Trabajo Fin de Grado
Ingeniería Aeroespacial

Desarrollo de Metodología y Aplicación Informática para Cálculo de Actuaciones Certificadas en Despegue de Aviones de Transporte Civil

Autor: Juan José Rodríguez Martín-Arroyo
Tutor: Diego Jerónimo Morillo Galeote

Dpto. Ingeniería de la Construcción y Proyectos de Ingeniería
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Trabajo Fin de Grado
Ingeniería Aeroespacial

Desarrollo de Metodología y Aplicación Informática para Cálculo de Actuaciones Certificadas en Despegue de Aviones de Transporte Civil

Autor:
Juan José Rodríguez Martín-Arroyo

Tutor:
Diego Jerónimo Morillo Galeote
Profesor asociado

Dpto. Ingeniería de la Construcción y Proyectos de Ingeniería
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2021

Proyecto Fin de Carrera: Desarrollo de Metodología y Aplicación Informática para Cálculo de Actuaciones Certificadas en Despegue de Aviones de Transporte Civil

Autor: Juan José Rodríguez Martín-Arroyo

Tutor: Diego Jerónimo Morillo Galeote

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

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2021

El Secretario del Tribunal

Agradecimientos

Ha sido un duro camino el llegar hasta este punto, muchos momentos de agobio y de frustración, en los que piensas que nunca vas a conseguir superarlos. Sin embargo, al final se superan y la prueba de ello es que aquí estoy escribiendo estas líneas. Analizando mi paso por la carrera, creo que lo más importante que he aprendido es precisamente eso, por muy duro que sea algo, no hay que rendirse nunca y trabajar con determinación en ello.

Pero por supuesto, no podría haber superado esos duros momentos sin el apoyo de mis seres queridos. Quiero agradecer sobre todo a mi familia, mis padres y mi hermana, su incondicional apoyo a pesar de cualquier circunstancia.

Agradecer también a mis amigos, los de toda la vida y los que encontré en mis años en la carrera, a estar siempre ahí para cuando necesitaba desahogarme o despejar mi mente.

Por último, me gustaría agradecer a mi tutor, Diego, el darme la oportunidad de realizar este proyecto tan apasionante, el cual he disfrutado mucho.

Juan José Rodríguez Martín-Arroyo

Grado en Ingeniería Aeroespacial

Sevilla, 2021

Resumen

El despegue es de las maniobras más peligrosas y complejas que realiza un avión. Por ello, para realizarlo con la máxima seguridad posible existe una extensa cantidad de condiciones que el fabricante del avión debe asegurar que se cumplan, recogidas en las distintas normas de aviación que existen.

Surge entonces la idea de realizar una herramienta informática que, sin exigir una excesiva cantidad de datos y parámetros, y pensada para un uso ingenieril de análisis, sea capaz de dar una aproximación suficientemente correcta de las actuaciones en el despegue de un avión comercial, definidas por la norma.

Entonces, este proyecto consiste en la creación de dicha herramienta o aplicación. Para ello el proyecto se divide en diferentes capítulos, cada uno con un objetivo distinto, con el fin de organizar de forma coherente esta memoria.

Lo primero sería realizar un estudio de la normativa vigente aplicable a las actuaciones en despegue de aeronaves comerciales de tamaño medio grande, en este caso se tendrán en cuenta la normativa europea y la normativa americana.

Después, se pasaría a construir un modelo dinámico sencillo pero lo suficientemente completo como para que se aproxime a la realidad y nos ofrezca buenos resultados al resolverlo. El siguiente punto sería buscar una solución analítica de este modelo e implementar una solución numérica con ayuda de un programa informático de cálculo.

Una vez hemos logrado esto, comenzaremos a crear una aplicación cuya base sea el modelo que hemos creado anteriormente. Se buscará crear una interfaz sencilla y amigable para el usuario, y que ofrezca buenos resultados dentro de lo esperado para un modelo aproximado.

Finalmente, con nuestra aplicación ya terminada, probaremos su alcance y efectividad mediante dos ejemplos de aviones comerciales populares, un turbofán y un turbohélice, y compararemos nuestros resultados con valores reales de estos aviones. Acto seguido, a la vista de los resultados, emitiremos unas conclusiones finales sobre el proyecto.

Abstract

Take-off is one of the most dangerous and complex maneuvers that an aircraft can perform. For that reason, to be able to do it with the maximum security, there are an extensive quantity of conditions that the manufacturer must ensure to be fulfil, reunited in the different aircraft standards that exist.

The idea is to create an informatic app that, without demanding an excessive quantity of data or parameters, and designed for an engineering use, be able to give a good approach of the take-off performances of a commercial aircraft, defined in the standards.

So, this project consists in the creation of that app. For this, the project is divided into different chapters, each with a different objective, in order to organize this report in a coherent way.

First, we will carry out a study of the current regulations applicable to the take-off performances of medium-large commercial aircraft, in this case European regulations and American regulations will be taken into account.

Afterwards, we would go on to build a simple but complete dynamic model so that it approximates reality and offers us good results when solving it. The next point would be to find an analytical solution to this model and implement a numerical solution with the help of a computer calculation program.

Once we have achieved this, we will begin to create an application based on the model that we created previously. We will seek to create a simple and friendly interface for the user, and that offers good results, always considering that we are talking about an approximate model.

Finally, with our application complete, we will test its range and effectiveness using two popular commercial aircraft examples, a turbofan and a turboprop, and compare our results with actual values from these aircraft. Then, in view of the results, we will issue final conclusions about the project.

Índice

Índice de Figuras

Notación

δ_T: Posición palanca de gases en porcentaje

T_{SL}: Empuje a nivel del mar

γ: Coeficiente de dilatación adiabática

M: Mach de vuelo

ρ: Densidad

ρ_{SL}: Densidad a nivel del mar

P: Potencia

P_{SL}: Potencia a nivel del mar

p: Presión

p_{SL}: Presión a nivel del mar

T: Empuje

V: Velocidad de vuelo

η_P: Rendimiento propulsivo

C_L: Coeficiente de sustentación

C_D: Coeficiente de resistencia

D: Resistencia aerodinámica

L: Sustentación

AR: Alargamiento

b: Envergadura

S_{alar}: Superficie alar

Δ: Ángulo de flecha

α: Ángulo de ataque

α_{0L}: Ángulo de ataque para sustentación nula

C_{D0}: Coeficiente de resistencia sin sustentación

ΔC_{Dflap}: Incremento del coeficiente de resistencia debido a flaps

ΔC_{Dgear}: Incremento del coeficiente de resistencia debido al tren de aterrizaje

K: parámetro de resistencia inducida unitario

W: Fuerza peso

F_R: Fuerza de rozamiento con la pista

N: Fuerza normal con respecto a la pista

V_A: Velocidad aerodinámica

V_G: Velocidad con respecto al suelo

ϕ: Ángulo de la pendiente de la pista

α: Ángulo de ataque

γ_A: Ángulo de asiento de la velocidad aerodinámica

γ_G: Ángulo de asiento respecto al suelo de la velocidad respecto al suelo

θ: Ángulo de asiento del avión

ϵ: Ángulo de ataque del empuje

h: Altura del centro de gravedad del avión con respecto a la pista

x: coordenada horizontal del sistema de referencia centrado en el punto donde arranca el avión

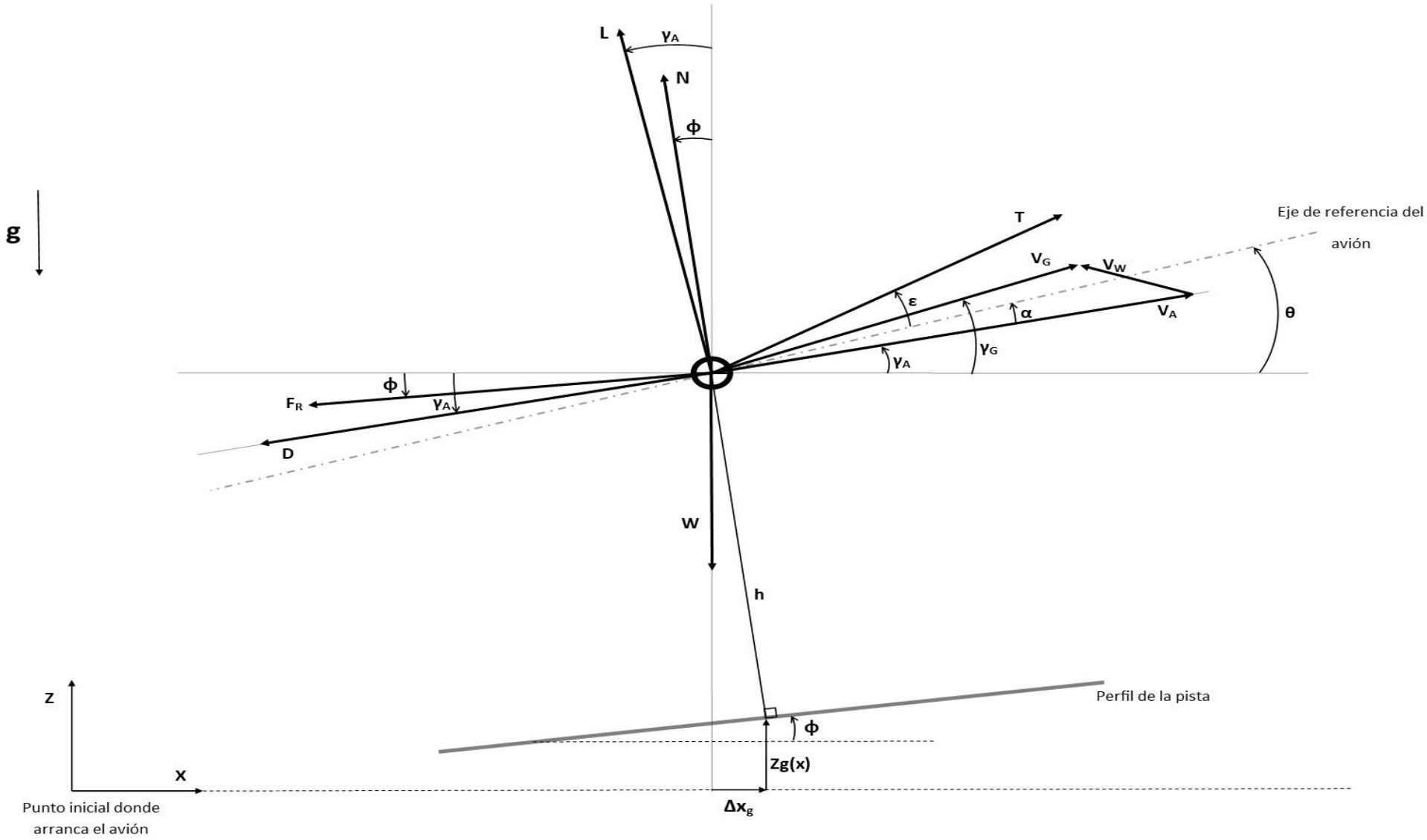
z: coordenada vertical del sistema de referencia centrado en el punto donde arranca el avión

$\mathrm{Z_g}(x)$: Altura de la pista con respecto al punto inicial, en función de x

$\Delta \mathrm{x_g}$: Diferencia entre la coordenada x con respecto a pista y la coordenada x horizontal

a: velocidad del sonido

Para mayor claridad y comprensión de estas definiciones, se adjunta una imagen en la siguiente página.


L
γ_A
N
ϕ
g
T
Eje de referencia del avión
V_G
V_W
ε
α
V_A
θ
γ_G
γ_A
ϕ
F_R
γ_A
D
W
h
Z
X
Perfil de la pista
ϕ
Zg(x)
Punto inicial donde arranca el avión
Δx_g

1. INTRODUCCIÓN

El objetivo de este trabajo, de manera resumida, es realizar una herramienta informática para el cálculo y análisis de las actuaciones en despegue de aviones comerciales de transporte, procurando que el resultado final sea un programa preciso, intuitivo y que no requiera de una cantidad excesiva de datos sobre la aeronave para poder hacer unas estimaciones de las actuaciones.

En la guía de contenidos de este trabajo, el tutor propone los siguientes puntos como objetivos:

1. Analizar la normativa aplicable (visión general y partes, partes aplicables, identificación e interpretación de requisitos de diseño, etc.), tanto europea (EASA) como americana (FAA), identificando las diferencias (si las hubiera). Justificar si es viable (sin gran esfuerzo adicional) considerar ambas normativas en el proyecto o, en otro caso, aplicar solo a EASA.
2. Crear un modelo de senda de despegue con los cuatro segmentos típicos habituales (primero, segundo, aceleración y final) que cumpla con la normativa (EASA y/o FAA), identificando claramente los segmentos y parámetros y limitaciones sujetos a normativa en cada uno de ellos.
3. Crear un modelo de dinámica de punto para el avión que tenga en cuenta los requisitos y limitaciones que establece la normativa. Determinar, analizar y clasificar las variables y parámetros que intervienen (ambientales, entrada, salida, control, parámetros). Los parámetros que intervienen en el problema se suponen conocidos en el contexto del proyecto (aerodinámicos, velocidades características, rodadura, motor, etc.).
4. Implementar los modelos (senda y dinámica) en una aplicación informática orientada a análisis de ingeniería (no a operadores), que permita de una forma amigable y eficaz introducir los datos y determinar, analizar, visualizar y extraer los resultados. Probar y poner a punto la aplicación comprobando que se obtienen resultados coherentes. Desarrollar un manual de usuario.
5. Contrastar los resultados de la aplicación desarrollada con datos disponibles de actuaciones para aviones actualmente en servicio certificados según la normativa (EASA y/o FAA). Seleccionar al menos un turbofán (ej. A320) y un turbohélice (ej. ATR72) típicos. En caso de no disponer de todos los parámetros necesarios (que es lo más probable), hacer una estimación de los mismos en base a los datos disponibles. Analizar y justificar posibles discrepancias y, en su caso, poner a punto la aplicación mediante parámetros de ajuste (ej. velocidad y técnica de rotación).

6. Seleccionar dos aviones representativos (un turbofán y un turbohélice) y hacer un análisis exhaustivo de las actuaciones para mostrar la capacidad y al alcance del método desarrollado (gráficas de actuaciones).
7. Conclusiones y juicio crítico del autor del proyecto sobre el trabajo realizado.

El segundo punto se acordó de reducirlo, de modo que el objetivo cubriese hasta los 35 pies de altura, y no los cuatro segmentos habituales de la senda de despegue.

Debido a la similitud en la temática del punto 5 y 6, se ha optado por fusionar ambos puntos en uno, de manera que a la vez que se hace un análisis de dos aviones representativos, estos análisis se contrastarán con datos disponibles de las actuaciones de ambos aviones, logrando con ello el objetivo de la misma forma.

Para crear dicha aplicación informática, haremos uso del programa informático Matlab, más en concreto su bloque de diseño de aplicaciones. Este bloque, combinado con el habitual bloque de programación nos permitirá crear una interfaz de usuario enlazada con los códigos que nos permitan calcular las actuaciones durante el despegue.

El proceso a seguir para lograr nuestro objetivo sería definir las actuaciones de despegue según la norma o normas más habituales en el ámbito aeronáutico, definir un modelo dinámico mediante el cual vamos a simular el despegue de un avión, implementar dicho modelo en el programa informático Matlab y crear a su vez una interfaz de usuario vinculada a dicho modelo que permita de forma efectiva calcular las actuaciones definidas anteriormente por la norma, y, por último, comprobar mediante ejemplos de aviones reales cuan efectivo es nuestro programa, cuáles son sus puntos fuertes y cuáles son sus puntos débiles.

En esta memoria, se hará referencia a menudo de comandos y secciones del programa Matlab. Se explicará su función la primera vez que se nombren, pero para más información y para mayor comodidad a la hora de leer y entender el documento, se puede consultar en la página oficial de Matlab cualquier duda, que vendrá indicada en las referencias. Por ejemplo, la página principal de Matlab es [1] y la página principal del bloque de diseño de aplicaciones es [2].

2. ANÁLISIS DE NORMATIVA APLICABLE

En este capítulo vamos a introducir las actuaciones en despegue definidas por la norma, para luego analizar la misma, ver qué condiciones o restricciones imponen sobre el despegue e incluir dichas restricciones en nuestro programa.

De esta forma seremos capaces no solo de analizar las actuaciones del avión, sino también ver si estamos cumpliendo la norma vigente.

Las normas que vamos a analizar en este trabajo van a ser la europea CS 25 Amendment 22 [3], desde el punto 25.105 hasta el punto 25.115, y la americana CFR Part 25 de 2013 [4], desde el punto 25.105 hasta el punto 25.115. A su vez, compararemos ambas normas, viendo si existen diferencias entre ellas o si por el contrario son similares, pudiéndose usar una u otra sin que exista variación en las restricciones. Se expondrán las conclusiones al final del capítulo. Además, también nos apoyaremos en el libro [5] para definir las actuaciones.

Es importante, antes de comenzar a analizar la normativa, aclarar que en este punto del trabajo se busca hacer una interpretación de la normativa, en vez de simplemente transcribirla literalmente. De esta forma, nos enfocaremos en las partes de la norma que más afectan al despegue y cómo esta norma limita o restringe las actuaciones del avión, que, al fin y al cabo, es lo que necesitamos tener claro para luego poder implementar dichas restricciones en nuestra aplicación. Además, se consigue de esta manera una lectura más clara y concisa sobre la norma, ya que tal y como está escrita la misma, puede resultar algo complejo y confuso de entender.

En resumen, se analizará la normativa vigente correspondiente al despegue y de este análisis se extraerá lo más importante, realizando una interpretación clara y sencilla sobre cómo afecta operativamente la norma al despegue y las restricciones que impone.

Empecemos analizando las velocidades que entran en juego en el despegue.

2.1 Velocidades en el Despegue

En esta sección vamos a describir y definir las velocidades que intervienen en el despegue según la norma. Para ello, haremos uso de la norma CS 25 y CFR part 25 en el punto 25.107, el mismo para

ambas.

Resaltar que todas las velocidades que vamos a ver aquí están expresadas en términos de CAS. Esto resultará importante luego en la aplicación informática, ya que todas las restricciones que se impongan sobre la velocidad deben ser contrastadas en términos de CAS, como impone la norma.

2.1.1 Velocidad mínima de control en el suelo V_{MCG}

Se define como la velocidad mínima a la cual, cuando falla el motor crítico (en general, el más alejado del eje longitudinal del avión y, por tanto, el que produciría un mayor momento respecto al centro de masas del avión), es posible mantener el control del avión usando únicamente controles aerodinámicos.

Se fija como límite de fuerza en los pedales 150 libras.

Además, asumiendo que cuando el avión acelera con todos los motores operativos sigue una línea recta por el centro de la pista, se establece que la senda que sigue a partir del fallo del motor crítico hasta el punto en el que consigue recuperar una dirección paralela a una línea recta en el centro de la pista no puede desviarse más de 30 ft. lateralmente del centro de la pista en ningún punto.

La norma exige (tal y como está recogido en el punto 25.149 (e) tanto de la CS 25 [3] como de la CFR part 25 [4]) que esta velocidad se establezca siguiendo las siguientes condiciones:

- Avión en la configuración de despegue más crítica.
- Régimen de motor en despegue
- Posición del centro de gravedad más desfavorable.
- Avión compensado para despegue.
- Peso más desfavorable para despegue.

2.1.2 Velocidad de decisión V_1

Se utiliza como la velocidad que sirve como referencia para determinar si se aborta el despegue o si se continua en el caso de una emergencia.

La emergencia más importante y que más se referencia durante la definición de las actuaciones del despegue es el fallo de un motor.

Si un motor falla antes de alcanzar V_1 se debe abortar el despegue, si falla después de V_1 se debe continuar el despegue.

A parte de V_1, se definen también la velocidad de fallo de motor V_{EF} como la velocidad a la que falla el motor durante el despegue y t_{REC} como el tiempo que transcurre desde que falla el motor hasta que el piloto se da cuenta de dicho fallo y ejecuta la primera acción al respecto. Entonces, se puede expresar V1 como

$$V_1 = V_{EF} + (\Delta V)_{t_{REC}}$$

La norma impone (en el punto 25.107 (a) tanto en la CS 25 [3] como la CFR part 25 [4]) que V_{EF} sea mayor que V_{MCG} y, por tanto, se debe cumplir también que

$$V_1 \geq V_{MCG} + (\Delta V)_{t_{REC}}$$

2.1.3 Velocidad mínima de control en el aire V_{MCA}

La definición de esta velocidad es análoga a V_{MCG}, pero referida en este caso a la situación en el aire.

Esta velocidad debe cumplir (según el punto 25.149 (c) tanto en la CS 25 [3] como en la CFR part 25 [4]) las mismas condiciones que V_{MCG} y, además, que

$$V_{MCA} \leq 1.13\, V_{SR}$$

Siendo V_{SR} la velocidad de entrada en pérdida de referencia definida por la norma.

Llegados a este punto, conviene explicar ciertos aspectos correspondientes a V_{SR}. La velocidad V_{SR} de la que estamos hablando en este trabajo y la que hoy en día se utiliza en la norma (punto 25.103 tanto en la CS 25 [3] como en la CFR part 25 [4]) se conoce como velocidad de pérdida a 1 g, V_{S1g}, velocidad a la que el avión puede desarrollar una sustentación normal a la trayectoria igual al peso (definición según [5]), o sea, factor de carga igual a la unidad.

Sin embargo, antes no se usaba esta velocidad, sino que se usaba la velocidad de entrada en pérdida V_S. Esta velocidad se obtiene igualando sustentación máxima y peso, imaginándonos una maniobra de vuelo horizontal. Para conseguir sustentación máxima, y utilizando la expresión para la sustentación,

$$L = \frac{1}{2}\rho V^2 S C_L$$

Utilizaremos el máximo coeficiente de sustentación, siendo este el coeficiente de sustentación máximo que conseguimos a un cierto ángulo de ataque, y, a partir del cual dicho coeficiente empieza a decaer.

Sin embargo, según se nos cuenta en [5], muchos aviones modernos no tienen claramente definido el valor de $C_{L\,máx.}$, pues la curva del coeficiente de sustentación en función del ángulo de ataque es muy plana en la zona próxima a la pérdida. Por lo tanto, se conseguía que el avión no entrase en pérdida a valores mayores del ángulo de ataque. El problema reside en que, en estas zonas de la curva, el factor de carga ha disminuido de la unidad, por lo que, aunque el avión no haya entrado en pérdida (todavía es controlable) su altitud va disminuyendo.

Podemos concluir que, buscando mayor seguridad, se optó por cambiar la velocidad de entrada en pérdida a una más restrictiva.

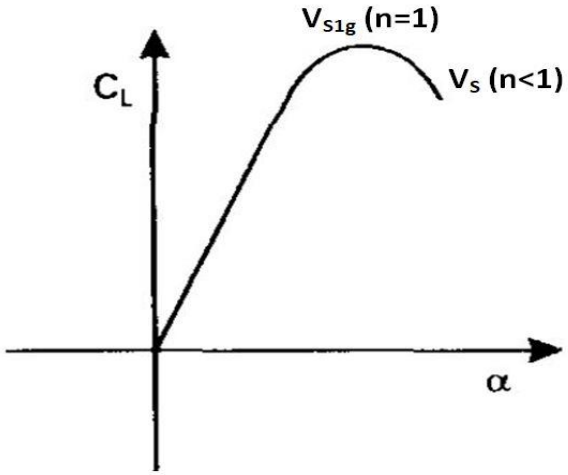


Figura 1. Gráfica comparativa entre V_S y V_{S1g}

Debido a este cambio, es posible que en algunos lugares aparezcan factores diferentes a los de la norma, ya que la velocidad V_{S1g} es más restrictiva que la velocidad de entrada en pérdida tradicional. A continuación, se muestra una imagen con algunos ejemplos de factores antiguos y su correspondencia con los nuevos.

Como podemos observar, se ha aplicado un factor de conversión de un 94% al factor antiguo para obtener el nuevo.

Velocidad	Factor antiguo	Factor nuevo
V_2	1,20 V_S	1,13 V_{S1g}
V_2	1,15 V_S	1,08 V_{S1g}
V_{Ref}	1,30 V_S	1,23 V_{S1g}
V. final de despegue	1,25 V_S	1,18 V_{S1g}
V. de subida en aprox.	1,50 V_{S1}	1,40 V_{S1g}

Figura 2. Ejemplos de factores antiguos y sus correspondientes nuevos obtenido de [5]

2.1.4 Velocidad V_{MU} (Minimum Unstick)

Se define como la velocidad mínima a la que el avión es capaz de separarse del suelo y mantener un ángulo de subida positivo de forma segura (punto 25.107 (d) tanto en la CS 25 [3] como en la CFR part 25 [4]).

Por encima de esta velocidad evitamos que el cono de cola choque con la pista durante la rotación del avión. Además, permite evitar que el avión se vaya al aire con una actitud de balance inadecuada.

2.1.5 Velocidad de rotación V_R

Es la velocidad a la que se debe hacer girar el avión alrededor del tren principal para tomar la actitud adecuada para irse al aire.

V_R no puede ser menor que (punto 25.107 (e) tanto en la CS 25 [3] como en la CFR part 25 [4]).

- V_1
- 1.05 de V_{MCA}
- La velocidad que permite alcanzar V_2 antes de alcanzar una altura de 35 pies.
- Una velocidad tal que, si la rotación del avión se ejecuta con la máxima rapidez, resulte en una V_{LOF} no menor que 1.10 V_{MU} con todos los motores operativos o 1.05 V_{MU} con un motor inoperativo.
- Una velocidad tal que, si la rotación del avión se ejecuta con la máxima rapidez y la actitud de V_{MU} está limitada por la geometría del avión, resulte en una V_{LOF} no menor que 1.08 V_{MU} con todos los motores operativos o 1.04 V_{MU} con un motor inoperativo.

La norma también exige que, si se produce la rotación del avión a una velocidad 5 nudos inferior a V_R, no se exceda la distancia de despegue correspondiente a cuando se realiza con V_R.

Cabe destacar que es de gran importancia calcular bien esta velocidad y efectuar la rotación a la misma, ya que puede afectar de manera muy significativa las actuaciones del avión durante el despegue.

2.1.6 Velocidad de despegue V_{LOF}

Se define como la velocidad a la que el avión despega el tren principal del suelo (punto 25.107 (f) tanto en la CS 25 [3] como en la CFR part 25 [4]).

No tiene mucho interés operacional, ya que la velocidad que condiciona la maniobra del despegue es V_R.

2.1.7 Velocidad de seguridad al despegue V_2

Se establece como la velocidad a la que se alcanzan los 35 pies de altura.

Guarda una estrecha relación con V_R, ya que, si esta aumenta o disminuye, lo mismo hará V_2.

La norma dice (puntos 25.107 (b) y (c) tanto en la CS 25 [3] como en la CFR part 25 [4]) que esta velocidad no puede ser menor que

- 1.13 V_{SR} para aviones turborreactores y para aviones turbohélices bimotor o trimotor.
- 1.08 V_{SR} para aviones turbohélices con más de tres motores y para aviones con motor alternativo.
- 1.10 V_{MCA}.

En la siguiente figura podemos ver un resumen de las velocidades durante el despegue, así como el orden en el que según la norma se debería ir alcanzando.

Aclarar que, aunque todas estas velocidades influyen en las actuaciones de despegue, las velocidades operativas, controladas por el piloto, son V_1, V_R y V_2.

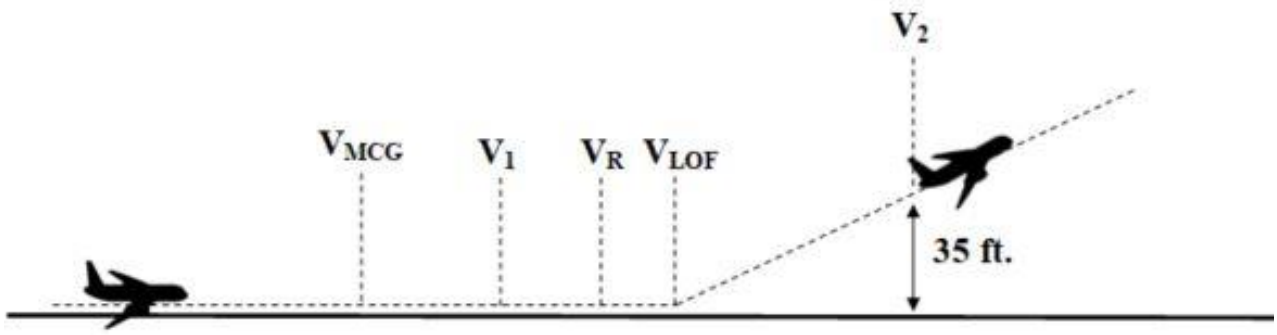


Figura 3. Resumen de las Velocidad en el Despegue

2.1.8 Velocidad máxima de neumáticos

Los neumáticos de los aviones están sometidos a grandes presiones y temperaturas, por lo que tener en cuenta su desgaste es importante para evitar accidentes durante el despegue.

Por ello, se limita la velocidad a la que el avión puede recorrer la pista por una velocidad máxima de neumáticos, que depende fundamentalmente del peso. Entonces, si para un peso dado, vemos que necesitamos alcanzar una velocidad en pista mayor a la velocidad máxima de neumáticos para este peso, sabemos que tenemos que bajar el peso. Por lo tanto, esta velocidad impone una limitación al despegue, y suele incluirse en las gráficas que aparecen en los manuales de vuelo como limitación al peso en despegue.

Para mayor información sobre los neumáticos, puede consultarse el punto CS 25.733 de la normativa europea [3], o el mismo punto 25.733 en la normativa americana [4].

2.1.9 Velocidad de máxima energía de frenado

Como veremos a continuación, si se decide abortar el despegue en un cierto instante durante el recorrido del avión sobre la pista, es necesario frenarlo. Gran parte de la energía cinética que tiene el avión tiene que ser absorbida por los frenos e las ruedas.

Por lo tanto, existe una velocidad máxima para cada peso que limita la velocidad máxima que puede tener un avión en caso de querer abortar el despegue, ya que, si se supera esta velocidad, los frenos no serían capaces de frenar el avión. Entonces, esta velocidad impone una limitación a la velocidad durante el despegue, y se suele incluir en las gráficas de los manuales de vuelo como limitación a tener en cuenta.

Para mayor información sobre los neumáticos, puede consultarse el punto CS 25.735 de la normativa europea [3], o el mismo punto 25.733 en la normativa americana [4].

Para concluir con las velocidades, comentar que se han omitido ciertos detalles y ciertas velocidades por considerarse de menor importancia para este trabajo. Se ha intentado resumir lo más importante y lo que más afecta de forma directa a las actuaciones durante el despegue, así como no transcribir la norma tal cual, sino ofrecer una interpretación más práctica de la misma. Todo con el objetivo de que esta parte del trabajo, algo más teórica, no resulte demasiado tediosa. No obstante, para más detalle puede consultarse la normativa que se incluye en la bibliografía.

2.2 Distancia de Aceleración-Parada (ASD)

Esta distancia, conocida normalmente como ASD (Accelerate Stop Distance) se define según la norma (punto 25.109 tanto en la CS 25 [3] como en la CFR part 25 [4]) como la mayor de las dos distancias siguientes:

1. La suma de las distancias necesarias para:
    i. Acelerar el avión desde la suelta de frenos hasta V_{EF}, momento en el que se supone que falla un motor crítico.
    ii. Acelerar el avión desde V_{EF} hasta V_1 y continuar dos segundos a esa velocidad, tiempo que se supone que tarda el piloto en reaccionar ante el fallo del motor crítico.
    iii. Detener completamente el avión, desde el punto final alcanzado cuando pasan los dos segundos del punto anterior.

2. La suma de las distancias necesarias para:
    i. Acelerar el avión desde la suelta de frenos hasta la velocidad V_1 y continuar la aceleración durante dos segundos, con todos los motores operativos.
    ii. Detener completamente el avión, desde el punto final alcanzado cuando pasan los dos segundos del punto anterior.

Para frenar el avión, no se tiene en cuenta el frenado con reversa, solo los medios de frenado convencionales están considerados, excepto si la pista se considera mojada, entonces se puede considerar (punto 25.109 (f) tanto en la CS 25 [3] como en la CFR part 25 [4]).

Dicho esto, aclarar que es una imposición normativa calcular las actuaciones de forma conservativa. En la operación real se puede usar reversa si es necesario o si se considera oportuno.

En la norma se explica extensamente los procedimientos que hay que tener en cuenta cuando se trata de pista mojada y los coeficientes de frenado que se deben de tener en cuenta en esta situación (puntos 25.109 (b), (c) y (d) tanto en la CS 25 [3] como en la CFR part 25 [4]).

Sin embargo, a efectos prácticos, y para no añadir excesiva dificultad al trabajo y demasiada carga teórica a la memoria, consideraremos que el efecto que provoca la pista mojada se tendrá en cuenta en el coeficiente de frenado que se indique en los datos y en las velocidades V_{EF} y V_1.

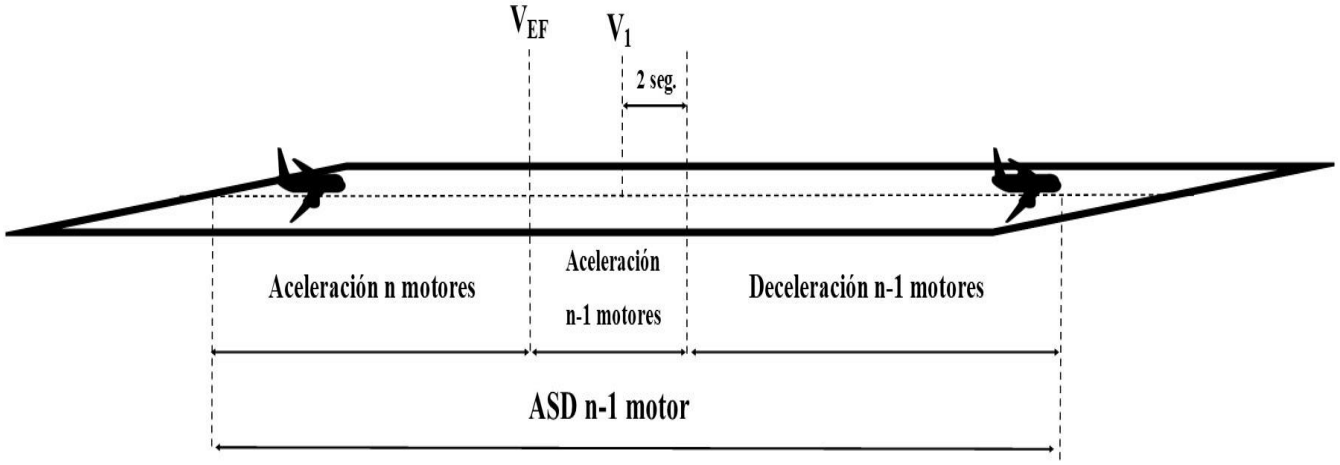


Figura 4. ASD con n-1 motores

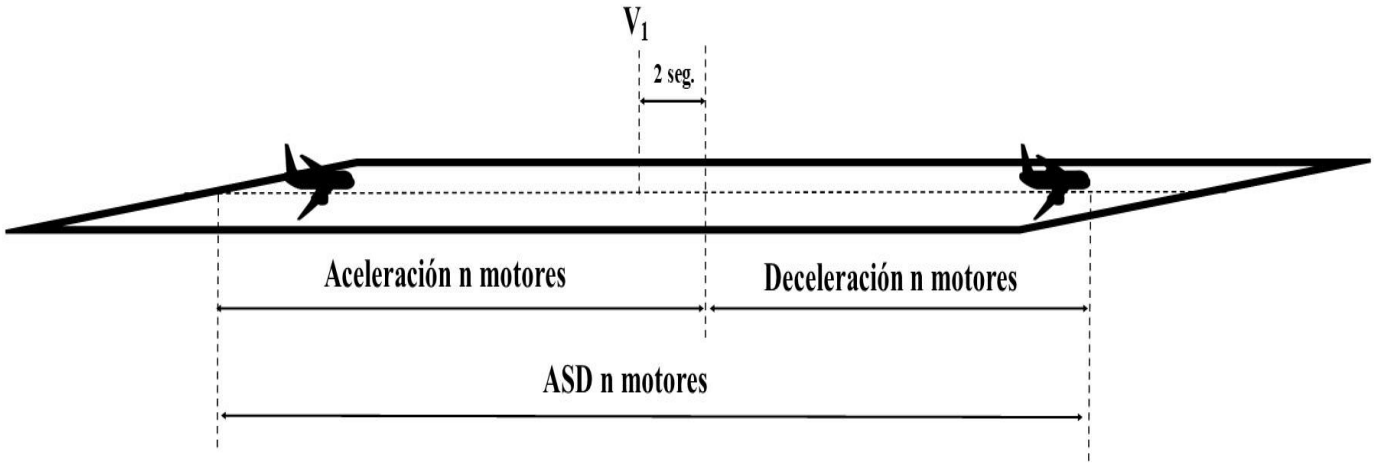


Figura 5. ASD con n motores

2.3 Distancia de Despegue (TOD)

Esta distancia, conocida normalmente como TOD (Take-Off Distance) se define según la norma (punto 25.113 (a) tanto en la CS 25 [3] como en la CFR part 25 [4]) como la mayor de las dos distancias siguientes:

1. Es el 115% de la distancia que se recorre desde que se sueltan los frenos del avión hasta que se alcanzan los 35 pies de altura a una velocidad V_2, con todos los motores operativos.
2. Es la distancia necesaria para alcanzar la velocidad V_{EF} con todos los motores operativos, y a partir de ese momento, suponiendo que se produce un fallo en un motor crítico, continuar el despegue hasta alcanzar una altura de 35 pies a una velocidad V_2.

Al igual que antes, omitimos la definición de la distancia de despegue en pista mojada, que se puede consultar en el punto 25.113 (b) tanto en la CS 25 [3] como en la CFR part 25 [4].

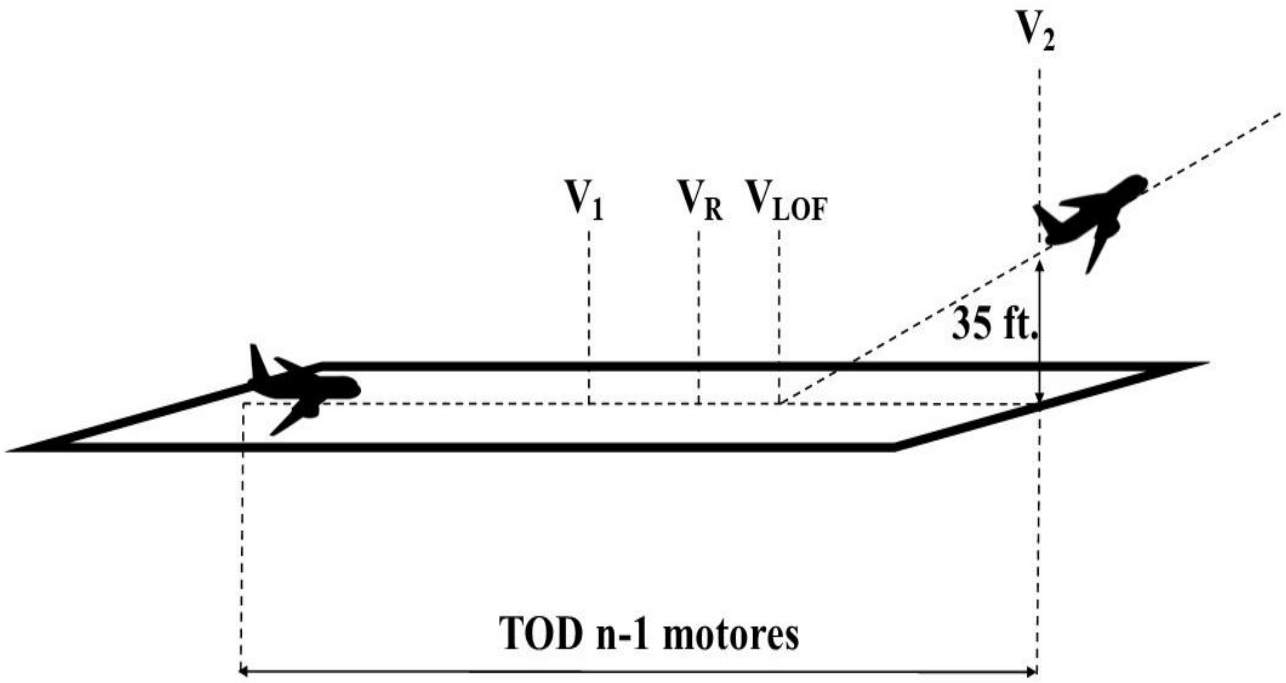


Figura 6. TOD con n-1 motores

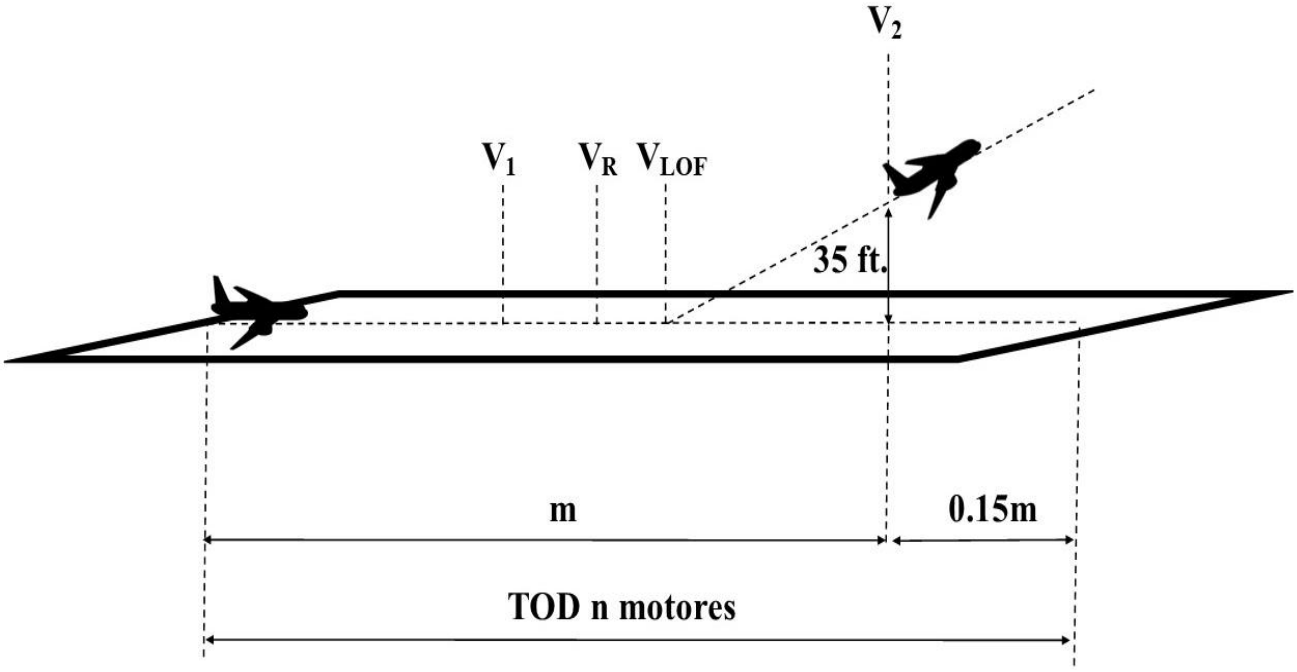


Figura 7. TOD con n motores

2.4 Carrera de Despegue (TOR)

Esta distancia, conocida normalmente como TOR (Take-Off Run) se define según la norma (punto 25.113 (c) tanto en la CS 25 [3] como en la CFR part 25 [4]) como la mayor de las dos distancias siguientes:

1. La distancia que existe desde que se sueltan los frenos, suponiendo que un motor falla a V_{EF}, hasta un punto equidistante entre el punto en el que se alcanza V_{LOF} y el punto en el que el avión alcanza una altura de 35 pies.
2. Es el 115% de la distancia que existe desde que se sueltan los frenos, con todos los motores operativos, hasta un punto equidistante entre el punto en el que se alcanza V_{LOF} y el punto en el que el avión alcanza una altura de 35 pies.

Como podemos leer en el punto de la norma antes citado, esta distancia solo se calculará si la pista incluye una zona libre de obstáculos (clearway). En caso contrario, será igual a la distancia de despegue.

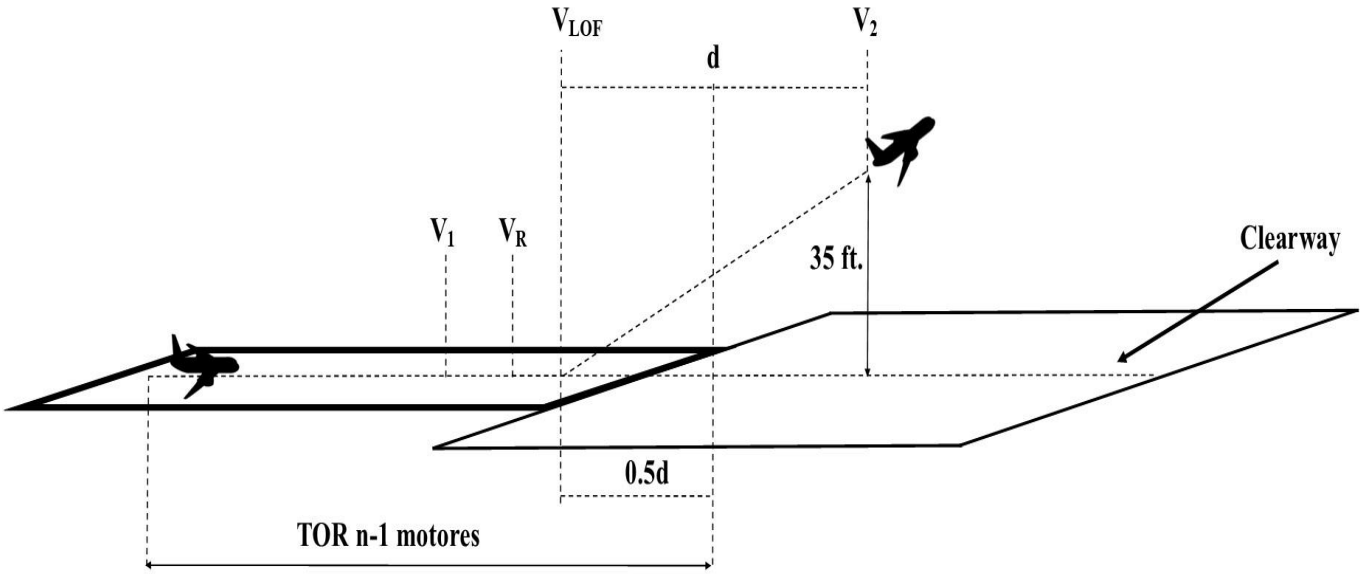


Figura 8. TOR con n-1 motores

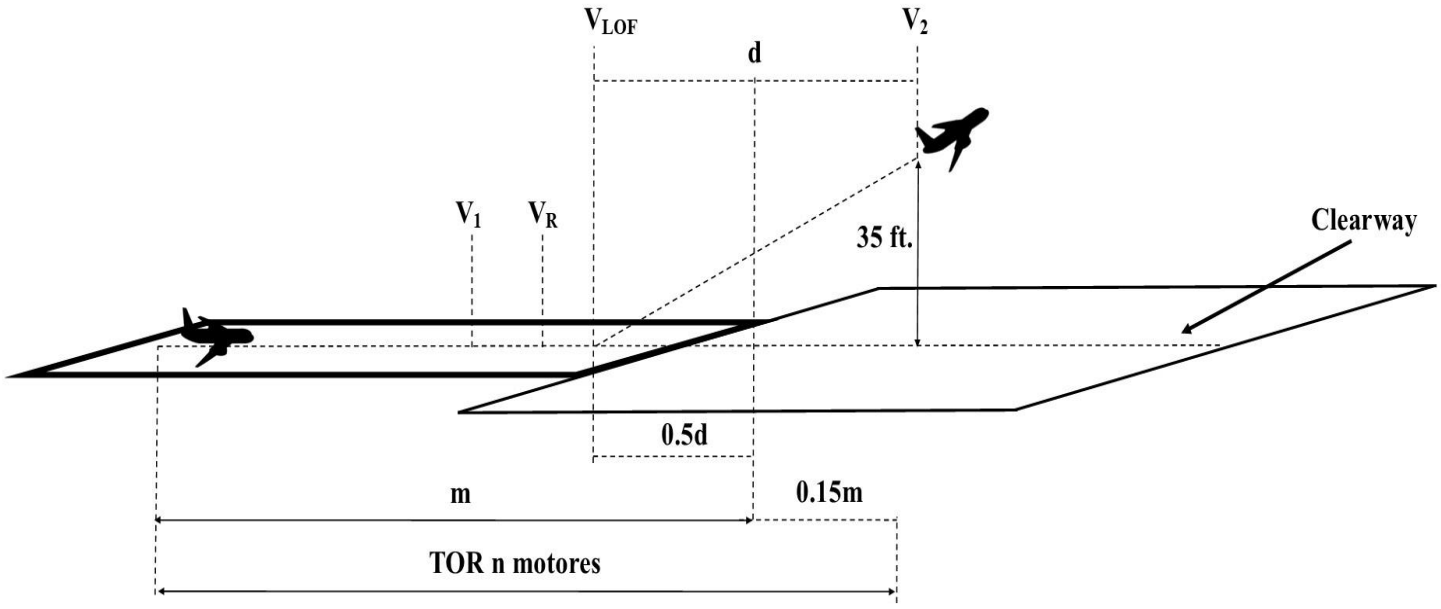


Figura 9. TOR con n motores

2.5 Distancias Declaradas

Brevemente definiremos las distancias que las autoridades exigen que los aeropuertos proporcionen como datos. Pero antes, definiremos dos zonas que influyen en estas distancias declaradas.

2.5.1 Zona de Parada (Stop-Way)

La zona de parada es un área que se encuentra al final de la pista cuyo objetivo es ser una longitud adicional de la pista que podría usar el avión para frenado en caso de emergencia.

Por lo tanto, para poder cumplir dicho cometido con seguridad, la zona de parada debe tener al menos la misma anchura que la pista y debe poder soportar el peso del avión sin sufrir daños.

2.5.2 Zona Libre de Obstáculos (Clearway)

La zona de parada es un área que se encuentra en la prolongación del eje de la pista y donde no hay obstáculos, de forma que puede usarse como espacio adicional para la subida en el despegue. Empieza al final de la pista y puede incluir (si existe) a la zona de parada.

Debe tener un mínimo de anchura de 500 pies, y si existen obstáculos, no pueden sobrepasar un plano de pendiente 1.25% que empiece al final de la pista (solo se permite que sobresalgan de este plano las luces de cabecera de la pista).

Según la norma (punto 25.113 (c) tanto en la CS 25 [3] como en la CFR part 25 [4]), si en una pista no existe zona libre de obstáculos (clearway), la carrera de despegue coincidirá con la distancia de despegue.

Una vez hemos definido estas dos zonas, pasamos a ver las distancias declaradas.

Para el despegue, tenemos las siguientes distancias declaradas:

- TORA (Take-Off Run Available): La Carrera de despegue disponible es la longitud de pista que se ha declarado como apta para el recorrido del avión en tierra.
- TODA (Take-Off Distance Available): La distancia de despegue disponible es la longitud de pista disponible más la zona libre de obstáculos.
- ASDA (Accelerate Stop Distance Available): La distancia de aceleración parada disponible es la longitud de pista disponible más la zona de parada.

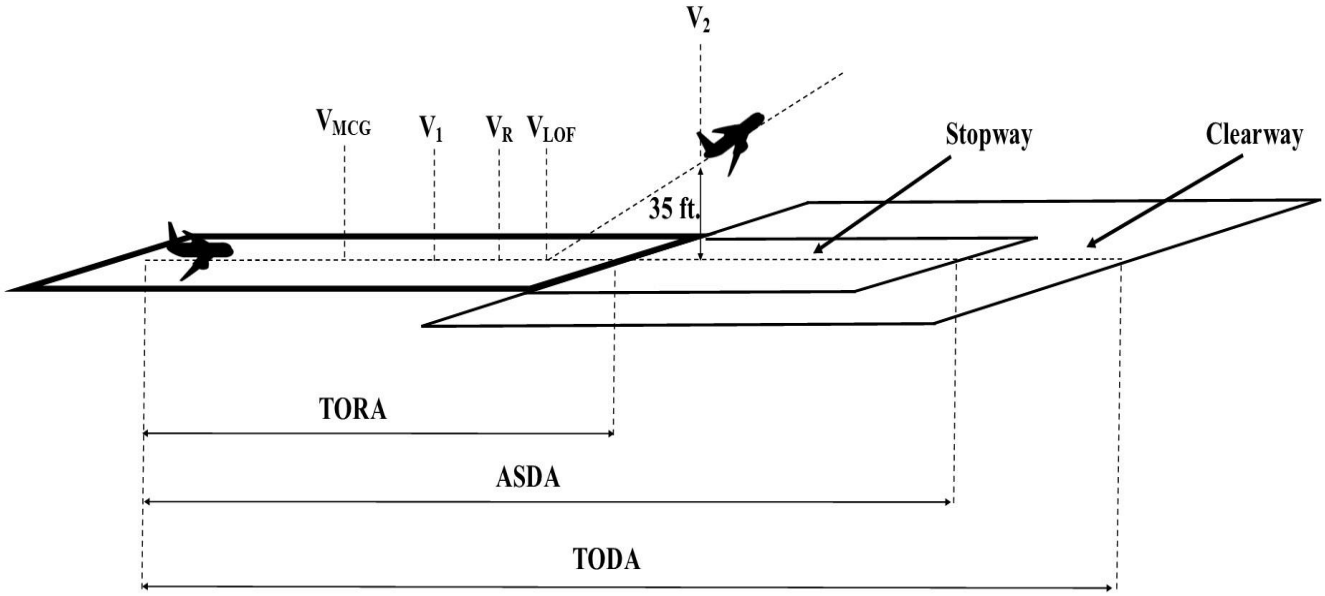


Figura 10. Distancias Declaradas

Continuemos analizando las fases o segmentos que componen el despegue.

2.6 Senda de Despegue

Comienza en el momento en el que se sueltan los frenos y finaliza cuando el avión alcanza al menos los 1500 ft. y se tiene la configuración y velocidad de subida en ruta.

La norma únicamente exige (punto 25.111 (d) tanto en la CS 25 [3] como en la CFR part 25 [4]) que el despegue esté definido de forma continua o por segmentos, y si es por segmentos que estos estén definidos con claridad y se especifiquen los distintos cambios en configuración, potencia y empuje, y velocidad.

Normalmente se divide en segmentos. La definición de estos segmentos puede variar según el fabricante, por lo que en este trabajo nos guiaremos por la descripción que se hace del despegue en el libro [5].

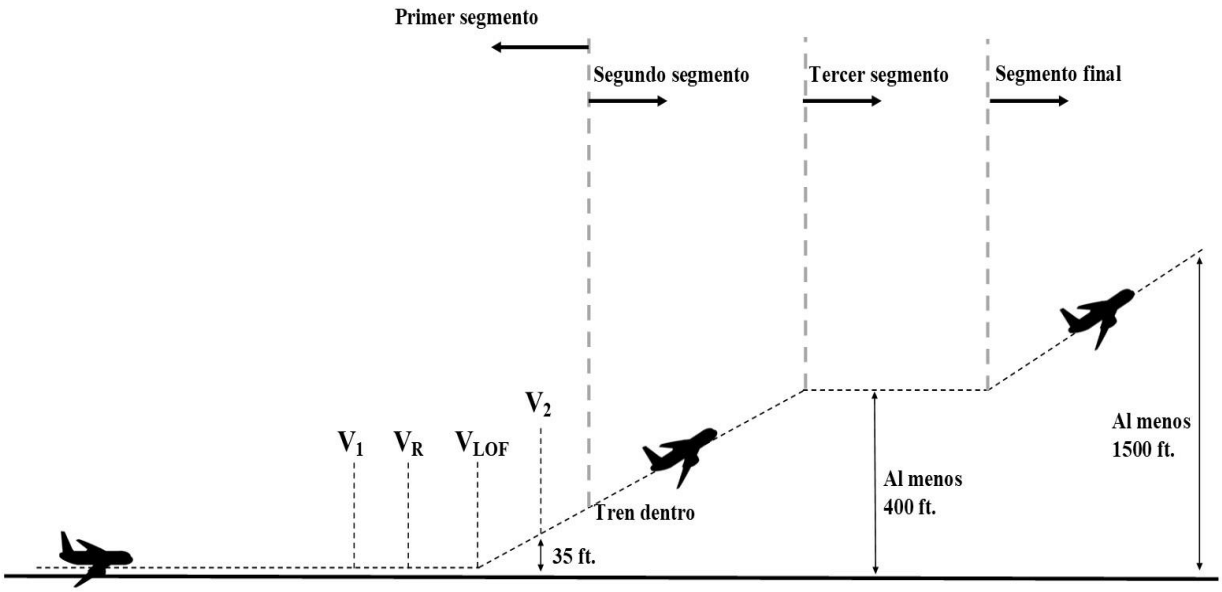


Figura 11. Senda de Despegue

2.6.1 Primer segmento

Se puede considerar que empieza o bien a 35 ft. de altura y con velocidad V_2, o bien cuando el avión despega las ruedas del suelo y la velocidad entonces variará entre V_{LOF} y V_2.

En cualquier caso, debe cumplirse también:

- Régimen de motor en despegue.
- Tren fuera, hasta el punto en el que se repliega.
- Flaps en posición de despegue.

La pendiente de la trayectoria de subida deberá ser:

- Para aviones bimotor, positiva.
- Para aviones trimotor, mayor que 0,3%.
- Para aviones cuatrimotor, mayor que 0,5%.

2.6.2 Segundo segmento

Se considera su comienzo a partir del momento en el que se recoge el tren de aterrizaje por completo.

Debe cumplirse:

- Régimen de motor en despegue.
- Tren replegado.
- Flaps en posición de despegue.
- Velocidad mínima V_2.

El fin del segundo segmento puede ser:

a) A una altura mínima de 400 ft.
b) En la máxima altura de vuelo nivelado, definida como la altura a la que el avión es capaz de mantener la altura constante, acelerar hasta la velocidad de meter flaps y slats y continuar acelerando hasta la velocidad del segmento final dentro del tiempo límite durante el cual se puede disponer del régimen de motor en despegue. (Normalmente el tiempo límite es de 5 o 10 minutos según el avión).
c) En la máxima altura que puede alcanzarse subiendo, dentro del tiempo del que se puede disponer del régimen de motor en despegue.

La pendiente de la trayectoria de subida deberá ser:

- Para aviones bimotor, mayor o igual que 2,4%.
- Para aviones trimotor, mayor o igual que 2,7%.
- Para aviones cuatrimotor, mayor o igual que 3%.

2.6.3 Tercer segmento

El avión acelera en vuelo nivelado, primero para recoger flaps y slats a la velocidad correspondiente y después alcanzar la velocidad del segmento final.

La velocidad no debe ser menor que 1,2 V_s.

Se suele seguir con régimen de motor en despegue.

2.6.4 Segmento final

Comienza en el momento en que los flaps se han metido y se ha alcanzado la velocidad de segmento final y dura hasta que se alcanza una altura de mínimo 1500 ft.

El empuje o la potencia es el máximo continuo y la velocidad no debe ser menor que 1,25 $V_{s.}$

La pendiente de la trayectoria de subida deberá ser:

- Para aviones bimotor, mayor o igual que 1,2%.
- Para aviones trimotor, mayor o igual que 1,5%.
- Para aviones cuatrimotor, mayor o igual que 1,7%.

Antes de terminar con las actuaciones en el despegue, mencionar que también existe la senda neta de despegue, la cual empieza a 35 pies de altura y se obtiene calculando la senda que sigue el avión y restándole un tanto por ciento en su pendiente de subida.

Sin embargo, por carecer de interés para este trabajo, ya que nos centramos en el tramo hasta los 35 pies, solo mencionamos su existencia, pero no entraremos en detalles sobre que exige la norma para la senda de despegue.

Se han omitido diversos detalles e indicaciones que también figuran en la norma por considerarse no tan esenciales para las actuaciones como los que sí se han añadido y para no hacer demasiado engorrosa la memoria.

2.7 Conclusiones respecto a la normativa aplicable

Finalmente, como punto final del capítulo, recordemos que uno de los objetivos era comparar ambas normas y ver si era necesario aplicar ambas normas o si, por el contrario, aplicando una sola norma era suficiente.

Como hemos podido ver a lo largo del capítulo, en lo que respecta al despegue y a sus actuaciones, los punto de las normas europea CS 25 Amendment 22 [3] y americana CFR Part 25 de 2013 [4], coinciden sin ninguna diferencia destacable, siendo las restricciones sobre velocidad o las definiciones de las distancias exactamente las mismas.

Por tanto, podemos concluir este apartado diciendo que está justificado la utilización de una sola de las normativas, ya que en realidad si se cumple una de ellas, se cumplen las dos.

3. MODELIZACIÓN DEL PROBLEMA

En este capítulo realizaremos una descripción del modelo dinámico que hemos utilizado para aproximar el problema real del despegue de un avión, y, a continuación, explicaremos como hemos resuelto este modelo dinámico.

Primero se hará una descripción teórica del modelo, explicando qué hipótesis y simplificaciones hemos usado para nuestro modelo y como lo hemos construido. Después se explicará la resolución analítica del mismo con la ayuda del programa informático Matlab, que recordemos que será la herramienta principal para este trabajo.

3.1 Modelo dinámico

Para poder analizar el despegue de un avión y sus actuaciones, resulta primordial poder simular dicho despegue. Sin embargo, esto no es tarea fácil, ya que un despegue es una maniobra tremendamente compleja en la que influyen un gran número de factores. Por ello es necesario crear un modelo dinámico simplificado que nos permita recrear la maniobra del despegue de una forma sencilla, pero también lo más precisa posible.

Este modelo servirá como los cimientos del programa informático y a su alrededor se construirá el resto del programa. Para poder construir este modelo, serán necesarias ciertas hipótesis simplificadoras que resten dificultad al problema, pero que a la vez se aproximen a la realidad.

Como primera hipótesis, se va a considerar al avión un punto, y el movimiento se va a considerar en 2D. De esta forma, se elimina toda la complejidad derivada de la geometría del avión. Ahora el avión se considerará un punto al cual le afectan una serie de fuerzas.

Aplicando la segunda ley de Newton a este punto, seremos capaces de construir un sistema de ecuaciones diferenciales cuya resolución nos proporcionará la distancia y la velocidad que lleva el punto, o sea, nuestro avión. Pero claro, para ello habrá que imponer una serie de condiciones. Esto se verá más adelante.

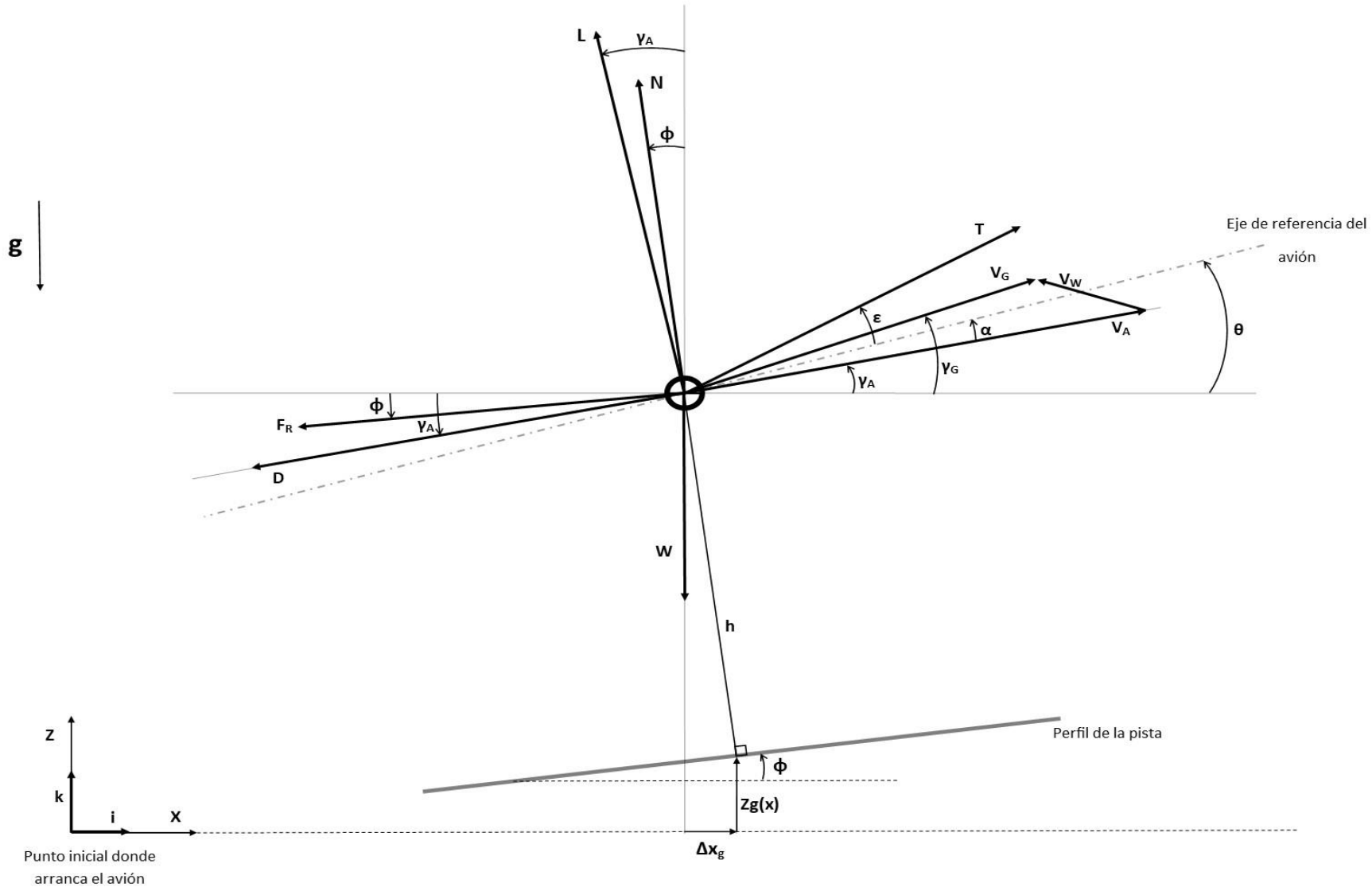


Figura 12. Modelo dinámico de despegue

En la figura 12 podemos observar cómo nuestro avión, ya convertido en un punto, se ve afectado por una serie de fuerzas. Estas fuerzas son,

- W: Fuerza peso
- T: Empuje
- D: Resistencia aerodinámica
- L: Sustentación
- F_R: Fuerza de rozamiento con la pista
- N: Fuerza normal con respecto a la pista

También observamos una serie de ángulos y distancias,

- ϕ: Ángulo de la pendiente de la pista
- α: Ángulo de ataque
- γ_A: Ángulo de asiento de la velocidad aerodinámica
- γ_G: Ángulo de asiento respecto al suelo de la velocidad respecto al suelo
- θ: Ángulo de asiento del avión
- ϵ: Ángulo de ataque del empuje
- h: Altura del centro de gravedad del avión con respecto a la pista
- x: coordenada horizontal del sistema de referencia centrado en el punto donde arranca el avión
- z: coordenada vertical del sistema de referencia centrado en el punto donde arranca el avión
- $Z_g(x)$: Altura de la pista con respecto al punto inicial, en función de x
- Δx_g: Diferencia entre la coordenada x del avión marcada por la perpendicular a la pista y la coordenada x del avión respecto al punto inicial

Por último, tenemos las siguientes velocidades,

- V_A: Velocidad aerodinámica
- V_G: Velocidad con respecto al suelo
- V_W: Velocidad del viento

Este conjunto conforma nuestro modelo dinámico de despegue de un avión. Como ya adelantábamos antes, se trata de un modelo simplificado, pero que incluye lo esencial para poder estudiar esta maniobra.

Con respecto a las velocidades, vemos que la velocidad V_A y V_G se relacionan mediante la velocidad del viento V_W. En la realidad, esta velocidad del viento puede tener cualquier dirección y sentido, siendo, por ejemplo, velocidad del viento lateral. Considerar esto supondría un aumento de dificultad considerable, por lo que se decidió realizar una segunda hipótesis.

La velocidad del viento V_W se considera paralela al perfil de la pista en todo momento. Esta hipótesis simplifica en gran medida la relación entre las velocidades. Entonces, el viento podrá ser de cara, si viene en dirección al morro del avión, o de cola, si viene en dirección a la cola del avión.

De esta forma, el esquema de las velocidades quedaría como se ve en la figura 13.

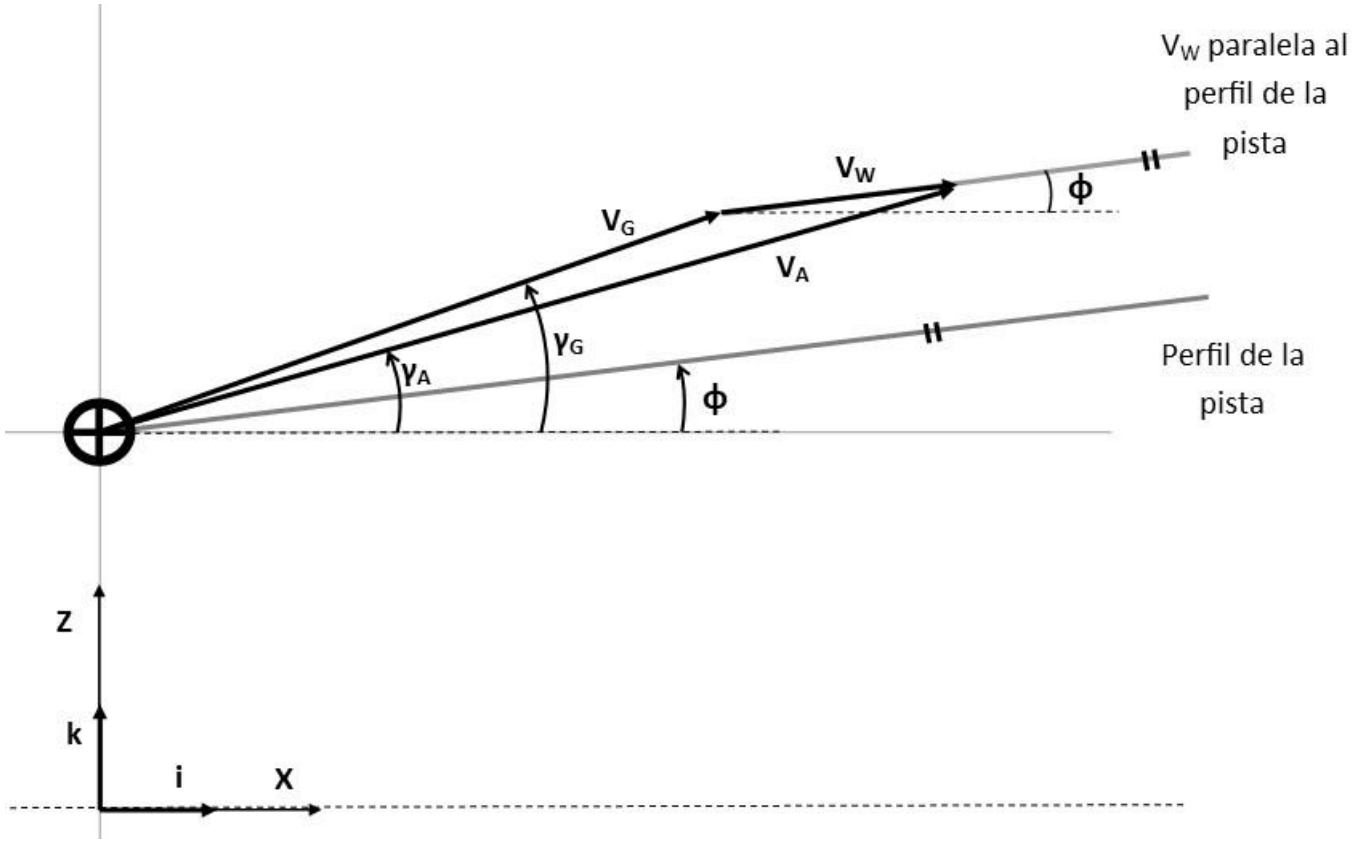


Figura 13. Modelo dinámico de las Velocidades

Una vez tenemos definido nuestro modelo dinámico, ahora lo que tenemos que hacer es construir nuestro sistema de ecuaciones diferenciales basado en este modelo que acabamos de ver. Una vez tengamos nuestro sistema de ecuaciones, podremos proceder a su resolución y análisis de resultados.

Para mayor claridad, dividiremos el sistema de ecuaciones en distintos grupos o categorías.

3.1.1 Relación entre velocidades

Primero, obtendremos la relación entre las velocidades que vemos en la figura 13.

Como comentábamos antes, la velocidad respecto a tierra y respecto a la corriente de aire están relacionadas mediante la velocidad del viento. Conocer esta relación es importante, lo cual se verá reflejado más adelante.

Usando el teorema del coseno y relaciones trigonométricas, la relación entre velocidades serían las siguientes,

$$V_W^2 = V_A^2 + V_G^2 - 2V_A V_G \cos(\gamma_A - \gamma_G)$$
$$V_G sen(\gamma_G - \phi) = V_A sen(\gamma_A - \phi)$$

3.1.2 Segunda ley de Newton aplicada al modelo

Ya hemos visto cómo nuestro avión, ahora convertido en punto, está sometido a una serie de fuerzas que generan un movimiento y unas velocidades. Se ha definido en el modelo una serie de distancias y ángulos para tener también definida la posición de nuestro avión en todo momento.

Una vez tenemos todo esto implementado, solo queda aplicar la segunda ley de Newton al modelo. Concretamente, se aplica al eje X y al eje Z, definidos en la figura 11.

$$m\ddot{x} = Tcos(\theta + \epsilon) - Lsen(\gamma_A) - Dcos(\gamma_A) - Nsen(\phi) - F_R\cos(\phi)$$

$$m\ddot{z} = Tsen(\theta + \epsilon) + Lcos(\gamma_A) - Dsen(\gamma_A) + Ncos(\phi) - F_R\,\mathrm{sen}(\phi) - W$$

Estas dos ecuaciones van a formar nuestro sistema de ecuaciones diferenciales a resolver. Para ello, deberemos definir todas las variables del problema en función de la velocidad, pues para resolver este sistema integraremos con respecto a la velocidad. Todo esto se verá con detalle más adelante, ahora mismo lo importante es saber que es necesario tener el resto de parámetros en función de la velocidad.

Así pues, continuamos con la definición de todos los parámetros que aparecen en estas ecuaciones.

3.1.3 Fuerzas propulsivas y aerodinámicas

Tanto la fuerza de sustentación como la fuerza de resistencia las definiremos con ayuda de los coeficientes de sustentación y de resistencia respectivamente. Más adelante, cuando hablemos del modelo aerodinámico, definiremos ambos coeficientes. Por ahora, es suficiente con obtener estar fuerzas en función de la velocidad.

La fuerza propulsiva de igual forma se verá más adelante cuando se defina el modelo propulsivo.

$$L = \frac{1}{2}\rho V_A^2 S_{alar} C_L$$

$$D = \frac{1}{2}\rho V_A^2 S_{alar} C_D$$

$$T = T(V_A)$$

Siendo,

- ρ: Densidad
- C_L: Coeficiente de sustentación
- C_D: Coeficiente de resistencia
- S_{alar}: Superficie alar
- T: Empuje
- D: Resistencia aerodinámica
- L: Sustentación

- V_A: Velocidad con respecto a la corriente

3.1.4 Perfil de la pista

Es sumamente importante, como ya veremos con más detalla más adelante, para este trabajo la definición del perfil de la pista de despegue. Para poder integrar este modelo, es necesario conocer dicho perfil, o sea, tener para cada coordenada x una coordenada z_g que marque el perfil de la pista.

$$z_g = z_g(x)$$

El modo de hacerlo se explicará más adelante cuando se trate el tema de cómo se va a programar todo el modelo, pero ya podemos ir adelantando que uno de los puntos más destacables del trabajo va a ser la posibilidad de que el usuario sea capaz de definir dicho perfil de pista, ya sea una pista constante con una cierta pendiente, o ya sea una pista real definida por diversos puntos y, por tanto, variable.

Como ya se ha dicho, todo este tema se tratará con más detalle. Se recuerda que, cuando se habla del perfil de la pista, se está hablando de que para cada posición horizontal del avión estará definida una posición vertical de la pista, y esta pista podrá ser constante o variable.

3.1.5 Parámetros relativos a ángulos y ligaduras geométricas

Estas expresiones relación los ángulos y las distancias que entran en juego en nuestro modelo. La obtención de la mayoría de ellas es muy sencilla, mientras que en otras se ha tenido que hacer uso de relaciones trigonométricas.

$$\theta = \gamma_A + \alpha$$

$$\Delta x_g = h z_g' \frac{1}{\sqrt{1 + \left(z_g'\right)^2}}$$

$$\mathrm{z} = z_g + \frac{h}{\sqrt{1 + \left(z_g'\right)^2}}$$

$$x_g = x + \Delta x_g$$

$$z_g = z_g(x)$$

Por último, y para poder relacionar las velocidades que aparecen en el modelo con la derivada de nuestra coordenada horizontal, tenemos que,

$$\dot{x} = V_G \cos(\gamma_G)$$

$$\dot{z} = V_G \text{sen}(\gamma_G)$$

Donde,

- α: Ángulo de ataque
- γ_A: Ángulo de asiento de la velocidad con respecto a la corriente
- h: Altura con respecto a la pista
- $Z_g(x)$: Altura del perfil de la pista en función de x
- Z_g': Derivada con respecto a la coordenada x de la altura del perfil de la pista
- Δx_g: Diferencia entre la coordenada x con respecto a pista y la coordenada x horizontal

A continuación, pasamos a definir las fases de las que suele constar un despegue. Estas fases nos ayudarán a resolver nuestro sistema ya que imponen condiciones que se han de cumplir y, por tanto, nos darán ecuaciones y datos extra.

3.1.6 Fases del despegue

Normalmente el despegue consta de tres fases, de manera simplificada. Durante estas tres fases el avión acelera, toma una posición adecuada para despegar y, finalmente despega hasta alcanzar una cierta altura.

Además de las ecuaciones que conforman nuestro modelo, para cada fase del despegue tendremos unas condiciones o ligaduras específicas. Estas condiciones las traduciremos como las siguientes ecuaciones:

- Fase de rodadura: Esta fase se extiende desde que el avión se encuentra parado hasta que ha alcanzado una velocidad óptima para empezar a rotar, ósea, hasta que ha alcanzado V_R. Las condiciones que impone esta fase son ángulo de asiento constante, altura sobre la pista constante y existe fuerza de rozamiento.

$$\theta = \theta_0 + \phi(x)$$
$$h = h_0$$
$$F_R = \mu N$$

- Fase de rotación: En esta fase se produce la rotación del avión desde un ángulo de asiento geométrico o inicial hasta un ángulo de asiento final, adecuado para poder despegar. Esta fase entonces duraría hasta que se alcanzase la velocidad de despegue V_{LOF}. Las condiciones que impone esta fase son ángulo de asiento sigue una ley de rotación, altura sobre la pista constante, y existe una fuerza de rozamiento.

$$\theta = \theta(t)$$
$$h = h_0$$
$$F_R = \mu N$$

- Fase de vuelo: Esta fase comprende el tramo en el que el avión ya ha despegado del suelo hasta que alcanza la altura final que estamos considerando, esto es, 35 pies. El ángulo de asiento sigue una ley de rotación, y en este caso no existen ya ni fuerza normal al peso ni fuerza de rozamiento.

$$\theta = \theta(t)$$
$$F_R = 0$$
$$N = 0$$

Donde,

- F_R: Fuerza de rozamiento con la pista
- N: Fuerza normal con respecto a la pista
- ϕ: Ángulo de la pendiente de la pista
- θ: Ángulo de asiento
- h: Altura con respecto a la pista

Ya tenemos establecido tanto el modelo dinámico como las ecuaciones asociadas, resultando un sistema de ecuaciones diferenciales de segundo orden.

A continuación, procederemos a explicar cómo se va a resolver este modelo. Como ya adelantábamos antes, su resolución se realizará con el programa informático Matlab. Por ello, deberemos adaptar nuestro modelo y nuestras ecuaciones a la forma que tiene este programa de resolver este tipo de sistemas de ecuaciones diferenciales de segundo orden.

3.2 Resolución analítica

Aunque existen métodos numéricos específicos para problemas de segundo orden, Matlab no los incorpora. Debido a esto, se usará el comando de Matlab ‘’ode45" [6], el cual utiliza el método de orden medio para resolver ecuaciones diferenciales no rígidas. La forma de resolver el nuestro sistema de segundo orden será convertirlo a uno de primer orden de la siguiente forma,

$$\dot{x} = V_x$$
$$\ddot{x} = \dot{V}_x = \frac{1}{m}(T cos(\theta + \epsilon) - L sen(\gamma_A) - D cos(\gamma_A) - N sen(\phi) - F_R \cos(\phi))$$

De esta forma podemos convertir nuestro sistema de segundo orden en uno de primer orden. Cabe mencionar llegados este punto que, como se puede observar, solo hemos transformado la ecuación correspondiente a la coordenada x, esto es porque durante las dos primeras fases del despegue la coordenada z del avión será un dato, el perfil de la pista. Por ello no es necesario calcularla, ahondaremos más en este tema cuando separemos el cálculo por fases.

Continuando con la resolución de nuestro sistema, como se comentaba anteriormente, es necesario que nuestro sistema solo esté en función de la coordenada x, su derivada respecto al tiempo, y del

tiempo.

$$\begin{cases} \dot{x} = f_1(x, \dot{x}, t) \\ \ddot{x} = f_2(x, \dot{x}, t) \end{cases}$$

Entonces, tenemos que poner todas las variables en como una función de x, $\dot{x}$ y el tiempo.

El empuje, tanto si es turbofán como turbohélice, está en función de la velocidad aerodinámica V_A

$$T = T(V_A)$$

La sustentación y la resistencia aerodinámica están también en función de la velocidad aerodinámica y el ángulo de ataque, que es conocido,

$$D = \frac{1}{2}\rho V_A^2 S_{alar} C_D$$
$$L = \frac{1}{2}\rho V_A^2 S_{alar} C_L$$

Más adelante veremos las expresiones tanto para el empuje como para los coeficientes de sustentación y resistencias, los cuales hemos formulado en función de los parámetros que nos interesa.

Todos los ángulos excepto γ_A son también conocidos.

Por tanto, lo único que nos queda es definir tanto γ_A como V_A en función de x y $\dot{x}$.

$$\overrightarrow{V_A} = \overrightarrow{V_g} + \overrightarrow{V_W} \quad \begin{cases} \overrightarrow{V_W} = V_{Wx}\vec{\imath} + V_{Wz}\vec{k} = V_W cos(\phi)\vec{\imath} + V_W sen(\phi)\vec{k} \\ \overrightarrow{V_g} = V_{gx}\vec{\imath} + V_{gz}\vec{k} = V_g cos(\gamma_g)\vec{\imath} + V_g sen(\gamma_g)\vec{k} = \dot{x}\vec{\imath} + \dot{z}\vec{k} \\ \overrightarrow{V_A} = V_{Ax}\vec{\imath} + V_{Az}\vec{k} = V_A cos(\gamma_A)\vec{\imath} + V_A sen(\gamma_A)\vec{k} \end{cases}$$

$$\begin{cases} V_{Ax} = V_{gx} + V_{Wx} \\ V_{Az} = V_{gz} + V_{Wz} \end{cases} \qquad \begin{cases} V_{Ax} = \dot{x} + V_{Wx} \\ V_{Az} = \dot{z} + V_{Wz} \end{cases}$$

Recordemos que la velocidad del viento es un dato.

A partir de este momento es conveniente separar la resolución en las distintas fases del despegue que hemos comentado anteriormente.

3.2.1 Fase de rodadura

Para la fase de rodadura tenemos las siguientes condiciones impuestas, todas comentadas anteriormente pero ahora en forma de ecuación,

$$\theta = \theta_0 + \phi(x)$$
$$h = h_0$$
$$F_R = \mu N$$
$$\gamma_A = \gamma_g = \phi$$
$$z = f(x) \qquad \dot{z} = f'(x)\,\dot{x} \qquad \ddot{z} = f''(x)\dot{x} + f'(x)\ddot{x}$$

Por lo tanto, con todo lo anterior y ya que mientras se rueda por la pista, el ángulo de asiento de la velocidad coincide con ángulo del perfil de la pista, ya tenemos γ_A y V_A en función de x y $\dot{x}$,

$$V_A = \frac{\dot{x}}{\cos(\phi)} + V_W$$
$$\gamma_A = \phi$$

Aún nos falta saber la expresión de la fuerza de rozamiento y de la fuerza normal a la pista, para ello usamos la ecuación correspondiente a la coordenada z, que recordemos que es un dato al coincidir durante esta fase con el perfil de la pista

$$F_R = \mu N$$
$$m\ddot{z} = Tsen(\theta + \epsilon) + Lcos(\gamma_A) - Dsen(\gamma_A) + Ncos(\phi) - \mu N\,\mathrm{sen}(\phi) - W$$

Al conocer la expresión de $\ddot{z}$, podemos despejar N en función de x y $\dot{x}$,

$$N = \frac{m\ddot{z} - Tsen(\theta + \epsilon) - Lcos(\gamma_A) + Dsen(\gamma_A) + W}{\cos(\phi) - \mu sen(\phi)}$$

Ya podemos volver a la expresión de $\ddot{x}$ y reordenarla,

$$\ddot{x} = \frac{Tcos(\theta + \epsilon) - Lsen(\gamma_A) - Dcos(\gamma_A) - (sen(\phi) + \mu\cos(\phi)\frac{mf''(x)\dot{x} - Tsen(\theta + \epsilon) - Lcos(\gamma_A) + Dsen(\gamma_A) + W}{\cos(\phi) - \mu sen(\phi)}}{m(1 + \frac{mf''(x)(sen(\phi) - \mu\cos(\phi))}{m(\cos(\phi) - \mu sen(\phi))})}$$

Llegados a este punto, la fase de rodadura se puede resolver de forma numérica con el comando anteriormente citado. De todas maneras, más adelante se detallará más el proceso de resolución.

3.2.2 Fase de rotación

Para la fase de rotación tenemos las siguientes condiciones:

$$\theta = \theta(t)$$
$$h = h_0$$
$$F_R = \mu N$$
$$\gamma_A = \gamma_g = \phi$$
$$z = f(x) \qquad \dot{z} = f'(x)\,\dot{x} \qquad \ddot{z} = f''(x)\dot{x} + f'(x)\ddot{x}$$

Entonces, todo en esta fase es exactamente igual que en la anterior excepto que ahora el ángulo de asiento es función del tiempo.

La resolución se realizará de la misma forma que en la anterior fase y con las mismas expresiones.

3.2.3 Fase de vuelo

En esta fase es donde más cambia el sistema, ya que ahora, a diferencia de las anteriores fases, la coordenada z no es un dato pues ya el avión ha despegado de la pista. Para la fase de vuelo tenemos las siguientes condiciones,

$$F_R = 0$$
$$N = 0$$
$$\mathrm{z} = z_g + \frac{h}{\sqrt{1 + \left(z_g'\right)^2}}$$

Con respecto al ángulo de asiento, tenemos dos opciones. La primera opción es que la anterior fase de rotación haya terminado (el avión se ha levantado del suelo, la fuerza normal se ha hecho cero) antes de que se haya alcanzado el ángulo de asiento final. En este caso, continúa la rotación en el aire.

$$\theta = \theta(t)$$

La segunda opción, generalmente no deseada en la maniobra de despegue real, es que alcancemos el ángulo de asiento final durante la rotación, pero todavía continúe la fase de rotación (el avión todavía no se ha separado del suelo, todavía existe fuerza normal mayor que cero). En este caso,

$$\theta = \theta_f$$

Llegados a este punto conviene hacer un comentario sobre este ángulo de asiento. Durante estas dos fases de vuelo y de rotación existe la limitación geométrica de que el cono de cola del avión no puede

chocar con el suelo, lo que se traduce en que existe un ángulo de asiento máximo que limita la ley de rotación. Por lo tanto, durante toda la maniobra el ángulo de asiento real deberá ser menor que el ángulo de asiento que limita dicha colisión.

Esta limitación existe y debe ser tenida en cuenta, aunque en este trabajo no se incluirá la opción de limitar el ángulo de asiento máximo. Sin embargo, constará en la sección de mejoras futuras del capítulo 6.

Continuando con la resolución, recordemos que la expresión para z se vio anteriormente, y fue obtenida a partir de relaciones trigonométricas y utilizando la definición de derivada con respecto a la coordenada x.

Conocemos z_g como función de x, pero no conocemos h.

Esta variable será necesaria para determinar cuando llegamos a los 35 pies de altura, punto en el cual consideraremos que se ha terminado el despegue. Además de esto, esta variable es importante debido al efecto suelo, el cual altera aerodinámicamente al avión, y depende en parte de la altitud a la que esté el avión sobre el suelo. Se hablará más delante de este efecto y de las consecuencias que acarreará, por ahora, adelantar que este efecto existe, tiene importancia, y la variable h está relacionada con él.

Ahora nuestro sistema tendrá dos ecuaciones adicionales, ya que ahora tendremos las dos ecuaciones diferenciales de segundo orden correspondientes a las coordenadas x y z, que deberán transformarse a primer orden como hemos visto anteriormente, por lo tanto, se añaden otras dos ecuaciones. El sistema quedaría de la siguiente forma:

$$\ddot{x} = \dot{V_x} = \frac{1}{m}(Tcos(\theta + \epsilon) - Lsen(\gamma_A) - Dcos(\gamma_A))$$

$$\ddot{z} = \dot{V_z} = \frac{1}{m}(Tsen(\theta + \epsilon) + Lcos(\gamma_A) - Dsen(\gamma_A) - W)$$

$$\begin{matrix} \dot{x} = V_X \\ \ddot{x} = \dot{V_X} \\ \dot{z} = V_Z \\ \ddot{z} = \dot{V_Z} \end{matrix} \qquad \begin{cases} x &= f_1(\dot{x}, \dot{x}, z, \dot{z}, t) \\ \ddot{x} &= f_2(x, \dot{x}, z, \dot{z}, t) \\ \dot{z} &= f_3(x, \dot{x}, z, \dot{z}, t) \\ \ddot{z} &= f_4(x, \dot{x}, z, \dot{z}, t) \end{cases}$$

Al igual que antes, empuje, sustentación y resistencia están en función de V_A. Si obtenemos V_A y γ_A en función de z, x, $\dot{x}$ y $\dot{z}$ tendremos el problema listo para su resolución. Recordando las relaciones de las velocidades anteriormente vistas llegamos a que,

$$\begin{cases} V_{Ax} = \dot{x} + V_{Wx} = V_A \cos(\gamma_A) \\ V_{Az} = \dot{z} + V_{Wz} = V_A sen(\gamma_A) \end{cases}$$

$$\tan(\gamma_A) = \frac{V_{Az}}{V_{Ax}}$$

$$\mathrm{V_A} = \frac{V_{Ax}}{\cos(\gamma_A)}$$

Con esto ya tenemos todo el sistema de ecuaciones listo para implementarlo en Matlab en todas las fases del despegue.

Si recordamos el objetivo 3 del trabajo comentado en la introducción, podemos decir que se alcanza en este capítulo.

Llegados a este punto, recordemos que, hemos analizado la normativa aplicable al despegue y a las actuaciones certificadas. Después hemos definido un modelo dinámico simplificado para poder modelar un despegue de un avión. Ahora lo siguiente será construir un programa que simule nuestro modelo y nos dé como resultado las actuaciones certificadas en despegue de nuestro avión.

4. APLICACIÓN INFORMÁTICA

Una vez tenemos establecidos el modelo dinámico y las actuaciones que queremos calcular, podemos pasar a explicar la aplicación informática objeto del trabajo. Este capítulo correspondería con el objetivo 4 del trabajo, además del 2, ya que en este capítulo también se establecen las restricciones de la normativa.

En este capítulo haremos un recorrido completo por toda la aplicación, explicando cada una de sus ventanas y las opciones que estas ofrecen, sirviendo este capítulo como manual de usuario. A la vez, se explicará cómo se ha implementado el cálculo del modelo y algunos detalles correspondientes a este cálculo como, por ejemplo, modelos propulsivos y aerodinámicos, la forma de definir la pista que comentábamos en el capítulo anterior y algunos más.

Entonces, la estructura a seguir será la siguiente. Primero mostraremos los diagramas de flujo correspondientes a cada parte del programa. A continuación, se mostrarán imágenes de la interfaz de la parte de la aplicación que estemos explicando. Finalmente, explicaremos su implementación en Matlab App Designer [2] y veremos algunos ejemplos creados para ilustrar el funcionamiento de la aplicación.

Al margen de la implementación de la interfaz de usuario, también explicaremos la implementación del cálculo del modelo en las ventanas que tengan dicha función.

Empecemos con la ventana inicial, menú inicial de la aplicación a partir de la cual podemos acceder a todas las demás ventanas.

4.1 Ventana inicial

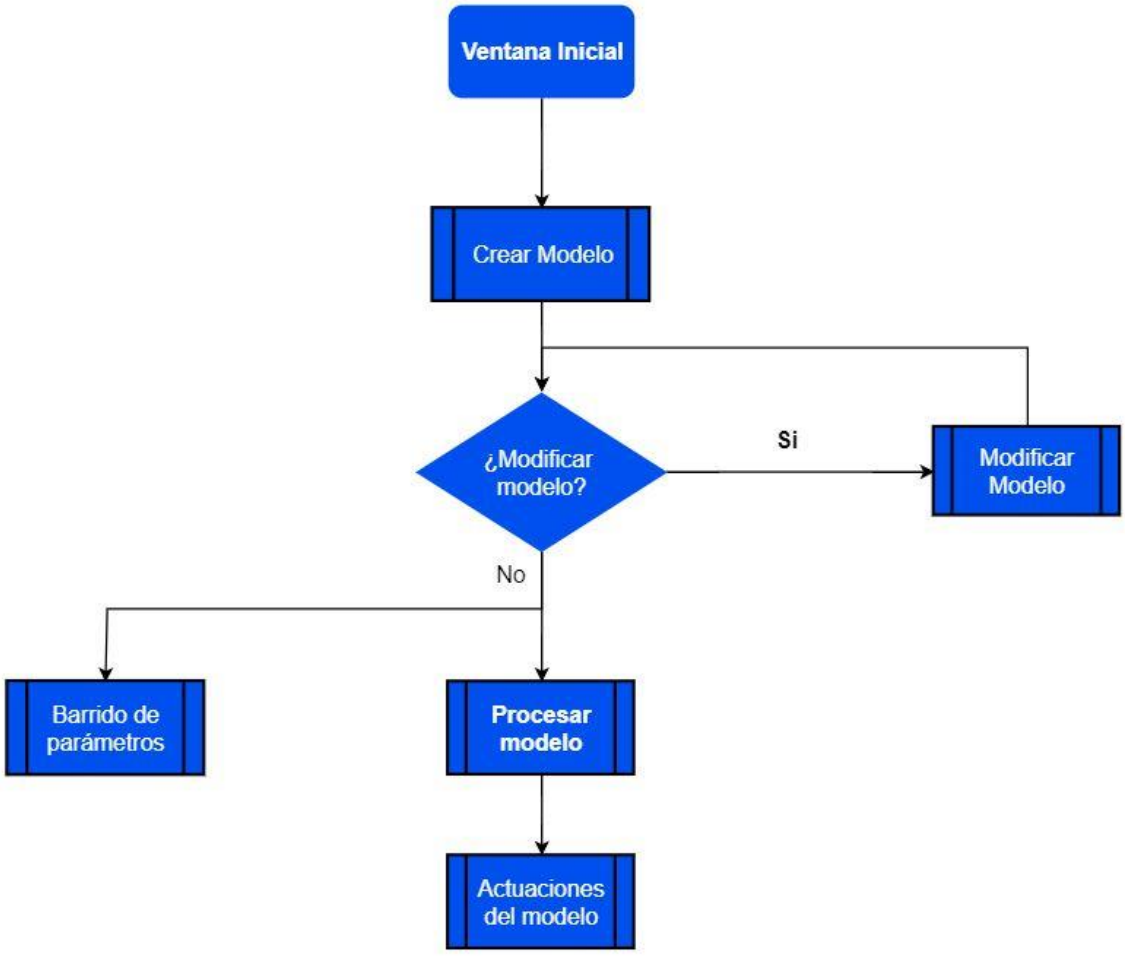


Figura 14. Diagrama de flujo de la ventana inicial

En la ventana inicial de la aplicación se nos ofrecen diversas opciones. Cada una de estas opciones se va a explicar en profundidad en los siguientes puntos de este capítulo. Por ahora, simplemente para explicar un poco esta ventana inicial, se hará una descripción general de los bloques que aparecen ella.

Siguiendo un orden lógico, comenzaríamos por crear un modelo de avión donde se introducen las variables de entrada que nos van a ayudar a resolver el modelo dinámico anteriormente explicado. Una vez creado, el usuario tiene la opción de modificar dicho modelo o de visualizar los modelos existentes.

Una vez tengamos nuestro modelo, este se procesaría. En esta opción es donde se producen todos los cálculos relacionados con el modelo dinámico y también con las actuaciones certificadas, guardándose los datos en un archivo al finalizar el proceso. Acto seguido, para visualizar los resultados usaríamos la opción de actuaciones del modelo, donde aparte de visualizar los resultados, existen diversas opciones.

Por último, la ventana de barrido de parámetros ofrece la posibilidad de estudiar las actuaciones

certificadas variando un parámetro del modelo en un cierto intervalo. Es un modelo independiente donde se producen los cálculos y se muestran los resultados directamente, no guardando ningún archivo de resultados, por lo que se puede utilizar independientemente de si hemos procesado el modelo o no.

Figura 15. Ventana Inicial de Actuaciones en Despegue

4.2 Ventana Crear Modelo

En esta ventana vamos a definir los parámetros que nos serán necesarios en la resolución de nuestro modelo. Algunos de los parámetros que vamos a ver en esta sección están relacionados con los que vimos en el capítulo 3 y otros serán nuevos debido a la introducción de nuevas ecuaciones como, por ejemplo, la del modelo propulsivo o el aerodinámico.

Todos estos modelos se explicarán en sus correspondientes apartados. Pero antes de eso, el diagrama de bloques sería el siguiente.

Figura 16. Diagrama de flujo de la ventana crear modelo

Como podemos ver, en esta ventana simplemente se van a recoger los parámetros que nos van a servir como variables de entrada para resolver nuestro modelo o para comprobar que la normativa se cumple, como el caso de las velocidades.

Estos parámetros están divididos por bloques, que pasaremos a explicar a continuación.

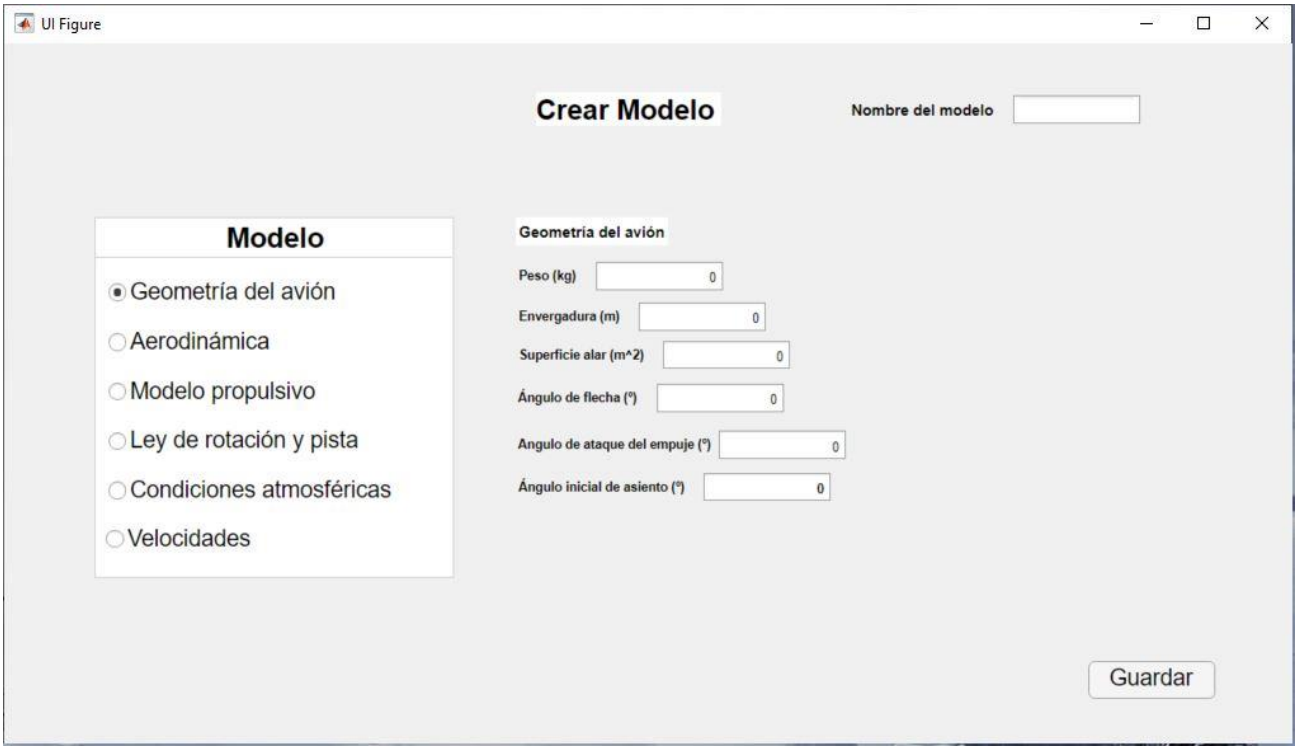


Figura 17. Ventana crear modelo, bloque de geometría del avión

4.2.1 Modelo propulsivo

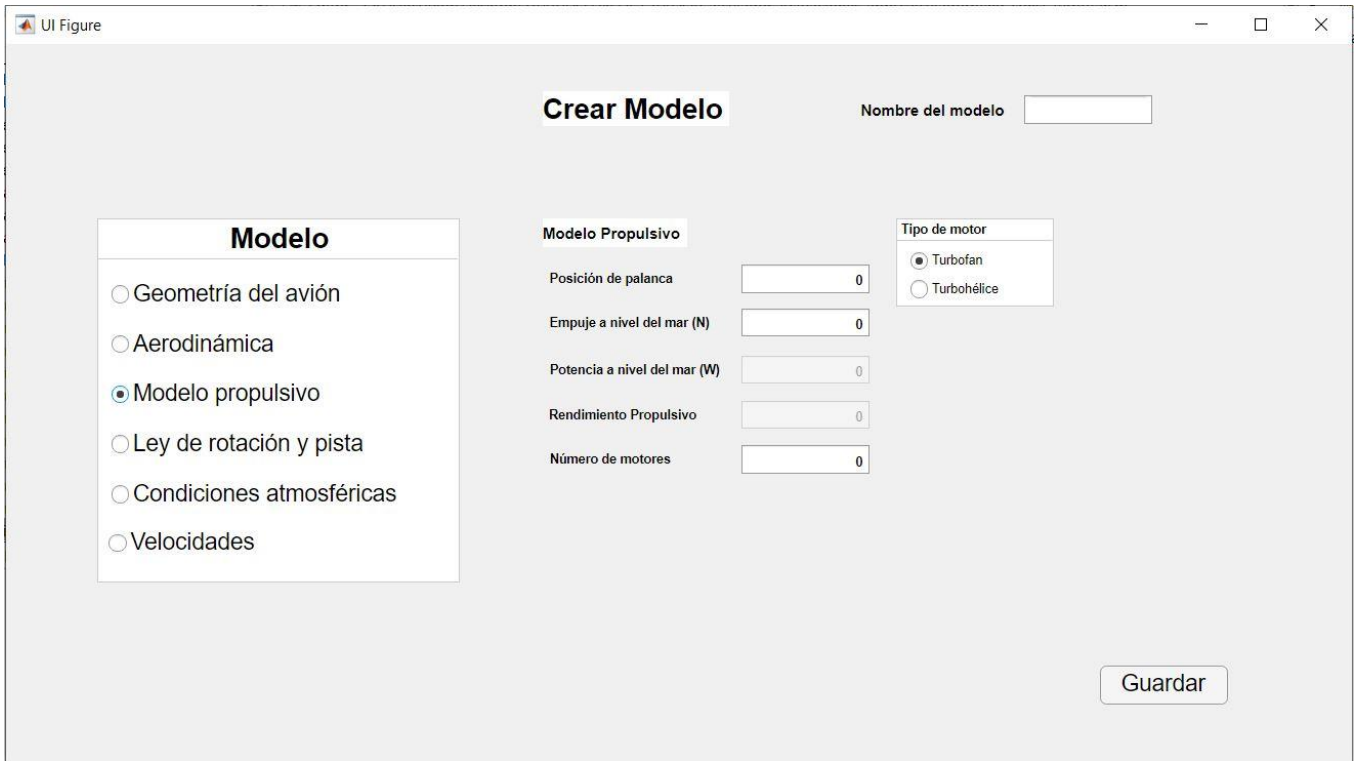


Figura 18. Ventana crear modelo, bloque de modelo propulsivo

Como ya hemos comentado, se buscaba un modelo propulsivo que dependiese de la velocidad para poder integrar el modelo. Por lo tanto, se optó por escoger un modelo propulsivo sencillo, que no dependiese de demasiados parámetros, los fundamentales que necesitaba nuestro modelo.

Es cierto que existen modelos propulsivos mucho más completos y precisos que el escogido, pero se optó por uno más sencillo ya que hay que recordar que este trabajo está centrado en las actuaciones certificadas y la normativa, no tanto en ser estrictamente precisos con la propulsión o de crear un modelo lo más completo y preciso posible.

Se dan dos opciones de motor, turbofán o turbohélice. Cada una de ellas definirá el empuje de una forma distinta. Estos modelos propulsivos se han obtenido de [7].

Turbofán: $T = \delta_T T_{SL}\left(1+\frac{\gamma-1}{2}M^2\right)^{\frac{\gamma}{\gamma-1}}\left(1{,}00-0{,}49\sqrt{M}\right)\frac{\rho}{\rho_{SL}}$ En Newtons

Turbohélice: $P = \delta_T P_{SL}\left(1+\frac{\gamma-1}{2}M^2\right)^{\frac{\gamma}{\gamma-1}}\frac{p}{p_{SL}}$ En Watios

$$T = \frac{P}{V}\,\eta_P \quad \text{En Newtons}$$

Donde,

- δ_T: Posición palanca de gases en porcentaje
- T_{SL}: Empuje a nivel del mar
- γ: Coeficiente de dilatación adiabática
- M: Mach de vuelo
- ρ: Densidad
- ρ_{SL}: Densidad a nivel del mar
- P: Potencia
- P_{SL}: Potencia a nivel del mar
- p: Presión
- p_{SL}: Presión a nivel del mar
- T: Empuje
- V: Velocidad de vuelo
- η_P: Rendimiento propulsivo

4.2.2 Modelo aerodinámico

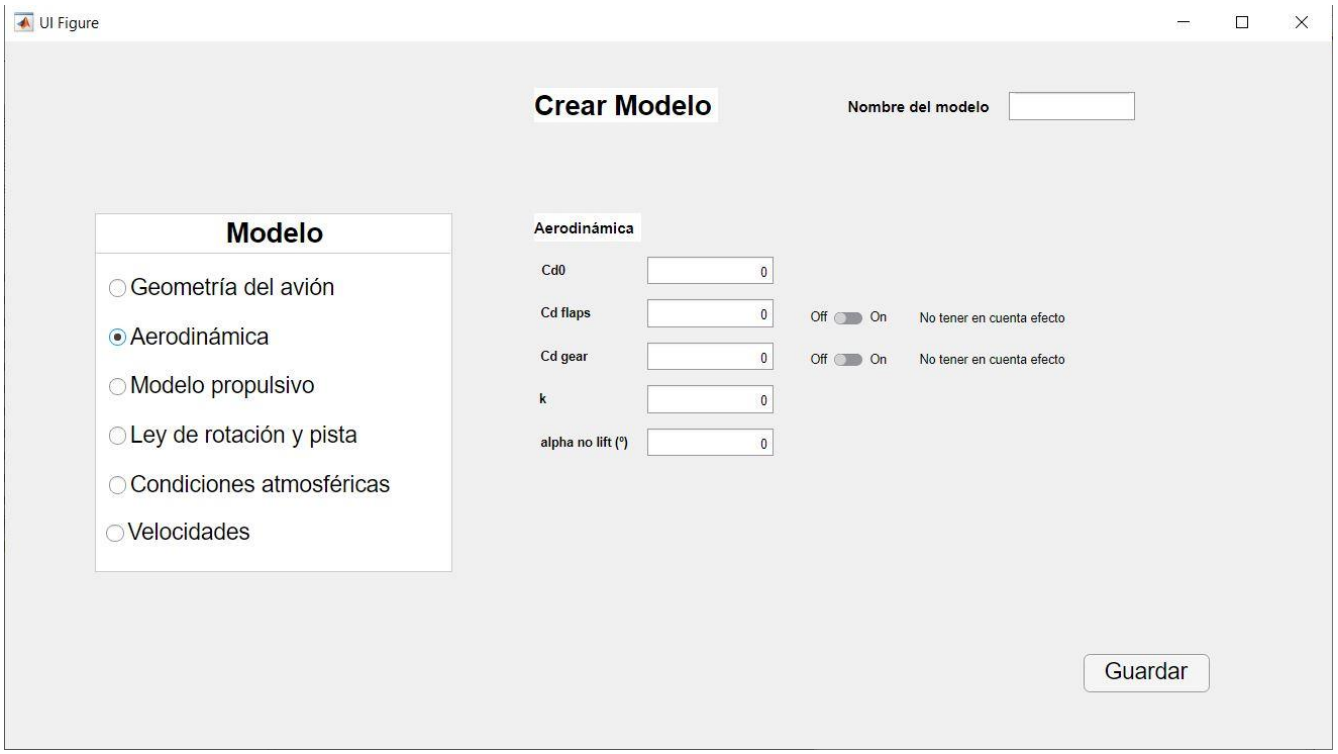


Figura 19. Ventana crear modelo, bloque de modelo aerodinámico

Este modelo ha sido obtenido del libro [8]. Como en el modelo propulsivo, se buscaba un modelo lo más sencillo posible pero que cumpliera ciertas condiciones.

La más importante sería que dependiese de la velocidad, y también, para dar mayor precisión, que tuviese en cuenta los efectos que producen los flaps y el tren de aterrizaje en la aerodinámica a la hora del despegue. Por lo tanto, se optó por este modelo, ya que cumple con dichos requisitos.

$$\frac{dC_L}{d\alpha} = \frac{2\pi AR}{2+\sqrt{4+AR^2\beta^2\left(1+\frac{\tan^2\Delta}{\beta^2}\right)}} \qquad AR = \frac{b^2}{S_{alar}}$$

$$C_L = \frac{dC_L}{d\alpha}(\alpha - \alpha_{0L}) \qquad \beta = \sqrt{1-M^2}$$

$$C_D = C_{D0} + \Delta C_{Dflap} + \Delta C_{Dgear} + KC_L^2$$

$$D = \frac{1}{2}\rho V_A^2 S_{alar} C_D \quad \text{En Newtons}$$

$$L = \frac{1}{2}\rho V_A^2 S_{alar} C_L \quad \text{En Newtons}$$

Donde,

- M: Mach de vuelo
- C_L: Coeficiente de sustentación
- C_D: Coeficiente de resistencia
- D: Resistencia aerodinámica
- L: Sustentación
- AR: Alargamiento
- b: Envergadura
- S_{alar}: Superficie alar
- Δ: Ángulo de flecha
- α: Ángulo de ataque
- α_{0L}: Ángulo de ataque para sustentación nula
- C_{D0}: Coeficiente de resistencia sin sustentación
- ΔC_{Dflap}: Coeficiente de resistencia de flaps
- ΔC_{Dgear}: Coeficiente de resistencia del tren de aterrizaje
- K: parámetro de resistencia inducida unitario
- V_A: Velocidad con respecto a la corriente

Destacar que los efectos de los flaps en el coeficiente de sustentación se han tenido en cuenta a través del ángulo de ataque para sustentación nula, por lo que dicho ángulo normalmente será negativo, para que de esta forma el coeficiente aumente para un mismo ángulo de ataque.

4.2.2.1 Efecto suelo

Dentro del modelo aerodinámico hay que comentar el efecto suelo ya que, si el avión se encuentra dentro del área de influencia de este efecto, las fuerzas aerodinámicas se verán modificadas.

Sin profundizar demasiado en los fundamentos físicos por los cuales se produce este efecto, podemos decir que cuando el avión vuela muy próximo al suelo, este interfiere con la corriente alrededor del ala, modificando la distribución de líneas de corriente. La proximidad del suelo provoca que se reduzcan las deflexiones verticales de la corriente tanto en el borde de ataque como en el borde de salida.

Esto provoca una reducción del ángulo de ataque inducido y por lo tanto una reducción de la resistencia inducida, que reducirá a su vez la resistencia total del avión. Aparte, la disminución del ángulo de ataque inducida también provoca un aumento de la sustentación.

Para analizar este efecto y cómo influye en la aerodinámica, volveremos a recurrir al libro [8].

La influencia del efecto suelo es función principalmente de la distancia del avión al suelo y del tamaño del ala. Así pues, este efecto puede tratarse como un incremente en el alargamiento efecto del avión AR. En la siguiente gráfica podemos ver cuál sería la relación entre el alargamiento real de la aeronave y el alargamiento efectivo con el efecto suelo AR_{eff} en función de la distancia del suelo al avión h y de la envergadura del avión b.

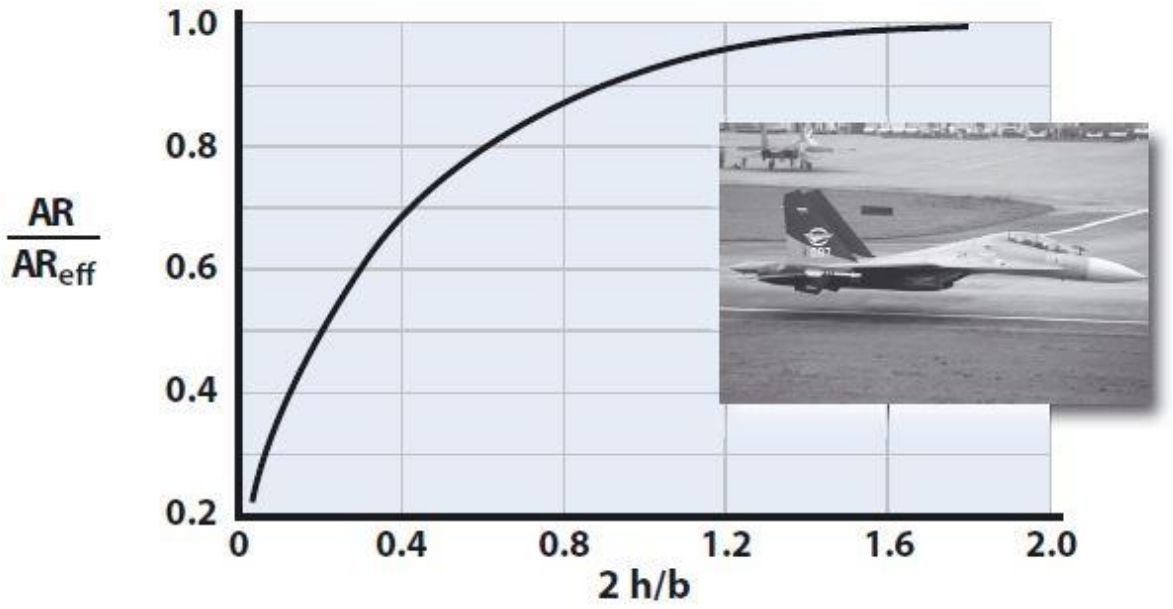


Figura 20. Gráfica de la magnitud del efecto suelo en función de la altitud [8].

Como podemos apreciar en esta gráfica, el efecto suelo es bastante considerable en la zona en la que la distancia entre el suelo y el avión, h, sea menor que la semienvergadura del mismo.

Teniendo en cuenta que este trabajo está dirigido principalmente a aviones de un tamaño medio o grande, una envergadura media razonable en la que pensar es de 40 metros. Además, solo se analiza el despegue hasta el punto en el que el avión alcanza los 35 pies (unos 11 metros), por lo que h será menor o igual a 11 metros (tomamos una media de 6 metros).

Entonces, con los valores aproximados que hemos comentado,

$$2\frac{h}{b} = 0.3$$

Y observando la gráfica vemos que más o menos,

$$\frac{AR}{AR_{eff}} = 0.6$$

E introduciendo el nuevo alargamiento efectivo en nuestro modelo aerodinámico anterior para tener cuantificado el efecto suelo durante el despegue, de forma aproximada.

Existen modelos más detallados respecto al efecto suelo, pero en lugar de añadir más dificultad al programa, se ha optado por usar este sencillo modelo de forma que al menos se tenga en cuenta este efecto, aunque sea de manera aproximada.

4.2.3 Ley de rotación

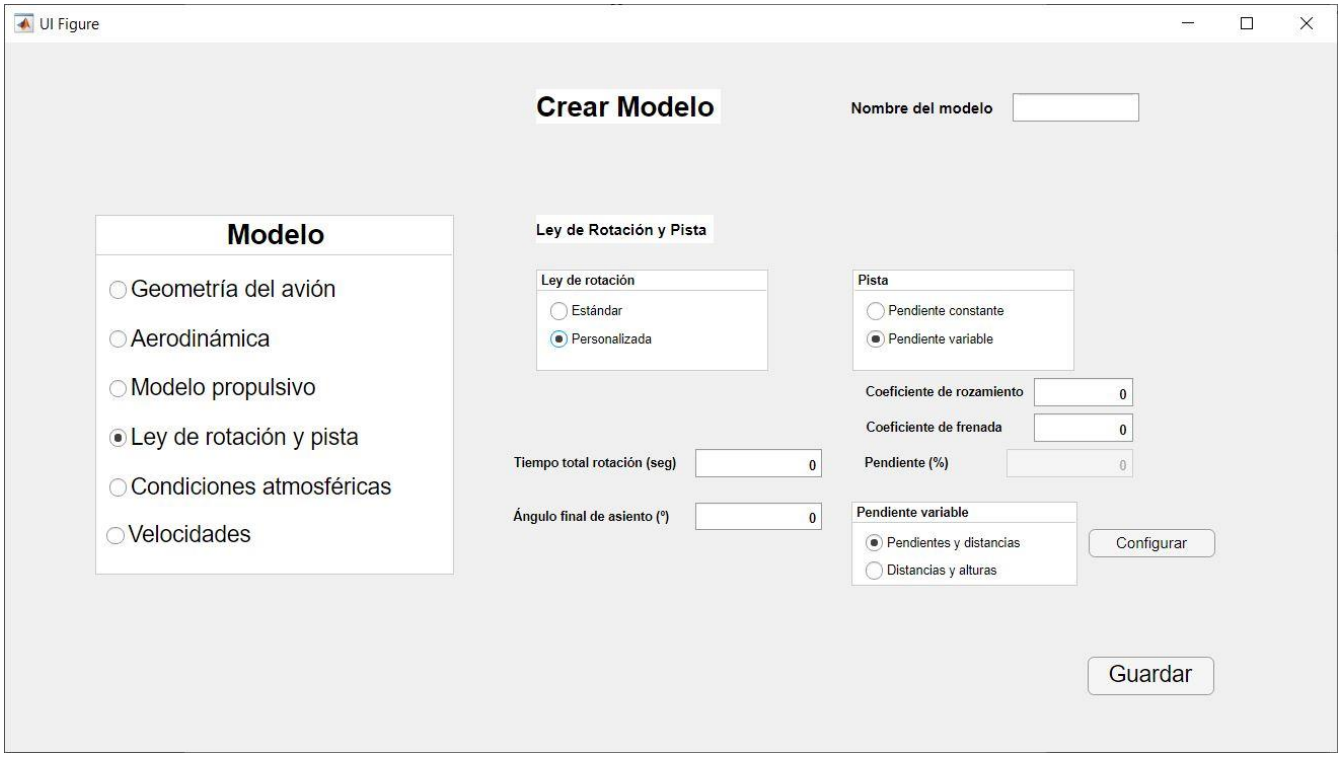


Figura 21. Ventana crear modelo, bloque de ley de rotación y pista

Se utiliza una ley de rotación lineal

$$\theta = \theta_0 + At$$

Si se selecciona una ley de rotación estándar solo se introducirá un ángulo de rotación inicial, y se utilizará un ángulo de rotación final y un tiempo de rotación estándar que serán fijos en el programa.

$$\theta_{final} = 12º$$

$$t_{final} = 3{,}5\ segundos$$

Si se selecciona una ley de rotación personalizada se introducen todos los datos de manera manual.

4.2.4 Pista

Se busca saber el perfil longitudinal de la pista. Esta puede ser de pendiente constante o variable.

Si es constante, se pedirán los datos de la pendiente en tanto por ciento, el coeficiente de rozamiento y el coeficiente de rozamiento en caso de frenada. El dato de la pendiente luego se pasará a grados y la pista quedará definida por una altura inicial y este ángulo de pendiente.

La pendiente constante está limitada por normativa en la CS 25, AMC 25.1581 Aeroplane Flight Manual punto 6.b.3.i.H ([3]). Se indica con un pequeño letrero en la ventana.

Si se selecciona variable, se dan dos opciones para introducir los datos de la pista, o bien pendientes y distancias, o distancias y alturas.

La primera de las opciones se refiere a dar la pendiente en tanto por ciento y la distancia que está con esa pendiente, por ejemplo, 500 metros con una pendiente de 0,1%. Con estos datos después se obtienen una serie de puntos que nos servirán para conocer el perfil de la pista. La forma de obtener esta serie de puntos se explicará más adelante, ahora lo importante es saber que se guardan estos datos en forma de vector.

La segunda opción es dar alturas y la distancia a la que están esas alturas, por lo tanto, en esta opción los datos son directamente los puntos x y z del perfil longitudinal de la pista. Por ejemplo, a 500 metros la altura es 10 metros, a 1500 metros la altura es 50 metros.

Cabe destacar que la opción de pista variable no es lo habitual, ya que en los manuales de fabricante se suele considerar pendiente constante por defecto. Sin embargo, se propuso como reto adicional implementar esta opción, ya que, desde el punto de vista del análisis para la ingeniería, esta opción resultaba bastante más interesante que la opción de pendiente constante.

Finalmente, lo datos que obtenemos van a ser una serie de puntos que se almacenarán en una matriz, en la cual una columna tendrá con las coordenadas x y otra columna las coordenadas z. La obtención de dichos puntos se explicará en los siguientes apartados de este capítulo.

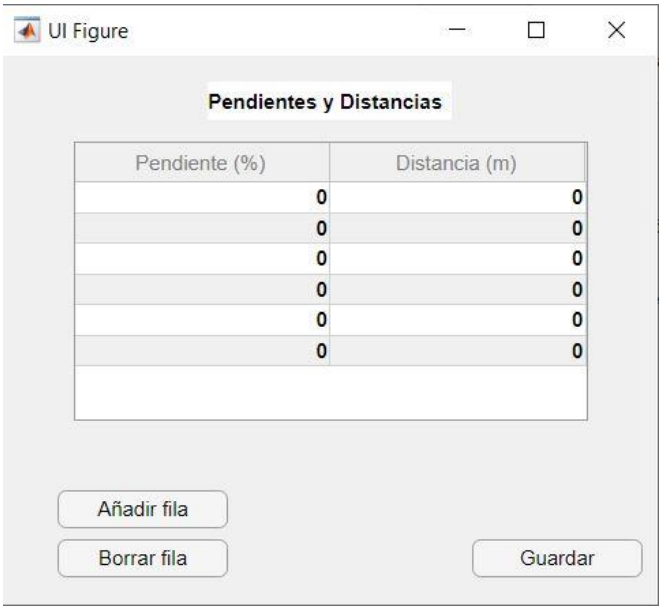


Figura 22. Pendientes y Distancias

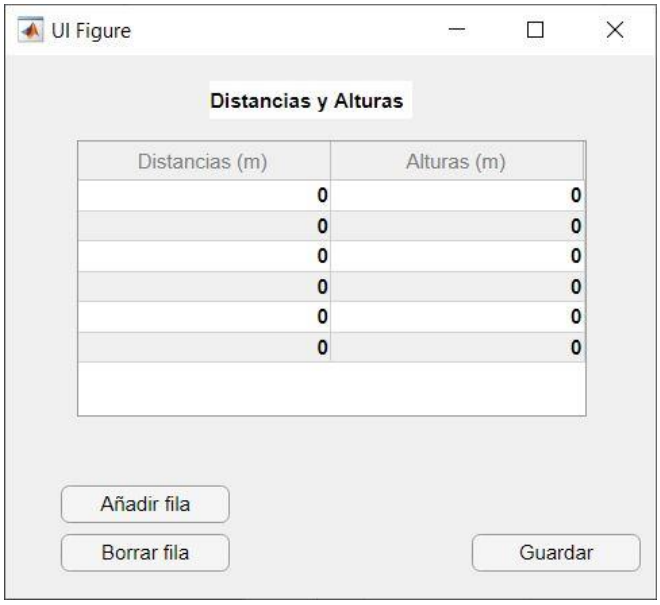


Figura 23. Distancias y alturas

En el siguiente diagrama podemos ver, a modo de resumen, las opciones que permiten estos dos bloques que hemos comentado, el de ley de rotación y el de pista.

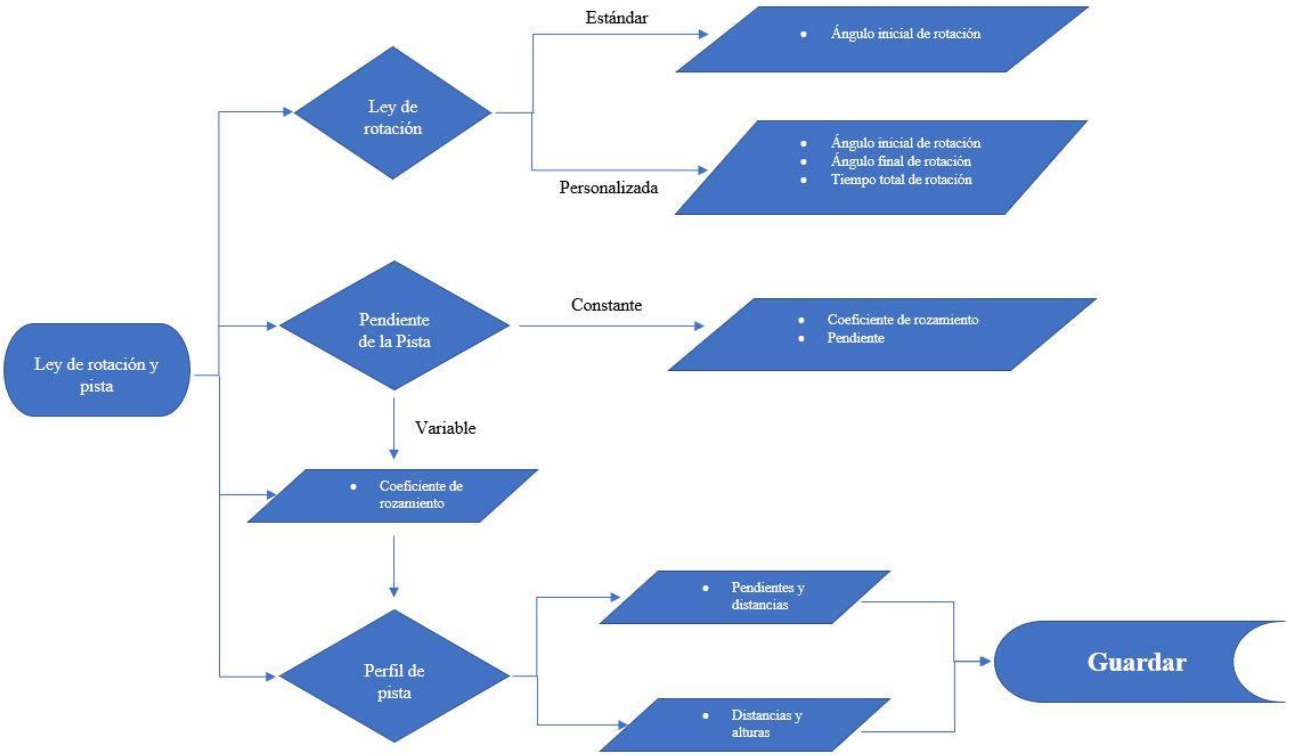


Figura 24. Diagrama de flujo del bloque Ley de rotación y pista

4.2.5 Condiciones atmosféricas

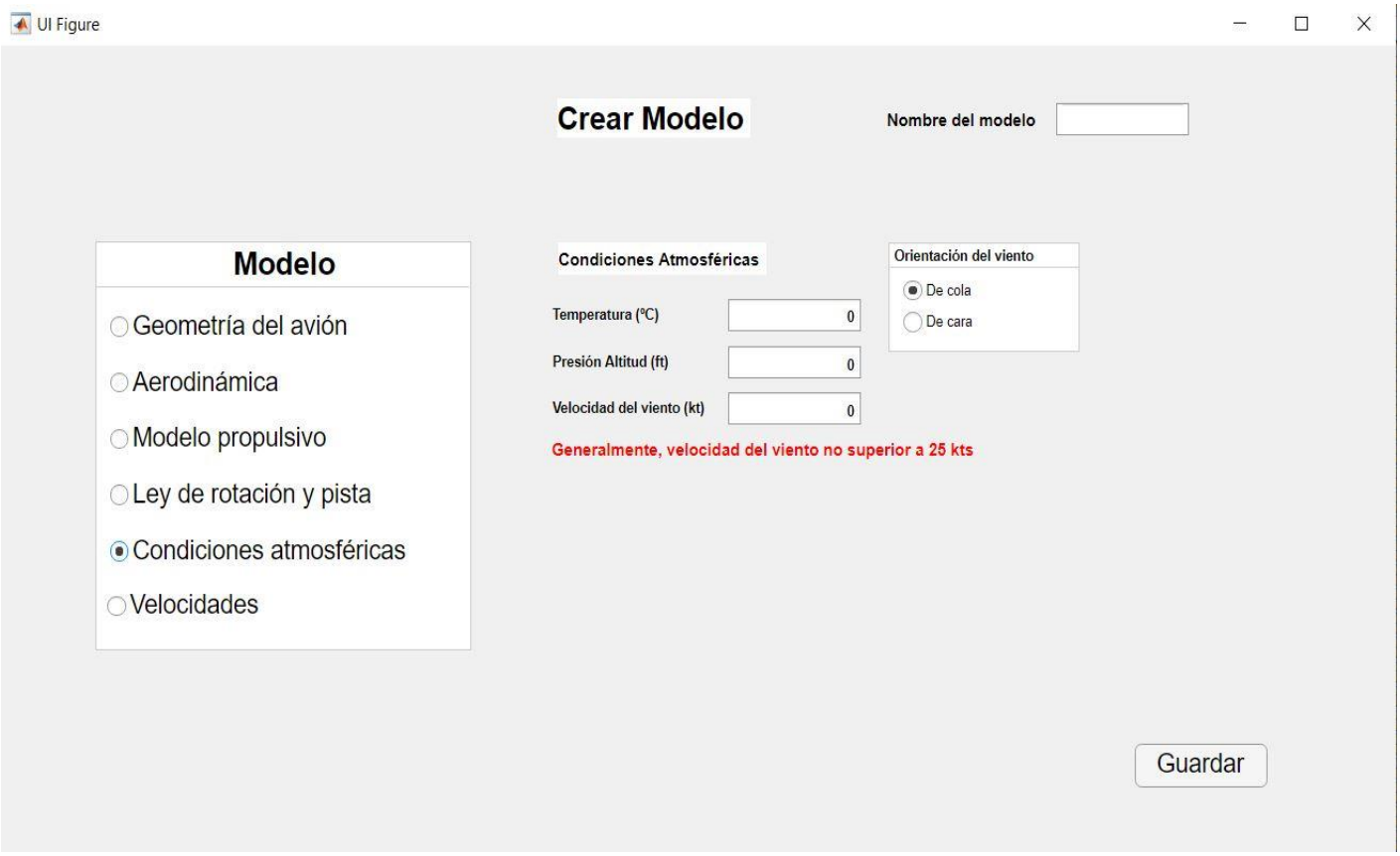


Figura 25. Ventana crear modelo, bloque de condiciones atmosféricas

Con una presión altitud y una temperatura específica podremos calcular tanto la densidad como la velocidad del sonido específicas para unas condiciones atmosféricas cualesquiera.

$$\rho = \frac{P}{R_G T} \qquad a = \sqrt{\gamma R_G T}$$

Donde,

- R_G: Constante de los gases ideales
- γ: Coeficiente de dilatación adiabática

De este modo, podremos ver y estudiar la influencia que tienen estos datos sobre las actuaciones durante el despegue y sobre el rendimiento de las mismas. También, de esta forma podemos conseguir un poco más de precisión que simplemente usando la atmósfera ISA.

Mencionar brevemente que la velocidad del viento, por normativa en el punto 25.237 tanto de [3] como de [4], suele ser menor que 25 nudos. Se indica en la ventana con un pequeño letrero.

4.2.6 Velocidades

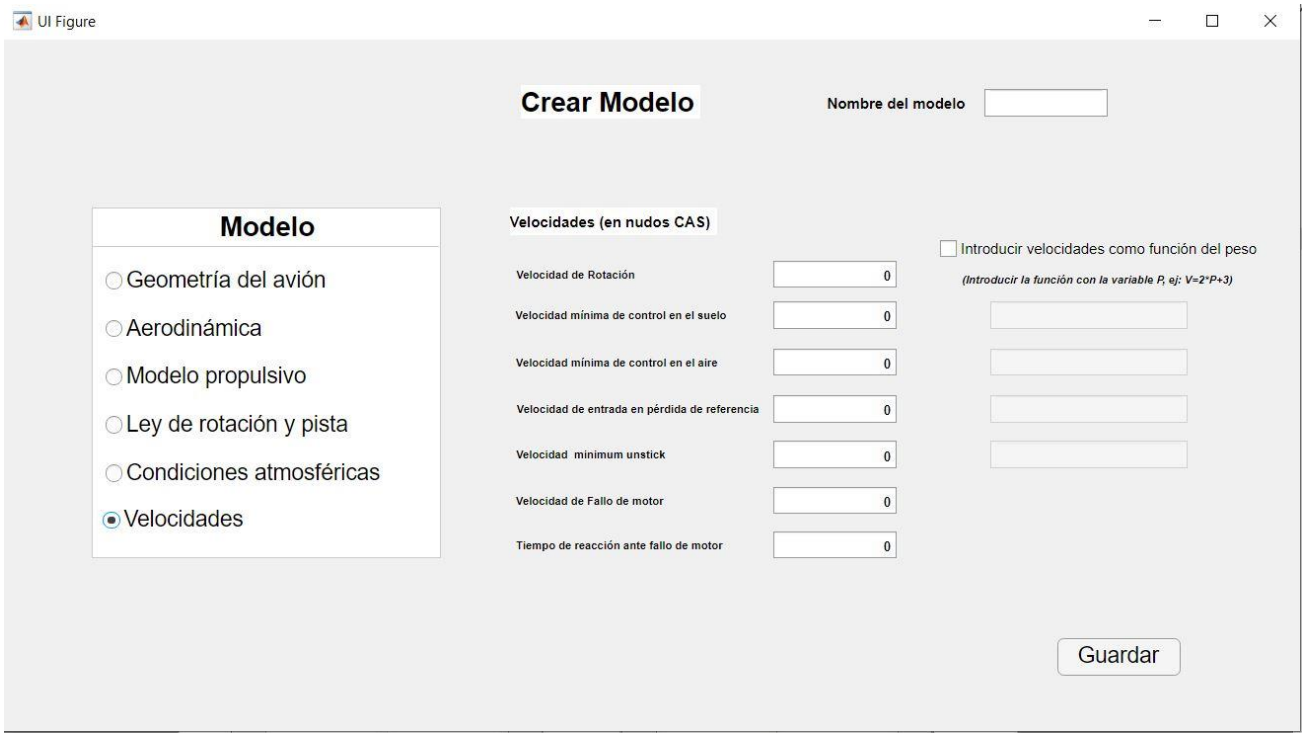


Figura 26. Ventana crear modelo, bloque de velocidades

Como ya vimos en el capítulo 2, la norma impone una serie de requisitos sobre las actuaciones del despegue.

Estos requisitos afectan tanto a las velocidades como a las distancias que calculamos, por lo que va a ser necesario conocer ciertas velocidades, que normalmente proporciona el fabricante de la aeronave, para poder calcular las actuaciones certificadas del despegue y comprobar que se cumplen las restricciones.

Estas velocidades normalmente son función del peso del avión, por lo que se incluye la opción de poder introducir una función en vez de un valor numérico para ciertas de estas velocidades. Esta función debe tener como variable independiente el peso, que se indicará como P.

4.2.6.1 Diferentes velocidades y anemómetro ideal

Es necesario hablar brevemente de las distintas velocidades que se consideran en la aeronáutica, ya que afectan a este trabajo. A continuación, un breve comentario de cada una de ellas (obtenido de [5]).

- IAS Velocidad Indicada: Velocidad que indica el anemómetro en el avión.

- CAS Velocidad Calibrada: Es igual a la IAS, pero después de corregir errores de instrumentación y posición.

- EAS Velocidad Equivalente: Es igual a la CAS, pero después de corregir los efectos de la compresibilidad adiabática de la corriente a la altitud considerada.

- TAS Velocidad Verdadera: Es la velocidad de la aeronave con respecto a la corriente de aire.

Se suele hablar de estas velocidades en nudos CAS, ya que, tal y como comentábamos en el capítulo 2, toda la normativa habla respecto a las velocidades en términos de CAS. Por ello, en el programa se considera que las velocidades van a ser introducidas en CAS, para posteriormente pasarlas a TAS si es necesario, pues el programa realmente va a funcionar y va a integrar con las velocidades en términos de TAS.

Ahora bien, para pasar de CAS a TAS y viceversa, vamos a considerar un anemómetro aeronáutico.

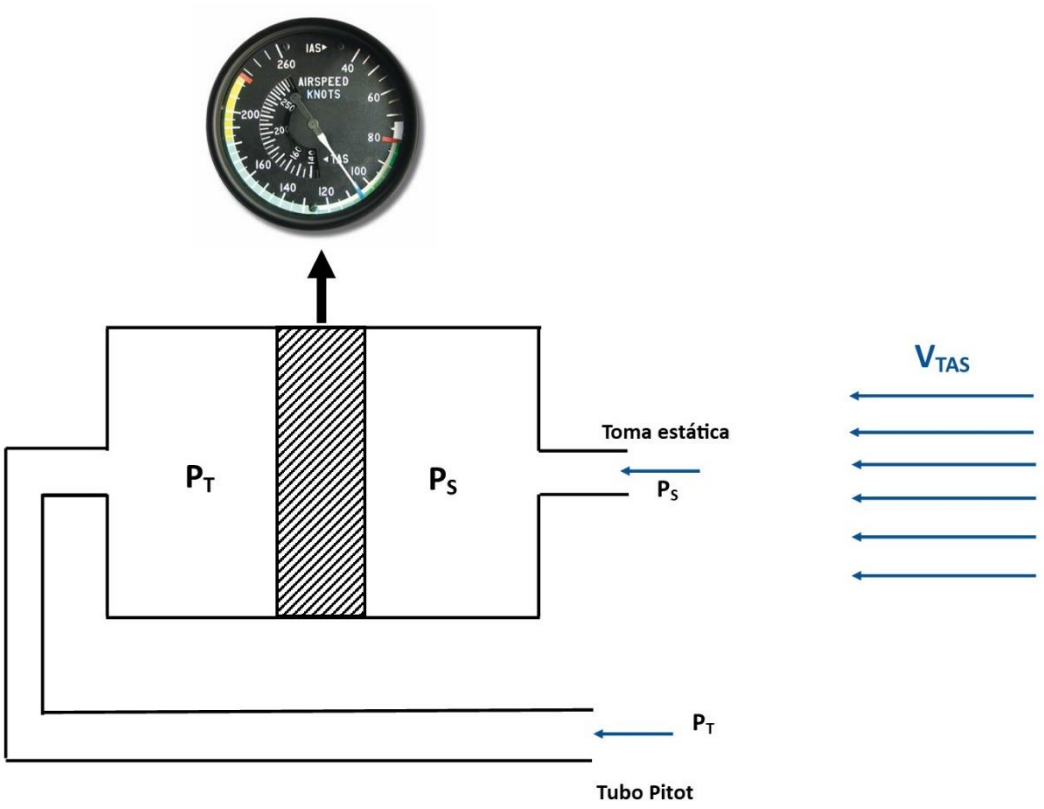


Figura 27. Esquema de anemómetro aeronáutico

No se van a considerar errores en la instrumentación ni posición, por lo que velocidades CAS e IAS van a ser iguales. También se va a considerar un proceso isentrópico en el tubo de Pitot.

Por lo tanto,

$$V_{IAS} = V_{CAS}$$

Proceso isentrópico: $P_T - P_S = P_S f(M_{TAS})$ $M_{TAS} = \frac{V_{TAS}}{a_T}$

Anemómetro: $P_T - P_S = P_0 f(M_{CAS})$ $M_{CAS} = \frac{V_{CAS}}{a_0}$

$$f(M) = -1 + \left(1 + \frac{\gamma - 1}{2} M^2\right)^{\frac{\gamma}{\gamma - 1}}$$

Donde,

- M_{CAS}: Mach de vuelo respectivo a CAS
- M_{TAS}: Mach de vuelo respectivo a TAS
- V_{TAS}: Velocidad TAS
- V_{CAS}: Velocidad CAS
- a_T: Velocidad del sonido respectiva a la altura de vuelo
- a_0: Velocidad del sonido respectiva al nivel del mar
- P_T: Presión de remanso
- P_S: Presión estática
- P_0: Presión a nivel del mar
- γ: Coeficiente de dilatación adiabática

Con este sencillo modelo somos capaces de pasar de una velocidad calibrada a una velocidad verdadera y viceversa simplemente igualando las dos ecuaciones que tenemos arriba.

Entonces, obtenemos las velocidades como datos en términos de CAS. Las velocidades que necesitemos para los cálculos del modelo se transformarán a términos de TAS, ya que toda la integración del modelo se produce en velocidad verdadera, ya que es la velocidad con la que estamos trabajando en el modelo.

Las velocidades que obtengamos como resultados las obtendremos de igual forma en términos de TAS, y para poder comprobar los requisitos que exige la normativa, será necesario volver a transformarlas a términos de CAS.

4.2.7 Guardar

Una vez pulsemos el botón de guardar, el programa creará un fichero ".mat" con el nombre del modelo que hayamos elegido y guardará cada uno de los valores que hemos introducido en dicho fichero.

De esta forma se pueden consultar fácilmente los datos e incluso modificarlos desde Matlab, aunque como veremos a continuación se pueden modificar dichos datos a través de la opción de modificar modelo que veremos a continuación.

A continuación, se muestra como quedarían guardados los datos en el archivo. mat

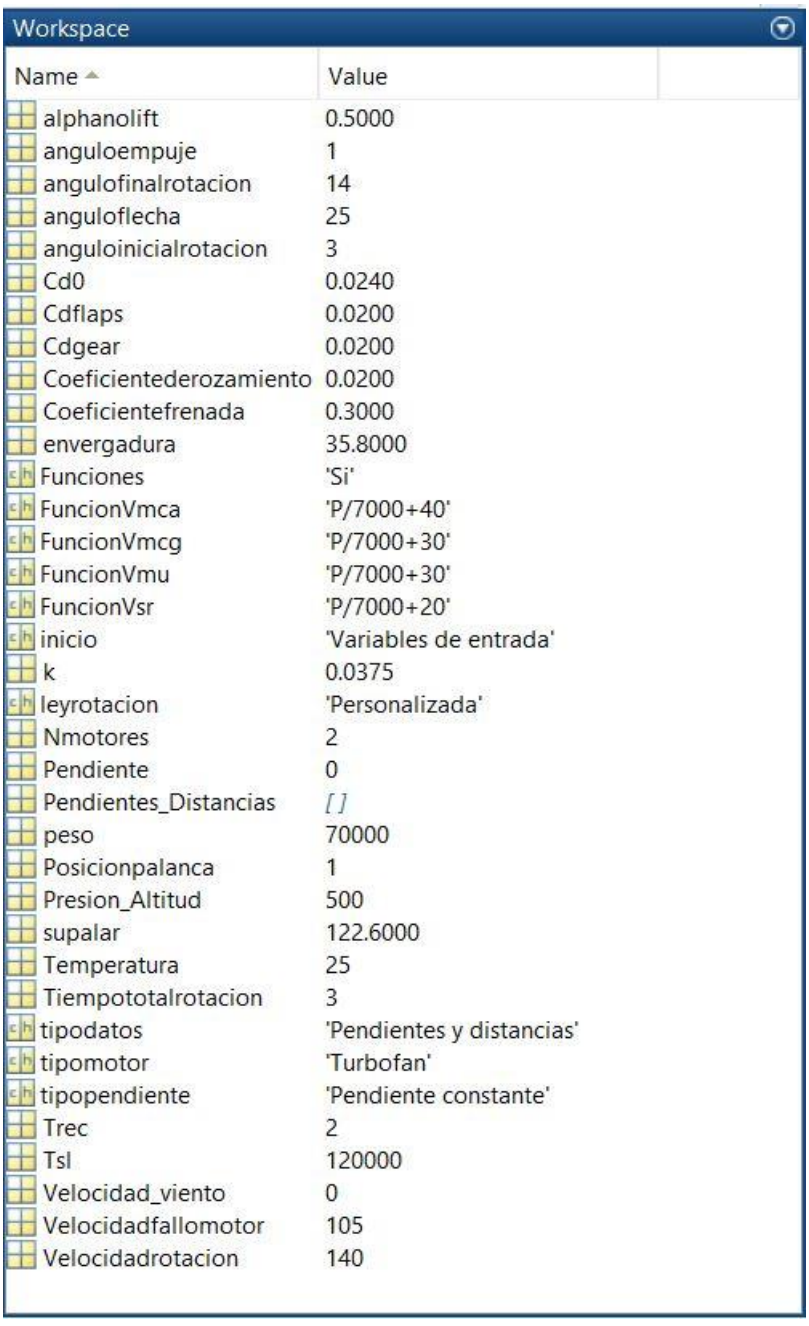


Figura 28. Archivo con modelo guardado tal y como se vería en el Workspace de Matlab

4.3 Ventana Modificar Modelo

Esta sección abre una pestaña que pide el nombre del modelo a modificar. Es importante introducir el nombre del modelo correctamente, por ejemplo, si tu archivo se llama “ejemplo.mat” debes introducir “ejemplo”.

Más adelante se explicará que en la pestaña de inicio hay una opción para mostrar todos los modelos que se tengan guardados, para poder facilitar la correcta introducción de los nombres, ya que para procesar los datos o los resultados también será necesario introducir el nombre del archivo.

Una vez introducimos el nombre del modelo a modificar saldrá una pantalla exactamente igual a la de crear modelo, pero con los datos del modelo seleccionado.

Dentro de esta pantalla podemos modificar los datos que queramos o simplemente consultarlos si es que se elige alguno de los modelos que vienen incluidos en el programa.

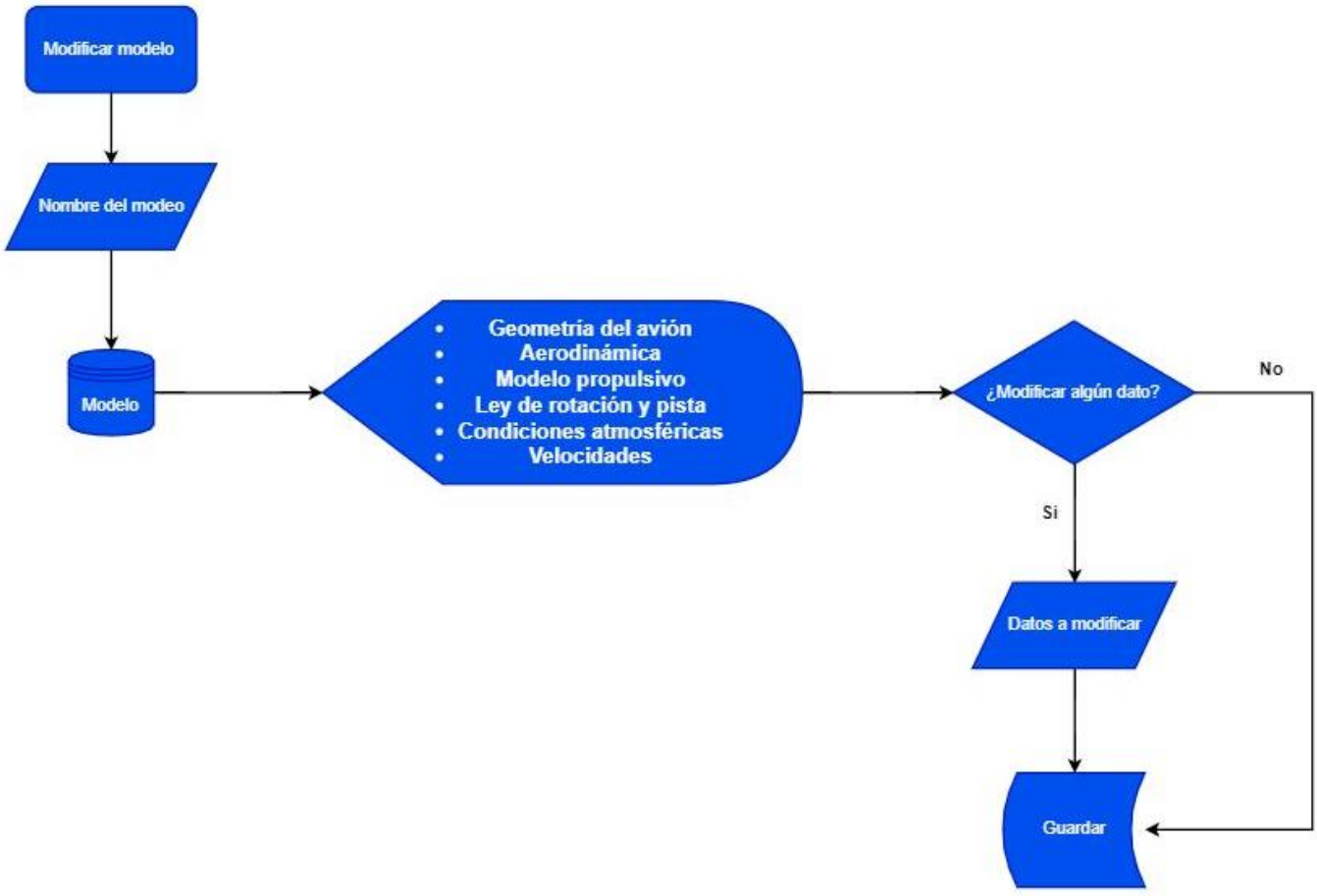


Figura 29. Diagrama de flujo de la ventana modificar modelo

Figura 30. Modificar modelo, bloque de aerodinámica

Figura 31. Modificar modelo, bloque de ley de rotación y pista

Es importante recalcar que, en el apartado de pendiente variable, si le damos al botón de configurar para abrir la ventana donde rellenar los datos del perfil de la pista, es necesario darle al botón de guardar en esa pestaña, aunque no cambiemos nada. Si no lo hacemos se guardará una matriz de ceros en vez de la que ya estaba.

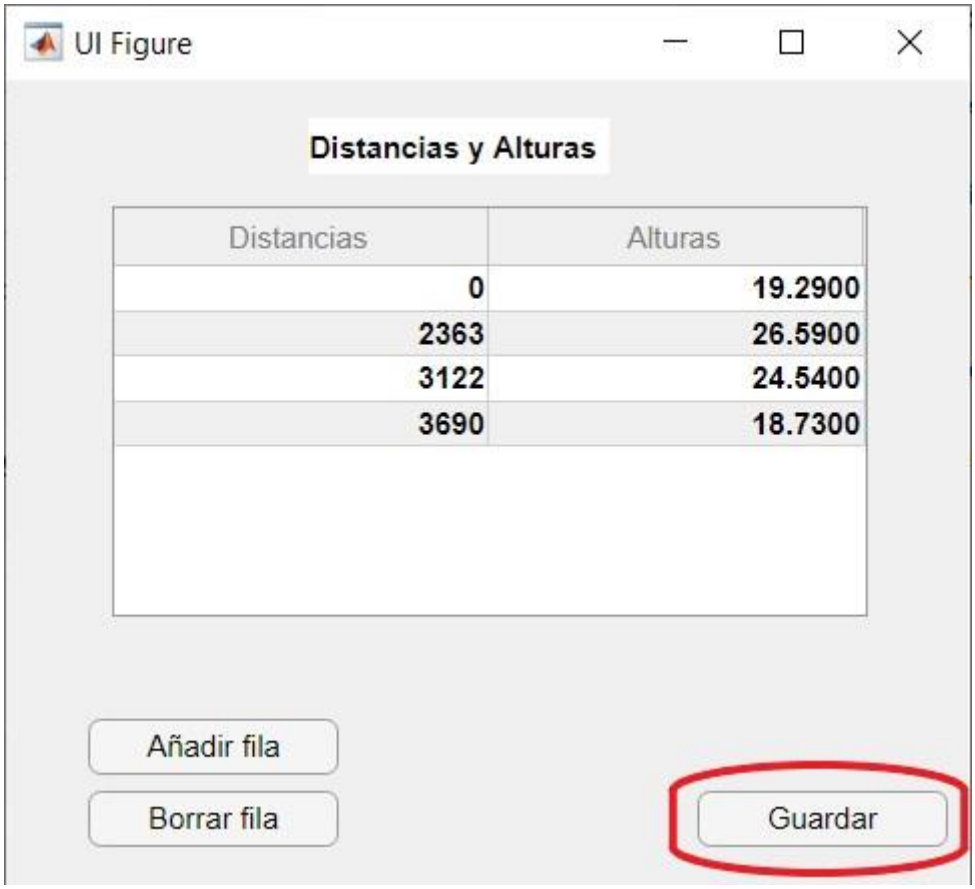


Figura 32. Es necesario guardar los datos de la pendiente variable siempre

4.4 Ventana Procesar Modelo

Este bloque del programa es el más importante de todos. Es donde tienen lugar todos los cálculos que se han ido introduciendo durante los capítulos anteriores.

Para ello, utilizando los datos del modelo que hemos introducido, y resolviendo numéricamente el modelo dinámico de anteriores capítulos, obtenemos los resultados deseados.

A continuación, se explicará cómo se ha implementado el modelo dinámico y su resolución. Recordemos que ya hemos explicado la solución analítica de nuestro modelo, pero todavía nos falta por saber cómo nuestro programa va a resolver numéricamente el problema. Todo esto se irá viendo poco a poco a la vez que se explica la estructuración de este bloque y como se dividen los cálculos según escojamos ciertas opciones que permite el programa u otras.

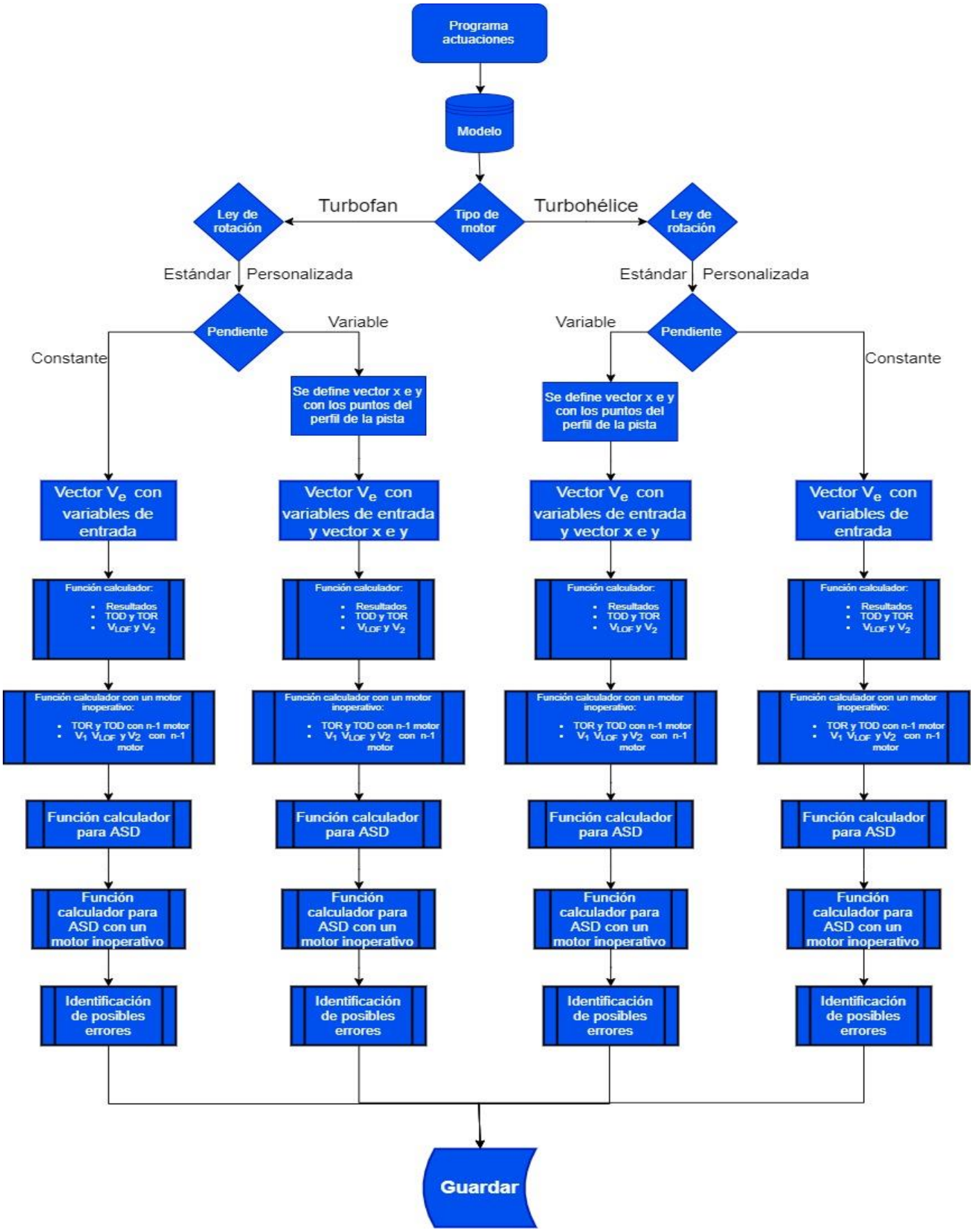


Figura 33. Diagrama de flujo de programa actuaciones

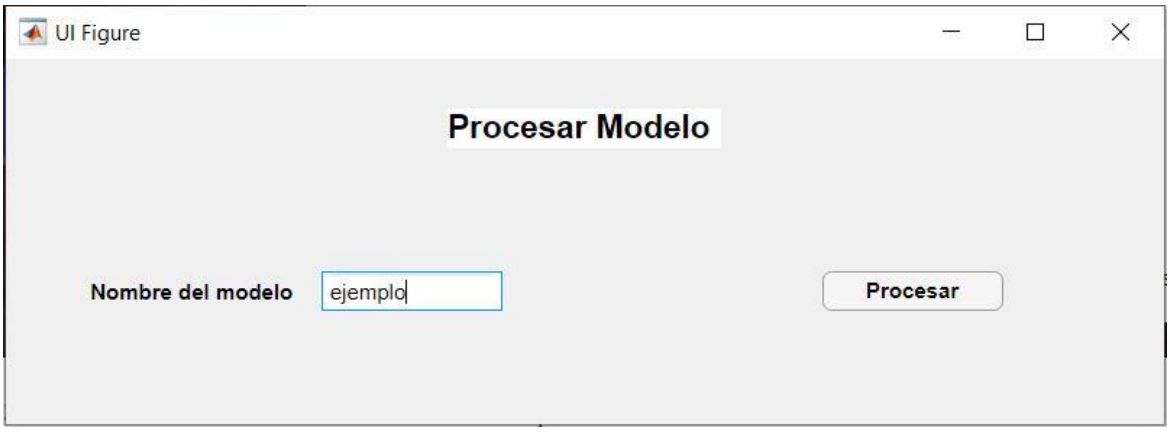


Figura 34. Ventana Procesar Modelo

La ventana principal de este bloque es muy sencilla, ya que en este bloque solo se producen los cálculos. La visualización de los mismos ocurrirá en el siguiente.

Entonces, introducimos el modelo del cual queremos calcular las actuaciones. El programa carga el modelo que anteriormente hemos creado y llama al archivo "programa_actuaciones".

Lo primero que ocurrirá en este archivo será la creación de un vector de variables de entrada con los datos que hemos introducido cuando hemos creado nuestro modelo. Como vimos en la ventana crear modelo, existen diferentes opciones para crear nuestro modelo. Dependiendo de estas opciones, nuestro modelo cambia en algunas ecuaciones, por lo que es necesario diferenciar dichas opciones desde el principio.

Por lo tanto, el programa, al crear el vector de variables de entrada, va reconociendo que función debe usar para resolver el modelo en función de los datos del modelo. Por ejemplo, si el tipo de motor es un turbofan, el vector de variables de entrada incluirá el empuje a nivel del mar como dato y la función que escogerá será la correspondiente a motor turbofán. Si fuese turbohélice, el dato incluido sería potencia a nivel del mar y la consiguiente función.

Las funciones a las que se llama son las funciones "calculador", que resuelven el modelo dinámico y calculan las actuaciones, y finalmente, guardar los resultados en otro fichero ".mat" que cuyo nombre será de la forma "nombredelmodelo_Resultados.mat".

Debido a las posibles opciones de las que dispone el programa, el archivo de "programa_actuaciones" tiene la estructura que vemos en el diagrama de flujo.

Vemos que, dependiendo de los datos del modelo, el programa llama a una determinada función "Calculador".

El funcionamiento de estas funciones es similar excepto por el tipo de motor y el tipo de pendiente. Ahora veremos cómo funcionan.

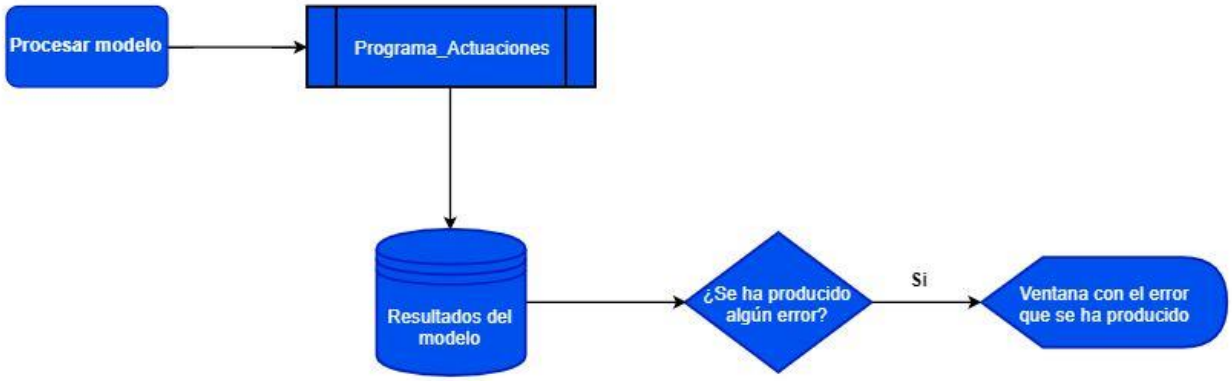


Figura 35. Diagrama de flujo de la ventana procesar modelo

4.4.1 Funciones calculador

En primer lugar, se asignan los datos del modelo a sus variables correspondientes dentro del modelo dinámico que hemos visto antes.

Acto seguido se define una pista. Si es constante tenemos la siguiente pista.

$$z = f(x) \qquad \dot{z} = f'(x)\,\dot{x} \qquad \ddot{z} = f''(x)\dot{x} + f'(x)\ddot{x}$$

$$z = \tan(\phi)x$$

$$f'(x) = \tan(\phi)$$

$$f''(x) = 0$$

Si la pendiente es variable se complica un poco definir la pista, pues es necesario hacer varias interpolaciones para que cada punto x tenga definido su correspondiente ángulo de pista $\phi(x)$, su coordenada z(x) y sus funciones $f'(x)$ y $f''(x)$.

Empezamos definiendo el perfil de la pista. Para ello, si recordamos, en los datos tenemos los puntos clave para definir la pista. Esto puntos pueden venir en forma de pendientes y distancias, o bien en forma de distancias y alturas.

Si es la primera de las opciones, se transforman los puntos para que estén en la forma distancias y alturas (simplemente se calculan las alturas correspondientes a cada punto de distancia en función de la pendiente) antes de ser introducidos en la función calculador.

Una vez hecho esto, nuestro vector distancias será el vector x y nuestro vector alturas será el vector z. Creamos un vector de puntos horizontales usando la distancia inicial (debería ser 0) y la distancia final, por ejemplo, de metro en metro, para tener un buen dominio sobre el que estudiar el problema. Este dominio se puede ampliar si se quiere, pero como de forma general una pista tendrá en torno a unos pocos km debería ser suficiente.

Ahora debemos crear un vector de puntos z que se ajuste al perfil longitudinal de la pista en cada punto x que tenemos. Normalmente esto se haría con una interpolación de spline cúbico, pero existe un problema, y es que esta interpolación crea un perfil que no se ajusta a la realidad, pues si se tiene pocos puntos el perfil oscila y zonas de pendiente positiva se convierten en zonas de pendiente negativa, por lo que los resultados con esta interpolación no serían demasiado precisos.

Por ello, se usará una interpolación de tipo spline cúbica de Hermite "pchip", que permite obtener un perfil más suave y uniforme. Si vemos el ejemplo comparativo que se muestra en la página oficial de Matlab [9] podemos apreciar mejor esta diferencia entre splines que comentábamos.

Se puede ver como la interpolación obtenida con el comando spline no sería demasiado acertada para el caso de una pista, mientras que la obtenida con "pchip" es mucho más uniforme.

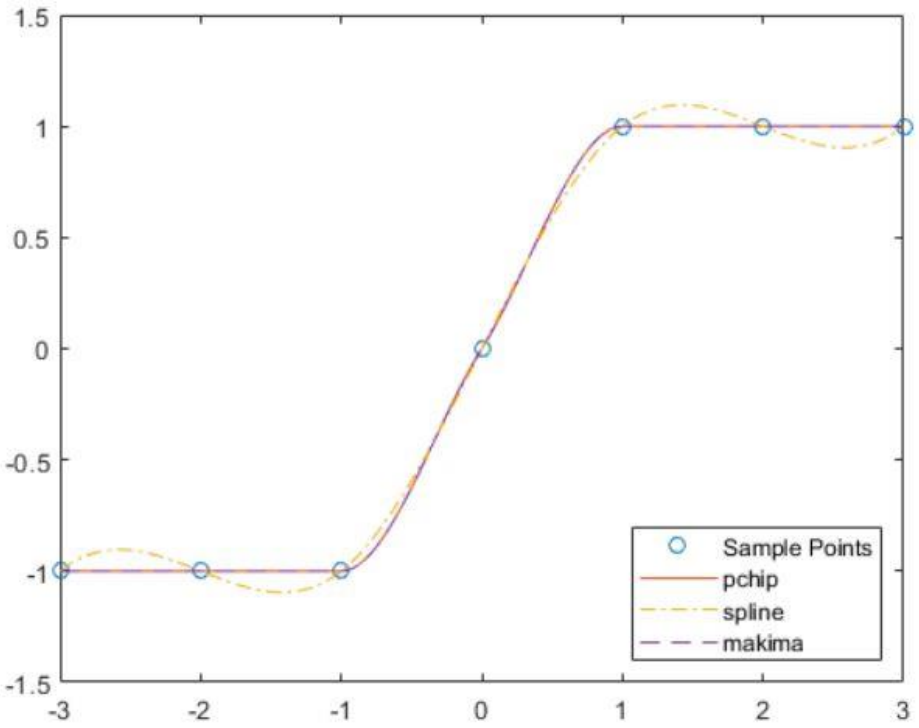


Figura 36. Comparación del comando pchip con otros más habituales [9]

A continuación, vemos un ejemplo para un aeropuerto hecho con el programa. La pista vendría definida por los siguientes puntos:

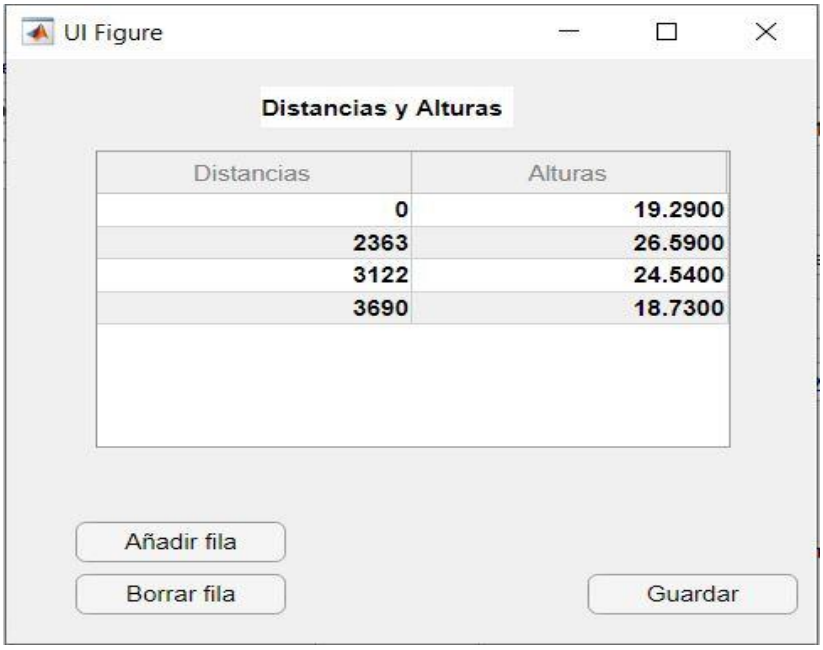


Figura 37. Ejemplo de puntos definitorios de una pista de aeropuerto

En la figura 38 podemos ver la comparación de ambos comandos.

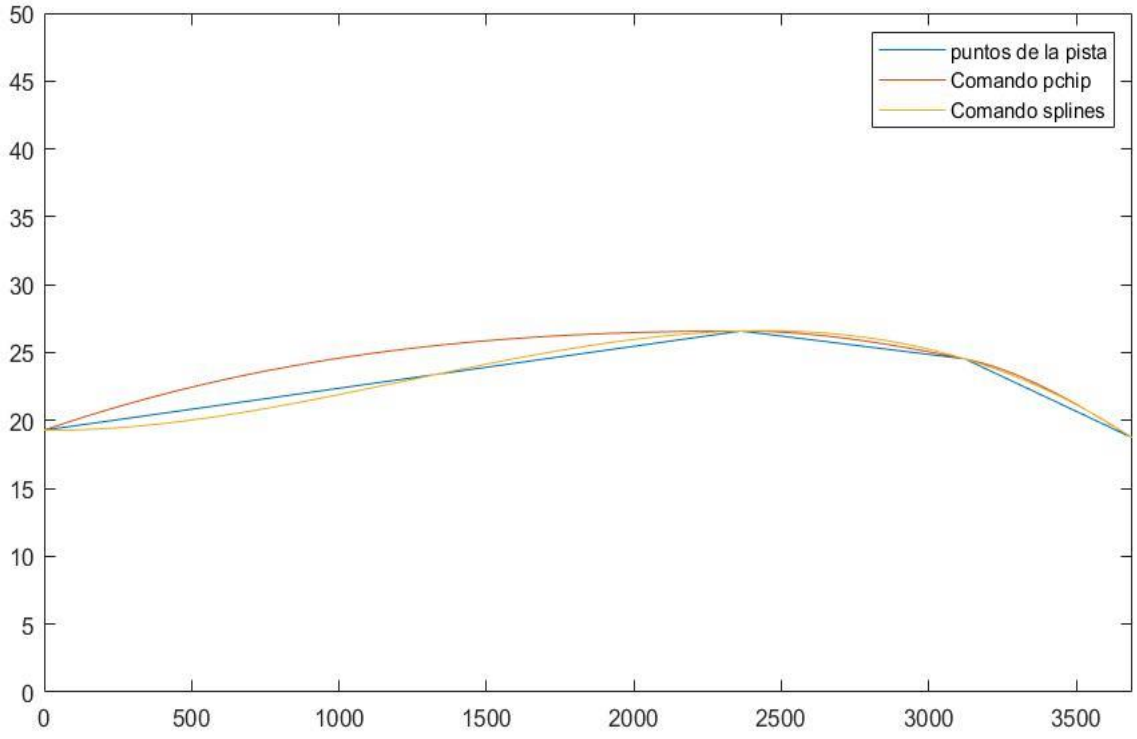


Figura 38. Comparación del comando pchip frente a otros más comunes con el aeropuerto de ejemplo

Podemos ver como el comando "pchip" proporciona un perfil más uniforme y que además no altera tanto el valor de la pendiente como la otra interpolación.

Pero no podemos trabajar con dos vectores de puntos para resolver nuestro modelo, necesitamos una función que me proporcione z para cualquier punto x, ya que, al ir integrando el modelo, es necesario conocer la posición del avión en cada momento. Usaremos el comando "polyfit" [10] que, para una pareja de vectores, devuelve los coeficientes para un polinomio del grado que definamos, siendo esta la mejor solución en el sentido de los mínimos cuadrados.

Usando el comando "polyval" [11], que evalúa un polinomio de los coeficientes que le indiquemos en un punto x que le indiquemos, tendremos nuestro perfil de pista de forma aproximada para poder resolver el sistema de ecuaciones, ósea, ya tenemos $z(x)$.

Ya tenemos definida el perfil de la pista, falta definir $\phi(x)$, $f'(x)$ y $f''(x)$.

Para el ángulo de pista, lo definimos como

$$\phi(x_i) = \arctan\left(\frac{z_{i+1} - z_i}{x_{i+1} - x_i}\right)$$

De esta forma tenemos definido un vector ϕ. Igual que para el perfil de la pista, creamos los coeficientes de un polinomio que nos proporcionará $\phi(x)$.

En el ejemplo anterior podemos ver cómo funciona la aproximación que hemos hecho, podemos ver que se aproxima bastante bien.

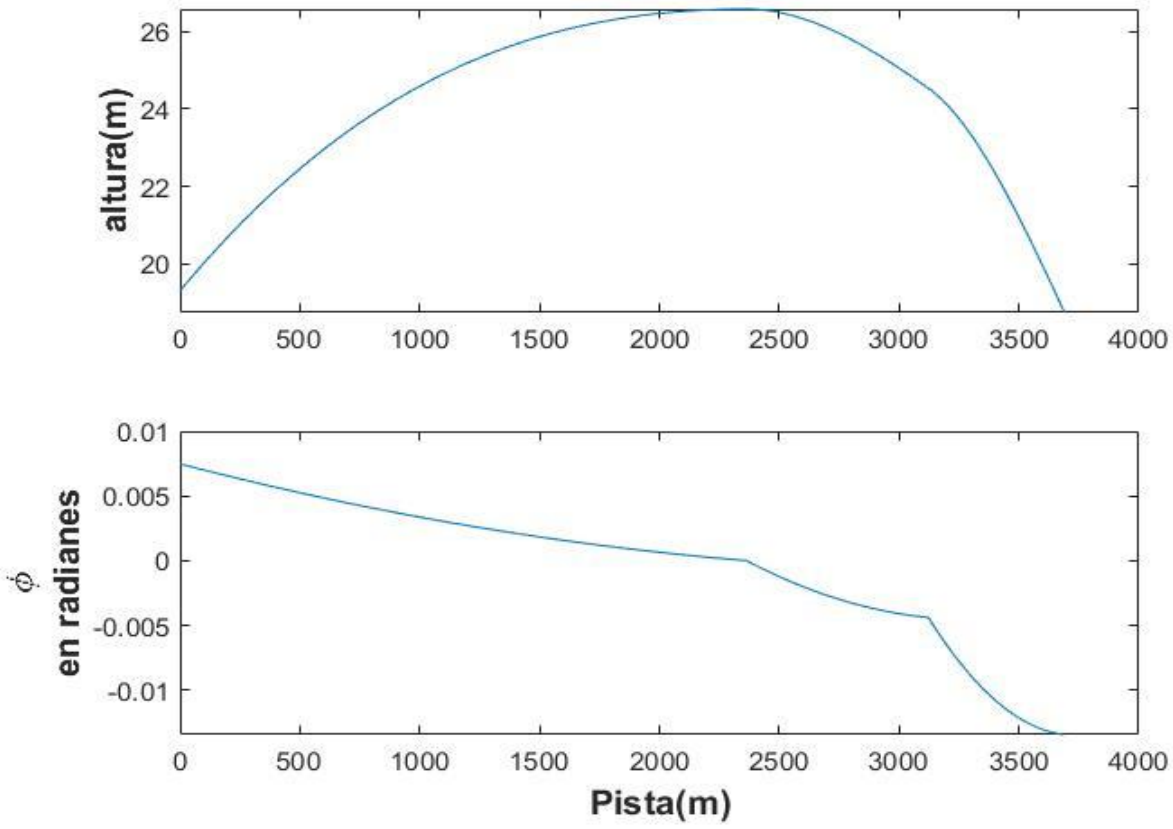


Figura 39. Ejemplo del cálculo del perfil de pista y del ángulo de pista ϕ

Para las funciones $f'(x)$ y $f''(x)$ definimos las derivadas de forma aproximada como,

$$f'(x_i) = \frac{z_{i+1} - z_{i-1}}{h}$$
$$f''(x_i) = \frac{z_{i+1} - 2z_i + z_{i-1}}{h^2}$$

Creamos los coeficientes de los polinomios y listo. Siguiendo con el ejemplo anterior, observamos las aproximaciones en la siguiente gráfica.

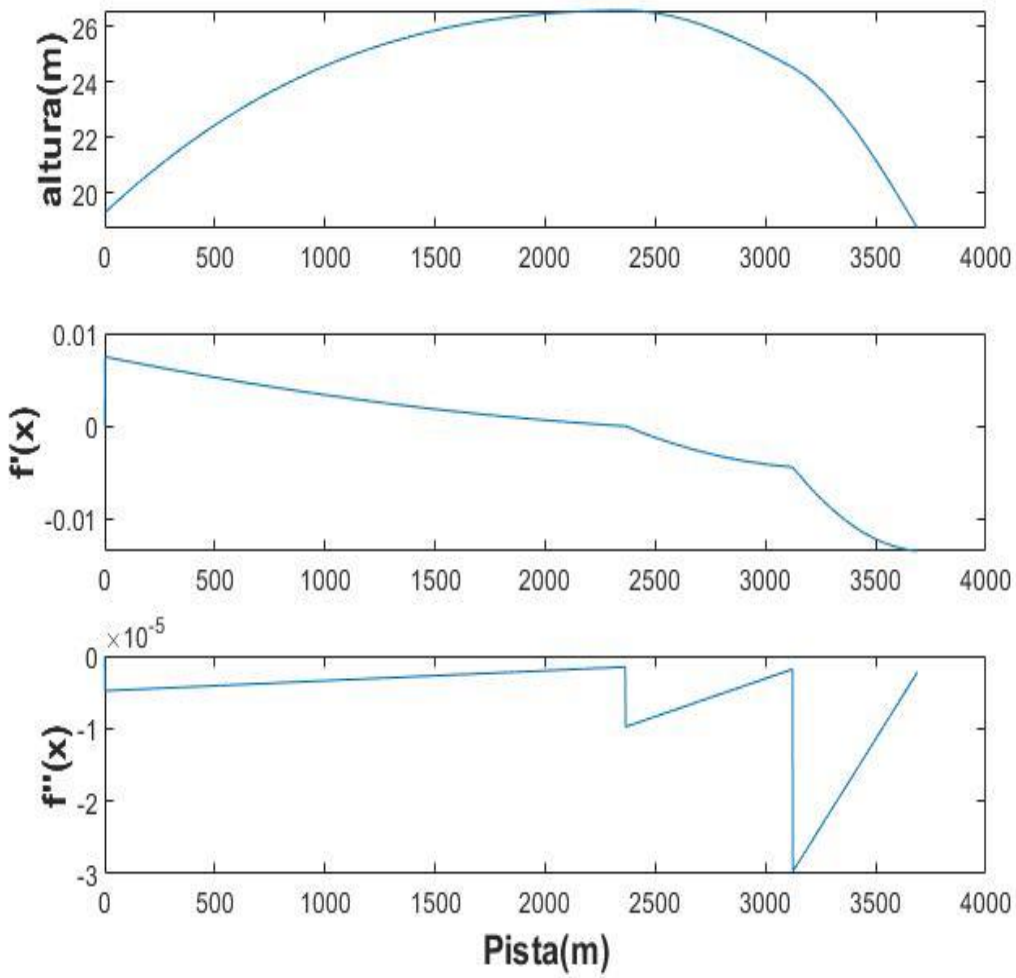


Figura 40. Ejemplo del cálculo del perfil de pista junto a las funciones $f'(x)$ y $f''(x)$

Como podemos observar, se producen saltos en la segunda derivada debido a cambios bruscos o esquinas en la primera derivada. Este problema se debe a que estas aproximaciones están muy limitadas a estos fallos numéricos y para solucionarlo tendríamos que recurrir a aproximaciones más complejas.

Ahora bien, si nos fijamos en el rango de valores del eje de ordenadas podemos ver que estos saltos

son en realidad de un orden muy pequeño y que no merece la pena complicarse mucho más en este asunto debido a la poca variación que supone este error en la aproximación.

Estos valores tan pequeños en la primera y segunda derivada se deben a que, como es de suponer en una pista de un aeropuerto, las variaciones en la pendiente de la pista son muy pequeñas y se producen muy poco a poco a lo largo de la pista. En el ejemplo que estamos siguiendo, se varía en altura unos cinco o seis metros hacia arriba y abajo, pero en más de tres mil metros de pista, por lo que podemos ver como la variación realmente es muy baja.

Una vez definida la pista, podemos empezar a resolver el sistema de ecuaciones planteado en el capítulo anterior. Recordemos que queremos resolver las ecuaciones diferenciales obtenidas de la segunda ley de Newton.

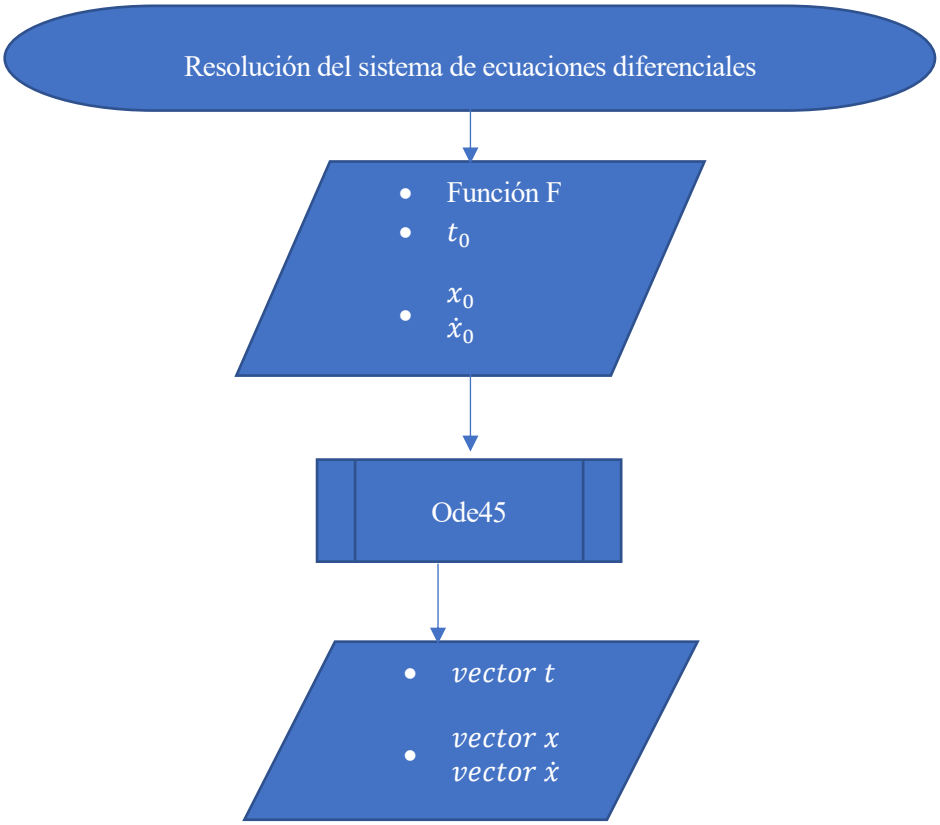
Para resolver el sistema de ecuaciones diferenciales planteado, Matlab incorpora el comando "ode45" [6], el cual resuelve ecuaciones diferenciales por el método de orden medio.

Su funcionamiento es el siguiente, se introduce una función que contenga el sistema de ecuaciones diferenciales con las variables derivadas despejadas (y todo en función de $x,\ \dot{x}\ o\ t$), un intervalo de tiempo en el cual se van a integrar el sistema de ecuaciones y unas condiciones iniciales para el sistema. Todo esto lo tenemos de nuestra resolución analítica.

$$\begin{cases} F(1) = \dot{x} = f_1(x, \dot{x}, t) \\ F(2) = \ddot{x} = f_2(x, \dot{x}, t) \end{cases}$$

Condiciones iniciales: $\begin{cases} t_0 = [t_{inicial}\ t_{final}] \\ x_0 \\ \dot{x}_0 \end{cases}$

El comando devolverá un vector tiempo y un vector con las soluciones del sistema correspondientes a cada punto del vector tiempo.



Como vimos antes, la resolución del sistema de ecuaciones estará dividido en tres partes, rodadura, rotación y vuelo. Por tanto, cada una de estas partes tendrá que tener una función, un intervalo de tiempo de integración y unas condiciones iniciales distintas.

Recordemos que ya tenemos definidas las ecuaciones que hay que integrar en cada parte del despegue en la resolución analítica. Entonces, en el programa simplemente debemos introducir dichas ecuaciones en funciones distintas, estas funciones se llamarán:

1. Función rodadura
2. Función rotación
3. Función vuelo

Se llamará de forma secuencial a estas funciones con el comando "ode45" para de esta forma resolver el aterrizaje [6].

Cabe mencionar que para definir $\theta = \theta(t)$ en la fase de rotación se ha tenido que actuar de la misma forma que para definir la pista, con una interpolación que me proporcione $\theta(t)$ para cada instante de la integración.

Las condiciones iniciales de la primera fase serán cero, pues se supone que el despegue comienza con el avión parado. A partir de la primera parte, las condiciones iniciales de las siguientes partes serán el punto final de la parte anterior, proporcionando así continuidad al problema a pesar de estar dividido en tres partes.

Por último, mencionaremos como tenemos definidos los intervalos de tiempo, pues anteriormente he comentado que se introduce un intervalo de tiempo y el comando calcula las soluciones asociadas a dicho intervalo. Para ello, sería necesario conocer el instante de tiempo en el que acaba cada fase del despegue, dato que a priori no tenemos.

Sin embargo, podemos introducir una condición en cada una de las funciones, y si dicha condición se cumple, entonces la integración se detiene en dicho punto. Para ello usamos el comando 'odeset' junto con la especificación 'Events' [12] para definir una función que será la condición de parada. Las condiciones para cada fase serían las siguientes:

1. Fase de rodadura: $\dot{x} = V_R$
2. Fase de rotación: $N = 0$
3. Fase de vuelo: $h = 11\ metros$

Nótese que debido a que algunas variables están definidas mediante interpolaciones, puede que el resultado no sea exacto, pero nos dará una aproximación razonable.

Una vez definidas todas las fases del despegue, se resuelve el sistema y tendríamos la solución al problema. Con las expresiones vistas en la parte analítica se encuentran todas las variables que influían en el problema.

Los resultados se guardarán con el nombre del modelo seguido de resultados, por ejemplo, "ejemplo_Resultados.mat".

Estos resultados podrán ser analizados en el siguiente bloque.

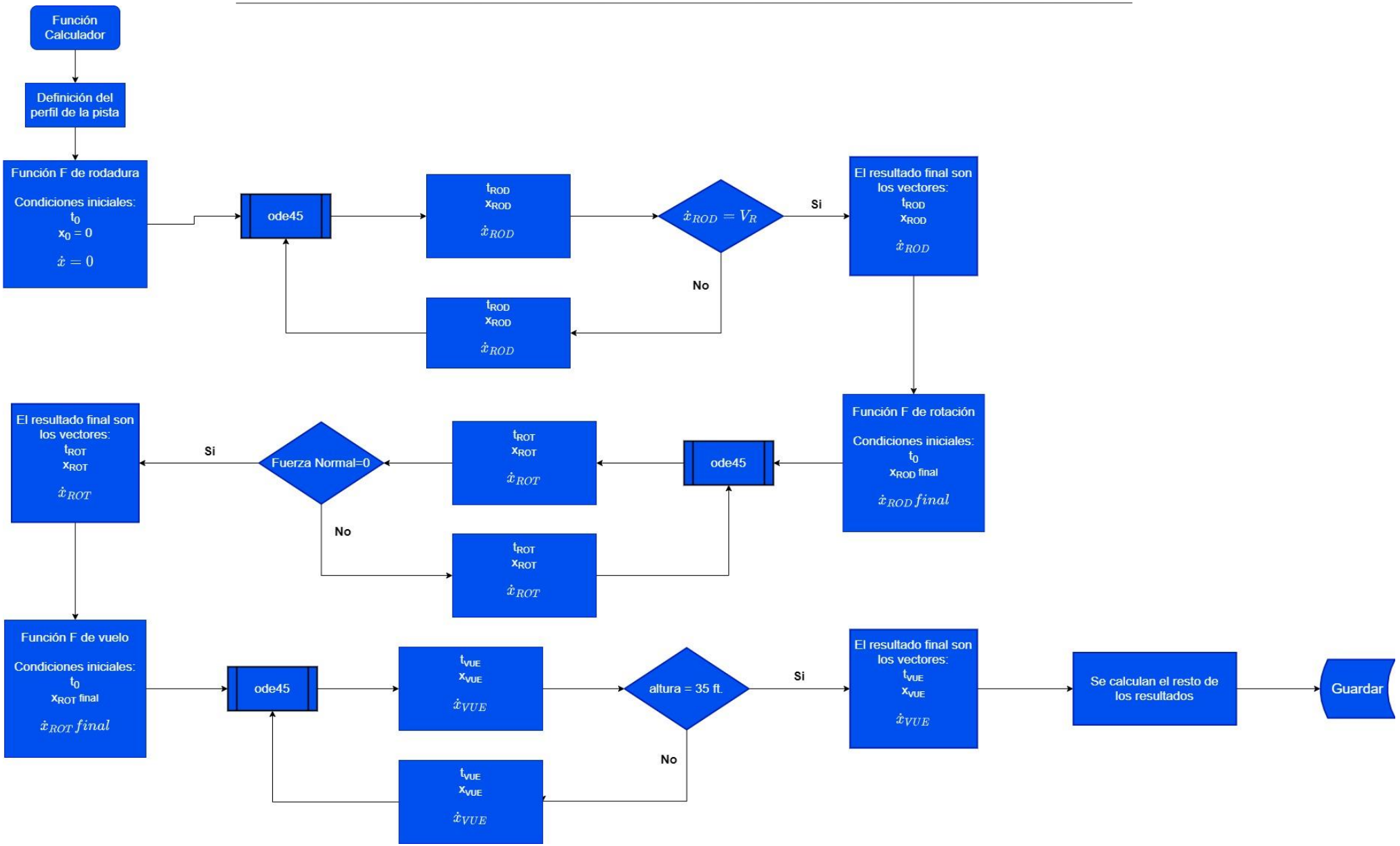


Figura 41. Diagrama de flujo de un archivo calculador

Llegados a este punto, podemos decir que hemos calculado un despegue en el que todo ha ido bien, desde la suelta de frenos hasta que se ha alcanzado una altura de 35 pies.

Sin embargo, hemos visto anteriormente que las actuaciones certificadas no están definidas exactamente de esta forma, sino que hay que tener en cuenta si falla un motor y elegir entre una distancia u otra, e incluso en la distancia de aceleración parada el avión ni siquiera llega a separarse del suelo. Entonces serán necesarios hacer algunos cambios en estas funciones para que podamos calcular dichas actuaciones, pero vamos a ver que van a ser mínimos y que el esquema general se mantiene, por eso se ha explicado al detalle estas funciones.

En realidad, más que cambios en la estructura, estos cambios van a ser sobre las condiciones de estos bloques. Por ejemplo, que en cierto momento falle un motor, en este caso habrá que restar uno al número de motores existente a partir de la velocidad de fallo de motor. Este es el tipo de cambios al que nos referimos.

A continuación, veremos como con unas mínimas modificaciones sobre las funciones calculador que ya tenemos hechas podremos sacar todas las actuaciones del despegue.

4.4.2 Cálculo de actuaciones certificadas

Para empezar, decir que el procedimiento es el mismo tanto si es turbofán o turbohélice y para perfil de pista constante o variable. Una vez obtenemos los resultados que hemos visto anteriormente, y recordando lo que hemos visto en la parte de teoría del capítulo 2, lo primero que hacemos es calcular la distancia de despegue TOD, la carrera de despegue TOR con todos los motores operativos, la velocidad de despegue y la velocidad V_2.

$$TOD = 1.15\, x_{vue}(t_{vue\ final})$$

$$V_{LOF} = \frac{\dot{x}_{ROT}(t_{ROT\ final})}{\cos(\phi(x_{rot}(t_{rot\ final})))} \qquad V_2 = \frac{V_{Ax}}{\cos(\gamma_A(x_{vue\ final}))} \qquad V_{Ax} = \dot{x}_{vue}(t_{vue\ final}) - V_{Wx}$$

$$d = x_{vue}(t_{vue\ final}) - x_{rot}(t_{rot\ final})$$

$$TOR = 1.15(x_{rot}(t_{rot\ final}) + \frac{d}{2})$$

Como vemos, estos resultados son prácticamente inmediatos, por lo que, con los datos que ya disponíamos hemos podido calcular estas actuaciones. Recordar que, para la velocidad, la que nos interesa es la velocidad aerodinámica, que está relacionada de esa forma con el resultado que obtenemos. Se recomienda revisar el capítulo 3 en caso de dudas.

Sin embargo, para calcular la distancia de despegue TOD y la carrera de despegue TOR con fallo en un motor crítico, será necesario realizar unos pequeños ajustes.

Entonces, para calcular estas actuaciones con fallo de motor se usa otra función calculador diferente.

Para calcular estas actuaciones es necesario tener en cuenta que a partir de la velocidad de fallo de motor V_{EF} se va a considerar que un motor crítico falla, por tanto, la única diferencia con respecto al caso anterior es que vamos a tener dos fases de rodadura en vez de una, y va a estar estructurado de la siguiente forma.

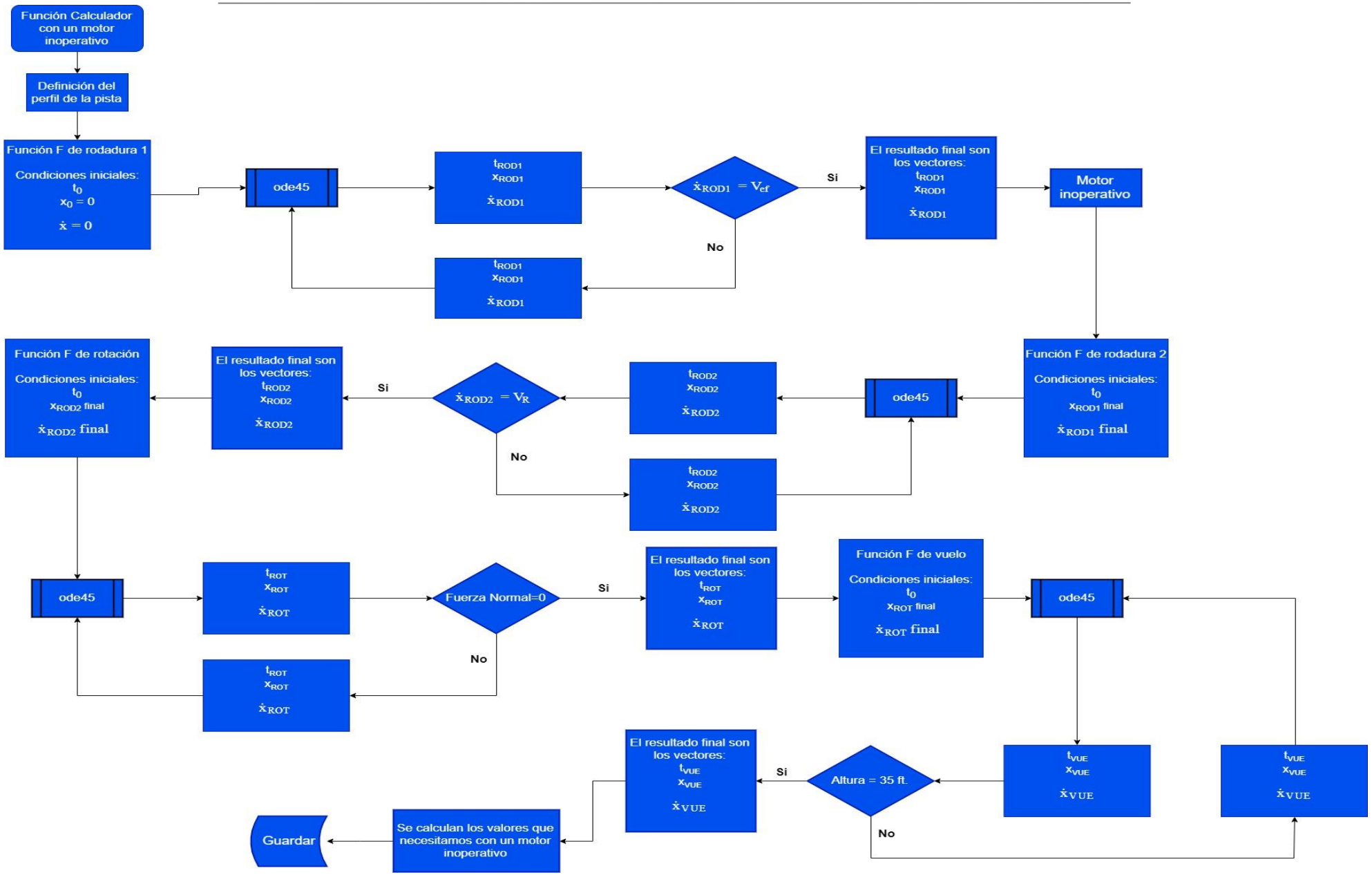


Figura 42. Diagrama de flujo de la función calculador con un motor inoperativo

Nótese que el procedimiento seguido es exactamente el mismo que para el caso con todos los motores operativos, pero introduciendo unas pequeñas variaciones. En esta parte también se calcula la velocidad de decisión V_1, que recordemos que era igual a V_{EF} más el incremento de velocidad en el tiempo t_{REC} que transcurre desde que falla el motor hasta que el piloto se da cuenta de dicho fallo.

Una vez tenemos los nuevos resultados con la variación de que un motor crítico falla, tenemos:

$$TOD_{n-1} = x_{vue\ n-1}(t_{vue\ final})$$

$$d_{n-1} = x_{vue\ n-1}(t_{vue\ n-1\ final}) - x_{rot\ n-1}(t_{rot\ n-1\ final})$$

$$TOR_{n-1} = x_{rot\ n-1}(t_{rot\ n-1\ final}) + \frac{d_{n-1}}{2}$$

$$V_{LOF\ n-1} \qquad V_{2\ n-1} \qquad V_1 = \frac{\dot{x}_{ROT_2}(t_{Rec})}{\cos(\phi(x_{rot_2}(t_{rec})))}$$

También calculamos las velocidades V_{LOF} y V_2 para un motor inoperativo, pero omitimos las expresiones ya que son exactamente iguales, pero con los resultados obtenidos de esta integración

Finalmente se elige la mayor de ambas distancias como la TOD o la TOR definitiva.

Mencionar que en esta nueva integración solo se guardan los resultados que hemos comentado y no todos los resultados obtenidos de la integración, como hacíamos en la integración con todos los motores. Esto se debe a la poca eficiencia que supone tener dos grandes matrices con resultados, y se optó por dejar solo la correspondiente al despegue sin fallo.

Por último, la actuación que requiere más cambios será la distancia de aceleración parada ASD. El procedimiento y la estructura siempre será la misma, lo único que cambia son las funciones que definen el despegue y las condiciones de parada de la integración de dichas funciones.

Para modelar el momento en el que el avión está frenando habrá que definir una función parada la cual en realidad será igual a la función rodadura anteriormente explicada salvo por dos cambios. El primero será el empuje nulo, y el segundo será la introducción del coeficiente de frenada $\mu_{frenada}$.

$$\dot{x} = V_x$$

$$\ddot{x} = \dot{V}_x = \frac{1}{m}(-Lsen(\gamma_A) - Dcos(\gamma_A) - Nsen(\phi) - F_R\cos(\phi))$$

$$\begin{matrix}\dot{x} \\ \ddot{x}\end{matrix} = f(x, \dot{x}, t)$$

$$F_R = \mu_{frenada} N$$

$$\mu_{frenada}$$

$$T = 0$$

También, al igual que antes distinguiremos entre distancia de aceleración parada ASD con todos los motores operativos y ASD con fallo en un motor crítico.

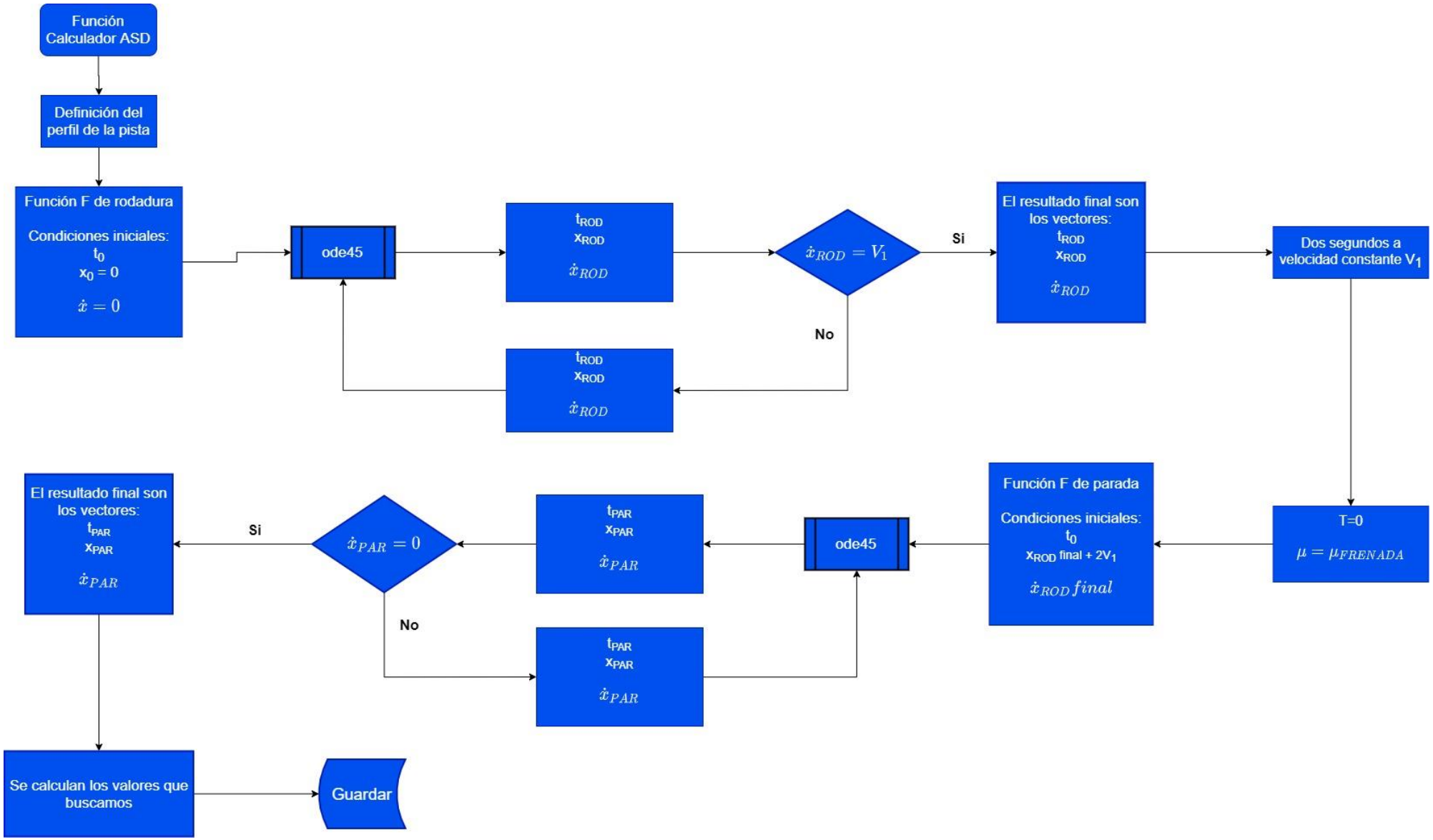


Figura 43. Diagrama de flujo de la función calculador para ASD

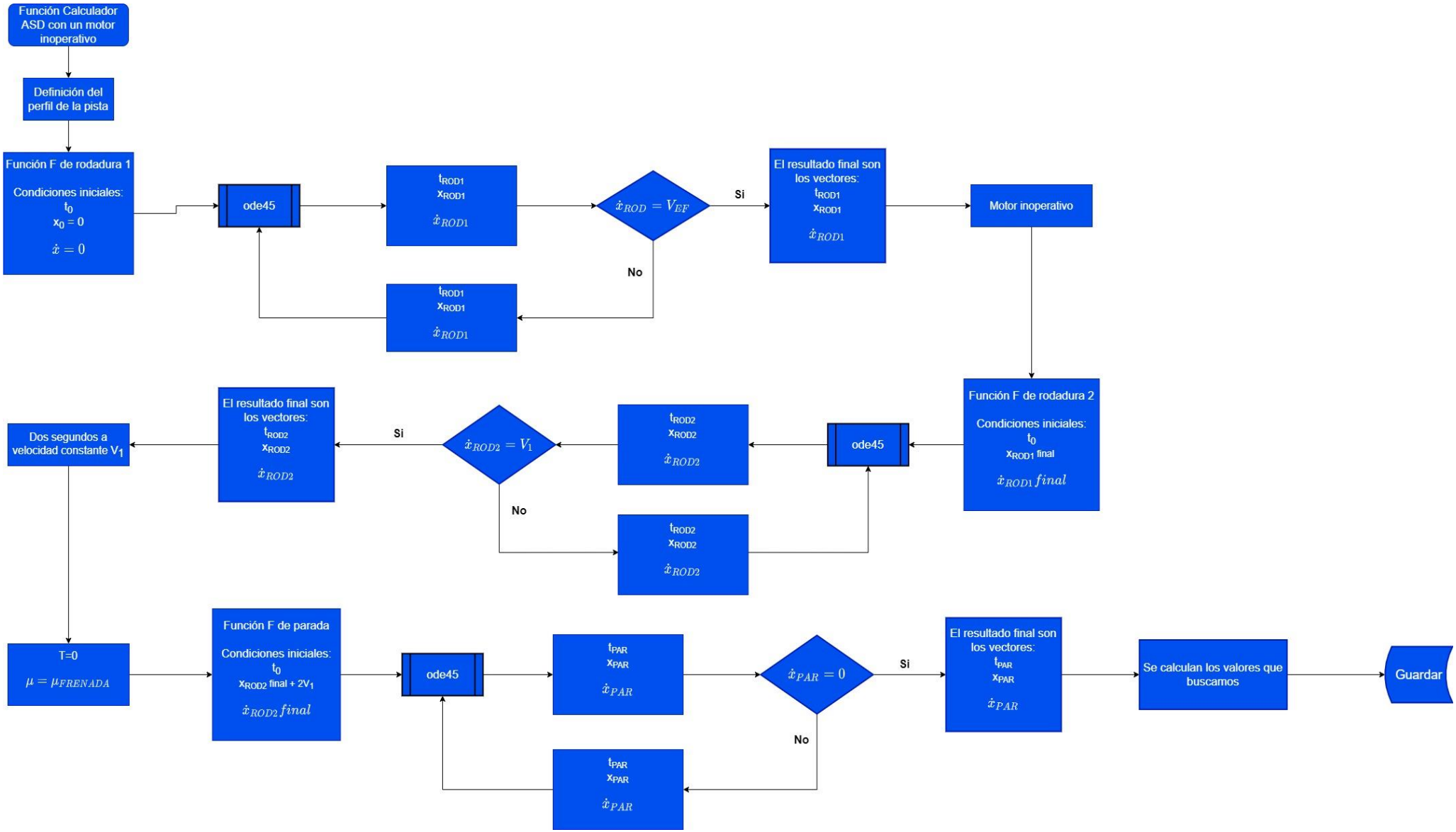


Figura 44. Diagrama de flujo de la función calculador para ASD con un motor inoperativo

La distancia de aceleración parada ASD con todos los motores operativos quedaría de la siguiente forma:

$$ASD = x_{parada}(t_{parada\ final})$$

Para calcular ASD con fallo en un motor crítico se considera que dicho motor falla a la velocidad de fallo de motor V_{EF}, el avión acelera hasta V_1 y luego frena hasta quedarse parado.

La distancia de aceleración parada ASD con fallo de motor crítico quedaría de la siguiente forma:

$$ASD_{n-1} = x_{parada\ n-1}(t_{parada\ n-1\ final})$$

También calculamos el tiempo que tarda en completarse la parada.

Para concluir con el archivo 'programa actuaciones', faltaría explicar el bloque en el que se comprueban posibles fallos en el cumplimiento de la normativa. Si recordamos del capítulo 2, las restricciones directas que teníamos para nuestro programa respecto a las velocidades en términos de CAS eran:

- $V_1 > V_{MCG} + (\Delta V)_{t_{REC}}$
- $V_{MCA} > 1.13 V_{SR}$
- $V_R > V_1$
- $V_R > 1.05 V_{MCA}$
- $V_{LOF} > 1.1 V_{MU}$
- $V_{LOF\ n-1} > 1.05 V_{MU}$
- $V_2 > 1.1 V_{MCA}$
- $V_2 > 1.13 V_{SR}$ para turbofán y turbohélice con 2 o 3 motores
- $V_2 > 1.08 V_{SR}$ para turbohélice con más de 3 motores
- $V_{EF} > V_{MCG}$

Adicionalmente, también se comprueba que el programa haya integrado correctamente.

Lo primero para comprobar los errores es pasar de TAS a CAS las velocidades que son resultado del programa como ya explicamos anteriormente, las velocidades que se introdujeron como dato ya estaban en CAS.

Acto seguido, se crea una variable llama 'error' numerada para cada error posible de los de arriba. Si alguna de estas condiciones se incumple, su variable 'error' correspondiente se guarda con el valor 1, si se cumple la condición se guarda como valor 0, y estas variables también se guardan en el fichero de resultados.

Después, en el código de la interfaz, se comprueban estas variables y si alguna tiene el valor de 1, se llama a una ventana personalizada para cada variable que informa al usuario de que se ha producido un error.

Si no se encuentran fallos, no salta ninguna ventana.

Figura 45. Ejemplo de una de las posibles ventanas que avisan de un error

Con esto, tendríamos calculadas todas las actuaciones certificadas de nuestro modelo, y nuestro fichero de resultados contendría la información que se muestra en la figura 46.

Workspace

Name	Value
ASD	923.6722
ASD_sinmotor	947.0951
ASD_t	28.5678
ASD_t_sinmotor	30.0748
error0	0
error1	0
error2	0
error3	0
error4	0
error5	0
error6	0
error7	0
error8	0
error9	0
error_sinmotor	0
Resultados	*764x12 string*
TOD	1.6052e+03
TOD_sinmotor	2.2294e+03
TOR	1.2787e+03
TOR_sinmotor	1.8390e+03
V1	51.7613
V2	82.0314
V2_sinmotor	78.9039
Vlof	71.5519
Vlof_sinmotor	72.4624

Figura 46. Archivo de Resultados guardados tras procesar el modelo en el Workspace

Así finalizarían las funciones de la ventana procesar modelo, con todos los resultados guardados en un archivo '.mat'.

Con esta información ya podemos avanzar a la siguiente ventana del programa.

4.5 Opción de Visualización de Modelos y Resultados

Existe la opción en la ventana inicial de poder visualizar la lista de modelos y de resultados que hay guardados en el fichero actual. Esto es útil para poder comprobar que después de procesar el modelo del usuario, los resultados se han guardado bien y en el fichero actual.

El programa trae unos ejemplos de serie, y para poder modificarlos o visualizarlos es necesario saber su nombre.

Simplemente, pulsando el botón Lista de Modelos y Resultados, aparecerá en el recuadro de al lado.

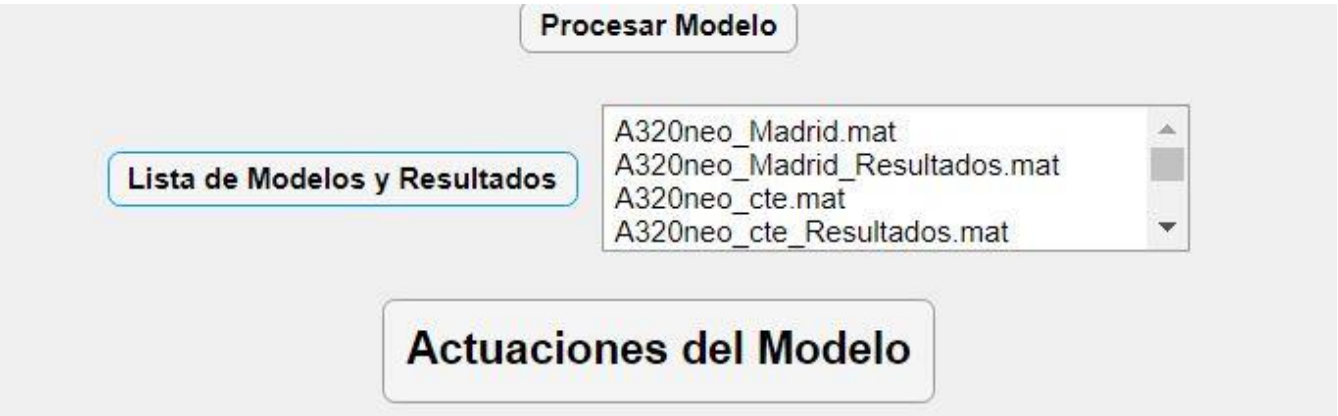


Figura 47. Opción de visualizar modelos y resultados disponibles

4.6 Ventana Actuaciones del Modelo

En esta ventana, tras todo el proceso de cálculo, podremos visualizar nuestros resultados y las condiciones que imponía la norma sin la necesidad de tener que abrir los archivos '.mat' en Matlab, de una forma intuitiva y agradable a la vista.

Primero introducimos el nombre del modelo que ya esté procesado, elegimos si queremos ver los resultados de la integración en forma de tabla o en forma de gráfica comparando dos de ello y le damos al botón de cargar. Se cargarán todas las actuaciones calculadas, así como las restricciones normativas y los resultados de la integración sin fallo de motor.

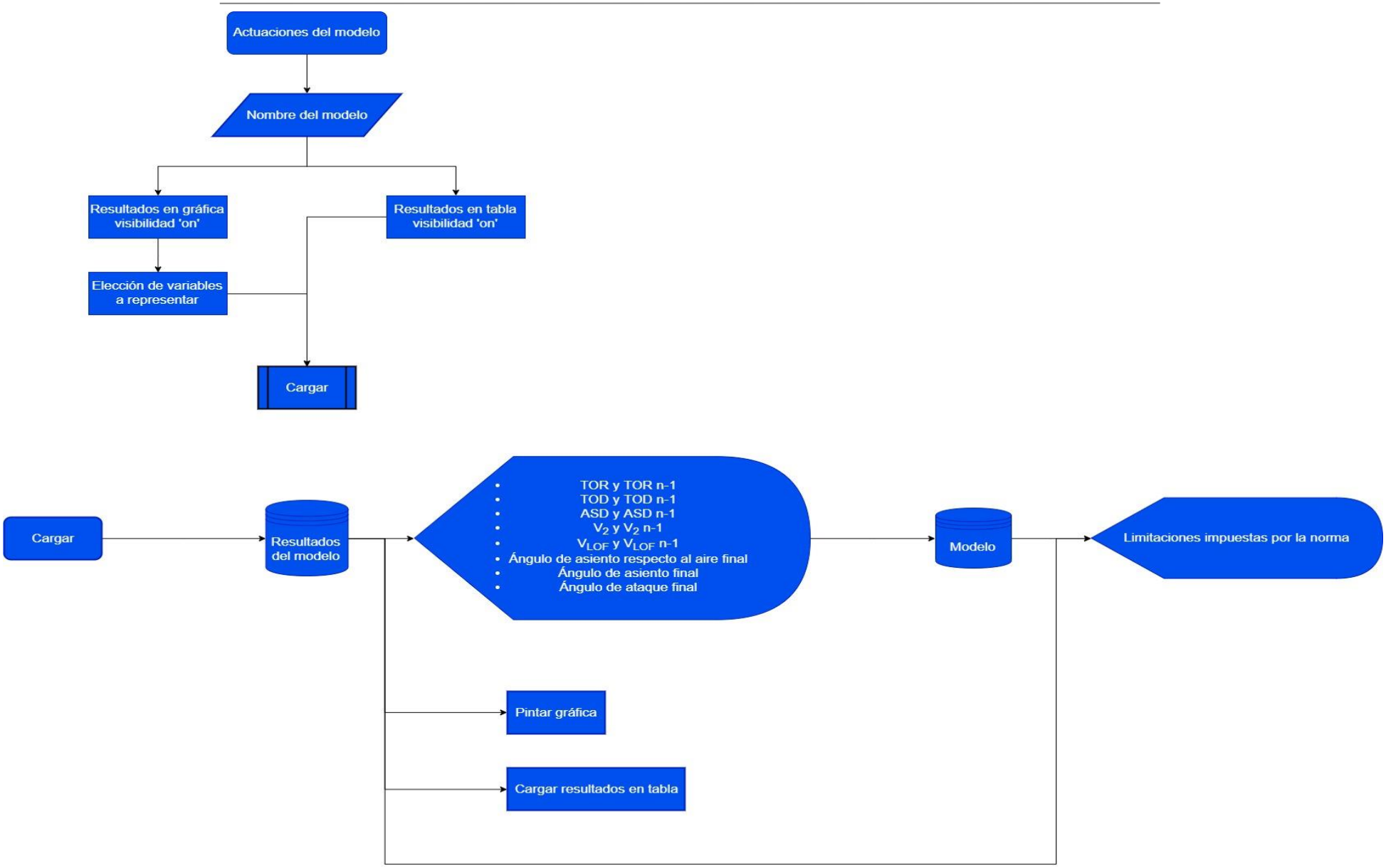


Figura 48. Diagrama de flujo de la ventana actuaciones del modelo

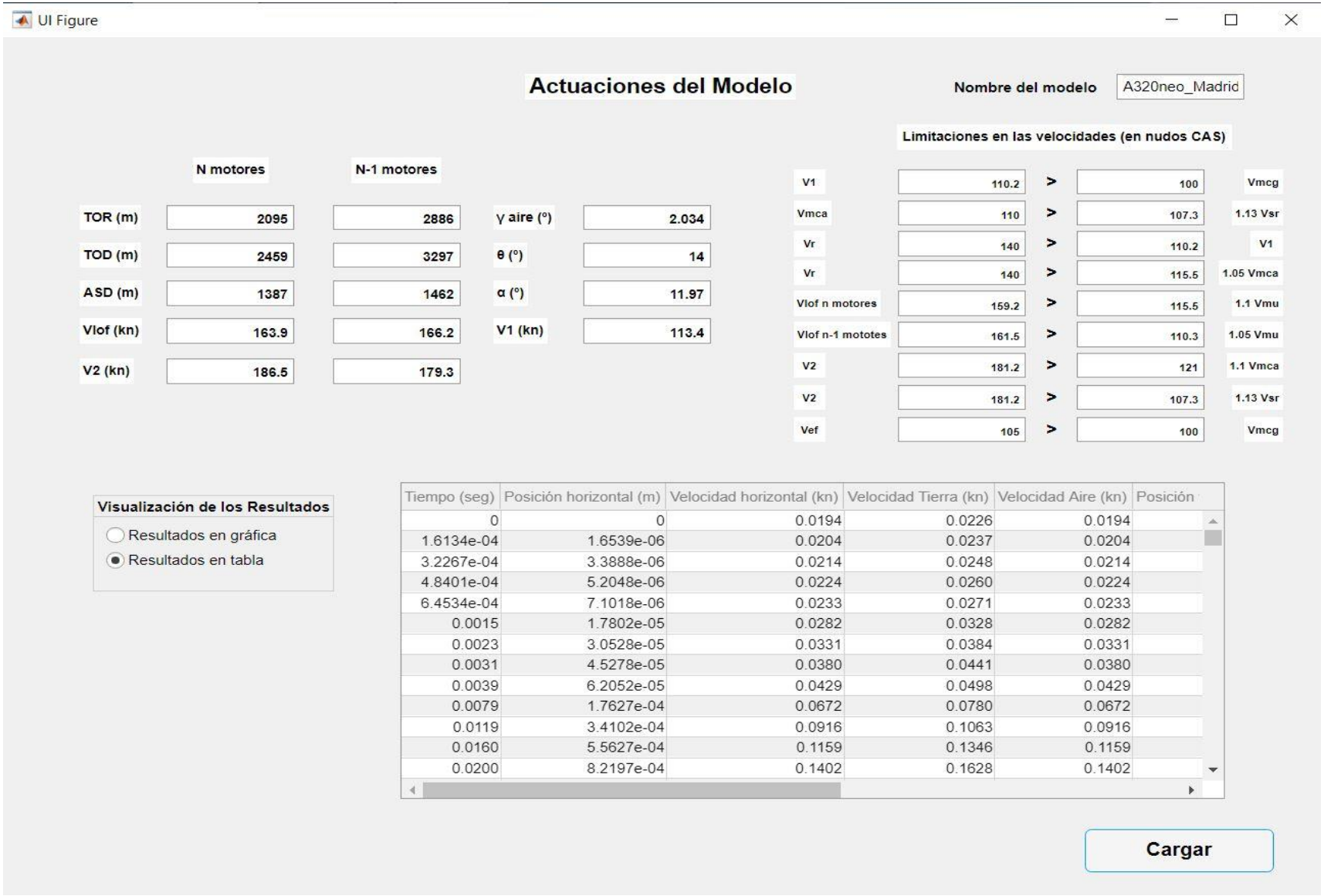


Figura 49. Ventana actuaciones del modelo, ejemplo de tabla

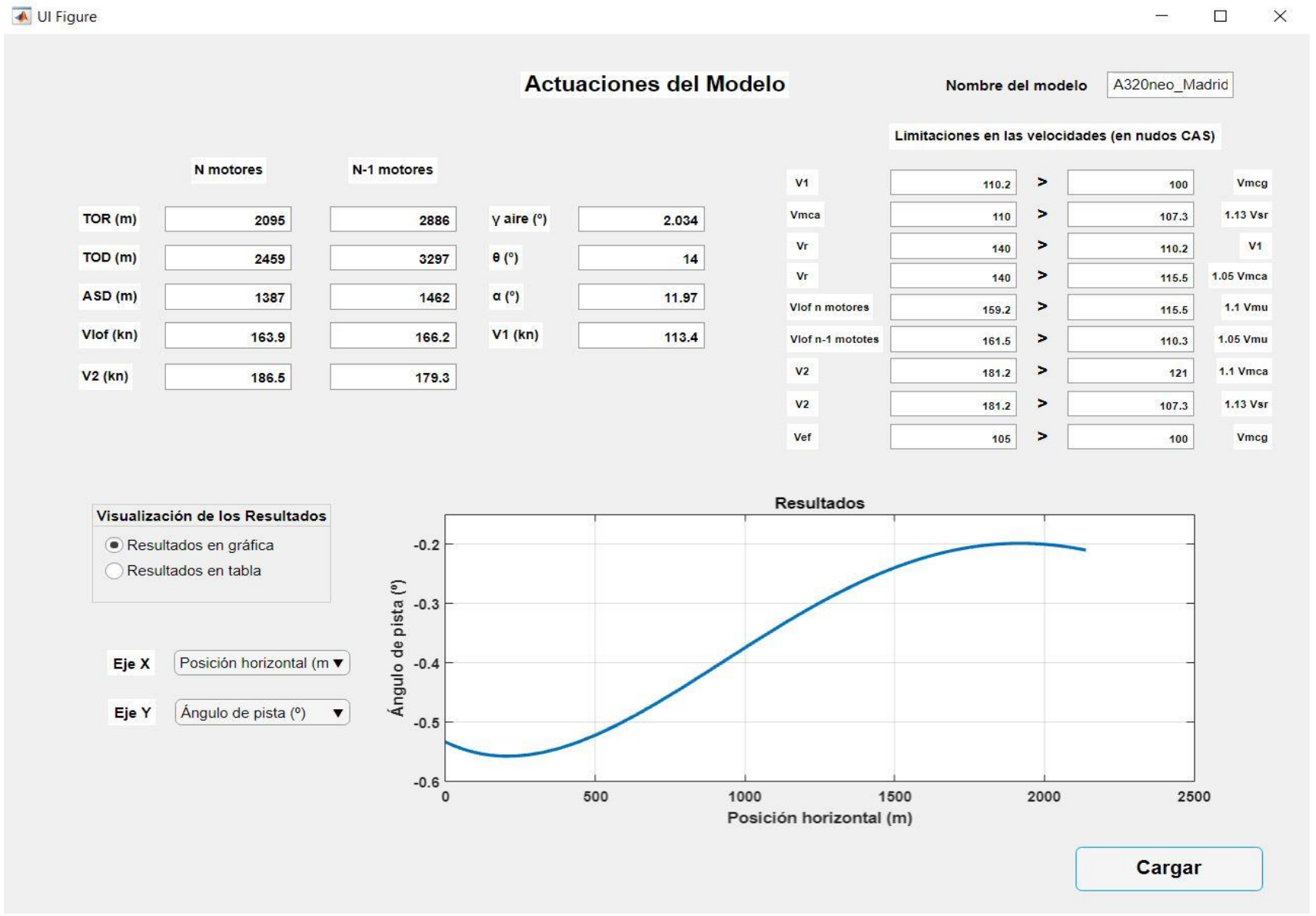


Figura 50. Ventana actuaciones del modelo, ejemplo de gráfica

Los resultados que se muestran en la zona inferior corresponderían al despegue sin ningún tipo de incidente, mientras que en la parte superior mostramos las actuaciones definidas por la norma.

Se va a mostrar las actuaciones tanto con fallo de motor crítico como sin fallo de motor, para poder comparar una con otra. También se muestran los ángulos de ataque, de asiento respecto al aire, y de asiento.

Adicionalmente se incluyen también las restricciones que impone la norma para poder comprobar que, efectivamente, se cumplen. Como ya comentamos anteriormente, estas restricciones están en términos de velocidad CAS, mientras que en la parte de los resultados están tal y como se calculan en el programa, en términos de TAS.

Tanto las columnas de la tabla como las opciones para graficar en ambos ejes son las variables que se han ido obteniendo a través de la integración sin fallo de motor, que son las siguientes.

- Tiempo (seg)
- Posición horizontal (m)
- Velocidad horizontal (kt)
- Velocidad en tierra (kt)
- Velocidad aerodinámica (kt)
- Posición vertical (m)
- Velocidad vertical (kt)
- Altura (ft)
- Ángulo de asiento (º)
- Ángulo de pista (º)
- Ángulo de asiento respecto al aire (º)
- Ángulo de ataque (º)

Sería interesante ver qué efectos producen en las actuaciones la variación de ciertas variables operacionales clave. Para ello, pasaremos a la siguiente ventana del programa.

4.7 Ventana Barrido de parámetros

Una vez hemos calculado las actuaciones de nuestro modelo nos podríamos preguntar cómo afectaría si cambiamos ciertos parámetros como, por ejemplo, el peso o la velocidad de decisión. Sin embargo, sería muy engorroso tener que modificar el modelo y volver a procesarlo cada vez que queramos variar algo.

Es por ello que existe esta ventana. La idea es que de forma aproximada se cree un modelo en la ventana crear modelo y a partir de ese modelo, analicemos las tendencias de las actuaciones en función de diversas variables que en principio no sabemos qué valor queremos darle exactamente. Por ejemplo, queremos despegar un avión en una pista en concreto, pero no sabemos cuál es el límite de peso para el avión.

Entonces, introducimos el nombre de nuestro modelo, elegimos en variable de estudio el peso y concretamos un intervalo de pesos en el que estudiar las actuaciones. Por último, elegimos las actuaciones que queremos visualizar en la gráfica, y de esta forma podemos hacernos una idea de las limitaciones con respecto al peso que presenta nuestro avión.

De igual forma que con el peso, podemos hacer este tipo de estudios con otras variables.

No se incluye diagrama de flujo de esta ventana ya que simplemente se trata de un bucle en el que se llama al archivo programa actuaciones variando la variable que queramos estudiar. El programa por defecto dividirá el intervalo en diez pasos siempre. Aunque este valor se puede modificar, es más recomendable estudiar intervalos más pequeños si se quiere mayor precisión.

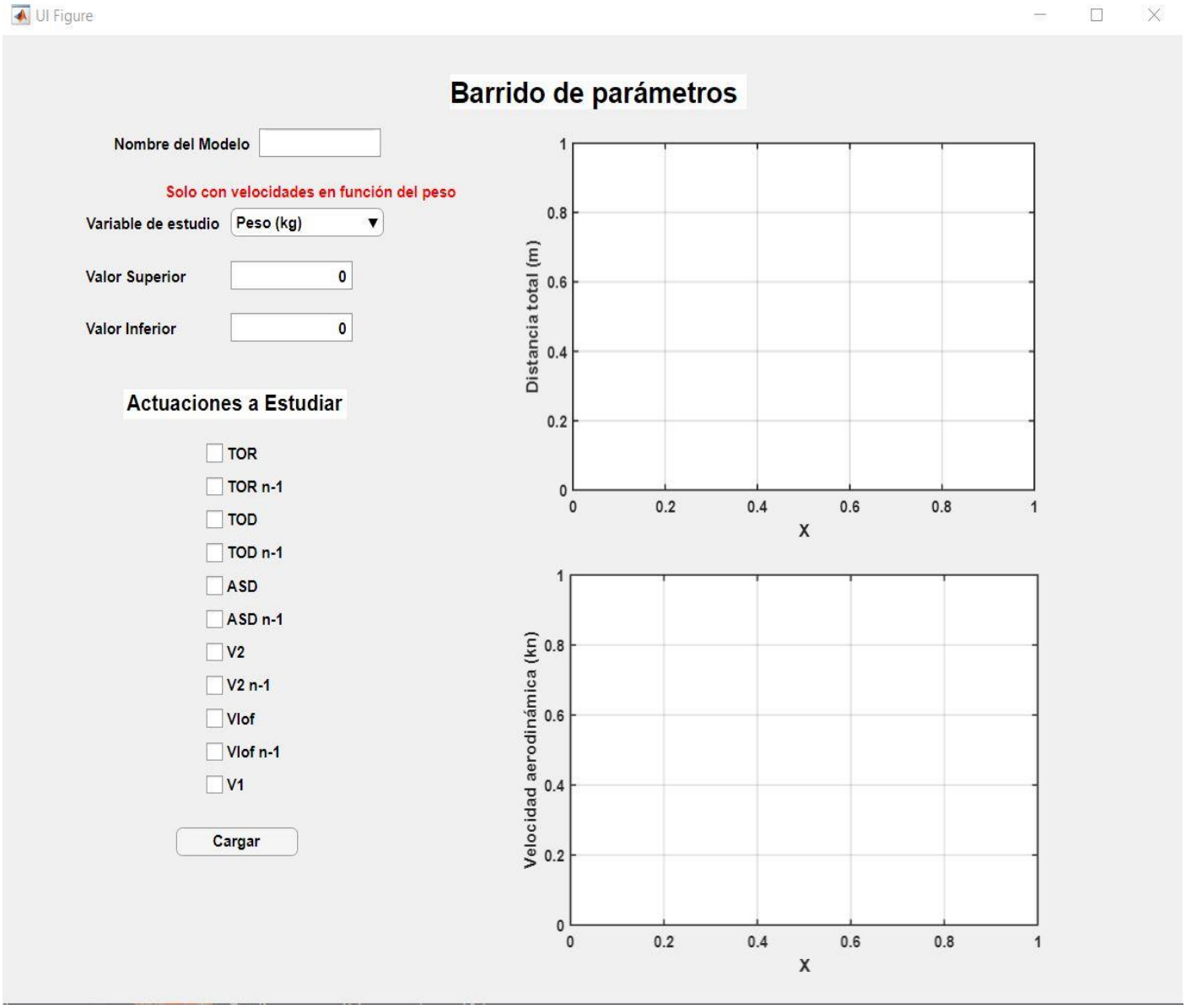


Figura 51. Ventana Procesar Resultados

Para las variables de velocidad de fallo de motor y de rotación hay una peculiaridad, y es que se debe cumplir que V_1 sea menor que V_R tal y como están definidas, por lo que se debe tener cuidado al introducir V_{EF}, ya que una velocidad de fallo de motor muy próxima o superior a la velocidad de rotación establecida en el modelo inducirá a errores.

Hay ciertas variables que, debido a cómo está creado el modelo, solo están disponibles según qué opciones hayamos escogido al crear el modelo. Por ejemplo, la opción de variar la pendiente solo está definida para la opción de pendiente de pista constante en la ventana crear modelo. No tendría sentido variar la pendiente cuando nosotros mismos hemos definido el perfil de la pista. Y así con otras variables también.

Para evitar estos problemas, se ha incluido una señalización en rojo justo arriba del panel de selección de variables que avisa de si hay alguna variable que no se puede usar.

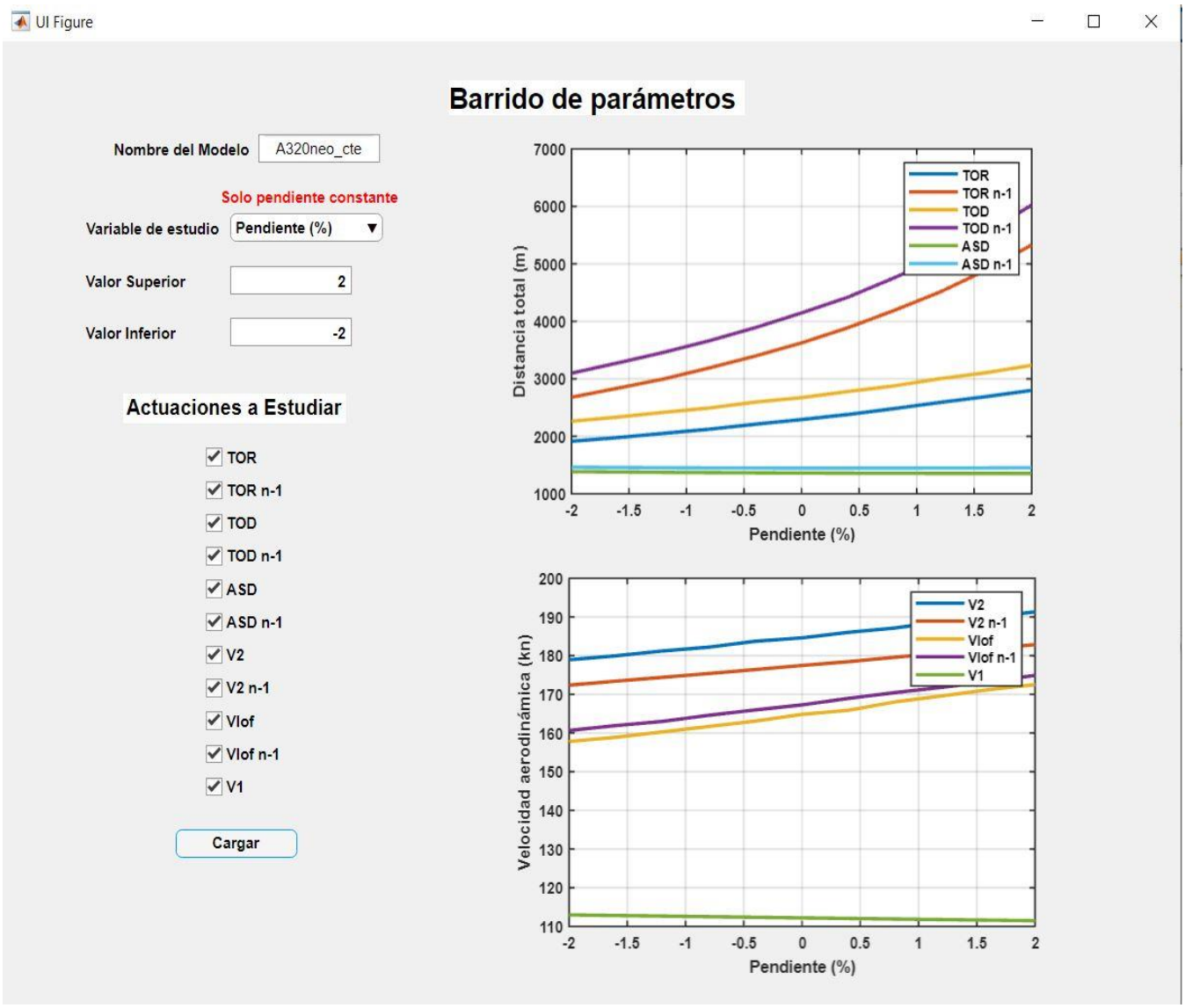


Figura 52. Ventana barrido de parámetros, ejemplo de pendiente

Las variables disponibles para hacer barridos son las siguientes.

- Peso (kg), solo disponible si hemos escogido la opción de las velocidades en función del peso.
- Velocidad de Rotación (kt)
- Pendiente (%), solo si hemos escogido la opción de pendiente constante.
- Velocidad de fallo de motor (kt)
- Velocidad de Viento (kt)
- Presion altitud (ft)
- Temperatura (ºC)
- Coeficiente de rozamiento
- Ángulo final de asiento (º), solo si hemos escogido ley de rotación personalizada.
- Tiempo total rotación (seg), solo si hemos escogido ley de rotación personalizada.
- Empuje nivel del mar Turbofán (N)
- Potencia nivel del mar Turbohélice (W)
- Cd0
- Cdflaps
- Cdgear
- K
- alpha no lift (º)
- Tiempo de reacción ante fallo de motor (seg)

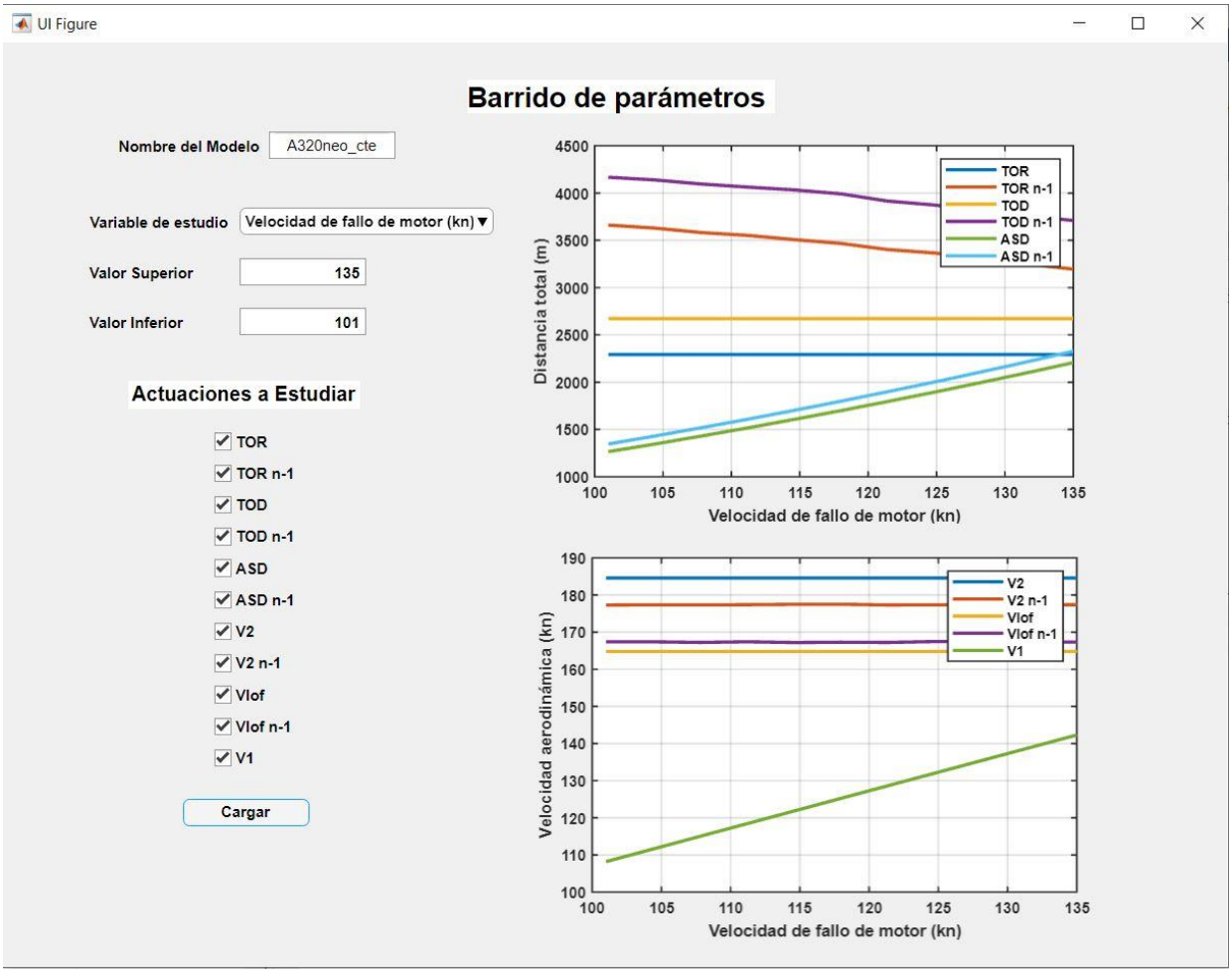


Figura 53. Ventana barrido de parámetros, ejemplo de velocidad de fallo de motor

Es importante comentar que estos resultados pueden variar muy rápidamente con el cambio de ciertas variables. Por ello hay que ser cuidadosos con que intervalos introducimos e interpretar los resultados que se muestran.

De todas maneras, si se introducen datos aproximados no debe ocurrir ningún problema. Y aunque se produzca un error, simplemente esto será señal de que debemos variar el intervalo introducido y listo.

Como conclusión de este capítulo, podemos decir que hemos cumplido con el cuarto objetivo marcado al comienzo del presente trabajo. Hemos implementado nuestro modelo en una aplicación informática que permite tanto el visualizado de datos como una opción para su análisis, de una forma amigable e intuitiva para el usuario.

A continuación, quedará probar o testear los resultados que da nuestra aplicación. Esto se hará a través de implementar unos ejemplos de aviones reales y contrastar el resultado ofrecido por la aplicación con el real de estos aviones.

5. CONTRASTE DE RESULTADOS MEDIANTE EJEMPLOS

Llegados a este punto, en el que ya tenemos la aplicación construida y en funcionamiento, probaremos cómo de precisos son los resultados que calcula y hasta dónde podemos llegar a analizar un modelo de avión. Como ya comentamos anteriormente, en este capítulo se mezclarán los objetivos 5 y 6 del trabajo, de modo que contrastaremos los resultados que proporciona la aplicación con resultados reales a través de dos ejemplos de aviones reales.

Elegiremos dos ejemplos de aviones populares en la aviación de transporte comercial, uno turbohélice y otro turbofán, y los probaremos con el programa.

La idea es saber cuáles son los puntos fuertes y los puntos débiles del programa, si es necesario realizar algún ajuste o si hay algunos puntos que simplemente, por estar definidos con aproximaciones, no funcionan demasiado bien y quedarán como posibles mejoras de futuro para añadir en las conclusiones del trabajo.

Empecemos primero probando el avión turbohélice.

5.1 ATR72

Figura 54. Imagen de un ATR72-600

El ATR 72 es un avión comercial bimotor turbohélice utilizado para viajes regionales y trayectorias de corta duración desarrollado por el fabricante de aviones franco-italiano ATR.

Tiene una capacidad de unos 78 pasajeros y presume de ser uno de los aviones de transporte regional más eficientes con respecto a gasto de combustible. Por ello, es una de las opciones más populares entre las aerolíneas que buscan cubrir trayectos o distancias cortas.

La versión ATR 72-600 es la versión más avanzada del modelo, implementando los últimos avances tecnológicos del momento a la vez que mantenía las virtudes del modelo.

Para comenzar el análisis del despegue de esta aeronave, buscamos sus características técnicas que vienen especificadas en la página web de su fabricante _[13] y _[14]. Las características que no encontremos tendremos que aproximarlas o buscar características típicas de aviones similares.

- $Peso\ máximo\ en\ despegue \sim 23000\ kg$
- $b = 27.05m$
- $S_{alar} = 61m^2$
- $\Delta = 0$
- $Potencia\ de\ despegue\ máxima \sim 2000kW$
- $Ángulo\ de\ ataque\ del\ empuje\ \epsilon \sim -1\ grado$
- $Ángulo\ de\ asiento\ de\ referencia\ \theta_0 \sim -3\ grados$
- $Coeficiente\ de\ rozamiento\ \mu \sim 0.02$
- $Coeficiente\ de\ frenada\ \mu_{FRE} \sim 0.3$

Las características aerodinámicas con las que construimos el modelo aerodinámico son aproximaciones, por lo que el fabricante no proporciona dichos parámetros. Usaremos una tabla que viene en el libro _[8] para tener una idea de los órdenes de magnitud que tiene estos parámetros y hacer una estimación, aunque al ser datos tan específicos la precisión de los resultados se verá afectada.

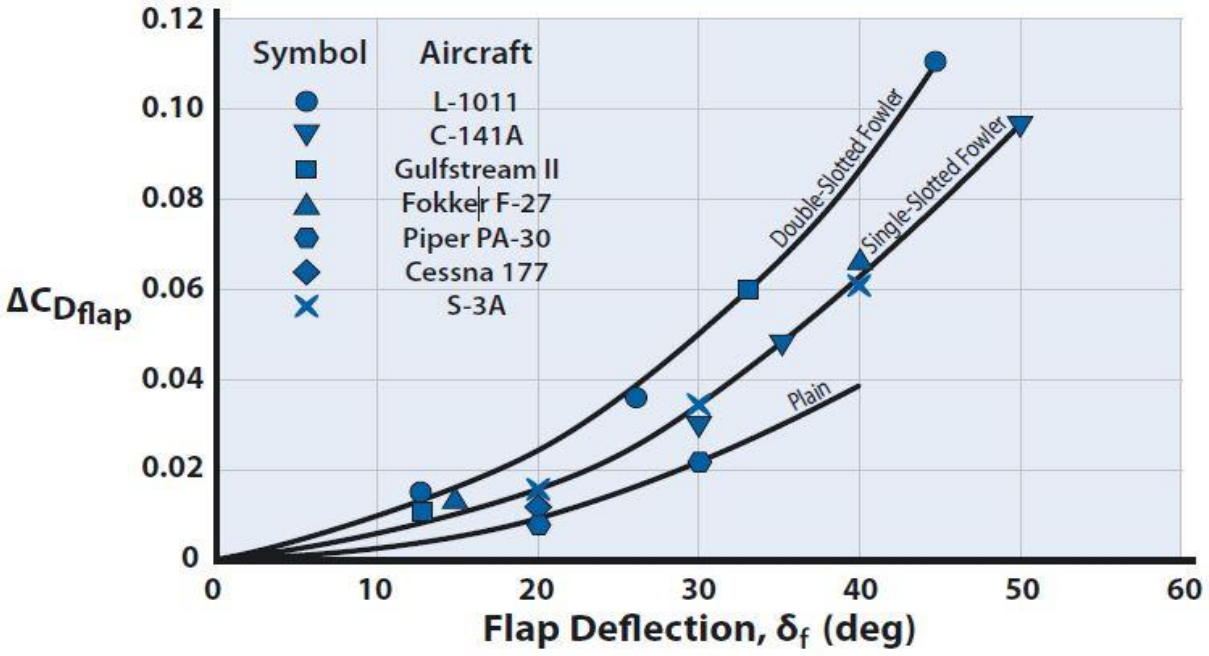


Figura 55. Gráfica para estimar el valor de ΔC_{Dflap} obtenida de [8]

El modelo más similar sería el Fokker F-27, otro bimotor turbohélice.

- $\Delta C_{Dflap} \sim 0.03$

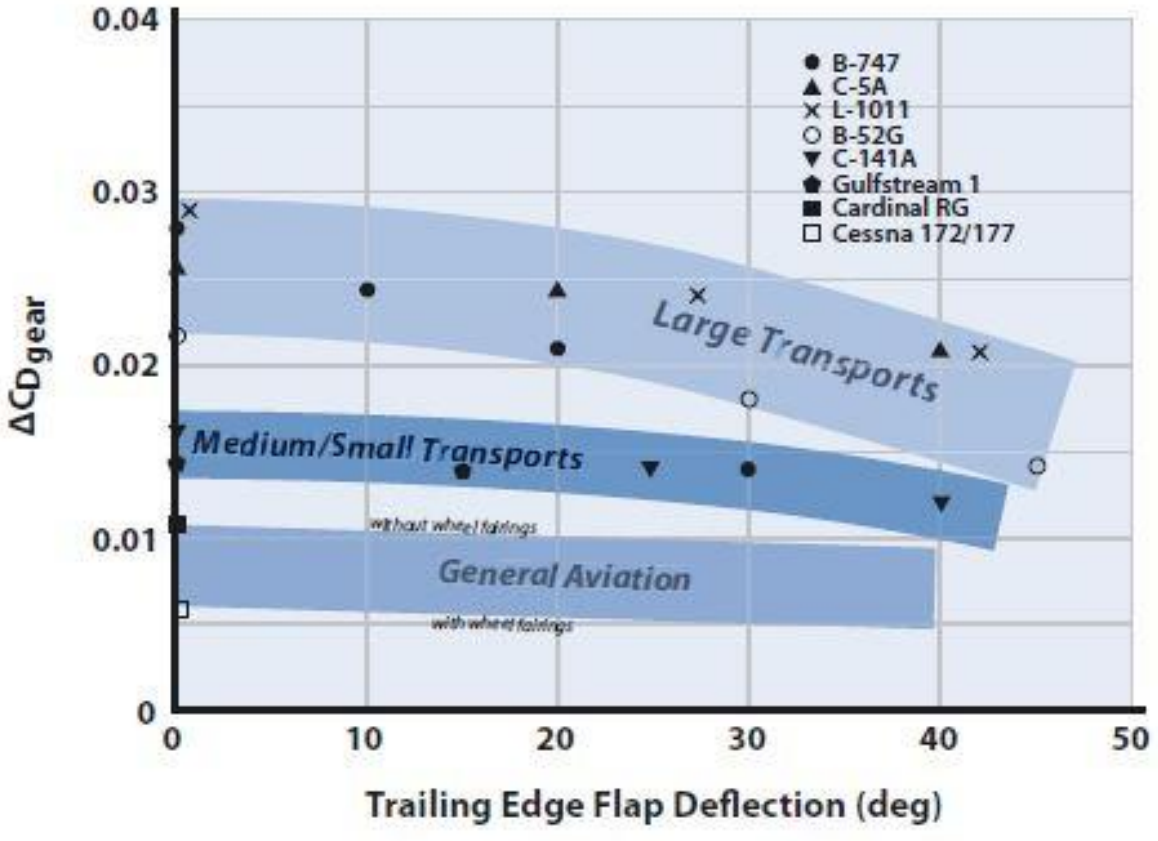


Figura 56. Gráfica para estimar el valor de ΔC_{Dgear} obtenida de [8]

Para la resistencia del tren de aterrizaje cogemos un valor promedio para aviones de tamaño medio.

- $\Delta C_{Dgear} \sim 0.025$

Table 5.2 Representative Values for Subsonic C_{D_0}

Aircraft Type	Subsonic C_{D_0}
High-subsonic jet transport	0.014–0.02
Supersonic fighter aircraft	0.014–0.022
Blended wing–body (tailless) jet aircraft	0.008–0.014
Large turboprop aircraft	0.018–0.024
Low-altitude subsonic cruise missile (high *W/S*)	0.03–0.04
Small single-engine propeller aircraft	
Retractable gear	0.022–0.030
Fixed gear	0.026–0.04
Agricultural aircraft	
With spray system	0.07–0.08
Without spray system	0.06
High-performance sailplane	0.006–0.01

Figura 57. Tabla con valores representativos de C_{D0} obtenida de [8]

Nuestro modelo en este caso se trata de un avión de transporte medio, por lo tanto, aproximando

- $C_{D0} \sim 0.03$
- $k \sim 0.035$

Por último, queda definir el bloque de velocidades. Esta información es bastante complicada de encontrar, ya que en la realidad estas velocidades dependen de muchos parámetros como la presión-altitud, la temperatura y otros más.

Sin embargo, intentaremos aproximar estas velocidades lo máximo posible, consultando manuales de vuelo y de actuaciones que proporcionan los fabricantes de la aeronave, y con esta información podremos construir un modelo aproximado y compararlo con las actuaciones reales.

Entonces, con la información que hemos podido encontrar sobre el ATR 72 en [15] y [16], tenemos que aproximadamente,

- $V_{MCG} \sim 88\ kt$
- $V_{MCA} \sim 97\ kt$
- $V_{SR} \sim 85\ kt$
- $V_{MU} \sim 95\ kt$
- $V_{EF} \sim 120\ kt$
- $T_{REC} \sim 1\ seg$
- $V_R \sim 130\ kt$

Con estos datos, podemos empezar a crear nuestro modelo.

Empezaremos analizando el caso de pista constante primero, y luego elegiremos algún aeropuerto como ejemplo para pista variable.

5.1.1 Modelo con pendiente constante

Partimos desde la ventana inicial y creamos nuestro modelo en la primera ventana, rellenando los parámetros pedidos.

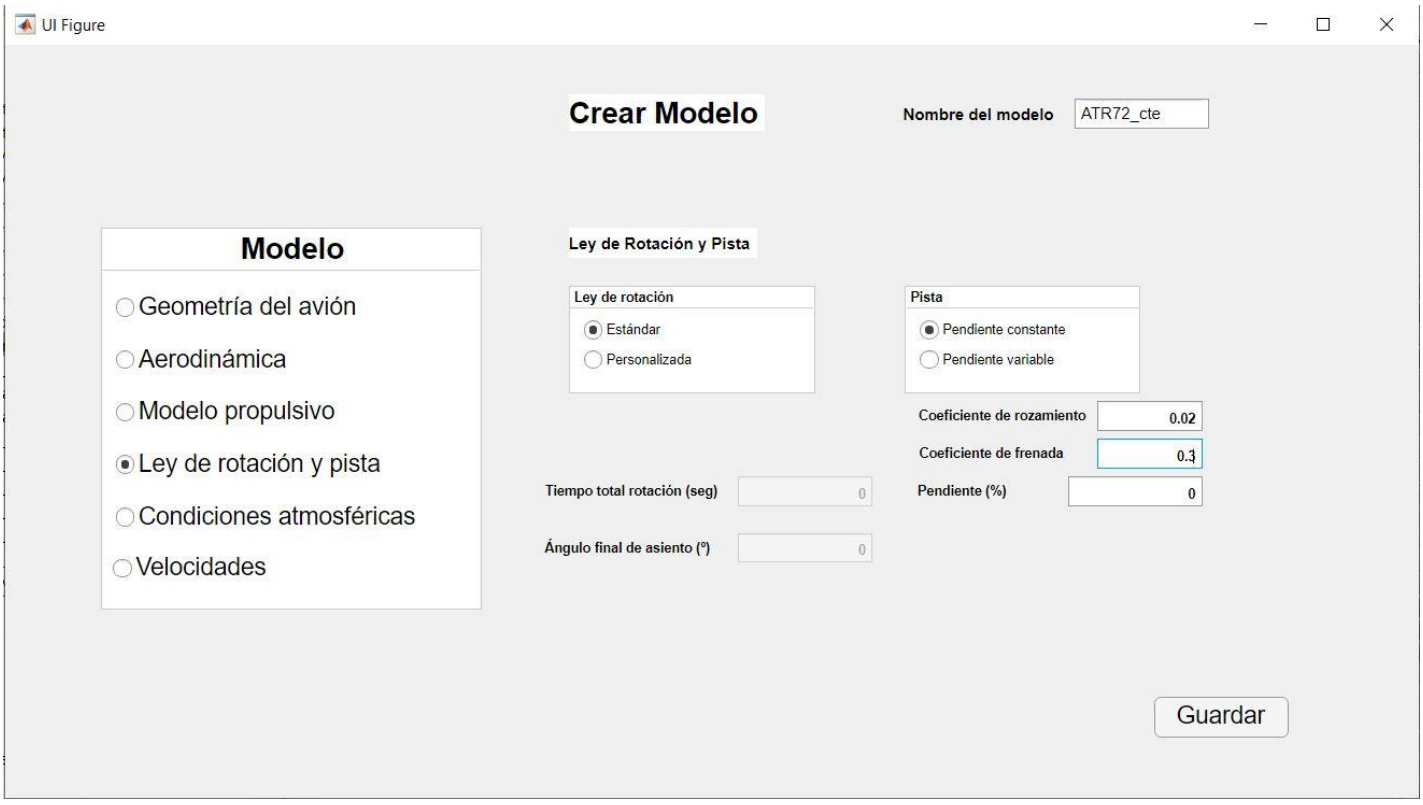


Figura 58. Modelo del ATR72-600

Introducimos los datos que hemos comentado anteriormente. Elegimos una ley de rotación estándar, que recordemos que trae definido unos parámetros estándar por defecto, y pendiente constante sin inclinación.

Luego podremos variar la pendiente en la ventana de barrido de parámetros si queremos ver cómo afecta a las actuaciones, pero para comparar con los resultados reales escogemos sin inclinación, ya que es la forma en la que lo proporcionan los manuales.

Introducimos también una presión- altitud distinta de la del nivel del mar

- $Presión - altitud = 1000\ ft$

Ya tenemos nuestro modelo creado, pero imaginemos que queremos cambiar algo porque nos hemos equivocado o por cualquier otro motivo. Desde la ventana inicial, abrimos la ventana modificar modelo, introducimos el nombre y cambiamos los parámetros deseados.

Finalmente, volvemos a guardar.

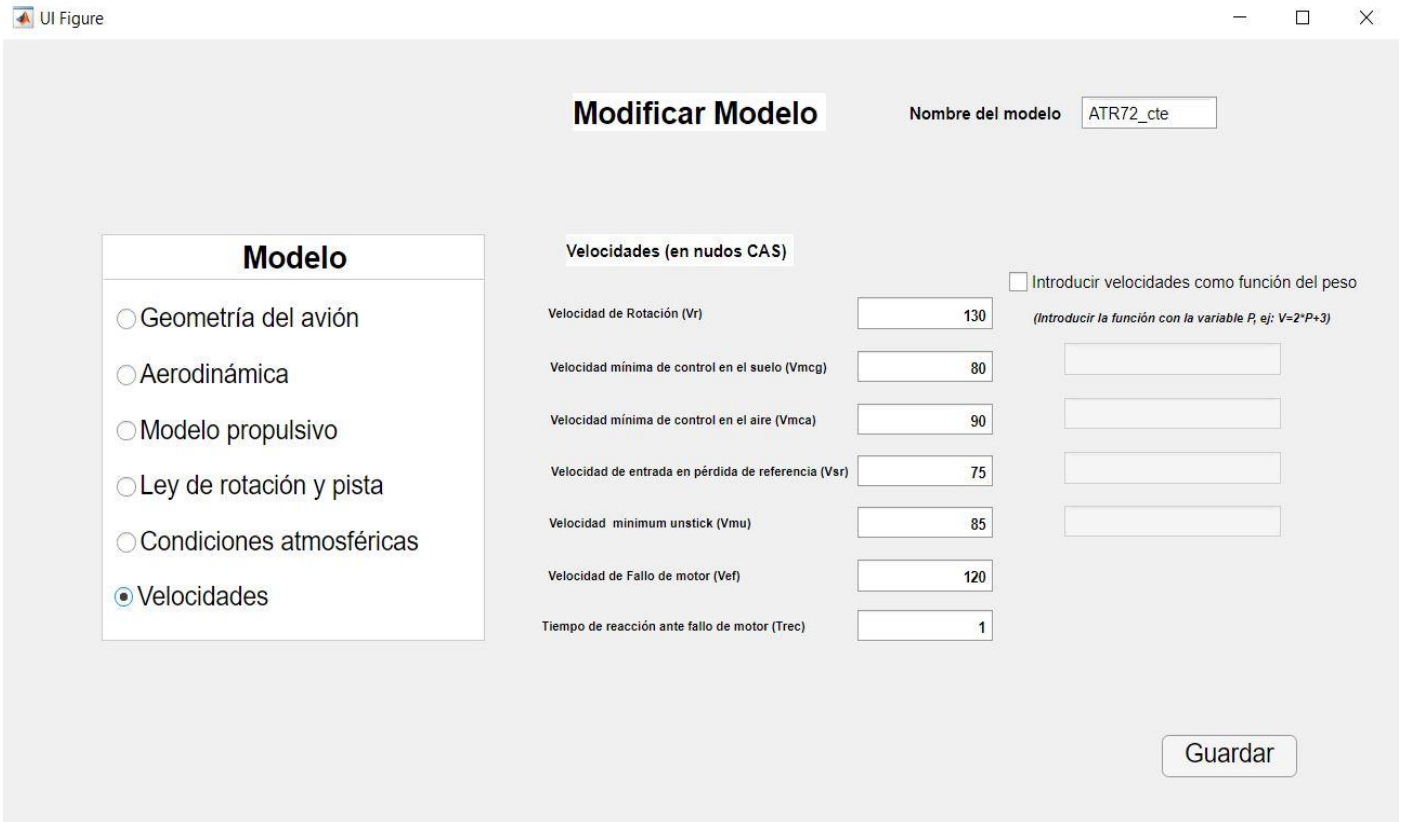


Figura 59. Modelo del ATR72-600 modificado

Con nuestro modelo creado y modificado a nuestro gusto, vamos a la ventana de procesar modelo y de nuevo, introducimos el nombre del modelo.

Le damos a procesar y si no hay ningún error, no aparecerá ninguna pestaña avisando y por tanto, nuestros resultados ya están calculados. Para comprobar esto, está la opción de visualizar el listado de modelos y sus resultados en la ventana inicial. Lo comprobamos.

Figura 60. Ventana de procesar modelo procesando el ejemplo del ATR72

Vemos que tanto nuestro modelo como sus resultados están guardados de forma correcta, por lo que podemos continuar y ver las actuaciones de nuestro modelo.

Figura 61. Listado de modelos y resultados para comprobar que los resultados se han guardado

Como ya se ha comentado antes, en esta ventana se pueden ver todos los resultados obtenidos, teniendo separados los más importantes y las restricciones normativas.

También se pueden ver los resultados de la integración en forma de gráfica o en forma de tabla.

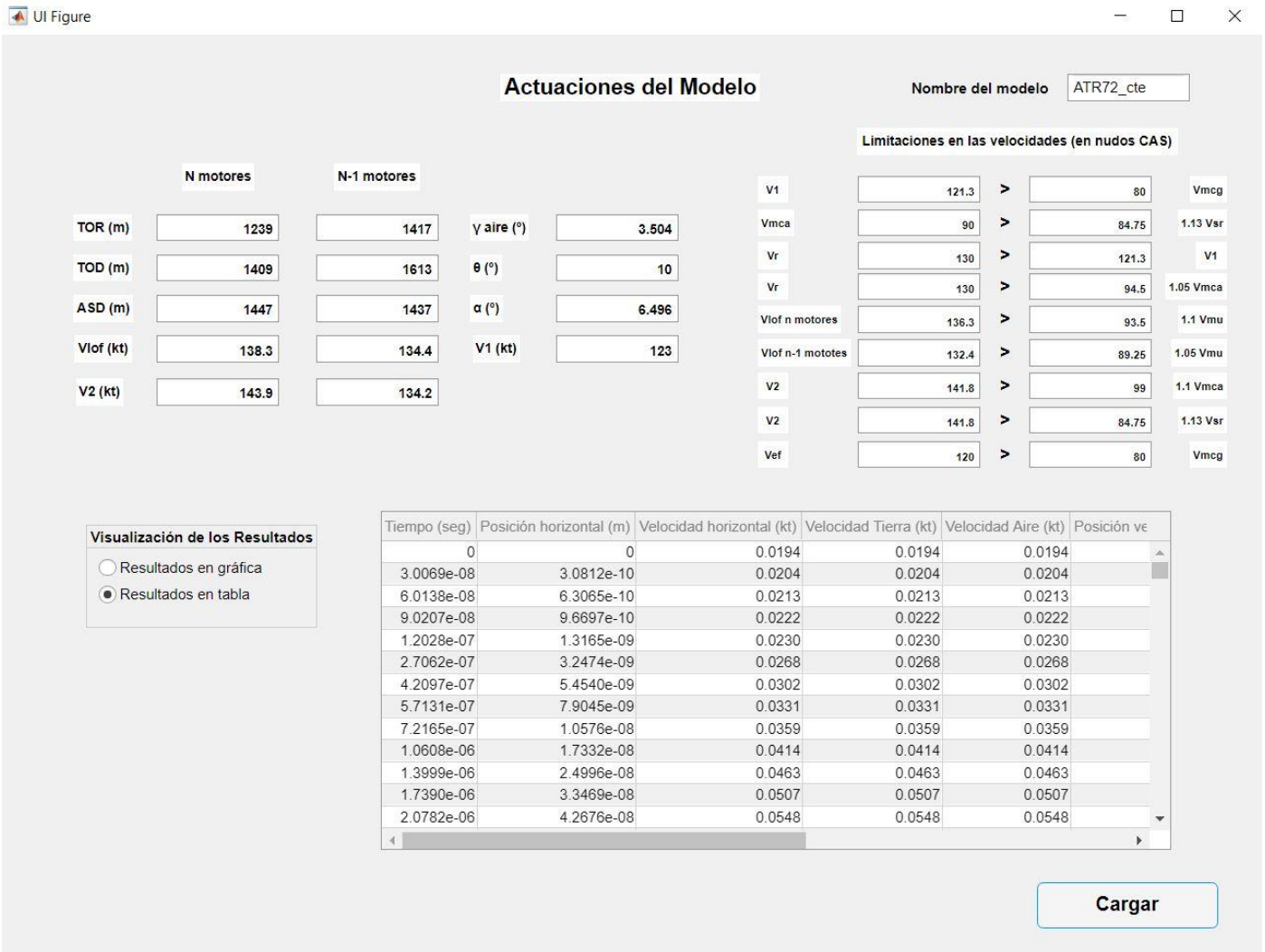


Figura 62. Ventana actuaciones del modelo

En este ejemplo, podemos ver la opción de los datos en forma de tabla. Vemos también como todas las condiciones normativas se cumplen, por lo que no ha habido ningún fallo y en principio todo ha funcionado.

Veamos ahora la opción de gráfica, mostrando en el eje x la posición horizontal y en el eje y la velocidad aerodinámica

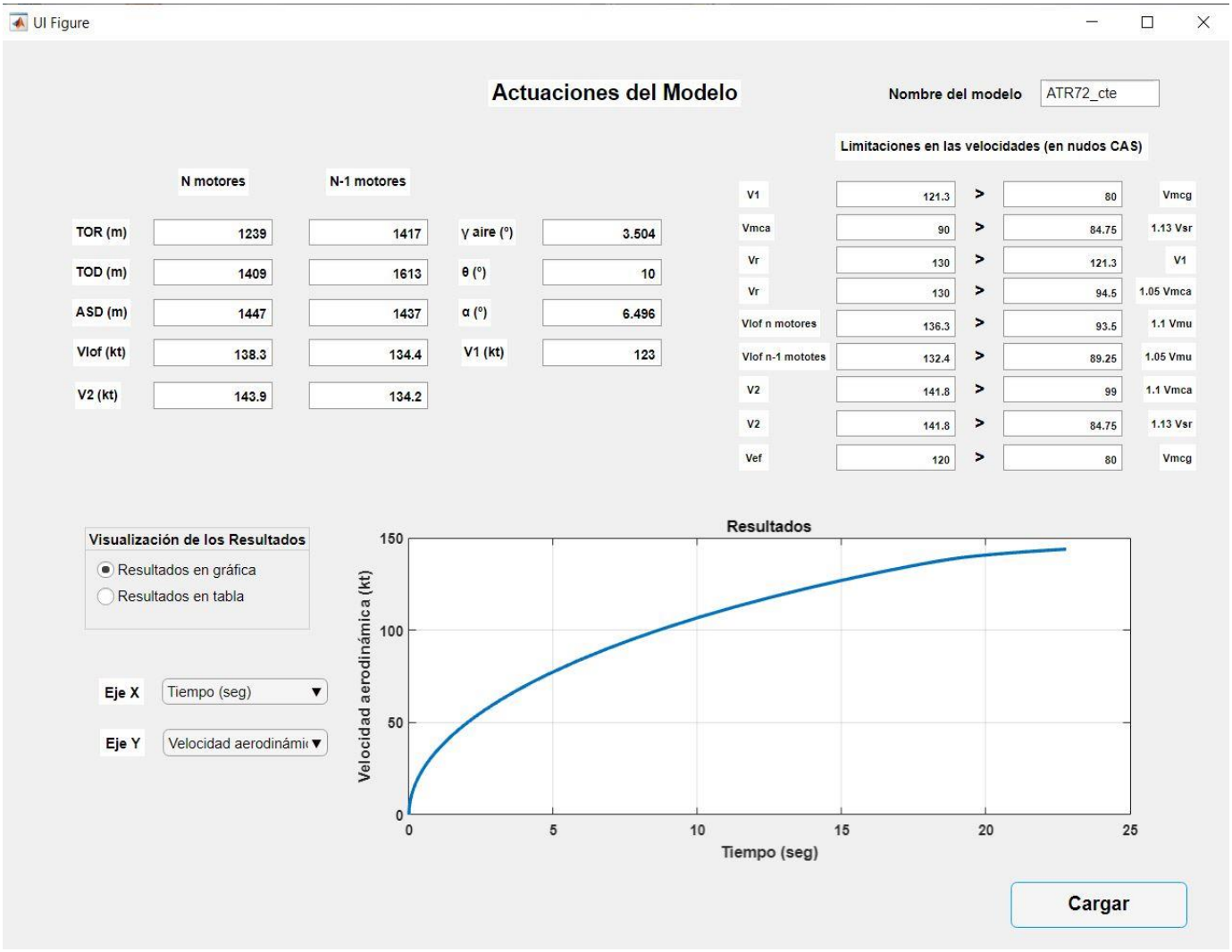


Figura 63. Actuaciones del modelo del ATR72

5.1.1.1 Ventana Barrido de parámetros

Una vez hemos terminado de ver los resultados, podemos pasar a la siguiente ventana del programa para estudiar las tendencias que siguen las actuaciones.

Estudiaremos con cada caso la variación de unos cuantos parámetros, para hacernos una idea del potencial de esta sección y, de paso, comprobar que los resultados son coherentes. Empecemos por ejemplo con la pendiente

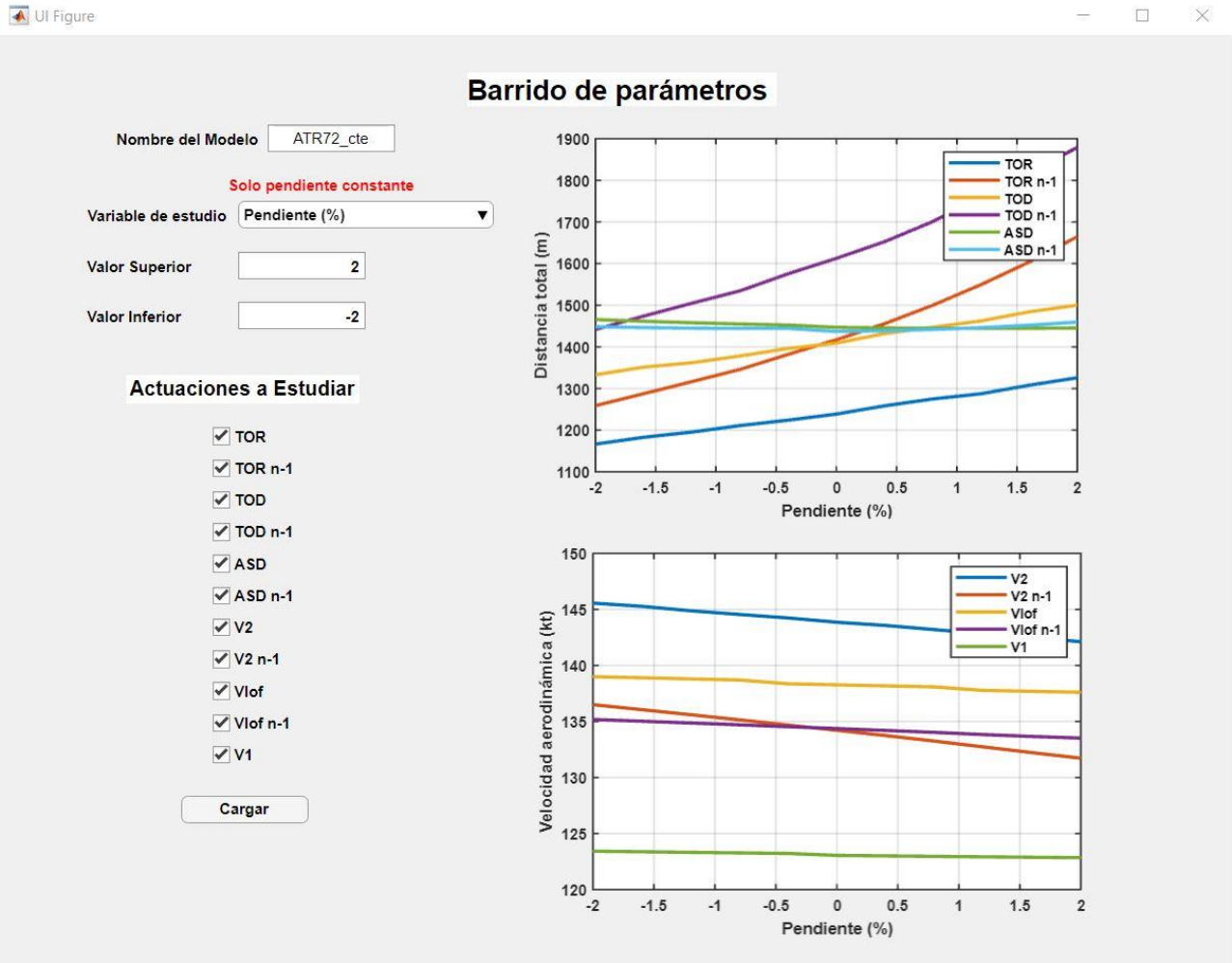


Figura 64. Barrido de parámetros, pendiente

Existe una clara relación directamente proporcional a la pendiente, cuanto menor sea la pendiente, mayor serán las distancias en general, y también las velocidades

Vemos además que la dependencia es bastante grande en las actuaciones con fallo de motor crítico TOR y TOD.

La ASD no varía mucho ya que, la pendiente afecta sobre todo a las fases en las que el avión tiene que elevarse o separase del suelo. Sin embargo, vemos que en a la derecha de la gráfica si empieza a aumentar levemente, puede deberse a que a partir de cierto punto perjudica más la parte en la que tiene que acelerar que el beneficio que puede tener al frenar. Las velocidades en general también disminuyen un poco.

Observemos ahora como variarían las actuaciones que dependen de la velocidad de fallo de motor. Recordemos que de esta velocidad depende directamente V_1, todas las actuaciones con fallo de motor y ambas ASD.

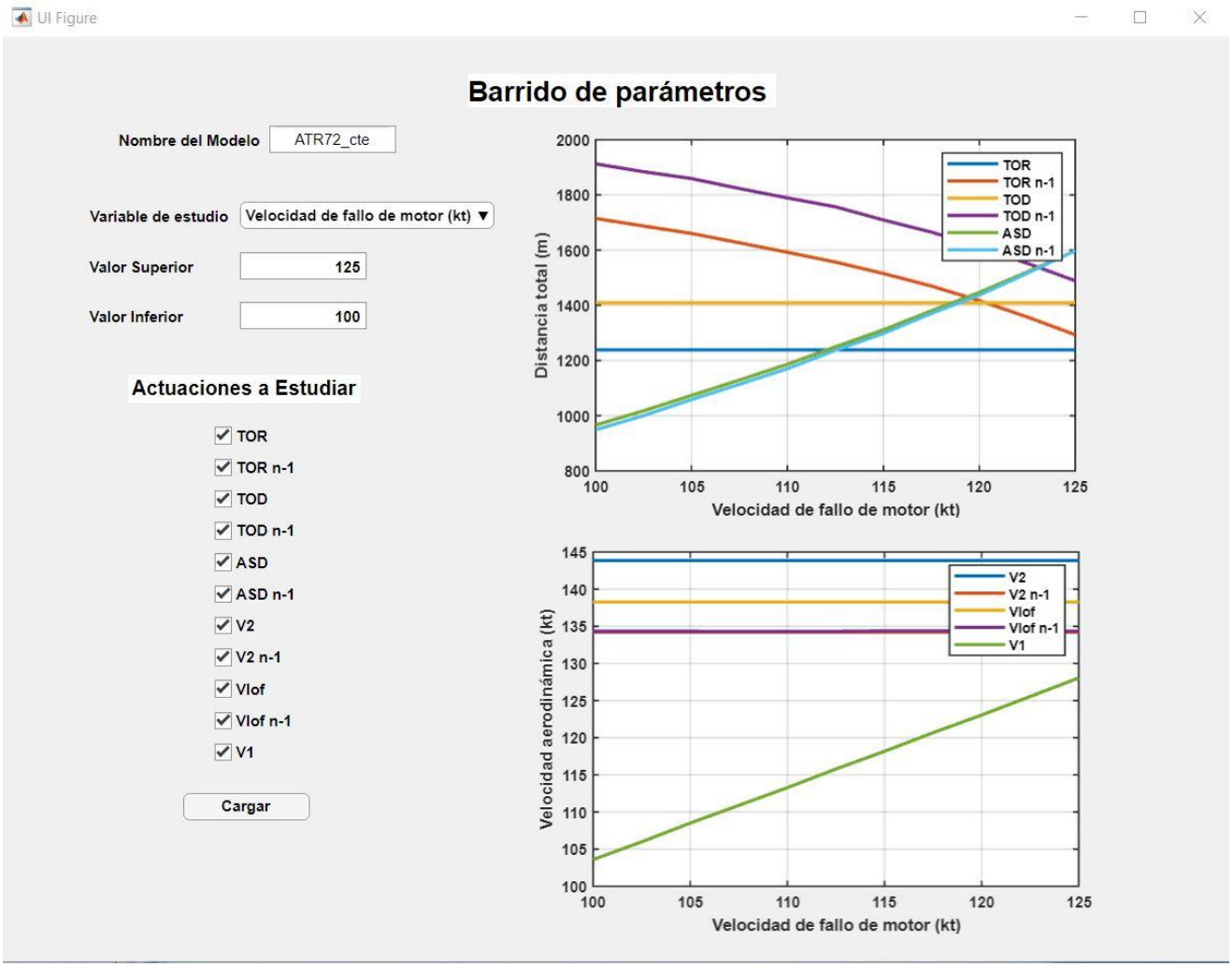


Figura 65. Barrido de parámetros, velocidad de fallo de motor

Vemos como los resultados son los esperados, a medida que aumentamos la velocidad de fallo de motor, aumentan tanto la velocidad de decisión como las dos distancias de aceleración-parada.

Sin embargo, las actuaciones TOD y TOR con fallo de motor disminuyen. Esto se debe a que, al aumentar la velocidad de fallo de motor, cuando se corta el motor en estas actuaciones el avión tiene mayor velocidad, y, por tanto, tiene que acelerar menos para continuar el despegue, por lo que requiere menos distancia.

Las otras actuaciones no se mantienen constantes porque no depende de esta velocidad.

Esta gráfica es además muy interesante ya que de ella podemos extraer la velocidad V_1 para criterio de pista compensada, es decir, el punto en el que se igualan ASD y TOD_{n-1}.

Por último, para terminar con el modelo con pendiente constante, veamos cómo afecta algún otro parámetro al modelo. Por ejemplo, el coeficiente de rozamiento de la pista.

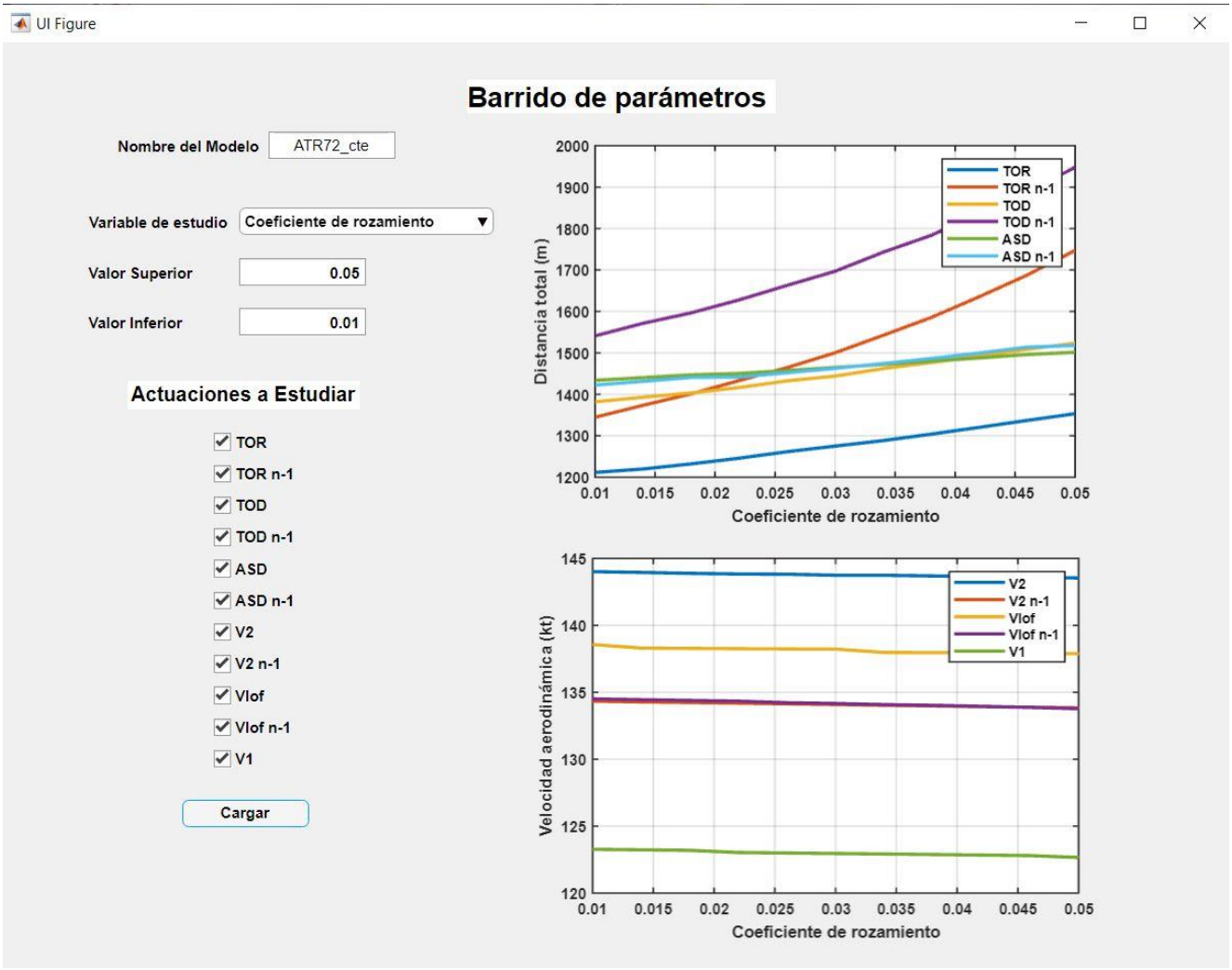


Figura 66. Barrido de parámetros, coeficiente de rozamiento

Podemos ver como las distancias sí que aumentan al aumentar el coeficiente de rozamiento. Claramente, ante un aumento de este coeficiente, mayor va a ser la fuerza de rozamiento y por tanto mayor dificultad va a conllevar acelerar, por lo que este resultado concuerda con lo que cabría esperarse.

Las velocidades no se ven demasiado afectadas por el aumento del coeficiente de rozamiento.

En los próximos apartados veremos más parámetros en esta ventana, algunos que por la definición del modelo no podíamos aplicar. De esta forma, repartiremos la visualización completa de esta ventana entre los dos aviones y los casos de pendiente variable y constante.

Continuemos ahora con el mismo avión, pero ahora definiendo una pista variable. Veremos cómo afecta esto a los resultados y si realmente es útil que el programa sea capaz de definir una pista variable.

5.1.2 Modelo con pendiente variable

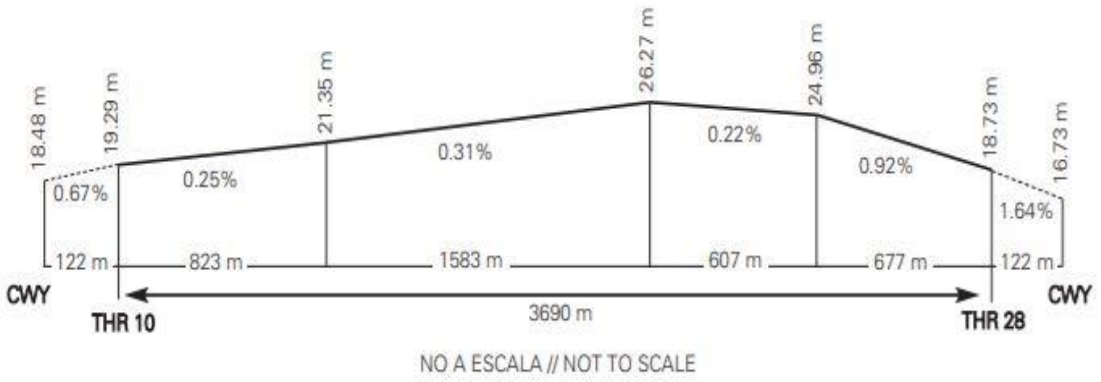
Una vez hemos analizado al completo este ejemplo con pista constante, vamos a hacer lo mismo con pista variable. Para esto, es interesante conocer los perfiles longitudinales de las pistas de los aeródromos de España, ya que de esta forma podemos probar nuestro modelo en aeródromos reales.

Podemos consultar todos los aeródromos de España en [17].

Por ejemplo, vamos a ver el aeropuerto de la base militar de Rota. Se puede consultar una gran cantidad de información sobre el aeródromo, pero nos fijaremos en lo que nos interesa para este caso, en el perfil longitudinal de la pista.

Observaciones: Ninguna.
Perfil:

Remarks: None.
Profile:



13. DISTANCIAS DECLARADAS		DECLARED DISTANCES		
RWY	TORA (m)	TODA (m)	ASDA (m)	LDA (m)
10	3690	3812	3735	3690
28	3690	3812	3735	3690

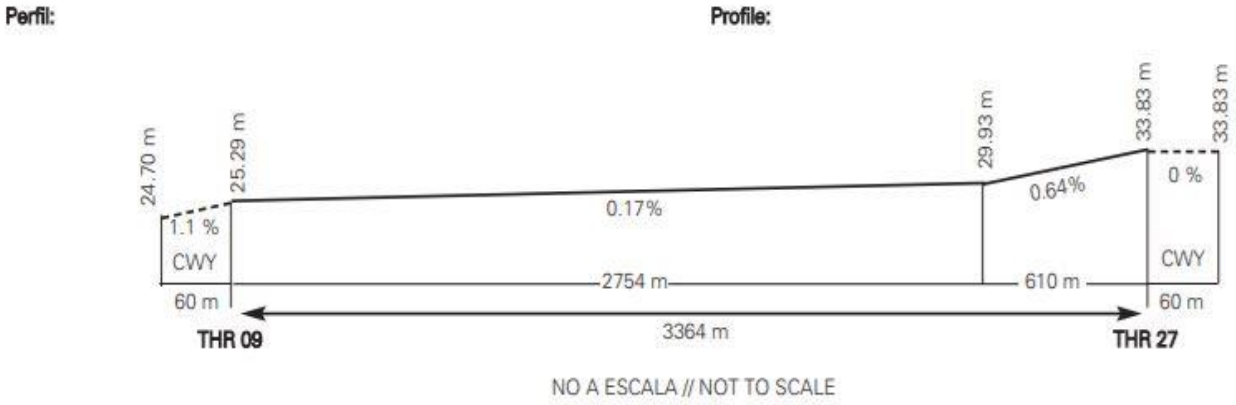
Observaciones: Ninguna.
Remarks: None.

Figura 67. Perfil longitudinal de la pista del aeródromo de Rota

Vemos que se definen pendientes, distancias, distancias declaradas, y, si existen, zonas libres de obstáculos (Clearway, CWY en el gráfico) y zonas de parada (Stopway).

Si recordamos la sección que explicaba la ventana de crear modelo, había dos formas de definir una pista, mediante distancias y alturas, o de la forma más habitual, con distancias y pendientes.

Miremos también, por tener otro ejemplo, el de Sevilla.



13. DECLARED DISTANCES — **DECLARED DISTANCES**

RWY	TORA (m)	TODA (m)	ASDA (m)	LDA (m)
09	3364	3424	3364	3364
27	3364	3424	3364	3364

Observaciones: Ninguna. **Remarks:** None.

Figura 68. Perfil de la pista del aeródromo de Sevilla

Vemos que todas las pistas se definen de forma similar, primero un perfil, y las distancias declaradas.

Una vez que hemos visto un par de ejemplos de aeropuertos, vamos a seleccionar el de Rota para implementarlo en nuestro modelo.

Empezamos igual que antes, en la ventana crear modelo. Todos los datos serán los mismos que los definidos al principio del apartado del ATR 72, a excepción de la altura inicial, que vendrá marcado por el perfil de pista, de la pendiente, que ahora será variable y también de la ley de rotación, que en este caso vamos a variar y a escoger personalizada.

Los datos de la ley de rotación van a ser los siguientes,

- $Ángulo\ final\ de\ asiento\ \theta_{final} = 10\ grados$
- $Tiempo\ total\ de\ rotación = 3\ seg$

Entonces, el bloque de ley de rotación y pista queda de la siguiente forma.

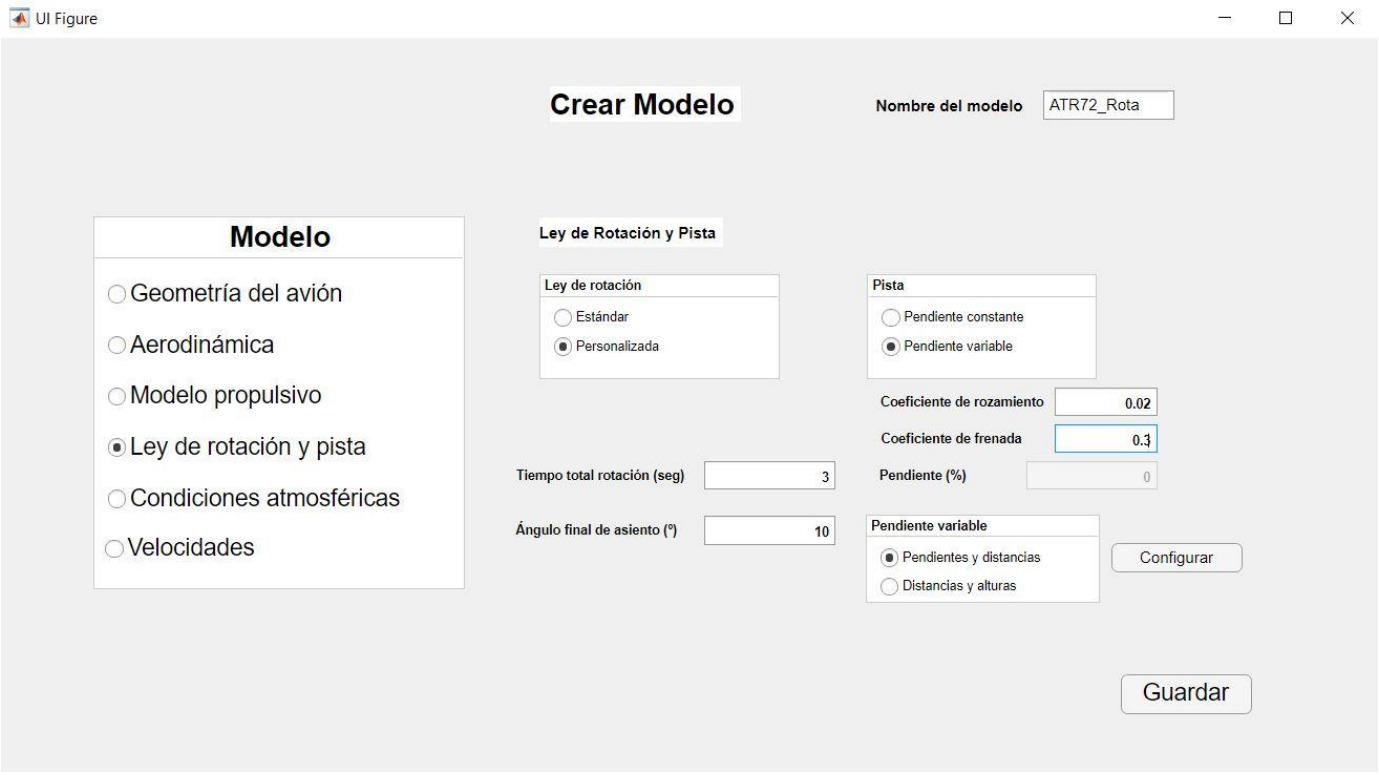


Figura 69. Modelo del ATR72 con pendiente variable

Seleccionamos en la opción de pista pendiente variable, luego elegimos como definir la pista, si mediante pendientes y distancias, o si mediante distancias y alturas.

Como hemos visto antes, elegimos la primera y le damos a configurar.

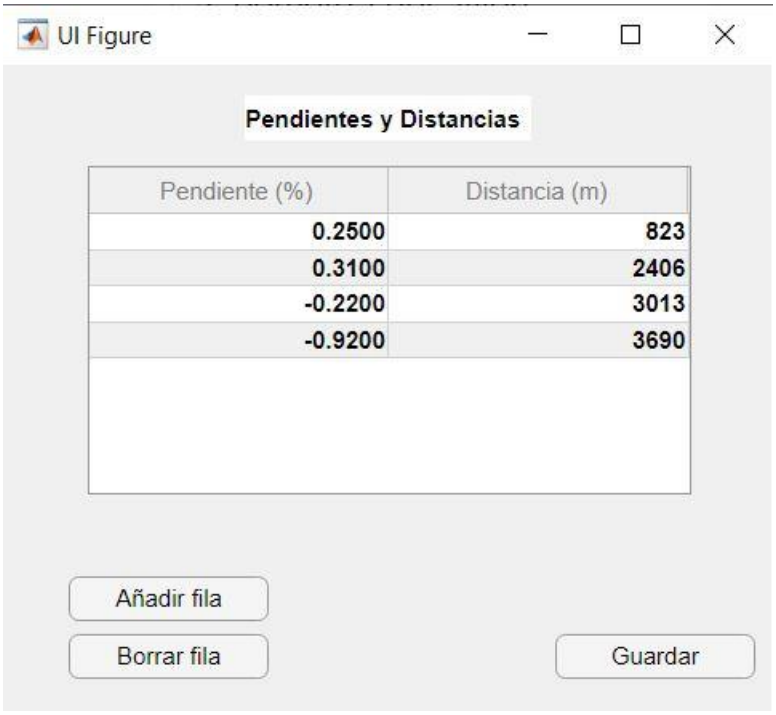


Figura 70. Definición del perfil de la pista del aeródromo de Rota

Le damos a guardar en la ventana de pendientes y distancias y volvemos a guardar el modelo. Omitiremos ahora la opción de modificar el modelo y pasamos directamente a procesar el modelo.

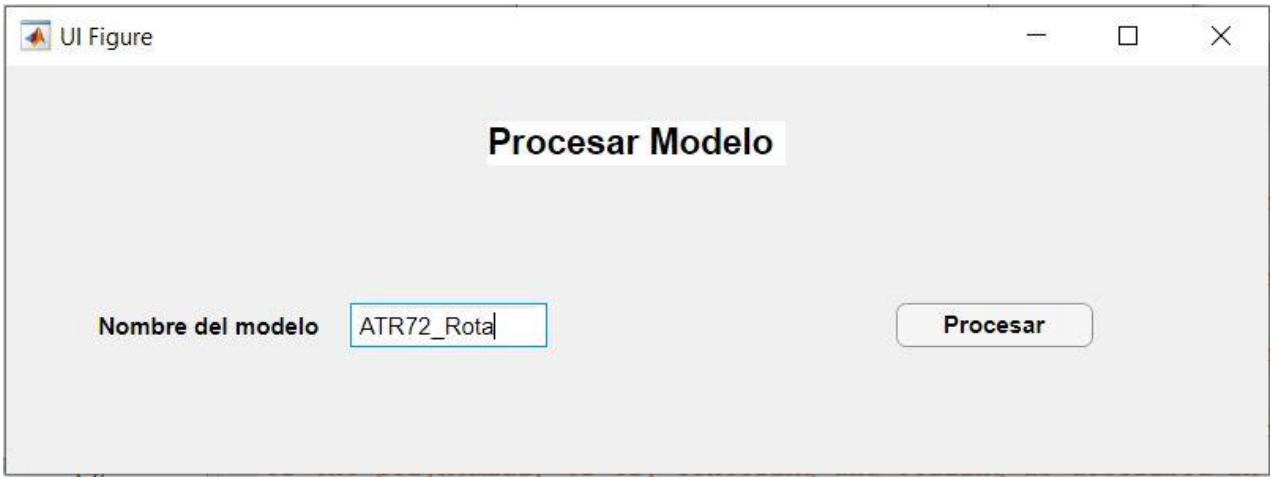


Figura 71. Procesado del modelo del ATR72 con la pista de Rota

Comprobamos que los resultados se han procesado correctamente, que no aparecen alertas sobre incumplimiento de las condiciones normativas, y que aparece en el listado de la ventana principal.

Figura 72. Listado de modelos y resultados incluyendo los del modelo con pendiente variable

Vamos a ver como ha quedado el perfil longitudinal de la pista.

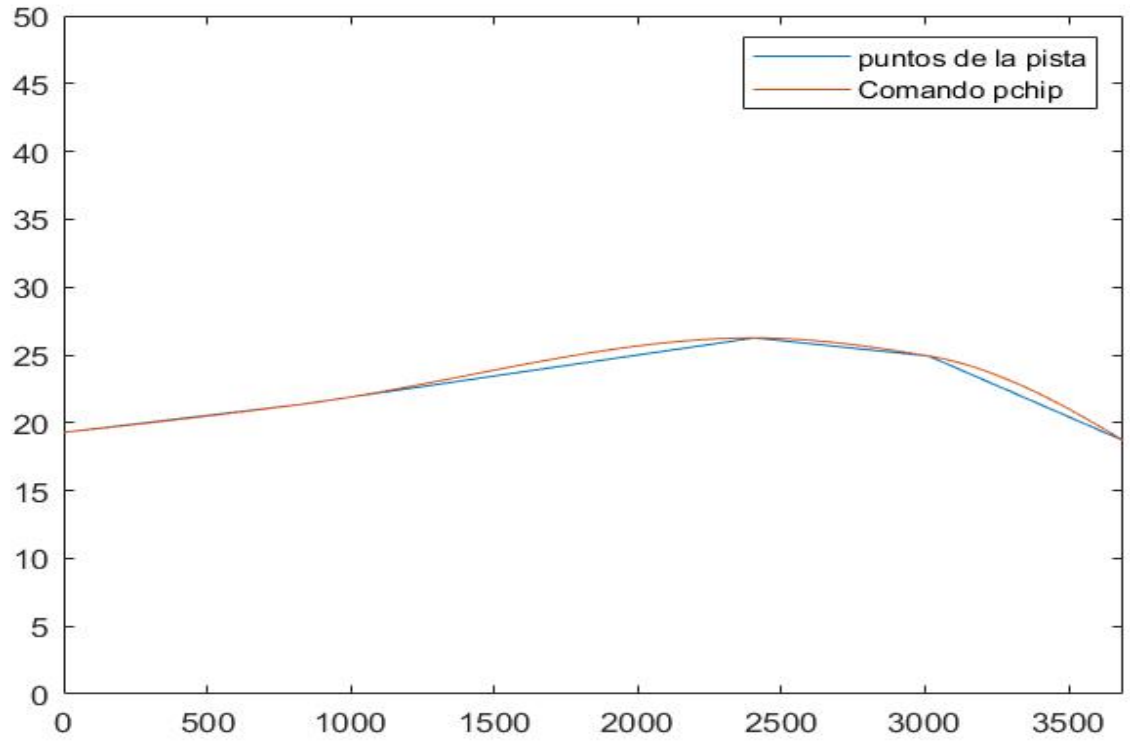


Figura 73. Aproximación del perfil de pista del aeródromo de Rota

Vemos que nuestra interpolación es bastante aproximada a la realidad.

Además, se consigue el efecto deseado al introducir el comando ‘’pchip’’, y es que en los tramos que presentan picos, este comando logra siempre que la solución sea suave y no se produzcan cambios bruscos, lo que ayuda a que las aproximaciones que usamos para las derivadas tengan menos error.

Después de comentar esto, pasemos a ver los resultados de nuestra simulación en la ventana de actuaciones del modelo y comprobemos si, con este modelo conseguimos acercarnos más a las actuaciones reales que hemos encontrado.

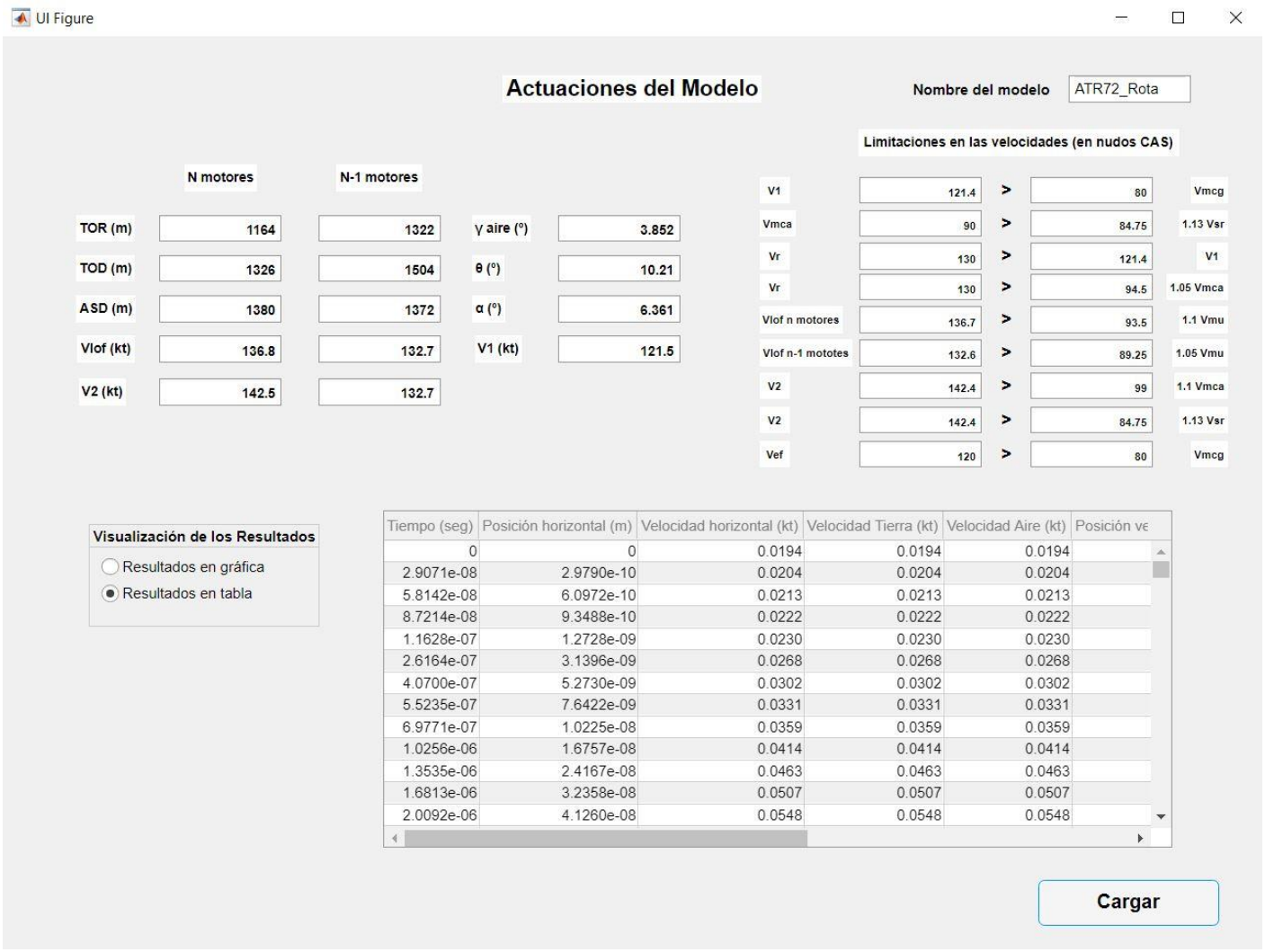


Figura 74. Ventana actuaciones del modelo, ejemplo con tabla

Lo primero que observamos es que, efectivamente, una vez más se cumplen las condiciones impuestas por la normativa. En la figura anterior hemos escogido la opción de mostrar los datos en la tabla, de la misma forma la opción gráfica está representada en la siguiente figura

Destacar la importancia de los parámetros que se introducen en el modelo, pudiendo variar sustancialmente los resultados obtenidos, desde unos próximos a la realidad a otros algo más alejados.

En la siguiente imagen podemos ver representado el ángulo de pista, aprovechando que este modelo tiene pendiente variable.

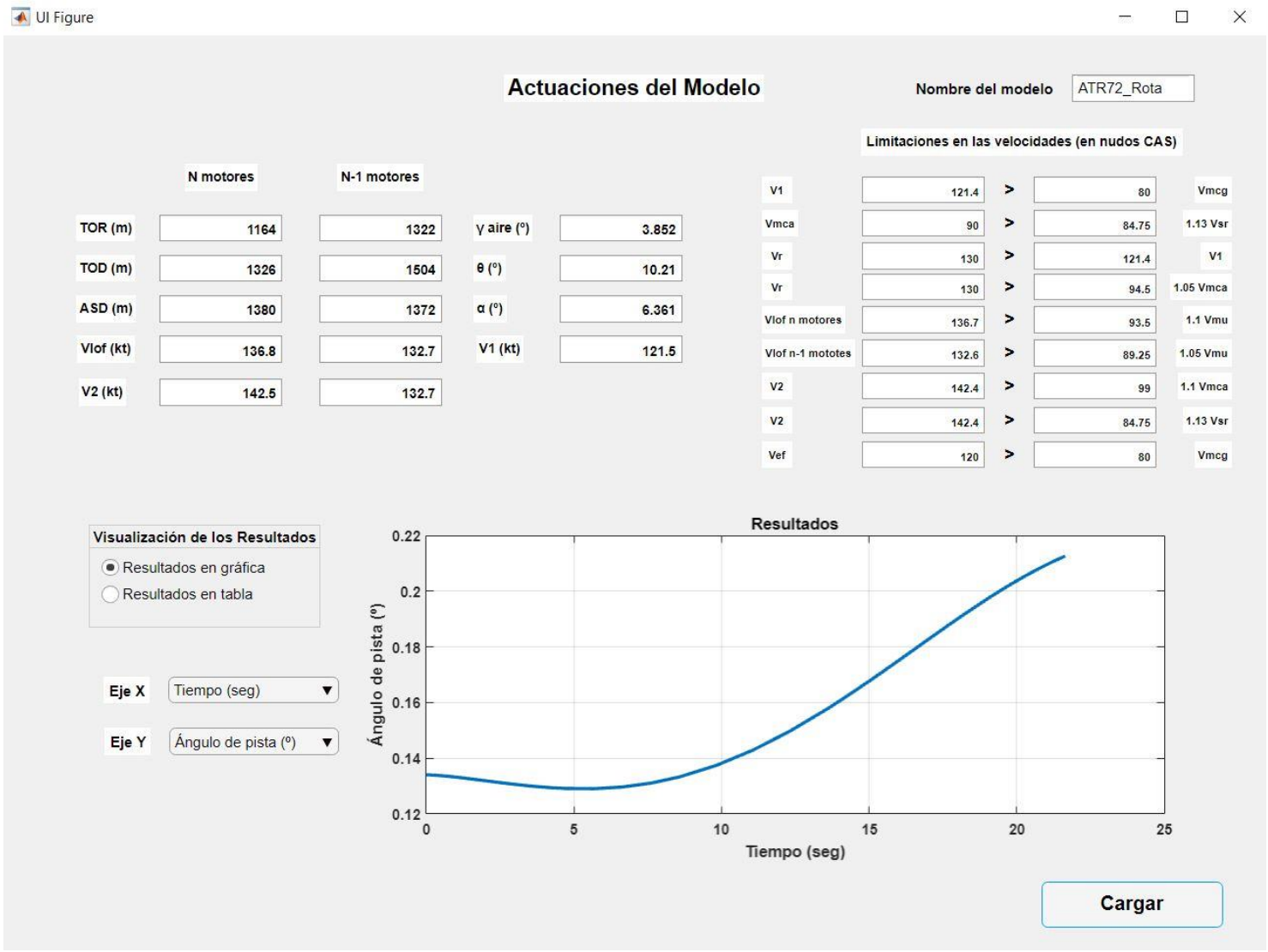


Figura 75. Ventana Actuaciones del modelo, gráfica

Si nos fijamos en la gráfica, y recordando la definición del perfil longitudinal de la pista, vemos que efectivamente salen resultados coherentes, siendo el ángulo de pista positivo en este recorrido, y no llegando a hacerse negativo porque el avión despega antes de llegar a la zona donde la pendiente se vuelve negativa.

5.1.2.1 Ventana Barrido de parámetros

Vamos a comprobar cómo influye el parámetro de ángulo de asiento final en los resultados finales.

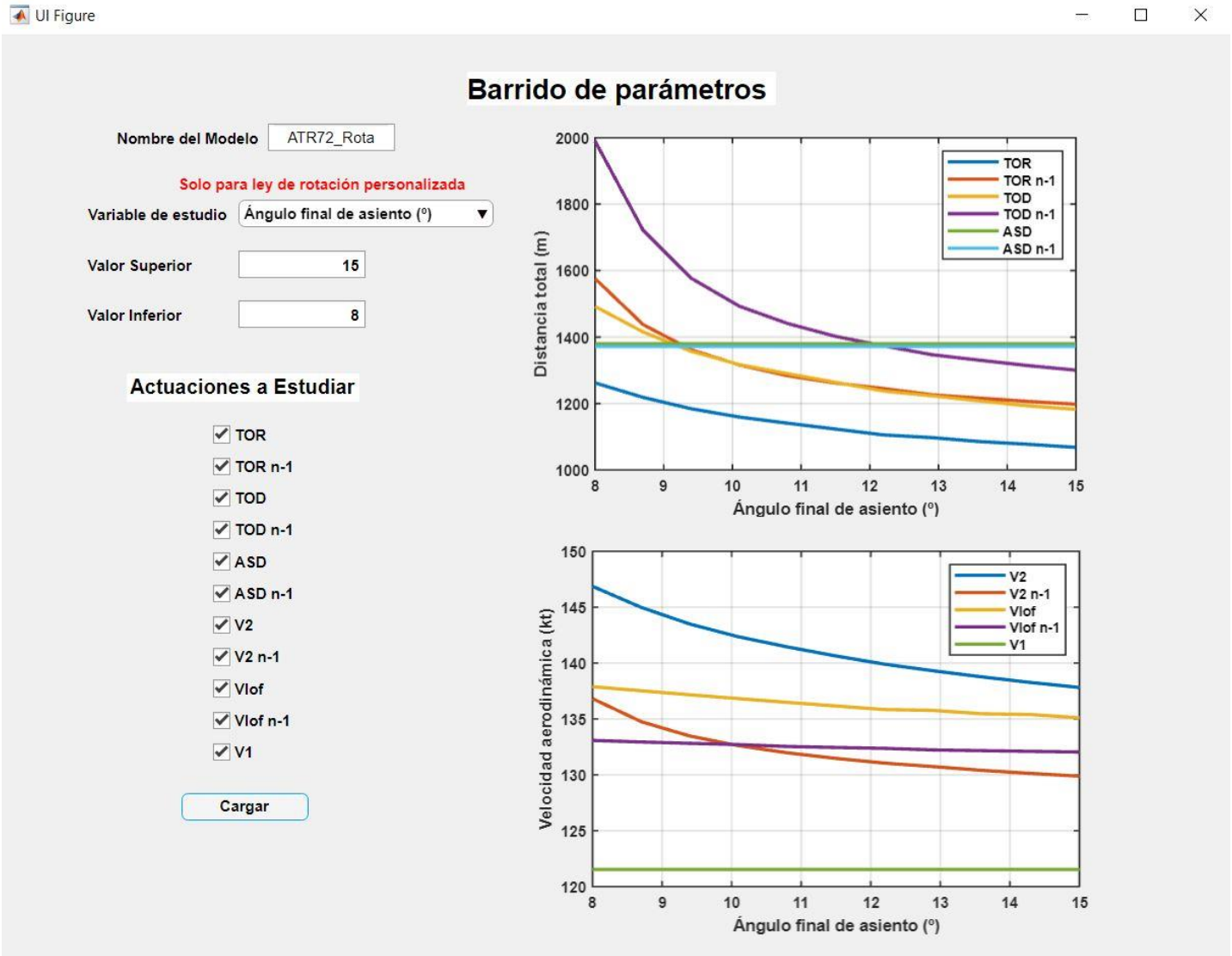


Figura 76. Ventana barrido de parámetros, ángulo de asiento final

Podemos ver que, con tan solo variar este parámetro unos pocos grados, la variación en los resultados es considerable, reduciéndose tanto las distancias como las velocidades. Esto se debe a que, al aumentar este ángulo, aumenta el ángulo de ataque y con ello la sustentación, por lo que el avión despegue en menor distancia.

Veamos en la siguiente imagen como afecta un cambio en la velocidad del viento.

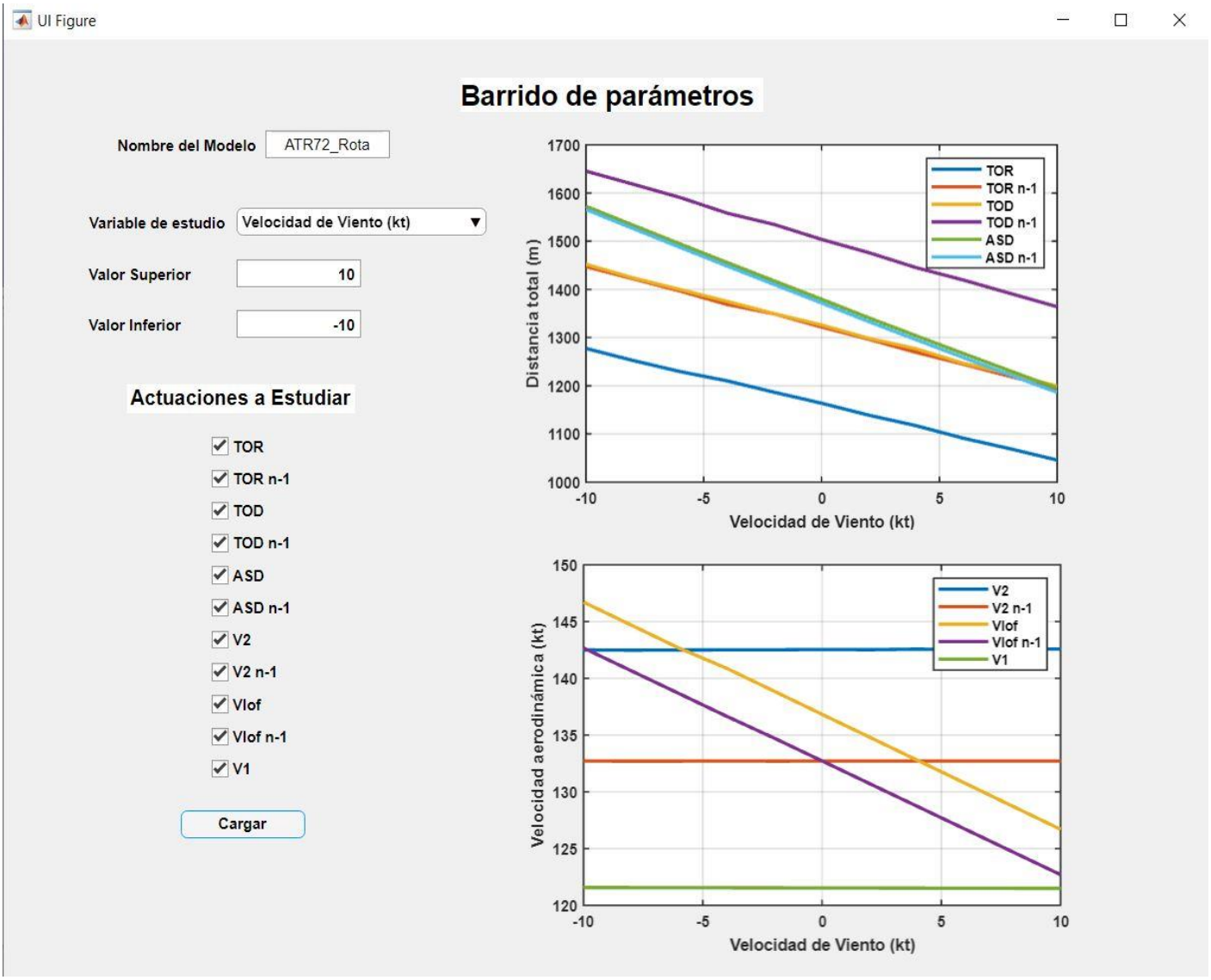


Figura 77. Ventana barrido de parámetros, velocidad del viento

Viendo la variación de los resultados y, recordando que un viento negativo significa viento de cola y viento positivo significa viento de cara, podemos comprobar como al aumentar la velocidad del viento, disminuyen las distancias. Con las velocidades, vemos cómo afecta mucho a la velocidad V_{lof}, pero no demasiado a las demás.

Con respecto a las distancias, parece intuitivo que al tener mayor viento de cara la distancia de despegue sea menor, ya que el avión ve más velocidad aerodinámica y por tanto tiene que conseguir menos velocidad por sí mismo.

Con respecto a las velocidades, al tener viento de cola estas aumentan ya que el avión se ve impulsado por este viento. Y ocurre al contrario con el viento de cara, las velocidades disminuyen porque es más difícil acelerar con estas condiciones. Aunque como se puede apreciar, la variación no es demasiado grande.

Veamos ahora como afecta la variación del tiempo de reacción del piloto a la velocidad de decisión V_1 y a las distancias de aceleración-parada ASD con todos los motores y con un motor inoperativo.

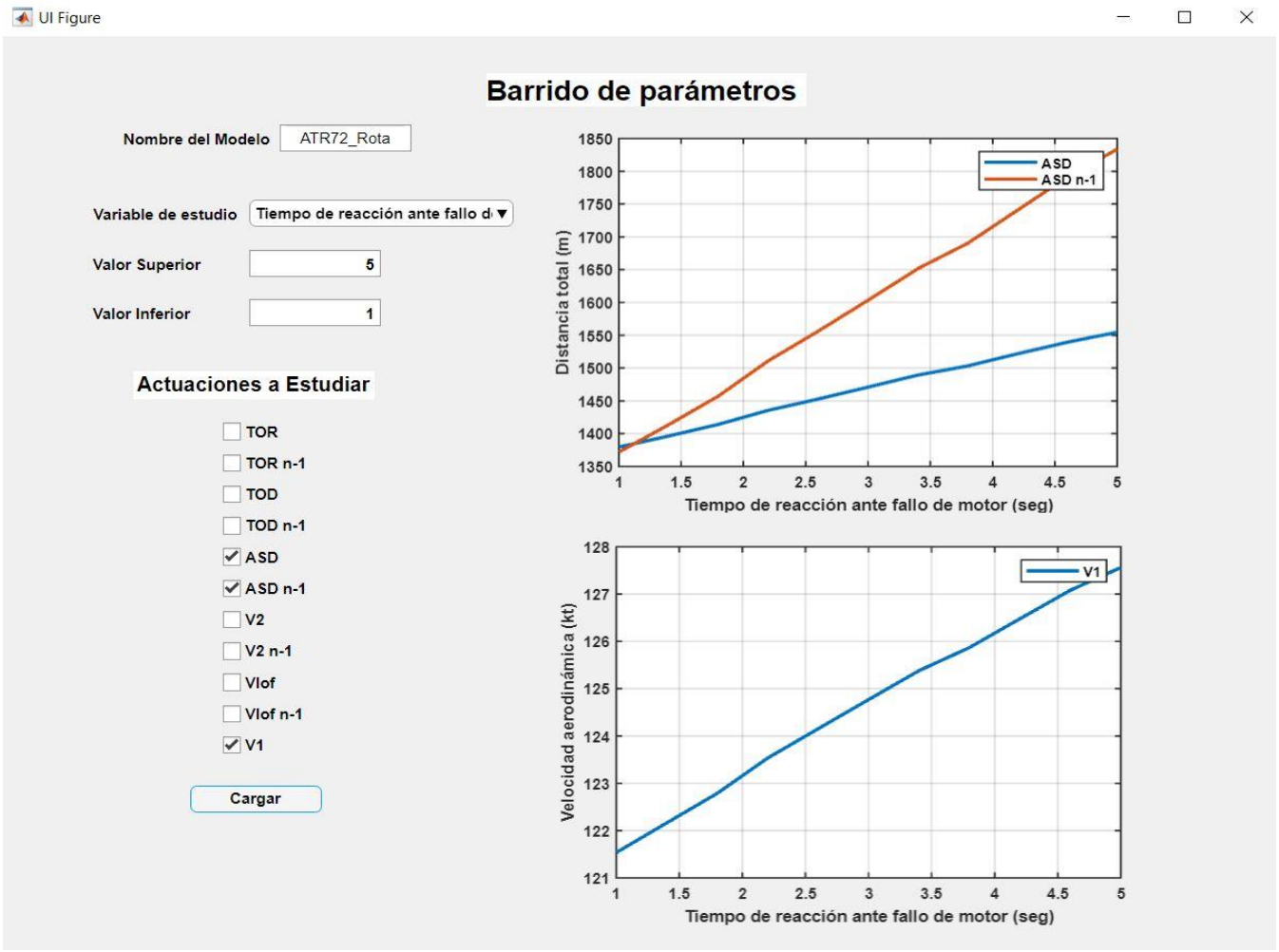


Figura 78. Ventana barrido de parámetros, tiempo de reacción

Aquí la tendencia es clara, cuando mayor sea este tiempo de reacción, mayor será la velocidad de decisión y la distancia de aceleración-parada, simplemente por tal y como están definidas.

5.1.3 Comparación de resultados con los reales

Para hacer las comparaciones con los valores reales, por un lado, veremos la distancia de despegue TOD que proporciona el fabricante (_[13]), y por otro lado veremos las velocidades típicas de despegue que constan en los manuales de vuelo y actuaciones de la aeronave (_[15], _[16]).

La TOD que proporciona el fabricante es de unos 1300 metros para máximo peso en despegue (unos 23000 kilogramos), para condiciones ISA a nivel del mar y con pendiente nula. Sin embargo, no nos dice para que velocidades de rotación o fallo de motor es esta distancia. Por lo tanto, asumiremos que esta distancia es la asociada a la V_1 que hace que TOD_{n-1} y ASD sean iguales (normalmente, los fabricantes proporcionan las actuaciones correspondientes a esta velocidad, llamada V_1 de 'Balanced field length').

Sin embargo, V_1 es dato de salida, no de entrada. El dato que nosotros podemos controlar es V_{EF}. Gracias a la ayuda de la opción de barrido de parámetros, encontraremos la velocidad de fallo de motor correspondiente al caso que estamos buscando. Una vez tengamos esta velocidad, podremos comparar de forma correcta las actuaciones del fabricante con las que proporciona nuestro programa.

Entonces, introduciendo los datos correspondientes a nuestro modelo, ajustando el resto de parámetros de entrada, y barriendo el parámetro V_{EF} tenemos,

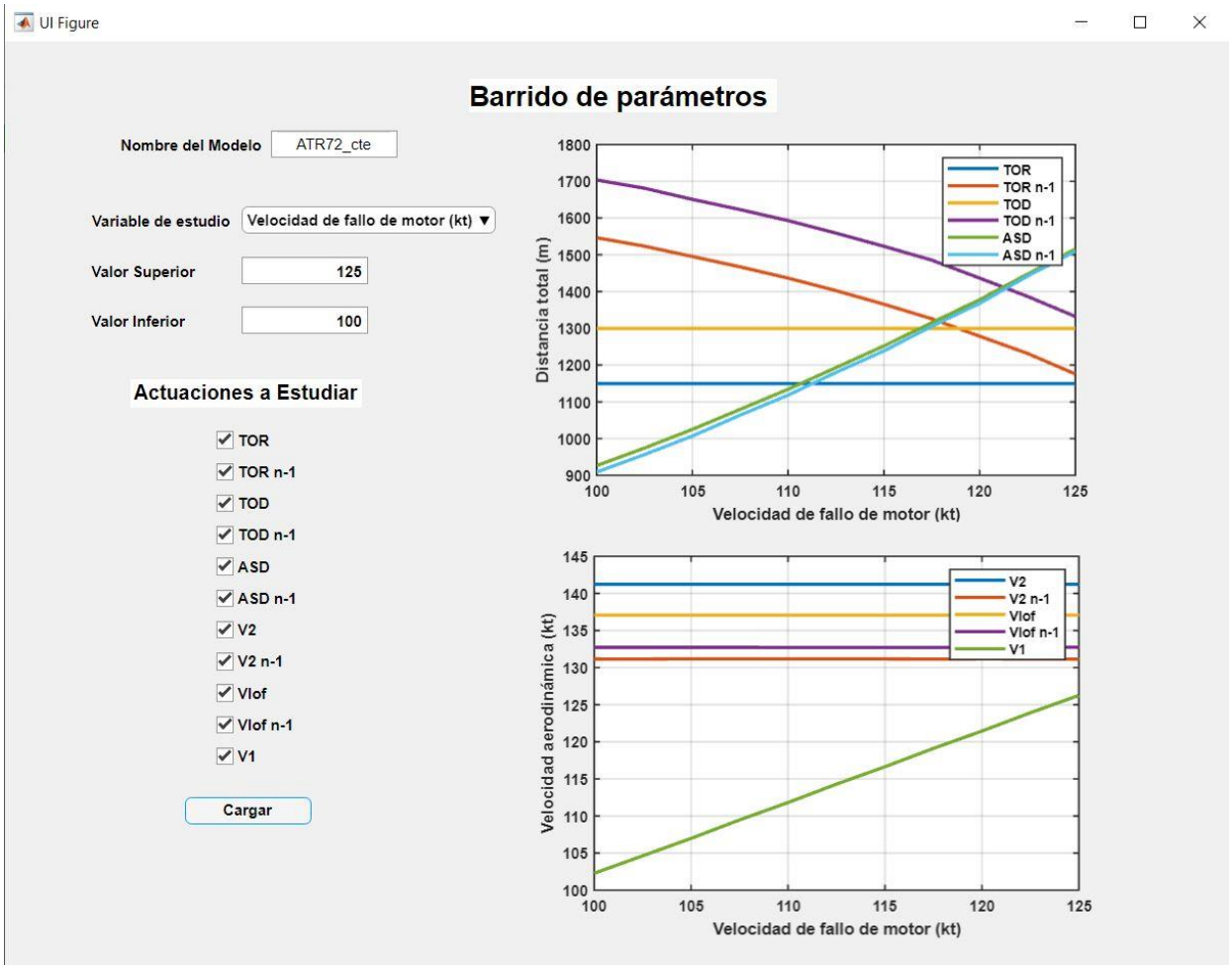


Figura 79. Ventana barrido de parámetros, velocidad de fallo de motor

Vemos que la velocidad de fallo de motor que buscamos está en torno a los 122 nudos, y que corresponden en la gráfica inferior a una V_1 de unos 123 o 124 nudos.

Si introducimos esta velocidad como parámetro de entrada y volvemos a procesar el modelo podremos comparar las actuaciones con las proporcionadas con el fabricante.

	Programa	**Reales**
TOD n motores (m)	1300	1279
TOD n-1 motores (m)	1395	

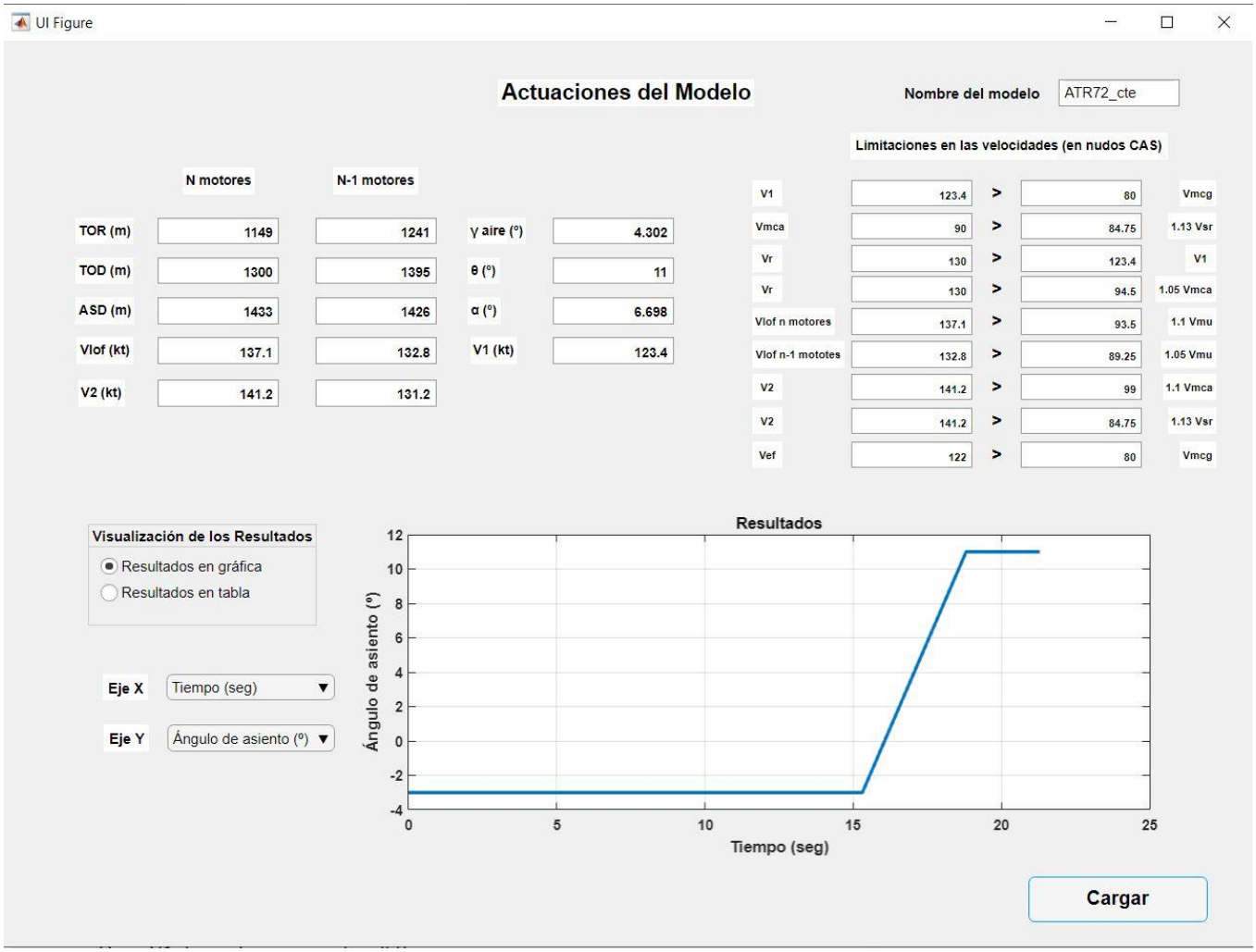


Figura 80. Ventana actuaciones del modelo

Vemos que nuestro resultado se ajusta bastante bien al proporcionado por el fabricante, lo cual es algo positivo. También es algo positivo a destacar la capacidad de análisis que ofrece el programa como acabamos de ver en este ejemplo, pues sin conocer a primera vista los parámetros de debemos introducir en el modelo, a base de ensayo y error y con la ventana de barrido de parámetros, podemos llegar a unos buenos resultados que ajusten nuestro modelo al real.

Del resto de las distancias no se ha podido encontrar ni siquiera una aproximación, por lo que nos tendremos que conformar con la aproximación de la TOD, aunque con esto nos basta para hacernos una idea de cómo funciona el programa.

Pasemos ahora a ver las velocidades en el despegue que se proporcionan en los manuales de vuelo. A diferencia de antes, estas velocidades se dan para una pista de una longitud definida de pendiente nula y a un peso específico, por lo que son diferentes a las obtenidas antes.

Veamos como vienen estas velocidades para unas ciertas condiciones en los manuales de vuelo

ATR72 F.C.O.M. | TAKE-OFF | 3.03.04 | P 2 | 500 | JUL 98

QUICK REFERENCE TABLES

AA

PRESSURE ALTITUDE ZP=0 FT– FLAPS 15
NORMAL CONDITIONS

T– CORRECTED E – RUNWAY M – LENGTH P(°C) – (M)	MAX TAKE–OFF WEIGHT (KG) – LIMITATIONS V1(IAS–KT)–VR(IAS–KT)–V2(IAS–KT) 1000 m	1100 m	1200 m	1300 m
–10.0	20304 3-3 104 104 109	21382 3-3 107 107 112	22356 3-3 110 110 115	23261 3-3 113 113 117
0.0	- - - - - - -	20896 3-3 106 106 111	21878 3-3 109 109 114	22779 3-3 112 112 116
5.0	- - - - - - -	20613 3-3 105 105 110	21598 3-3 108 108 113	22503 3-3 111 111 115
10.0	- - - - - - -	20333 3-3 104 104 109	21318 3-3 107 107 112	22227 3-3 110 110 114
15.0	- - - - - - -	20062 3-3 104 104 109	21044 3-3 107 107 111	21954 3-3 109 109 114
20.0	- - - - - - -	19791 3-3 103 103 108	20771 3-3 106 106 111	21680 3-3 109 109 113
25.0	- - - - - - -	19529 3-3 102 102 107	20504 3-3 105 105 110	21412 3-3 108 108 112
30.0	- - - - - - -	19271 3-3 101 101 106	20241 3-3 104 104 109	21145 3-3 107 107 112
35.0	- - - - - - -	19021 3-3 101 101 105	19984 3-3 104 104 108	20883 3-3 106 106 111
40.0	- - - - - - -	18595 3-3 99 99 104	19543 3-3 103 103 107	20434 3-3 105 105 110

Figura 81. Manual de vuelo ATR72 [16]

Vemos que estas velocidades vienen en función de la temperatura, la presión altitud, la longitud de la pista, y el peso. Introducimos estos parámetros en nuestro modelo y volvemos a buscar estos resultados, variando un poco nuestros parámetros de entrada. Llegamos a los siguiente.

	Programa	**Reales**
V_1 (nudos CAS)	106.8	107
V_2 (nudos CAS)	124.8	111
V_R (nudos CAS)	107	107

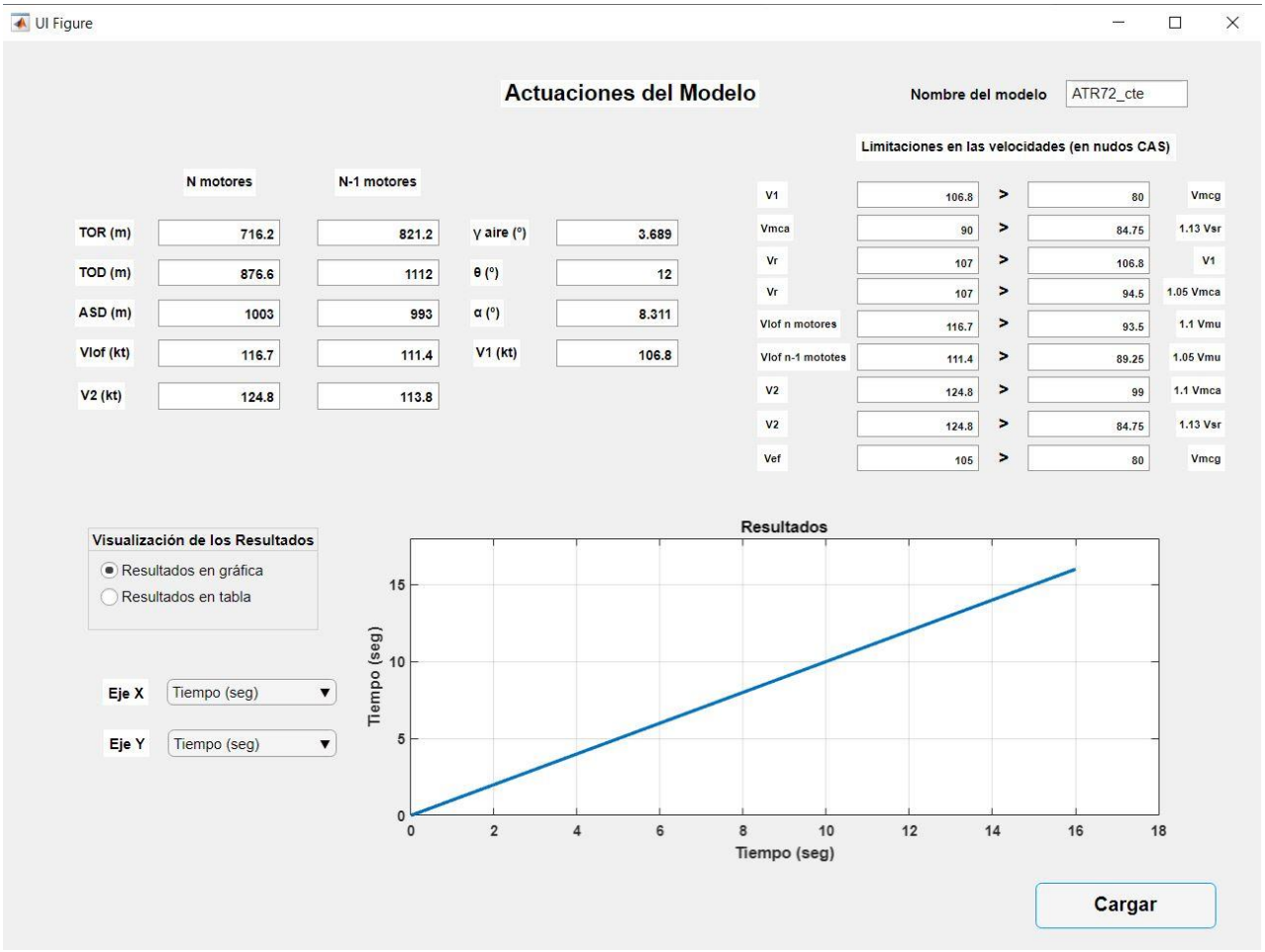


Figura 82. Ventana actuaciones del modelo

Vemos como, analizando y variando parámetros, somos capaces de llegar a resultados muy próximos a los reales, teniendo siempre en cuenta que este modelo es una aproximación y que usamos parámetros aproximados.

Es importante recordar que el objetivo de este programa es servir de herramienta para el análisis ingenieril, objetivo que hemos podido comprobar que se alcanza, pues si bien no conocemos parámetros exactos, mediante el estudio de nuestro modelo y viendo que parámetros afectan y de qué forma, somos capaces de construir un modelo aproximado que se ajusta de forma bastante precisa a resultados reales.

Ya hemos visto un modelo tanto con pendiente constante como con pendiente variable del avión ATR 72. Pasemos ahora a hacer lo mismo, pero con otro avión diferente, el A320 neo, un avión turbofán en este caso, y así podemos seguir explorando las diferentes opciones que ofrece el programa.

Figura 83. Imagen del Airbus A320 neo

El Airbus A320neo es un avión comercial bimotor turbofán desarrollado en la década de los ochenta por Airbus, siendo en 2010 cuando se introdujo una nueva generación del avión denominada A320neo (New Engine Option) que, gracias a distintas opciones de motor y algunos ajustes en el diseño, permitía ahorrar hasta un 15% de combustible.

Tiene una capacidad de unos 150 o 160 pasajeros y es uno de los aviones comerciales más populares y que se vendió de forma más rápida, siendo un éxito absoluto en ventas.

Es un avión de tamaño medio pensado para viajes de distancia y duración media.

Al igual que antes, para definir el modelo de este avión recurrimos al fabricante para obtener las características generales que buscamos, [18] y [19]. Aquellas que no encontremos tendremos que aproximarlas o usar datos de aviones similares.

- $Peso\ máximo\ en\ despegue \sim 78000\ kg$
- $b = 35.8\ m$
- $S_{alar} = 122.6\ m^2$
- $\Delta = 25$
- $Potencia\ de\ despegue\ máxima \sim 120\ kN$
- $Ángulo\ de\ ataque\ del\ empuje\ \epsilon \sim -1\ grado$
- $Ángulo\ de\ asiento\ de\ referencia\ \theta_0 \sim -\ 3\ grados$
- $Coeficiente\ de\ rozamiento\ \mu \sim 0.02$
- $Coeficiente\ de\ frenada\ \mu_{FRE} \sim 0.3$

Las características aerodinámicas no están disponibles en la web del fabricante, así que de igual forma que antes, usaremos las tablas que hemos visto. Además, también tenemos de referencia un documento donde encontramos información al respecto [20]. Con esta información construimos nuestro modelo.

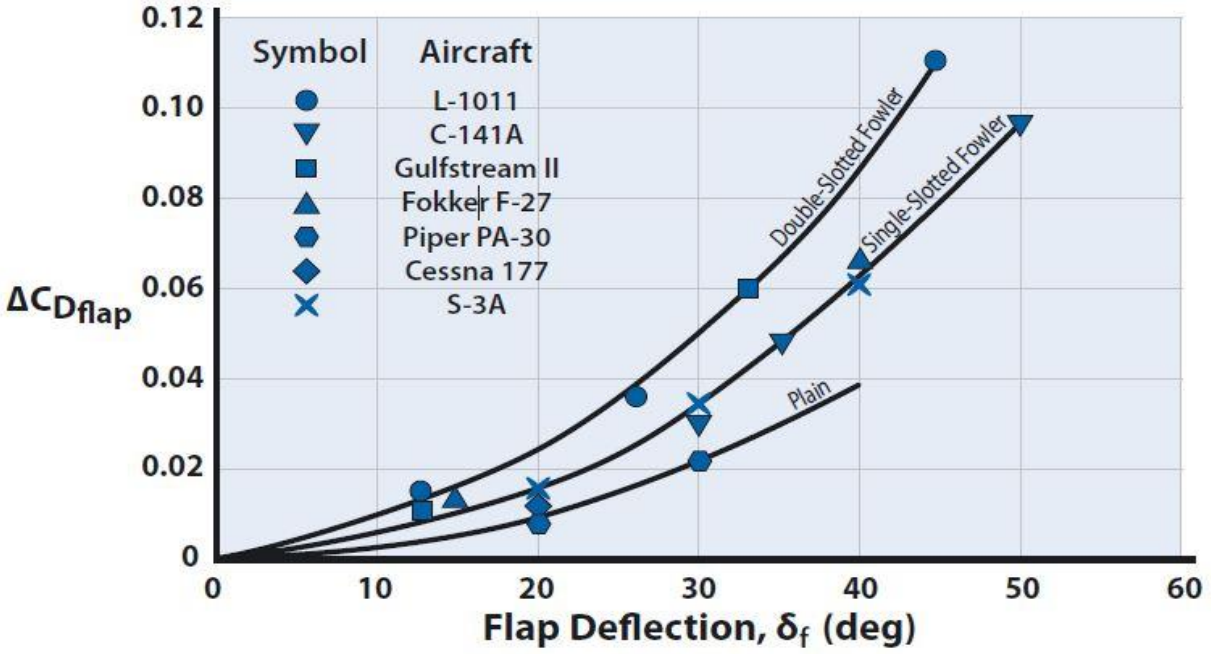


Figura 84. Gráfica para estimar el valor de ΔC_{Dflap} obtenida de [8]

El modelo más similar sería el L-1011, aunque este cuenta con tres motores.

- $\Delta C_{Dflap} \sim 0.02$

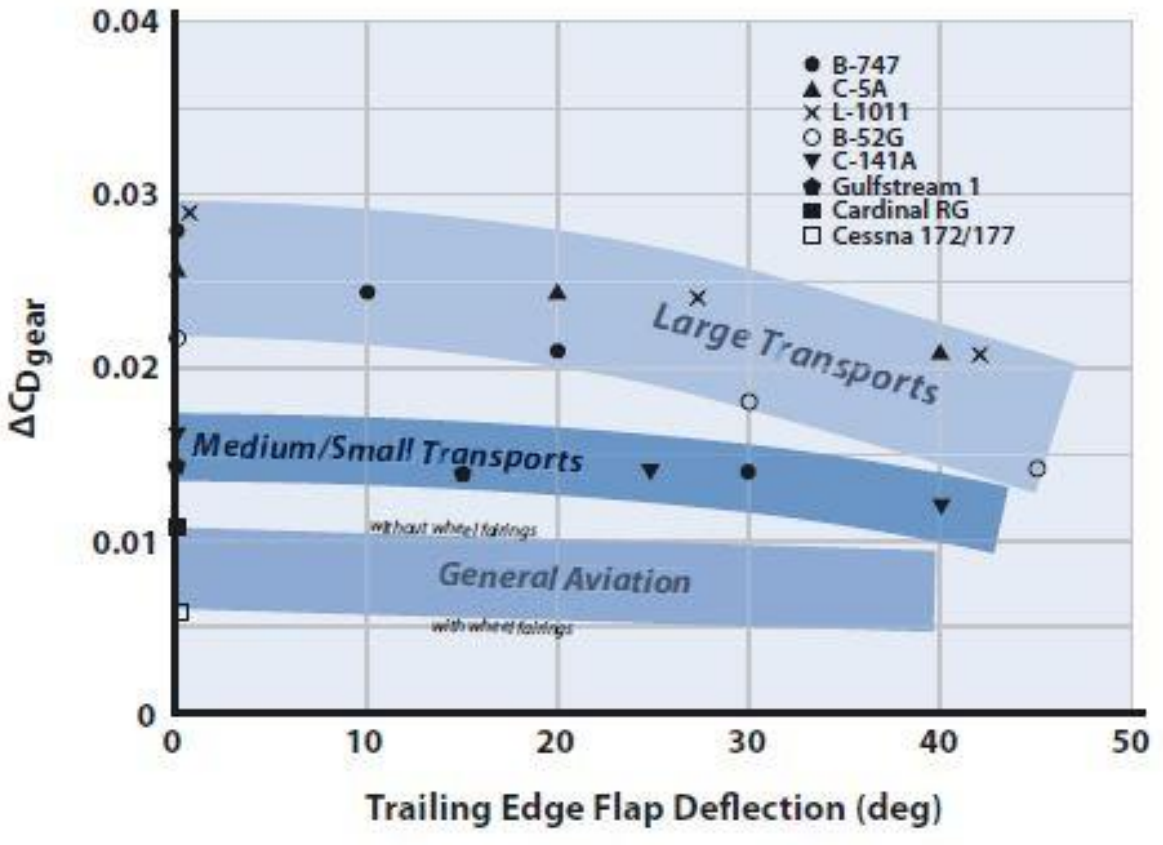


Figura 85. Gráfica para estimar el valor de ΔC_{Dgear} obtenida de [8]

Para la resistencia del tren de aterrizaje cogemos un valor promedio para aviones de tamaño medio grande.

- $\Delta C_{Dgear} \sim 0.025$

Table 5.2 Representative Values for Subsonic C_{D_0}

Aircraft Type	Subsonic C_{D_0}
High-subsonic jet transport	0.014–0.02
Supersonic fighter aircraft	0.014–0.022
Blended wing–body (tailless) jet aircraft	0.008–0.014
Large turboprop aircraft	0.018–0.024
Low-altitude subsonic cruise missile (high *W/S*)	0.03–0.04
Small single-engine propeller aircraft	
Retractable gear	0.022–0.030
Fixed gear	0.026–0.04
Agricultural aircraft	
With spray system	0.07–0.08
Without spray system	0.06
High-performance sailplane	0.006–0.01

Figura 86. Tabla con valores representativos de C_{D0} obtenida de [8]

Nuestro modelo en este caso se trata de un avión de transporte medio, por lo tanto, aproximando,

- $C_{D0} \sim 0.023$
- $k \sim 0.0334$

Por último, queda definir el bloque de velocidades. De nuevo, esta información es bastante complicada de encontrar, por lo que al igual que con el anterior modelo, intentaremos aproximar estas velocidades con la información disponible.

Entonces, con la información que hemos podido encontrar sobre el A320 neo en [21] y [22], tenemos que aproximadamente,

- $V_{MCG} \sim 105\ kt$
- $V_{MCA} \sim 110\ kt$
- $V_{SR} \sim 95\ kt$
- $V_{MU} \sim 100\ kt$
- $V_{EF} \sim 140\ kt$
- $T_{REC} \sim 2\ seg$
- $V_R \sim 150\ kt$

Con estos datos, podemos empezar a crear nuestro modelo.

Procederemos igual que con el ejemplo del ATR72, primero analizaremos el caso con pendiente constante y luego probaremos algún aeropuerto de ejemplo.

5.2.1 Modelo con pendiente constante

Primero creamos nuestro modelo. Para este modelo, vamos a usar la opción de definir las velocidades como función del peso. Como realmente no tenemos estas funciones, las inventaremos para que salgan las velocidades buscadas.

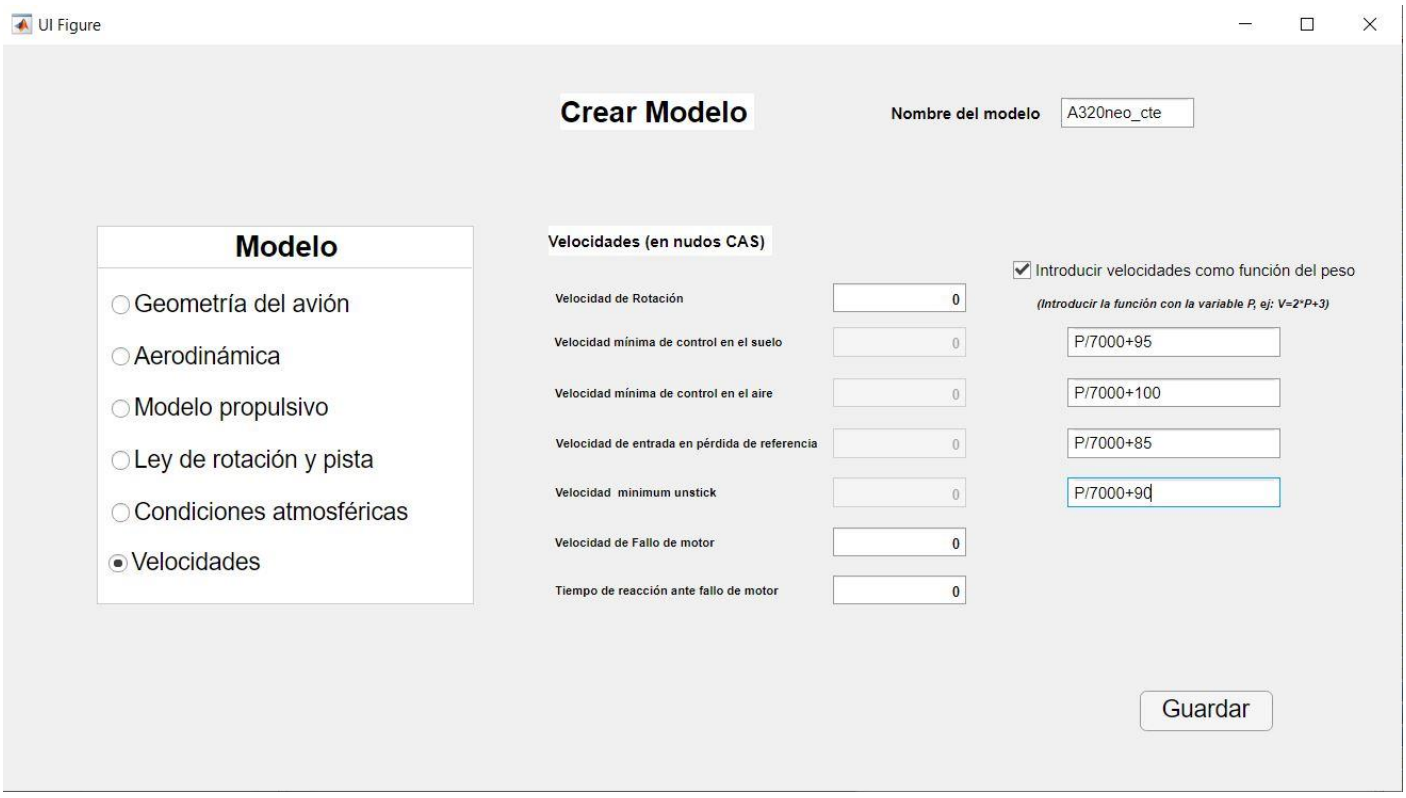


Figura 87. Ventana crear modelo

También, como en el anterior modelo de pendiente constante, elegimos una presión-altitud diferente de cero,

- $Presión - altitud = 1000\ ft$

Para este modelo elegimos ley de rotación personalizada, con los datos siguientes,

- $Ángulo\ final\ de\ asiento\ \theta_{final} = 15\ grados$
- $Tiempo\ total\ de\ rotación = 4\ seg$

Un poco más altos que los elegidos en el anterior ejemplo, pero todavía dentro de los órdenes de magnitud habituales. Ponemos una pendiente del 0%, que al igual que antes variaremos en la ventana barrido de parámetros, aprovechando que este modelo si lo permite.

Si necesitásemos modificar algo, se podría hacer a través de la ventana modificar modelo.

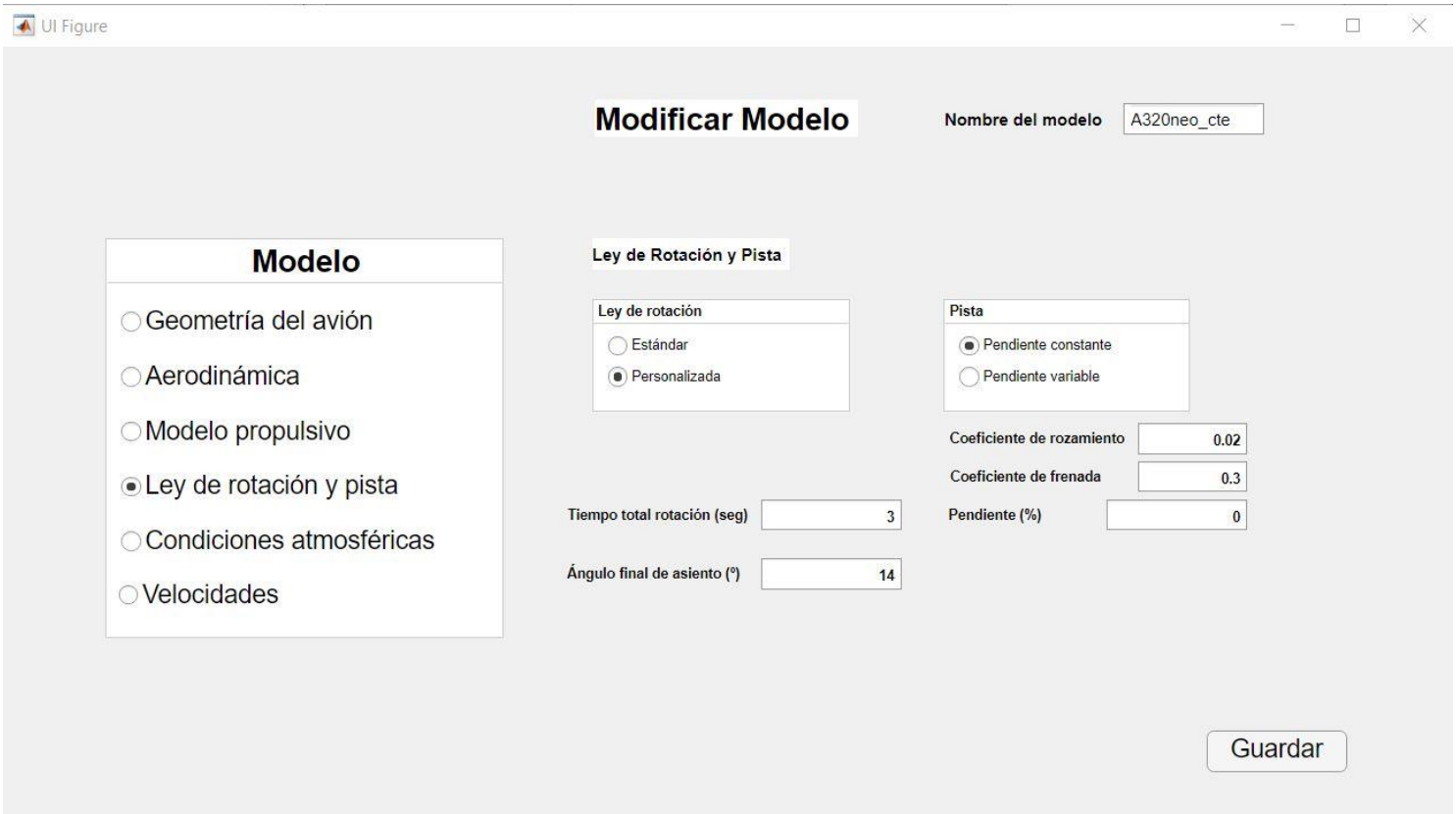


Figura 89. Ventana modificar modelo

Procesamos el modelo.

Figura 90. Ventana procesar modelo con el modelo del A320

Comprobamos antes de ver los resultados que estos se han guardado de forma correcta en el listado de la ventana inicial.

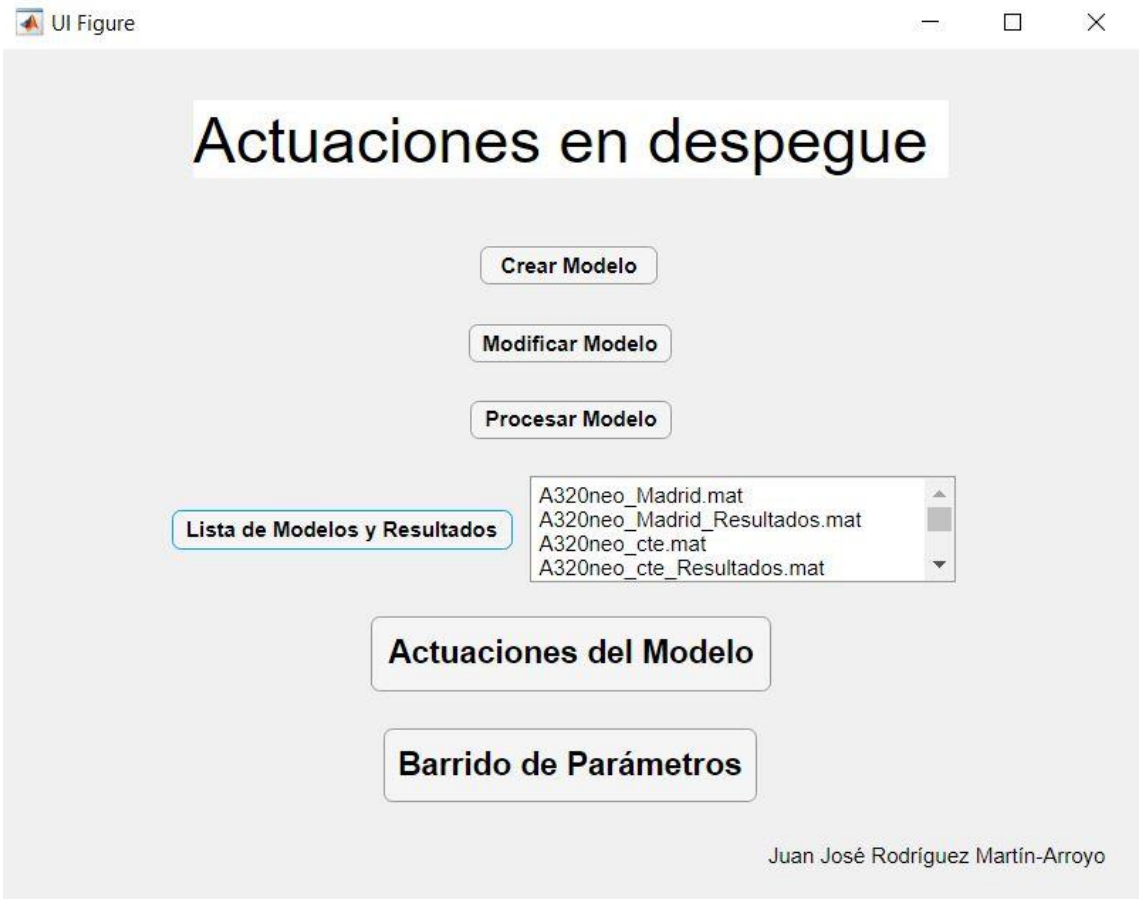


Figura 91. Listado de modelos y resultados incluyendo los resultados del A320

Una vez que hemos comprobado esto y que no ha saltado ningún aviso de que se hayan incumplido las condiciones de la normativa, pasamos a la ventana de actuaciones del modelo para ver los resultados que hemos obtenido con este modelo. Luego pasaremos a compararlos con los resultados reales.

Como viene siendo costumbre primeros veremos la opción de resultados en tabla, y después veremos algún ejemplo con la opción de gráfica.

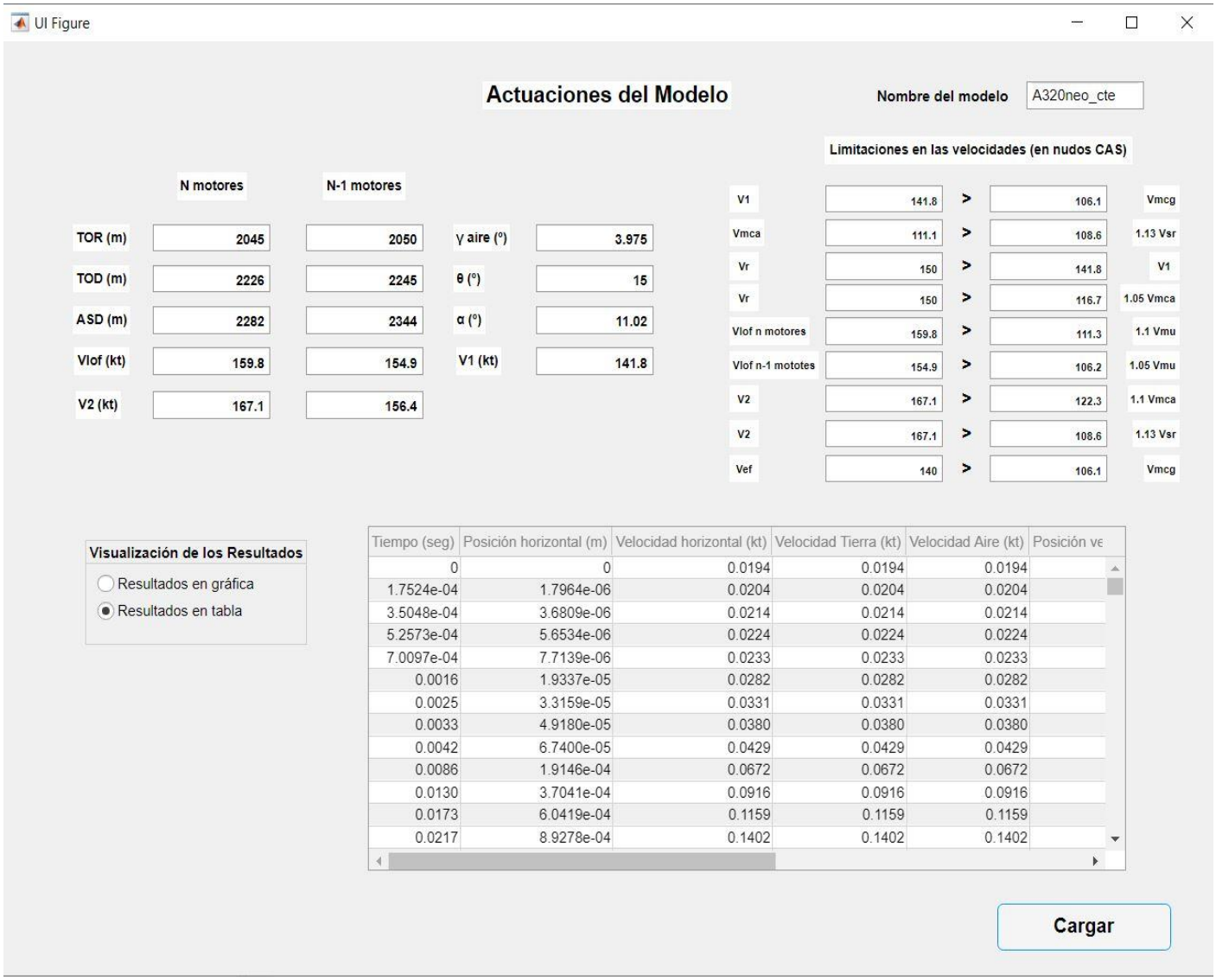


Figura 92. Ventana Actuaciones del modelo, A320

Vemos que se cumplen las condiciones de la normativa y que, las velocidades que hemos introducido como función del peso en efecto son las que nosotros queríamos que fuesen, por lo que esta opción puede servir si se tienen como función del peso.

Fijándonos en los resultados, vemos que, respecto al otro ejemplo de avión, tanto velocidades como distancias han aumentado, pero hay que tener en cuenta que este avión es bastante más grande y pesado. Aun así, como veremos a continuación, siguen estando dentro de los órdenes de magnitud que manejan aviones similares.

Miremos ahora la ventana de actuaciones con la gráfica de ángulo de ataque frente a posición horizontal.

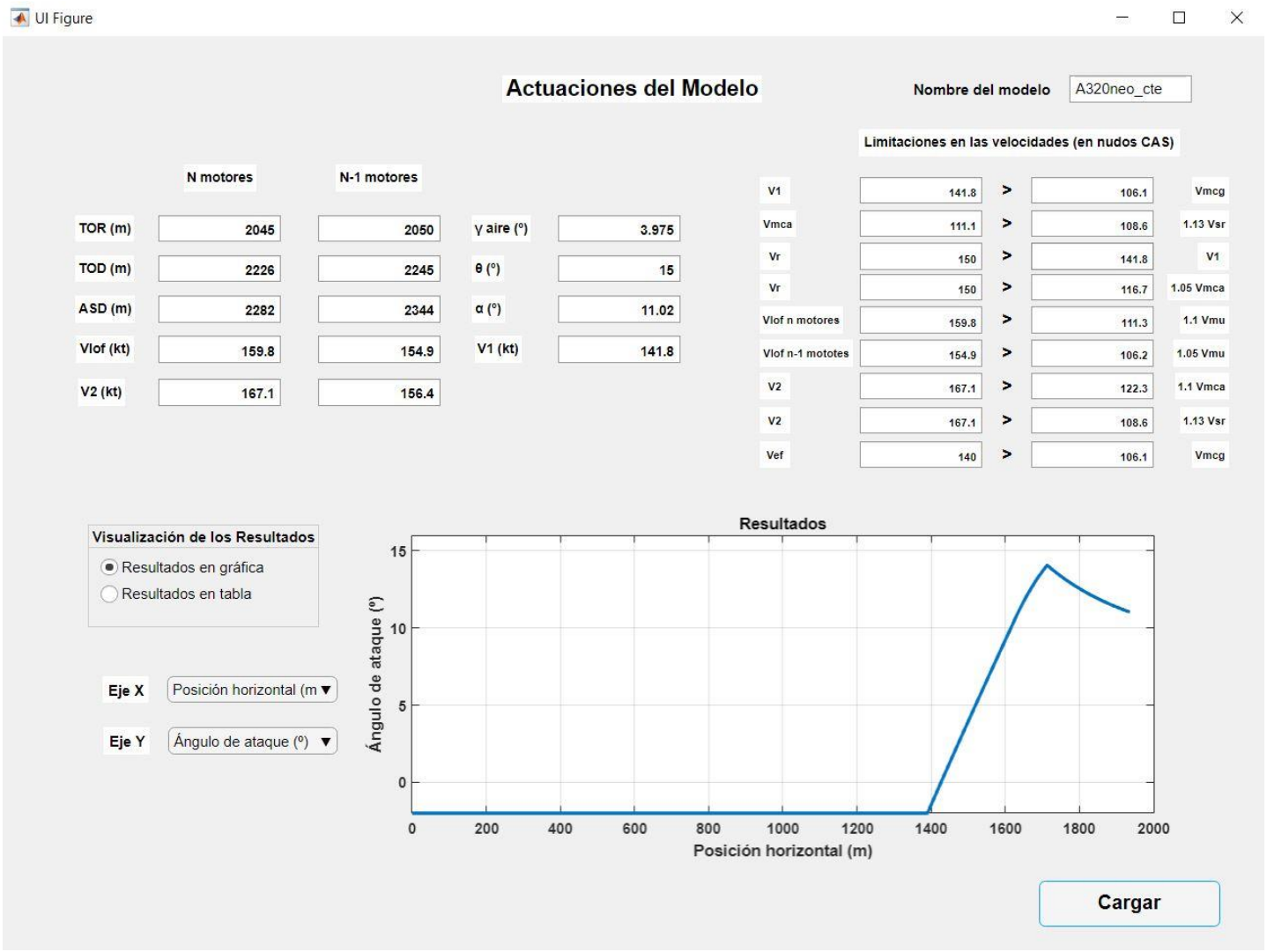


Figura 93. Ventana Actuaciones del modelo, gráfica

Vemos cómo mientras el avión se encuentra en fase de rodadura, el ángulo de ataque se mantiene constante, ya que recordemos que la pista tiene pendiente nula.

Una vez empieza la fase de rotación y aumenta el ángulo de asiento, aumenta el ángulo de ataque de la misma forma.

Por último, en la fase de vuelo el ángulo de asiento respecto a la velocidad empieza a crecer también y esto produce una disminución del ángulo de ataque.

Pasemos ahora a comparar los resultados obtenidos con los valores reales que hemos podido encontrar.

5.2.1.1 Ventana barrido de parámetros

Para empezar, veremos cómo varía este modelo con la pendiente, ya que siempre que tengamos un modelo con pendiente constante es interesante ver las variaciones de las actuaciones con este parámetro.

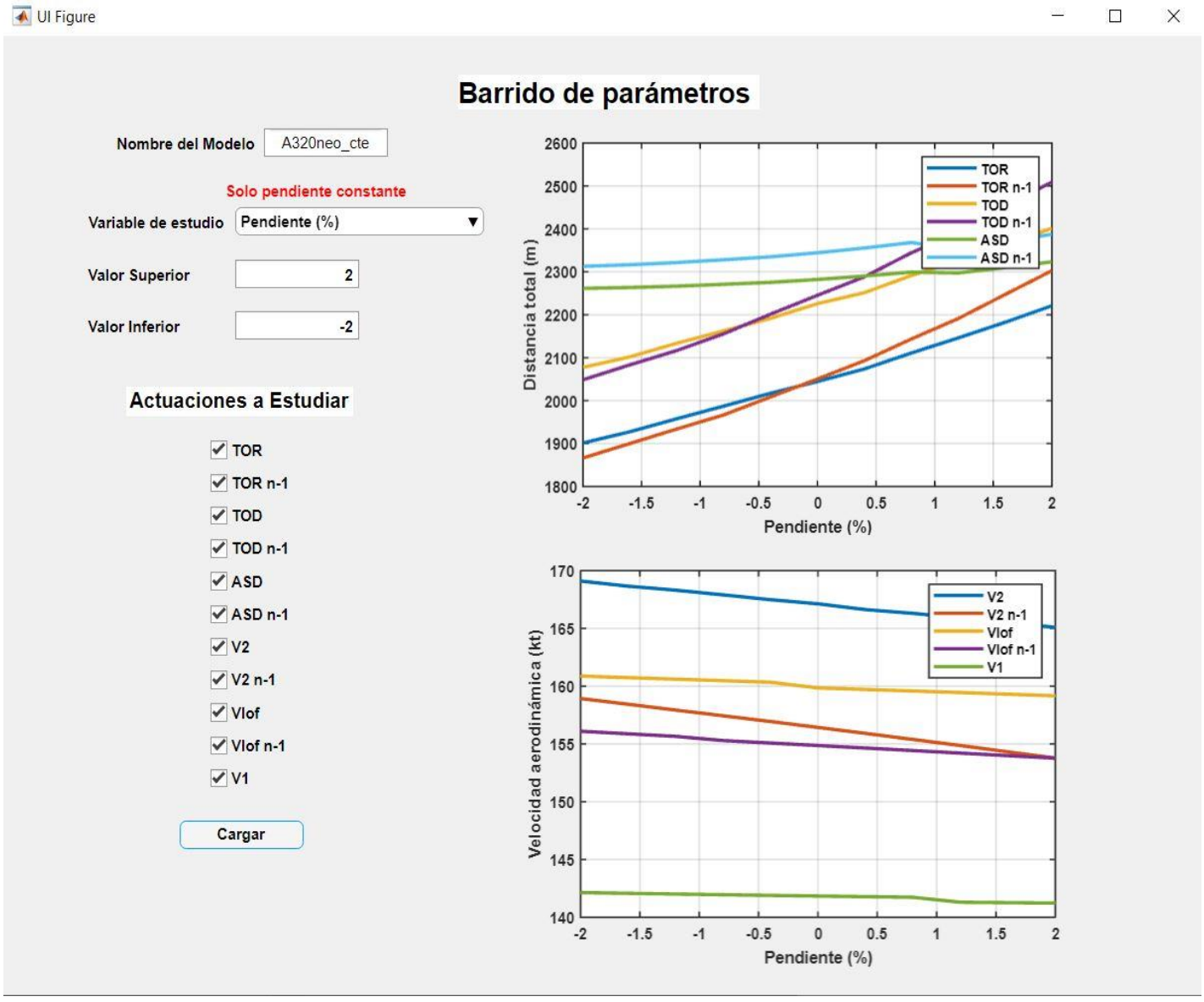


Figura 94. Ventana barrido de parámetros, pendiente

Vemos como la pendiente, en general, afecta de la forma en que, si esta aumenta su valor, todas las actuaciones aumentan también su valor, excepto las velocidades.

En particular, las distancias son las que se ven más afectadas, ya que las velocidades, aunque

también son afectadas por este cambio, no sufren unas variaciones tan grandes.

Ya que hemos escogido en este modelo la opción de velocidades en función del peso, aprovechemos y veamos como varía las actuaciones en función de este parámetro.

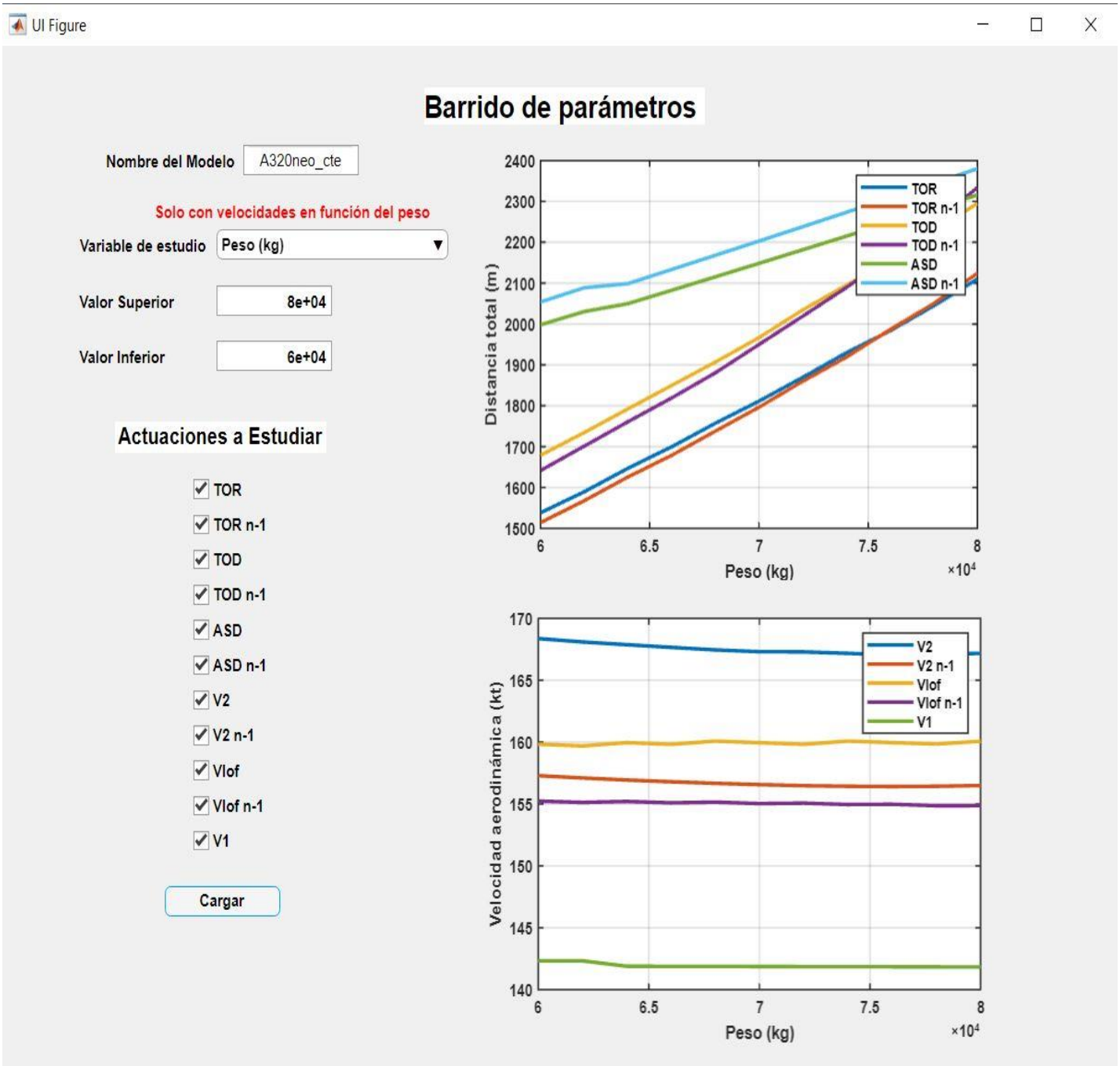


Figura 95. Ventana barrido de parámetros, peso

El resultado que podemos observar es el esperado, a mayor peso, mayor será el valor de las actuaciones, aunque las velocidades varían poco.

Como podemos observar, sobre todo se verán afectadas las distancias.

Veamos ahora como cambian las actuaciones si decidimos cambiar el empuje a nivel del mar de los motores.

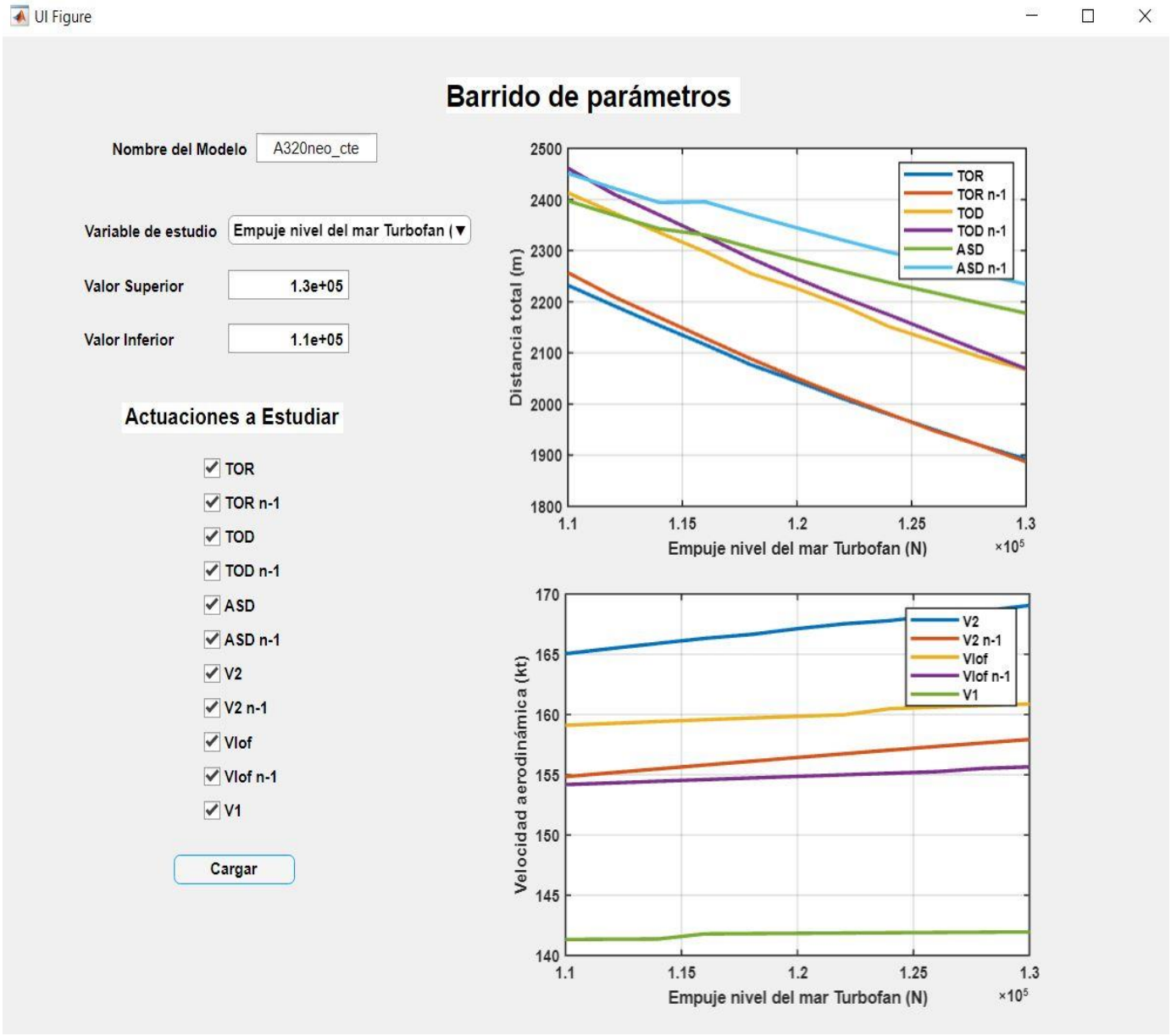


Figura 96. Ventana barrido de parámetros, empuje

Como podemos observar, un mayor empuje se traduce en una reducción de las distancias de despegue, un resultado lógico que podíamos esperarnos.

Al igual que antes, ahora pasaremos a modelar este avión, pero con la opción de pendiente variable, para ver como son los resultados que salen en esta ocasión, y de paso seguir probando cómo funciona la opción de pendiente variable.

5.2.2 Modelo con pendiente variable

Estudiemos ahora el caso de pendiente variable con un aeropuerto diferente, por ejemplo, el de una de las pistas del aeropuerto Adolfo Suarez de Madrid.

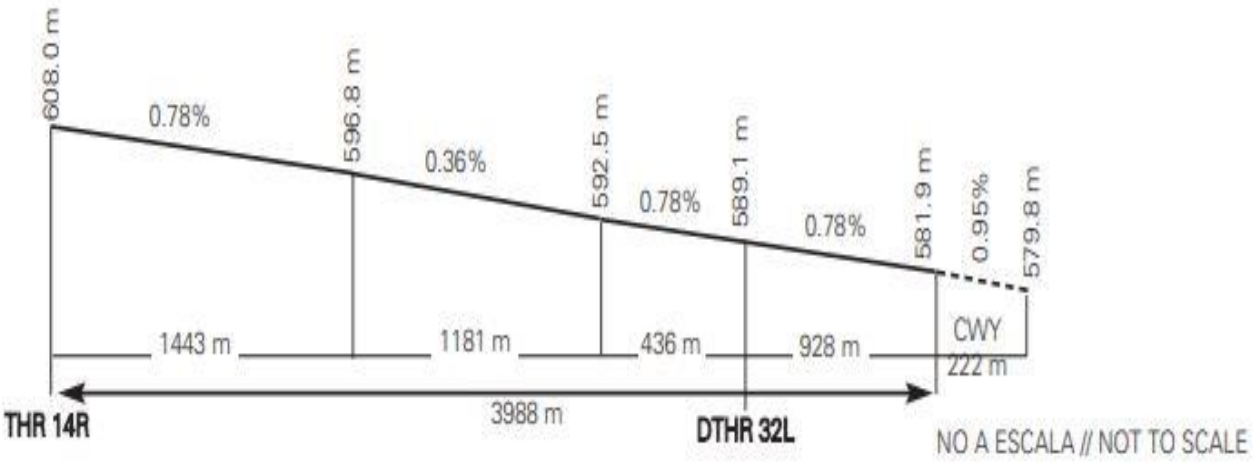


Figura 97. Perfil de pista de una de las pistas del aeródromo de Madrid [17]

Al igual que antes, nuestro modelo será el mismo, pero definiremos la pista variable del aeródromo de Madrid. Quedaría de la siguiente forma.

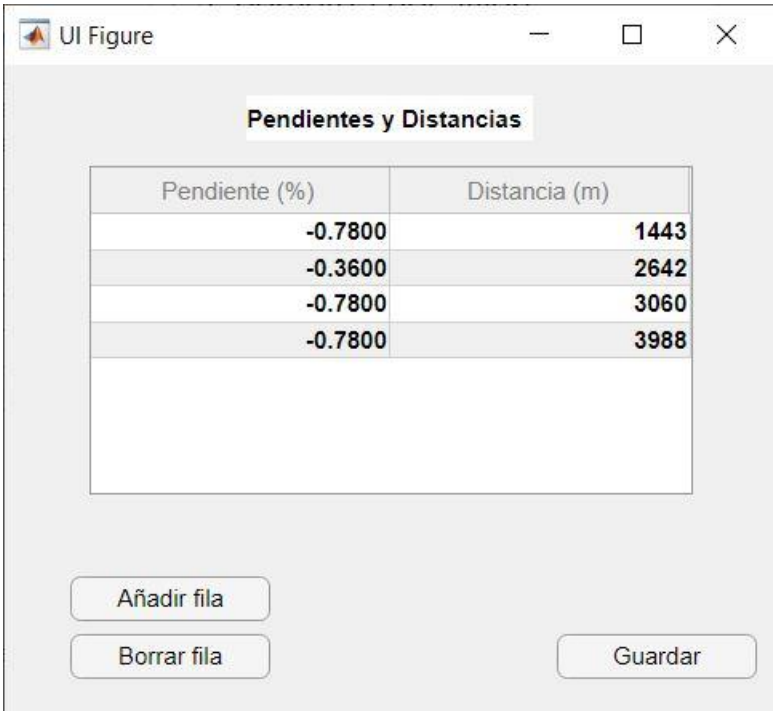


Figura 98. Puntos para definir la pista en la ventana crear modelo

A parte de la opción de pendiente variable, volvemos a escoger la opción de ley de rotación personalizada, con los siguientes parámetros,

- $Ángulo\ final\ de\ asiento\ \theta_{final} = 15\ grados$
- $Tiempo\ total\ de\ rotación = \ 4\ seg$

Y, al estar usando el perfil longitudinal de pista del aeropuerto de Madrid, nos fijamos en que el punto inicial el recorrido está a una presión-altitud determinada, por lo tanto,

- $Presión - altitud = \ 608\ m$

Por lo tanto, ya tenemos todos los datos para nuestro modelo.

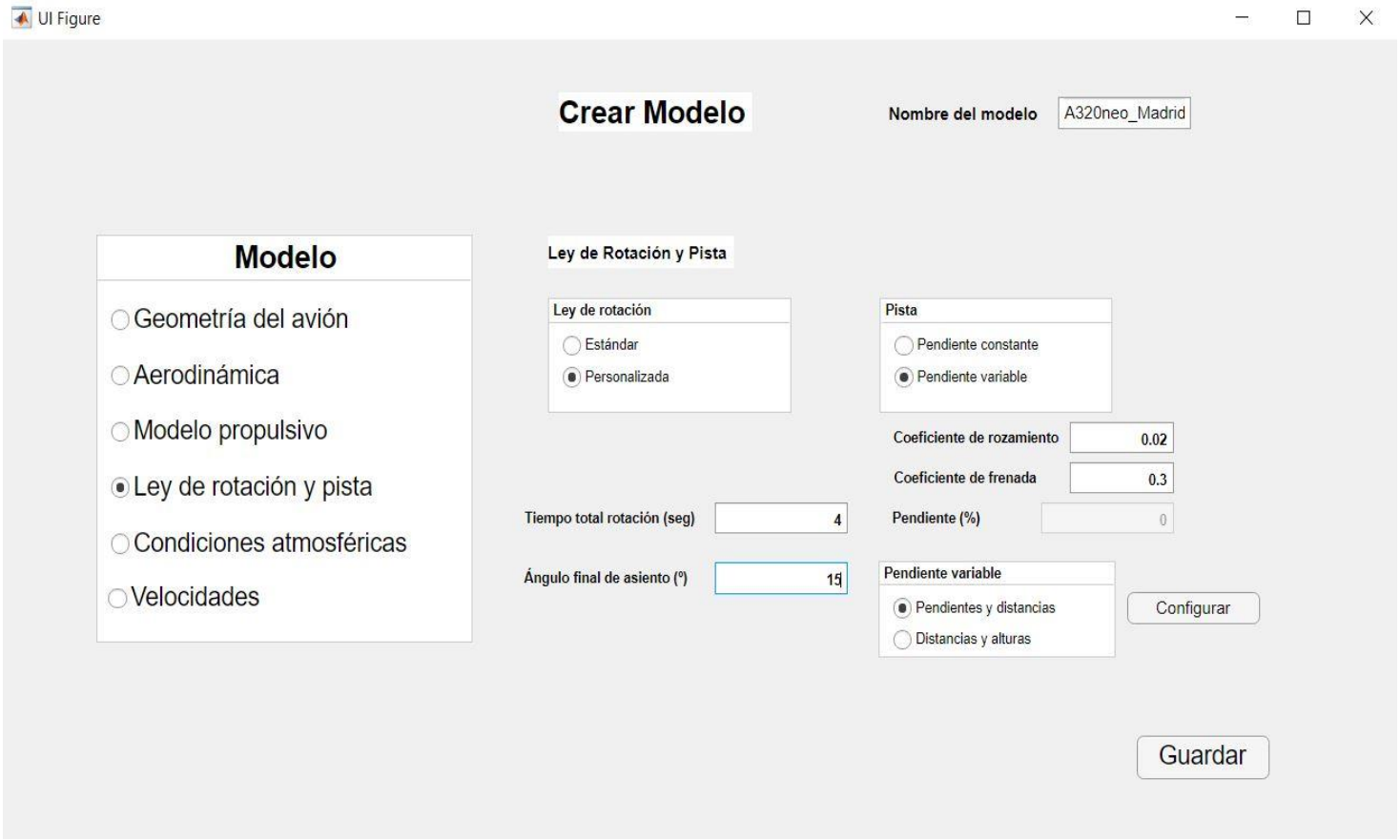


Figura 99. Modelo con pendiente variable del A320

Una vez tenemos el modelo, procesamos el modelo y comprobamos que se guardan los resultados.

Figura 100. Ventana procesar modelo con el modelo del A320 en la pista de Madrid

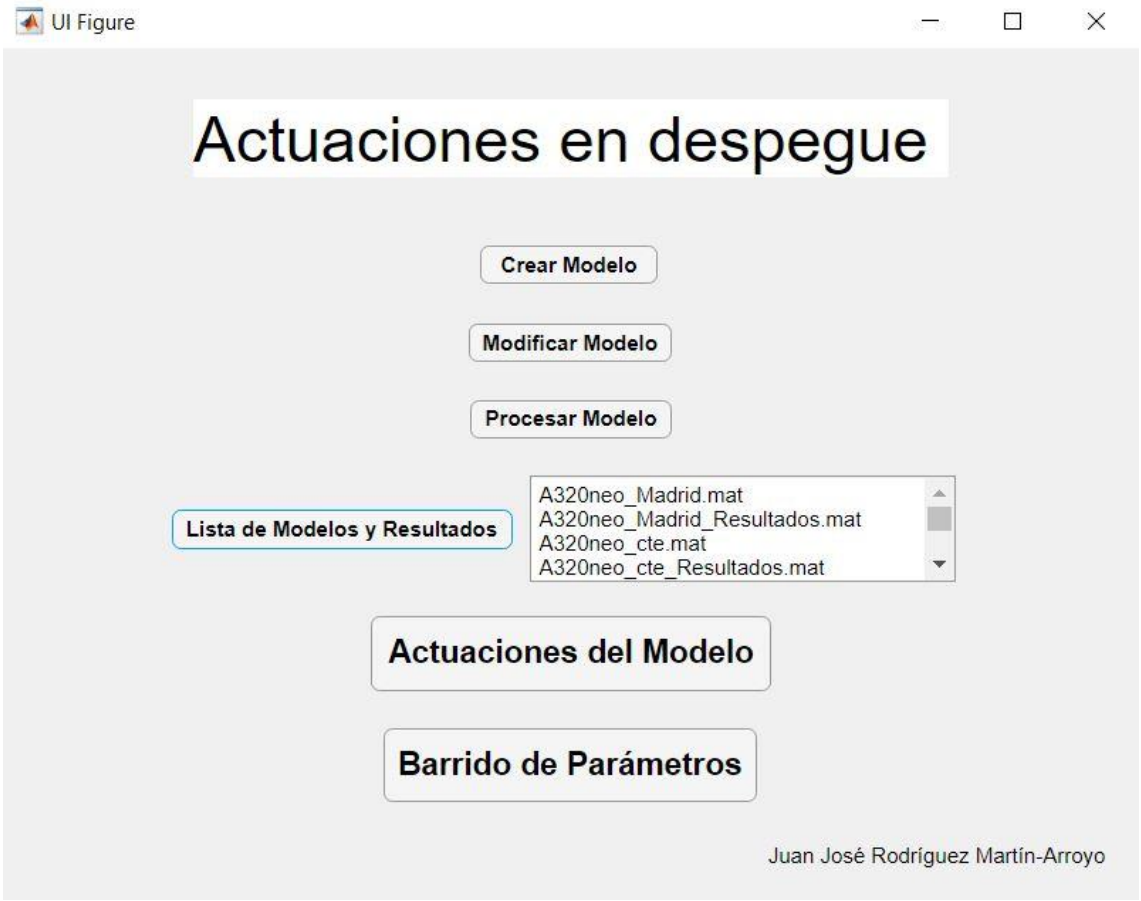


Figura 101, Listado de modelos y resultados incluyendo los resultados del A320 en Madrid

Fijémonos en este caso en lo preciso que es la aproximación al perfil de la pista.

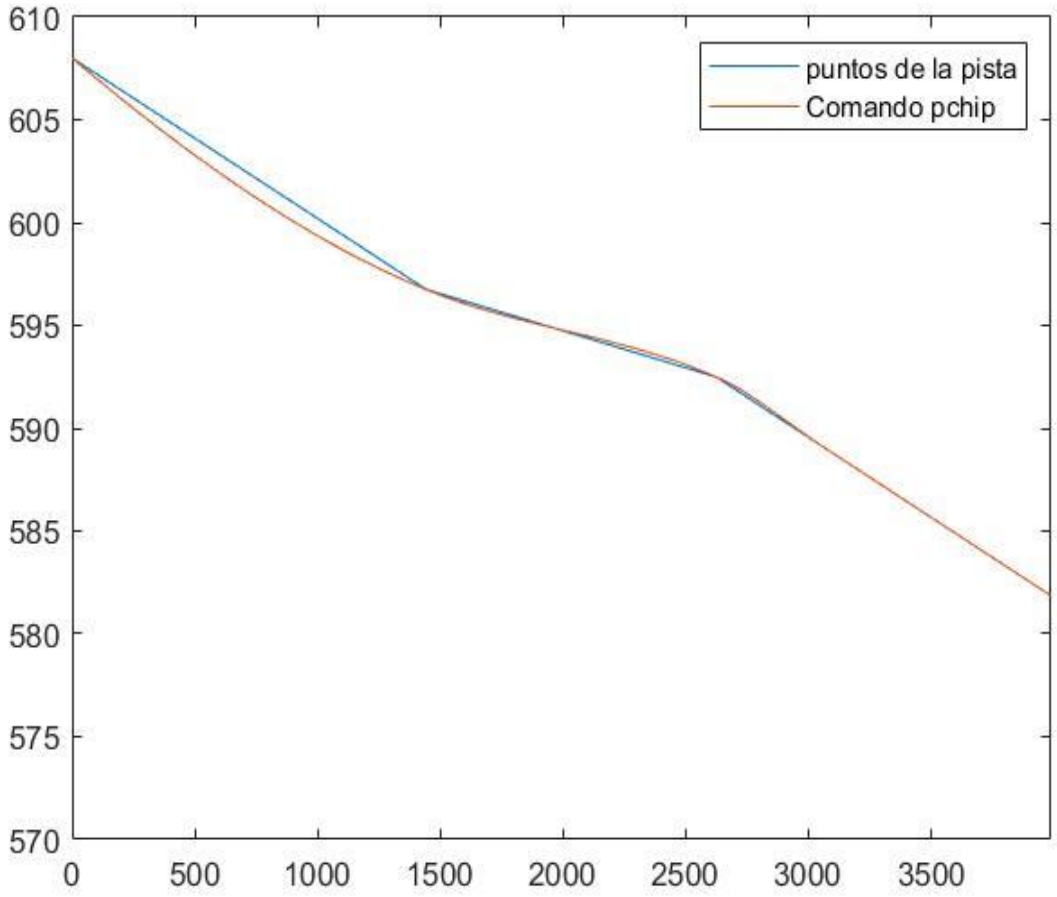


Figura 102. Aproximación del perfil de pista del aeródromo de Madrid

Esto pone de manifiesto lo explicado en apartados anteriores del buen funcionamiento del comando de Matlab ‘pchip’ en vez de la forma habitual de interpolación mediante el comando ‘splines’.

Una vez dicho esto, veamos los resultados que se obtienen con este modelo. Tanto la presión-altitud como el perfil de la pista no son especialmente diferentes, una pendiente un tanto negativa pero un poco más de presión-altitud, por lo que al final los resultados deberían ser bastante similares.

Probamos primero la opción de la ventana de actuaciones del modelo de la tabla de resultados, y luego cambiaremos a la opción de gráfica.

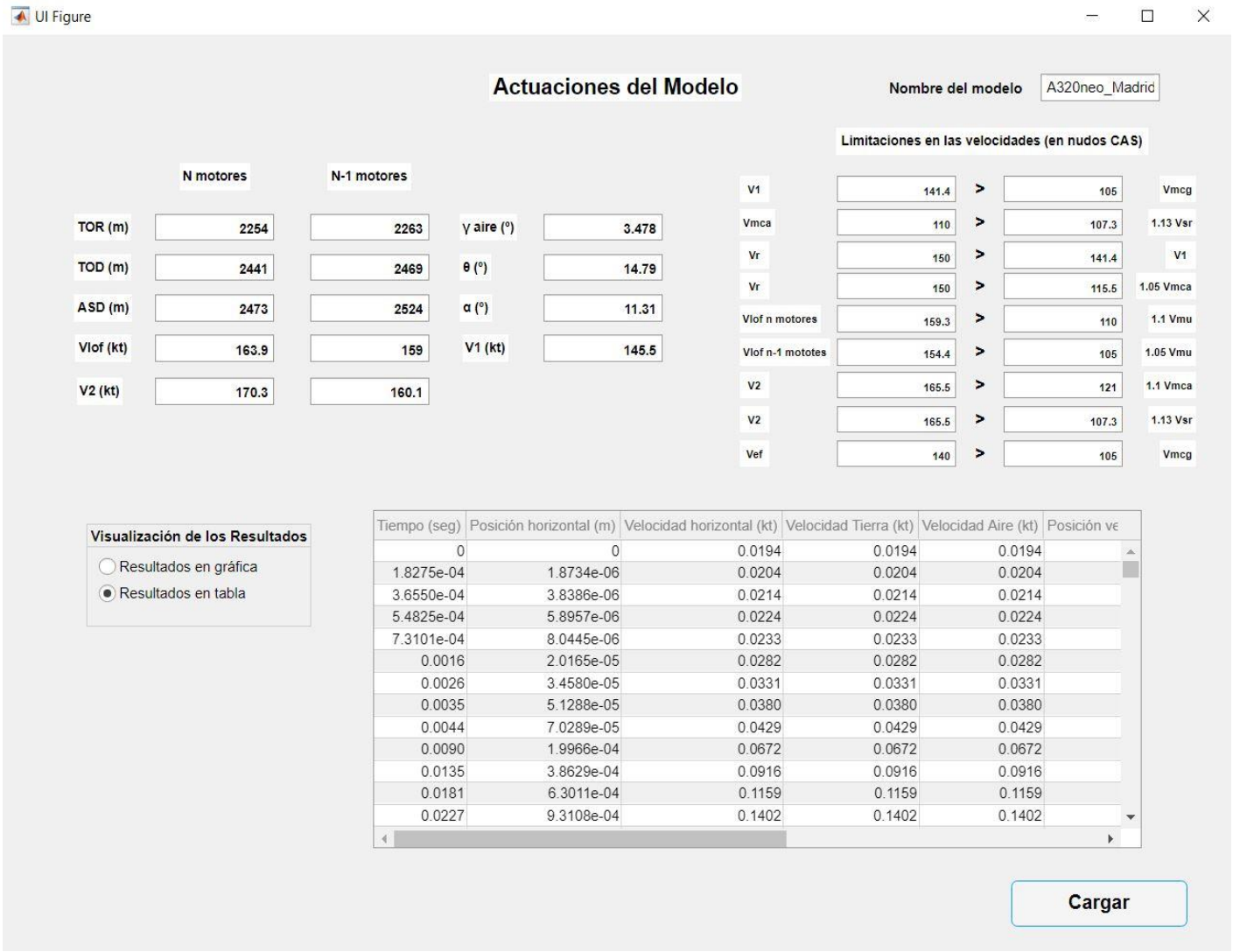


Figura 103. Ventana actuaciones del modelo con el modelo del A320 en Madrid

Podemos ver cómo, efectivamente, los resultados se han mantenido bastante estables, estando todavía dentro de los márgenes para este tipo de aviones de tamaño medio-grande.

En la siguiente imagen escogeremos para la gráfica que nos muestre la posición vertical frente a la posición horizontal. De esta forma, podremos observar como el avión al principio desciende debido a que la pista tiene pendiente negativa, pero que finalmente cuando este se separa del suelo, empieza a elevarse hasta alcanzar los 35 pies de altura.

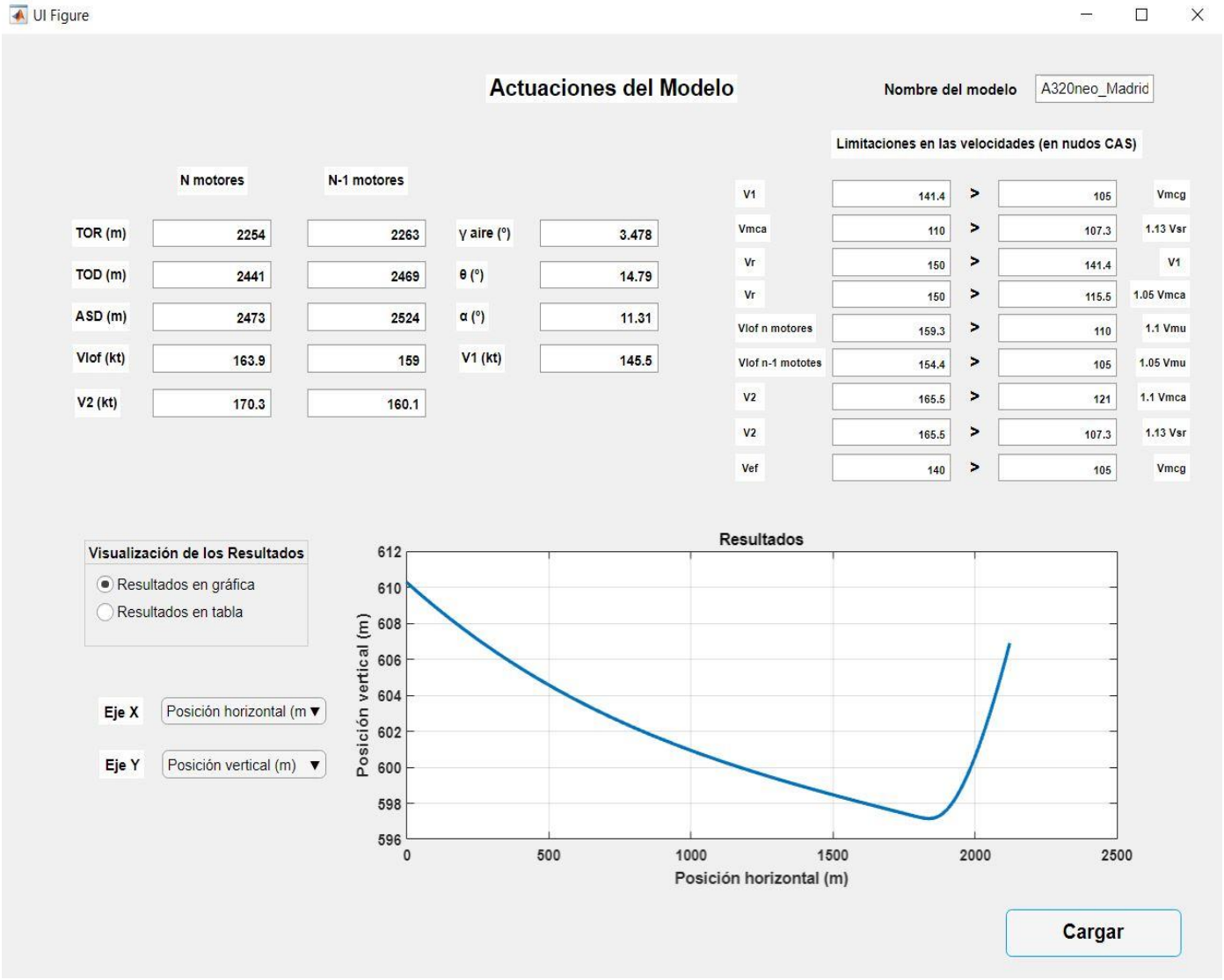


Figura 104. Ventana modificar modelo con el modelo del A320 en Madrid modificado

5.2.2.1 Ventana barrido de parámetros

El primer parámetro que vamos a estudiar va a ser de nuevo el ángulo de asiento final. Vamos a observar que, al igual que antes, este ángulo tiene una gran influencia en los resultados de las actuaciones, ya que variando pocos grados este ángulo hace que los resultados varíen considerablemente.

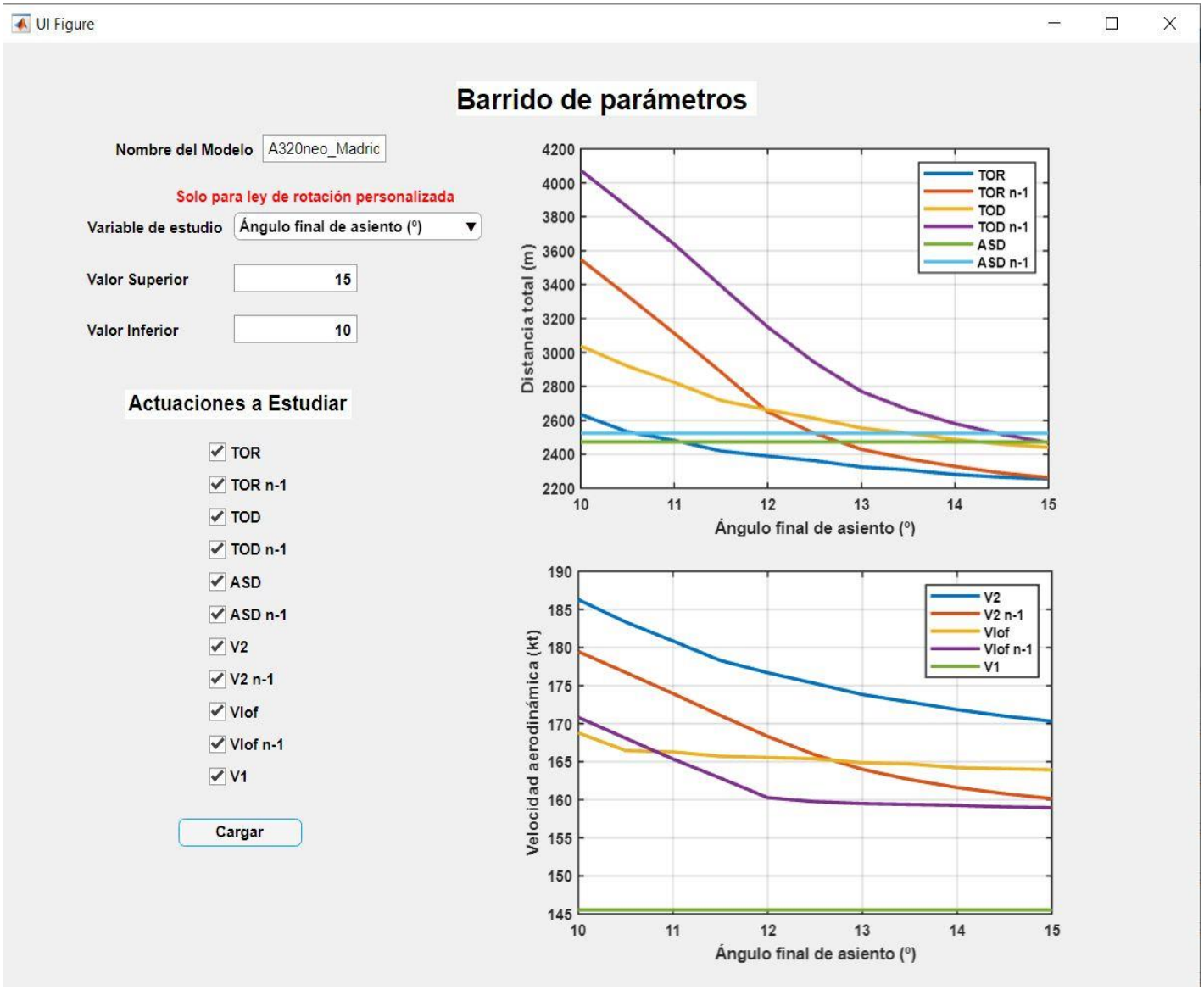


Figura 105. Ventana barrido de parámetros, ángulo final de asiento

Podemos ver cómo, efectivamente, pasa justo lo que esperábamos. Tanto las velocidades como las distancias en las actuaciones se reducen bastante si aumentamos este ángulo de asiento final. Aunque hay que recordar que este ángulo no puede ser tan grande como se quiera, pues está limitado geométricamente por la cola, ya que si aumentamos mucho este ángulo la cola podría chocar contra la pista.

Pasemos ahora a ver el efecto de la variación de la velocidad de fallo de motor. Recordemos que, tal y como están definidas las actuaciones en el programa, el cambio de esta velocidad debería afectar a las actuaciones con fallo de motor y a la velocidad de decisión V_1.

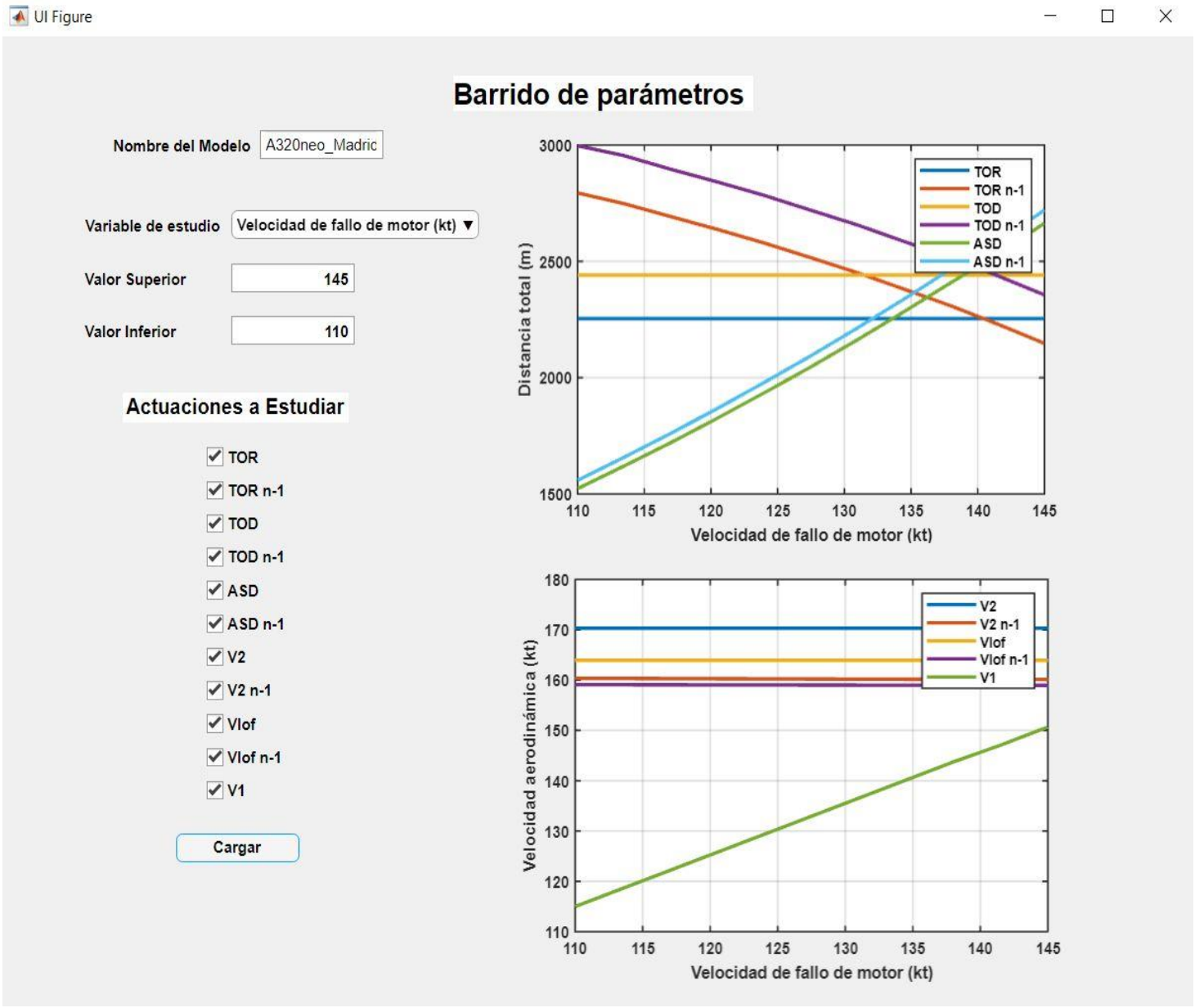


Figura 106. Ventana barrido de parámetros, velocidad de fallo de motor

Efectivamente, vemos que un aumento de la velocidad de fallo de motor se traduce en un aumento de la velocidad V_1 y en un aumento en las distancias de aceleración-parada tanto con fallo de motor como sin fallo de motor.

Sin embargo, tanto la distancia de despegue como la carrera de despegue con fallo de motor se ven reducidas, porque al aumentar la velocidad de fallo de motor se reduce el tramo en el que el avión tiene que acelerar con un solo motor.

Por último, observemos como afecta un cambio en el coeficiente de resistencia con sustentación nula, que forma parte del modelo aerodinámico.

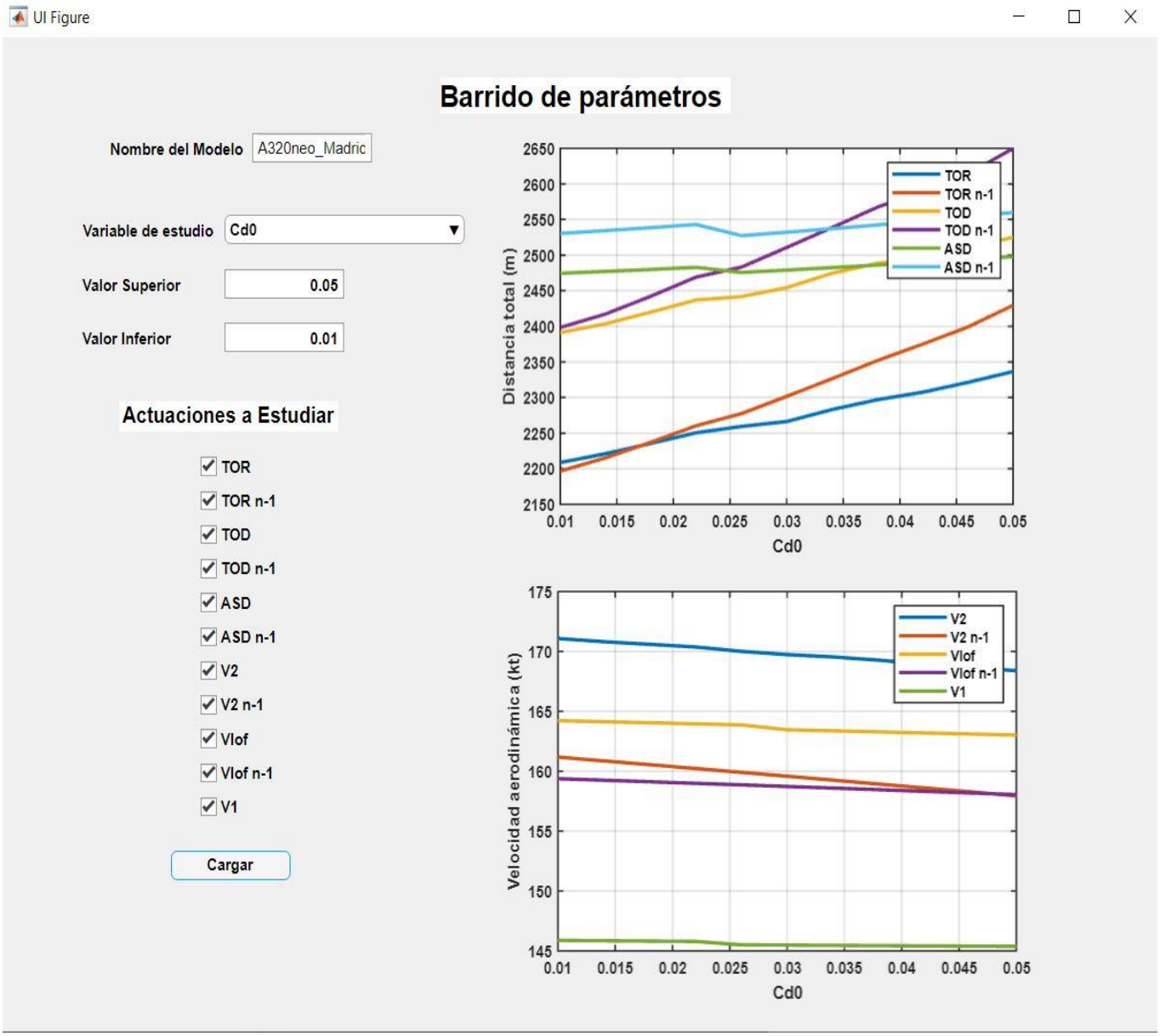


Figura 107. Ventana barrido de parámetros, coeficiente de resistencia sin sustentación

Vemos que esta variación no afecta especialmente a las velocidades, pero que un aumento de este coeficiente se traduce en un aumento de las distancias. Esto se debe a que, al aumentar este coeficiente, está aumentando a su vez la resistencia, lo que provoca un aumento de las distancias, sobre todo en el caso de fallo de motor.

Pasemos ahora a comparar estos resultados con los datos reales que podemos encontrar en los manuales de actuaciones y en otros documentos que proporciona el fabricante.

5.2.3 Comparación de resultados con los reales

Al igual que antes, para hacer las comparaciones con los valores reales, por un lado, veremos la distancia de despegue TOD que proporciona el fabricante ([19]), y por otro lado veremos las velocidades típicas de despegue que constan en los manuales de vuelo y actuaciones de la aeronave ([21], [22], [23]).

La TOD que proporciona el fabricante es de unos 2000 metros para máximo peso en despegue (unos 78000 kilogramos), para condiciones ISA a nivel del mar y con pendiente nula. Sin embargo, tenemos el mismo problema de antes de no saber para qué velocidades de rotación o fallo de motor es esta distancia. Volvemos a asumir que esta distancia es la asociada a la V_1 que hace que TOD_{n-1} y ASD sean iguales (V_1 de 'Balanced field length').

Introducimos los datos correspondientes en nuestro modelo igual que antes, ajustamos el resto de parámetros de entrada, y, barriendo el parámetro V_{EF} tenemos,

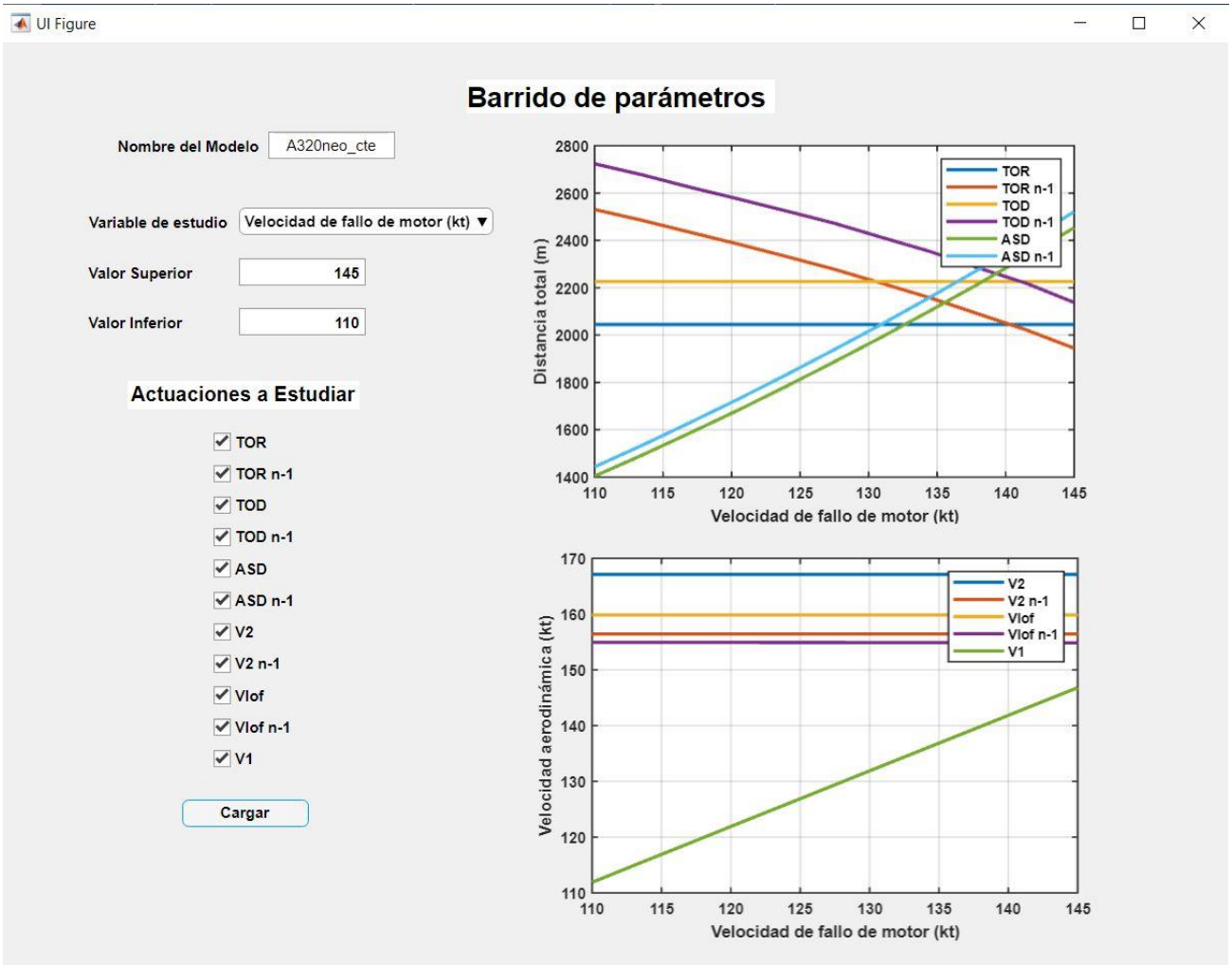


Figura 108. Ventana barrido de parámetros, velocidad de fallo de motor

Vemos que la velocidad de fallo de motor que buscamos está en torno algo por debajo de los 140 nudos, y que corresponden en la gráfica inferior a una V_1 de unos 140 nudos.

Si introducimos esta velocidad como parámetro de entrada y volvemos a procesar el modelo podremos comparar las actuaciones con las proporcionadas con el fabricante.

	Programa	Reales
TOD n motores (m)	2085	2090
TOD n-1 motores (m)	2116	

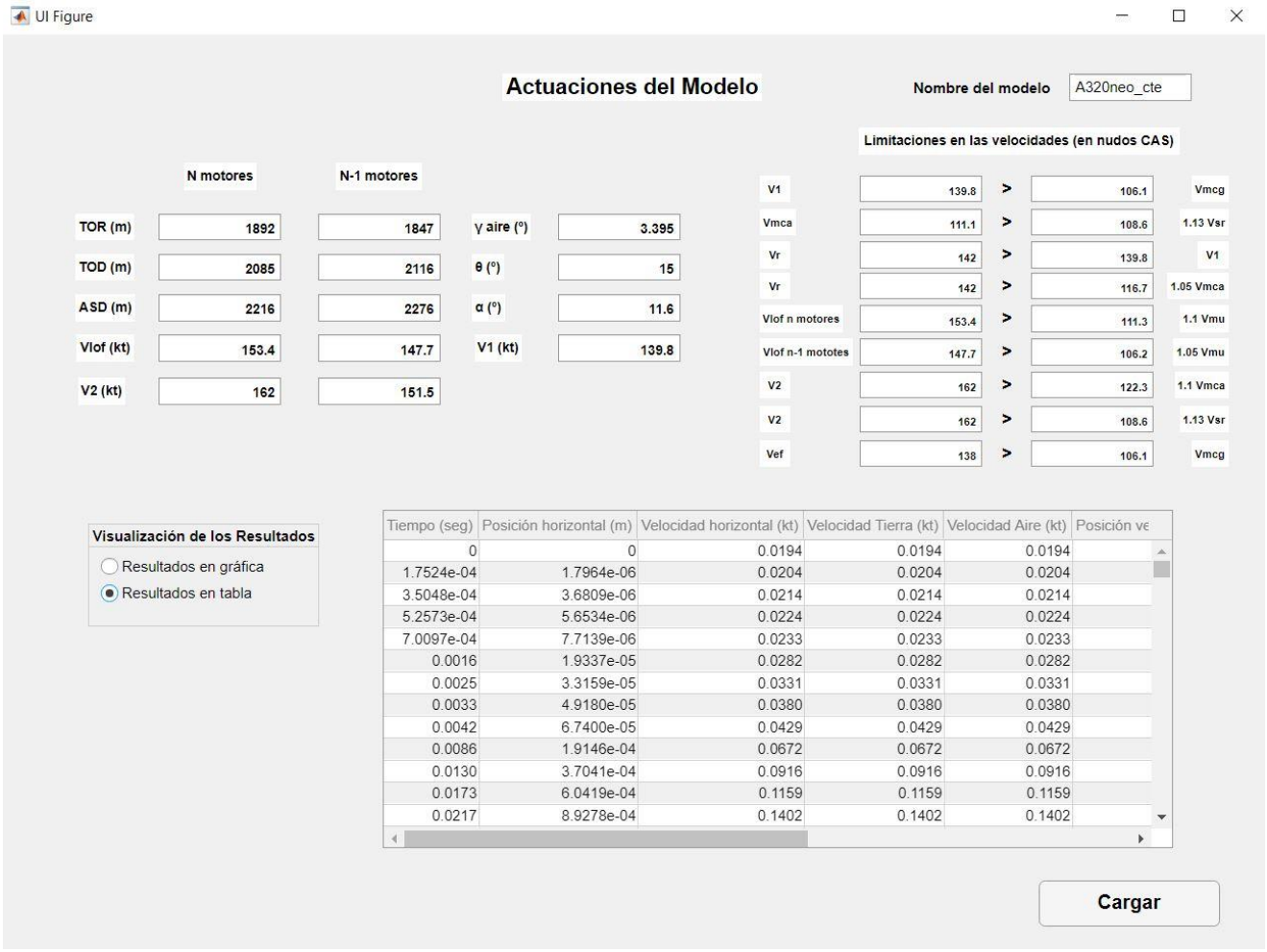


Figura 109. Ventana actuaciones del modelo

Vemos que nuestro resultado prácticamente el mismo al proporcionado por el fabricante. De nuevo, destacar la capacidad de análisis que ofrece el programa, pues conseguimos muy buenos resultados con un modelo aproximado a base de ajustar los parámetros de entrada.

Como antes, del resto de las distancias no se ha podido encontrar ni siquiera una aproximación.

Pasemos ahora a ver las velocidades en el despegue que se proporcionan en los manuales de vuelo. Volvemos a ver una imagen extraída de un manual de vuelo y actuaciones. Esta vez vamos a coger un ejemplo para una altitud presión de 1000 pies, temperatura de 10 grados centígrados y una longitud de pista de 2500 metros. Vamos a ver cuánto podemos ajustar nuestros resultados a los reales.

A319/A320/A321 FLIGHT CREW OPERATING MANUAL	TAKEOFF QUICK REFERENCE TABLES	2.02.40 SEQ 300	P 5 REV 34

CONFIGURATION 1+F			PRESSURE ALTITUDE = 1000 FT		FWD CG
TREF = 42 °C TMAX = 53 °C			DRY RUNWAY SLOPE = 0 %	MAX TO WEIGHT(1000KG) CODES IAS(KT) : V1 / VR / V2	
TEMP. (°C)	CORRECTED RUNWAY LENGTH (M)				
	2250	2500	2750	3000	3250
-20	78.9 3/9 148/51/54	81.6 3/6 154/55/57	82.5 3/6 153/58/60	83.2 3/6 152/60/62	83.8 3/6 152/63/64
-10	77.9 3/9 147/50/52	80.8 3/6 151/54/56	81.8 3/6 150/56/57	82.6 3/6 150/58/60	83.2 3/6 149/61/62
0	76.7 3/9 145/49/51	79.9 3/6 149/53/55	81.0 3/6 148/54/56	81.9 3/6 147/56/58	82.6 3/6 147/59/60
10	75.5 3/9 143/49/51	78.9 3/6 148/52/53	80.0 3/6 146/53/55	81.2 3/6 145/54/56	81.9 3/6 145/57/59
20	74.2 3/9 142/48/50	77.8 3/6 146/50/52	79.1 3/6 145/52/54	80.2 3/6 143/53/55	81.2 3/6 143/55/57
30	73.0 3/9 141/47/49	76.6 3/6 144/49/51	78.0 3/6 143/51/53	79.1 3/6 141/52/54	80.3 3/6 141/53/55
40	71.9 3/9 139/46/48	75.4 3/6 143/49/51	77.0 3/6 141/50/52	78.3 3/6 140/51/53	79.5 3/6 139/52/54
42	71.6 3/9 139/46/48	75.2 3/6 142/49/51	76.7 3/6 141/50/51	78.1 3/6 140/51/53	79.3 3/6 138/52/54
44	70.7 3/9 139/45/47	74.2 3/6 143/48/50	75.8 3/6 142/49/51	77.1 3/6 140/50/52	78.2 3/6 139/51/53
46	69.8 3/9 138/44/46	73.1 3/9 143/48/49	74.8 3/6 142/49/50	76.1 3/6 141/49/51	77.0 3/6 140/51/52
48	68.9 3/9 138/43/45	72.0 3/9 142/47/48	73.9 3/6 143/48/50	75.0 3/6 142/49/50	75.8 3/6 141/51/52
50	67.9 3/9 138/42/44	71.0 3/9 142/46/47	73.0 3/6 144/48/49	74.0 3/6 143/49/50	74.6 3/6 142/51/52

Figura 110. Manual de vuelo A320 [21]

Volvemos a modificar nuestro modelo con estas condiciones que acabamos de comentar y buscamos estas velocidades. Obtenemos lo siguiente.

	Programa	Reales
V_1 (nudos CAS)	147.1	148
V_2 (nudos CAS)	161.6	153
V_R (nudos CAS)	152	152

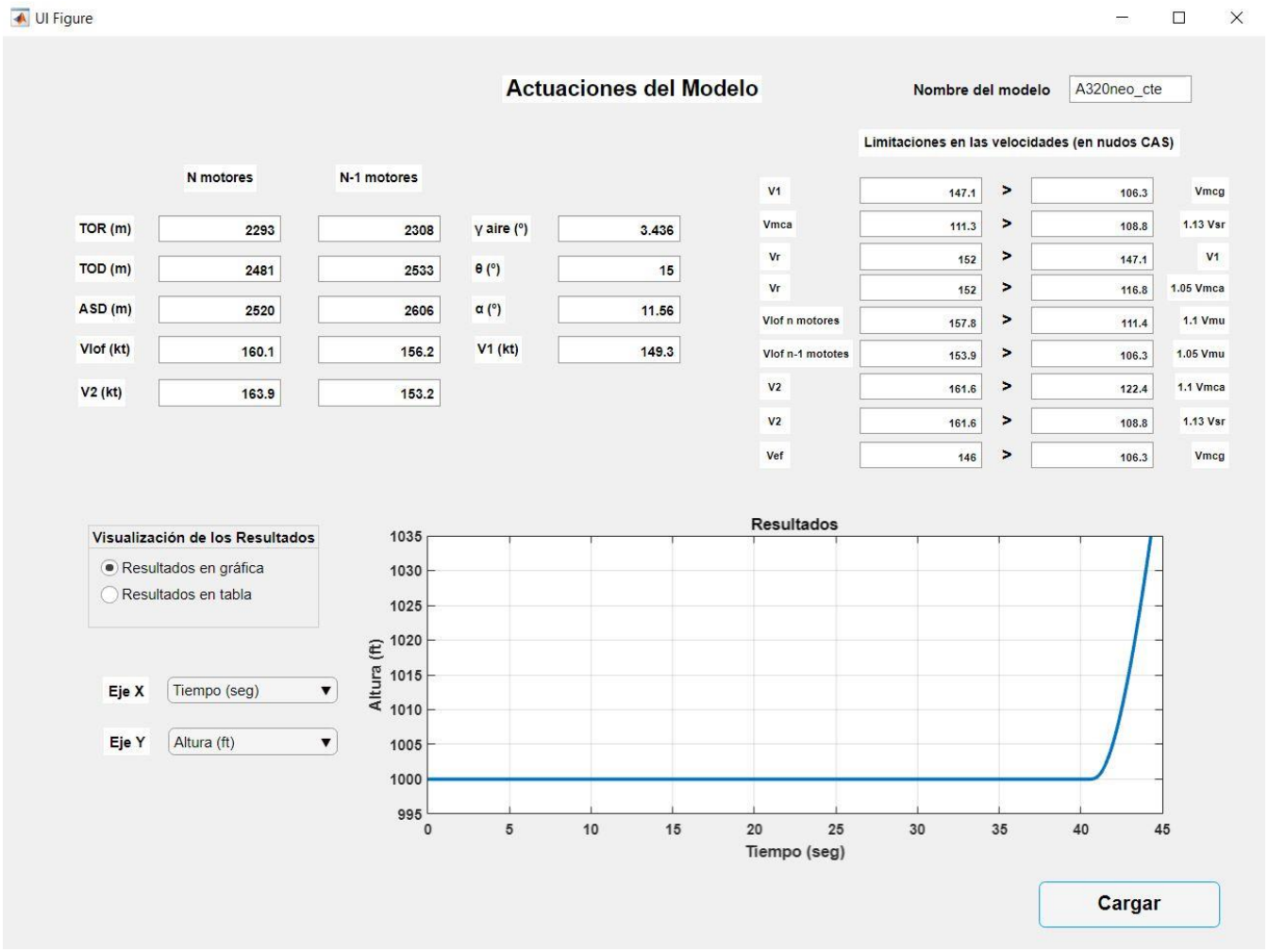


Figura 111. Ventana actuaciones del modelo

Vemos de nuevo como somos capaces de llegar a resultados muy próximos a los reales, por lo que el programa realmente es capaz de producir buenos resultados. Volvemos a recalcar que para obtener estos resultados es necesario tener un buen modelo que se ajuste de forma adecuada a la maniobra real.

Con esto damos por concluido el capítulo 5, donde hemos llevado a cabo los objetivos 5 y 6 de forma simultánea, comparando los resultados que se obtienen de nuestra aplicación con datos reales de aviones reales, y a su vez, hemos realizado la modelización de dos aviones característicos turbofán y turbohélice y hemos visto cómo funciona la aplicación a través de estos ejemplos.

Como conclusión final, que detallaremos más en el capítulo 6, podemos decir que nuestra aplicación proporciona resultados que se acercan bastante a los resultados reales, pero necesita que los parámetros de entrada sean precisos, ya que unos malos parámetros de entrada pueden dar lugar a malos resultados.

Entonces, teniendo en cuenta la complejidad del problema y que nuestro modelo es un modelo simplificado del problema real, podemos decir que el resultado final cumple los objetivos del proyecto, habiendo desarrollado una aplicación sencilla y amigable que permite obtener actuaciones certificadas con una precisión más que aceptable y que además permite un buen análisis de los modelos.

Para concluir, pasaremos al capítulo 6 donde hablaremos de las conclusiones finales del proyecto y se expondrán algunas ideas o conceptos que sería interesante añadir a la aplicación de cara al futuro o zonas a mejorar de la misma.

6. CONCLUSIONES

En este capítulo llevaremos a cabo el objetivo 7, el cual incluye las conclusiones y el juicio crítico del autor sobre el proyecto.

Una vez que hemos visto los resultados y analizado los dos ejemplos de aviones comerciales podemos decir que el programa, en líneas generales, da unos resultados bastante próximos a los reales, pero es necesario crear un modelo que se ajuste a la realidad.

Además, también hemos visto cómo podemos realizar un estudio de las tendencias de las actuaciones con diversas variables y que, en principio, lo observado tiene sentido físico, por lo que este objetivo también podría considerarse cumplido.

Otro punto que hemos experimentado es que la precisión de los resultados depende en gran medida de la correcta elección de las variables de entrada, por lo que es necesario ajustar los parámetros de entrada para aproximarse lo máximo posible al modelo real, aunque nuestro modelo sea uno simplificado.

Como aportación extra de este proyecto, se ha logrado modelar el perfil longitudinal de la pista de un aeródromo en concreto y poder calcular las actuaciones de un avión en dicha pista, lo que aporta a la aplicación una versatilidad muy interesante y que normalmente no se incluye en este tipo de cálculos de actuaciones certificadas.

Como líneas futuras, sería interesante ir mejorando el modelo añadiendo modelos propulsivos y aerodinámicos más próximos a la realidad, un modelo dinámico más complejo y real y mejorar otros aspectos con el fin de acercar más los resultados a los reales, y poder crear listas de resultados, donde las actuaciones certificadas vengan en función de diversas variables, que es como vienen expresados en los manuales de vuelo y de actuaciones en la realidad. Sin embargo, para este trabajo de fin de grado se ha considerado suficiente con obtener estas actuaciones para unos datos en concreto.

En conclusión, el problema que se ha intentado resolver en este trabajo es muy complejo, y depende de muchos parámetros y variables, muchos de ellos experimentales, por lo que su modelización es complicada. Sin embargo, dentro del ámbito académico e ingenieril, esta aplicación podría ser una forma sencilla de obtener primeras aproximaciones bastante precisas de las actuaciones de una

aeronave en una supuesta pista infinita, además de que, como opción adicional, también se permite recrear el perfil longitudinal de la pista de un aeropuerto real.

También cabe destacar la utilidad académica y de análisis que permite el programa, al contar con opciones que permiten un visualizado de la influencia de los parámetros de forma gráfica, facilitándose de esta forma la comparación de resultados y su optimización.

Por lo tanto, podemos concluir con que los objetivos del trabajo se han cumplido.

Veamos ahora una pequeña lista de posibles puntos a mejorar para el futuro.

6.1 Puntos a mejorar para el futuro

6.1.1 Gradiente del Viento

Un fenómeno meteorológico que afecta de manera sustancial al vuelo de un avión es el conocido como viento de cizalladura o gradiente del viento.

Se puede definir como el cambio en dirección y/o intensidad del viento en un plano y en poca distancia. Por lo tanto, puede ser vertical, horizontal, o ambas. Al estar teniendo en cuenta solo viento paralelo a la pista en este trabajo, el tipo de gradiente de viento que nos interesaría estudiar sería el cambio de la intensidad del viento con respecto a la altura sobre la pista.

Según la OACI, [23], podemos definir distintos niveles de intensidad en el viento de cizalladura para una distancia vertical de 100 pies:

- Ligera: Una variación menor a 4 nudos en la velocidad del viento.
- Moderada: Una variación entre 5 y 8 nudos en la velocidad del viento.
- Fuerte: Una variación de 9 a 12 nudos en la velocidad del viento.
- Muy fuerte: Una variación mayor a 12 nudos en la velocidad del viento.

Existen modelos que nos permiten conocer aproximaciones sobre los gradientes de viento en función de diversas variables y en función del terreno sobre el que nos encontremos. Algunos de estos modelos están incluso implementados a las ecuaciones dinámicas del problema de despegue de un avión, con el objetivo de estudiar una actuación segura del avión bajo este tipo de vientos.

Sin embargo, estos modelos son complicados de implementar, ya que se basan en una gran cantidad de variables y de datos experimentales y tienen cierta complejidad matemática.

Además, vemos que los niveles de intensidad de la cizalladura están considerados sobre una distancia de 100 pies, y recordemos que este trabajo solo considera una distancia vertical de 35 pies, por lo que se puede considerar para simplificar este trabajo que al ser una distancia pequeña este fenómeno no afectará demasiado, aunque en la realidad no tiene por qué ser así y sería interesante poder estudiar su efecto.

Algunos modelos que hemos comentados están expuestos en los artículos [24] y [25], uno de ellos estudia el viento de cizalladura sobre distintos terrenos y el otro estudia su efecto sobre el despegue.

6.1.1 Mejorar la eficiencia de la aplicación

Sería interesante revisar y mejorar la eficiencia del código de la aplicación y de los cálculos que se realizan.

Ahora mismo la aplicación es eficiente y no presenta problemas de rendimiento en absoluto, pero es cierto que su código se puede mejorar de forma que sea más eficiente y sencillo. Por ejemplo, se podría intentar reducir el número de bloques en el apartado de cálculo y agruparlos todos en uno o buscar otra forma de reducir su número. Detalles como esos ayudarían a reducir los tiempos de cálculo, incrementando su eficiencia.

Sin embargo, como ya hemos comentado no es una mejora necesaria, pero sí que sería algo a considerar de cara al futuro si se pretende mejorar al máximo esta aplicación.

6.1.2 Mejora de modelos propulsivo y aerodinámico

Para poder crear un modelo más complejo y próximo a la realidad sería necesario sustituir los actuales modelos propulsivo y aerodinámico, que son modelos aproximados que dependen principalmente de la velocidad de la aeronave, por unos modelos más complejos donde se tengan en cuenta más variable y efectos que en la realidad también influyen en las actuaciones en el despegue.

Ya lo comentamos antes, pero en los manuales de vuelo y de actuaciones existen listas de resultados de las actuaciones en función de una serie de variables. Para poder lograr esto sería imprescindible añadir unos modelos que tengan en cuenta esta serie de variables, de forma que queden establecidas las dependencias entre los resultados y estas variables.

Sin embargo, el añadir estos modelos supondría un aumento considerable de la dificultad, ya que, por ejemplo, un modelo propulsivo complejo y que tenga en cuenta todos los posibles parámetros que entran en juego en un despegue supone de por sí solo ya una dificultad considerable para modelarlo. Si a ello le sumásemos a su vez un modelo aerodinámico que tenga en cuenta inclinaciones de flaps, tren de aterrizaje y demás la dificultad aumenta exponencialmente.

6.1.3 Mejora de la interfaz

Aunque no es un objetivo primordial de este trabajo el tener una interfaz estéticamente atractiva, sí que se buscaba que fuese una interfaz sencilla y amigable, intuitiva para el usuario.

Por ello, como mejora futura también estaría bien que esta interfaz se mejorase, añadiendo una calidad estética superior a la actual y que está limitada en parte a las opciones que ofrece el bloque App Designer de Matlab.

6.1.4 Mejora de la altimetría, inclusión del QNH

Una mejora interesante sería la inclusión de un sistema o función que fuese capaz de leer como dato de entrada el QNH y la temperatura del aeropuerto donde se quiera realizar el estudio y esta función nos diese como resultado la presión altitud del aeropuerto.

En la realidad, para conocer la presión altitud del aeropuerto se utiliza este parámetro, que depende de muchos factores, tanto de posicionamiento como meteorológicos. Para explicar de una forma sencilla y simplificada, podríamos decir que el QNH es la presión que identifica el nivel medio del mar. Si fijamos la referencia barométrica como el QNH proporcionado, el altímetro marcará la altitud a la que nos encontramos respecto al nivel medio del mar.

Aunque nuestro programa está enfocado a una aplicación ingenieril y académica, esta mejora podría ser interesante de cara al futuro, ya que de esta forma se añadiría una cierta aplicación operativa al programa.

6.1.5 Inclusión de limitaciones a la velocidad por frenos y neumáticos

En los manuales de vuelo y de actuaciones de las aeronaves, dos factores limitantes a tener en cuenta y que aparecen de forma habitual son la velocidad de máxima energía de frenado V_{MBE} y la velocidad máxima de frenado V_{TIRE}.

Estas velocidades fueron comentadas en el capítulo de análisis normativo, y ya entonces se explicó que podían ser una limitación durante el despegue (la más habitual es V_{MBE}, la velocidad máxima de neumáticos no suele limitar en la mayoría de las ocasiones, pero, aun así, hay que tenerla en cuenta).

Sin embargo, para nuestro programa su inclusión fue tenida en cuenta en un punto muy avanzado del programa, por lo que se decidió no incluir dichas limitaciones. Sin embargo, es importante conocer su existencia y relevancia, y constituiría una mejora de futuro para la aplicación.

6.1.6 Inclusión de limitación al ángulo de asiento máximo

Ya comentamos en el capítulo 3 que durante las fases de rotación y vuelo existe el riesgo de que, si se ha producido una rotación hasta un ángulo de asiento excesivamente alto, se produzca una colisión entre el cono de cola del avión y el suelo.

Esta limitación debe ser tenida en cuenta, limitando la ley de rotación de manera que exista un ángulo de asiento máximo a partir del cual no se permite rotar, ya que el cono de cola podría chocar.

La necesidad de incluir esta limitación fue descubierta en un punto muy avanzado del programa, por lo que se decidió no incluirla debido al trabajo que conllevaría variar el programa en un estado avanzado.

Sin embargo, es importante conocer la existencia de esta limitación y ser conscientes de que el ángulo de asiento no puede ser todo lo grande que queramos. De cara al futuro, esta mejora sería imprescindible.

BIBLIOGRAFÍA

[1] «Página Principal de Matlab,» [En línea]. Available: https://es.mathworks.com/products/matlab.html.

[2] «Página principal de App Desginer,» [En línea]. Available: https://es.mathworks.com/products/matlab/app-designer.html?s_tid=srchtitle.

[3] «Norma Europea CS 25 Última Actualización,» [En línea]. Available: https://www.easa.europa.eu/document-library/certification-specifications/cs-25-amendment-25.

[4] «Norma Americana CFR Part 25,» [En línea]. Available: https://www.ecfr.gov/cgi-bin/text-idx?node=14:1.0.1.3.11.

[5] A. I. Carmona, Aerodinámica y Actuaciones del Avión, Madrid: Paraninfo, 2000.

[6] «Página de comando ode45,» [En línea]. Available: https://es.mathworks.com/help/matlab/ref/ode45.html.

[7] S. E. Roncero, «Página de la Asignatura Cálculo de Aeronaves de la Universidad de Sevilla,» [En línea]. Available: http://www.aero.us.es/adesign/Paginas/Diapositivas.html.

[8] N. Leland M. y C. Grant E., «Takeoff and Landing Analysis,» de Fundamentals of Aircraft and Airship Design, Virginia, AIAA Education Series, 1975, pp. 255-283.

[9] «Página oficial de Matlab del comando pchip,» [En línea]. Available: https://es.mathworks.com/help/matlab/ref/pchip.html.

[10] «Página de Matlab dobre el comando polyfit,» [En línea]. Available: https://es.mathworks.com/help/matlab/ref/polyfit.html.

[11] «Página de Matlab sobre el comando polyval,» [En línea]. Available: https://es.mathworks.com/help/matlab/ref/polyval.html.

[12] «Página de Matlab sobre el comando odeset y events,» [En línea]. Available: https://es.mathworks.com/help/matlab/math/ode-event-location.html.

[13] «Página del fabricante del ATR72,» [En línea]. Available: https://www.atr-aircraft.com/our-aircraft/atr-72-600/.

[14] «Wikipedia sobre el ATR72,» [En línea]. Available: https://es.wikipedia.org/wiki/ATR_72#cite_note-guide-69.

[15] «Manual de actuaciones para ATR,» [En línea]. Available: https://es.scribd.com/document/369931415/ATR-Aircraft-Performance-2.

[16] «FCOM del ATR 72,» [En línea]. Available: https://skytestdotblog.files.wordpress.com/2019/03/atr-72-600-fcom-2.pdf.

[17] «Página oficial donde consultar los aeropuertos de España,» [En línea]. Available: https://ais.enaire.es/AIP/AIPS/AMDT_333_2020_AIRAC_10_2020/AIP.html.

[18] «Página oficial del fabricante de Airbus,» [En línea]. Available: https://www.airbus.com/aircraft/passenger-aircraft/a320-family/a320neo.html.

[19] «Página con información sobre el Airbus A320,» [En línea]. Available: https://www.skytamer.com/Airbus_A320.html.

[20] J. M. H. J. E. Junzi Sun, «Árticulo de la Delf University of Technology, the Netherlands,» [En línea]. Available: https://www.sesarju.eu/sites/default/files/documents/sid/2018/papers/SIDs_2018_paper_75.pdf.

[21] «Página con datos sobre las actuaciones del A320,» [En línea]. Available: https://www.cockpitseeker.com/wp-content/uploads/goodies/ac/a320/pdf/Print_Only/PTM%20with%20airbus%20doc/pdf/U0S2SP0-L.pdf.

[22] «Página sobre las limitaciones en actuaciones del A320,» [En línea]. Available: https://www.theairlinepilots.com/forumarchive/a320/a320-limitations.pdf.

[23] «Documento del gobierno sobre vientos de cizalladura y turbulencia,» [En línea]. Available: https://repositorio.aemet.es/bitstream/20.500.11765/9383/1/cizalladura_turbulencia_BGonzalez_JCOPAC2014.pdf.

[24] «Documento sobre el efecto del viento de cizalladura sobre distintos terrenos,» [En línea]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.574.7468&rep=rep1&type=pdf.

[25] «Página con un estudio sobre el despegue de un avión con viento de gradiente,» [En línea]. Available: https://www.sciencedirect.com/science/article/pii/S0895717702001644?via%3Dihub.

[26] «Documento con datos sobre el Airbus A320,» [En línea]. Available: https://www.airbus.com/content/dam/corporate-topics/publications/backgrounders/techdata/aircraft_characteristics/Airbus-Commercial-Aircraft-AC-A320.pdf.

[27] «Manual de vuelo del ATR 72,» [En línea]. Available: https://aviation-is.better-than.tv/atr72fcom.pdf.

ANEXOS

A. Códigos de la aplicación

- Programa_actuaciones

```
Ve=zeros(40,1);

Ve(1)=peso;

Ve(2)=anguloempuje*pi/180;

Ve(3)=supalar;

Ve(4)=envergadura;

Ve(5)=anguloflecha*pi/180;

Ve(6)=Cd0;

Ve(7)=Cdflaps;

Ve(8)=Cdgear;

Ve(9)=k;

Ve(10)=alphanolift*pi/180;

Ve(11)=Temperatura;

Ve(12)=0.3048*Presion_Altitud;

Ve(13)=0.514444*Velocidad_viento;

if strcmp(tipomotor,'Turbofan')
    Ve(14)=Posicionpalanca;
    Ve(15)=Tsl;
    Ve(16)=Nmotores;
    if strcmp(leyrotacion,'Estándar')
        Ve(17)=anguloinicialrotacion*pi/180;
        Ve(18)=Coeficientederozamiento;
        angulofinalrotacion=12*pi/180;
        Tiempototalrotacion=3.5;
        Ve(19)=angulofinalrotacion;
        Ve(20)=Tiempototalrotacion;
    elseif strcmp(leyrotacion,'Personalizada')
        Ve(17)=anguloinicialrotacion*pi/180;
        Ve(18)=Coeficientederozamiento;
```

```
        Ve(19)=angulofinalrotacion*pi/180;
        Ve(20)=Tiempototalrotacion;
    end
    if strcmp(tipopendiente,'Pendiente constante')
        Ve(21)=atan(Pendiente/100);
        Vr_TAS=deCASaTAS(0.514444*Velocidadrotacion,Presion_Altitud,1.4);
        Vr_CAS=0.514444*Velocidadrotacion;
        Ve(22)=Vr_TAS;
        if strcmp(Funciones,'Si')
            funcionVmcg=str2func( ['@(P)',FuncionVmcg]);
            funcionVmca=str2func( ['@(P)',FuncionVmca]);
            funcionVsr=str2func( ['@(P)',FuncionVsr]);
            funcionVmu=str2func( ['@(P)',FuncionVmu]);
            Ve(24)=0.514444*funcionVmcg(Ve(1));
            Ve(25)=0.514444*funcionVmca(Ve(1));
            Ve(26)=0.514444*funcionVsr(Ve(1));
            Ve(27)=0.514444*funcionVmu(Ve(1));
        else
            Ve(24)=0.514444*Velocidadmcg;
            Ve(25)=0.514444*Velocidadmca;
            Ve(26)=0.514444*Velocidadsr;
            Ve(27)=0.514444*Velocidadmu;
        end
        Vef_TAS=deCASaTAS(0.514444*Velocidadfallomotor,Presion_Altitud,1.4);
        Vef_CAS=0.514444*Velocidadfallomotor;
        Ve(28)=Vef_TAS;
        Ve(30)=Trec;
        Ve(29)=Coeficientefrenada;
        [Resultados,TOR,Vlof,error0]=calculadorpend_const_turbofan(Ve);
        Vlof_CAS=deCASaTAS(Vlof,Presion_Altitud,1.4);
        save([Nombre_modelo '_Resultados','.mat'],'Resultados')
        save([Nombre_modelo '_Resultados','.mat'],'TOR','-append')
        save([Nombre_modelo '_Resultados','.mat'],'Vlof','-append')
        save([Nombre_modelo '_Resultados','.mat'],'error0','-append')
        TOD=1.15*str2num(Resultados(end,2));
        save([Nombre_modelo '_Resultados','.mat'],'TOD','-append')

[TOD_sinmotor,TOR_sinmotor,Vlof_sinmotor,error_sinmotor,V2_sinmotor,V1]=calcul
adorpend_TOD_sinmotor_const_turbofan(Ve);
        Vlof_sinmotor_CAS=deCASaTAS(Vlof_sinmotor,Presion_Altitud,1.4);
        save([Nombre_modelo '_Resultados','.mat'],'TOD_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'TOR_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'Vlof_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'error_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'V2_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'V1','-append')
        Ve(23)=V1;
        V1_CAS=deTASaCAS(V1,Presion_Altitud,1.4);
        [ASD,ASD_t]=calculador_ASD_pendconst_turbofan(Ve);
        save([Nombre_modelo '_Resultados','.mat'],'ASD','-append')
        save([Nombre_modelo '_Resultados','.mat'],'ASD_t','-append')

[ASD_sinmotor,ASD_t_sinmotor]=calculador_ASD_sinmotor_pendconst_turbofan(Ve);
        save([Nombre_modelo '_Resultados','.mat'],'ASD_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'ASD_t_sinmotor','-append')
```

```
if V1_CAS<Ve(24)
    error1=1;
else
    error1=0;
end
save([Nombre_modelo '_Resultados','.mat'],'error1','-append')
if Ve(25)<1.13*Ve(26)
    error2=1;
else
    error2=0;
end
save([Nombre_modelo '_Resultados','.mat'],'error2','-append')
if Vr_CAS<V1_CAS
    error3=1;
else
    error3=0;
end
save([Nombre_modelo '_Resultados','.mat'],'error3','-append')
if Vr_CAS<1.05*Ve(25)
    error4=1;
else
    error4=0;
end
save([Nombre_modelo '_Resultados','.mat'],'error4','-append')
if Vlof_CAS<1.1*Ve(27)
    error5=1;
else
    error5=0;
end
save([Nombre_modelo '_Resultados','.mat'],'error5','-append')
if Vlof_sinmotor_CAS<1.05*Ve(27)
    error6=1;
else
    error6=0;
end
save([Nombre_modelo '_Resultados','.mat'],'error6','-append')
V2=str2num(Resultados(end,5));
V2_CAS=deTASaCAS(V2,Presion_Altitud,1.4);
save([Nombre_modelo '_Resultados','.mat'],'V2','-append')
if V2_CAS<1.1*Ve(25)
    error7=1;
else
    error7=0;
end
save([Nombre_modelo '_Resultados','.mat'],'error7','-append')
if V2_CAS<1.13*Ve(26)
    error8=1;
else
    error8=0;
end
save([Nombre_modelo '_Resultados','.mat'],'error8','-append')
if Vef_CAS<Ve(24)
```

```
            error9=1;
        else
            error9=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error9','-append')

    elseif strcmp(tipopendiente,'Pendiente variable')
        Vr_TAS=deCASaTAS(0.514444*Velocidadrotacion,Presion_Altitud,1.4);
        Vr_CAS=0.514444*Velocidadrotacion;
        Ve(21)=Vr_TAS;
        if strcmp(Funciones,'Si')
            funcionVmcg=str2func( ['@(P)',FuncionVmcg]);
            funcionVmca=str2func( ['@(P)',FuncionVmca]);
            funcionVsr=str2func( ['@(P)',FuncionVsr]);
            funcionVmu=str2func( ['@(P)',FuncionVmu]);
            Ve(23)=0.514444*funcionVmcg(Ve(1));
            Ve(24)=0.514444*funcionVmca(Ve(1));
            Ve(25)=0.514444*funcionVsr(Ve(1));
            Ve(26)=0.514444*funcionVmu(Ve(1));
        else
            Ve(23)=0.514444*Velocidadmcg;
            Ve(24)=0.514444*Velocidadmca;
            Ve(25)=0.514444*Velocidadsr;
            Ve(26)=0.514444*Velocidadmu;
        end
        Vef_TAS=deCASaTAS(0.514444*Velocidadfallomotor,Presion_Altitud,1.4);
        Vef_CAS=0.514444*Velocidadfallomotor;
        Ve(27)=Vef_TAS;
        Ve(29)=Trec;
        Ve(28)=Coeficientefrenada;
        if strcmp(tipodatos,'Pendientes y distancias')
            xpista=[0 Pendientes_Distancias(:,2)'];
            ypista(1)=Ve(12);
            for i=2:length(xpista)
                ypista(i)=ypista(i-1)+(xpista(i)-xpista(i-
1))*Pendientes_Distancias(i-1,1)/100;
            end
        elseif strcmp(tipodatos,'Distancias y alturas')
            xpista=Distancias_Alturas(:,1)';
            ypista=Distancias_Alturas(:,2)';
        end

[Resultados,TOR,Vlof,error0]=calculadorpend_var_turbofan(Ve,xpista,ypista);
        Vlof_CAS=deCASaTAS(Vlof,Presion_Altitud,1.4);
        save([Nombre_modelo '_Resultados','.mat'],'Resultados')
        save([Nombre_modelo '_Resultados','.mat'],'TOR','-append')
        save([Nombre_modelo '_Resultados','.mat'],'Vlof','-append')
        save([Nombre_modelo '_Resultados','.mat'],'error0','-append')
        TOD=1.15*str2num(Resultados(end,2));
        save([Nombre_modelo '_Resultados','.mat'],'TOD','-append')

[TOD_sinmotor,TOR_sinmotor,Vlof_sinmotor,error_sinmotor,V2_sinmotor,V1]=calcul
adorpend_TOD_sinmotor_var_turbofan(Ve,xpista,ypista);
        Vlof_sinmotor_CAS=deCASaTAS(Vlof_sinmotor,Presion_Altitud,1.4);
        save([Nombre_modelo '_Resultados','.mat'],'TOD_sinmotor','-append')
```

```
        save([Nombre_modelo '_Resultados','.mat'],'TOR_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'Vlof_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'error_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'V2_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'V1','-append')
        Ve(22)=V1;
        V1_CAS=deTASaCAS(V1,Presion_Altitud,1.4);
        [ASD,ASD_t]=calculador_ASD_pendvar_turbofan(Ve,xpista,ypista);
        save([Nombre_modelo '_Resultados','.mat'],'ASD','-append')
        save([Nombre_modelo '_Resultados','.mat'],'ASD_t','-append')

[ASD_sinmotor,ASD_t_sinmotor]=calculador_ASD_sinmotor_pendvar_turbofan(Ve,xpis
ta,ypista);
        save([Nombre_modelo '_Resultados','.mat'],'ASD_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'ASD_t_sinmotor','-append')

        if V1_CAS<Ve(23)
            error1=1;
        else
            error1=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error1','-append')
        if Ve(24)<1.13*Ve(25)
            error2=1;
        else
            error2=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error2','-append')
        if Vr_CAS<V1_CAS
            error3=1;
        else
            error3=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error3','-append')
        if Vr_CAS<1.05*Ve(24)
            error4=1;
        else
            error4=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error4','-append')
        if Vlof_CAS<1.1*Ve(26)
            error5=1;
        else
            error5=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error5','-append')
        if Vlof_sinmotor_CAS<1.05*Ve(26)
            error6=1;
        else
            error6=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error6','-append')
        V2=str2num(Resultados(end,5));
        V2_CAS=deTASaCAS(V2,Presion_Altitud,1.4);
        save([Nombre_modelo '_Resultados','.mat'],'V2','-append')
```

```
        if V2_CAS<1.1*Ve(24)
            error7=1;
        else
            error7=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error7','-append')
        if V2_CAS<1.13*Ve(25)
            error8=1;
        else
            error8=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error8','-append')
        if Vef_CAS<Ve(23)
            error9=1;
        else
            error9=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error9','-append')

    end
elseif strcmp(tipomotor,'Turbohélice')
    Ve(14)=Posicionpalanca;
    Ve(15)=Psl;
    Ve(16)=nup;
    Ve(17)=Nmotores;
    if strcmp(leyrotacion,'Estándar')
        Ve(18)=anguloinicialrotacion*pi/180;
        Ve(19)=Coeficientederozamiento;
        angulofinalrotacion=12*pi/180;
        Tiempototalrotacion=3.5;
        Ve(20)=angulofinalrotacion;
        Ve(21)=Tiempototalrotacion;
    elseif strcmp(leyrotacion,'Personalizada')
        Ve(18)=anguloinicialrotacion*pi/180;
        Ve(19)=Coeficientederozamiento;
        Ve(20)=angulofinalrotacion*pi/180;
        Ve(21)=Tiempototalrotacion;
    end
    if strcmp(tipopendiente,'Pendiente constante')
        Ve(22)=atan(Pendiente/100);
        Vr_TAS=deCASaTAS(0.514444*Velocidadrotacion,Presion_Altitud,1.4);
        Vr_CAS=0.514444*Velocidadrotacion;
        Ve(23)=Vr_TAS;
        if strcmp(Funciones,'Si')
            funcionVmcg=str2func( ['@(P)',FuncionVmcg]);
            funcionVmca=str2func( ['@(P)',FuncionVmca]);
            funcionVsr=str2func( ['@(P)',FuncionVsr]);
            funcionVmu=str2func( ['@(P)',FuncionVmu]);
            Ve(25)=0.514444*funcionVmcg(Ve(1));
            Ve(26)=0.514444*funcionVmca(Ve(1));
            Ve(27)=0.514444*funcionVsr(Ve(1));
            Ve(28)=0.514444*funcionVmu(Ve(1));
        else
            Ve(25)=0.514444*Velocidadmcg;
            Ve(26)=0.514444*Velocidadmca;
```

```
            Ve(27)=0.514444*Velocidadsr;
            Ve(28)=0.514444*Velocidadmu;
        end
        Vef_TAS=deCASaTAS(0.514444*Velocidadfallomotor,Presion_Altitud,1.4);
        Vef_CAS=0.514444*Velocidadfallomotor;
        Ve(29)=Vef_TAS;
        Ve(31)=Trec;
        Ve(30)=Coeficientefrenada;
        [Resultados,TOR,Vlof,error0]=calculadorpend_const_turbohelice(Ve);
        Vlof_CAS=deCASaTAS(Vlof,Presion_Altitud,1.4);
        save([Nombre_modelo '_Resultados','.mat'],'Resultados')
        save([Nombre_modelo '_Resultados','.mat'],'TOR','-append')
        save([Nombre_modelo '_Resultados','.mat'],'Vlof','-append')
        save([Nombre_modelo '_Resultados','.mat'],'error0','-append')
        TOD=1.15*str2num(Resultados(end,2));
        save([Nombre_modelo '_Resultados','.mat'],'TOD','-append')

[TOD_sinmotor,TOR_sinmotor,Vlof_sinmotor,error_sinmotor,V2_sinmotor,V1]=calcul
adorpend_TOD_sinmotor_const_turbohelice(Ve);
        Vlof_sinmotor_CAS=deCASaTAS(Vlof_sinmotor,Presion_Altitud,1.4);
        save([Nombre_modelo '_Resultados','.mat'],'TOD_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'TOR_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'Vlof_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'error_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'V2_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'V1','-append')
        Ve(24)=V1;
        V1_CAS=deTASaCAS(V1,Presion_Altitud,1.4);
        [ASD,ASD_t]=calculador_ASD_pendconst_turbohelice(Ve);
        save([Nombre_modelo '_Resultados','.mat'],'ASD','-append')
        save([Nombre_modelo '_Resultados','.mat'],'ASD_t','-append')

[ASD_sinmotor,ASD_t_sinmotor]=calculador_ASD_sinmotor_pendconst_turbohelice(Ve
);
        save([Nombre_modelo '_Resultados','.mat'],'ASD_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'ASD_t_sinmotor','-append')

        if V1_CAS<Ve(25)
            error1=1;
        else
            error1=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error1','-append')
        if Ve(26)<1.13*Ve(27)
            error2=1;
        else
            error2=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error2','-append')
        if Vr_CAS<V1_CAS
            error3=1;
        else
            error3=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error3','-append')
```

```
        if Vr_CAS<1.05*Ve(26)
            error4=1;
        else
            error4=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error4','-append')
        if Vlof_CAS<1.1*Ve(28)
            error5=1;
        else
            error5=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error5','-append')
        if Vlof_sinmotor_CAS<1.05*Ve(28)
            error6=1;
        else
            error6=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error6','-append')
        V2=str2num(Resultados(end,5));
        V2_CAS=deTASaCAS(V2,Presion_Altitud,1.4);
        save([Nombre_modelo '_Resultados','.mat'],'V2','-append')
        if V2_CAS<1.1*Ve(26)
            error7=1;
        else
            error7=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error7','-append')
        if Ve(17)<3
            if V2_CAS<1.13*Ve(27)
                error8=1;
            else
                error8=0;
            end
        else
            if V2_CAS<1.08*Ve(27)
                error8=1;
            else
                error8=0;
            end
        end
        save([Nombre_modelo '_Resultados','.mat'],'error8','-append')
        if Vef_CAS<Ve(25)
            error9=1;
        else
            error9=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error9','-append')

    elseif strcmp(tipopendiente,'Pendiente variable')
        Vr_TAS=deCASaTAS(0.514444*Velocidadrotacion,Presion_Altitud,1.4);
        Vr_CAS=0.514444*Velocidadrotacion;
        Ve(22)=Vr_TAS;
        if strcmp(Funciones,'Si')
            funcionVmcg=str2func( ['@(P)',FuncionVmcg]);
            funcionVmca=str2func( ['@(P)',FuncionVmca]);
```

```
            funcionVsr=str2func( ['@(P)',FuncionVsr]);
            funcionVmu=str2func( ['@(P)',FuncionVmu]);
            Ve(24)=0.514444*funcionVmcg(Ve(1));
            Ve(25)=0.514444*funcionVmca(Ve(1));
            Ve(26)=0.514444*funcionVsr(Ve(1));
            Ve(27)=0.514444*funcionVmu(Ve(1));
        else
            Ve(24)=0.514444*Velocidadmcg;
            Ve(25)=0.514444*Velocidadmca;
            Ve(26)=0.514444*Velocidadsr;
            Ve(27)=0.514444*Velocidadmu;
        end
        Vef_TAS=deCASaTAS(0.514444*Velocidadfallomotor,Presion_Altitud,1.4);
        Vef_CAS=0.514444*Velocidadfallomotor;
        Ve(28)=Vef_TAS;
        Ve(30)=Trec;
        Ve(29)=Coeficientefrenada;
        if strcmp(tipodatos,'Pendientes y distancias')
            xpista=[0 Pendientes_Distancias(:,2)'];
            ypista(1)=Ve(12);
            for i=2:length(xpista)
            ypista(i)=ypista(i-1)+(xpista(i)-xpista(i-
1))*Pendientes_Distancias(i-1,1)/100;
            end
        elseif strcmp(tipodatos,'Distancias y alturas')
            xpista=Distancias_Alturas(:,1)';
            ypista=Distancias_Alturas(:,2)';
        end

[Resultados,TOR,Vlof,error0]=calculadorpend_var_turbohelice(Ve,xpista,ypista);
        Vlof_CAS=deCASaTAS(Vlof,Presion_Altitud,1.4);
        save([Nombre_modelo '_Resultados','.mat'],'Resultados')
        save([Nombre_modelo '_Resultados','.mat'],'TOR','-append')
        save([Nombre_modelo '_Resultados','.mat'],'Vlof','-append')
        save([Nombre_modelo '_Resultados','.mat'],'error0','-append')
        TOD=1.15*str2num(Resultados(end,2));
        save([Nombre_modelo '_Resultados','.mat'],'TOD','-append')

[TOD_sinmotor,TOR_sinmotor,Vlof_sinmotor,error_sinmotor,V2_sinmotor,V1]=calcul
adorpend_TOD_sinmotor_var_turbohelice(Ve,xpista,ypista);
        Vlof_sinmotor_CAS=deCASaTAS(Vlof_sinmotor,Presion_Altitud,1.4);
        save([Nombre_modelo '_Resultados','.mat'],'TOD_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'TOR_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'Vlof_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'error_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'V2_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'V1','-append')
        Ve(23)=V1;
        V1_CAS=deTASaCAS(V1,Presion_Altitud,1.4);
        [ASD,ASD_t]=calculador_ASD_pendvar_turbohelice(Ve,xpista,ypista);
        save([Nombre_modelo '_Resultados','.mat'],'ASD','-append')
        save([Nombre_modelo '_Resultados','.mat'],'ASD_t','-append')

[ASD_sinmotor,ASD_t_sinmotor]=calculador_ASD_sinmotor_pendvar_turbohelice(Ve,x
pista,ypista);
```

```
        save([Nombre_modelo '_Resultados','.mat'],'ASD_sinmotor','-append')
        save([Nombre_modelo '_Resultados','.mat'],'ASD_t_sinmotor','-append')

        if V1_CAS<Ve(24)
            error1=1;
        else
            error1=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error1','-append')
        if Ve(25)<1.13*Ve(26)
            error2=1;
        else
            error2=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error2','-append')
        if Vr_CAS<V1_CAS
            error3=1;
        else
            error3=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error3','-append')
        if Vr_CAS<1.05*Ve(25)
            error4=1;
        else
            error4=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error4','-append')
        if Vlof_CAS<1.1*Ve(27)
            error5=1;
        else
            error5=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error5','-append')
        if Vlof_sinmotor_CAS<1.05*Ve(27)
            error6=1;
        else
            error6=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error6','-append')
        V2=str2num(Resultados(end,5));
        V2_CAS=deTASaCAS(V2,Presion_Altitud,1.4);
        save([Nombre_modelo '_Resultados','.mat'],'V2','-append')
        if V2_CAS<1.1*Ve(25)
            error7=1;
        else
            error7=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error7','-append')
        if Ve(17)<3
            if V2_CAS<1.13*Ve(26)
                error8=1;
            else
                error8=0;
            end
        else
```

```
            if V2_CAS<1.08*Ve(26)
                error8=1;
            else
                error8=0;
            end
        end
        save([Nombre_modelo '_Resultados','.mat'],'error8','-append')
        if Vef_CAS<Ve(24)
            error9=1;
        else
            error9=0;
        end
        save([Nombre_modelo '_Resultados','.mat'],'error9','-append')

    end
end
```

- Funciones Calculador ASD con n motores

```
function [ASD,ASD_t]=calculador_ASD_pendconst_turbofan(Ve)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);Tsl=Ve(16)*Ve(15);h0=Ve(12);mu=V
e(18);
theta0=Ve(17);

global V1
V1=Ve(23)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
phi=Ve(21);rhosl=densidadmar;
fprima=tan(phi);fdosprima=0;

x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod_ASD);
Optparada=odeset('Events',@myEventparada_ASD);

[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_const_turbofan(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alp
```

```
ha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,psl,m,
g,mu,epsilon),t0rod,x0rod,Optrod);
x0rot=xrod(end,:)';

x_v1=x0rot(1)+V1*2;
xrot_v1=(x0rot(1):x_v1)';
[sizerot_v1,~]=size(xrot_v1);
xrot(:,1)=xrot_v1;
xrot(:,2)=V1*ones(sizerot_v1,1);
trot=(0:(2/(sizerot_v1-1)):2)';

x0parada=[xrot(end,1);V1];
t0parada=[0 50];
mu=Ve(29);
[tparada,xparada]=ode45(@(t,x)
funcionparada_pend_const(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alpha0,theta0,
S,rho,Cd0,deltaCdflap,deltaCdgear,k,m,g,mu,epsilon),t0parada,x0parada,Optparad
a);

ASD_t=ttotal(end);
xtotal=[xrod(:,1);xrot(:,1);xparada(:,1)];
ASD=xtotal(end);

end
```

```
function [ASD,ASD_t]=calculador_ASD_pendconst_turbohelice(Ve)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);nuP=Ve(16);Psl=Ve(17)*Ve(15);h0=
Ve(12);mu=Ve(19);
theta0=Ve(18);

global V1
V1=Ve(24)-Vw;


temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
phi=Ve(22);
fprima=tan(phi);fdosprima=0;

x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod_ASD);
Optparada=odeset('Events',@myEventparada_ASD);
```

```
[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_const_turbohelice(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,
alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,psl,m
,g,mu,epsilon),t0rod,x0rod,Optrod);
x0rot=xrod(end,:)';

x_v1=x0rot(end,:)+V1*2;
xrot_v1=(x0rot(end,:):x_v1)';
[sizerot_v1,~]=size(xrot_v1);
xrot(:,1)=xrod(end,1)+xrot_v1;
xrot(:,2)=V1*ones(sizerot_v1,1);
trot=(0:(2/(sizerot_v1-1)):2)';

x0parada=[xrot(end,1);V1];
t0parada=[0 50];
mu=Ve(30);
[tparada,xparada]=ode45(@(t,x)
funcionparada_pend_const(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alpha0,theta0,
S,rho,Cd0,deltaCdflap,deltaCdgear,k,m,g,mu,epsilon),t0parada,x0parada,Optparad
a);


ttotal=[trod;trot+trod(end);tparada+trod(end)+trot(end)];
ASD_t=ttotal(end);
xtotal=[xrod(:,1);xrot(:,1);xparada(:,1)];
ASD=xtotal(end);

end
```

```
function [ASD,ASD_t]=calculador_ASD_pendvar_turbofan(Ve,xpista,ypista)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);Tsl=Ve(16)*Ve(15);h0=Ve(12);mu=V
e(18);
theta0=Ve(17);

global V1
V1=Ve(22)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
rhosl=densidadmar;

step=1;
```

```
xxpista=xpista(1):step:xpista(end);
yypista=pchip(xpista,ypista,xxpista);
aux=atan(diff(yypista)./diff(xxpista));
phi=[aux(1) aux];
p_phi=polyfit(xxpista,phi,length(xpista)-1);
p_zg=polyfit(xxpista,yypista,length(xpista)-1);
fprima=diff(yypista)/step;
p_fprima=polyfit(xxpista(:,1:length(fprima)),fprima,length(xpista));
fdosprima=diff(fprima)/step;
p_fdosprima=polyfit(xxpista(:,1:length(fdosprima)),fdosprima,length(xpista));
x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod_ASD);
Optparada=odeset('Events',@myEventparada_ASD);

[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_var_turbofan(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta
,alpha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,ps
l,m,g,mu,epsilon),t0rod,x0rod,Optrod);
x0rot=xrod(end,:)';

x_v1=x0rot(1)+V1*2;
xrot_v1=(x0rot(1):x_v1)';
[sizerot_v1,~]=size(xrot_v1);
xrot(:,1)=xrot_v1;
xrot(:,2)=V1*ones(sizerot_v1,1);
trot=(0:(2/(sizerot_v1-1)):2)';

x0parada=[xrot(end,1);V1];
t0parada=[0 50];
mu=Ve(28);
[tparada,xparada]=ode45(@(t,x)
funcionparada_pend_var(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta,alpha0,the
ta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,m,g,mu,epsilon),t0parada,x0parada,Optp
arada);

ttotal=[trod;trot+trod(end);tparada+trod(end)+trot(end)];
ASD_t=ttotal(end);
xtotal=[xrod(:,1);xrot(:,1);xparada(:,1)];
ASD=xtotal(end);

end
```

```
function [ASD,ASD_t]=calculador_ASD_pendvar_turbohelice(Ve,xpista,ypista)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);nuP=Ve(16);Psl=Ve(17)*Ve(15);h0=
Ve(12);mu=Ve(19);
theta0=Ve(18);
```

```
global V1;
V1=Ve(23)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;

step=1;
xxpista=xpista(1):step:xpista(end);
yypista=pchip(xpista,ypista,xxpista);
aux=atan(diff(yypista)./diff(xxpista));
phi=[aux(1) aux];
p_phi=polyfit(xxpista,phi,length(xpista)-1);
fprima=diff(yypista)/step;
p_fprima=polyfit(xxpista(:,1:length(fprima)),fprima,length(xpista));
fdosprima=diff(fprima)/step;
p_fdosprima=polyfit(xxpista(:,1:length(fdosprima)),fdosprima,length(xpista));
x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod_ASD);
Optparada=odeset('Events',@myEventparada_ASD);

[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_var_turbohelice(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,de
lta,alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,p
sl,m,g,mu,epsilon),t0rod,x0rod,Optrod);
x0rot=xrod(end,:)';

x_v1=x0rot(end,:)+V1*2;
xrot_v1=(x0rot(end,:):x_v1)';
[sizerot_v1,~]=size(xrot_v1);
xrot(:,1)=xrod(end,1)+xrot_v1;
xrot(:,2)=V1*ones(sizerot_v1,1);
trot=(0:(2/(sizerot_v1-1)):2)';

x0parada=[xrot(end,1);V1];
t0parada=[0 50];
mu=Ve(29);
[tparada,xparada]=ode45(@(t,x)
funcionparada_pend_var(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta,alpha0,the
ta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,m,g,mu,epsilon),t0parada,x0parada,Optp
arada);

ttotal=[trod;trot+trod(end);tparada+trod(end)+trot(end)];
ASD_t=ttotal(end);
xtotal=[xrod(:,1);xrot(:,1);xparada(:,1)];
ASD=xtotal(end);

end
```

- Funciones Calculador ASD con n-1 motores

```
function
[ASD_sinmotor,ASD_t_sinmotor]=calculador_ASD_sinmotor_pendconst_turbofan(Ve)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);Tsl=Ve(16)*Ve(15);h0=Ve(12);mu=V
e(18);
theta0=Ve(17);
global V1
V1=Ve(23)-Vw;
global Vef
Vef=Ve(28)-Vw;


temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
phi=Ve(21);rhosl=densidadmar;
fprima=tan(phi);fdosprima=0;

x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod_ASDsinmotor);
Optacel=odeset('Events',@myEventacel_ASDsinmotor);
Optparada=odeset('Events',@myEventparada_ASD);

[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_const_turbofan(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alp
ha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,psl,m,
g,mu,epsilon),t0rod,x0rod,Optrod);
x0acel=xrod(end,:)';
t0acel=[0 30];

Tsl=(Ve(16)-1)*Ve(15);
[tacel,xacel]=ode45(@(t,x)
funcionrodadura_pend_const_turbofan(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alp
ha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,psl,m,
g,mu,epsilon),t0acel,x0acel,Optacel);
x0rot=xacel(end,:)';

x_v1=x0rot(1)+V1*2;
xrot_v1=(x0rot(1):x_v1)';
```

```
[sizerot_v1,~]=size(xrot_v1);
xrot(:,1)=xrot_v1;
xrot(:,2)=V1*ones(sizerot_v1,1);
trot=(0:(2/(sizerot_v1-1)):2)';

x0parada=[xrot(end,1);V1];
t0parada=[0 50];
mu=Ve(29);
[tparada,xparada]=ode45(@(t,x)
funcionparada_pend_const(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alpha0,theta0,
S,rho,Cd0,deltaCdflap,deltaCdgear,k,m,g,mu,epsilon),t0parada,x0parada,Optparad
a);

ttotal=[trod;tacel+trod(end);trot+tacel(end)+trod(end);tparada+trod(end)+tacel
(end)+trot(end)];
ASD_t_sinmotor=ttotal(end);
xtotal=[xrod(:,1);xacel(:,1);xrot(:,1);xparada(:,1)];
ASD_sinmotor=xtotal(end);

end
```

```
function
[ASD_sinmotor,ASD_t_sinmotor]=calculador_ASD_sinmotor_pendconst_turbohelice(Ve
)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);nuP=Ve(16);Psl=Ve(17)*Ve(15);h0=
Ve(12);mu=Ve(19);
theta0=Ve(18);

global V1
V1=Ve(24)-Vw;
global Vef
Vef=Ve(29)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
phi=Ve(22);
fprima=tan(phi);fdosprima=0;

x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod_ASDsinmotor);
Optacel=odeset('Events',@myEventacel_ASDsinmotor);
```

```
Optparada=odeset('Events',@myEventparada_ASD);

[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_const_turbohelice(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,
alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,psl,m
,g,mu,epsilon),t0rod,x0rod,Optrod);
x0acel=xrod(end,:)';
t0acel=[0 30];

Psl=(Ve(17)-1)*Ve(15);
[tacel,xacel]=ode45(@(t,x)
funcionrodadura_pend_const_turbohelice(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,
alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,psl,m
,g,mu,epsilon),t0acel,x0acel,Optacel);
x0rot=xacel(end,:)';

x_v1=x0rot(1)+V1*2;
xrot_v1=(x0rot(1):x_v1)';
[sizerot_v1,~]=size(xrot_v1);
xrot(:,1)=xrot_v1;
xrot(:,2)=V1*ones(sizerot_v1,1);
trot=(0:(2/(sizerot_v1-1)):2)';

x0parada=[xrot(end,1);V1];
t0parada=[0 50];
mu=Ve(30);
[tparada,xparada]=ode45(@(t,x)
funcionparada_pend_const(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alpha0,theta0,
S,rho,Cd0,deltaCdflap,deltaCdgear,k,m,g,mu,epsilon),t0parada,x0parada,Optparad
a);


ttotal=[trod;tacel+trod(end);trot+tacel(end)+trod(end);tparada+trod(end)+tacel
(end)+trot(end)];
ASD_t_sinmotor=ttotal(end);
xtotal=[xrod(:,1);xacel(:,1);xrot(:,1);xparada(:,1)];
ASD_sinmotor=xtotal(end);

end
```

```
function
[ASD_sinmotor,ASD_t_sinmotor]=calculador_ASD_sinmotor_pendvar_turbofan(Ve,xpis
ta,ypista)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);Tsl=Ve(16)*Ve(15);h0=Ve(12);mu=V
e(18);
theta0=Ve(17);

global V1
```

```
V1=Ve(22)-Vw;
global Vef
Vef=Ve(27)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
rhosl=densidadmar;

step=1;
xxpista=xpista(1):step:xpista(end);
yypista=pchip(xpista,ypista,xxpista);
aux=atan(diff(yypista)./diff(xxpista));
phi=[aux(1) aux];
p_phi=polyfit(xxpista,phi,length(xpista)-1);
p_zg=polyfit(xxpista,yypista,length(xpista)-1);
fprima=diff(yypista)/step;
p_fprima=polyfit(xxpista(:,1:length(fprima)),fprima,length(xpista));
fdosprima=diff(fprima)/step;
p_fdosprima=polyfit(xxpista(:,1:length(fdosprima)),fdosprima,length(xpista));
x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod_ASDsinmotor);
Optacel=odeset('Events',@myEventacel_ASDsinmotor);
Optparada=odeset('Events',@myEventparada_ASD);

[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_var_turbofan(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta
,alpha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,ps
l,m,g,mu,epsilon),t0rod,x0rod,Optrod);
x0acel=xrod(end,:)';
t0acel=[0 30];

Tsl=(Ve(16)-1)*Ve(15);
[tacel,xacel]=ode45(@(t,x)
funcionrodadura_pend_var_turbofan(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta
,alpha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,ps
l,m,g,mu,epsilon),t0acel,x0acel,Optacel);
x0rot=xacel(end,:)';


x_v1=x0rot(1)+V1*2;
xrot_v1=(x0rot(1):x_v1)';
[sizerot_v1,~]=size(xrot_v1);
xrot(:,1)=xrot_v1;
xrot(:,2)=V1*ones(sizerot_v1,1);
trot=(0:(2/(sizerot_v1-1)):2)';

x0parada=[xrot(end,1);V1];
t0parada=[0 50];
mu=Ve(28);
```

```
[tparada,xparada]=ode45(@(t,x)
funcionparada_pend_var(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta,alpha0,the
ta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,m,g,mu,epsilon),t0parada,x0parada,Optp
arada);


ttotal=[trod;tacel+trod(end);trot+tacel(end)+trod(end);tparada+trod(end)+tacel
(end)+trot(end)];
ASD_t_sinmotor=ttotal(end);
xtotal=[xrod(:,1);xacel(:,1);xrot(:,1);xparada(:,1)];
ASD_sinmotor=xtotal(end);

end
```

```
function
[ASD_sinmotor,ASD_t_sinmotor]=calculador_ASD_sinmotor_pendvar_turbohelice(Ve,x
pista,ypista)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);nuP=Ve(16);Psl=Ve(17)*Ve(15);h0=
Ve(12);mu=Ve(19);
theta0=Ve(18);

global V1;
V1=Ve(23)-Vw;
global Vef
Vef=Ve(28)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;

step=1;
xxpista=xpista(1):step:xpista(end);
yypista=pchip(xpista,ypista,xxpista);
aux=atan(diff(yypista)./diff(xxpista));
phi=[aux(1) aux];
p_phi=polyfit(xxpista,phi,length(xpista)-1);
fprima=diff(yypista)/step;
p_fprima=polyfit(xxpista(:,1:length(fprima)),fprima,length(xpista));
fdosprima=diff(fprima)/step;
p_fdosprima=polyfit(xxpista(:,1:length(fdosprima)),fdosprima,length(xpista));
x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod_ASDsinmotor);
Optacel=odeset('Events',@myEventacel_ASDsinmotor);
```

```
Optparada=odeset('Events',@myEventparada_ASD);

[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_var_turbohelice(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,de
lta,alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,p
sl,m,g,mu,epsilon),t0rod,x0rod,Optrod);
x0acel=xrod(end,:)';
t0acel=[0 30];

Psl=(Ve(17)-1)*Ve(15);
[tacel,xacel]=ode45(@(t,x)
funcionrodadura_pend_var_turbohelice(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,de
lta,alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,p
sl,m,g,mu,epsilon),t0acel,x0acel,Optacel);
x0rot=xacel(end,:)';

x_v1=x0rot(1)+V1*2;
xrot_v1=(x0rot(1):x_v1)';
[sizerot_v1,~]=size(xrot_v1);
xrot(:,1)=xrot_v1;
xrot(:,2)=V1*ones(sizerot_v1,1);
trot=(0:(2/(sizerot_v1-1)):2)';

x0parada=[xrot(end,1);V1];
t0parada=[0 50];
mu=Ve(29);
[tparada,xparada]=ode45(@(t,x)
funcionparada_pend_var(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta,alpha0,the
ta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,m,g,mu,epsilon),t0parada,x0parada,Optp
arada);


ttotal=[trod;tacel+trod(end);trot+tacel(end)+trod(end);tparada+trod(end)+tacel
(end)+trot(end)];
ASD_t_sinmotor=ttotal(end);
xtotal=[xrod(:,1);xacel(:,1);xrot(:,1);xparada(:,1)];
ASD_sinmotor=xtotal(end);

end
```

- Funciones calculador TOR, TOD y despegue normal

```
function [Resultados,TOR,Vlof,error]=calculadorpend_const_turbofan(Ve)
[~, ~, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);Tsl=Ve(16)*Ve(15);h0=Ve(12);mu=V
e(18);
```

```
theta0=Ve(17);thetaf=Ve(19);thetatf=Ve(20);

global Vr
Vr=Ve(22)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
phi=Ve(21);rhosl=densidadmar;
fprima=tan(phi);fdosprima=0;

x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod);
Optrot=odeset('Events',@myEventrot,'RelTol',1e-16,'AbsTol',1e-16);
Optvue=odeset('Events',@myEventvue,'RelTol',1e-10,'AbsTol',1e-10);

[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_const_turbofan(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alp
ha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,psl,m,
g,mu,epsilon),t0rod,x0rod,Optrod);
x0rot=xrod(end,:)';
t0rot=[0 200];

[trot,xrot]=ode45(@(t,x)
funcionrotacion_pend_const_turbofan(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alp
ha0,theta0,thetaf,thetatf,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl
,gamma,p,psl,m,g,mu,epsilon),t0rot,x0rot,Optrot);


[sizerot,~]=size(xrot);
ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
for i=1:sizerot
    t=trot(i);
    if t>thetatf
        thetarot(i)=thetaf+phi;
    else
        thetarot(i)=interp1(ftheta,thetat,t)+phi;
    end
end
thetafinalrot=thetarot(end)-phi;
if thetafinalrot>=thetaf
    tiemporestanterotacion=0;
else
    tiemporestanterotacion=thetatf-trot(end);
end


if trot(end)>=200
    error=1;
```

```
else
    error=0;
end

x0vue=[xrot(end,:),xrot(end,1)*fprima,xrot(end,2)*fprima]';
t0vue=[0 50];

[tvue,xvue]=ode45(@(t,x)
funcionvuelo_pend_const_turbofan(t,x,Vw,phi,fprima,a,AR,delta,alpha0,thetaf,th
etafinalrot,tiemporestanterotacion,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,d
eltaT,Tsl,gamma,p,psl,m,g,epsilon),t0vue,x0vue,Optvue);

[sizerod,~]=size(xrod);[sizevue,~]=size(xvue);
ttotal=[trod;trot+trod(end);tvue+trod(end)+trot(end)];

d=xvue(end,1)-xrot(end,1);

TOR=1.15*(xrot(end,1)+d/2);
xtotal=[xrod(:,1);xrot(:,1);xvue(:,1)];
vtotal=[xrod(:,2);xrot(:,2);xvue(:,2)];
zrod=tan(phi)*xrod(:,1);
zrot=tan(phi)*xrot(:,1);
ztotal=[zrod; zrot;xvue(:,3)];
zpuntorod=tan(phi)*ones(sizerod,1);
zpuntorot=tan(phi)*ones(sizerot,1);
zpuntototal=[zpuntorod;zpuntorot;xvue(:,4)];
hrod=h0*ones(sizerod,1);
hrot=h0*ones(sizerot,1);
hvue=h0+(xvue(:,3)-xvue(:,1)*tan(phi))*sqrt(1+fprima^2);
htotal=[hrod;hrot;hvue];
thetarod=(theta0+phi)*ones(sizerod,1);
phivue=phi*ones(sizevue,1);
if thetafinalrot>=thetaf
    thetavue=(thetaf+phivue)';
else
    fthetavue=linspace(0,tiemporestanterotacion);
    thetatvue=thetafinalrot+((thetaf-
thetafinalrot)/tiemporestanterotacion)*fthetavue;
    for ii=1:sizevue
        tt=tvue(ii);
        if tt>tiemporestanterotacion
            thetavue(ii)=thetaf+phi;
        else
            thetavue(ii)=interp1(fthetavue,thetatvue,tt)+phi;
        end
    end
end

thetatotal=180/pi*[thetarod;thetarot';thetavue'];
phirod=phi*ones(sizerod,1);
phirot=phi*ones(sizerot,1);
phitotal=180/pi*[phirod;phirot;phivue];
gammaArod=phirod;
gammaArot=phirot;
Vax=xvue(:,2)+Vw.*cos(phivue);
```

```
Vaz=xvue(:,4)+Vw.*sin(phivue);
gammaAvue=atan(Vaz./Vax);
gammaAtotal=180/pi*[gammaArod;gammaArot;gammaAvue];
alpharod=thetarod-phirod;
alpharot=thetarot'-phirot;
alphavue=thetavue'-gammaAvue;
alphatotal=180/pi*[alpharod;alpharot;alphavue];
Vlof=xrot(end,2)/cos(phirot(end));


vaire=Vax./cos(gammaAvue);
vtotalaire=[xrod(:,2)./cos(phirod)+Vw;xrot(:,2)./cos(phirot)+Vw;vaire];
vgroundvuelo=sqrt((Vax-Vw.*cos(phivue)).^2+(Vaz-Vw.*sin(phivue)).^2);
vground=[xrod(:,2)./cos(phirod);xrot(:,2)./cos(phirot);vgroundvuelo];

[Nfilas,~]=size(xtotal);
Resultados(1,:)=["Tiempo","Posición horizontal","Velocidad
horizontal","Velocidad Tierra","Velocidad Aire","Posición vertical","Velocidad
vertical","Altura","Theta","Ángulo pista","Gamma","Alpha"];
Resultados(2:Nfilas+1,:)=[ttotal,xtotal,vtotal,vground,vtotalaire,ztotal,zpunt
ototal,htotal,thetatotal,phitotal,gammaAtotal,alphatotal];


end
```

```
function [Resultados,TOR,Vlof,error]=calculadorpend_const_turbohelice(Ve)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);
delta=Ve(5);Cd0=Ve(6);deltaCdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);nuP=Ve(16);Psl=Ve(17)*Ve(15);h0=
Ve(12);mu=Ve(19);
theta0=Ve(18);thetaf=Ve(20);thetatf=Ve(21);

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
phi=Ve(22);
fprima=tan(phi);fdosprima=0;

global Vr
Vr=(Ve(23)-Vw);


x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod);
Optrot=odeset('Events',@myEventrot,'RelTol',1e-16,'AbsTol',1e-16);
```

```
Optvue=odeset('Events',@myEventvue,'RelTol',1e-10,'AbsTol',1e-10);

[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_const_turbohelice(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,
alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,psl,m
,g,mu,epsilon),t0rod,x0rod,Optrod);
x0rot=xrod(end,:)';
t0rot=[0 200];


[trot,xrot]=ode45(@(t,x)
funcionrotacion_pend_const_turbohelice(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,
alpha0,theta0,thetaf,thetatf,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Ps
l,gamma,p,psl,m,g,mu,epsilon),t0rot,x0rot,Optrot);


[sizerot,~]=size(xrot);
ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
for i=1:sizerot
    t=trot(i);
    if t>thetatf
        thetarot(i)=thetaf+phi;
    else
        thetarot(i)=interp1(ftheta,thetat,t)+phi;
    end
end
thetafinalrot=thetarot(end)-phi;
if thetafinalrot>=thetaf
    tiemporestanterotacion=0;
else
    tiemporestanterotacion=thetatf-trot(end);
end



if trot(end)>=200
    error=1;
else
    error=0;
end


x0vue=[xrot(end,:),xrot(end,1)*fprima,xrot(end,2)*fprima]';
t0vue=[0 50];

[tvue,xvue]=ode45(@(t,x)
funcionvuelo_pend_const_turbohelice(t,x,Vw,phi,fprima,a,AR,delta,alpha0,thetaf
,thetafinalrot,tiemporestanterotacion,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,
deltaT,Psl,gamma,p,psl,m,g,epsilon),t0vue,x0vue,Optvue);


[sizerod,~]=size(xrod);[sizerot,~]=size(xrot);[sizevue,~]=size(xvue);
ttotal=[trod;trot+trod(end);tvue+trod(end)+trot(end)];
xtotal=[xrod(:,1);xrot(:,1);xvue(:,1)];
```

```
d=xvue(end,1)-xrot(end,1);

TOR=1.15*(xrot(end,1)+d/2);
vtotal=[xrod(:,2);xrot(:,2);xvue(:,2)];
zrod=tan(phi)*xrod(:,1);
zrot=tan(phi)*xrot(:,1);
ztotal=[zrod; zrot;xvue(:,3)];
zpuntorod=tan(phi)*ones(sizerod,1);
zpuntorot=tan(phi)*ones(sizerot,1);
zpuntototal=[zpuntorod;zpuntorot;xvue(:,4)];
hrod=h0*ones(sizerod,1);
hrot=h0*ones(sizerot,1);
hvue=h0+(xvue(:,3)-xvue(:,1)*tan(phi))*sqrt(1+fprima^2);
htotal=[hrod;hrot;hvue];
thetarod=(theta0+phi)*ones(sizerod,1);
phivue=phi*ones(sizevue,1);
if thetafinalrot>=thetaf
    thetavue=(thetaf+phivue)';
else
    fthetavue=linspace(0,tiemporestanterotacion);
    thetatvue=thetafinalrot+((thetaf-
thetafinalrot)/tiemporestanterotacion)*fthetavue;
    for ii=1:sizevue
        tt=tvue(ii);
        if tt>tiemporestanterotacion
            thetavue(ii)=thetaf+phi;
        else
            thetavue(ii)=interp1(fthetavue,thetatvue,tt)+phi;
        end
    end
end


thetatotal=180/pi*[thetarod;thetarot';thetavue'];
phirod=phi*ones(sizerod,1);
phirot=phi*ones(sizerot,1);
phitotal=180/pi*[phirod;phirot;phivue];
gammaArod=phirod;
gammaArot=phirot;
Vax=xvue(:,2)+Vw.*cos(phivue);
Vaz=xvue(:,4)+Vw.*sin(phivue);
gammaAvue=atan(Vaz./Vax);
gammaAtotal=180/pi*[gammaArod;gammaArot;gammaAvue];
alpharod=thetarod-phirod;
alpharot=thetarot'-phirot;
alphavue=thetavue'-gammaAvue;
alphatotal=180/pi*[alpharod;alpharot;alphavue];


vaire=Vax./cos(gammaAvue);
vtotalaire=[xrod(:,2)./cos(phirod)+Vw;xrot(:,2)./cos(phirot)+Vw;vaire];
vgroundvuelo=sqrt((Vax-Vw.*cos(phivue)).^2+(Vaz-Vw.*sin(phivue)).^2);
vground=[xrod(:,2)./cos(phirod);xrot(:,2)./cos(phirot);vgroundvuelo];
```

```
Vlof=xrot(end,2)/cos(phirot(end));

[Nfilas,~]=size(xtotal);
Resultados(1,:)=["Tiempo","Posición horizontal","Velocidad
horizontal","Velocidad Tierra","Velocidad Aire","Posición vertical","Velocidad
vertical","Altura","Theta","Ángulo pista","Gamma","Alpha"];
Resultados(2:Nfilas+1,:)=[ttotal,xtotal,vtotal,vground,vtotalaire,ztotal,zpunt
ototal,htotal,thetatotal,phitotal,gammaAtotal,alphatotal];



end
```

```
function
[Resultados,TOR,Vlof,error]=calculadorpend_var_turbofan(Ve,xpista,ypista)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);Tsl=Ve(16)*Ve(15);h0=Ve(12);mu=V
e(18);
theta0=Ve(17);thetaf=Ve(19);thetatf=Ve(20);

global Vr
Vr=Ve(21)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
rhosl=densidadmar;

step=1;
xxpista=xpista(1):step:xpista(end);
yypista=pchip(xpista,ypista,xxpista);


aux=atan(diff(yypista)./diff(xxpista));
phi=[aux(1) aux];
p_phi=polyfit(xxpista,phi,length(xpista)-1);
p_zg=polyfit(xxpista,yypista,length(xpista)-1);
fprima=diff(yypista)/step;
p_fprima=polyfit(xxpista(:,1:length(fprima)),fprima,length(xpista));
fdosprima=diff(fprima)/step;
p_fdosprima=polyfit(xxpista(:,1:length(fdosprima)),fdosprima,length(xpista));
x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod);
```

```
Optrot=odeset('Events',@myEventrot,'RelTol',1e-16,'AbsTol',1e-16);
Optvue=odeset('Events',@myEventvue,'RelTol',1e-10,'AbsTol',1e-10);

[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_var_turbofan(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta
,alpha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,ps
l,m,g,mu,epsilon),t0rod,x0rod,Optrod);
x0rot=xrod(end,:)';
t0rot=[0 200];

[trot,xrot]=ode45(@(t,x)
funcionrotacion_pend_var_turbofan(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta
,alpha0,theta0,thetaf,thetatf,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT
,Tsl,gamma,p,psl,m,g,mu,epsilon),t0rot,x0rot,Optrot);

phirod=polyval(p_phi,xrod(:,1));
phirot=polyval(p_phi,xrot(:,1));
[sizetrot,~]=size(trot);
ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
for i=1:sizetrot
    t=trot(i);
    if t>thetatf
        thetarot(i)=thetaf+phirot(i);
    else
        thetarot(i)=interp1(ftheta,thetat,t)+phirot(i);
    end
end
thetafinalrot=thetarot(end)-phirot(end);
if thetafinalrot>=thetaf
    tiemporestanterotacion=0;
else
    tiemporestanterotacion=thetatf-trot(end);
end


if trot(end)>=200
    error=1;
else
    error=0;
end


z0vue=polyval(p_zg,xrot(end,1));
f0prima_vue=polyval(p_fprima,xrot(end,1));
z0puntovue=xrot(end,2)*f0prima_vue;
x0vue=[xrot(end,:),z0vue,z0puntovue]';
t0vue=[0 50];

[tvue,xvue]=ode45(@(t,x)
funcionvuelo_pend_var_turbofan(t,x,Vw,p_phi,p_zg,p_fprima,p_fdosprima,a,AR,del
ta,alpha0,thetaf,thetafinalrot,tiemporestanterotacion,S,rho,rhosl,Cd0,deltaCdf
lap,deltaCdgear,k,deltaT,Tsl,gamma,p,psl,m,g,epsilon),t0vue,x0vue,Optvue);
```

```
[sizerod,~]=size(xrod);[sizerot,~]=size(xrot);[sizevue,~]=size(xvue);
ttotal=[trod;trot+trod(end);tvue+trod(end)+trot(end)];
xtotal=[xrod(:,1);xrot(:,1);xvue(:,1)];
d=xvue(end,1)-xrot(end,1);
TOR=1.15*(xrot(end,1)+d/2);
vtotal=[xrod(:,2);xrot(:,2);xvue(:,2)];
zrod=polyval(p_zg,xrod(:,1));
zrot=polyval(p_zg,xrot(:,1));
ztotal=[zrod; zrot;xvue(:,3)];
zpuntorod=polyval(p_fprima,xrod(:,1)).*xrod(:,2);
zpuntorot=polyval(p_fprima,xrot(:,1)).*xrot(:,2);
zpuntototal=[zpuntorod;zpuntorot;xvue(:,4)];
hrod=h0*ones(sizerod,1);
hrot=h0*ones(sizerot,1);
hvue=h0+(xvue(:,3)-
polyval(p_zg,xvue(:,1))).*sqrt(1+(polyval(p_fprima,xvue(:,1))).^2);
htotal=[hrod;hrot;hvue];
thetarod=theta0+phirod;

phivue=polyval(p_phi,xvue(:,1));
if thetafinalrot>=thetaf
    thetavue=(thetaf+phivue)';
else
    fthetavue=linspace(0,tiemporestanterotacion);
    thetatvue=thetafinalrot+((thetaf-
thetafinalrot)/tiemporestanterotacion)*fthetavue;
    for ii=1:sizevue
        tt=tvue(ii);
        if tt>tiemporestanterotacion
            thetavue(ii)=thetaf+phivue(ii);
        else
            thetavue(ii)=interp1(fthetavue,thetatvue,tt)+phivue(ii);
        end
    end
end


thetatotal=180/pi*[thetarod;thetarot';thetavue'];
phitotal=180/pi*[phirod;phirot;phivue];
gammaArod=phirod;
gammaArot=phirot;
Vax=xvue(:,2)+Vw.*cos(phivue);
Vaz=xvue(:,4)+Vw.*sin(phivue);
gammaAvue=atan(Vaz./Vax);
gammaAtotal=180/pi*[gammaArod;gammaArot;gammaAvue];
alpharod=thetarod-phirod;
alpharot=thetarot'-phirot;
alphavue=thetavue'-gammaAvue;
alphatotal=180/pi*[alpharod;alpharot;alphavue];

vaire=Vax./cos(gammaAvue);
vtotalaire=[xrod(:,2)./cos(phirod)+Vw;xrot(:,2)./cos(phirot)+Vw;vaire];
vgroundvuelo=sqrt((Vax-Vw.*cos(phivue)).^2+(Vaz-Vw.*sin(phivue)).^2);
vground=[xrod(:,2)./cos(phirod);xrot(:,2)./cos(phirot);vgroundvuelo];
```

```
Vlof=xrot(end,2)/cos(phirot(end));


[Nfilas,~]=size(xtotal);
Resultados(1,:)=["Tiempo","Posición horizontal","Velocidad
horizontal","Velocidad Tierra","Velocidad Aire","Posición vertical","Velocidad
vertical","Altura","Theta","Ángulo pista","Gamma","Alpha"];
Resultados(2:Nfilas+1,:)=[ttotal,xtotal,vtotal,vground,vtotalaire,ztotal,zpunt
ototal,htotal,thetatotal,phitotal,gammaAtotal,alphatotal];



end
```

```
function
[Resultados,TOR,Vlof,error]=calculadorpend_var_turbohelice(Ve,xpista,ypista)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);nuP=Ve(16);Psl=Ve(17)*Ve(15);h0=
Ve(12);mu=Ve(19);
theta0=Ve(18);thetaf=Ve(20);thetatf=Ve(21);
global Vr
Vr=Ve(22)-Vw;


temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;

step=1;
xxpista=xpista(1):step:xpista(end);
yypista=pchip(xpista,ypista,xxpista);
L=length(yypista);


aux=atan(diff(yypista)./diff(xxpista));
phi=[aux(1) aux];


p_phi=polyfit(xxpista,phi,length(xpista)-1);
p_zg=polyfit(xxpista,yypista,length(xpista)-1);
for i=2:L-1
    fprima(i)=(yypista(i+1)-yypista(i-1))/(2*step);
    fdosprima(i)=(yypista(i+1)-2*yypista(i)+yypista(i-1))/(step^2);
end
p_fprima=polyfit(xxpista(:,1:length(fprima)),fprima,length(xpista));
p_fdosprima=polyfit(xxpista(:,1:length(fdosprima)),fdosprima,length(xpista));
```

```
x0rod=[0; 0.01];
t0rod=[0 50];

Optrod=odeset('Events',@myEventrod);
Optrot=odeset('Events',@myEventrot,'RelTol',1e-16,'AbsTol',1e-16);
Optvue=odeset('Events',@myEventvue,'RelTol',1e-10,'AbsTol',1e-10);

[trod,xrod]=ode45(@(t,x)
funcionrodadura_pend_var_turbohelice(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,de
lta,alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,p
sl,m,g,mu,epsilon),t0rod,x0rod,Optrod);
x0rot=xrod(end,:)';
t0rot=[0 200];

[trot,xrot]=ode45(@(t,x)
funcionrotacion_pend_var_turbohelice(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,de
lta,alpha0,theta0,thetaf,thetatf,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,delta
T,Psl,gamma,p,psl,m,g,mu,epsilon),t0rot,x0rot,Optrot);


phirod=polyval(p_phi,xrod(:,1));
phirot=polyval(p_phi,xrot(:,1));
[sizetrot,~]=size(trot);
ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
for i=1:sizetrot
    t=trot(i);
    if t>thetatf
        thetarot(i)=thetaf+phirot(i);
    else
        thetarot(i)=interp1(ftheta,thetat,t)+phirot(i);
    end
end
thetafinalrot=thetarot(end)-phirot(end);
if thetafinalrot>=thetaf
    tiemporestanterotacion=0;
else
    tiemporestanterotacion=thetatf-trot(end);
end




if trot(end)>=200
    error=1;
else
    error=0;
end


z0vue=polyval(p_zg,xrot(end,1));
f0prima_vue=polyval(p_fprima,xrot(end,1));
```

```
z0puntovue=xrot(end,2)*f0prima_vue;
x0vue=[xrot(end,:),z0vue,z0puntovue]';
t0vue=[0 50];

[tvue,xvue]=ode45(@(t,x)
funcionvuelo_pend_var_turbohelice(t,x,Vw,p_phi,p_zg,p_fprima,p_fdosprima,a,AR,
delta,alpha0,thetaf,thetafinalrot,tiemporestanterotacion,S,rho,Cd0,deltaCdflap
,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,psl,m,g,epsilon),t0vue,x0vue,Optvue);


[sizerod,~]=size(xrod);[sizerot,~]=size(xrot);[sizevue,~]=size(xvue);
ttotal=[trod;trot+trod(end);tvue+trod(end)+trot(end)];
xtotal=[xrod(:,1);xrot(:,1);xvue(:,1)];
d=xvue(end,1)-xrot(end,1);
TOR=1.15*(xrot(end,1)+d/2);
vtotal=[xrod(:,2);xrot(:,2);xvue(:,2)];
zrod=polyval(p_zg,xrod(:,1));
zrot=polyval(p_zg,xrot(:,1));
ztotal=[zrod; zrot;xvue(:,3)];
zpuntorod=polyval(p_fprima,xrod(:,1)).*xrod(:,2);
zpuntorot=polyval(p_fprima,xrot(:,1)).*xrot(:,2);
zpuntototal=[zpuntorod;zpuntorot;xvue(:,4)];
hrod=h0*ones(sizerod,1);
hrot=h0*ones(sizerot,1);
hvue=h0+(xvue(:,3)-
polyval(p_zg,xvue(:,1))).*sqrt(1+(polyval(p_fprima,xvue(:,1)).*xvue(:,2)).^2);
htotal=[hrod;hrot;hvue];
thetarod=theta0+phirod;

phivue=polyval(p_phi,xvue(:,1));
if thetafinalrot>=thetaf
    thetavue=(thetaf+phivue)';
else
    fthetavue=linspace(0,tiemporestanterotacion);
    thetatvue=thetafinalrot+((thetaf-
thetafinalrot)/tiemporestanterotacion)*fthetavue;
    for ii=1:sizevue
        tt=tvue(ii);
        if tt>tiemporestanterotacion
            thetavue(ii)=thetaf+phivue(ii);
        else
            thetavue(ii)=interp1(fthetavue,thetatvue,tt)+phivue(ii);
        end
    end
end



thetatotal=180/pi*[thetarod;thetarot';thetavue'];
phirod=polyval(p_phi,xrod(:,1));
phitotal=180/pi*[phirod;phirot;phivue];
gammaArod=phirod;
gammaArot=phirot;
Vax=xvue(:,2)+Vw.*cos(phivue);
```

```
Vaz=xvue(:,4)+Vw.*sin(phivue);
gammaAvue=atan(Vaz./Vax);
gammaAtotal=180/pi*[gammaArod;gammaArot;gammaAvue];
alpharod=thetarod-phirod;
alpharot=thetarot'-phirot;
alphavue=thetavue'-gammaAvue;
alphatotal=180/pi*[alpharod;alpharot;alphavue];



vaire=Vax./cos(gammaAvue);
vtotalaire=[xrod(:,2)./cos(phirod)+Vw;xrot(:,2)./cos(phirot)+Vw;vaire];
vgroundvuelo=sqrt((Vax-Vw.*cos(phivue)).^2+(Vaz-Vw.*sin(phivue)).^2);
vground=[xrod(:,2)./cos(phirod);xrot(:,2)./cos(phirot);vgroundvuelo];


Vlof=xrot(end,2)/cos(phirot(end));


[Nfilas,~]=size(xtotal);
Resultados(1,:)=["Tiempo","Posición horizontal","Velocidad
horizontal","Velocidad Tierra","Velocidad Aire","Posición vertical","Velocidad
vertical","Altura","Theta","Ángulo pista","Gamma","Alpha"];
Resultados(2:Nfilas+1,:)=[ttotal,xtotal,vtotal,vground,vtotalaire,ztotal,zpunt
ototal,htotal,thetatotal,phitotal,gammaAtotal,alphatotal];

end
```

- Funciones calculador TOD y TOR con n-1 motores

```
function
[TOD_sinmotor,TOR_sinmotor,Vlof_sinmotor,error_sinmotor,V2_sinmotor,V1]=calcul
adorpend_TOD_sinmotor_const_turbofan(Ve)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);



m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);Tsl=Ve(16)*Ve(15);h0=Ve(12);mu=V
e(18);
theta0=Ve(17);thetaf=Ve(19);thetatf=Ve(20);

global Vr
Vr=Ve(22)-Vw;
global Vef
Vef=Ve(28)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
```

```
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
phi=Ve(21);rhosl=densidadmar;
fprima=tan(phi);fdosprima=0;

x0rod1=[0; 0.01];
t0rod1=[0 500];

Optrod1=odeset('Events',@myEventrod1);
Optrod2=odeset('Events',@myEventrod2,'RelTol',1e-12,'AbsTol',1e-12);
Optrot=odeset('Events',@myEventrot,'RelTol',1e-16,'AbsTol',1e-16);
Optvue=odeset('Events',@myEventvue,'RelTol',1e-10,'AbsTol',1e-10);

[trod1,xrod1]=ode45(@(t,x)
funcionrodadura_pend_const_turbofan(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alp
ha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,psl,m,
g,mu,epsilon),t0rod1,x0rod1,Optrod1);
x0rod2=xrod1(end,:)';
t0rod2=[0 200];

Tsl=1.2*(Ve(16)-1)*Ve(15);
[trod2,xrod2]=ode45(@(t,x)
funcionrodadura_pend_const_turbofan(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alp
ha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,psl,m,
g,mu,epsilon),t0rod2,x0rod2,Optrod2);
Trec=Ve(30);
indicev1=find(abs(trod2-Trec)<0.2);
V1=xrod2(indicev1(1),2)/cos(phi)+Vw;
x0rot=xrod2(end,:)';
t0rot=[0 200];

[trot,xrot]=ode45(@(t,x)
funcionrotacion_pend_const_turbofan(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alp
ha0,theta0,thetaf,thetatf,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl
,gamma,p,psl,m,g,mu,epsilon),t0rot,x0rot,Optrot);

[sizerot,~]=size(xrot);
ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
for i=1:sizerot
    t=trot(i);
    if t>thetatf
        thetarot(i)=thetaf+phi;
    else
        thetarot(i)=interp1(ftheta,thetat,t)+phi;
    end
end
thetafinalrot=thetarot(end)-phi;
if thetafinalrot>=thetaf
    tiemporestanterotacion=0;
else
    tiemporestanterotacion=thetatf-trot(end);
end
```

```
if trot(end)>=200
    error_sinmotor=1;
else
    error_sinmotor=0;
end

x0vue=[xrot(end,:),xrot(end,1)*fprima,xrot(end,2)*fprima]';
t0vue=[0 50];

[tvue,xvue]=ode45(@(t,x)
funcionvuelo_pend_const_turbofan(t,x,Vw,phi,fprima,a,AR,delta,alpha0,thetaf,th
etafinalrot,tiemporestanterotacion,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,d
eltaT,Tsl,gamma,p,psl,m,g,epsilon),t0vue,x0vue,Optvue);


[sizevue,~]=size(xvue);
ttotal=[trod1;trod2+trod1(end);trot+trod2(end)+trod1(end);tvue+trot(end)+trod2
(end)+trod1(end)];
xtotal=[xrod1(:,1);xrod2(:,1);xrot(:,1);xvue(:,1)];
TOD_sinmotor=xtotal(end);
d=xvue(end,1)-xrot(end,1);
TOR_sinmotor=xrot(end,1)+d/2;
Vlof_sinmotor=xrot(end,2)/cos(phi);
phivue=phi*ones(sizevue,1);
Vax=xvue(:,2)+Vw.*cos(phivue);
Vaz=xvue(:,4)+Vw.*sin(phivue);
gammaAvue=atan(Vaz./Vax);
V2_sinmotor=Vax(end)/cos(gammaAvue(end));

end
```

```
function
[TOD_sinmotor,TOR_sinmotor,Vlof_sinmotor,error_sinmotor,V2_sinmotor,V1]=calcul
adorpend_TOD_sinmotor_const_turbohelice(Ve)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);nuP=Ve(16);Psl=Ve(17)*Ve(15);h0=
Ve(12);mu=Ve(19);
theta0=Ve(18);thetaf=Ve(20);thetatf=Ve(21);
global Vr
Vr=Ve(23)-Vw;
global Vef
Vef=Ve(29)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
```

```
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
phi=Ve(22);
fprima=tan(phi);fdosprima=0;

x0rod1=[0; 0.01];
t0rod1=[0 200];

Optrod1=odeset('Events',@myEventrod1);
Optrod2=odeset('Events',@myEventrod2,'RelTol',1e-12,'AbsTol',1e-12);
Optrot=odeset('Events',@myEventrot,'RelTol',1e-16,'AbsTol',1e-16);
Optvue=odeset('Events',@myEventvue,'RelTol',1e-10,'AbsTol',1e-10);

[trod1,xrod1]=ode45(@(t,x)
funcionrodadura_pend_const_turbohelice(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,
alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,psl,m
,g,mu,epsilon),t0rod1,x0rod1,Optrod1);
x0rod2=xrod1(end,:)';
t0rod2=[0 200];

Psl=1.2*(Ve(17)-1)*Ve(15);
[trod2,xrod2]=ode45(@(t,x)
funcionrodadura_pend_const_turbohelice(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,
alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,psl,m
,g,mu,epsilon),t0rod2,x0rod2,Optrod2);
Trec=Ve(31);
indicev1=find(abs(trod2-Trec)<0.2);
V1=xrod2(indicev1(1),2)/cos(phi)+Vw;
x0rot=xrod2(end,:)';
t0rot=[0 200];

[trot,xrot]=ode45(@(t,x)
funcionrotacion_pend_const_turbohelice(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,
alpha0,theta0,thetaf,thetatf,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Ps
l,gamma,p,psl,m,g,mu,epsilon),t0rot,x0rot,Optrot);

[sizerot,~]=size(xrot);
ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
for i=1:sizerot
    t=trot(i);
    if t>thetatf
        thetarot(i)=thetaf+phi;
    else
        thetarot(i)=interp1(ftheta,thetat,t)+phi;
    end
end
thetafinalrot=thetarot(end)-phi;
if thetafinalrot>=thetaf
    tiemporestanterotacion=0;
else
    tiemporestanterotacion=thetatf-trot(end);
end
```

```
if trot(end)>=200
    error_sinmotor=1;
else
    error_sinmotor=0;
end


x0vue=[xrot(end,:),xrot(end,1)*fprima,xrot(end,2)*fprima]';
t0vue=[0 200];

[tvue,xvue]=ode45(@(t,x)
funcionvuelo_pend_const_turbohelice(t,x,Vw,phi,fprima,a,AR,delta,alpha0,thetaf
,thetafinalrot,tiemporestanterotacion,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,
deltaT,Psl,gamma,p,psl,m,g,epsilon),t0vue,x0vue,Optvue);

[sizevue,~]=size(xvue);
ttotal=[trod1;trod2+trod1(end);trot+trod2(end)+trod1(end);tvue+trot(end)+trod2
(end)+trod1(end)];
xtotal=[xrod1(:,1);xrod2(:,1);xrot(:,1);xvue(:,1)];
TOD_sinmotor=xtotal(end);
d=xvue(end,1)-xrot(end,1);
TOR_sinmotor=xrot(end,1)+d/2;
Vlof_sinmotor=xrot(end,2)/(cos(phi));
phivue=phi*ones(sizevue,1);
Vax=xvue(:,2)+Vw.*cos(phivue);
Vaz=xvue(:,4)+Vw.*sin(phivue);
gammaAvue=atan(Vaz./Vax);
V2_sinmotor=Vax(end)/cos(gammaAvue(end));

end
```

```
function
[TOD_sinmotor,TOR_sinmotor,Vlof_sinmotor,error_sinmotor,V2_sinmotor,V1]=calcul
adorpend_TOD_sinmotor_var_turbofan(Ve,xpista,ypista)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);


m=Ve(1);
epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);deltaCdflap=V
e(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);Tsl=Ve(16)*Ve(15);h0=Ve(12);mu=V
e(18);
theta0=Ve(17);thetaf=Ve(19);thetatf=Ve(20);

global Vr
Vr=Ve(21)-Vw;
global Vef
Vef=Ve(27)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
```

```
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;
rhosl=densidadmar;

step=1;
xxpista=xpista(1):step:xpista(end);
yypista=pchip(xpista,ypista,xxpista);
aux=atan(diff(yypista)./diff(xxpista));
phi=[aux(1) aux];
p_phi=polyfit(xxpista,phi,length(xpista)-1);
p_zg=polyfit(xxpista,yypista,length(xpista)-1);
fprima=diff(yypista)/step;
p_fprima=polyfit(xxpista(:,1:length(fprima)),fprima,length(xpista));
fdosprima=diff(fprima)/step;
p_fdosprima=polyfit(xxpista(:,1:length(fdosprima)),fdosprima,length(xpista));
x0rod1=[0; 0.01];
t0rod1=[0 500];

Optrod1=odeset('Events',@myEventrod1);
Optrod2=odeset('Events',@myEventrod2,'RelTol',1e-12,'AbsTol',1e-12);
Optrot=odeset('Events',@myEventrot,'RelTol',1e-16,'AbsTol',1e-16);
Optvue=odeset('Events',@myEventvue,'RelTol',1e-10,'AbsTol',1e-10);

[trod1,xrod1]=ode45(@(t,x)
funcionrodadura_pend_var_turbofan(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta
,alpha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,ps
l,m,g,mu,epsilon),t0rod1,x0rod1,Optrod1);
x0rod2=xrod1(end,:)';
t0rod2=[0 200];

Tsl=1.2*(Ve(16)-1)*Ve(15);
[trod2,xrod2]=ode45(@(t,x)
funcionrodadura_pend_var_turbofan(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta
,alpha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,ps
l,m,g,mu,epsilon),t0rod2,x0rod2,Optrod2);
Trec=Ve(29);
indicev1=find(abs(trod2-Trec)<0.2);
phirodv1=polyval(p_phi,xrod2(indicev1(1),1));
V1=xrod2(indicev1(1),2)/cos(phirodv1)+Vw;
x0rot=xrod2(end,:)';
t0rot=[0 200];

[trot,xrot]=ode45(@(t,x)
funcionrotacion_pend_var_turbofan(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta
,alpha0,theta0,thetaf,thetatf,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT
,Tsl,gamma,p,psl,m,g,mu,epsilon),t0rot,x0rot,Optrot);

phirot=polyval(p_phi,xrot(:,1));
[sizetrot,~]=size(trot);
ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
for i=1:sizetrot
    t=trot(i);
```

```
    if t>thetatf
        thetarot(i)=thetaf+phirot(i);
    else
        thetarot(i)=interp1(ftheta,thetat,t)+phirot(i);
    end
end
thetafinalrot=thetarot(end)-phirot(end);
if thetafinalrot>=thetaf
    tiemporestanterotacion=0;
else
    tiemporestanterotacion=thetatf-trot(end);
end


if trot(end)>=200
    error_sinmotor=1;
else
    error_sinmotor=0;
end


z0vue=polyval(p_zg,xrot(end,1));
f0prima_vue=polyval(p_fprima,xrot(end,1));
z0puntovue=xrot(end,2)*f0prima_vue;
x0vue=[xrot(end,:),z0vue,z0puntovue]';
t0vue=[0 50];

[tvue,xvue]=ode45(@(t,x)
funcionvuelo_pend_var_turbofan(t,x,Vw,p_phi,p_zg,p_fprima,p_fdosprima,a,AR,del
ta,alpha0,thetaf,thetafinalrot,tiemporestanterotacion,S,rho,rhosl,Cd0,deltaCdf
lap,deltaCdgear,k,deltaT,Tsl,gamma,p,psl,m,g,epsilon),t0vue,x0vue,Optvue);


ttotal=[trod1;trod2+trod1(end);trot+trod2(end)+trod1(end);tvue+trot(end)+trod2
(end)+trod1(end)];
xtotal=[xrod1(:,1);xrod2(:,1);xrot(:,1);xvue(:,1)];
TOD_sinmotor=xtotal(end);
d=xvue(end,1)-xrot(end,1);
TOR_sinmotor=xrot(end,1)+d/2;
phirot=polyval(p_phi,xrot(end,1));
Vlof_sinmotor=xrot(end,2)/cos(phirot);
phivue=polyval(p_phi,xvue(:,1));
Vax=xvue(:,2)+Vw.*cos(phivue);
Vaz=xvue(:,4)+Vw.*sin(phivue);
gammaAvue=atan(Vaz./Vax);
V2_sinmotor=Vax(end)/cos(gammaAvue(end));

end
```

```
function
[TOD_sinmotor,TOR_sinmotor,Vlof_sinmotor,error_sinmotor,V2_sinmotor,V1]=calcul
adorpend_TOD_sinmotor_var_turbohelice(Ve,xpista,ypista)
[temperaturamar, amar, presionmar, densidadmar] = atmosisa(0);

m=Ve(1);epsilon=Ve(2);S=Ve(3);AR=((Ve(4)^2/S)/0.6);delta=Ve(5);Cd0=Ve(6);delta
Cdflap=Ve(7);deltaCdgear=Ve(8);
k=Ve(9);alpha0=Ve(10);Vw=Ve(13);deltaT=Ve(14);nuP=Ve(16);Psl=Ve(17)*Ve(15);h0=
Ve(12);mu=Ve(19);
theta0=Ve(18);thetaf=Ve(20);thetatf=Ve(21);

global Vr
Vr=Ve(22)-Vw;
global Vef
Vef=Ve(28)-Vw;

temperatura=Ve(11)+273.15;
[~,~, presion,~] = atmosisa(h0);
densidad=presion/(286.9*temperatura);
a=sqrt(1.4*287.05*temperatura);

rho=densidad;gamma=1.4;p=presion;psl=presionmar;g=9.81;

step=1;
xxpista=xpista(1):step:xpista(end);
yypista=pchip(xpista,ypista,xxpista);
L=length(yypista);

aux=atan(diff(yypista)./diff(xxpista));
phi=[aux(1) aux];


p_phi=polyfit(xxpista,phi,length(xpista)-1);
p_zg=polyfit(xxpista,yypista,length(xpista)-1);
for i=2:L-1
    fprima(i)=(yypista(i+1)-yypista(i-1))/(2*step);
    fdosprima(i)=(yypista(i+1)-2*yypista(i)+yypista(i-1))/(step^2);
end
p_fprima=polyfit(xxpista(:,1:length(fprima)),fprima,length(xpista));
p_fdosprima=polyfit(xxpista(:,1:length(fdosprima)),fdosprima,length(xpista));



x0rod1=[0; 0.01];
t0rod1=[0 500];

Optrod1=odeset('Events',@myEventrod1);
Optrod2=odeset('Events',@myEventrod2,'RelTol',1e-12,'AbsTol',1e-12);
Optrot=odeset('Events',@myEventrot,'RelTol',1e-16,'AbsTol',1e-16);
Optvue=odeset('Events',@myEventvue,'RelTol',1e-10,'AbsTol',1e-10);

[trod1,xrod1]=ode45(@(t,x)
funcionrodadura_pend_var_turbohelice(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,de
```

```
lta,alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,p
sl,m,g,mu,epsilon),t0rod1,x0rod1,Optrod1);
x0rod2=xrod1(end,:)';
t0rod2=[0 200];

Psl=1.2*(Ve(17)-1)*Ve(15);
[trod2,xrod2]=ode45(@(t,x)
funcionrodadura_pend_var_turbohelice(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,de
lta,alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,p
sl,m,g,mu,epsilon),t0rod2,x0rod2,Optrod2);
Trec=Ve(30);
indicev1=find(abs(trod2-Trec)<0.2);
phirodv1=polyval(p_phi,xrod2(indicev1(1),1));
V1=xrod2(indicev1(1),2)/cos(phirodv1)+Vw;
x0rot=xrod2(end,:)';
t0rot=[0 200];

[trot,xrot]=ode45(@(t,x)
funcionrotacion_pend_var_turbohelice(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,de
lta,alpha0,theta0,thetaf,thetatf,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,delta
T,Psl,gamma,p,psl,m,g,mu,epsilon),t0rot,x0rot,Optrot);


phirot=polyval(p_phi,xrot(:,1));
[sizetrot,~]=size(trot);
ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
for i=1:sizetrot
    t=trot(i);
    if t>thetatf
        thetarot(i)=thetaf+phirot(i);
    else
        thetarot(i)=interp1(ftheta,thetat,t)+phirot(i);
    end
end
thetafinalrot=thetarot(end)-phirot(end);
if thetafinalrot>=thetaf
    tiemporestanterotacion=0;
else
    tiemporestanterotacion=thetatf-trot(end);
end



if trot(end)>=200
    error_sinmotor=1;
else
    error_sinmotor=0;
end


z0vue=polyval(p_zg,xrot(end,1));
f0prima_vue=polyval(p_fprima,xrot(end,1));
z0puntovue=xrot(end,2)*f0prima_vue;
x0vue=[xrot(end,:),z0vue,z0puntovue]';
```

```
t0vue=[0 50];

[tvue,xvue]=ode45(@(t,x)
funcionvuelo_pend_var_turbohelice(t,x,Vw,p_phi,p_zg,p_fprima,p_fdosprima,a,AR,
delta,alpha0,thetaf,thetafinalrot,tiemporestanterotacion,S,rho,Cd0,deltaCdflap
,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,psl,m,g,epsilon),t0vue,x0vue,Optvue);


ttotal=[trod1;trod2+trod1(end);trot+trod2(end)+trod1(end);tvue+trot(end)+trod2
(end)+trod1(end)];
xtotal=[xrod1(:,1);xrod2(:,1);xrot(:,1);xvue(:,1)];
TOD_sinmotor=xtotal(end);
d=xvue(end,1)-xrot(end,1);
TOR_sinmotor=xrot(end,1)+d/2;
phirot=polyval(p_phi,xrot(end,1));
Vlof_sinmotor=xrot(end,2)/cos(phirot);
phivue=polyval(p_phi,xvue(:,1));
Vax=xvue(:,2)+Vw.*cos(phivue);
Vaz=xvue(:,4)+Vw.*sin(phivue);
gammaAvue=atan(Vaz./Vax);
V2_sinmotor=Vax(end)/cos(gammaAvue(end));

end
```

- Función parada ASD

```
function
F=funcionparada_pend_const(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,alpha0,theta
0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,m,g,mu,epsilon)

theta1=theta0+phi;

alpha=theta1-phi;
Va=(x(2)/cos(phi))+Vw;
gammaA=phi;

beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;

T=0;

F=zeros(2,1);
F(1)=x(2);
F(2)=(1/(m*(1+((sin(phi)+mu*cos(phi))*m*fdosprima)/(m*(cos(phi)-
mu*sin(phi))))))*(T*cos(theta0+epsilon)-L*sin(gammaA)-D*cos(gammaA)-
```

```
(sin(phi)+mu*cos(phi))*((m*fdosprima*x(2)-T*sin(epsilon+theta0)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi))));


end
```

```
function
F=funcionparada_pend_var(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,delta,alpha0,t
heta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,m,g,mu,epsilon)

phi=polyval(p_phi,x(1));
fprima=polyval(p_fprima,x(1));
fdosprima=polyval(p_fdosprima,x(1));

theta1=theta0+phi;

alpha=theta1-phi;
Va=(x(2)/cos(phi))+Vw;
gammaA=phi;

beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;

T=0;

F=zeros(2,1);
F(1)=x(2);
F(2)=(1/(m*(1+((sin(phi)+mu*cos(phi))*m*fdosprima)/(m*(cos(phi)-
mu*sin(phi))))))*(T*cos(theta0+epsilon)-L*sin(gammaA)-D*cos(gammaA)-
(sin(phi)+mu*cos(phi))*((m*fdosprima*x(2)-T*sin(epsilon+theta0)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi))));

end
```

- Función rodadura

```
function
F=funcionrodadura_pend_const_turbofan(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,a
lpha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,psl,
m,g,mu,epsilon)


theta1=theta0+phi;
```

```
alpha=theta1-phi;
Va=(x(2)/cos(phi))+Vw;
gammaA=phi;


beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=deltaT*Tsl*(1+0.5*(gamma-1)*(Va/a)^2)^(gamma/(gamma-1))*(1-
0.49*sqrt(abs(Va)/a))*(rho/rhosl);

F=zeros(2,1);
F(1)=x(2);
F(2)=(1/(m*(1+((sin(phi)+mu*cos(phi))*m*fdosprima)/(m*(cos(phi)-
mu*sin(phi))))))*(T*cos(theta0+epsilon)-L*sin(gammaA)-D*cos(gammaA)-
(sin(phi)+mu*cos(phi))*((m*fdosprima*x(2)-T*sin(epsilon+theta0)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi))));


end
```

```
function
F=funcionrodadura_pend_const_turbohelice(t,x,Vw,phi,fprima,fdosprima,a,AR,delt
a,alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,psl
,m,g,mu,epsilon)

theta1=theta0+phi;

alpha=theta1-phi;
Va=(x(2)/cos(phi))+Vw;
gammaA=phi;

beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=((nuP/abs(Va))*(deltaT*Psl*(1+((gamma-1)/2)*(Va/a)^2)^(gamma/(gamma-
1))*p/psl));

F=zeros(2,1);
F(1)=x(2);
F(2)=(1/(m*(1+((sin(phi)+mu*cos(phi))*m*fdosprima)/(m*(cos(phi)-
mu*sin(phi))))))*(T*cos(theta0+epsilon)-L*sin(gammaA)-D*cos(gammaA)-
(sin(phi)+mu*cos(phi))*((m*fdosprima*x(2)-T*sin(epsilon+theta0)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi))));

End
```

```
function
F=funcionrodadura_pend_var_turbofan(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,del
ta,alpha0,theta0,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,Tsl,gamma,p,
psl,m,g,mu,epsilon)


phi=polyval(p_phi,x(1));
fprima=polyval(p_fprima,x(1));
fdosprima=polyval(p_fdosprima,x(1));

theta1=theta0+phi;

alpha=theta1-phi;
Va=(x(2)/cos(phi))+Vw;
gammaA=phi;

beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=deltaT*Tsl*(1+0.5*(gamma-1)*(Va/a)^2)^(gamma/(gamma-1))*(1-
0.49*sqrt(abs(Va)/a))*(rho/rhosl);

F=zeros(2,1);
F(1)=x(2);
F(2)=(1/(m*(1+((sin(phi)+mu*cos(phi))*m*fdosprima)/(m*(cos(phi)-
mu*sin(phi))))))*(T*cos(theta0+epsilon)-L*sin(gammaA)-D*cos(gammaA)-
(sin(phi)+mu*cos(phi))*((m*fdosprima*x(2)-T*sin(epsilon+theta0)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi))));

end
```

```
function
F=funcionrodadura_pend_var_turbohelice(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,
delta,alpha0,theta0,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p
,psl,m,g,mu,epsilon)


phi=polyval(p_phi,x(1));
fprima=polyval(p_fprima,x(1));
fdosprima=polyval(p_fdosprima,x(1));

theta1=theta0+phi;

alpha=theta0-phi;
Va=(x(2)/cos(phi))+Vw;
gammaA=phi;

beta=sqrt(1-(Va/a)^2);
```

```
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=((nuP/abs(Va))*(deltaT*Psl*(1+((gamma-1)/2)*(Va/a)^2)^(gamma/(gamma-
1))*p/psl));

F=zeros(2,1);
F(1)=x(2);
F(2)=(1/(m*(1+((sin(phi)+mu*cos(phi))*m*fdosprima)/(m*(cos(phi)-
mu*sin(phi))))))*(T*cos(theta0+epsilon)-L*sin(gammaA)-D*cos(gammaA)-
(sin(phi)+mu*cos(phi))*((m*fdosprima*x(2)-T*sin(epsilon+theta0)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi))));

end
```

- Función rotación

```
function
F=funcionrotacion_pend_const_turbofan(t,x,Vw,phi,fprima,fdosprima,a,AR,delta,a
lpha0,theta0,thetaf,thetatf,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,deltaT,T
sl,gamma,p,psl,m,g,mu,epsilon)

ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
if t>thetatf
    theta=thetaf+phi;
else
    theta=interp1(ftheta,thetat,t)+phi;
end

alpha=theta-phi;
Va=(x(2)/cos(phi))+Vw;
gammaA=phi;

beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=deltaT*Tsl*(1+0.5*(gamma-1)*(Va/a)^2)^(gamma/(gamma-1))*(1-
0.49*sqrt(Va/a))*(rho/rhosl);


F=zeros(2,1);
F(1)=x(2);
F(2)=(1/(m*(1+((sin(phi)+mu*cos(phi))*m*fdosprima)/(m*(cos(phi)-
mu*sin(phi))))))*(T*cos(theta+epsilon)-L*sin(gammaA)-D*cos(gammaA)-
```

```
(sin(phi)+mu*cos(phi))*((m*fdosprima*x(2)-T*sin(epsilon+theta)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi))));

global N
N=(m*(fdosprima*x(2)+fprima*F(2))-T*sin(epsilon+theta)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi));
end
```

```
function
F=funcionrotacion_pend_const_turbohelice(t,x,Vw,phi,fprima,fdosprima,a,AR,delt
a,alpha0,theta0,thetaf,thetatf,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,deltaT,
Psl,gamma,p,psl,m,g,mu,epsilon)

ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
if t>thetatf
    theta=thetaf+phi;
else
    theta=interp1(ftheta,thetat,t)+phi;
end

alpha=theta-phi;
Va=(x(2)/cos(phi))+Vw;
gammaA=phi;

beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=((nuP/Va)*(deltaT*Psl*(1+((gamma-1)/2)*(Va/a)^2)^(gamma/(gamma-1))*p/psl));


F=zeros(2,1);
F(1)=x(2);
F(2)=(1/(m*(1+((sin(phi)+mu*cos(phi))*m*fdosprima)/(m*(cos(phi)-
mu*sin(phi))))))*(T*cos(theta+epsilon)-L*sin(gammaA)-D*cos(gammaA)-
(sin(phi)+mu*cos(phi))*((m*fdosprima*x(2)-T*sin(epsilon+theta)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi))));

global N
N=(m*(fdosprima*x(2)+fprima*F(2))-T*sin(epsilon+theta)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi));
end
```

```
function
F=funcionrotacion_pend_var_turbofan(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,del
ta,alpha0,theta0,thetaf,thetatf,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k,delt
aT,Tsl,gamma,p,psl,m,g,mu,epsilon)
```

```
phi=polyval(p_phi,x(1));
fprima=polyval(p_fprima,x(1));
fdosprima=polyval(p_fdosprima,x(1));

ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
if t>thetatf
    theta=thetaf+phi;
else
    theta=interp1(ftheta,thetat,t)+phi;
end



alpha=theta-phi;
Va=(x(2)/cos(phi))+Vw;
gammaA=phi;

beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=deltaT*Tsl*(1+0.5*(gamma-1)*(Va/a)^2)^(gamma/(gamma-1))*(1-
0.49*sqrt(Va/a))*(rho/rhosl);


F=zeros(2,1);
F(1)=x(2);
F(2)=(1/(m*(1+((sin(phi)+mu*cos(phi))*m*fdosprima)/(m*(cos(phi)-
mu*sin(phi))))))*(T*cos(theta+epsilon)-L*sin(gammaA)-D*cos(gammaA)-
(sin(phi)+mu*cos(phi))*((m*fdosprima*x(2)-T*sin(epsilon+theta)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi))));

global N
N=(m*(fdosprima*x(2)+fprima*F(2))-T*sin(epsilon+theta)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi));
end
```

```
function
F=funcionrotacion_pend_var_turbohelice(t,x,Vw,p_phi,p_fprima,p_fdosprima,a,AR,
delta,alpha0,theta0,thetaf,thetatf,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nuP,del
taT,Psl,gamma,p,psl,m,g,mu,epsilon)

phi=polyval(p_phi,x(1));
fprima=polyval(p_fprima,x(1));
fdosprima=polyval(p_fdosprima,x(1));

ftheta=linspace(0,thetatf);
thetat=theta0+((thetaf-theta0)/thetatf)*ftheta;
if t>thetatf
```

```
    theta=thetaf+phi;
else
    theta=interp1(ftheta,thetat,t)+phi;
end


alpha=theta-phi;
Va=(x(2)/cos(phi))+Vw;
gammaA=phi;

beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=((nuP/Va)*(deltaT*Psl*(1+((gamma-1)/2)*(Va/a)^2)^(gamma/(gamma-1))*p/psl));


F=zeros(2,1);
F(1)=x(2);
F(2)=(1/(m*(1+((sin(phi)+mu*cos(phi))*m*fdosprima)/(m*(cos(phi)-
mu*sin(phi))))))*(T*cos(theta+epsilon)-L*sin(gammaA)-D*cos(gammaA)-
(sin(phi)+mu*cos(phi))*((m*fdosprima*x(2)-T*sin(epsilon+theta)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi))));

global N
N=(m*(fdosprima*F(1)+fprima*F(2))-T*sin(theta+epsilon)-
L*cos(gammaA)+D*sin(gammaA)+m*g)/(cos(phi)-mu*sin(phi));
end
```

- Función vuelo

```
function
F=funcionvuelo_pend_const_turbofan(t,x,Vw,phi,fprima,a,AR,delta,alpha0,thetaf,
thetafinalrot,tiemporestanterotacion,S,rho,rhosl,Cd0,deltaCdflap,deltaCdgear,k
,deltaT,Tsl,gamma,p,psl,m,g,epsilon)


if thetafinalrot>=thetaf
    theta=thetaf+phi;
else
    ftheta=linspace(0,tiemporestanterotacion);
    thetat=thetafinalrot+((thetaf-
thetafinalrot)/tiemporestanterotacion)*ftheta;
    if t>tiemporestanterotacion
        theta=thetaf+phi;
    else
        theta=interp1(ftheta,thetat,t)+phi;
    end
```

```
end

Vax=x(2)+Vw*cos(phi);
Vaz=x(4)+Vw*sin(phi);
gammaA=atan(Vaz/Vax);
Va=Vax/cos(gammaA);
alpha=theta-gammaA;

beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=deltaT*Tsl*(1+0.5*(gamma-1)*(Va/a)^2)^(gamma/(gamma-1))*(1-
0.49*sqrt(Va/a))*(rho/rhosl);

global h
h=(x(3)-x(1)*tan(phi))*sqrt(1+fprima^2);

F=zeros(4,1);
F(1)=x(2);
F(2)=(1/m)*(T*cos(theta+epsilon)-L*sin(gammaA)-D*cos(gammaA));
F(3)=x(4);
F(4)=(1/m)*(T*sin(theta+epsilon)+L*cos(gammaA)-D*sin(gammaA)-m*g);


end
```

```
function
F=funcionvuelo_pend_const_turbohelice(t,x,Vw,phi,fprima,a,AR,delta,alpha0,thet
af,thetafinalrot,tiemporestanterotacion,S,rho,Cd0,deltaCdflap,deltaCdgear,k,nu
P,deltaT,Psl,gamma,p,psl,m,g,epsilon)


if thetafinalrot>=thetaf
    theta=thetaf+phi;
else
    ftheta=linspace(0,tiemporestanterotacion);
    thetat=thetafinalrot+((thetaf-
thetafinalrot)/tiemporestanterotacion)*ftheta;
    if t>tiemporestanterotacion
        theta=thetaf+phi;
    else
        theta=interp1(ftheta,thetat,t)+phi;
    end
end



Vax=x(2)+Vw*cos(phi);
Vaz=x(4)+Vw*sin(phi);
gammaA=atan(Vaz/Vax);
```

```
Va=Vax/cos(gammaA);
alpha=theta-gammaA;

beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=((nuP/Va)*(deltaT*Psl*(1+((gamma-1)/2)*(Va/a)^2)^(gamma/(gamma-1))*p/psl));

global h
h=(x(3)-x(1)*tan(phi))*sqrt(1+fprima^2);

F=zeros(4,1);
F(1)=x(2);
F(2)=(1/m)*(T*cos(theta+epsilon)-L*sin(gammaA)-D*cos(gammaA));
F(3)=x(4);
F(4)=(1/m)*(T*sin(theta+epsilon)+L*cos(gammaA)-D*sin(gammaA)-m*g);


end
```

```
function
F=funcionvuelo_pend_var_turbofan(t,x,Vw,p_phi,p_zg,p_fprima,p_fdosprima,a,AR,d
elta,alpha0,thetaf,thetafinalrot,tiemporestanterotacion,S,rho,rhosl,Cd0,deltaC
dflap,deltaCdgear,k,deltaT,Tsl,gamma,p,psl,m,g,epsilon)

phi=polyval(p_phi,x(1));

if thetafinalrot>=thetaf
    theta=thetaf+phi;
else
    ftheta=linspace(0,tiemporestanterotacion);
    thetat=thetafinalrot+((thetaf-
thetafinalrot)/tiemporestanterotacion)*ftheta;
    if t>tiemporestanterotacion
        theta=thetaf+phi;
    else
        theta=interp1(ftheta,thetat,t)+phi;
    end
end



fprima=polyval(p_fprima,x(1));
fdosprima=polyval(p_fdosprima,x(1));
zg=polyval(p_zg,x(1));
Vax=x(2)+Vw*cos(phi);
Vaz=x(4)+Vw*sin(phi);
gammaA=atan(Vaz/Vax);
Va=Vax/cos(gammaA);
```

```
alpha=theta-gammaA;

beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=deltaT*Tsl*(1+0.5*(gamma-1)*(Va/a)^2)^(gamma/(gamma-1))*(1-
0.49*sqrt(Va/a))*(rho/rhosl);


F=zeros(4,1);
F(1)=x(2);
F(2)=(1/m)*(T*cos(theta+epsilon)-L*sin(gammaA)-D*cos(gammaA));
F(3)=x(4);
F(4)=(1/m)*(T*sin(theta+epsilon)+L*cos(gammaA)-D*sin(gammaA)-m*g);

global h
h=(x(3)-zg)*sqrt(1+(fprima)^2);

end
```

```
function
F=funcionvuelo_pend_var_turbohelice(t,x,Vw,p_phi,p_zg,p_fprima,p_fdosprima,a,A
R,delta,alpha0,thetaf,thetafinalrot,tiemporestanterotacion,S,rho,Cd0,deltaCdfl
ap,deltaCdgear,k,nuP,deltaT,Psl,gamma,p,psl,m,g,epsilon)

phi=polyval(p_phi,x(1));

if thetafinalrot>=thetaf
    theta=thetaf+phi;
else
    ftheta=linspace(0,tiemporestanterotacion);
    thetat=thetafinalrot+((thetaf-
thetafinalrot)/tiemporestanterotacion)*ftheta;
    if t>tiemporestanterotacion
        theta=thetaf+phi;
    else
        theta=interp1(ftheta,thetat,t)+phi;
    end
end



fprima=polyval(p_fprima,x(1));
fdosprima=polyval(p_fdosprima,x(1));
zg=polyval(p_zg,x(1));
Vax=x(2)+Vw*cos(phi);
Vaz=x(4)+Vw*sin(phi);
gammaA=atan(Vaz/Vax);
Va=Vax/cos(gammaA);
alpha=theta-gammaA;
```

```
beta=sqrt(1-(Va/a)^2);
Cl=((2*pi*AR)/(2+sqrt(4+AR^2*beta^2*(1+(tan(delta)^2)/(beta^2)))))*(alpha-
alpha0);
L=0.5*rho*Va^2*S*Cl;
Cd=Cd0+deltaCdflap+deltaCdgear+k*Cl^2;
D=0.5*rho*Va^2*S*Cd;
T=((nuP/Va)*(deltaT*Psl*(1+((gamma-1)/2)*(Va/a)^2)^(gamma/(gamma-1))*p/psl));


F=zeros(4,1);
F(1)=x(2);
F(2)=(1/m)*(T*cos(theta+epsilon)-L*sin(gammaA)-D*cos(gammaA));
F(3)=x(4);
F(4)=(1/m)*(T*sin(theta+epsilon)+L*cos(gammaA)-D*sin(gammaA)-m*g);

global h
h=(x(3)-zg)*sqrt(1+(fprima*x(2))^2);

end
```

- Funciones condición de parada

```
function [value, isterminal, direction] = myEventacel_ASDsinmotor(t,x)
global V1
value      = x(2)-V1;
isterminal = 1;   % Stop the integration
direction  = 0;
end
```

```
function [value, isterminal, direction] = myEventparada_ASD(t,x)

value      = x(2);
isterminal = 1;   % Stop the integration
direction  = 0;
end
```

```
function [value, isterminal, direction] = myEventrod(t,x)
global Vr
value      = x(2)-Vr;
isterminal = 1;   % Stop the integration
direction  = 0;
end
```

```
function [value, isterminal, direction] = myEventrod1(t,x)
global Vef
value      = x(2)-Vef;
isterminal = 1;   % Stop the integration
direction  = 0;
end
```

```
function [value, isterminal, direction] = myEventrod2(t,x)
global Vr
value      = x(2)-Vr;
isterminal = 1;   % Stop the integration
direction  = 0;
end
```

```
function [value, isterminal, direction] = myEventrod_ASD(t,x)
global V1
value      = x(2)-V1;
isterminal = 1;   % Stop the integration
direction  = 0;
end
```

```
function [value, isterminal, direction] = myEventrod_ASDsinmotor(t,x)
global Vef
value      = x(2)-Vef;
isterminal = 1;   % Stop the integration
direction  = 0;
end
```

```
function [value, isterminal, direction] = myEventrot(t,x)
global N
value      = N;
isterminal = 1;   % Stop the integration
direction  = 0;
end
```

```
function [value, isterminal, direction] = myEventvue(t,x)
global h
value      = h-11;
isterminal = 1;   % Stop the integration
direction  = 0;
end
```