Escenario estándar

Se define este segundo análisis bajo el contexto de un escenario convencional, con una serie de parámetros y condiciones estándar, donde se evalúa cómo responde el código bajo condiciones que bien podrían ser reales.

En este análisis se realizan dos ensayos con condiciones estándar. En el primer ensayo se define un escenario real y se mantienen los factores de capacidades y meteorología iguales para todas las áreas. En el segundo ensayo se modifican todos los parámetros para simular condiciones reales y analizar cómo responde el código.

En este sentido, se plantea un escenario en la que se navega bajo las órdenes de ALMART. La Armada despliega operaciones de vigilancia en el STROG, zona de alto interés estratégico debido a su elevada densidad de tráfico marítimo y vulnerabilidad a actividades ilícitas.

El área de patrullaje se divide en cinco regiones estratégicas, determinadas en base a patrones históricos de actividad ilícita y condiciones operativas:

- Región 1: Acceso occidental al STROG, alto tráfico mercante.
- Región 2: Proximidades de la Bahía de Algeciras, potenciales actividades de inteligencia de armadas no aliadas.
- Región 3: Rutas de tránsito de narcolanchas entre Marruecos y la Península.
- Región 4: Zonas de actividad submarina de expolio arqueológico.
- Región 5: Aproximación oriental del STROG, tránsito de buques de interés estratégico.

Se consideran tres tipos de amenazas principales:

- Suceso A: Transporte ilícito de sustancias estupefacientes en embarcaciones de alta velocidad.
- Suceso B: Actividades ilegales de extracción/expolio de recursos submarinos en aguas jurisdiccionales.
- Suceso C: Presencia de unidades con posibles fines de recolección de inteligencia EM.

Bajo este escenario, las prioridades de ALMART y CMOM para la vigilancia marítima son:

- Localizar lanchas rápidas no identificadas de tránsito entre Marruecos y España.
- Proteger nuestro espectro EM y obtener posible inteligencia de armadas no aliadas. Prevenir actividades ilícitas y el expolio ilegal del entorno submarino.



Figura 4-9 Perfil naval BAM Furor P-46 [54]

Este análisis se realiza simulando disponer de un BAM, plataforma utilizada para el patrullaje por la FAM, mostrado en la Figura 4-9. Esta plataforma se plantea como un recurso idóneo para este escenario ya que reúne los requisitos operacionales necesarios para patrullar una zona extensa de manera eficiente, garantizando la detección y respuesta a los eventos simulados en este escenario.

4.3.1 Condiciones estándar sin factor meteorológico

Este ensayo permitirá evaluar el rendimiento del algoritmo de optimización de patrullaje al asignar tiempos de vigilancia en diferentes zonas y comprobar la efectividad del BAM en la detección e intervención ante actividades sospechosas.

Para este ensayo se simulan condiciones meteorológicas y de capacidad iguales para todas las áreas, como se refleja en la Tabla 9.

La distribución del tiempo de patrullaje de la Figura 4-10 refleja una asignación estratégica basada en el nivel de amenaza de cada región, priorizando las áreas con mayor probabilidad de sucesos e importancia por parte de ALMART.

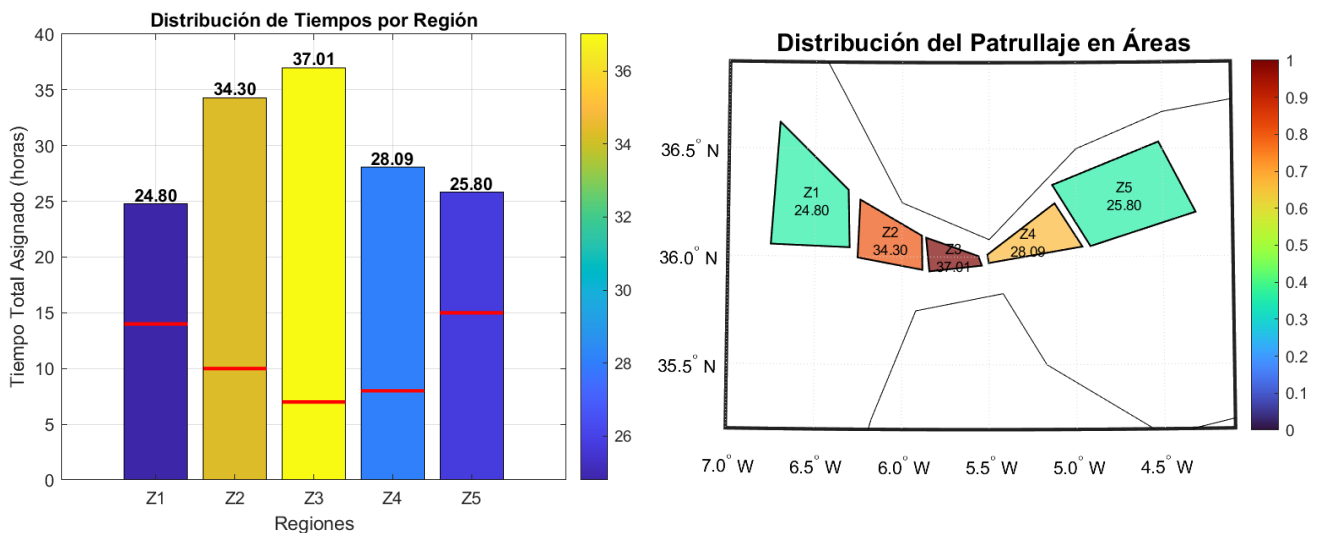


Figura 4-10 Asignación de tiempos en Análisis 2

La Región Z3 recibe la mayor cantidad de horas (37.01), lo que es coherente con su papel como ruta principal de tránsito de narcolanchas, y ser esta la prioridad de la autoridad responsable; mientras que la Región Z2 recibe la segunda mayor cantidad de horas (34.30 horas) debido a la presencia de posibles unidades de recolección de inteligencia EM (Suceso C) y su correspondiente prioridad estratégica.

La Región Z4 también cuenta con un tiempo considerable (28.09 horas), acorde con la necesidad de evitar el expolio arqueológico submarino (Suceso B).

Las Regiones Z1 y Z5, aunque estratégicas por su tráfico mercante y tránsito de buques de interés, reciben una asignación menor (24.80 y 25.80 horas, respectivamente), lo que indica que, si bien son relevantes, el impacto de los sucesos identificados es comparativamente inferior.

La distribución ha sido correcta, ya que garantiza que todas las zonas clave reciban patrullaje suficiente sin exceder los recursos disponibles.

Hasta ahora, los modelos de optimización del patrullaje han priorizado la asignación del tiempo de vigilancia según la criticidad de las zonas. No obstante, la influencia de las condiciones meteorológicas y las capacidades operativas de las unidades constituyen un factor clave que debe ser considerado para mejorar la eficiencia.

4.3.2 Condiciones estándar con factor meteorológico

Este segundo ensayo tiene como objetivo analizar el impacto de la variabilidad meteorológica en la operatividad de los patrulleros y el equipamiento de detección en la efectividad de la vigilancia. Se explora cómo la inclusión de estos factores en un modelo matemático de optimización permite una asignación de recursos más precisa y adaptable a condiciones cambiantes.

Se mantiene el escenario definido en el ensayo anterior y se añaden factores operativos reales tales que:

- El BAM dispone de muy buena capacidad de localizar contactos en superficie.
- Se dispone de un equipo EW de última generación.
- No se dispone de equipo ASW
- El último parte meteorológico informa de una borrasca en el STROG dirigiéndose de W a E.

Estas condiciones quedan reflejadas en los datos de entrada, como se refleja en la Tabla 10, con los que se ejecuta el programa para este ensayo.

Los resultados obtenidos al ejecutar el modelo con los parámetros de entrada del escenario y las condiciones operativas citadas anteriormente, se obtienen los resultados mostrados en la Figura 4-11.

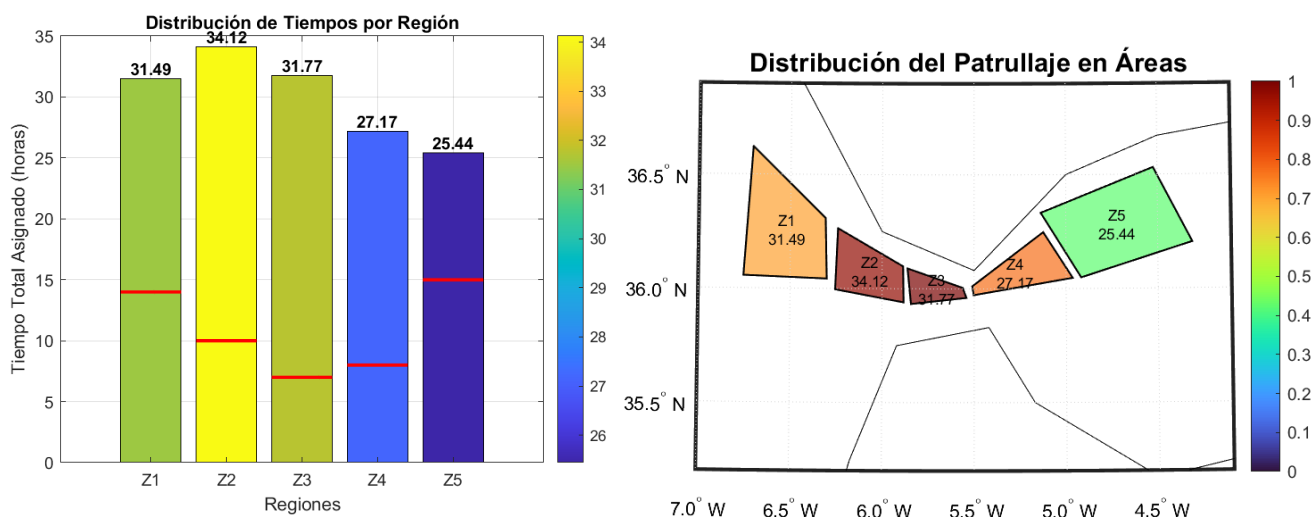


Figura 4-11 Asignación de tiempos escenario estándar

Los gráficos de la Figura 4-11 reflejan una distribución optimizada del tiempo de patrullaje basada en la combinación de factores de riesgo, condiciones meteorológicas y disponibilidad de recursos.

En particular, se observa que Z2 y Z3 son las zonas más prioritarias según el modelo, lo que puede deberse a un mayor riesgo de incidentes o a condiciones climáticas adversas que requieren una vigilancia más intensiva. Por otro lado, Z4 y Z5 presentan un menor tiempo asignado, lo que indica que han sido clasificadas como zonas de menor prioridad en la estrategia de patrullaje.

Bajo estas condiciones, el algoritmo ha encontrado la solución más óptima en 0.2059 segundos y ha necesitado 10 iteraciones para ello, lo que supone un incremento del 22% respecto al ensayo anterior.

Este segundo análisis confirma que el algoritmo ha priorizado estratégicamente las zonas con mayor necesidad de patrullaje, asegurando una distribución equitativa y eficiente del tiempo disponible en función de parámetros estándar en la optimización.

4.4 Análisis 3: Saturación operativa

El tercer análisis busca evaluar el rendimiento del modelo en situaciones donde el tiempo de patrullaje disponible se encuentra en su límite máximo permitido. Es decir, se plantean escenarios en los que el algoritmo se ve obligado a distribuir todo el tiempo de patrullaje, T_{max} , entre las diferentes áreas, maximizando la eficiencia operativa sin exceder las restricciones impuestas.

El objetivo de este estudio es analizar la robustez del modelo cuando se encuentra en un régimen de saturación operativa, donde cada unidad de tiempo debe ser utilizada con la máxima eficiencia posible. Para ello, se generarán diversos escenarios en los que T_{max} será ajustado de manera que se vea completamente consumido, considerando diferentes configuraciones de criticidad en las áreas de vigilancia, condiciones meteorológicas y capacidades operativas del sistema.

Este análisis es fundamental para validar la aplicabilidad operativa del modelo en escenarios donde no existe margen de optimización adicional, garantizando que la distribución del tiempo de patrullaje siga siendo efectiva incluso bajo condiciones de máxima exigencia operativa.

4.4.1 Área extensa con T_{max} reducido

En este primer ensayo se evalúa el comportamiento del modelo cuando una de las áreas de patrullaje es desproporcionadamente grande en comparación con el resto, en un escenario donde T_{max} es reducido.

Esta configuración busca analizar cómo el algoritmo distribuye los tiempos extra de patrullaje en condiciones donde una única zona domina la demanda de recursos, dejando a las demás con un margen de asignación significativamente menor.

El objetivo es observar si el modelo prioriza de manera adecuada la cobertura de todas las áreas o si la zona de mayor tamaño monopoliza la distribución del tiempo extra, afectando la vigilancia del resto de la región.

Para este escenario se introducen parámetros de entrada estándar para todas las áreas excepto para la Z5, cuya superficie será 8 veces superior al resto de áreas.

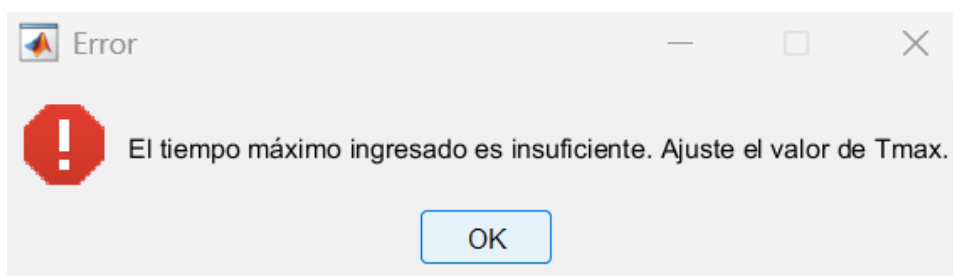


Figura 4-12 Error T_{max} insuficiente

El programa generó la advertencia de la Figura 4-12 indicando que el tiempo máximo ingresado era insuficiente para completar el patrullaje requerido. Este mensaje de error confirma que la validación de restricciones en el código funciona correctamente, evitando asignaciones de tiempo incoherentes.

Este resultado es positivo, ya que demuestra que el código es capaz de detectar y advertir de situaciones en las que los parámetros introducidos no permiten una asignación factible del tiempo de patrullaje.

Ajustando los parámetros de entrada de Z5 hasta que es 5 veces mayor que el resto de áreas, y Tmax es suficiente, se ejecuta el código con los datos de entrada de la Tabla 11, y se obtienen los resultados de la Figura 4-13.

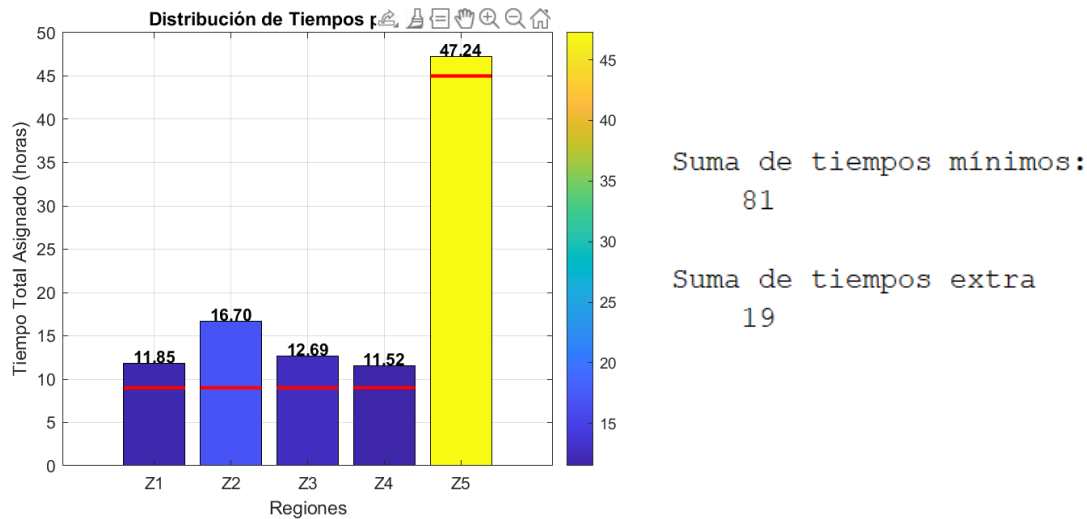


Figura 4-13 Asignación tiempos con área extensa

De la figura 4-13 es especialmente reseñable el valor de la suma de tiempos mínimos de 81 horas, ya que será el valor introducido en el ensayo inmediatamente después para analizar el comportamiento del modelo en una situación en la que solo dispone del Tmax para cubrir el tiempo mínimo de cada área sin capacidad para asignar tiempos extras.

Manteniendo las extensiones anteriores de las regiones y reduciendo el Tmax hasta el mínimo ejecutable por el programa (81.01) mostrado en la Figura 4-13, se ejecuta el programa con los datos de entrada de la Tabla 12, y se obtienen los resultados de la Figura 4-14 donde se observa que el código es capaz de encontrar una solución óptima ante un escenario sin tiempo extra que aplicar.

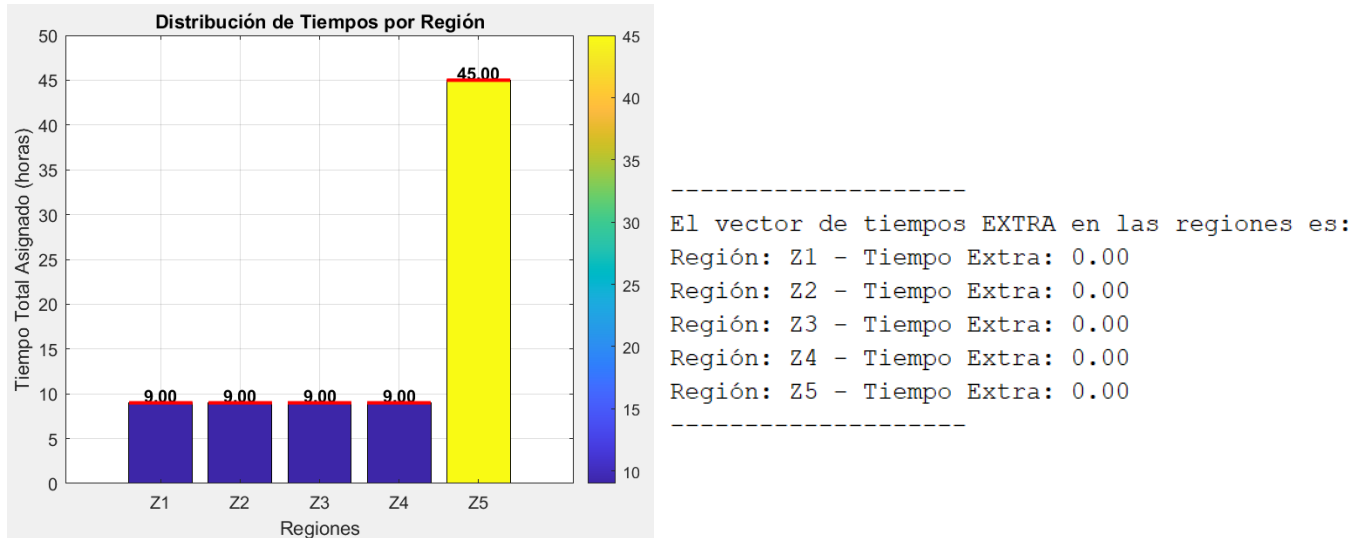


Figura 4-14 Asignación de tiempos sin tiempos extras

Cabe destacar que para esta simulación, el tiempo que tardó el algoritmo en encontrar una solución fue de 0.0454 segundos, notablemente inferior a la media de tiempos empleados en simulaciones anteriores; y el número de iteraciones necesarias fue de 23, notablemente superior a la media de anteriores simulaciones.

4.4.2 Áreas de mucho interés con T_{max} reducido

El propósito de este segundo ensayo es evaluar la capacidad del algoritmo para distribuir el tiempo de patrullaje cuando todas las áreas presentan un alto nivel de interés estratégico.

En este ensayo, se modifican los parámetros de entrada de tal manera que todas las áreas son consideradas como altamente críticas, asignando valores elevados a los coeficientes de probabilidad de sucesos (PA, PB, PC) y a los factores de ponderación WA, WB y WC. De este modo, el código ha sido probado en una situación donde no existen diferencias significativas entre las zonas, pero se trabaja con valores altos, lo que representa un reto para la optimización de los tiempos de patrullaje.

Para este ensayo, se configuraron los siguientes valores:

- Probabilidad de sucesos en todas las áreas: Se estableció en valores máximos (1.6) para asegurar que todas las zonas presenten una alta actividad de eventos relevantes.
- Factores de ponderación WA, WB y WC: Se incrementaron uniformemente, asignando gran importancia a todas las amenazas.
- Factor meteorológico: Se mantuvo constante y en valores favorables para evitar que condiciones climáticas influyeran en la distribución del patrullaje.
- Tamaño de las áreas: Permanece homogéneo para evitar que el tamaño afectara la asignación de tiempos.

Tras la ejecución del modelo con esta configuración de entrada, los resultados mostrados en la figura 4-15 indican que el código ha distribuido el tiempo de patrullaje de manera equilibrada entre todas las áreas, sin favorecer ninguna en particular, lo que indica que el algoritmo ha interpretado correctamente la equidad en las condiciones.

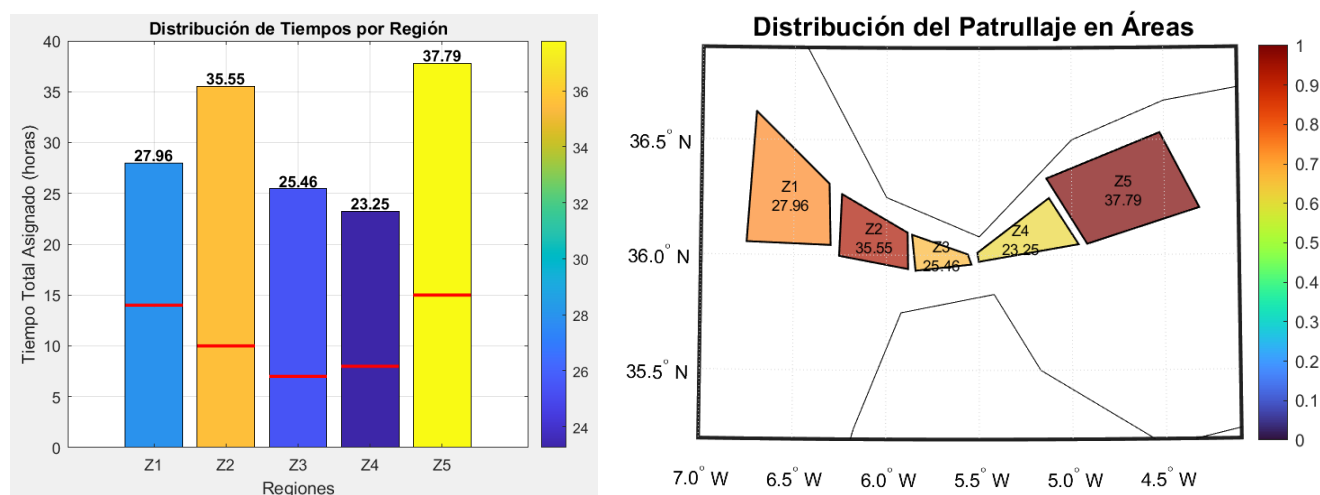


Figura 4-15 Asignación de tiempos en saturación operativa

No se ha generado una asignación desproporcionada de tiempos hacia ninguna región, demostrando que el método de optimización utilizado es capaz de adaptarse incluso cuando las diferencias entre zonas son mínimas o inexistentes.

El algoritmo ha mantenido tiempos de cálculo óptimos, sin un aumento significativo en el número de iteraciones respecto a ensayos anteriores.

Se ha alcanzado una solución óptima, lo que significa que el código ha encontrado una distribución válida que cumple con las restricciones impuestas, a pesar de que puede que en un entorno operativo real, esta distribución de tiempos no sea suficiente para cubrir todas las áreas de manera óptima al haberse alcanzado el estado de saturación operativa.

4.5 Análisis 4: Sensibilidad a variabilidad meteorológica

El propósito de este ensayo es evaluar la sensibilidad del código ante variaciones extremas en las condiciones meteorológicas de las áreas patrulladas. En operaciones reales, las condiciones climáticas afectan significativamente la eficiencia de la vigilancia marítima, ya que factores como tormentas, visibilidad reducida y oleaje pueden limitar la operatividad de las unidades.

Para este análisis, se introducen factores meteorológicos altamente variables entre las zonas de patrullaje, desde condiciones muy adversas hasta óptimas, con el fin de comprobar si el algoritmo ajusta correctamente la distribución del tiempo de vigilancia en función de estos cambios.

Se configuraron los siguientes parámetros de entrada:

- Factores meteorológicos: Se establecen variaciones extremas con valores que oscilan entre 0.5 y 1.5, representando desde tormentas severas hasta condiciones ideales. Se distribuyen de la forma siguiente:
 - Z1: 0.5
 - Z2: 0.8
 - Z3: 1
 - Z4: 1.3
 - Z5: 1.5
- Probabilidades de sucesos (PA, PB, PC): Se mantienen constantes en todas las regiones para aislar el impacto del factor meteorológico.
- Ponderaciones (WA, WB, WC): Se establecen valores homogéneos para evitar que otros parámetros influyan en la asignación de tiempos.
- Tiempo total disponible (Tmax): Se fija en un valor medio dentro del rango habitual de simulaciones anteriores.

Se ejecuta el código con los datos de entrada del usuario mostrados en la Tabla 13.

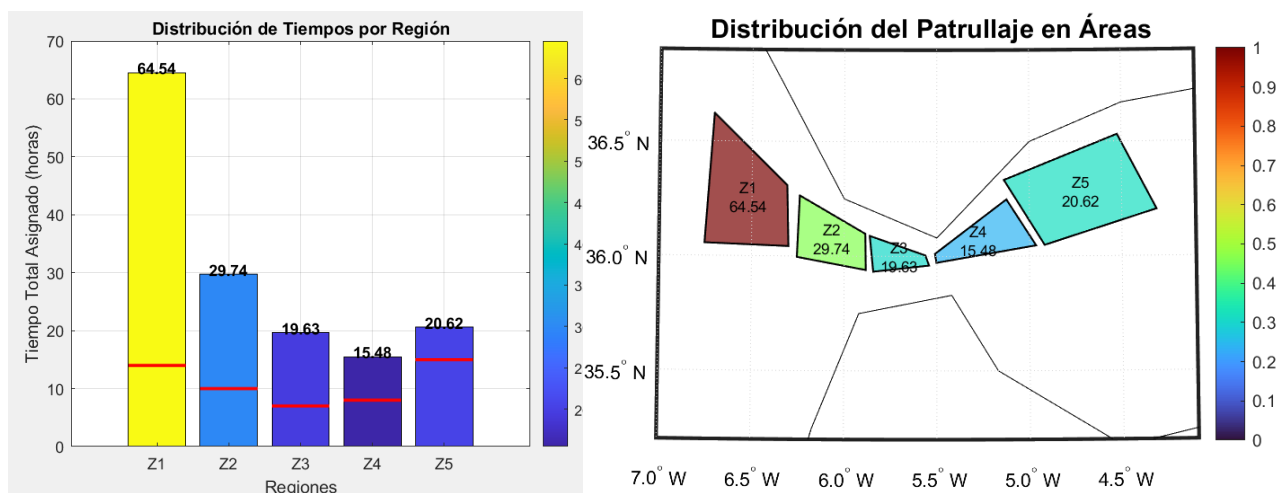


Figura 4-16 Asignación de tiempos con meteorología cambiante

El análisis de los resultados mostrados en la Figura 4-16 confirma que el algoritmo ha respondido correctamente a la variabilidad meteorológica en la asignación del tiempo de patrullaje.

Dado que el factor meteorológico es inversamente proporcional a la facilidad de búsqueda, se esperaba que las zonas con peores condiciones meteorológicas (Z4 y Z5) recibieran más tiempo de patrullaje, mientras que las áreas con mejor clima (Z1 y Z2) experimentarían una reducción en el tiempo asignado.

Los resultados obtenidos han validado esta hipótesis, lo que indica que el modelo ha interpretado adecuadamente la influencia del clima en la optimización del patrullaje.

Sin embargo, la asignación del tiempo no ha dependido únicamente de la meteorología, sino que también se ha visto influenciada por el tamaño de las áreas, lo que ha llevado a que Z1, posiblemente una de las zonas más extensas, reciba la mayor cantidad de tiempo asignado, independientemente de sus condiciones climáticas favorables.

Para verificar la correcta asignación del código, es esencial, en este caso, analizar el tiempo extra asignado, que se representa en la gráfica de la Figura 4-16 por la porción superior de cada barra sobre la línea roja. En este sentido, se observa que Z5 y Z4 han recibido menos tiempo extra, lo que indica que el algoritmo ha priorizado correctamente las áreas con condiciones meteorológicas más adversas.

Esto confirma que el modelo ha funcionado según lo esperado, optimizando la distribución de recursos en función de la dificultad de búsqueda y ajustando la asignación de patrullaje de manera coherente con las condiciones climáticas y el tamaño de las zonas.

4.6 Análisis 5: Robustez ante asignaciones extremas de Tmax

En este análisis se evalúa la capacidad del algoritmo para gestionar eficientemente el tiempo máximo disponible bajo dos condiciones extremas.

4.6.1 Tmax insuficiente

En este primer ensayo se analiza cómo el modelo distribuye los recursos cuando el tiempo disponible es significativamente menor al requerido para patrullar las áreas definidas. Esto permitirá comprobar si el algoritmo prioriza adecuadamente las zonas de mayor importancia y si es capaz de ajustar la asignación de tiempo de manera eficiente bajo restricciones severas.

Cuando el tiempo máximo permitido es inferior a la suma del tiempo mínimo requerido por todas las regiones, el algoritmo se enfrenta a un problema de insuficiencia de recursos. En este contexto, la optimización del tiempo de patrullaje se vuelve crítica, ya que es imposible cubrir de manera equitativa todas las áreas de vigilancia.

Se introducen valores realistas, asegurando que la suma del tiempo mínimo total requerido sea superior al Tmax inicial. Se asignan probabilidades de eventos, factores de ponderación y tamaños de zonas, mientras que Tmax se establece en un valor insuficiente, como puede ser 20.

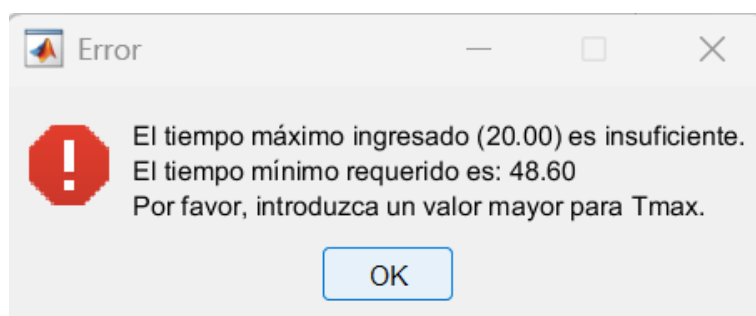


Figura 4-17 Advertencia error Tmax insuficiente

Durante la ejecución, el código calcula el tiempo mínimo necesario y lo compara con el Tmax introducido. Al ser menor, se genera el mensaje de error, como muestra la Figura 4-17, que informa al usuario del valor mínimo requerido y solicita una nueva entrada.

El usuario entonces introduce un nuevo Tmax válido (por ejemplo, 110), permitiendo que la optimización continúe. Es importante destacar que, si en el mensaje de advertencia se recomienda un Tmax mínimo de 48.60, como ocurre en la Figura 4-17, se debe introducir un valor ligeramente superior, como 48.61, para que el modelo pueda ejecutarse correctamente.

Cuando el usuario introduce un nuevo Tmax siguiendo la recomendación del mensaje de error, el código se ejecuta correctamente sin generar nuevas advertencias ni interrupciones, como muestra la Figura 4-18. En este caso, el modelo procede a realizar la optimización de tiempos de patrullaje de manera eficiente, distribuyendo los recursos según los criterios establecidos.

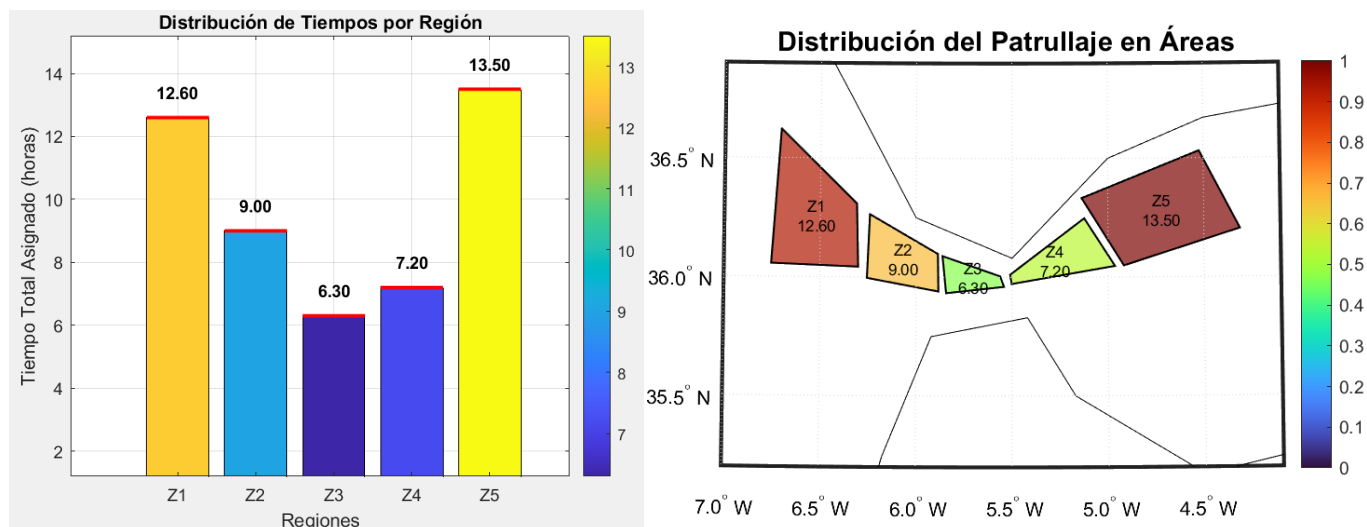


Figura 4-18 Asignación de tiempos tras error

Este comportamiento confirma que el sistema de validación funciona adecuadamente, asegurando que la optimización solo se realice cuando el tiempo disponible es suficiente para cubrir todas las zonas de patrullaje de manera efectiva.

Como resultado, el código previene la ejecución con valores no factibles, asegurando que la optimización se realice solo con un Tmax adecuado.

Este mecanismo mejora la robustez del modelo, evitando asignaciones imposibles y garantizando la funcionalidad en escenarios de restricciones severas.

4.6.2 Tmax excesivo

Este segundo ensayo se enfoca en evaluar el comportamiento del modelo cuando el tiempo máximo disponible (Tmax) es considerablemente superior al necesario para cubrir todas las zonas de patrullaje.

A diferencia del primer ensayo, en el que se probó la reacción del sistema ante una restricción severa de tiempo, en este caso se busca analizar si el algoritmo gestiona eficientemente el exceso de recursos o si, por el contrario, distribuye el tiempo adicional de manera arbitraria o ineficiente.

El usuario introduce un Tmax significativamente alto para evaluar cómo el modelo maneja un exceso de tiempo disponible.

Se observa si el modelo prioriza correctamente las áreas estratégicas o si algunas zonas de menor relevancia reciben tiempos de patrullaje desproporcionados. Para esto, se ejecuta el código con los parámetros de entrada del ensayo anterior y se compara la priorización de las áreas y la asignación de tiempos extra, ya que el orden de prioridades no debería verse modificado.

Si el algoritmo funciona correctamente, debería asignar el tiempo extra de manera racional, maximizando la efectividad de la vigilancia. En caso contrario, se identificarán posibles mejoras en la lógica de optimización para evitar una distribución ineficiente del exceso de recursos.

Los gráficos mostrados en la Figura 4-19, reflejan la distribución de tiempos por región cuando Tmax ha sido incrementado en exceso. Se observa que el modelo no ha mantenido la priorización esperada, ya que las regiones no siguen la misma jerarquía de asignación que en escenarios previos con Tmax más ajustado.

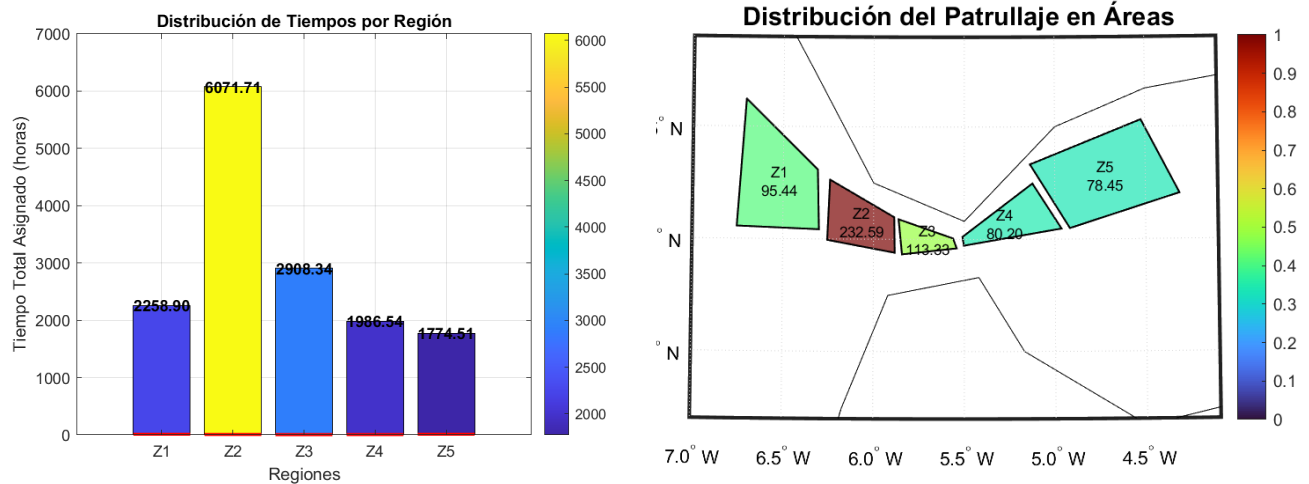


Figura 4-19 Asignación de tiempos con Tmax excesivo

Este resultado indica que el algoritmo, en presencia de un Tmax excesivo, no está optimizando eficientemente la asignación de tiempos, sino que distribuye el recurso de forma más uniforme o sin mantener la lógica de prioridad inicial.

Para corregir este comportamiento, sería necesario introducir restricciones o ajustes adicionales que refuercen la importancia estratégica de cada zona, evitando que el tiempo extra distorsione la distribución ideal.

Esto sugiere que el modelo debe incorporar mecanismos para no sobredimensionar patrullajes en zonas menos relevantes, asegurando una asignación más eficiente del recurso disponible.

4.7 Análisis 6: Prueba de escalabilidad. Modificación del número de áreas

Este análisis busca evaluar si el código mantiene su eficiencia y estabilidad a medida que se incrementa el número de áreas de vigilancia.

Si el código es escalable y robusto, debería manejar sin problemas un mayor número de áreas sin que el tiempo de ejecución se incremente de manera exponencial.

Se quiere determinar si el código mantiene un crecimiento lineal en la evolución del tiempo de ejecución, o si en contraposición aparecen cuellos de botella al incrementar el número de áreas.

También se analiza si el algoritmo sigue alcanzando una solución óptima en un número razonable de iteraciones o si necesita ajustes para evitar una ralentización.

Este análisis permitirá validar si el modelo es eficiente en entornos con mayor cantidad de áreas de vigilancia o si requiere mejoras en su implementación para garantizar un rendimiento óptimo en escenarios más complejos.

4.7.1 Ejecución con 3 áreas

Para esta primera ejecución, se configura el modelo con tres áreas de vigilancia, manteniendo los mismos parámetros de optimización utilizados en escenarios anteriores. Se miden los siguientes factores:

- Tiempo total de ejecución del código.
- Número de iteraciones necesarias para alcanzar la solución óptima.
- Distribución del tiempo de patrullaje entre las tres áreas.

Esta prueba inicial permitirá evaluar si el código opera de manera fluida y rápida cuando el número de áreas es bajo, lo cual servirá como referencia comparativa para determinar cómo varía su desempeño al aumentar la complejidad del problema.

Se ejecuta el código con los parámetros establecidos en ensayos anteriores. Estos parámetros se mantendrán fijos para todo el presente análisis, permitiendo una comparación objetiva entre diferentes ejecuciones.

Como refleja la Figura 4-20, los resultados obtenidos en la ejecución con tres áreas han sido favorables, mostrando que el código ha manejado eficientemente la optimización en este escenario reducido.

Además, el código ha demostrado su adaptabilidad, ya que ha reconocido automáticamente que se han introducido solo tres áreas de patrullaje y, en consecuencia, ha representado únicamente tres regiones en la visualización de los resultados.

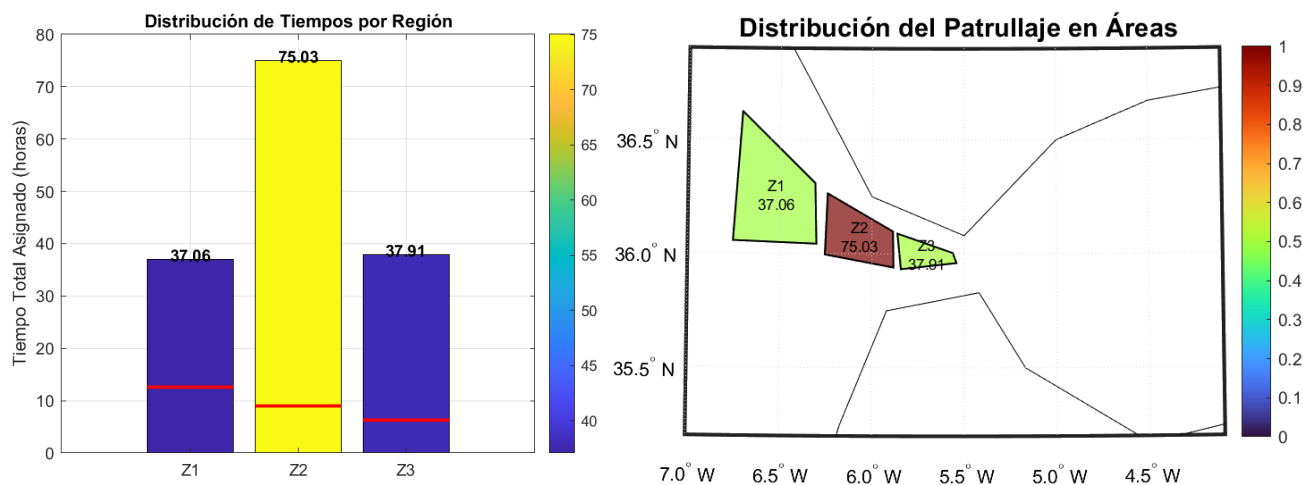


Figura 4-20 Asignación de tiempos con 3 áreas

Este comportamiento confirma que el modelo se ajusta dinámicamente al número de zonas ingresadas, sin necesidad de modificaciones manuales en la estructura del código.

En términos de eficiencia, el código ha ejecutado la optimización en 0.3555 segundos, alcanzando la solución óptima en 6 iteraciones, lo que demuestra un buen desempeño computacional en este escenario. No se han detectado errores ni bloqueos, y el uso de memoria ha permanecido dentro de los valores esperados.

4.7.2 Ejecución con 5 áreas

Este ensayo busca evaluar cómo afecta el incremento del número de áreas de vigilancia al rendimiento del código en términos de tiempo de ejecución, número de iteraciones y estabilidad del algoritmo. En este caso, se aumenta de 3 a 5 áreas de patrullaje, lo que añade más restricciones y variables a la optimización, permitiendo analizar si el modelo mantiene una escalabilidad eficiente.

Se ejecuta el código con los parámetros de entrada por defecto y se registran los datos obtenidos.

Los resultados obtenidos en la ejecución con cinco áreas, mostrados en la Figura 4-21, confirman que el modelo sigue funcionando correctamente, adaptándose al incremento del número de zonas sin comprometer su estabilidad.

En este ensayo cabe destacar que se representan cinco áreas de vigilancia, reflejando que el código ha reconocido automáticamente la nueva configuración y ha ajustado la optimización en consecuencia.

En un principio, este ensayo debería conducirse sin errores del código, ya que supone una ejecución con parámetros estándar al haber sido diseñado el modelo para responder a un patrullaje de la zona con una división de 5 áreas.

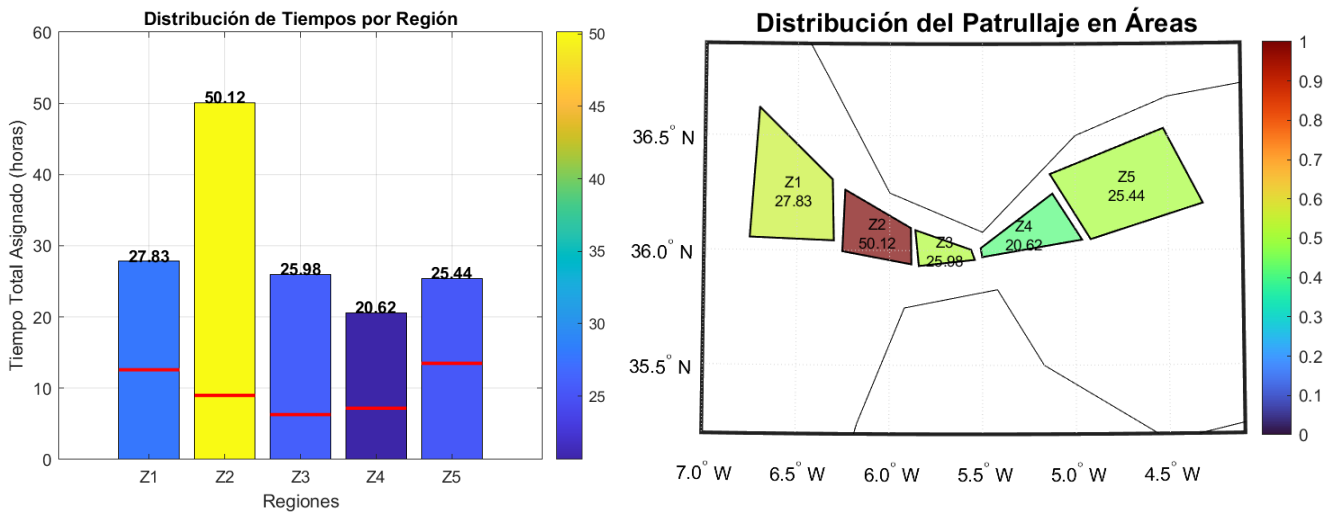


Figura 4-21 Asignación de tiempos con 5 áreas

En términos de rendimiento computacional, se ha registrado un tiempo de ejecución de 1.1757 segundos, lo que representa un incremento respecto a la prueba con tres áreas, pero dentro de un rango esperado dada la mayor cantidad de restricciones y variables a procesar.

Además, el número de iteraciones ha aumentado a 8, lo que indica que el algoritmo requiere más pasos para encontrar la solución óptima, pero sin comprometer su eficiencia.

4.7.3 Ejecución con 8 áreas

Este ensayo tiene como finalidad evaluar el impacto de aumentar a ocho áreas de patrullaje en el rendimiento del código, analizando si se mantiene la eficiencia en términos de tiempo de ejecución, número de iteraciones y estabilidad. Se busca determinar si el crecimiento en el número de zonas afecta de manera lineal o si empiezan a surgir problemas de escalabilidad.

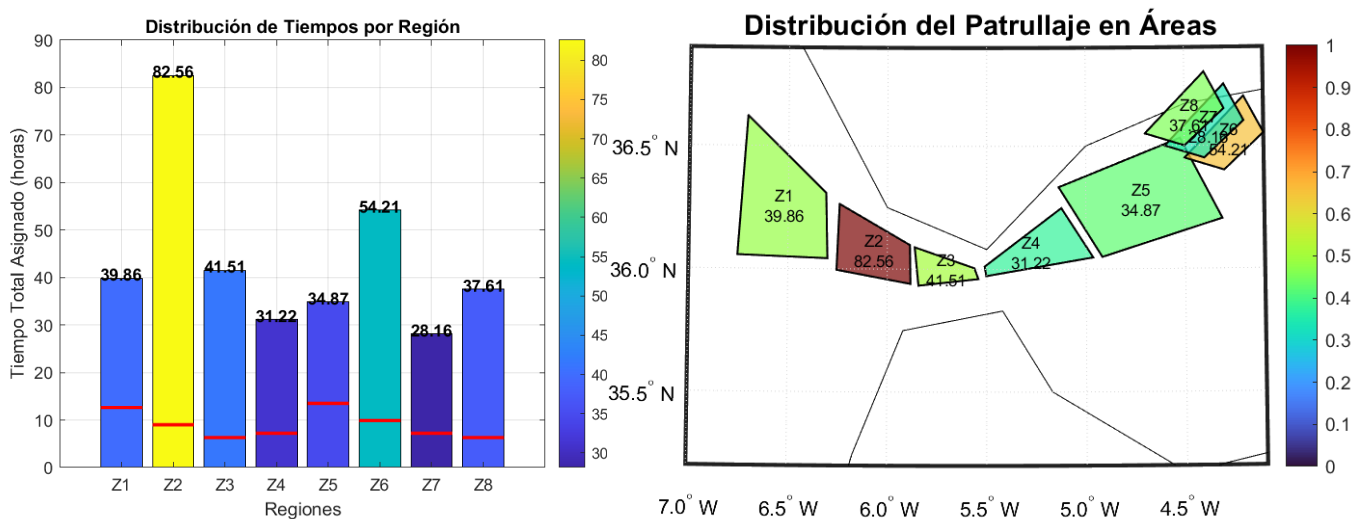


Figura 4-22 Asignación de tiempos con 8 áreas

Los resultados obtenidos y mostrados en la Figura 4-22, reflejan cómo el modelo ha gestionado correctamente la optimización en un entorno con ocho áreas de patrullaje, sin comprometer su estabilidad ni generar errores.

La representación espacial en el mapa confirma que el código ha pintado las ocho áreas en el mapa, el cual supone el número máximo de áreas definidas en el código para ser representadas. Estas 3 áreas

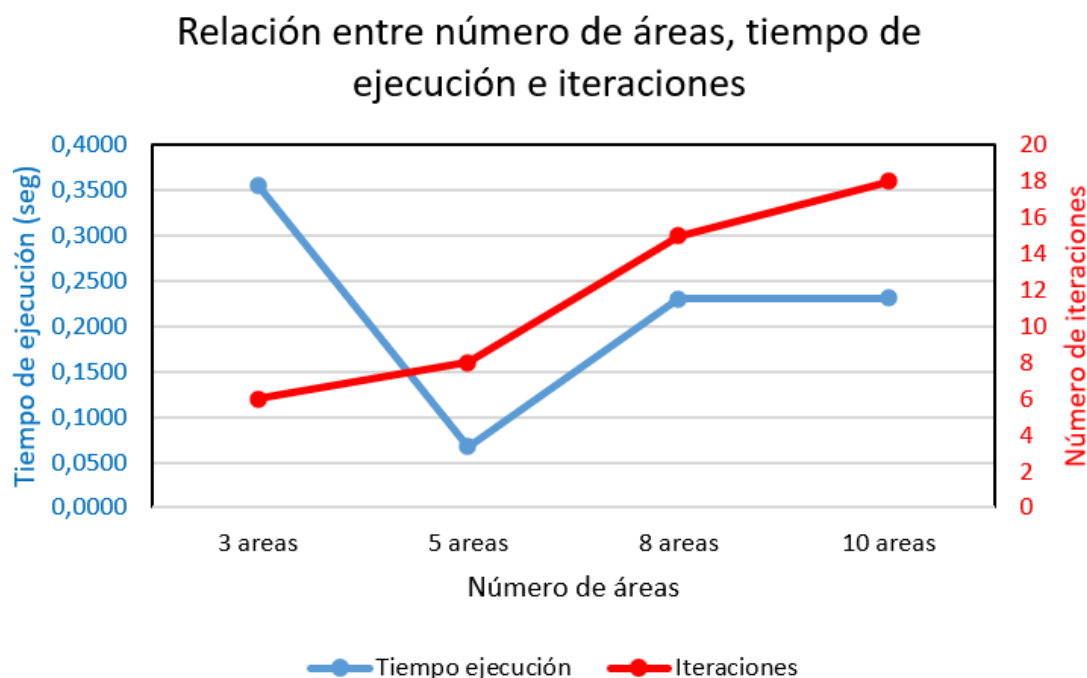


Figura 4-24 Comparación resultados

En la ejecución con 10 áreas de patrullaje, el código ha completado la optimización en 0.2309 segundos y ha requerido 18 iteraciones para alcanzar la solución óptima. Este resultado confirma que el modelo sigue siendo eficiente en términos de cálculo, manteniendo tiempos de ejecución bajos incluso con un mayor número de áreas.

Como se muestra en la Figura 4-24, los ensayos han demostrado que el número de iteraciones aumenta progresivamente con el número de áreas, mientras que el tiempo de ejecución varía ligeramente, manteniéndose en valores bajos. Esto indica que el modelo es eficiente y escalable, capaz de manejar más restricciones sin que el tiempo de cálculo se dispare.

A lo largo de este análisis en el que se fue variando el número de áreas, se identificó una limitación en la representación gráfica, ya que el código solo permite visualizar hasta 8 áreas, aunque la optimización sigue funcionando correctamente con más zonas.

4.8 Análisis 7: Sensibilidad ante parámetros no lógicos

En este análisis se realizan ensayos para evaluar la robustez y estabilidad del modelo cuando el usuario introduce parámetros inválidos o inconsistentes. El objetivo es determinar cómo reacciona el código ante datos erróneos o situaciones poco realistas, asegurando que el sistema pueda gestionar adecuadamente entradas no válidas sin generar fallos críticos o resultados incoherentes.

4.8.1 Datos mal introducidos

Este primer ensayo tiene como objetivo analizar cómo reacciona el código ante la introducción de datos incorrectos, como ingresar una letra en un campo donde se espera un número. Se busca determinar si el sistema detecta el error y emite un mensaje de advertencia, o si por el contrario continúa la ejecución con valores inválidos, lo que podría comprometer la validez de la optimización.

Cabe destacar que el código está diseñado para detectar cuándo una entrada de la interfaz del usuario no es un número, y lanzar un mensaje de advertencia para que se vuelvan a introducir valores.

Para evaluar la respuesta del modelo ante datos mal introducidos, se realizan pruebas en los campos numéricos del sistema, introduciendo valores no válidos y observando el comportamiento del código.

Según puede verse en la Figura 4-25, al ejecutar el código con un parámetro de entrada no numérico el código reconoce que se trata de un error y genera un mensaje de advertencia. Tras generar el mensaje y el usuario aceptar el botón OK, el programa vuelve a mostrar la interfaz de solicitud de datos al usuario. Solo se podrá salir de este bucle cuando todos los parámetros introducidos sean valores numéricos.

Una vez se introducen datos numéricos correctos, el programa se ejecuta como en el Análisis 2: Escenario estándar.

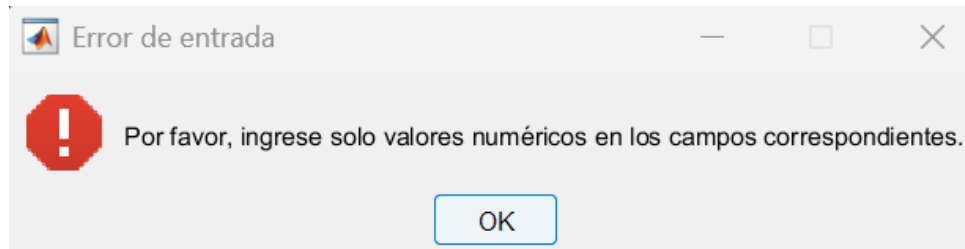


Figura 4-25 Mensaje advertencia

Se verifica que el programa maneja correctamente valores incorrectos y ofrece retroalimentación al usuario antes de continuar con la ejecución.

Durante las pruebas, se introducen valores no numéricos como letras, signos especiales y cadenas de texto aleatorias, y el código ha reaccionado de manera adecuada en cada caso.

Estos mecanismos permiten que el código sea robusto y reduzca la posibilidad de fallos debido a entradas no lógicas, garantizando que el usuario introduzca datos correctos antes de ejecutar la optimización del patrullaje.

4.8.2 Área con probabilidades y factor meteorológico cero

Este ensayo tiene como objetivo evaluar cómo se comporta el modelo cuando una de las áreas de patrullaje tiene una probabilidad de ocurrencia del suceso igual a cero. En un escenario real, esto podría representar una región en la que no se han detectado amenazas en el pasado o una zona de bajo interés operativo.

El propósito del ensayo es comprobar si el modelo asigna correctamente un tiempo mínimo de patrullaje a esa área o si la excluye completamente de la planificación.

Se ejecuta el código con los datos mostrados en la Tabla 14, donde se define Z5 con las probabilidades y el factor meteorológico anulados.

El análisis de la distribución del tiempo de patrullaje, mostrado en la Figura 4-26, revela una deficiencia en el código, ya que, a pesar de que la Z5 tiene probabilidades de ocurrencia de sucesos (PA, PB, PC) iguales a 0 y un factor meteorológico de 0, recibe la mayor cantidad de tiempo de patrullaje (114.90 horas).

Este comportamiento es contrario a la expectativa lógica, donde se esperaría que Z5 no recibiera asignación de tiempo o fuese minimizada.

Se identifican posibles causas de este error, entre ellas, que el modelo esté priorizando el tamaño del área sobre otros factores, omitiendo la restricción del factor meteorológico, o asignando patrullaje de manera equitativa sin considerar adecuadamente las probabilidades de suceso.

Para corregir esta deficiencia, se realiza un nuevo ensayo en la que el factor meteorológico de Z5 no sea cero, con el objetivo de evaluar si el modelo sigue asignando un tiempo desproporcionado a esta zona y verificar si la restricción meteorológica está siendo correctamente implementada.

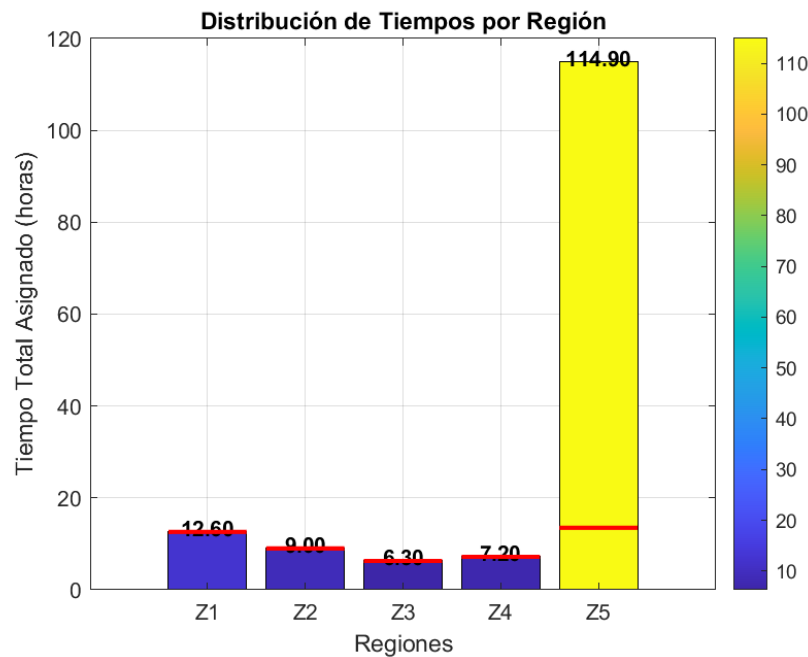


Figura 4-26 Tiempos con Z5 ceros

4.8.3 Área con probabilidades cero

A partir de la anomalía detectada en el ensayo anterior, donde la Z5 recibió un tiempo de patrullaje desproporcionado a pesar de tener probabilidades de ocurrencia de suceso (PA, PB, PC) y factor meteorológico igual a 0, se ha diseñado un nuevo experimento con el objetivo de verificar el impacto real del factor meteorológico en la asignación de tiempos.

Para ello, se mantendrán las probabilidades de suceso en cero, pero se asignará un factor meteorológico distinto de 0, como se relaciona en la Tabla 15, asegurando que la zona no sea completamente inhabilitada por restricciones climáticas.

El objetivo es determinar si el modelo asigna un tiempo de patrullaje reducido a Z5 cuando su factor meteorológico es distinto de 0, lo que permite evaluar si el modelo de optimización está considerando el factor meteorológico como una restricción o si la asignación de tiempos sigue viéndose afectada principalmente por el tamaño del área u otros factores.

Los resultados obtenidos y mostrados en la Figura 4-27 demuestran que Z5 ahora recibe el menor tiempo total y el menor tiempo extra, lo cual es un comportamiento lógico y esperado.

Esto indica que la deficiencia observada en el ensayo anterior se debía, en gran parte, a la asignación de un factor meteorológico igual a 0, lo que posiblemente hacía que el modelo interpretara erróneamente su exclusión de la planificación.

La asignación de tiempo extra a Z5, a pesar de que sus probabilidades de suceso son cero, sugiere que el modelo está considerando otros factores además de la probabilidad de ocurrencia para distribuir el tiempo de patrullaje; como pueden ser la influencia del factor meteorológico o la penalización por desbalance en la optimización.

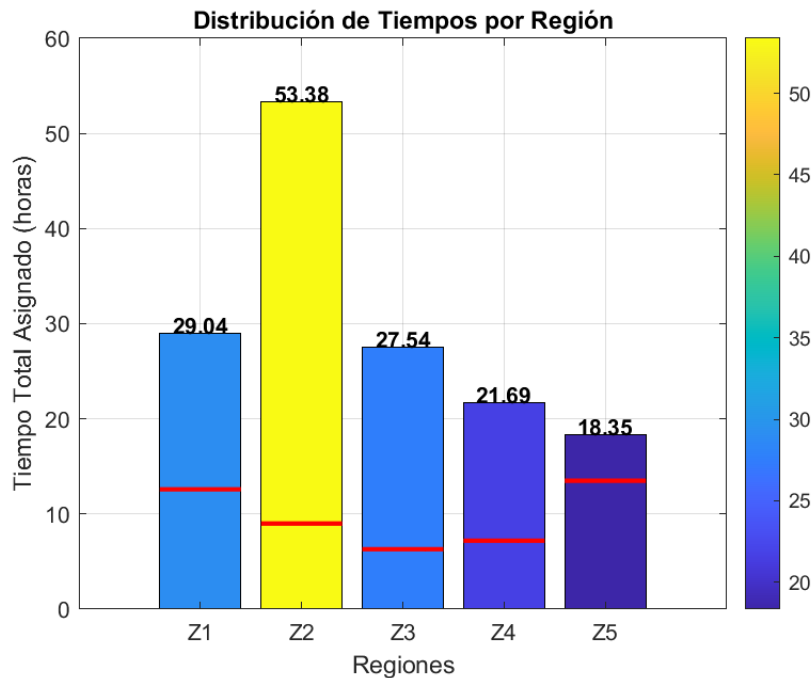


Figura 4-27 Tiempos con Z5 ceros salvo factor meteorológico

Los resultados mostrados tanto en la Figura 4-26 como en la Figura 4-27, confirman que el factor meteorológico influye en la asignación de tiempos de patrullaje. La anomalía en Z5 se debe a que el modelo interpreta erróneamente su exclusión cuando el factor meteorológico es 0. Al asignar un valor distinto de 0, el tiempo de patrullaje se reduce significativamente, validando que el modelo considera este factor en la optimización.

4.9 Validación del código y conclusiones

En este apartado se valida el correcto funcionamiento del modelo de optimización de patrullaje desarrollado en MATLAB y se comparan los resultados obtenidos con diferentes enfoques. La evaluación se realiza desde dos perspectivas principales: verificación del funcionamiento y convergencia del algoritmo.

Esta validación se realiza tras la serie de ensayos previos en los que se han evaluado distintas configuraciones del problema, abarcando variaciones en la cantidad de regiones, restricciones temporales y condiciones meteorológicas.

4.9.1 Verificación del funcionamiento

Para garantizar que el modelo cumple con las restricciones definidas en el problema de optimización, se han realizado pruebas en diferentes escenarios planteados, donde el código puede conducir a una mala ejecución:

- **Tmax insuficiente:** Se estableció un valor de Tmax menor al tiempo mínimo requerido por todas las áreas. Como esperado, el código detectó la incoherencia y generó un mensaje de error, evitando la ejecución del modelo.
- **Tmax excesivo:** En este caso, el código no respondió como se esperaba. En lugar de distribuir el tiempo extra de manera estratégica, se observó una asignación menos racional, en la que ciertas zonas recibían tiempos de patrullaje desproporcionados sin seguir la priorización definida en escenarios previos con los mismos parámetros de entrada. Este comportamiento sugiere la necesidad de refinar el modelo para mejorar la distribución de Tmax cuando es excesivo, utilizando para este caso restricciones de cota superior.

- Probabilidad de sucesos nula: Se verificó que cuando una región tenía probabilidad cero de ocurrencia de eventos, el algoritmo no siempre asignaba únicamente el tiempo mínimo requerido. En ciertos casos, la asignación de tiempos se veía influenciada por otros factores como el tamaño del área y el factor meteorológico. En particular, cuando el factor meteorológico de la región era cero, el código otorgaba tiempos de patrullaje desproporcionados. Sin embargo, al asignar un factor meteorológico distinto de cero, el modelo corregía la distribución y reducía significativamente el tiempo de patrullaje en esas zonas.
- Factores meteorológicos variables: Se evaluaron condiciones climáticas adversas y favorables en distintas regiones. Se observó que el modelo asigna mayor tiempo de patrullaje a aquellas zonas con peores condiciones meteorológicas, en concordancia con la lógica del modelo.
Sin embargo, en casos donde el factor meteorológico se reducía significativamente (por ejemplo 0.5), se observó que la asignación de tiempo de patrullaje tendía a ser mayor de lo esperado en comparación con otras áreas. Aunque esta tendencia es lógica, dado que las condiciones adversas dificultan la vigilancia, sería adecuado contar con un factor limitante para evitar sobreasignaciones por motivo de meteorología.

4.9.2 Análisis de la convergencia del algoritmo

El análisis de la convergencia del algoritmo es un aspecto fundamental para evaluar la estabilidad y eficiencia del modelo de optimización.

Resulta relevante analizar la sensibilidad del algoritmo ante perturbaciones en las condiciones iniciales o en los parámetros de entrada, ya que una convergencia consistente frente a pequeñas variaciones constituye un indicador clave de robustez numérica y estabilidad estructural. Este tipo de análisis permite no solo validar la fiabilidad del algoritmo en condiciones ideales, sino también anticipar su comportamiento en escenarios reales donde la incertidumbre o el ruido en los datos de entrada pueden afectar significativamente la trayectoria de convergencia y la calidad de las soluciones alcanzadas.

Conviene destacar que el estudio de la convergencia debe complementarse con una evaluación del comportamiento del algoritmo frente a distintos perfiles de restricciones y configuraciones del espacio de búsqueda, ya que estos factores pueden alterar significativamente tanto la velocidad de convergencia como la posibilidad de estancamiento en óptimos locales.

Este tipo de análisis comparativo resulta especialmente útil en contextos aplicados, donde la estructura del problema puede variar dinámicamente y se requiere garantizar un rendimiento robusto del algoritmo bajo condiciones no controladas.

En este apartado se estudia el comportamiento del solver fmincon en términos de iteraciones requeridas, estabilidad en distintos escenarios y tiempos de ejecución.

Para realizar este análisis, se utilizan los datos obtenidos en las pruebas previas con diferentes configuraciones. En este sentido, resulta clave la herramienta del código que engloba los resultados en una carpeta y los guarda en el mismo directorio en el que se encuentra el código, haciendo fácil la tarea de consultar datos introducidos y resultados obtenidos en ensayos previos.

La Figura 4-28 muestra el resultado del número de iteraciones empleadas por el modelo, con valores entre 3 y 23. Mientras que el Análisis 1 presentó una convergencia rápida con pocas iteraciones, los Análisis 3 y 5 requirieron el mayor número de iteraciones (23), indicando mayor complejidad. El Análisis 6, referido al aumento de regiones muestra un incremento progresivo, alcanzando 18 iteraciones.

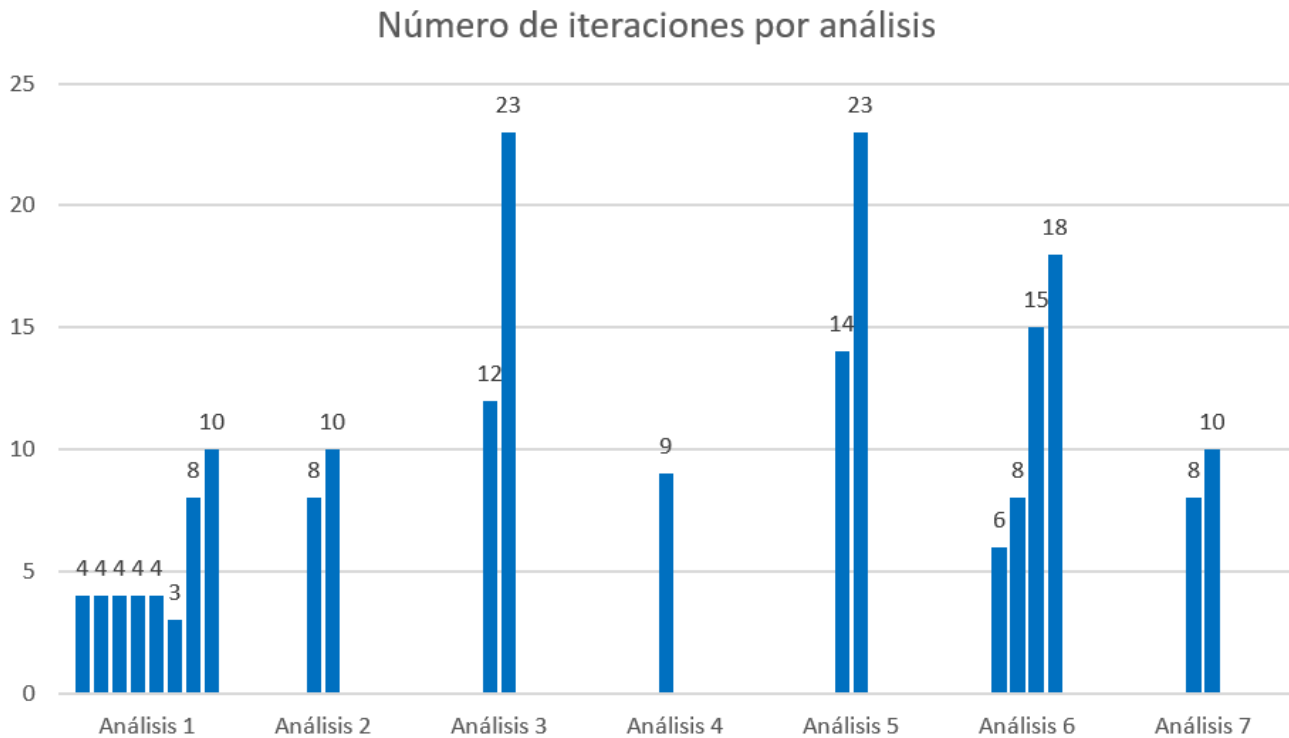


Figura 4-28 Número de iteraciones por análisis

Esto sugiere que la eficiencia del algoritmo varía según el caso, y que algunos análisis pueden requerir ajustes en los parámetros o mejores condiciones iniciales para optimizar la convergencia.

El análisis de la convergencia del algoritmo confirma que en todas las ejecuciones el modelo siempre devolvió un $\text{exitflag} = 1$, lo que indica que la optimización finalizó exitosamente en cada caso.

Esto demuestra la fiabilidad y estabilidad del método empleado, asegurando que siempre se alcanzó una solución óptima dentro de los parámetros establecidos. A pesar de la variabilidad en el número de iteraciones, la consistencia en la convergencia valida la efectividad del algoritmo y su capacidad para resolver el problema planteado sin interrupciones o fallos en la optimización.

4.10 Análisis de desempeño

4.10.1 Tiempo de ejecución

El tiempo de ejecución del algoritmo es un parámetro crítico, especialmente en escenarios operativos donde la rapidez en la toma de decisiones es fundamental.

Los resultados experimentales han mostrado que el tiempo de ejecución varía en función de:

- **Número de regiones a optimizar:** A medida que aumenta el número de zonas de patrullaje, el tiempo de ejecución incrementa de forma no lineal.
- **Restricciones impuestas en el modelo:** Cuando el problema tiene restricciones más estrictas, el tiempo de optimización se prolonga.
- **Valores iniciales:** Un punto inicial más cercano a la solución óptima permite una convergencia más rápida.
- **Número de iteraciones:** Cuando el código emplea más iteraciones necesita de más tiempo de ejecución.

En pruebas realizadas, el tiempo medio de ejecución se mantiene entre 0.2 y 1.5 segundos, dependiendo de la complejidad del problema.

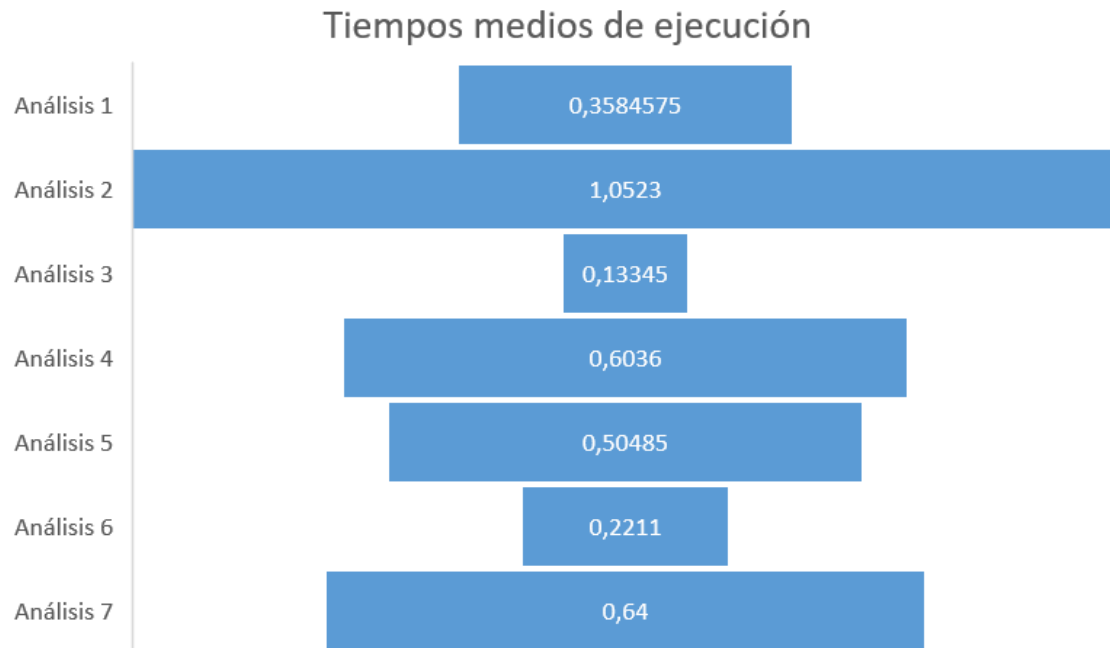


Figura 4-29 Tiempos medios de ejecución

La Figura 4-29 muestra los tiempos medios de ejecución del algoritmo en cada análisis, evidenciando una variabilidad significativa entre ellos. La mayoría de los análisis presentan tiempos medios entre 0,2 y 0,64 segundos, lo que demuestra que el algoritmo es eficiente en la mayoría de los casos. La variabilidad observada sugiere que ciertos análisis pueden beneficiarse de ajustes en los parámetros de inicialización o en la configuración del optimizador para reducir el tiempo de ejecución en escenarios más complejos.

5 CONCLUSIONES Y LÍNEAS FUTURAS

5.1 Conclusiones

El presente trabajo ha permitido el desarrollo y validación de una herramienta computacional orientada a la optimización del tiempo de patrullaje en escenarios de vigilancia marítima bajo condiciones de recursos limitados. Se ha diseñado un modelo matemático que permite asignar de forma eficiente el tiempo de vigilancia a distintas zonas geográficas, considerando factores operativos, meteorológicos y de criticidad estratégica.

Los análisis realizados en el apartado anterior, han puesto de manifiesto la validez funcional y operativa del modelo, el cual ha respondido de forma adecuada a la variabilidad de parámetros de entrada y a las diferentes restricciones impuestas.

En concreto, los resultados han demostrado que:

- El código es robusto ante escenarios realistas, ajustándose correctamente a factores como la capacidad del buque, la criticidad de los eventos o la meteorología. Pruebas de sensibilidad han evidenciado cómo pequeñas variaciones en estos parámetros se traducen en ajustes coherentes en la asignación del tiempo.
- Se ha verificado que el algoritmo implementado, basado en la función `fmincon`, converge consistentemente a soluciones óptimas en todos los escenarios planteados, retornando un `exitflag` = 1 de manera sistemática. Además, el número de iteraciones se ha mantenido en un rango aceptable (entre 3 y 23 iteraciones), garantizando la eficiencia computacional.
- El código detecta automáticamente errores en la entrada de datos y valida las condiciones mínimas necesarias para el cálculo, como en los casos de tiempo máximo insuficiente (T_{max}). Esta validación previene ejecuciones erróneas y refuerza la fiabilidad del sistema.
- En escenarios con distribución de probabilidades y capacidades diferenciadas, el algoritmo ha demostrado una discriminación lógica y eficaz en la asignación de tiempos, priorizando de forma operativa las zonas más críticas.

Asimismo, el sistema ha evidenciado limitaciones relevantes que se deben considerar en futuras versiones. Por ejemplo, cuando se dispone de un T_{max} excesivo, el algoritmo pierde sensibilidad y puede asignar tiempos de forma menos priorizada. Este fenómeno sugiere incluir penalizaciones adicionales que impidan sobredimensionamientos innecesarios. Del mismo modo, la representación gráfica está actualmente limitada a un máximo de 8 zonas, lo que restringe la escalabilidad visual del sistema a escenarios más complejos, aunque la optimización numérica permanece funcional.

En términos de rendimiento, los tiempos de ejecución han sido competitivos, variando entre 0.2 y 1.5 segundos para problemas de hasta 10 zonas. Esto demuestra la viabilidad del código para su aplicación en entornos operativos en tiempo casi real, especialmente tras la paralelización de procesos como la generación de interfaces auxiliares.

A nivel conceptual, se concluye que el sistema desarrollado mejora sustancialmente la eficiencia en la asignación de recursos de vigilancia marítima, respetando criterios técnicos, operativos y logísticos. Asimismo, el modelo es adaptable a distintos escenarios, lo que lo convierte en una herramienta versátil con potencial de aplicación más allá del caso de estudio, siendo útil en cualquier proceso de asignación óptima de vigilancia con restricciones.

Desde un punto de vista metodológico, el trabajo ha servido para poner en práctica los conocimientos adquiridos en ingeniería mecánica y programación, aplicándolos a un problema de interés estratégico y real en nuestro ámbito profesional, como es la seguridad marítima.

En resumen, se ha logrado cumplir con éxito los objetivos planteados: el código permite al usuario introducir múltiples parámetros, simular diversos contextos y obtener una solución optimizada de patrullaje tanto en términos numéricos como gráficos. Este TFG, por tanto, no solo resuelve el problema planteado, sino que abre la puerta al desarrollo de sistemas inteligentes de planificación operativa, que podrían integrar en el futuro técnicas de aprendizaje automático, datos en tiempo real o modelos multiobjetivo.

5.2 Líneas futuras

El presente trabajo ha demostrado la eficacia de un modelo de optimización para la distribución del tiempo de patrullaje en entornos marítimos complejos y con recursos limitados. Sin embargo, este modelo representa una base sobre la que pueden construirse desarrollos más avanzados, alineados con los retos operativos y tecnológicos actuales y futuros en el ámbito de la seguridad marítima.

Una de las líneas de desarrollo más prometedoras consiste en la incorporación de la dimensión temporal en los parámetros de entrada. En el modelo actual, tanto las probabilidades de ocurrencia de sucesos como el factor meteorológico son tratados como constantes. No obstante, en un entorno real, estos parámetros evolucionan con el tiempo. Por tanto, una extensión lógica del modelo sería permitir que ambos factores se modelen como variables dinámicas, dependientes del instante o franja horaria de patrullaje.

Para ello, se podría integrar una estructura de datos temporal (por ejemplo, un vector por zona con discretización horaria) y adaptar la función objetivo para que evalúe una distribución de tiempos en función del valor que estos parámetros presentan en cada intervalo. Esta propuesta convertiría el problema en una optimización por tramos temporales, permitiendo al sistema adaptarse a variaciones diarias, estacionales o a eventos puntuales como tormentas, alertas de seguridad o migraciones estacionales.

Además, el sistema podría evolucionar hacia una solución reactiva en tiempo real mediante la integración de datos externos como sistemas AIS. La información proporcionada por estas plataformas (como pueden ser posición, velocidad o tipo de buque) permitiría detectar cambios en el entorno operativo (congestión de tráfico, presencia de embarcaciones sospechosas o no identificadas, desvíos de rutas habituales), ajustando dinámicamente el plan de patrullaje en función del contexto actual.

Otra dirección interesante consiste en la incorporación de modelos predictivos para anticipar la aparición de eventos de interés, basados en datos históricos o inteligencia previa. La implementación de técnicas de aprendizaje automático permitiría inferir tendencias y adaptar de forma anticipada la asignación de vigilancia, mejorando la capacidad de prevención.

Por último, el modelo podría ampliarse hacia problemas multiobjetivo, donde no solo se busque minimizar la dispersión del patrullaje, sino también, por ejemplo, minimizar el consumo energético, el desgaste de la plataforma o la distancia total recorrida. Este tipo de enfoque requeriría reformular la función objetivo como una combinación ponderada de distintos criterios o mediante técnicas específicas.

Las líneas futuras aquí planteadas permitirían transformar el modelo actual en una herramienta inteligente, adaptativa y de decisión táctica para entornos de vigilancia marítima. La viabilidad computacional demostrada y la estructura modular del código desarrollado lo convierten en una base sólida sobre la que edificar evoluciones tecnológicas avanzadas con impacto directo en la seguridad marítima nacional.

6 BIBLIOGRAFÍA

- [1] Puertos del Estado, "Estadísticas" [En línea]. Disponible en: <https://www.puertos.es/datos/estadisticas>. [Accedido: 12 de febrero de 2025].
- [2] Puertos del Estado, "Estadísticas Anuales," [En línea]. Disponible en: <https://www.puertos.es/datos/estadisticas/anuales>. [Accedido: 26 de febrero de 2025].
- [3] Puertos del Estado, "El sector portuario español genera un impacto económico de 24.304 millones de euros y 250.000 empleos," [En línea]. Disponible en: <https://www.puertos.es/comunicacion/el-sector-portuario-espanol-genera-un-impacto-economico-de-24304-millones-de-euros-y-250000-empleos>. [Accedido: 11 de febrero de 2025].
- [4] Oficina de Naciones Unidas contra la Droga y el Delito (UNODC), "Publications & Resources," [En línea]. Disponible en: https://www.unodc.org/unodc/en/piracy/resources_Template.html. [Accedido: 22 de febrero de 2025].
- [5] Armada Española, "Las misiones marítimas de la Armada", *Ministerio de Defensa*, [En línea]. Disponible en: <https://armada.defensa.gob.es/ArmadaPortal/page/Portal/ArmadaEspañola/conocenosorganizacion/prefLang-es/03Flota--04FAM> [Accedido: 12 de febrero de 2024]
- [6] InfoDefensa, "La nueva Estrategia Nacional de Seguridad Marítima y la necesidad de una Guardia Costera", [En línea]. Disponible en: <https://www.infodefensa.com/texto-diario/mostrar/4903583/nueva-estrategia-nacional-seguridad-maritima-necesidad-guardia-costera>. [Accedido: 28 de enero de 2025].
- [7] INTERPOL, "Delincuencia marítima" [En línea]. Disponible en: <https://www.interpol.int/es/Delitos/Delincuencia-maritima>. [Accedido: 12 de febrero de 2025].
- [8] EL PAÍS, "26 días de travesía en un narcosubmarino", 12-febrero-2025. [En línea]. Disponible en: <https://elpais.com/especiales/2019/26-dias-de-travesia-en-un-narcosubmarino>. [Accedido: 12 de febrero de 2025].
- [9] EMODnet, "EMODnet Map Viewer," *European Marine Observation and Data Network* [En línea]. Disponible en: <https://emodnet.ec.europa.eu/geoviewer/>. [Accedido: 11 de febrero de 2025].
- [10] M. G. Sánchez, "Vigilancia y control marítimo: optimizando la seguridad en el mar," *Militar y Arsenal*, vol. 12, pp. 45-60, 2025.

- [11] Frontex, "Maritime Surveillance," *European Border and Coast Guard Agency*. [En línea]. Disponible en: <https://www.frontex.europa.eu/what-we-do/operations/operations/?ettrans=es>. [Accedido: 28 de enero de 2025].
- [12] Organización Marítima Internacional (OMI), "Seguridad marítima," *IMO* [En línea]. Disponible en: <https://www.imo.org/es/OurWork/Safety/Paginas/Default.aspx>. [Accedido: 28 de enero de 2025].
- [13] Ministerio de Defensa de España, *Estrategia de Seguridad Nacional Marítima (ENSM) 2024*, Madrid, España, 2024. Disponible en: <https://www.dsn.gob.es/sites/default/files/2024-07/ACCESIBLE%20CASTELLANO%20ENSM%202024.pdf> . [Accedido: 16 de febrero de 2025]
- [14] L. Etienne, A. Hjelmfelt, R. Pelot y M. Fournier, *Global Maritime Situational Awareness*. Turkish Naval Forces, junio 2014, p. 75.
- [15] El HuffPost, "Europa busca acabar con la amenaza de la ‘flota en la sombra’ de Putin," *Global*, [En línea]. Disponible en: <https://www.huffingtonpost.es/global/europa-busca-acabar-amenaza-flota-sombra-putin.html>. [Accedido: 15 de febrero de 2025].
- [16] Cadena SER, "Intervienen en aguas de Ibiza un alijo de 34 kilos de cocaína," [En línea]. Disponible en: <https://cadenaser.com/baleares/2025-de-febrero-de-12/intervienen-en-aguas-de-ibiza-un-alijo-de-34-kilos-de-cocaina>. [Accedido: 15 de febrero de 2025].
- [17] Real Instituto Elcano, "La lucha contra la piratería en Somalia: el problema persiste a pesar del esfuerzo militar," [En línea]. Disponible en: <https://www.realinstitutoelcano.org/analisis/la-lucha-contra-la-pirateria-en-somalia-el-problema-persiste-a-pegar-del-esfuerzo-militar-ari/>. [Accedido: 15 de febrero de 2025].
- [18] Cofayx, "La Armada Española: Esencial en el Control del Tráfico Marítimo," *Tablada Centenaria Aviación*, [En línea]. Disponible en: <https://tabladacentenariaaviacion.es/la-armada-espanola-y-su-papel-en-el-control-del-traffic-maritimo/>. [Accedido: 28 de enero de 2025].
- [19] Ministerio de Defensa de España, "La visión estratégica de la Armada: ‘Armada 2050’," vol. 1, 2025.
- [20] Aeronaves Militares Españolas, "SCAN EAGLE," [En línea]. Disponible en: <https://aeronavesmilitaresespanolas.com/scan-eagle-armada-espanola/>. [Accedido: 28 de enero de 2025].
- [21] A. C. de los Ríos, "Los Sistemas Marítimos No Tripulados, la Armada y la industria española de defensa," *Revista Ejércitos*, [En línea]. Disponible en: <https://www.revistaejercitos.com/articulos/los-sistemas-maritimos-no-tripulados-la-armada-y-la-industria-espanola-de-defensa/> [Accedido: 23 de febrero de 2025].
- [22] CESEDEN, "La visión estratégica de la Armada: ‘Armada 2050’," *Ministerio de Defensa de España*, [En línea]. Disponible en: https://www.defensa.gob.es/ceseden/-/la_vision_estrategica_de_la_armada_2050. [Accedido: 13 de febrero de 2025].
- [23] R. D. InfoDefensa, "Gral. Millán (Cestic): ‘IA, nube, computación cuántica y 5G serán los habilitadores del combate moderno’," *Infodefensa*, [En línea]. Disponible en: <https://www.infodefensa.com/texto-diario/mostrar/4248802/gral-millan-cestic-inteligencia-artificial-nube-computacion-cuantica-5g-seran-habilitadores-combate-moderno-y2>. [Accedido: 13 de febrero de 2025].
- [24] A. C. de los Ríos, "Los Sistemas Marítimos No Tripulados, la Armada y la industria española de defensa," *Revista Ejércitos*, [En línea]. Disponible en: <https://www.revistaejercitos.com/articulos/los-sistemas-maritimos-no-tripulados-la-armada-y-la-industria-espanola-de-defensa/>. [Accedido: 13 de febrero de 2025].

- [25] Defensa.com, "El Sead 23 de Seadrone, integrado en el grupo Dédalo 23 de la Armada," *Defensa.com*, [En línea]. Disponible en: <https://www.defensa.com/espana/usv-sead-23-seadrone-grupo-anfibio-aeronaval-dedalo-23-armada>. [Accedido: 29 de enero de 2025].
- [26] L. M. Petry, A. Soares, V. Bogorny, B. Brandoli, y S. Matwin, "Challenges in Vessel Behavior and Anomaly Detection: From Classical Machine Learning to Deep Learning," *arXiv*, vol. 2004.03722, 7-abril-2020. doi: 10.48550/arXiv.2004.03722.
- [27] Comisión Europea, "Una estrategia de seguridad marítima de la UE reforzada para hacer frente a unas amenazas marítimas cambiantes," *EUR-Lex*, [En línea]. Disponible en: <https://eur-lex.europa.eu/legal-content/ES/TXT/HTML/?uri=CELEX%3A52023JC0008>. [Accedido: 15 de febrero de 2025].
- [28] El HuffPost, "La joya de la flota de la Guardia Civil contra el narcotráfico se rompe," *ElHuffpost*, [En línea]. Disponible en: <https://www.huffingtonpost.es/politica/la-joya-flota-guardia-civil-narcotrafico-rompe.html>. [Accedido: 15 de febrero de 2025].
- [29] HGH Infrared Systems, "Inteligencia Artificial GAIA con CYCLOPE," *HGH Infrared Systems*, [En línea]. Disponible en: <https://hgh-infrared.com/es/home-es/inteligencia-artificial-gaia-con-cyclope/>. [Accedido: 13 de febrero de 2025].
- [30] El Español, "Airbus y la UPM usan redes neuronales contra la pesca ilegal y las narcolanchas," *El Español*, [En línea]. Disponible en: https://www.elespanol.com/invertia/disruptores/grandes-actores/20200915/airbus-upm-redes-neuronales-pesca-ilegal-narcolanchas/520949468_0.html. [Accedido: 13 de febrero de 2025].
- [31] NTT Data, "Tecnología blockchain para impulsar puertos marítimos cognitivos," *NTT Data*, [En línea]. Disponible en: <https://es.nttdata.com/insights/case-studies/tecnologia-blockchain-para-impulsar-puertos-maritimos-cognitivos>. [Accedido: 13 de febrero de 2025].
- [32] Comisión Europea, "Pruebas de tecnología de detección para mejorar la seguridad marítima," *CORDIS*, [En línea]. Disponible en: <https://cordis.europa.eu/article/id/450551-testing-detection-technology-to-improve-maritime-safety/es>. [Accedido: 13 de febrero de 2025].
- [33] ESSS, "Optimización en ingeniería: técnicas y aplicaciones en diferentes industrias," *ESS*, [En línea]. Disponible en: <https://www.esss.com/es/blog/optimizacion-en-ingenieria/>. [Accedido: 2 de febrero de 2025].
- [34] StudySmarter, "Optimización matemática: ‘Algoritmos’, ‘Teorías’," *StudySmarter ES*, [En línea]. Disponible en: <https://www.studysmarter.es/resumenes/matematicas/matematicas-aplicadas/optimizacion-matematica/>. [Accedido: 2 de febrero de 2025].
- [35] L. de Sardi, "Optimización Multiobjetivo Evolutiva a través un método híbrido," *Redalyc*, [En línea]. Disponible en: <https://www.redalyc.org/pdf/2150/215024822002.pdf>. [Accedido: 2 de febrero de 2025].
- [36] M. Chaveinte García, "Caso de estudio de aplicación de algoritmos genéticos para la optimización de rutas marítimas," *UVaDOC*, 2023. [En línea]. Disponible en: <https://uvadoc.uva.es/handle/10324/62912>. [Accedido: 2 de febrero de 2025].
- [37] VA DE BARCOS, "Los buques de la US Navy encenderán el AIS en zonas de alto tráfico marítimo," *Va de Barcos*. [En línea]. Disponible en: <https://vadebarcos.net/2017/09/26/los-buques-de-la-us-navy-encenderan-el-ais-en-zonas-de-alto-traffic-marino/>. [Accedido: 2 de febrero de 2025].
- [38] SOM-INNPORT, "Solución innovadora en la vigilancia marítima," [En línea]. Disponible en: <https://www.sominnport.cat/es/solucion-innovadora-en-la-vigilancia-maritima/>. [Accedido: 2 de febrero de 2025].

- [39] B. S. López, "Método Simplex en Programación Lineal e Investigación de Operaciones," *Ingeniería Industrial Online*, [En línea]. Disponible en: <https://ingenieriaindustrialonline.com/investigacion-de-operaciones/metodo-simplex/>. [Accedido: 30 de enero de 2025].
- [40] A. Mendez, "Método Simplex Paso a Paso: Ejemplos de Maximizar y Minimizar," *Plan de Mejora*. [En línea]. Disponible en: <https://www.plandemejora.com/metodo-simplex-paso-a-paso-ejemplos-maximizar-minimizar/>. [Accedido: 30 de enero de 2025].
- [41] B. S. López, "Método Simplex en Programación Lineal e Investigación de Operaciones," *Ingeniería Industrial Online*. [En línea]. Disponible en: <https://ingenieriaindustrialonline.com/investigacion-de-operaciones/metodo-simplex/>. [Accedido: 30 de enero de 2025].
- [42] W. Palacios Alvarado, A. Caicedo-Rolón y N. Calixto, "*Optimización lineal, mediante el método SIMPLEX*". Bogotá: Editorial Creser S.A.S., 2022.
- [43] MathWorks, "fmincon," [En línea]. Disponible en: <https://es.mathworks.com/help/optim/ug/fmincon.html>. [Accedido: 13 de febrero de 2025].
- [44] M. O. Fernández, "*Algoritmos de búsqueda heurística en tiempo real. Aplicación a la navegación en los juegos de vídeo*," Trabajo de Cátedra, Análisis y Diseño de Algoritmos II, Universidad Nacional del Centro de la Provincia de Buenos Aires, Argentina, 2006.
- [45] F. Herrera, "Introducción a los Algoritmos Metaheurísticos," Universidad de Granada, 2-febrero-2025. [En línea]. Disponible en: <https://sci2s.ugr.es/sites/default/files/files/Teaching/OtherPostGraduateCourses/Metaheuristics/Int-Metaheuristics-CAEPIA-2009.pdf>. [Accedido: 2 de febrero de 2025].
- [46] L. E. de Sardi, L. F. Osuna y L. J. Valencia, "Comparación de métodos metaheurísticos de optimización: recocido simulado, algoritmos genéticos y búsqueda del cuco," *Revista Técnica de la Facultad de Ingeniería*, vol. 37, n.º 2, pp. 1-8, 2014. [En línea]. Disponible en: https://ve.scielo.org/scielo.php?pid=S1316-48212014000200001&script=sci_arttext. [Accedido: 2 de febrero de 2025].
- [47] GAMS, "Modelado de vanguardia,". [En línea]. Disponible en: <https://www.gams.com/>. [Accedido: 1 de febrero de 2025].
- [48] ScienceDirect, "General Algebraic Modeling System,". [En línea]. Disponible en: <https://www.sciencedirect.com/topics/engineering/general-algebraic-modeling-system>. [Accedido: 1 de febrero de 2025].
- [49] GAMS, "Programa académico GAMS,". [En línea]. Disponible en: <https://gams.com/academics/>. [Accedido: 1 de febrero de 2025].
- [50] LINDO Systems, "LINGO y modelado de optimización,". [En línea]. Disponible en: <https://www.lindo.com/index.php/products/lingo-and-optimization-modeling>. [Accedido: 01 de febrero de 2025].
- [51] LINDO Systems, "LINGO and optimization modeling,". [En línea]. Disponible en: <https://www.lindo.com/index.php/products/lingo-and-optimization-modeling>. [Accedido: 01 de febrero de 2025].
- [52] MATLAB Helper, "Creating A GUI In MATLAB,". [En línea]. Disponible en: <https://matlabhelper.com/blog/matlab/creating-a-gui-in-matlab/>. [Accedido: 02 de febrero de 2025].
- [53] M. H. Vico, "Nuevas tecnologías en los equipos de seguridad marítima," Euroinnova International Online Education. [En línea]. Disponible en: <https://www.euroinnova.com/profesiones-y-oficios/articulos/equipos-de-seguridad-maritima>. [Accedido: 29 de enero de 2025].

- [54] Exponav, "Entrega del FUROR, el BAM-6, buque de acción marítima construido en Ferrol", Exponav, 2019. [En línea]. Disponible en: <https://exponav.org/blog/construccion-naval/entrega-del-furor-el-bam-6-buque-de-accion-maritima-construido-en-ferrol/>. [Accedido: 12 de marzo de 2025]

ANEXO I: IMPLICACIONES SOCIALES, Y/O ECONÓMICAS, Y/O AMBIENTALES

A lo largo de este trabajo, no solo se ha desarrollado un modelo matemático de optimización para la vigilancia marítima, sino que también se ha analizado cómo su implementación influye en diversos ámbitos estratégicos. La optimización del patrullaje marítimo no solo permite mejorar la eficiencia operativa, sino que también tiene repercusiones significativas en la seguridad, la gestión de recursos y la sostenibilidad ambiental.

Desde una perspectiva social, la optimización de la vigilancia marítima impacta directamente en la seguridad de las aguas nacionales y en la lucha contra actividades ilícitas como el narcotráfico, la inmigración irregular y la pesca ilegal. Una mejor distribución de los recursos permite aumentar la capacidad de detección y respuesta ante amenazas, garantizando la protección de los ciudadanos y la integridad de las fronteras marítimas. Además, esta perspectiva contribuye a la mejora de las operaciones de búsqueda y rescate (SAR), permitiendo una intervención más rápida en situaciones de emergencia y aumentando la probabilidad de salvar vidas. La optimización del patrullaje también beneficia a las comunidades costeras, ya que reduce el impacto de actividades ilegales en sectores clave como la pesca y el turismo, asegurando así la estabilidad económica y social en las regiones marítimas. Estos aspectos se reflejan en la sección 3.1 de la memoria, donde se expone el planteamiento del problema y su impacto en la seguridad marítima.

Desde el punto de vista económico, la optimización del patrullaje marítimo conlleva una reducción significativa en los costos operativos al minimizar el consumo de combustible y el desgaste de los buques, lo que se traduce en menores gastos de mantenimiento y logística. Un patrullaje más eficiente permite además una mejor distribución de los efectivos, reduciendo la carga de trabajo y la fatiga de las tripulaciones, lo que a su vez mejora el rendimiento operativo. Asimismo, la estabilidad y seguridad en las aguas nacionales fortalecen sectores estratégicos como el comercio y el transporte marítimo, garantizando la fluidez de las rutas comerciales y la reducción de riesgos logísticos. En la sección 4 de la memoria se analizan los beneficios económicos derivados de la implementación del modelo de optimización, evidenciando cómo una asignación más eficiente de los recursos genera ahorros operativos y optimiza la capacidad de vigilancia marítima.

Desde un punto de vista ambiental, la optimización de los recursos de patrullaje reduce la huella de carbono de las operaciones de vigilancia al minimizar los desplazamientos innecesarios y, por ende, el consumo de combustibles fósiles. Además, permite mejorar la detección de actividades que afectan negativamente al ecosistema marino, como la pesca ilegal, el vertido de residuos contaminantes y la explotación descontrolada de los recursos oceánicos. Un patrullaje más eficiente también facilita el control y la preservación de zonas protegidas, evitando una sobreexposición innecesaria en áreas ecológicamente sensibles. Asimismo, la optimización de rutas de vigilancia fomenta el uso de tecnologías más sostenibles, como la integración de drones para la supervisión aérea y sistemas de inteligencia artificial que pueden reducir la necesidad de despliegues constantes de buques con alto consumo energético. Estos aspectos están desarrollados en la sección 3.5 de la memoria, donde se discuten las posibles mejoras del modelo y su impacto en la sostenibilidad ambiental.

ANEXO II: REFLEXIONES ÉTICAS Y SOCIALES

La optimización de los procedimientos de vigilancia marítima no solo supone un avance en la eficiencia operativa, sino que también plantea cuestiones éticas y sociales relevantes, relacionadas con la seguridad, la privacidad, la equidad en la distribución de recursos y el impacto en la sociedad.

Desde una perspectiva ética, la vigilancia marítima es una actividad que debe garantizar la seguridad y la legalidad en las aguas jurisdiccionales sin vulnerar derechos fundamentales. Uno de los principales dilemas en este ámbito es la delimitación entre seguridad y privacidad. La implementación de sistemas optimizados de patrullaje implica una monitorización más eficiente de embarcaciones y actividades en el mar, lo que puede generar debate sobre el derecho a la privacidad de los navegantes y el uso de los datos obtenidos en estas operaciones. Es fundamental que cualquier mejora en la vigilancia marítima respete las normativas internacionales en materia de protección de datos y derechos humanos, asegurando que las operaciones se realicen dentro de un marco legal que priorice la proporcionalidad y la necesidad en la recopilación de información.

Otro aspecto ético a considerar es la distribución equitativa de los recursos de vigilancia. La optimización del patrullaje busca maximizar la eficiencia bajo condiciones de recursos limitados, pero es necesario garantizar que las zonas más vulnerables o con mayor incidencia de actividades ilícitas no queden desatendidas por decisiones basadas únicamente en criterios matemáticos. En este sentido, el modelo de optimización desarrollado en este trabajo incorpora factores de riesgo y criticidad que permiten asignar los recursos de manera justa y acorde a la realidad operativa, evitando situaciones en las que ciertos sectores marítimos queden expuestos a amenazas por una mala asignación de vigilancia.

Desde un punto de vista social, la mejora en la vigilancia marítima tiene un impacto positivo en la lucha contra el crimen organizado y la protección de la vida humana, pero también puede generar tensiones con ciertos colectivos. Por ejemplo, las operaciones de patrullaje optimizadas pueden afectar a comunidades pesqueras o a actividades comerciales marítimas que dependen de la libre circulación. Es crucial que cualquier proceso de optimización tenga en cuenta estos efectos colaterales y busque un equilibrio entre la seguridad y la viabilidad de las actividades económicas que dependen del entorno marítimo.

Asimismo, la vigilancia marítima juega un papel fundamental en la gestión de flujos migratorios y la protección de derechos humanos. En regiones como el Estrecho de Gibraltar, la optimización del patrullaje puede contribuir a una mayor capacidad de respuesta en operaciones de rescate, evitando tragedias humanas asociadas a la migración irregular. Sin embargo, también se debe evitar que estos avances sean utilizados con fines exclusivamente disuasorios que vulneren el derecho al asilo o dificulten el acceso a rutas seguras para personas en situación de emergencia. La ética en la vigilancia marítima exige que las decisiones operativas se alineen con los principios de humanidad, proporcionalidad y respeto a los derechos fundamentales.

En términos de responsabilidad profesional, este trabajo refleja la capacidad del ingeniero para aplicar conocimientos técnicos con conciencia ética y social, asegurando que las soluciones desarrolladas no solo sean eficientes, sino también justas y sostenibles. La optimización de recursos en defensa y seguridad debe ir siempre acompañada de un análisis riguroso de sus consecuencias y de un compromiso con los valores de integridad, equidad y respeto por el bienestar social.

ANEXO III: CÓDIGO FUENTE EN MATLAB

```
function interfaz_bienvenida()
% Crear la figura principal
fig = figure('Name', 'Bienvenida', 'NumberTitle', 'off', ...
    'Position', [500, 300, 500, 300], 'MenuBar', 'none', ...
    'Resize', 'off', 'Color', [1, 1, 1]);

% Agregar una imagen (asegúrate de tener 'imagen.jpg' en el directorio de
trabajo)
ax = axes('Parent', fig, 'Units', 'pixels', 'Position', [100, 60, 300, 130]);
img = imread('imagen.jpg'); % Carga la imagen
imshow(img, 'Parent', ax);
axis off; % Ocultar ejes

% Agregar un texto de bienvenida
uicontrol('Style', 'text', 'String', 'Optimización de la vigilancia marítima',
...
    'FontSize', 16, 'FontWeight', 'bold', ...
    'Position', [50, 220, 400, 50], 'BackgroundColor', [1, 1, 1]);

% Agregar un texto con la información del autor
uicontrol('Style', 'text', 'String', 'CUD ENM 2025 - AF García Alonso', ...
    'FontSize', 12, 'FontWeight', 'bold', ...
    'Position', [50, 180, 400, 30], 'BackgroundColor', [1, 1, 1]);

% Agregar un botón para comenzar
uicontrol('Style', 'pushbutton', 'String', 'Comenzar', ...
    'FontSize', 12, 'FontWeight', 'bold', ...
    'Position', [200, 20, 100, 40], ...
    'Callback', @comenzar_codigo);
end

function comenzar_codigo(~, ~)
close(gcf); % Cierra la interfaz de bienvenida sin saltos

% Ejecutar la interfaz meteorológica en paralelo
t1 = parfeval(@interfaz_meteorologica, 0);
t2 = parfeval(@interfaz_capacidad, 0);

function interfaz_capacidad()
% Crear la figura principal
fig = figure('Name', 'Capacidad Operativa del Barco', 'NumberTitle', 'off', ...
    'Position', [900, 400, 400, 250], 'MenuBar', 'none', 'Resize', 'off');

% Descripciones de los niveles de capacidad
descripciones = { ...
    'Muy Capaz (menos tiempo patrullaje)', ...
    'Capacidad Normal', ...
    'Menos Capaz (más tiempo patrullaje)' ...
};

% Valores de capacidad operativa correspondientes
valores_CA = [0.5, 1.0, 1.5];

% Etiqueta del menú desplegable
uicontrol('Style', 'text', 'Position', [50, 200, 300, 30], ...
    'String', 'Capacidad operativa del barco:', 'FontSize', 12,
    'HorizontalAlignment', 'center');

% Menú desplegable para seleccionar la capacidad operativa
```

```

popup = uicontrol('Style', 'popupmenu', 'Position', [50, 150, 300, 50], ...
    'String', descripciones, 'FontSize', 12, 'Callback',
@actualizar_capacidad);

% Área de texto para mostrar la capacidad seleccionada
capacidad_texto = uicontrol('Style', 'text', 'Position', [50, 90, 300, 50], ...
    'String', '', 'FontSize', 12, 'HorizontalAlignment', 'center', ...
    'BackgroundColor', [0.9, 0.9, 0.9]);

% Botón para cerrar la interfaz
uicontrol('Style', 'pushbutton', 'Position', [150, 30, 100, 40], ...
    'String', 'Cerrar', 'FontSize', 12, 'Callback', @(src, event) close(fig));

% Función para actualizar la capacidad operativa seleccionada
function actualizar_capacidad(src, ~)
    indice = get(src, 'Value');
    set(capacidad_texto, 'String', sprintf('Capacidad Operativa: %.1f',
valores_CA(indice)));
end
end

function interfaz_meteorologica()
    % Crear la figura principal desplazada más a la izquierda
    fig = figure('Name', 'Condiciones Meteorológicas', 'NumberTitle', 'off', ...
        'Position', [150, 400, 400, 250], 'MenuBar', 'none', 'Resize', 'off');

    % Condiciones meteorológicas y sus factores
    descripciones = { ...
        'Tormenta severa, visibilidad reducida, oleaje alto', ...
        'Lluvias intensas, fuertes vientos, mar agitado', ...
        'Lluvias moderadas, visibilidad afectada, viento fuerte', ...
        'Lluvias ligeras, brisa moderada, mar movido', ...
        'Condiciones normales, cielo despejado o nublado parcial', ...
        'Cielo despejado, viento leve, mar tranquilo', ...
        'Buen clima, visibilidad óptima, condiciones ideales', ...
        'Excelente clima, sin vientos significativos, mar en calma', ...
        'Condiciones perfectas, cielo claro, mar completamente tranquilo' ...
    };
    factores = [0.8, 0.85, 0.9, 0.95, 1.0, 1.05, 1.1, 1.15, 1.2];

    % Etiqueta del menú desplegable
    uicontrol('Style', 'text', 'Position', [50, 180, 300, 30], ...
        'String', 'Seleccione la condición meteorológica:', 'FontSize', 12, ...
        'HorizontalAlignment', 'center');

    % Menú desplegable para seleccionar la condición meteorológica
    popup = uicontrol('Style', 'popupmenu', 'Position', [100, 150, 200, 30],
'String', descripciones, 'FontSize', 12, 'Callback', @actualizar_factor);
    % Área de texto para mostrar el factor meteorológico
    factor_texto = uicontrol('Style', 'text', 'Position', [50, 80, 300, 50], ...
        'String', '', 'FontSize', 12, 'HorizontalAlignment', 'center', ...
        'BackgroundColor', [0.9, 0.9, 0.9]);

    % Botón para cerrar la interfaz
    uicontrol('Style', 'pushbutton', 'Position', [150, 20, 100, 40], ...
        'String', 'Cerrar', 'FontSize', 12, 'Callback', @(src, event) close(fig));

    % Función para actualizar el factor meteorológico según la selección
    function actualizar_factor(src, ~)
        indice = get(src, 'Value');
        set(factor_texto, 'String', sprintf('Factor Meteorológico: %.2f',
factores(indice)));
    end
end

```

```
end
end

% Esperar a que la interfaz meteorológica termine antes de continuar
wait(tl);

% Proceder con la optimización del patrullaje
optimizacion_patrullaje();
end

function optimizacion_patrullaje()
while true

% -----
% SOLICITAR NOMBRE PARA LA EJECUCIÓN Y CREAR CARPETA
% -----
nombre_ejecucion = inputdlg('Ingrese un nombre para la ejecución:', ...
    'Nombre de la Ejecución', [1 50], {'Ejemplo_01'});

if isempty(nombre_ejecucion)
    error('Debe ingresar un nombre válido para la ejecución.');
```

end

```
nombre_carpeta = fullfile(pwd, nombre_ejecucion{1});
if ~exist(nombre_carpeta, 'dir')
    mkdir(nombre_carpeta);
end

disp(['Se guardarán los archivos en la carpeta: ', nombre_carpeta]);

% -----
% SOLICITUD DE DATOS AL USUARIO
% -----
datos = interfaz(nombre_carpeta); % Llamada a la interfaz gráfica

% Validación del tiempo mínimo requerido
tiempo_minimo_total = datos.k * sum(datos.S);
if tiempo_minimo_total > datos.Tmax
    mensaje_error = sprintf(['El tiempo máximo ingresado (%.2f) es
insuficiente.\n', ...
        'El tiempo mínimo requerido es: %.2f\n', ...
        'Por favor, introduzca un valor mayor para Tmax.'], ...
        datos.Tmax, tiempo_minimo_total);

    h = errorDlg(mensaje_error, 'Error', 'modal');
    uiwait(h); % Espera a que el usuario cierre el mensaje
    continue; % Vuelve a pedir los datos
end

% -----
% VALIDACIÓN DEL NÚMERO DE REGIONES
% -----
n = length(datos.PA);

% -----
% FUNCIÓN DE OPTIMIZACIÓN
% -----
[x, fval, exitflag, output, tiempo_ejecucion] = optimizar_patrullaje(datos, n);

% Verificación del tiempo total asignado con factor meteorológico
tiempos_minimos = datos.k * datos.S;
tiempos_totales = (tiempos_minimos(:) + x(:)) .* datos.factor_meteorologico';
```

```
% Verificación del tiempo total asignado
tiempos_minimos = datos.k * datos.S;
tiempos_totales = tiempos_minimos(:) + x(:);
if sum(tiempos_totales) > datos.Tmax
    h = errordlg('El tiempo total asignado supera el tiempo máximo permitido.
Ajuste los parámetros.', 'Error', 'modal');
    uiwait(h); % Espera a que el usuario cierre el mensaje
    continue; % Volver a pedir los datos si se excede el límite
end

% -----
% COORDENADAS FIJAS DE LAS REGIONES
% -----
regiones = {
[36.621, 36.309, 36.043, 36.056; -6.706, -6.309, -6.301, -6.754]; % Región 1
[36.264, 36.098, 35.94, 35.996; -6.241, -5.886, -5.883, -6.255]; % Región 2
[36.089, 36.004, 35.96, 35.933; -5.862, -5.562, -5.542, -5.843]; % Región 3
[36.011, 36.248, 36.047, 35.972; -5.51, -5.124, -4.964, -5.502]; % Región 4
[36.333, 36.531, 36.204, 36.049; -5.137, -4.523, -4.312, -4.918]; % Región 5
[36.450, 36.700, 36.550, 36.400; -4.500, -4.200, -4.100, -4.300]; % Región 6
[36.500, 36.750, 36.600, 36.450; -4.600, -4.300, -4.200, -4.400]; % Región 7
[36.550, 36.800, 36.650, 36.500; -4.700, -4.400, -4.300, -4.500] % Región 8
};

% -----
% REPRESENTACIÓN DE ÁREAS EN MAPA REAL
% -----
fig_map = figure;
fig_map.Position(1) = fig_map.Position(1) + 300; % Mueve la interfaz del mapa 200
píxeles a la izquierda

hold on;
worldmap([35.2 36.9], [-7 -4.1]);
load coastlines;
geoshow(coastlat, coastlon, 'DisplayType', 'line', 'Color', 'k');

colormap('turbo');
cmap = colormap;
nColors = size(cmap, 1);
colores = round((tiempos_totales / max(tiempos_totales)) * (nColors - 1)) + 1;

for i = 1:n
    latitudes = regiones{i}(1, :);
    longitudes = regiones{i}(2, :);
    patchm(latitudes, longitudes, 'FaceColor', cmap(colores(i), :), ...
        'FaceAlpha', 0.7, 'EdgeColor', 'k', 'LineWidth', 1);

    lat_centro = mean(latitudes);
    lon_centro = mean(longitudes);

    % Calcular el tiempo total (mínimo + extra) para la región i
    total_time = datos.k * datos.S(i) + x(i);
    textm(lat_centro, lon_centro, sprintf('%s\n%.2f', datos.nombres_S{i}, total_time),
        'Color', 'k', 'FontSize', 8, 'HorizontalAlignment', 'center');
end

colorbar;
title('\bf Distribución del Patrullaje en Áreas', 'FontSize', 14);
grid on;
hold off;
```

```
saveas(gcf, fullfile(nombre_carpeta, 'Mapa_Patrullaje.png'));

% -----
% CALCULAR Y MOSTRAR RESULTADOS
% -----
mostrar_resultados(datos, x, n, nombre_carpeta);

% -----
% EXPORTAR RESULTADOS A EXCEL
% -----
% Verificar tamaños antes de crear la tabla
num_regiones = length(datos.nombres_S);

if length(tiempos_minimos) ~= num_regiones || length(x) ~= num_regiones ||
length(tiempos_totales) ~= num_regiones
error("Los vectores de datos no tienen el mismo tamaño. Revisa las
asignaciones.");
end

% Crear tabla con los datos de patrullaje
T = table(datos.nombres_S(:,), tiempos_minimos(:), x(:), tiempos_totales(:), ...
    'VariableNames', {'Region', 'Tiempo_Minimo', 'Tiempo_Extra',
'Tiempo_Total'});

% Generar nombre único para el archivo usando fecha y hora
timestamp = datestr(now, 'yyyymmdd_HHMMSS'); % Formato
YYYYMMDD_HHMMSS
filename = fullfile(nombre_carpeta, 'Resultados_Patrullaje.xlsx');

% Guardar la tabla de tiempos en la hoja "Resultados"
writetable(T, filename, 'Sheet', 'Resultados');
disp(['Resultados guardados en: ', filename]);

% Crear una tabla con información adicional (tiempo de ejecución y número de
iteraciones)
info_adicional = table(tiempo_ejecucion, output.iterations, ...
    'VariableNames', {'Tiempo_Ejecucion_segundos', 'Numero_Iteraciones'});

% Crear una tabla con los datos introducidos por el usuario
datos_usuario = table( ...
datos.PA, datos.PB, datos.PC, datos.WA, datos.WB, datos.WC, ...
datos.CA, datos.CB, datos.CC, datos.S, datos.nombres_S, ...
datos.factor_meteorologico, datos.k, datos.Tmax, ...
    'VariableNames', {'PA', 'PB', 'PC', 'WA', 'WB', 'WC', 'CA', 'CB', 'CC', ...
    'Tamaño_Zonas', 'Nombres_Zonas', 'Factor_Meteorologico',
'k', 'Tmax'});

% Guardar los datos introducidos por el usuario en otra hoja del Excel
writetable(datos_usuario, filename, 'Sheet', 'Datos_Usuario');

% Guardar la información adicional en otra hoja de Excel
writetable(info_adicional, filename, 'Sheet', 'Info_Algoritmo');

% Mostrar mensaje de confirmación
disp(['Resultados guardados en el archivo: ', filename]);

break;
end
end
```

```
% -----
% FUNCIÓN PARA OPTIMIZAR EL PATRULLAJE
% -----
function [x, fval, exitflag, output, tiempo_ejecucion] =
optimizar_patrullaje(datos, n)

    Aeq = ones(1, n);
    beq = datos.Tmax - datos.k * sum(datos.S);
    lb = zeros(n, 1);
    ub = [];
    x0 = ones(n, 1) * 20;

    % Medición de memoria antes de la ejecución
    [memory_info_before] = memory;
    disp(['Memoria usada antes de la ejecución: ',
    num2str(memory_info_before.MemUsedMATLAB / 1e6), ' MB']);

    tic;

    [x, fval, exitflag, output] = fmincon(@(x) fun(n, x, datos), x0, [], [], Aeq,
    beq, lb, ub);

    % Medición de memoria después de la ejecución
    [memory_info_after] = memory;
    disp(['Memoria usada después de la ejecución: ',
    num2str(memory_info_after.MemUsedMATLAB / 1e6), ' MB']);

    tiempo_ejecucion=toc;
    fprintf('Tiempo de ejecución del algoritmo: %.4f segundos\n', tiempo_ejecucion);

    % Mostrar en consola el número de iteraciones realizadas
    disp(['Número de iteraciones ejecutadas: ', num2str(output.iterations)]);

    disp("El valor de la función objetivo (fval) es:");
    disp(fval);

    disp("El valor de exitflag es:");
    disp(exitflag);
end

% -----
% FUNCIÓN PARA MOSTRAR RESULTADOS
% -----
function mostrar_resultados(datos, x, n, nombre_carpeta)
    pp = datos.WA * datos.CA * datos.PA + datos.WB * datos.CB * datos.PB +
    datos.WC * datos.CC * datos.PC;

    % Cálculo de tiempos mínimos
    tiempos_minimos = datos.k * datos.S(:);

    % Cálculo del tiempo total (mínimo + extra)
    tiempos_totales = tiempos_minimos(:) + x(:); % Asegurar que ambos sean vectores
    columna

    % Depuración: Verificar dimensiones y valores
    disp("Verificación de tamaños:");
    disp("Tamaño de datos.S:");
    disp(size(datos.S(:)));
    disp("Tamaño de x:");
    disp(size(x));
    disp("Tamaño de tiempos_totales:");
    disp(size(tiempos_totales));
```

```
% -----
% MOSTRAR RESULTADOS EN CONSOLA
% -----

disp(repmat('-', 1, 20)); % Línea separadora
disp("Tiempos MÍNIMOS asignados por región:");
for i = 1:n
    fprintf("Región: %s - Tiempo Mínimo: %.2f\n", datos.nombres_S{i},
    tiempos_minimos(i));
end

disp(repmat('-', 1, 20)); % Línea separadora
disp("El vector de tiempos EXTRA en las regiones es:");
for i = 1:n
    fprintf("Región: %s - Tiempo Extra: %.2f\n", datos.nombres_S{i}, x(i));
end

disp(repmat('-', 1, 20)); % Línea separadora
disp("Tiempo TOTAL asignado por región:");
for i = 1:n
    fprintf("Región: %s - Tiempo Total: %.2f\n", datos.nombres_S{i},
    tiempos_totales(i));
end
disp(repmat('-', 1, 20)); % Línea final

disp("El valor de la función ponderada es:");
disp((3 * ones(n,1) - pp')' * x);

disp("El valor de pp es:");
disp(pp);

disp("Suma de tiempos mínimos:");
disp(sum((datos.k)*(datos.S)));

disp("Suma de tiempos extra");
disp(sum(x));

% -----
% GENERACIÓN DE LA GRÁFICA DE BARRAS
% -----

fig = figure;
fig.Position(1) = fig.Position(1) - 300; % Mueve la interfaz 200 píxeles a la
izquierda
b=bar(tiempos_totales, 'FaceColor', 'flat'); % Color azul oscuro
colormap('parula');

% Configurar etiquetas en el eje X con los nombres de las regiones
set(gca, 'XTick', 1:n, 'XTickLabel', datos.nombres_S);

%Colores gradientes según valores
b.CData=tiempos_totales;
colorbar;

% Etiquetas y título
xlabel('Regiones');
ylabel('Tiempo Total Asignado (horas)');
title('Distribución de Tiempos por Región');

% Agregar valores sobre las barras
for i = 1:n
    text(i, tiempos_totales(i) + 0.8, sprintf('%.2f', tiempos_totales(i)), ...
        'HorizontalAlignment', 'center', 'FontSize', 10, 'FontWeight', 'bold');
```

```

end
hold on;
% Obtener el ancho de las barras (por defecto es 0.8)
barWidth = b.BarWidth;
for i = 1:n
    % Calcular el centro y los extremos horizontales de cada barra
    xCenter = i;
    xLeft = xCenter - barWidth/2;
    xRight = xCenter + barWidth/2;
    % Dibujar una línea horizontal en el nivel del tiempo mínimo para la región i
    plot([xLeft, xRight], [tiempos_minimos(i), tiempos_minimos(i)], 'r-',
'LineWidth', 2);
end
hold off;

grid on;
drawnow; % Forzar actualización de la interfaz gráfica
saveas(gcf, fullfile(nombre_carpeta, 'Grafico_Barras.png'));
hold off;
end

% -----
% INTERFAZ GRÁFICA PARA INGRESAR DATOS
% -----

function datos = interfaz(nombre_carpeta)
    prompt = { ...
        '\bf Probabilidad PA (ej: 0.25,0.5,0.3,0.4,0.2):', ...
        '\bf Probabilidad PB (ej: 0.5,0.5,0.3,0.4,0.3):', ...
        '\bf Probabilidad PC (ej: 0.75,0.6,0.6,0.5,0.2):', ...
        '\bf Peso WA:', ...
        '\bf Peso WB:', ...
        '\bf Peso WC:', ...
        '\bf Capacidad CA:', ...
        '\bf Capacidad CB:', ...
        '\bf Capacidad CC:', ...
        '\bf Tamaño de zonas S (ej: 100,100,200,150,180):', ...
        '\bf Nombres de zonas (ej: A,B,C,D,E):', ...
        '\bf Factor Meteorológico (ej: 1.2,0.8,1.0,1.1,0.9):', ...
        '\bf Constante k:', ...
        '\bf Tiempo total Tmax:' ...
    };

    defaults = { ...
        '0.25,0.5,0.5,0.4,0.4', ...
        '0.5,0.5,0.3,0.4,0.3', ...
        '0.75,0.6,0.6,0.5,0.2', ...
        '1', ...
        '1.2', ...
        '0.5', ...
        '1.5', ...
        '1', ...
        '2', ...
        '140,100,70,80,150', ...
        'Z1,Z2,Z3,Z4,Z5', ...
        '1.15,0.85,1.0,1.1,0.9', ...
        '0.09', ...
        '150' ...
    };
};

```

```
options.WindowStyle = 'normal';
options.Interpreter = 'tex';
options.FontSize = 14;

while true
    respuesta = inputdlg(prompt, "Ingrese Datos", [1 70], defaults, options);

    if isempty(respuesta) % Si el usuario cierra la ventana
        error('Debe ingresar datos válidos para continuar.');
```

end

```
    try
        datos.PA = str2num(respuesta{1});
        datos.PB = str2num(respuesta{2});
        datos.PC = str2num(respuesta{3});
        datos.WA = str2double(respuesta{4});
        datos.WB = str2double(respuesta{5});
        datos.WC = str2double(respuesta{6});
        datos.CA = str2double(respuesta{7});
        datos.CB = str2double(respuesta{8});
        datos.CC = str2double(respuesta{9});
        datos.S = str2num(respuesta{10});
        datos.nombres_S = strsplit(strtrim(respuesta{11}), ',');
        datos.factor_meteorologico = str2num(respuesta{12});
        datos.k = str2double(respuesta{13});
        datos.Tmax = str2double(respuesta{14});
        % Verificar que las entradas numéricas sean válidas
        if any(isnan([datos.WA, datos.WB, datos.WC, datos.CA, datos.CB,
datos.CC, datos.k, datos.Tmax])) || ...
            any(isempty([datos.PA, datos.PB, datos.PC, datos.S,
datos.factor_meteorologico]))
            error('Algunos valores ingresados no son numéricos o están
vacíos.');
```

end

```
        break; % Si todo está correcto, salir del bucle

    catch
        h = errordlg('Por favor, ingrese solo valores numéricos en los campos
correspondientes.', 'Error de entrada', 'modal');
        uiwait(h);
    end
end
end

% -----
% FUNCIÓN OBJETIVO PARA `fmincon`
% -----
function f = fun(n, x, datos)
    pp = datos.WA * datos.CA * datos.PA + datos.WB * datos.CB * datos.PB +
datos.WC * datos.CC * datos.PC;
    for i = 1:n
        w(i) = (x(i) * (3 - (datos.WA * datos.CA * datos.PA(i) + datos.WB *
datos.CB * datos.PB(i) + datos.WC * datos.CC * datos.PC(i))) *
datos.factor_meteorologico(i))^2;
    end
    f = sum(w);
end
```

ANEXO IV: DATOS UTILIZADOS PARA LA SIMULACIÓN

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.6	0.6	0.6	1	0.8	0.8	2	1	1	100	1
Z2	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z3	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z4	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z5	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1

Tabla 1 Datos entrada Análisis 1 – Ensayo 1

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z2	0.6	0.6	0.6	1	0.8	0.8	2	1	1	100	1
Z3	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z4	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z5	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1

Tabla 2 Datos entrada Análisis 1 – Ensayo 2

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z2	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z3	0.6	0.6	0.6	1	0.8	0.8	2	1	1	100	1
Z4	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z5	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1

Tabla 3: Datos entrada Análisis 1 – Ensayo 3

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z2	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z3	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z4	0.6	0.6	0.6	1	0.8	0.8	2	1	1	100	1
Z5	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1

Tabla 4 Datos entrada Análisis 1 – Ensayo 4

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z2	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z3	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z4	0.5	0.5	0.5	1	0.8	0.8	2	1	1	100	1
Z5	0.6	0.6	0.6	1	0.8	0.8	2	1	1	100	1

Tabla 5 Datos entrada Análisis 1 – Ensayo 5

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.6	0.6	0.6	1.3	1	1	1	1	1	100	1
Z2	0.5	0.5	0.5	1.3	1	1	1	1	1	100	1
Z3	0.5	0.5	0.5	1.3	1	1	1	1	1	100	1
Z4	0.5	0.5	0.5	1.3	1	1	1	1	1	100	1
Z5	0.5	0.5	0.5	1.3	1	1	1	1	1	100	1

Tabla 6 Datos entrada Análisis 1 – Ensayo 6

OPTIMIZACIÓN DE UN PROCEDIMIENTO DE VIGILANCIA
MARÍTIMA EN UN ESCENARIO DE RECURSOS LIMITADOS

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	1	0.5	0.5	1	1	1	0.4	1	1.4	100	1
Z2	0.5	1	0.5	1	1	1	0.4	1	1.4	100	1
Z3	0.5	0.5	1	1	1	1	0.4	1	1.4	100	1
Z4	0.5	0.5	0.5	1	1	1	0.4	1	1.4	100	1
Z5	0.5	0.5	0.5	1	1	1	0.4	1	1.4	100	1

Tabla 7 Datos entrada Análisis 1 – Ensayo 7

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.5	0.5	0.5	1	1	1	1	1	1	100	0.85
Z2	0.5	0.5	0.5	1	1	1	1	1	1	100	0.95
Z3	0.5	0.5	0.5	1	1	1	1	1	1	100	1
Z4	0.5	0.5	0.5	1	1	1	1	1	1	100	1
Z5	0.5	0.5	0.5	1	1	1	1	1	1	100	1.15

Tabla 8 Datos entrada Análisis 1 – Ensayo 8

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.5	0.5	0.5	1.2	1	0.8	0.8	1.2	1	140	1
Z2	0.5	0.5	1	1.2	1	0.8	0.8	1.2	1	100	1
Z3	1	0.5	0.5	1.2	1	0.8	0.8	1.2	1	70	1
Z4	0.5	1	0.5	1.2	1	0.8	0.8	1.2	1	80	1
Z5	0.5	0.5	0.5	1.2	1	0.8	0.8	1.2	1	150	1

Tabla 9 Datos entrada Análisis 2 – Ensayo 1

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.5	0.5	0.5	1.2	1	0.8	0.8	1.2	1	140	0.85
Z2	0.5	0.5	1	1.2	1	0.8	0.8	1.2	1	100	0.95
Z3	1	0.5	0.5	1.2	1	0.8	0.8	1.2	1	70	1
Z4	0.5	1	0.5	1.2	1	0.8	0.8	1.2	1	80	1.1
Z5	0.5	0.5	0.5	1.2	1	0.8	0.8	1.2	1	150	1.1

Tabla 10 Datos entrada Análisis 2 – Ensayo 2

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.25	0.5	0.75	1	1.2	0.5	1.5	1	2	100	1.15
Z2	0.5	0.5	0.6	1	1.2	0.5	1.5	1	2	100	0.85
Z3	0.5	0.3	0.6	1	1.2	0.5	1.5	1	2	100	1
Z4	0.4	0.4	0.5	1	1.2	0.5	1.5	1	2	100	1.1
Z5	0.4	0.3	0.2	1	1.2	0.5	1.5	1	2	500	0

Tabla 11 Datos entrada Análisis 3 – Ensayo 1

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.25	0.5	0.75	1	1.2	0.5	1.5	1	2	100	1.15
Z2	0.5	0.5	0.6	1	1.2	0.5	1.5	1	2	100	0.85
Z3	0.5	0.3	0.6	1	1.2	0.5	1.5	1	2	100	1
Z4	0.4	0.4	0.5	1	1.2	0.5	1.5	1	2	100	1.1
Z5	0.4	0.3	0.2	1	1.2	0.5	1.5	1	2	500	0

Tabla 12 Datos entrada Análisis 3 – Ensayo 2

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.5	0.5	0.5	1	1	1	1	1	1	140	0.5
Z2	0.5	0.5	0.5	1	1	1	1	1	1	100	0.8
Z3	0.5	0.5	0.5	1	1	1	1	1	1	70	1
Z4	0.5	0.5	0.5	1	1	1	1	1	1	80	1.3
Z5	0.5	0.5	0.5	1	1	1	1	1	1	150	1.5

Tabla 13 Datos entrada Análisis 4

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.25	0.5	0.75	1	1.2	0.5	1.5	1	2	140	1.15
Z2	0.5	0.5	0.6	1	1.2	0.5	1.5	1	2	100	0.85
Z3	0.5	0.3	0.6	1	1.2	0.5	1.5	1	2	70	1
Z4	0.4	0.4	0.5	1	1.2	0.5	1.5	1	2	80	1.1
Z5	0	0	0	1	1.2	0.5	1.5	1	2	150	0

Tabla 14 Datos entrada Análisis 7 – Ensayo 1

Zona	PA	PB	PC	WA	WB	WC	CA	CB	CC	Tamaño	Factor meteorológico
Z1	0.25	0.5	0.75	1	1.2	0.5	1.5	1	2	140	1.15
Z2	0.5	0.5	0.6	1	1.2	0.5	1.5	1	2	100	0.85
Z3	0.5	0.3	0.6	1	1.2	0.5	1.5	1	2	70	1
Z4	0.4	0.4	0.5	1	1.2	0.5	1.5	1	2	80	1.1
Z5	0	0	0	1	1.2	0.5	1.5	1	2	150	0.9

Tabla 15 Datos entrada Análisis 7 – Ensayo 2

ANEXO V: DICCIONARIO DE ACRÓNIMOS

- AIS: *Automatic Identification System* (Sistema de Identificación automática)
- ALMART: Almirante de Acción Marítima
- AWS: Amazon Web Services
- BAM: Buque de Acción Marítima
- CMOM: Comandante del Mando Operativo Marítimo
- ENSM: Estrategia Nacional de Seguridad Marítima
- FAM: Fuerza de Acción Marítima
- GPS: *Global Positioning System* (Sistema de Geoposicionamiento Global)
- IA: Inteligencia Artificial
- IoT: *Internet of Things* (Internet de las cosas)
- MDA: *Maritime Domain Awareness* (Conocimiento del Entorno Marítimo)
- MSA: *Maritime Situational Awareness* (Conciencia Situacional Marítima)
- OMI: Organización Marítima Internacional
- RMP: *Recognized Maritime Picture*
- SMNT: Sistemas Marítimos No Tripulados
- STROG: Estrecho de Gibraltar
- USV: *Unmanned Surface Vehicle* (Vehículo de Superficie No Tripulado)
- UXV: *Unmanned [Any] Vehicle* (Vehículo No Tripulado (genérico))
- ZEE: Zona Económica Exclusiva