UNIVERSIDAD EUROPEA DE MADRID

ESCUELA DE ARQUITECTURA, INGENIERÍA Y DISEÑO

Máster Universitario en Ingeniería Aeronáutica

INFORME FINAL DEL PROYECTO

El problema de Lambert con variables universales. Algoritmo en *MATLAB*.

Ricardo Salazar González

Curso 2022-2023

Título: El problema de Lambert. Algoritmo en *MATLAB.*

Autor: Ricardo Salazar González

Tutor: Alejandro Ibrahim Ojea

Titulación: Máster universitario en Ingeniería Aeronáutica

Curso: 2022/23

Resumen

El objetivo final de este documento es el desarrollo de un código en *MATLAB* para la resolución de casos prácticos del problema de Lambert. Éste problema aborda, entre otros casos más generales, la determinación de la órbita de un cuerpo bajo la influencia de la acción gravitatoria de la tierra conociendo únicamente dos posiciones y un tiempo de vuelo entre ellas. Las bases matemáticas para llevar a cabo dicho cometido serán formuladas en los primeros capítulos.

Palabras clave: Problema de Lambert, mecánica orbital, algoritmo, MATLAB

Abstract

The main point of this project is the development of a code in *MATLAB* for solving practical cases of the Lambert's problem. This problem addresses, among other more general cases, the determination of the orbit of a body under the gravitational influence of the Earth knowing only two positions and a flight time between them. The mathematical bases to carry out this task will be presented in the first chapters.

Key words: Lambert's problem, orbital mechanics, algorithm, MATLAB

A mi familia

A mis maestros

Índice de contenidos

Índice de ilustraciones

Índice de tablas

Capítulo 1. Introducción

El siglo XX es una época trascendental para el desarrollo de la humanidad, siendo el avance tecnológico un factor fundamental para este desarrollo. En concreto, un punto de inflexión crucial en este avance tecnológico corre a cargo del vertiginoso ritmo de la competición entre dos superpotencias mundiales, Estados Unidos y la Unión Soviética, por la conquista de la última frontera: el espacio. Éstos países gastaron fortunas entre los últimos años de los 50 y los años 70 en impulsar proyectos que originaron impactos tanto científicos como geopolíticos, y que culminaron con las misiones Apolo y, con ello, la llegada del hombre a la luna en 1969. La carrera espacial había finalizado y Estados Unidos era el vencedor, pero como ocurre con cada final, un nuevo inicio tuvo lugar, y así es como llegó el comienzo de un nuevo mundo de posibilidades tecnológicas del cual todos somos hemos sido beneficiarios. Son muchos los campos que participaron en la carrera espacial y es por tanto muy amplio el legado que ésta nos ha dejado. Avances en la navegación, como el *GPS*, electrónica o tecnología de materiales son solo unos pocos ejemplos de la inconmensurable e inestimable huella que las investigaciones espaciales han dejado.

Dentro de la mecánica orbital, disciplina indispensable para que todo esto haya tenido lugar, y que estudia la compleja coreografía que siguen los cuerpos celestes movidos por las fuerzas gravitatorias, destaca un problema el cual su resolución ha supuesto un antes y un después en la comprensión de las trayectorias de los cuerpos. La resolución del problema de Lambert, formulado por primera vez por el matemático suizo-alemán Johann Heinrich Lambert en el siglo XVII, y resuelto gracias a contribuciones de enormes figuras como Lagrange y Gauss, supone el conocimiento de la trayectoria de un objeto a partir de dos posiciones conocidas y un tiempo de vuelo, lo cual, como bien evidente se hace, tiene profundas aplicaciones en la planificación de misiones espaciales.

Ilustración 1. Johann Heinrich Lambert

En este documento se establecerán las bases sobre las que se asienta tanto el planteamiento como la resolución del problema, y se elaborará un algoritmo para, mediante el uso de la aplicación *MATLAB*, poder resolver casos prácticos. En los capítulos 2 y 3 se estudiarán las ecuaciones que rigen el problema, en el capítulo 4 se desarrollará el algoritmo de resolución y se darán las pinceladas matemáticas necesarias para disponer del código completo, el cual se presentará en el Anexo I, tras el capítulo de conclusiones.

La complejidad del problema puede hacerse infinita, puesto que ésta es proporcional al número de cuerpos que intervengan. En el caso de estudio de este proyecto se abordará el problema de Lambert bajo la acción gravitacional de un único cuerpo influyente: la tierra. Es por tanto que las aplicaciones del código resultante se verán limitadas a las observaciones que puedan llegar a hacerse desde aquí.

El alcance de este proyecto englobará pues el presente documento en el que irá incluido tanto el desarrollo del proyecto como el código de *MATLAB* y sus funciones asociadas.

Capítulo 2. Introducción a la mecánica orbital

En este capítulo estableceremos las bases de la mecánica orbital necesarias para poder plantear, comprender y resolver el Problema de Lambert.

2.1 Problema de los n-cuerpos

El problema de los n-cuerpos es uno de los problemas fundamentales de la mecánica orbital que surge en el siglo XVII a raíz de la ley de la gravitación universal formulada por Sr. Isaac Newton, y comprende la predicción del movimiento de N cuerpos bajo la influencia del campo gravitacional propio del sistema. Puede plantearse de forma simplificada como:

> *Dadas las propiedades orbitales (masa, posición y velocidad) de un grupo de cuerpos, determinar las fuerzas interactivas actuantes; y consiguientemente, calcular sus movimientos orbitales para cualquier instante futuro.*

Estas fuerzas corresponden a las descritas en la ley de la gravitación universal. Según esta ley, un cuerpo de masa m_i experimenta una fuerza $\boldsymbol{F_{ij}}$ cuando está bajo la acción gravitacional de un cuerpo de masa m_j.

$$\boldsymbol{F_{ij}} = G\frac{m_i m_j(\boldsymbol{r_i} - \boldsymbol{r_j})}{\left\|\boldsymbol{r_i} - \boldsymbol{r_j}\right\|^3} \tag{2.1}$$

Donde G representa la constante de gravitación universal, y $\left\|\boldsymbol{r_i} - \boldsymbol{r_j}\right\|$ la distancia entre los vectores de posición $\boldsymbol{r_i}$ y $\boldsymbol{r_j}$.

Se trata de un problema caótico y complejo, cuya dificultad principalmente radica en las no-linealidades y en la variación continua del centro de gravedad del sistema debido a la variación en la posición de los cuerpos, resultando exponencialmente creciente con el número de cuerpos involucrados. Tanto es así que su resolución general aún no ha sido encontrada.

A pesar la dificultad para encontrar una solución analítica general, se han desarrollado diversas soluciones aproximadas para abordar el problema. Éstas van desde los métodos numéricos de integración de rangos de tiempo discretizados (Euler, Runge-Kutta), pasando por simplificaciones del problema, como el tratar los grandes cúmulos de masa como masas puntuales; y hasta llegar a la resolución de *casos especiales*. Entre estos casos especiales cabe destacar el problema de los 3 cuerpos restringido, en el cual se tratan dos cuerpos con masas de orden de magnitud similar y una tercera masa de orden significativamente inferior (muy usado para el desarrollo de las misiones lunares); o el *problema de los dos cuerpos*.

El uso de métodos numéricos queda restringido a los casos en los que se requiere mucha precisión en los resultados, puesto que el coste computacional asociado es muy elevado. No obstante, se pueden conseguir resultados muy útiles y prácticos usando soluciones de *casos especiales* anteriormente mencionadas.

Sin duda, este problema de fundamental importancia para múltiples campos, como la astrodinámica, la astrofísica o la cosmología, supone un auténtico reto para la ciencia moderna. Sin embargo, con los avances tecnológicos en computación podremos estudiar más a fondo para así avanzar en la comprensión de las leyes que gobiernan el universo.

2.2 El problema de los dos cuerpos

El problema de los dos cuerpos es un caso especial del problema de los n-cuerpos muy importante puesto que es tiene solución cerrada. Es muy usado como primera aproximación en programas espaciales ya que proporciona valores muy aproximados con modelos relativamente sencillos. Mediante usos de modelos de esferas de influencia permite separar el problema general de n-cuerpos en diversos problemas de dos cuerpos y así ir resolviendo el problema general por etapas.

El planteamiento general del problema considera la existencia de dos masas puntuales, m_1 y m_2, ubicadas en las posiciones $\boldsymbol{r_1}$ y $\boldsymbol{r_2}$ respecto a un sistema de referencia inercial $OXYZ$, y bajo la influencia de un campo de fuerzas de atracción mutua inversamente proporcional a la distancia que los separa, $\boldsymbol{r}$.

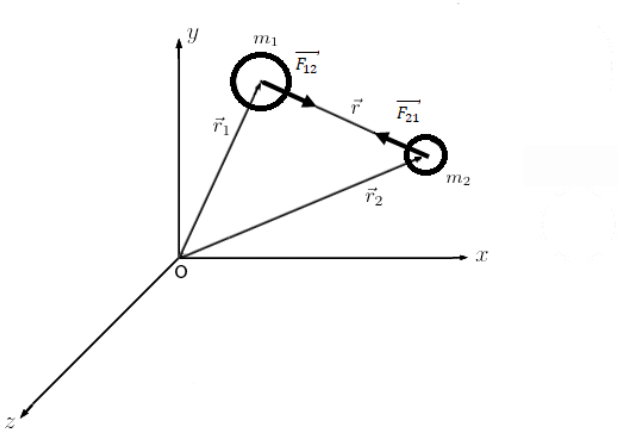


Ilustración 2. Formulación del problema de dos cuerpos.

De esta manera podemos escribir la distancia $\boldsymbol{r}$ como la diferencia entre las posiciones $\boldsymbol{r_2}$ y $\boldsymbol{r_1}$ para luego derivar en el tiempo dos veces.

$$\boldsymbol{r} = \boldsymbol{r_2} - \boldsymbol{r_1} \Longrightarrow \ddot{\boldsymbol{r}} = \ddot{\boldsymbol{r}}_2 - \ddot{\boldsymbol{r}}_1 \tag{2.2}$$

Ahora, teniendo en cuenta la primera ley de Newton ($\boldsymbol{F} = m \cdot \boldsymbol{a}$) podemos sustituir la aceleración como la fuerza por unidad de masa.

$$\ddot{\boldsymbol{r}} = \frac{\boldsymbol{F_{21}}}{m_2} - \frac{\boldsymbol{F_{12}}}{m_1} \tag{2.3}$$

Sustituyendo (2.1) y aplicando la tercera ley de Newton ($\boldsymbol{F_{12}} = -\boldsymbol{F_{21}}$)

$$\ddot{\boldsymbol{r}} = -\frac{G(m_1 + m_2)}{r^3}\boldsymbol{r} \tag{2.4}$$

Definiendo ahora el parámetro gravitacional estándar, μ

$$\mu = G(m_1 + m_2) \tag{2.5}$$

Sustituyendo en (2.4) obtenemos la ecuación que gobierna el movimiento de m_2.

$$\ddot{\boldsymbol{r}} = -\frac{\mu}{r^3}\boldsymbol{r} \tag{2.6}$$

2.3 La solución de Kepler

Para la resolución de la ecuación del movimiento, lo cual no entra dentro del alcance de este documento, se debe aplicar el principio de conservación del momento angular y la energía, así como la suposición de que la masa del segundo cuerpo es despreciable frente a la del primero, de tal manera que se llega a una ecuación que describe una trayectoria cónica.

$$r = \frac{h^2/\mu}{1 + e\cos\theta} \tag{2.7}$$

Siendo h el módulo del vector momento angular específico, $h = \|\boldsymbol{h}\| = \|\boldsymbol{r} \times \boldsymbol{v}\|$, e el módulo del vector excentricidad, $e = \|\boldsymbol{e}\| = \left\|\frac{\boldsymbol{v}\times\boldsymbol{h}}{\mu} - \frac{\boldsymbol{r}}{r}\right\|$, y θ el ángulo entre $\boldsymbol{e}$ y $\boldsymbol{r}$, es decir, la anomalía verdadera.

Esta ecuación corresponde a la conocida *órbita de Kepler*, la cual describe una trayectoria bien circular, elíptica, parabólica o hiperbólica, dependiendo del valor de e.

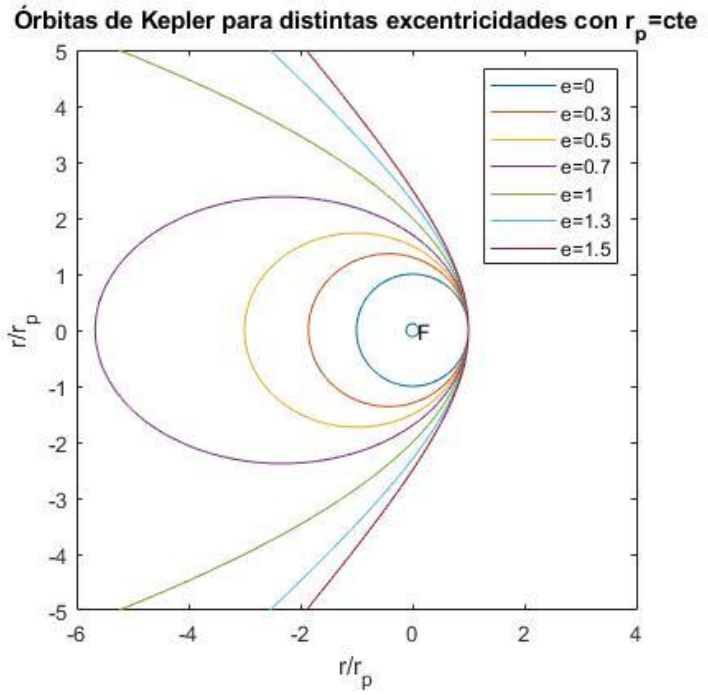


Ilustración 3. Órbitas de Kepler par distintas excentricidades con mismo foco y $r_p = cte$.

Es importante tener en cuenta que esta órbita es una idealización de las órbitas reales, las cuales varían con el tiempo debido a perturbaciones externas no contempladas en el modelo de los dos cuerpos aquí presentado.

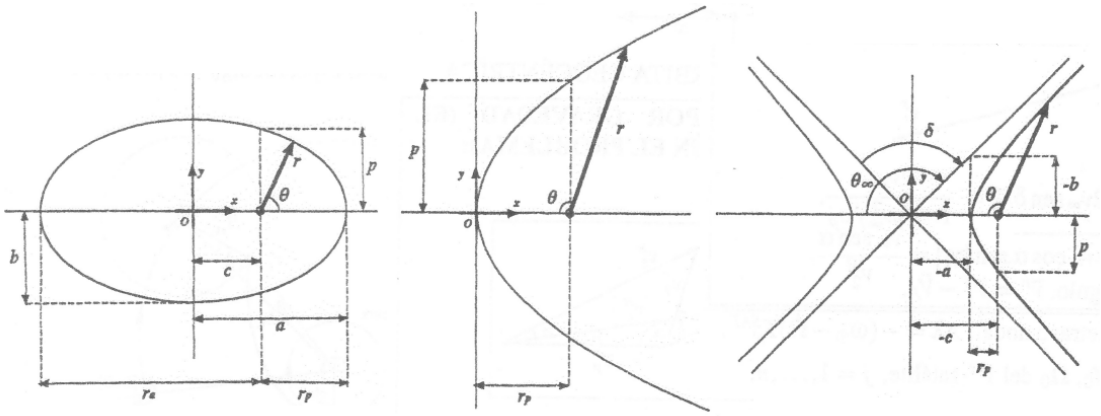


Ilustración 4. Geometría de una órbita elíptica (izquierda), parabólica (centro) e hiperbólica (derecha).

2.3.1 Parámetros de la cónica

Resulta necesario establecer ciertos parámetros para poder describir las propiedades de las órbitas, comenzaremos definiendo los parámetros más generales de las cónicas y luego definiremos ciertos parámetros dentro del sistema de referencia topográfico ecuatorial, es decir, dentro del marco contextual de la definición de órbitas alrededor de la tierra. El radio desde el foco, momento angular específico, la excentricidad y la anomalía verdadera definidas en el capítulo anterior son ejemplos de estos parámetros generales. Se definirá el resto de parámetros generales para poder describir las órbitas y se presentará una tabla resumiendo las particularidades ya que éstas varían en función de la excentricidad. En concreto, y para simplificar futuras definiciones, definimos el parámetro p.

$$p = h^2/\mu \tag{2.8}$$

El siguiente parámetro que abordamos es el *semieje mayor*, a, la cual es la suma de la distancia al semieje mayor y el semieje menor dividida por dos. Haciendo uso de la expresión (2.7) y (2.8) llegamos a:

$$a = \frac{r(\theta = 0) + r(\theta = \pi)}{2} = \frac{p}{1 - e^2} \tag{2.9}$$

También definiremos la *energía específica*, ε, la cual resulta de sumar la *energía potencial específica*, ε_p, y la *energía cinética específica*, ε_k. Además, por la ecuación *vis-viva* (conservación de la energía orbital específica) sabemos que ésta suma es constante.

$$\varepsilon = \frac{v^2}{2} - \frac{\mu}{r} = -\frac{\mu}{2a} \tag{2.10}$$

De esta manera podemos elaborar la siguiente tabla con los parámetros de las órbitas en función del valor de la excentricidad.

$\boldsymbol{e}$	$\boldsymbol{a}$	$\boldsymbol{\varepsilon}$	$\boldsymbol{\theta}$	***tipo de orbita***
$\mathbf{0}$	> 0	< 0	$[0,2\pi)$	$circular$
$\mathbf{0 < e < 1}$	> 0	< 0	$(-\pi,\pi)$	$elíptica$
$\mathbf{1}$	∞	0	$(-\pi,\pi)$	$parabólica$
$\mathbf{> 1}$	< 0	> 0	$(-\theta_\infty,\theta_\infty), \theta_\infty = \cos^{-1}(-1/e)$	$hiperbólica$

Tabla 1. Parámetros de las órbitas en función de la excentricidad, e.

2.3.2 Elementos orbitales clásicos

Existe un conjunto de seis parámetros que permiten especificar la posición de un cuerpo dentro de una órbita de Kepler específica. Este conjunto de parámetros son los denominados *elementos orbitales* clásicos o *elementos keplerianos*. La excentricidad y el semieje mayor son dos parámetros generales que forman parte de los elementos orbitales. La diferencia es que estos parámetros son propios de la cónica, y el resto indican la orientación de ésta respecto a un sistema de referencia concreto, o, en el caso del *argumento del perigeo*, la posición del satélite dentro de la órbita. Puesto que en nuestro caso el objetivo del problema es la identificación de órbitas en torno a el planeta tierra, entonces nuestro sistema de referencia respecto al cual estarán definidos los elementos clásicos orbitales, y sobre el cual se resolverá el problema de Lambert, será el *Sistema Geocéntrico Ecuatorial* "0".

Éste sistema es un sistema $Oxyz$ cuyo origen, O, está centrado en la tierra y cuyo eje $\boldsymbol{Oz}$ apunta hacia el polo norte. Esto deja al ecuador en el plano Oxy. Para completar la definición queda fijar la orientación del eje $\boldsymbol{Ox}$, el cual está direccionado hacia el primer punto de Aries, Υ, el cual resulta del corte del plano de la eclíptica con el plano ecuatorial, cuando el sol pasa del hemisferio sur al hemisferio norte iniciándose así la primavera en el hemisferio norte. El eje $\boldsymbol{Oy}$ por su parte sería el producto vectorial $\boldsymbol{Oz} \times \boldsymbol{Ox}$.

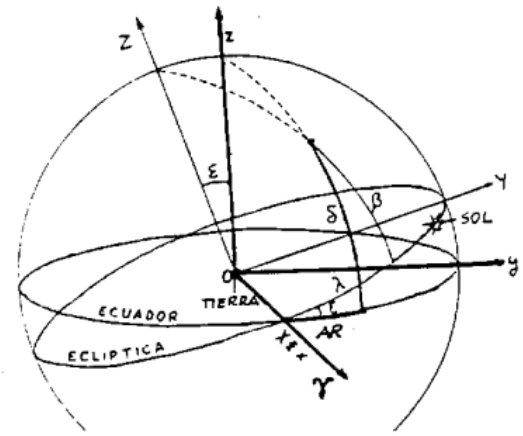


Ilustración 5. Sistema Geocéntrico Ecuatorial

El siguiente elemento que vamos a definir la inclinación de la órbita, puesto que dependiendo del valor tendremos diferencias en la definición de ciertos elementos, es decir, tendremos *casos especiales*. En estos casos especiales, que también dependerán de la excentricidad, el número de elementos orbitales será menor que los 6 que previamente hemos establecido como necesarios. Siguiendo con la inclinación, ésta se define como el ángulo entre el momento angular específico y el eje $\boldsymbol{Oz}$. Definiendo $\boldsymbol{h}$ en coordenadas geocéntricas, tenemos:

$$i = \cos^{-1}\frac{h_z}{|\boldsymbol{h}|}\,, con\ i \in [0°, 180°] \tag{2.11}$$

Por convención, definiremos en grados éste y todos los ángulos que definamos en éste sub capítulo.

El siguiente parámetro que definiremos será la *longitud del nodo ascendente* o *RAAN*, Ω, que es la distancia angular entre la dirección del primer punto de Aries y la dirección en la que el satélite (u otro cuerpo celeste) cruza del hemisferio sur al hemisferio norte. Ésta dirección es conocida como el *nodo ascendente*, y está representada por el símbolo ☊. El vector unitario que define esta dirección, n, está definido de la siguiente manera:

$$\boldsymbol{n} = \frac{\boldsymbol{Oz} \times \boldsymbol{h}}{|\boldsymbol{Oz} \times \boldsymbol{h}|} \tag{2.12}$$

De esta manera, el ángulo Ω queda definido como

$$\Omega = \begin{cases} 360 - \cos^{-1} n_x, & n_y < 0 \\ \cos^{-1} n_x, & n_y \geq 0 \end{cases} \tag{2.13}$$

El siguiente elemento que definiremos será el *argumento del periastro*, o en este caso, argumento *del perigeo*, ω, esto es el ángulo entre el nodo ascendente y el vector excentricidad.

$$\omega = \begin{cases} 360 - \cos^{-1}\left(\dfrac{\boldsymbol{n}\cdot\boldsymbol{e}}{e}\right), & e_z < 0 \\ \cos^{-1}\left(\dfrac{\boldsymbol{n}\cdot\boldsymbol{e}}{e}\right), & e_z \geq 0 \end{cases} \tag{2.14}$$

Por último, queda la definición de la posición del cuerpo en la órbita. Para ello definimos la *anomalía verdadera*, θ, que es el ángulo entre el vector posición, $\boldsymbol{r}$, y el vector excentricidad, $\boldsymbol{e}$.

$$\theta = \begin{cases} 360 - \cos^{-1}\left(\dfrac{\boldsymbol{r}\cdot\boldsymbol{e}}{re}\right), & \boldsymbol{r}\cdot\boldsymbol{v} < 0 \\ \cos^{-1}\left(\dfrac{\boldsymbol{r}\cdot\boldsymbol{e}}{re}\right), & \boldsymbol{r}\cdot\boldsymbol{v} \geq 0 \end{cases} \tag{2.15}$$

Para el caso de elipses y circunferencias, es normal dar este último valor como la *anomalía media, M*, la cual es la fracción del periodo orbital transcurrido desde el paso del objeto por el periápside, t_0, hasta la posición determinada, t. Éste valor está dado como ángulo. También puede explicarse como el ángulo que forma con el eje principal de la elipse un *objeto ficticio* que se mueve con movimiento circular uniforme trazando una órbita cuyo diámetro es equivalente al semieje mayor de la elipse. Está definido de la siguiente manera.

$$M = \sqrt{\frac{\mu}{a^3}}(t - t_0) \tag{2.16}$$

Con este conjunto de elementos $(e, a, i, \Omega, \omega, \theta)$ ya tendríamos una definición precisa de la posición de un objeto dentro de una órbita específica. Ahora vamos a exponer los casos específicos donde el número de elementos de este conjunto se ve reducido.

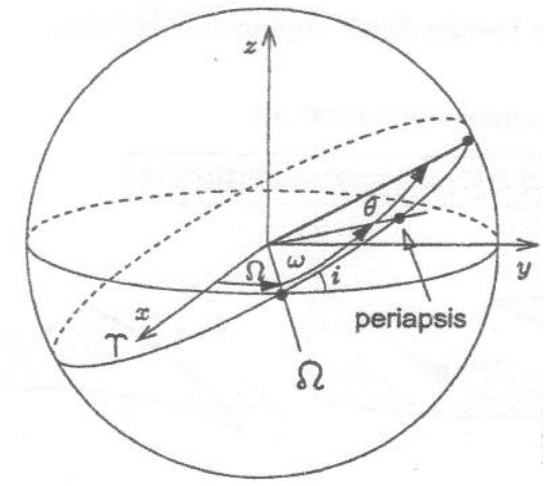


Ilustración 6. Elementos orbitales clásicos.

2.3.2.1 Casos especiales

Los casos especiales se producen cuando la excentricidad y la inclinación toman valores muy determinados. En cualquiera de los casos la excentricidad y el semieje mayor están completamente definidos y, por tanto, permanecen inalterados.

2.3.2.1.1 Caso $e = 0$ y $i \neq 0° \neq 180°$

En este caso tenemos que el vector excentricidad es nulo, por tanto, el argumento del perigeo y la anomalía verdadera están indefinidas. En este caso tenemos una variable u que será equivalente a la suma de ambos parámetros, $u = \omega + \theta$. Se define entonces como el ángulo entre el vector de posición, $\boldsymbol{r}$, y el nodo ascendente, $\boldsymbol{n}$.

$$u = \begin{cases} 360 - \cos^{-1}\left(\dfrac{\boldsymbol{r}\cdot\boldsymbol{n}}{r}\right), & r_z < 0 \\ \cos^{-1}\left(\dfrac{\boldsymbol{r}\cdot\boldsymbol{n}}{r}\right), & r_z \geq 0 \end{cases} \tag{2.17}$$

Éste valor determina la posición del objeto en la órbita respecto al nodo ascendente. El conjunto de elementos orbitales pasa a tener los siguientes 5 elementos, (e, a, i, Ω, u).

2.3.2.1.2 Caso $e \neq 0$ y $i = 0°$ o $i = 180°$

En este caso lo que sucede es que el nodo ascendente está indefinido porque en ningún momento el objeto pasa del hemisferio sur al hemisferio norte. Es por ello que definimos la *longitud del perigeo*, $\bar{\omega}$. Éste es equivalente a la suma de la longitud del nodo ascendente y el argumento del perigeo, $\bar{\omega} = \Omega + \omega$. El perigeo estará definido respecto a la orientación del primer punto de Aries. De esta manera, cuando $i = 0°$:

$$\bar{\omega} = \begin{cases} 360 - \cos^{-1}\left(\dfrac{e_x}{e}\right), & e_y < 0;\ i = 0° \\ \cos^{-1}\left(\dfrac{e_x}{e}\right), & e_y \geq 0;\ i = 0° \end{cases} \tag{2.18}$$

Mientras que para el caso $i = 180°$ sería al revés. De esta manera el conjunto de elementos orbitales vuelve a tener 5 elementos, aunque en este caso serían $(e, a, i, \bar{\omega}, \theta)$.

2.3.2.1.3 Caso $e = 0$ y $i = 0°$ o $i = 180°$

En este caso la longitud del nodo ascendente, el argumento del perigeo y la anomalía verdadera se conglomeran en un único parámetro, λ_T. Éste está definido como el ángulo entre el vector posición, $\boldsymbol{r}$, y el primer punto de Aries. Entonces, para órbitas con $i = 0°$ quedaría:

$$\lambda_T = \begin{cases} 360 - \cos^{-1}\left(\frac{r_x}{r}\right), & r_y < 0;\ i = 0° \\ \cos^{-1}\left(\frac{r_x}{r}\right), & r_y \geq 0;\ i = 0° \end{cases} \tag{2.19}$$

De igual manera al caso anterior, si $i = 180°$ sería al revés. De esta manera pasamos a tener 4 elementos orbitales, es decir (e, a, i, λ_T), de los cuales evidentemente algunos ya conocemos.

2.4 Coeficientes de Lagrange

Para la resolución del problema de Lambert haremos uso de las funciones de Lagrange. Éstas funciones, f y g, y sus derivadas, permiten expresar la posición y velocidad de un objeto en órbita, $\boldsymbol{r}$ y $\boldsymbol{v}$, como combinación lineal de la posición y velocidad en un instante t_0, $\boldsymbol{r_0}$ y $\boldsymbol{v_0}$.

$$\boldsymbol{r} = f\boldsymbol{r_0} + g\boldsymbol{v_0} \tag{2.20}$$

$$\boldsymbol{v} = \dot{f}\boldsymbol{r_0} + \dot{g}\boldsymbol{v_0} \tag{2.21}$$

Estas funciones toman su nombre en honor al matemático francés Joseph-Louis Lagrange (1736-1813), famoso entre otras cosas por su reformulación de la mecánica clásica en 1788, dando lugar a la *mecánica lagrangiana*. El desarrollo matemático queda fuera del alcance de este proyecto, pero puede encontrarse en (Curtis, 2013).

El resultado que mostramos a continuación representa los valores de las funciones de Lagrange y sus derivadas en función de la diferencia de la anomalía verdadera entre los instantes t y t_0, $\Delta\theta$.

$$f = 1 - \frac{\mu r}{h^2}(1 - \cos\Delta\theta) \tag{2.22}$$

$$g = \frac{r r_0}{h}\sin\Delta\theta \tag{2.23}$$

$$\dot{f} = \frac{\mu}{h}\frac{1 - \cos\Delta\theta}{\sin\Delta\theta}\left[\frac{\mu}{h^2}(1 - \cos\Delta\theta) - \frac{1}{r_0} - \frac{1}{r}\right] \tag{2.24}$$

$$\dot{g} = 1 - \frac{\mu r_0}{h^2}(1 - \cos\Delta\theta) \tag{2.25}$$

Una propiedad importante a destacar, y que nos será útil, es que $f\dot{g} - \dot{f}g = 1$.

Capítulo 3. Problema de Lambert

Como ya se ha especificado con anterioridad, el principal objetivo de este documento es la elaboración de un código numérico para la resolución del problema de Lambert. Habiendo hecho una introducción breve de los elementos de la mecánica orbital de los cuales haremos uso, pasamos al capítulo donde planteamos y resolvemos el problema.

3.1 Planteamiento del problema de Lambert

El problema de Lambert es un problema de condición de frontera para la ecuación (2.6). La formulación del problema es la siguiente:

> *Dados dos instantes de tiempo t_1 y t_2, y dos puntos en el espacio P_1 y P_2, con vectores de posición, $\boldsymbol{r_1}$ y $\boldsymbol{r_2}$, hallar la función $\boldsymbol{r}(t)$ que satisface $\boldsymbol{r}(t=t_1)=\boldsymbol{r_1}$ y $\boldsymbol{r}(t=t_2)=\boldsymbol{r_2}$ sabiendo que el movimiento está regido por la ecuación diferencial $\ddot{\boldsymbol{r}}=-\frac{\mu}{r^3}\boldsymbol{r}$.*

En este planteamiento se está teniendo en cuenta la hipótesis de que la masa del segundo objeto es despreciable frente a la del primero, siendo por tanto la solución del problema una órbita de Kepler.

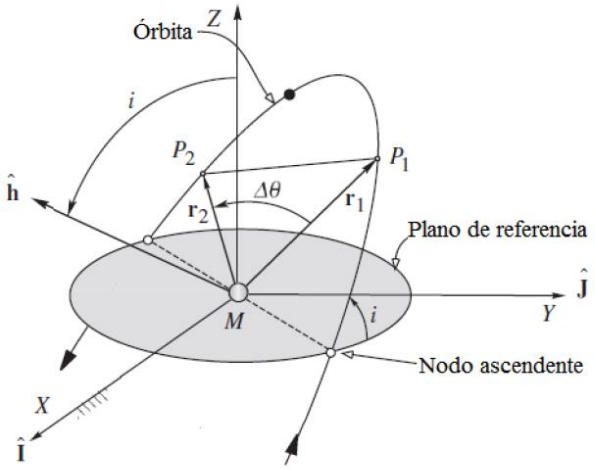


Ilustración 7. Problema de Lambert.

3.2 Resolución del problema de Lambert

La resolución del problema se llevará a cabo usando la formulación del problema de los dos cuerpos usando *variables universales*. En esta primera parte del capítulo serán presentadas las herramientas matemáticas necesaria para la resolución del problema.

Lo más importante a tener en cuenta es que, como el problema quedaría resuelto encontrando $\boldsymbol{v_1}$ y, por los coeficientes de Lagrange, existe una relación entre $\boldsymbol{r_1}$, $\boldsymbol{r_2}$ y $\boldsymbol{v_1}$ (ecuación 2.20), el objetivo final consistirá en encontrar un valor para estos coeficientes, que

estará definido intrínsecamente por $\Delta t = t_2 - t_1$. Como es de esperar, en esta primera parte presentaremos la formulación de los coeficientes de Lagrange en función de las previamente mencionadas *variables universales*.

Éstas variables surgen cuando se lleva a cabo la *transformación de Sundman* definida en (Battin, 1999).

$$\sqrt{\mu}\frac{dt}{d\chi} = r \tag{3.1}$$

Donde χ representa una variable independiente nueva cuyo significado es una especie de una *anomalía generalizada*. La razón de ser del uso de esta variable radica en que cuando se usa χ en lugar de t como variable independiente las ecuaciones diferenciales no lineales se transforman en unas ecuaciones diferenciales lineales de coeficientes constantes.

La variable χ puede escribirse de la siguiente manera como función de las anomalías más clásicas dadas en dos tiempos t y t_0 determinados.

$$\chi = \begin{cases} \sqrt{a}(E - E_0), & 0 < e < 1 \\ \sqrt{a}(\tan\frac{\theta}{2} - \tan\frac{\theta_0}{2}), & e = 1 \\ \sqrt{a}(H - H_0), & e > 1 \end{cases} \tag{3.2}$$

Donde E y H son la anomalía excéntrica elíptica y parabólica respectivamente.

Para poder escribir los coeficientes de Lagrange usando la formulación de variables universales debemos recurrir al uso de una familia de funciones llamadas funciones de Stumpff (Danby, 1998) las cuales vienen dadas por la siguiente expresión, la cual es convergente para todo z real, donde $z = \chi^2/a$:

$$C_k(z) = \sum_{n=0}^{\infty} \frac{(-1)^n z^n}{(2n+k)!}\ , \qquad k = 0, 1, 2, \ldots \tag{3.3}$$

A partir de esta ecuación, comparando esta serie con la serie de Taylor de las funciones trigonométricas, podemos hallar el siguiente resultado:

$$C_0(z) = \begin{cases} \cos\sqrt{z}, & z > 0 \\ 1, & z = 0 \\ \cosh\sqrt{-z}, & z < 0 \end{cases} \tag{3.4}$$

$$C_1(z) = \begin{cases} \dfrac{\sin\sqrt{z}}{\sqrt{z}}, & z > 0 \\ 1, & z = 0 \\ \dfrac{\sinh\sqrt{-z}}{\sqrt{-z}}, & z < 0 \end{cases} \tag{3.5}$$

Además, puede demostrarse que las funciones de Stumpff satisfacen la siguiente relación:

$$zC_{k+2}(z) = \frac{1}{k!} - C_k(z), \qquad k = 0, 1, 2, \ldots \tag{3.6}$$

Combinando la ecuación (3.6) con las ecuaciones (3.4) y (3.5) podemos obtener $C_2(z)$ y $C_3(z)$, que son los que, como veremos más adelante, utilizaremos para escribir los coeficientes de Lagrange en función de las variables universales. De esta manera quedaría:

$$C_2(z) = \begin{cases} \dfrac{1 - \cos\sqrt{z}}{z}, & z > 0 \\ 1/2, & z = 0 \\ \dfrac{\cosh\sqrt{-z} - 1}{-z}, & z < 0 \end{cases} \tag{3.7}$$

$$C_3(z) = \begin{cases} \dfrac{\sqrt{z} - \sin\sqrt{z}}{z^{3/2}}, & z > 0 \\ 1/6, & z = 0 \\ \dfrac{\sinh\sqrt{-z} - \sqrt{-z}}{(-z)^{3/2}}, & z < 0 \end{cases} \tag{3.8}$$

Es importante destacar que el valor de z permite identificar la naturaleza de la órbita, siendo ésta elíptica para valores de $z > 0$, parabólica para $z = 0$ e hiperbólica para $z < 0$.

Para la resolución del problema de Lambert necesitamos dos vectores de posición. Más adelante veremos que, para que este problema sea más realista, estos vectores de posición se darán en unas coordenadas específicas con origen en una estación de tierra. En la resolución del problema de Lambert que veremos en este capítulo vamos a suponer que los vectores $\boldsymbol{r_1}$ y $\boldsymbol{r_2}$ están referenciados en el Sistema Geocéntrico Ecuatorial, sistema el cual tomaremos como *inercial* a pesar de que no lo sea.

La resolución comienza con hallar el cambio de anomalía verdadera entre los tiempos t_1 y t_2, es decir, $\Delta\theta$.

$$\Delta\theta = \cos^{-1}\frac{\boldsymbol{r_1} \cdot \boldsymbol{r_2}}{r_1 r_2} \tag{3.9}$$

Puede verse que surge una ambigüedad la cual podríamos resolver fijándonos en la componente z de $\boldsymbol{h}$. Sin embargo, como no conocemos la velocidad en ningún punto, no podemos hallar el momento angular específico, con lo cual hemos de valernos de otros métodos para resolver la ambigüedad. Puesto que $\boldsymbol{r_1} \times \boldsymbol{r_2}$ es paralelo a $\boldsymbol{h}$ podemos utilizarlo para resolver la ambigüedad, no obstante, este producto vectorial no tiene por qué tener el mismo signo que $\boldsymbol{h}$. Es por ello que tenemos que deducirlo a partir de si la órbita es *posígrada*, $0° \leq i \leq 90°$, o *retrógrada*, $90° < i \leq 180°$. Para el caso de órbitas posígradas tenemos:

$$\Delta\theta = \begin{cases} 360 - \cos^{-1}\dfrac{\boldsymbol{r_1}\cdot\boldsymbol{r_2}}{r_1 r_2}, & (\boldsymbol{r_1}\times\boldsymbol{r_2})_z < 0 \\ \cos^{-1}\dfrac{\boldsymbol{r_1}\cdot\boldsymbol{r_2}}{r_1 r_2}, & (\boldsymbol{r_1}\times\boldsymbol{r_2})_z \geq 0 \end{cases} \tag{3.10}$$

En cambio, para órbitas retrógradas tenemos:

$$\Delta\theta = \begin{cases} 360 - \cos^{-1}\dfrac{\boldsymbol{r_1}\cdot\boldsymbol{r_2}}{r_1 r_2}, & (\boldsymbol{r_1}\times\boldsymbol{r_2})_z > 0 \\ \cos^{-1}\dfrac{\boldsymbol{r_1}\cdot\boldsymbol{r_2}}{r_1 r_2}, & (\boldsymbol{r_1}\times\boldsymbol{r_2})_z \leq 0 \end{cases} \tag{3.11}$$

Es necesario en este punto hacer uso de las expresiones de los ya mencionados coeficientes de Lagrange en función de la anomalía universal:

$$f = 1 - \frac{\chi^2}{r_1} C_2(z) \tag{3.12}$$

$$g = \Delta t - \frac{1}{\sqrt{\mu}} \chi^3 C_3(z) \tag{3.13}$$

$$\dot{f} = \frac{\sqrt{\mu}}{r_1 r_2} \chi [z C_3(z) - 1] \tag{3.14}$$

$$\dot{g} = 1 - \frac{\chi^2}{r_2} C_2(z) \tag{3.15}$$

Igualando estas expresiones con las ecuaciones (2.22), (2.23), (2.24) y (2.25) se obtiene un sistema de 4 ecuaciones con tres incógnitas, h, χ y z, donde $\Delta\theta$, Δt, r_1 y r_2 son valores conocidos. Sin embargo, sirviéndonos de la propiedad anteriormente descrita $f\dot{g} - \dot{f}g = 1$ podemos observar que de estas 4 ecuaciones realmente solo hay 3 ecuaciones independientes. El sistema, y por ende los coeficientes, queda entonces determinado.

Si igualamos las expresiones de f, es decir las ecuaciones (2.22) y (3.12) obtenemos la siguiente relación

$$1 - \frac{\chi^2}{r_1} C_2(z) = 1 - \frac{\mu r}{h^2}(1 - \cos\Delta\theta) \tag{3.16}$$

Despejando para h nos queda:

$$h = \sqrt{\frac{\mu r_1 r_2 (1 - \cos\Delta\theta)}{\chi^2 C_2(z)}} \tag{3.17}$$

Por otro lado, igualando las expresiones (2.23) y (3.13) se obtiene una ecuación que relaciona la diferencia de anomalía verdadera con el tiempo de vuelo:

$$\Delta t - \frac{1}{\sqrt{\mu}}\chi^3 C_3(z) = \frac{rr_0}{h}\sin\Delta\theta \tag{3.18}$$

Introduciendo la expresión (3.17) en la (3.18) nos queda:

$$\sqrt{\mu}\Delta t = \chi^3 C_3(z) + \sin\Delta\theta\,\chi\sqrt{\frac{r_1 r_2 C_2(z)}{1-\cos\Delta\theta}} \tag{3.19}$$

Se define ahora un parámetro, A, útil para simplificar la expresión, que será un valor conocido y estará en función de $\Delta\theta$, r_1 y r_2:

$$A = \sin\Delta\theta\sqrt{\frac{r_1 r_2}{1-\cos\Delta\theta}} \tag{3.20}$$

Puesto que la relación $z = \chi^2/a$ no podemos usarla porque a es el semieje mayor de la órbita que desconocemos, se debe usar la relación obtenida al igualar las expresiones para $\dot{f}$, es decir las ecuaciones (3.14) y (2.24).

$$\frac{\sqrt{\mu}}{r_1 r_2}\chi[zC_3(z)-1] = \frac{\mu}{h}\frac{1-\cos\Delta\theta}{\sin\Delta\theta}\left[\frac{\mu}{h^2}(1-\cos\Delta\theta) - \frac{1}{r_0} - \frac{1}{r}\right] \tag{3.21}$$

Operando con esta ecuación e introduciendo las ecuaciones (3.17) y (3.20) se obtiene<.

$$\chi^2 C_2(z) = r_1 + r_2 + A\frac{zC_3(z)-1}{\sqrt{C_2(z)}} \tag{3.22}$$

Despejando para χ se llega a una expresión donde se introducirá la siguiente definición para la expresión a la derecha del igual:

$$y(z) = r_1 + r_2 + A\frac{zC_3(z)-1}{\sqrt{C_2(z)}} \tag{3.23}$$

De esta manera obtenemos:

$$\chi = \sqrt{\frac{y(z)}{C_2(z)}} \tag{3.24}$$

Introduciendo ahora las expresiones (3.20) y (3.24) en la ecuación (3.19) llegamos a una igualdad que nos permitirá hallar el valor de z.

$$\sqrt{\mu}\,\Delta t = \left(\frac{y(z)}{C_2(z)}\right)^{\frac{3}{2}} C_3(z) + A\sqrt{y(z)} \tag{3.25}$$

Puesto que esta expresión supone difícil despejarla para hallar el valor de z se resuelve por el método iterativo de *Newton-Raphson* para la función $F(z)$.

$$F(z) = \left(\frac{y(z)}{C_2(z)}\right)^{\frac{3}{2}} C_3(z) + A\sqrt{y(z)} - \sqrt{\mu}\Delta t \tag{3.26}$$

Este proceso básicamente halla de forma iterativa los valores de z a partir de un valor inicial z_0 y hallando valores progresivamente siguiendo la siguiente regla:

$$z_{k+1} = z_k - \frac{F(z_k)}{F'(z_k)}$$

Donde $F'(z) = \frac{dF(z)}{dz}$.

$$\begin{matrix} \left(\frac{y(z)}{C_2(z)}\right)^{\frac{3}{2}}\left[\frac{1}{2z}\left(C_2(z) - \frac{3}{2}\frac{C_3(z)}{C_2(z)}\right) + \frac{3}{4}\frac{C_3^2(z)}{C_2(z)}\right] + \frac{A}{8}\left[\frac{3C_3(z)\sqrt{y(z)}}{C_2(z)} + A\sqrt{\frac{C_2(z)}{y(z)}}\right] & si\ z \neq 0 \\ \frac{\sqrt{2}}{40}\,y(0)^{\frac{3}{2}} + \frac{A}{8}\left[\sqrt{y(0)} + \frac{A}{\sqrt{2y(0)}}\right] & si\ z = 0 \end{matrix} \tag{3.27}$$

Esta definición es obtenida teniendo en cuenta que

$$\lim_{z\to 0}\frac{1}{2z}\left(C_2(z) - \frac{3}{2}\frac{C_3(z)}{C_2(z)}\right) = 0$$

Es conocido que una buena aproximación para el valor z_0 sería $z_0 = E^2$ para órbitas elípticas y $z_0 = -F^2$ para el caso de órbitas hiperbólicas. Sin embargo, al no conocer a priori la forma de la órbita no se puede utilizar esta aproximación. Es por ello, por el hecho de que estamos buscando la primera raíz de $F(z)$ y por la forma creciente de F que se comenzará con un valor *suficientemente* bajo para z_0 y se calculará $F(z_0)$ de manera que se vaya incrementando el valor de z_0 mientras $F(z_0) < 0$. El valor seleccionado será el que haga que el signo pase de negativo a positivo. Esto nos proporcionará un valor inicial bastante aproximado a la solución final, $z = z_{sol}$.

Una vez obtenido el valor $z = z_{sol}$ el siguiente paso es calcular los coeficientes de Lagrange para este valor de z. Para ello debemos sustituir (3.24) y (3.25) en las expresiones (3.12), (3.13), (3.14) y (3.15).

$$f = 1 - \frac{y(z)}{r_1} \tag{3.28}$$

$$g = A\sqrt{\frac{y(z)}{\mu}} \tag{3.29}$$

$$\dot{f} = \frac{\sqrt{\mu}}{r_1 r_2}\sqrt{\frac{y(z)}{C_2(z)}}[zC_3(z) - 1] \tag{3.30}$$

$$\dot{g} = 1 - \frac{y(z)}{r_2} \tag{3.31}$$

Con estos valores ya podemos calcular las velocidades en los dos puntos de observación, $\boldsymbol{r_1}$ y $\boldsymbol{r_2}$. Sustituyendo la propiedad anteriormente vista $f\dot{g} - \dot{f}g = 1$ en (2.20) y (2.21), y aplicando estas ecuaciones para $\boldsymbol{r_1}$ y $\boldsymbol{r_2}$, podemos despejar para hallar $\boldsymbol{v_1}$ y $\boldsymbol{v_2}$.

$$\boldsymbol{v_1} = \frac{1}{g}(\boldsymbol{r_2} - f\boldsymbol{r_1}) \tag{3.32}$$

$$\boldsymbol{v_2} = \frac{1}{g}(\dot{g}\boldsymbol{r_2} - \boldsymbol{r_1}) \tag{3.33}$$

De esta manera ya quedaría resuelto el problema de Lambert. Es menester especificar que, siempre y cuando se proporcione si la órbita es retrógrada o posígrada, la solución es única. Si esto no estuviera especificado podría viajarse de $\boldsymbol{r_1}$ a $\boldsymbol{r_2}$ en Δt siguiendo dos trayectorias completamente diferentes, siendo una de ellas retrógrada y otra posígrada. Es por tanto que se parámetro será un dato a pedir en el algoritmo de resolución que veremos en el siguiente capítulo.

Capítulo 4. Algoritmo de resolución

En este capítulo desarrollaremos el algoritmo implantado en *MATLAB* para llevar a cabo la resolución del problema de Lambert planteada en el capítulo anterior. Este algoritmo tiene tres bloques destacados, el bloque de introducción de datos, el bloque de cálculos numéricos y almacenamiento de datos, y, por último, el bloque de gráficos.

El programa dispondrá de la capacidad de resolver N problemas de Lambert, de almacenar los resultados para el posterior uso y de mostrar todas las órbitas juntas en un gráfico $3D$ así como mostrar cada órbita por separado en un gráfico $2D$ para cada una.

Un diagrama de flujo general de éste algoritmo es el siguiente.

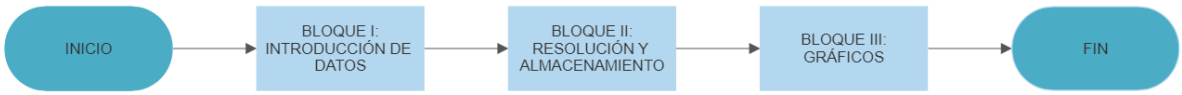


Ilustración 8. Diagrama de flujo general del algoritmo implementado en MATLAB.

4.1 Bloque I: Introducción de datos

Este bloque comienza con unos datos fijos que se le proporciona al programa, como por ejemplo los valores del radio terrestre polar, R_p, y ecuatorial, R_e, o el valor de μ. Tras esto, el programa pide la introducción de varios valores generales del problema, estos son el número de problemas a resolver, N, la precisión de la órbita en número de puntos y un parámetro entre 0 y 100 para la opacidad en la representación de la tierra en el gráfico $3D$.

Tras esto comienza un bucle *for* entre 1 y N en el que se piden, para cada uno de los problemas a resolver, un conjunto de datos relacionados tanto con la Estación de Observación como con el *objeto orbitante*.

Previo a la especificación de los datos a introducir cabría destacar la comprobación de datos implementada para evitar errores durante la introducción de datos, evitando así errores durante la ejecución del algoritmo. Para todas las variables se hará una comprobación del tipo de variable introducida con el tipo de variable esperada, pudiéndose almacenar ésta variable únicamente en el caso de que lo esperado coincida con lo introducido. Por ejemplo, en el caso de un variable numérica se comprobará tanto que ésta sea de tipo numérico como que las dimensiones sean las adecuadas y estén dentro de un intervalo. En casos en los que se esperen cadenas de caracteres el programa únicamente aceptará como válidos los datos que coincidan con las opciones posibles. En el caso de que la variable no esté correctamente introducida no se avanzará en el algoritmo y se pedirá que se vuelva a introducir.

4.1.1 Datos de la estación de observación

El algoritmo implementado permite la introducción de la posición de la estación de observación, GS, respecto el *Sistema Geográfico,* "1" mediante las clásicas coordenadas cartesianas (x_{GS}, y_{GS}, z_{GS}) o mediante las coordenadas geodéticas $(\phi_{GS}, \lambda_{GS}, H_{GS})$, donde ϕ_{GS} es la latitud de la estación, λ_{GS} la longitud y H_{GS} la altitud. Conviene recordar que el Sistema Geográfico tiene como origen el centro de la tierra, como eje $0X$ el vector contenido en el plano ecuatorial que pasa por el *Meridiano de Greenwich*, OZ perpendicular al plano ecuatorial y OY cerrando el triedro. Las coordenadas geodéticas permiten la conversión a coordenadas cartesianas al suponer la tierra como un elipsoide de revolución, sin embargo, estos cálculos los veremos más adelante.

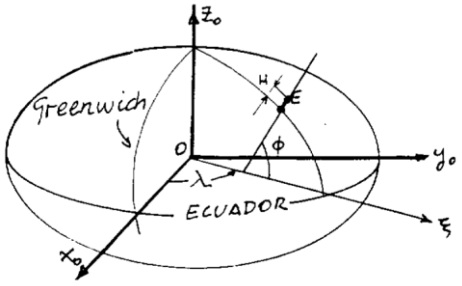


Ilustración 9. Sistema Geográfico con coordenadas geodéticas

Entonces, para cada problema tendremos un parámetro que nos indicara el tipo de coordenada que se está introduciendo y un grupo de tres coordenadas que bien serán cartesianas, en km, o bien serán geodéticas, como vectores $[grados, minutos, segundos]$.

4.1.2 Datos del objeto orbitante

En este caso se pedirá la introducción de la posición del objeto orbitante relativa a la estación de observación referenciándola respecto al *Sistema Topocéntrico,* "2". Éste tiene como origen la posición de la Estación de Observación, y contiene los ejes OX y OY en el plano de *Horizonte Local*, definido como el plano tangente al elipsoide terrestre en el punto O. El eje OX apunta hacia el Este, el OY hacia el norte, y el OZ, perpendicular al plano del horizonte local, apunta hacia el *cénit*, es decir, hacia "arriba".

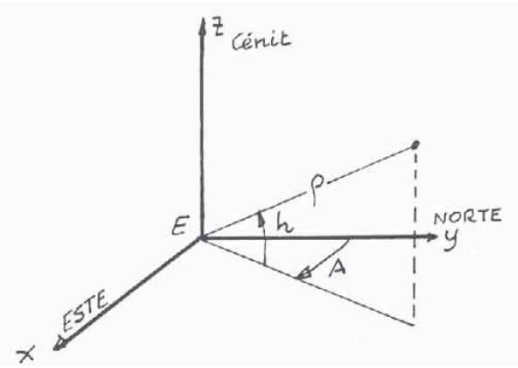


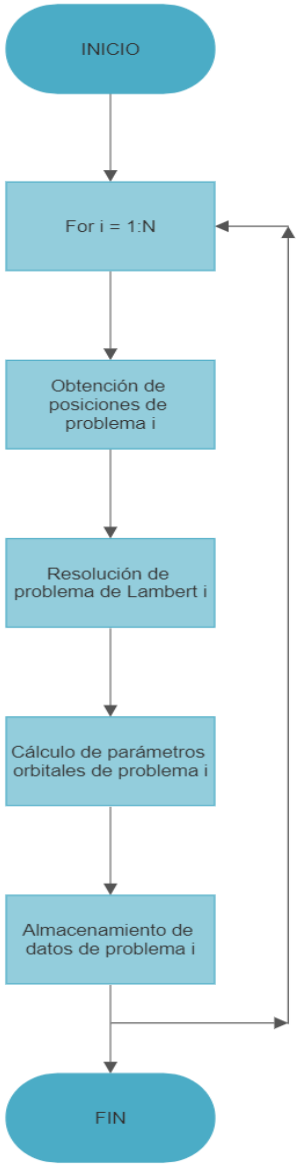
Ilustración 10. Sistema Topocéntrico

Las coordenadas que se utilizan son (ρ, A, h), donde ρ es la distancia del objeto orbitante en km, A es el azimut en grados, $A \in [0, 360)$ y h la elevación respecto al plano de horizonte local, también en grados $h \in [0, 90]$. Es importante recordar que el problema de Lambert necesita de dos observaciones y de una diferencia de tiempo, es por tanto que habrá dos grupos de tres coordenadas, acompañadas de una fecha de observación para cada una de ellas. La fecha de observación tendrá el formato "$DD/MM/YYYY\ HH{:}MM{:}SS$". Esto es así puesto que *MATLAB* podrá interpretarlo como una fecha y operar con él siempre y cuando el formato sea el anterior, lo cual, como veremos más adelante, nos será de mucha utilidad.

Otro dato adicional que se pide para que, como se explicó anteriormente, haya una única solución, es una cadena de caracteres que indicará si la órbita es retrógrada o posígrada.

4.2 Bloque II: Resolución y almacenamiento

En este bloque se encuentran todas las funciones utilizadas para resolver el problema de Lambert. Aunque la función principal consiste en la implementación de la resolución vista en el capítulo 3, *SolveLambert.m*, existe una serie de funciones complementarias. En la ilustración 10, a la derecha, se describe mediante un diagrama de flujo el bucle *for* que se lleva a cabo para resolver los problemas uno a uno, y las funciones que tienen lugar dentro del mismo.



4.2.1 Obtención de posiciones

La primera de estas funciones se utiliza para transformar los datos que se piden en el Bloque I del algoritmo en los datos de entrada de la función *SolveLambert.m*. Básicamente se trata de expresar las posiciones de la estación de observación y del objeto observado en un sistema de de referencia geocéntrico ecuatorial para luego sumarlas y obtener la posición del objeto desde el origen de este sistema, es decir el centro del elipsoide terrestre, foco orbital desde el cual se lleva a cabo la acción gravitacional supuesta en el problema de Lambert. Puesto que los datos de la estación de tierra se pueden introducir de dos formas diferentes habrá dos tipos de funciones de posición en función del tipo de coordenada utilizada. Habrá un parámetro, *Modo_GS*, el cual se pide como dato de entrada y que indica si las coordenadas a usar para la estación son geodéticas $(Modo\_GS = 0)$ o cartesianas $(Modo\_GS = 1)$.

El caso más sencillo corresponde a la introducción de la posición en coordenadas cartesianas. Ésta función, *posicion_Modo_GS_1.m*, empieza por la obtención de GST_0

Para ello ha de operarse con la fecha de observación del objeto de la siguiente manera. Primero se ha de usar la siguiente fórmula para hallar de forma precisa el día juliano de observación a las $00{:}\,00$, J_0.

$$J_0 = 367A - \left\lfloor \frac{7A + 7\left\lfloor \frac{M+9}{12} \right\rfloor}{4} \right\rfloor + \left\lfloor \frac{275M}{9} \right\rfloor + D + 1721013.5$$

En ésta fórmula A representa el año de observación, M el mes, y D el día. No olvidar que el símbolo $\lfloor x \rfloor$ indica que debe tomarse únicamente la parte entera de x. Esta función está implementada en *MATLAB* y se llama *floor*.

Con este dato podemos hallar, mediante la siguiente fórmula, la *centuria juliana,* T_0.

$$T_0 = \frac{J_0 - 2451545}{36525}$$

Esto nos permite hallar el tiempo sidéreo a las $00{:}\,00$, es decir GST_0, mediante la siguiente fórmula. Conviene recordar que el resultado de esta fórmula es en grados, y expresa la longitud del meridiano de Greenwich con respecto al primer punto de Aries a las $00{:}\,00$ de la época de observación.

$$GST_0 = 100.4606184 + 36000.77004\, T_0 - 0.000387933\, T_0^2 - 2.583 \times 10^{-8}\, T_0^3$$

Con éste parámetro podemos hallar de forma sencilla el valor de GST, fundamental para poder referenciar nuestros datos de entrada iniciales en el sistema geocéntrico ecuatorial. Hemos de utilizar una fórmula de propagación para la rotación de la tierra. Esto no es más que una simple regla de tres cuyo punto inicial es GST_0. Ha de calcularse la hora del día en la que se observa el objeto, $UT = HH + \frac{MM}{60} + \frac{SS}{3600}$, con HH representando las horas, MM los minutos, y SS los segundos.

$$GST = GST_0 + 360.98564724 * \frac{UT}{24}$$

Es importante indicar que $GST \in [0, 360)$ y que cada observación j tendrá un GST_j asociado.

Tras esto, podemos hacer una matriz de giro que permita obtener la posición de la estación $\boldsymbol{r}_{GS}$ en el sistema geocéntrico ecuatorial, 0, en función de la posición de la estación en el sistema geográfico, 1.

$$\boldsymbol{r}_{GS,j} = \begin{pmatrix} x_{GS,j} \\ y_{GS,j} \\ z_{GS,j} \end{pmatrix}_0 = \begin{pmatrix} \cos GST_j & -\sin GST_j & 0 \\ \sin GST_j & \cos GST_j & 0 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_{GS} \\ y_{GS} \\ z_{GS} \end{pmatrix}_1$$

El siguiente paso es transformar el vector de posición del objeto del sistema Topocéntrico al sistema geocéntrico ecuatorial. Para esto necesitamos tener en cuenta el origen del sistema Topocéntrico, es decir, nuestra estación de observación. Primeramente se halla el plano tangente al elipsoide en la estación de observación, para hallar así la base

$(\boldsymbol{\epsilon_1}, \boldsymbol{\epsilon_2}, \boldsymbol{\epsilon_3})$ de nuestro sistema Topocéntrico en "1". El vector $\boldsymbol{\epsilon_3}$ no es más que el vector normal a la superficie del elipsoide. Esto es el gradiente de la superficie $f(x, y, z) = 0$ del elipsoide en el punto $(x_{GS}, y_{GS}, z_{GS})_1$. Puesto que $f(x, y, z) = \frac{x^2}{R_e} + \frac{y^2}{R_e} + \frac{z^2}{R_p} - 1$, tenemos:

$$\boldsymbol{\epsilon_3} = \frac{\boldsymbol{\nabla} f(x_{GS}, y_{GS}, z_{GS})_1}{\|\boldsymbol{\nabla} f(x_{GS}, y_{GS}, z_{GS})_1\|} = \begin{pmatrix} 2x_{GS}/R_e^2 \\ 2y_{GS}/R_e^2 \\ 2z_{GS}/R_p^2 \end{pmatrix} * \frac{1}{\sqrt{(2x_{GS}/R_e^2)^2 + (2y_{GS}/R_e^2)^2 + \left(2z_{GS}/R_p^2\right)^2}}$$

El vector $\boldsymbol{\epsilon_1}$ lo hallamos mediante la operación $\boldsymbol{\epsilon_1} = \boldsymbol{OZ} \times \boldsymbol{\epsilon_3}$, siendo $\boldsymbol{OZ}$ el tercer vector de la base del sistema geocéntrico ecuatorial.

Para completar el triedro, $\boldsymbol{\epsilon_2}$ lo hallamos mediante la fórmula $\boldsymbol{\epsilon_2} = \boldsymbol{\epsilon_3} \times \boldsymbol{\epsilon_1}$.

Con esto ya podemos establecer la matriz de giro entre el sistema Topocéntrico, 2, y el sistema geográfico, 1. Observar que esta base está escrita en coordenadas cartesianas del sistema geográfico.

Para expresar el vector de posición del objeto observado en el sistema geográfico tenemos entonces que multiplicar las coordenadas del objeto expresadas en la base del sistema Topocéntrico por las dos matrices de giro recién obtenidas. Sin embargo, previo a este paso, hay que transformar las coordenadas topográficas de la observación j, $(\rho_j, A_j, h_j)_2$, en coordenadas cartesianas $(x_j, y_j, z_j)_2$. Para ello utilizamos la siguiente fórmula:

$$x_j = \rho_j \cos h_j \sin A_j$$

$$y_j = \rho_j \cos h_j \cos A_j$$

$$y_j = \rho_j \sin A_j$$

Recordar que el subíndice i representa la observación realizada, y puede ser la observación número $j = 1$ o la observación $j = 2$.

Con esto ya podemos obtener la posición del objeto visto desde la estación escrito en coordenadas cartesianas y referenciado al sistema geocéntrico ecuatorial, $(x_j, y_j, z_j)_0$. Expresando los vectores de la base $(\boldsymbol{\epsilon_1}, \boldsymbol{\epsilon_2}, \boldsymbol{\epsilon_3})$ como vectores columna, podemos escribir:

$$\begin{pmatrix} x_j \\ y_j \\ z_j \end{pmatrix}_0 = \begin{pmatrix} \cos GST_j & -\sin GST_j & 0 \\ \sin GST_j & \cos GST_j & 0 \\ 0 & 0 & 1 \end{pmatrix} (\boldsymbol{\epsilon_1} \mid \boldsymbol{\epsilon_2} \mid \boldsymbol{\epsilon_3}) \begin{pmatrix} x_j \\ y_j \\ z_j \end{pmatrix}_2$$

Por tanto ya podemos obtener la posición del objeto del problema i en la observación j escrito en coordenadas cartesianas y referenciado al sistema geocéntrico ecuatorial, con origen en el centro de la tierra, sistema el cual consideraremos inercial y será utilizado para resolver el problema de Lambert.

$$\begin{pmatrix} X_j \\ Y_j \\ Z_j \end{pmatrix}_0 = \begin{pmatrix} x_{GS,j} \\ y_{GS,j} \\ z_{GS,j} \end{pmatrix}_0 + \begin{pmatrix} x_j \\ y_j \\ z_j \end{pmatrix}_0$$

No olvidar que el índice i, indicativo del problema, se ha omitido por simplicidad.

En el caso de que la posición de la estación de observación se haya proporcionado en coordenadas geodéticas la función de obtención de posiciones, *posicion_Modo_GS_0.m*, tendrá una diferencia con la anterior, y es que deberá de hacerse una transformación de coordenadas, de geodéticas a cartesianas, durante la obtención de la posición de la estación de tierra. Es decir, habrá que calcular $(x_{GS}, y_{GS}, z_{GS})_1$ a partir de los datos $(\phi_{GS}, \lambda_{GS}, H_{GS})$ mediante las siguientes fórmulas:

$$x_{GS} = \left(H_{GS} + \frac{R_e}{\sqrt{1 - f(2-f)\sin^2\phi_{GS}}}\right)\cos\phi_{GS}\cos\lambda_{GS}$$

$$y_{GS} = \left(H_{GS} + \frac{R_e}{\sqrt{1 - f(2-f)\sin^2\phi_{GS}}}\right)\cos\phi_{GS}\sin\lambda_{GS}$$

$$z_{GS} = \left(H_{GS} + \frac{R_e(1-f)^2}{\sqrt{1 - f(2-f)\sin^2\phi_{GS}}}\right)\sin\phi_{GS}$$

Donde $f = \frac{R_e - R_p}{R_e}$ y representa el aplanamiento de la tierra.

Con esto ya queda determinada las posiciones 1 y 2 del objeto observado desde la estación expresadas de manera que puedan ser utilizadas por la función *SolveLambert.m*.

4.2.2 Resolución del problema de Lambert

A pesar de que en el capítulo 3.2 ya se ha explicado con detalle el procedimiento a seguir, en este capítulo expondremos con mayor sencillez el algoritmo implementado para la resolución del problema de Lambert.

Los datos de entrada para el problema son los parámetros físicos específicos del problema i, los vectores $\boldsymbol{r_{i1}} = (X_{i1}, Y_{i1}, Z_{i1})_0$ y $\boldsymbol{r_{i2}} = (X_{i2}, Y_{i2}, Z_{i2})_0$, la diferencia de tiempo entre observaciones, Δt_i, y el parámetro que determina si la órbita es retrógrada o posígrada; los parámetros físicos comunes, en este caso μ; y los parámetros específicos del método de resolución en sí, es decir la tolerancia en la resolución y el número de iteraciones máximas, también generales para todos los N problemas.

Los datos de salida serán las velocidades $\boldsymbol{v_{i1}} = \left(v_{x_{i1}}, v_{y_{i1}}, v_{z_{i1}}\right)_0$ y $\boldsymbol{v_{i2}} = \left(v_{x_{i2}}, v_{y_{i2}}, v_{z_{i2}}\right)_0$ y el error cometido en la resolución, calculado como el valor absoluto de la función $F(z)$, $err = |F(z)|$.

El algoritmo, en primer lugar, calcula los siguientes valores

$$r_{ij} = \sqrt{X_{ij}^2 + Y_{ij}^2 + Z_{ij}^2} \quad , \quad i \in [1, N], \quad j \in [1, 2]$$

$$\boldsymbol{r_{i1}} \times \boldsymbol{r_{i2}}$$

Después calcula $\Delta\theta$. Para ello hace uso de las fórmulas (3.10) y (3.11).

El siguiente paso es calcular la constante A, mediante la fórmula (3.20).

Después se hallará el valor z_0. Para esto se comienza con un valor $z_0 = -100$ y se calcula $F(z_0)$ de manera que si ésta es negativa se le suma 1 al valor de z_0 y se vuelve a calcular $F(z_0)$. El proceso continuará hasta que $F(z_0) > 0$. El siguiente paso es la resolución del problema de Lambert mediante el método Netwon-Raphson para la función $F(z)$. Tanto para este paso como para el anterior es necesario el cálculo de $F(z)$, lo cual pasa por calcular las funciones de Stumpff, ecuaciones (3.7) y (3.8), y el parámetro $y(z)$, ecuación (3.23).

El método de Newton-Raphson, ya explicado con anterioridad, se llevará a cabo hasta que, o bien $err < tol$, o bien el número de iteraciones sea igual al número de iteraciones máxima.

Tras obtener el valor z_{sol} que verifica $F(z_{sol}) = 0$ se procede al cálculo de los coeficientes de Lagrange, ecuaciones (3.28), (3.29) y (3.31), para finalmente calcular las velocidades $\boldsymbol{v_{1n}}$ y $\boldsymbol{v_{2n}}$ mediante las ecuaciones (3.32) y (3.33).

Un diagrama de flujo del algoritmo es el siguiente:

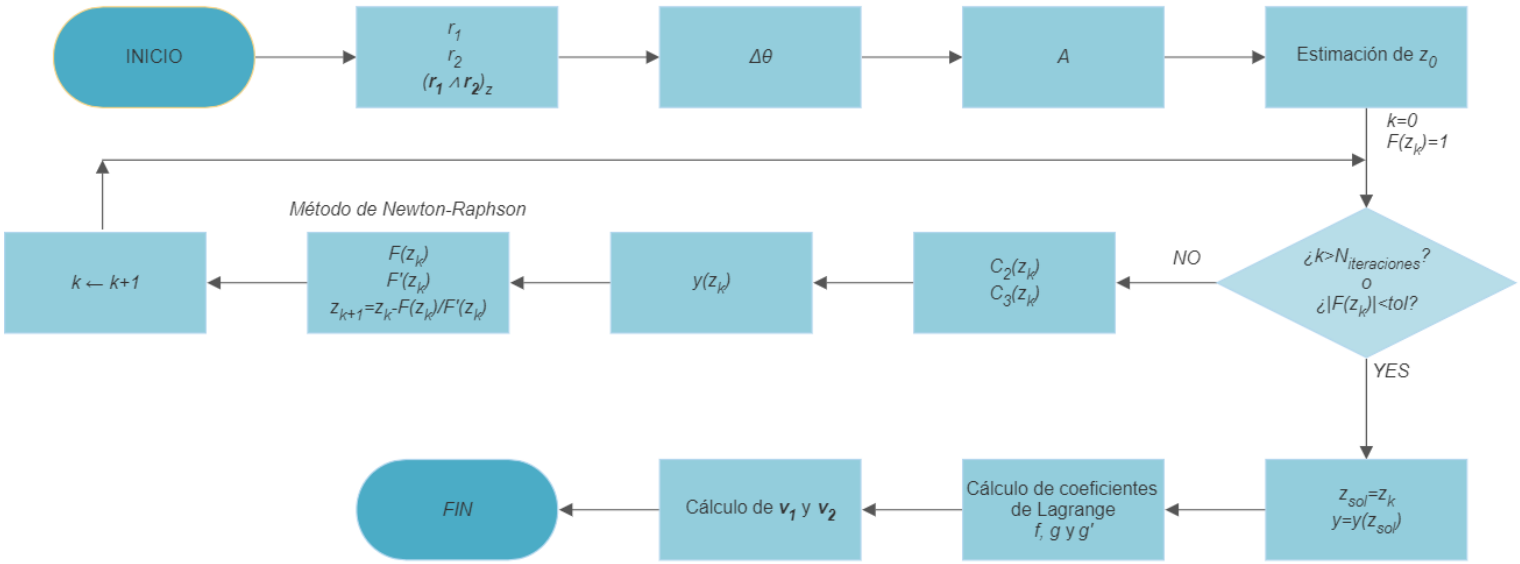


Ilustración 11. Diagrama de flujo de la función SolveLambert.m.

4.2.3 Cálculo de parámetros orbitales

En este capítulo vamos a desarrollar en detalle la función *Parametros_orbitales.m*. Ésta es una función que calcula, para cada problema i, los parámetros orbitales a partir de los valores de μ, $\boldsymbol{r_{ij}}$ y $\boldsymbol{v_{ij}}$, con $i \in [1, N]$ y $j \in [1, 2]$. Los datos de salida son $p_i, \boldsymbol{h}_i, e_i, a_i, T_i, \theta_{ij}, r_{p,i}$, $r_{a,i}, \Omega_i, i_i, \omega_i, u_{ij}, \overline{\omega}_i, \lambda_{T,ij}, \boldsymbol{M}_{giro,i}$ y $\boldsymbol{\theta}_{graficos,i}$. Algunos de estos parámetros requieren una explicación con más detalle. Los parámetros T_i y $r_{a,i}$ sólo se proporcionan en el caso de que la órbita sea cerrada, en los demás casos se darán como NaN. Los parámetros $\theta_{ij}, \Omega_i, \omega_i, u_{ij}, \overline{\omega}_i$ y $\lambda_{T,ij}$ se darán o no dependiendo de los valores de e_i y i_i, tal y como se explicó en el apartado de casos especiales del capítulo 2.3.2.

Los parámetros $\boldsymbol{M}_{giro,i}$ y $\boldsymbol{\theta}_{graficos,i}$ suponen parámetros de utilidad calculados para su uso durante los gráficos. En concreto la matriz $\boldsymbol{M}_{giro,i}$ es la matriz que se usa para poder representar la órbita en el plano orbital en el gráfico $2D$. Por su parte, el vector $\boldsymbol{\theta}_{graficos,i}$ proporciona valores máximos y mínimos de θ para que las gráficas, en el caso de que las órbitas sean abiertas, puedan verse a una escala *razonable*.

Una particularidad de la función es que, a pesar de que teóricamente sólo se necesitaría una posición y la velocidad en esa posición para el cálculo de los parámetros orbitales, el programa toma como datos de entrada dos posiciones y dos velocidades. Esto es así porque para el cálculo de los vectores a partir de los cuales se calcula el resto de parámetros, $\boldsymbol{h}_i$ y $\boldsymbol{e}_i$, se utilizan ambas parejas de posición y velocidad, $j = 1$ y $j = 2$, calculándose por tanto dos valores de $\boldsymbol{h}_i$ y otros dos de $\boldsymbol{e}_i$ y luego se hace una media, para minimizar el error cometido.

Esto supondría la primera parte del proceso de cálculo de parámetros orbitales. Se completaría, para cada problema i, el triedro que define Sistema de Referencia Orbital, "3", como $\boldsymbol{\epsilon_{2i}} = \boldsymbol{\epsilon_{3i}} \times \boldsymbol{\epsilon_{1i}}$, siendo $\boldsymbol{\epsilon_{1i}} = \boldsymbol{e}_i/\|\boldsymbol{e}_i\|$ y $\boldsymbol{\epsilon_{3i}} = \boldsymbol{h}_i/\|\boldsymbol{h}_i\|$. Este sistema tendría el foco en el elipsoide terrestre, primera dirección apuntando al periastro y la tercera dirección normal al plano orbital. Como matriz de giro de este sistema al sistema geocéntrico ecuatorial tendríamos:

$$\boldsymbol{M}_{giro,i} = [\boldsymbol{\epsilon_{1i}} \,|\boldsymbol{\epsilon_{2i}}\,|\,\boldsymbol{\epsilon_{3i}}]; \qquad \boldsymbol{r}_{i,2D} = \boldsymbol{r}_{i,3} = \boldsymbol{M}_{giro,i}^{-1}\boldsymbol{r}_{i,0} \tag{4.1}$$

Tras éstos cálculos se calculará p_i, i_i y $r_{p,i}$, luego se procederá a hacer la clasificación de la órbita y a calcular los parámetros correspondientes, los cuales explicaremos un poco más adelante cuales son. Tras esta primera clasificación viene otra clasificación que permitirá, además de especificar la matriz $\boldsymbol{M}_{giro,i}$ en casos límite, decidir qué parámetros (angulares) calcular dependiendo de si corresponde o no a uno de los casos especiales vistos en el apartado 2.3.2.1.

Un resumen de lo visto en ese apartado lo encontramos en la siguiente tabla.

$\boldsymbol{Caso}$	$\boldsymbol{i = 0°, 180°}$ $\boldsymbol{e = 0}$	$\boldsymbol{i = 0°, 180°}$ $\boldsymbol{e \neq 0}$	$\boldsymbol{i \neq 0°, 180°}$ $\boldsymbol{e = 0}$	$\boldsymbol{i \neq 0°, 180°}$ $\boldsymbol{e \neq 0}$
$Parámetros\ calculados$	λ_{ij}	$\bar{\omega}_i$ y θ_{ij}	Ω_i y u_{ij}	Ω_i, ω_i y θ_{ij}
$Variaciones\ en\ \boldsymbol{M}_{giro,i}$	$\boldsymbol{M}_{giro,i} = \boldsymbol{I}$*	–	$\boldsymbol{\epsilon_{1i}} = \boldsymbol{n_i}$**	–

*$\boldsymbol{I}$ corresponde a la Matriz Identidad. En el caso de que $i = 180°$ el valor $I_{33} = -1$.

** Esto afecta al valor de $\boldsymbol{\epsilon_{2i}}$ pero no a su método de cálculo, el cual deberá hacerse.

Tabla 2. Resumen de parámetros calculados en los casos especiales.

Al final de la función se ejecutan unos comandos que sacan por pantalla los datos del problema en cuestión. Un ejemplo se muestra a continuación, y, tras éste, una tabla resumen con los parámetros mostrados en función del tipo de órbita calculada. En el caso específico de hipérbolas se calcula tanto la deflexión de la órbita, δ_i, como el ángulo las asíntotas, θ_∞.

```
==========================================================================

RESULTADOS PARA EL PROBLEMA 1

Tipo de órbita: eliptica

Los parámetros orbitales, en km y °, son:

      a         e       Inclinacion      RAAN      Argumento_perigeo      r_p       r_a       T
    _____    _______    ___________    ______    _________________    ______    _____    _____

    11264    0.41981      136.17        228.99         88.055          6535.1    15992    11897

--------------------------------------------------------------------------

Vector posición [km] y velocidad [km/s] en el punto inicial (1) y final (2)

      r_1         v_1         r_2         v_2         h
    _______    _______    _______    ______    ______

    -1633.3     7.1106    -4412.3    5.1943    -31776
     4918.6     5.4492     1979.6    7.6132     27636
     4281.2    -1.7175     4442.6    1.0335    -43874

--------------------------------------------------------------------------

    anomalia_verdadera_1    anomalia_verdadera_2
    ____________________    ____________________

           25.064                  349.62

==========================================================================
```

Ilustración 12. Parámetros orbitales calculados con MATLAB y presentados por pantalla.

Tipo de órbita	$\boldsymbol{i \neq 0°, 180°}$	$\boldsymbol{i = 0°, 180°}$
Circular	$a_i, e_i, i_i, \Omega_i, u_{ij}, T_i, r_{p,i}, \boldsymbol{r}_{ij}, \boldsymbol{v}_{ij}$	$a_i, e_i, i_i, \lambda_{T,ij}, T_i, r_{p,i}, \boldsymbol{r}_{ij}, \boldsymbol{v}_{ij}$
Eliptica	$a_i, e_i, i_i, \Omega_i, \omega_i, \theta_{ij}, T_i, r_{p,i}, r_{a,i}, \boldsymbol{r}_{ij}, \boldsymbol{v}_{ij}$	$a_i, e_i, i_i, \bar{\omega}_i, \theta_{ij}, T_i, r_{p,i}, r_{a,i}, \boldsymbol{r}_{ij}, \boldsymbol{v}_{ij}$
Parabólica	$e_i, i_i, \Omega_i, \omega_i, \theta_{ij}, r_{p,i}, \boldsymbol{r}_{ij}, \boldsymbol{v}_{ij}$	$e_i, i_i, \bar{\omega}_i, \theta_{ij}, r_{p,i}, \boldsymbol{r}_{ij}, \boldsymbol{v}_{ij}$
Hiperbólica	$a_i, e_i, i_i, \Omega_i, \omega_i, \theta_{ij}, r_{p,i}, \boldsymbol{r}_{ij}, \boldsymbol{v}_{ij}, \delta_i, \theta_{\infty,i}$	$a_i, e_i, i_i, \bar{\omega}_i, \theta_{ij}, r_{p,i}, \boldsymbol{r}_{ij}, \boldsymbol{v}_{ij}, \delta_i, \theta_{\infty,i}$

Tabla 3. Parámetros mostrados por pantalla en función del tipo de órbita calculada.

.

4.2.4 Almacenamiento de datos

Al terminar este bloque se habrá resuelto el problema i, y previo al apartado gráfico se procede a almacenar los datos en matrices. Todos los vectores se referencian al sistema geocéntrico ecuatorial, "0", y utilizan coordenadas cartesianas. Los datos son. Básicamente los calculados durante el apartado anterior, sin embargo, no todos son almacenados en matrices. Esto se ha hecho así para que, dentro del bucle, al hacer los cálculos, se evite tener que poner índices a las variables. Los parámetros son los siguientes:

- $\boldsymbol{r}_{GS,ij}$
- $\boldsymbol{r}_{ij}$
- $\boldsymbol{v}_{ij}$
- $\boldsymbol{h}_i$
- p_i
- e_i
- a_i
- θ_{ij}
- $r_{p,i}$
- $r_{a,i}$
- Ω_i
- ω_i
- $\bar{\omega}_i$
- u_{ij}
- $\lambda_{T,ij}$
- $\boldsymbol{M}_{giro,i}$
- $\boldsymbol{GST}_{ij}$

4.3 Bloque III: Gráficos

En este apartado se especifica los pormenores relacionados con las gráficas que se presentan. Además, se incluirán imágenes para visualizar los resultados.

En cuanto a la gráfica $3D$ cabe especificar que se muestra un elipsoide de revolución con una imagen de la tierra y todas las órbitas calculadas. El instante de tiempo elegido para mostrar la gráfica es el instante de la primera observación del primer problema, por tanto, se ha rotado la tierra el valor GST_{11} de manera que para colocar los puntos de las estaciones de los problemas $i > 1$, $\boldsymbol{r}_{GS,i1}$, habrá que multiplicar estos vectores por la siguiente matriz.

$$M = \begin{pmatrix} \cos(GST_{11} - GST_{i1}) & -\sin(GST_{11} - GST_{i1}) & 0 \\ \sin(GST_{11} - GST_{i1}) & \cos(GST_{11} - GST_{i1}) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

De esta manera aparecerá la estación de observación de cada problema correctamente ubicada en el mapa.

Se ha elegido una imagen plana de la tierra en la que haya una parte iluminada por la luz solar. Se debe de rotar también el foco de luz la magnitud GST_{11} para que la zona iluminada coincida con la de la fotografía.

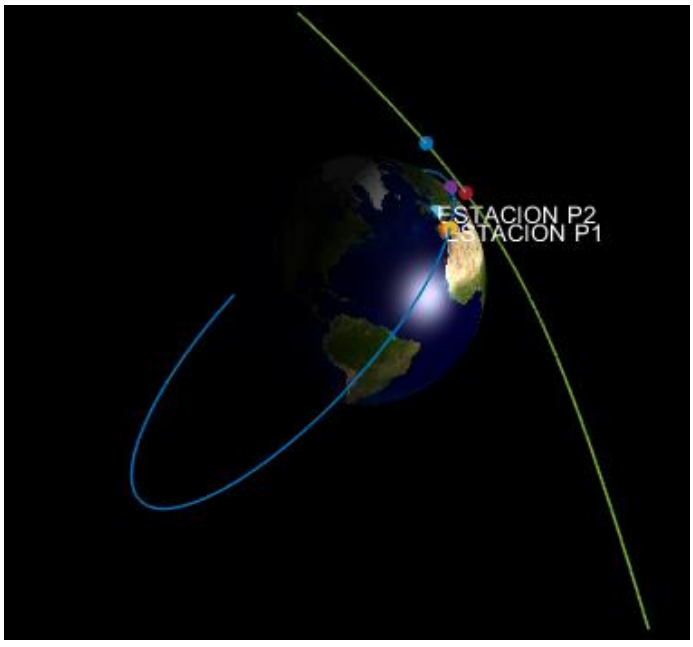


Ilustración 13. Gráfico 3D de una resolución de dos problemas

Una particularidad que tiene este algoritmo es que, al calcular los puntos de la órbita en $3D$ también ejecuta una comprobación sobre si los puntos calculados están en posiciones posibles para satélites, es decir, comprueba si estos puntos están fuera del elipsoide de revolución o si, de forma contraria, están dentro, correspondiéndose estas órbitas a trayectorias más propias de objetos balísticos. En éste último caso se dará un aviso, indicando la no viabilidad de la trayectoria para un satélite, y no se procederá a sacar la gráfica por pantalla.

```
--------------------------------------------------------------------
La solucion del problema 1 no es viable para un satelite
====================================================================
```

Ilustración 14. Aviso ante trayectorias incompatibles con trayectorias de satélites.

En cuanto al apartado $2D$ se ha optado por no mostrar las proyecciones del resto de órbitas. Esto nos deja con N gráficas, una para cada problema en concreto, cada una de las cuales sería una representación del plano orbital de la trayectoria calculada con la órbita en color negro y el borde de la tierra en color azul. Cabe destacar que las trayectorias se han adimensionalizado con R_e. En la siguiente imagen se puede ver las gráficas que se obtuvieron usando la ecuación (4.1), y con los datos de la *Tabla 4*.

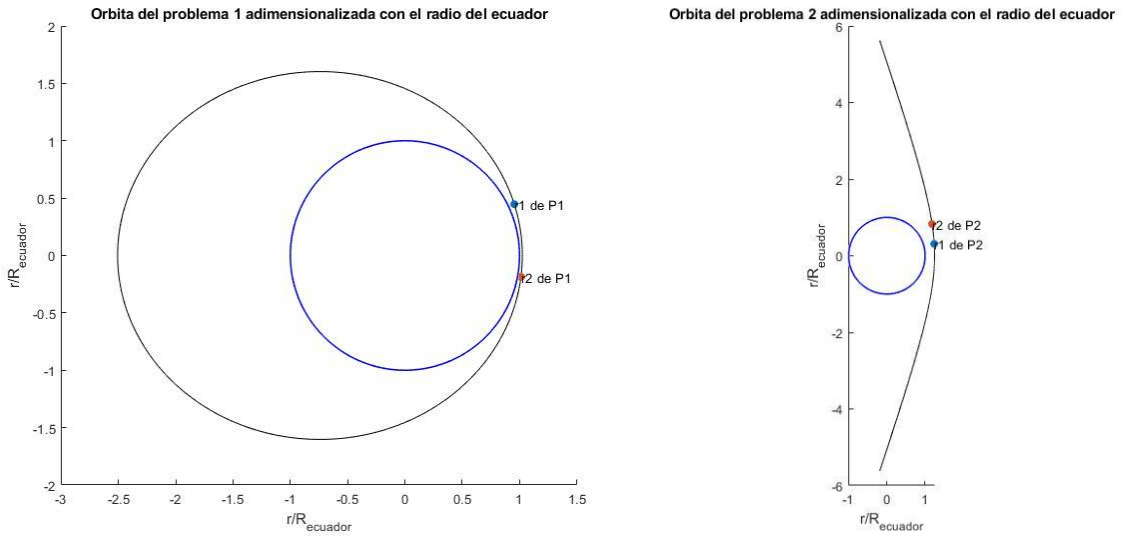


Ilustración 15. Gráficas $2D$ obtenidas durante la resolución de dos problemas.

A continuación, se deja una tabla con los datos utilizados para obtener estos resultados. Indicar que, debido al excesivo tamaño de la tabla, no se ha indicado en las mismas la naturaleza retrógrada de las trayectorias observadas.

Estaciones de observación			
	Latitud	*Longitud*	*Altura* $[m]$
***E* 1**	$N40°22'21.60"$	$W3°55'9.26"$	633
***E* 2**	$N48°51'12.24"$	$E2°20'54.96"$	35

Observaciones								
	$\rho_1\ [km]$	$A_1\ [°]$	$h_1[°]$	t_1	$\rho_2\ [km]$	$A_2[°]$	$h_2[°]$	t_2
***P* 1**	404.8	118.32	59.95	30/03/2017 18:49:45	407	2.12	28.18	31/03/2017 22:00:41
***P* 2**	2004.8	118.32	59.95	30/03/2017 18:49:45	5007	2.12	28.18	30/03/2017 18:53:41

Tabla 4. Datos utilizados para obtener los resultados mostrados como ejemplo.

A continuación, se muestra la pantalla de comandos con los resultados obtenidos tras el cálculo.

```
======================================================================

    PROBLEMA DE LAMBERT 2023. CREADO POR RICARDO SALAZAR GONZALEZ.

======================================================================

Numero de problemas a resolver: 2

Opacida de la tierra en la representacion [0 100]: 100

Precision de la obrita (valor recomendado 2000): 2000

----------------------------------------------------------------------

VALORES PARA EL PROBLEMA 1

Seleccione el tipo de coordenada para itroducir la posicion de la estacion de observacion
numero 1

[0:Lat, long and heigth || 1:XYZ] : 0

Introduzca la LATITUD de la estacion en el formato [grados, minutos, segundos]: [40, 22,
21.60]

Introduzca la orientacion (N o S): n

Introduzca la LONGITUD de la estacion en el formato [grados, minutos, segundos]: [3, 55,
09.26]

Introduzca la orientacion (E o W): w

Introduzca la ALTITUD de la estacion en [m]: 633

Introduzca la primera fecha de observacion ("DD/MM/YYYY HH:MM:SS"): "30/03/2017 18:49:45"

Introduzca el azimut de la primera observacion en grados [0, 360): 118.32

Introduzca la elevacion de la primera observacion en grados [0, 90]: 59.95

Introduzca la distancia de la primera observacion en km: 404.8

Introduzca la segunda fecha de observacion ("DD/MM/YYYY HH:MM:SS"): "31/03/2017 22:00:41"

Introduzca el azimut de la segunda observacion en grados [0, 360]: 2.12

Introduzca la elevacion de la segunda observacion en grados [0, 90]: 28.18

Introduzca la distancia de la segunda observacion en km: 407

Indique si la orbita es POSIGRADA o RETROGRADA: retrograda

----------------------------------------------------------------------

VALORES PARA EL PROBLEMA 2

Seleccione el tipo de coordenada para itroducir la posicion de la estacion de observacion
numero 2

[0:Lat, long and heigth || 1:XYZ] : 0
```

```
Introduzca la LATITUD de la estacion en el formato [grados, minutos, segundos]: [48, 51,
12.24]

Introduzca la orientacion (N o S): n

Introduzca la LONGITUD de la estacion en el formato [grados, minutos, segundos]: [2, 20,
54.96]

Introduzca la orientacion (E o W): e

Introduzca la ALTITUD de la estacion en [m]: 35

Introduzca la primera fecha de observacion ("DD/MM/YYYY HH:MM:SS"): "30/03/2017 18:49:45"

Introduzca el azimut de la primera observacion en grados [0, 360): 118.32

Introduzca la elevacion de la primera observacion en grados [0, 90]: 59.95

Introduzca la distancia de la primera observacion en km: 2004.8

Introduzca la segunda fecha de observacion ("DD/MM/YYYY HH:MM:SS"): "30/03/2017 18:53:41"

Introduzca el azimut de la segunda observacion en grados [0, 360]: 2.12

Introduzca la elevacion de la segunda observacion en grados [0, 90]: 28.18

Introduzca la distancia de la segunda observacion en km: 5007

Indique si la orbita es POSIGRADA o RETROGRADA: retrograda

======================================================================

RESULTADOS PARA EL PROBLEMA 1

Tipo de órbita: eliptica

Los parámetros orbitales, en km y °, son:

      a          e       Inclinacion      RAAN      Argumento_perigeo      r_p        r_a        T
    _____    ________    ___________    ______    _________________    ______    _____    _____

    11264    0.41981       136.17        228.99          88.055          6535.1    15992    11897

----------------------------------------------------------------------

Vector posición [km] y velocidad [km/s] en el punto inicial (1) y final (2)

      r_1          v_1          r_2         v_2         h
    ________    ________    ________    _______    ______

    -1633.3      7.1106     -4412.3     5.1943    -31776
     4918.6      5.4492      1979.6     7.6132     27636
     4281.2     -1.7175      4442.6     1.0335    -43874

----------------------------------------------------------------------

    anomalia_verdadera_1    anomalia_verdadera_2

    ____________________    ____________________

           25.064                   349.62

======================================================================

RESULTADOS PARA EL PROBLEMA 2

Tipo de órbita: hiperbolica

Los parámetros orbitales, en km y °, son:

       a          e       Inclinacion      RAAN      Argumento_perigeo      r_p
```

```
    _______    ______    ___________    ______    _________________    ____

    -3862.5    3.0567      92.445       121.65         32.269          7944

La deflexion de la trayectoria es de 38.1915°

Los angulos de las asintotas son 109.0957° y -109.0957°

------------------------------------------------------------------------

Vector posición [km] y velocidad [km/s] en el punto inicial (1) y final (2)

      r_1        v_1        r_2        v_2         h

    _______    ______    _______    _______    _______

    -2728.1    4.7655    -1559.1     5.0922      96398

     4905.2    -6.796     3219.2    -7.4127      59410

     5880.4    11.505     8469.9     10.442    -4835.8

------------------------------------------------------------------------

    anomalia_verdadera_1    anomalia_verdadera_2

    ____________________    ____________________

           14.12                   34.961

========================================================================

Tiempo de ejecucion t = 2.211324 segundos

========================================================================

>>
```

Capítulo 5. Conclusiones

En este capítulo desarrollaremos unas breves conclusiones extraídas a raíz del desarrollo de este documento. En primer lugar, destacar la importancia de la labor científica realizada, sin la cual hoy no estaríamos donde estamos. Es fundamental para el desarrollo comprender que sin el trabajo de tantos científicos durante tantos años no podríamos disponer de la tecnología actual, lo cual sin duda ha incrementado tanto la esperanza como la calidad de vida, no sólo de seres humanos, sino de muchos otros seres vivos. No cabe duda que las malas praxis están a la orden del día, pero el potencial de la tecnología es muy alto y, siempre que se utilice bien, podrá mejorar el mundo en el que vivimos.

En cuanto a la resolución del problema en sí, esto supone conocer las trayectorias con sólo dos observaciones lo cual implica profundas aplicaciones tanto en el campo de las comunicaciones y la investigación, como en el campo de la defensa militar. La resolución del problema significa todo un hito en el cálculo de trayectorias.

El potencial de este código está limitado puesto que es de carácter pedagógico, sin embargo, podría depurarse para optimizar el tiempo de cálculo y podría automatizarse la introducción de datos, dando lugar a un programa mucho más avanzado de control de elementos orbitantes. Bases de datos de seguimiento, sistemas de control y actuación automatizados sobre los elementos en función de señales enviadas por el código y sistemas de alarma podrían incluirse para hacer mucho más completo el algoritmo.

Otra mejora que podría incluirse sería el de la incorporación de un modelo de propagación con perturbaciones, de manera que podrían así hacerse predicciones y reducir el número de medidas, así como aumentar la eficiencia de las mismas, puesto que se podría *apuntar* hacia el lugar donde se espera que pase el objeto en cuestión.

Una aplicación sencilla, de la cual no se ha hablado en el documento, y la cual es de fácil implementación, sería la de la elaboración de gráficas de ventanas de vuelo (‘*porkchop plots*’) para hacer viajes interplanetarios. Para esto dentro del sistema solar habría que trabajar en el sistema heliocéntrico, dar como datos de entrada las posiciones de dos planetas y las épocas, y resolver los problemas de Lambert resultantes de las distintas combinaciones de posiciones y épocas.

Desde un punto de vista general, sin duda, para el alumno autor del documento, éste trabajo ha supuesto todo un reto, desde la comprensión de las ecuaciones que rigen la física del problema hasta los pormenores relacionados con la implementación del algoritmo en *MATLAB*. En primera persona, puedo decir que el grado de satisfacción es elevado y que espero que la lectura haya sido agradable y llevadera, a pesar de la densidad de la información.

Muchas gracias.

Bibliografía

Battin, R. H. 1999. *An Introduction to the Mathematics and Methods of Astrodynamics.* USA : Education Series, 1999.

Curtis, H. D. 2013. *Orbital Mechanics for Engineering Students.* s.l. : Elservier Science, 2013. ISBN: 9780080977485.

Danby, J. M. A. 1998. *Fundamental of Celestial Mechanics.* Virginia, USA : Atlantic Books, 1998. ISBN 13: 9780943396200.

Lancaster, E. R. y Blanchard, R. C. 1969. *A unified form of Lambert's Theorem.* Washington, D. C. : National Aeronautics and Space Administration, 1969.

Sharaf, M. A., Saad, A. S. y Nouh, Mohamed Ibrahim. 2003. *Lambert Universal Variable Algorithm.* s.l. : Arabian Journal for Science and Engineering, 2003.

Universidad de Sevilla. 2014. *Curso de Mecánica Orbital.* Sevilla : s.n., 2014.

Anexo I. Código *MATLAB*

En este anexo se adjuntarán los códigos de *MATLAB* usados.

AI.I. LambertV3

```
clear all; close all; clc
disp('=================================================================
=====')
disp('    PROBLEMA DE LAMBERT 2023. CREADO POR RICARDO SALAZAR
GONZALEZ.   ')
disp('=================================================================
=====')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
%%PARÁMETROS ESPECIFICOS
R_e=6378.137;
R_p=6356.75231424518;
mu=398600.4;
%TOLERANCIA EN LA RESOLUCIÓN%
tol=1e-10;
%NUMERO MAXIMO DE ITERACIONES%
N=1000;
%IMAGEN DE LA TIERRA
image_file='Planeta_Tierra.jpg';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
%% ENTRADA DE DATOS
fprintf('\n')

variable=0;
 while variable==0
     Num_problemas=input('Numero de problemas a resolver: ');
     fprintf('\n')
     if isnumeric(Num_problemas)==1
        sz=size(Num_problemas);
        if sz(1)==1 && sz(2)==1 && Num_problemas>0
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
 end
fprintf('\n')
variable=0;
 while variable==0
     Opacidad_tierra=input('Opacidad de la tierra en la representacion
[0 100]: ')/100;
     fprintf('\n')
     if isnumeric(Opacidad_tierra)==1
        sz=size(Opacidad_tierra);
```

```
        if sz(1)==1 && sz(2)==1 && Opacidad_tierra>=0 &&
Opacidad_tierra<=100
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
 end
fprintf('\n')
variable=0;
 while variable==0
     Precision_orbita=input('Precision de la obrita (valor recomendado
2000): ');
     fprintf('\n')
     if isnumeric(Precision_orbita)==1
        sz=size(Precision_orbita);
        if sz(1)==1 && sz(2)==1 && Precision_orbita>0 &&
Precision_orbita<=20000
              variable=1;
              Precision_orbita=round(Precision_orbita);
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
 end

fprintf('\n')

for i=1:Num_problemas

 disp('-----------------------------------------------------------------
----')
 disp("VALORES PARA EL PROBLEMA "+i)
 fprintf('\n')
 %%%%%%%%%%%%%%%%%%%%%%%%
 %DATOS DE LA ESTACION i%
 %%%%%%%%%%%%%%%%%%%%%%%%
 disp("Seleccione el tipo de coordenada para itroducir la posicion de
la estacion de observacion numero "+i)
 fprintf('\n')
 variable=0;
 while variable==0
 Modo_GS(i)=input('[0:Lat, long and heigth || 1:XYZ] : ');
 fprintf('\n')
     if isnumeric(Modo_GS(i))==1
        if Modo_GS(i)==1 || Modo_GS(i)==0
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
```

```
                disp('Por favor, introduzca datos validos')
                fprintf('\n')
      end
 end

 if Modo_GS(i)==0

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%POSICION [LAT, LON, H] DE LA ESTACION i%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%LATITUD DE LA ESTACIÓN DE TIERRA i
      variable=0;
      while variable==0
      GS_LAT=input('Introduzca la LATITUD de la estacion en el formato
[grados, minutos, segundos]: ');
      fprintf('\n')
      if isnumeric(GS_LAT)==1
         sz=size(GS_LAT);
         if sz(1)==1 && sz(2)==3
             GS_1(i)=GS_LAT(1)+GS_LAT(2)/60+GS_LAT(3)/3600;
             if GS_1(i)>=0 && GS_1(i)<=360
             variable=1;
             else
             disp('Por favor, introduzca datos validos')
             fprintf('\n')
             end
         else
                disp('Por favor, introduzca datos validos')
                fprintf('\n')
         end
      else
                disp('Por favor, introduzca datos validos')
                fprintf('\n')
      end
      end
%ORIENTACION DE LATITUD DE LA ESTACION DE TIERRA i
      variable=0;
      while variable==0
      GS_LAT_Dir=input('Introduzca la orientacion (N o S): ','s');
      fprintf('\n')
      if ischar(GS_LAT_Dir)==1
         if strcmp(GS_LAT_Dir,'N')||strcmp(GS_LAT_Dir,'n')
             variable=1;
         elseif strcmp(GS_LAT_Dir,'S')||strcmp(GS_LAT_Dir,'s')
             GS_1(i)=-GS_1(i);
                 variable=1;
         else
                disp('Por favor, introduzca datos validos')
                fprintf('\n')
         end
      else
                disp('Por favor, introduzca datos validos')
                fprintf('\n')
      end
      end
%LONGITUD DE LA ESTACION DE TIERRA i
      variable=0;
      while variable==0
```

```
     GS_LONG=input('Introduzca la LONGITUD de la estacion en el
formato [grados, minutos, segundos]: ');
     fprintf('\n')
     if isnumeric(GS_LONG)==1
        sz=size(GS_LONG);
        if sz(1)==1 && sz(2)==3
            GS_2(i)=GS_LONG(1)+GS_LONG(2)/60+GS_LONG(3)/3600;
            if GS_2(i)>=0 && GS_2(i)<=360
                variable=1;
            else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
            end
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
     end
%ORIENTACION DE LONGITUD DE LA ESTACION DE TIERRA i
     variable=0;
     while variable==0
     GS_LON_Dir=input('Introduzca la orientacion (E o W): ','s');
     fprintf('\n')
     if ischar(GS_LON_Dir)==1
        if strcmp(GS_LON_Dir,'E')||strcmp(GS_LON_Dir,'e')
            variable=1;
        elseif strcmp(GS_LON_Dir,'W')||strcmp(GS_LON_Dir,'w')
            GS_2(i)=-GS_2(i);
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
     end
%ALTITUD DE LA ESTACION DE TIERRA i
     variable=0;
     while variable==0
     GS_3(i)=input('Introduzca la ALTITUD de la estacion en [m]:
')*10^-3;
     fprintf('\n')
     if isnumeric(GS_3(i))==1
        sz=size(GS_3(i));
        if sz(1)==1 && sz(2)==1
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
     end
```

```
 elseif Modo_GS(i)==1

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%POSICION [X, Y, Z] DE LA ESTACION i%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
     variable=0;
     while variable==0
     GS_XYZ=input('Introduzca el vector de posicion [X, Y, Z] de la
estacion en [km]: ');
     fprintf('\n')
     if isnumeric(GS_XYZ)==1
        sz=size(GS_XYZ);
        if sz(1)==1 && sz(2)==3
            GS_1(i)=GS_XYZ(1);
            GS_2(i)=GS_XYZ(2);
            GS_3(i)=GS_XYZ(3);

validation_coordinates=(GS_1(i)/R_e)^2+(GS_2(i)/R_e)^2+(GS_3(i)/R_p)^2
;
            if validation_coordinates<=1.1 &&
validation_coordinates>=0.9
                variable=1;
            else
                disp('Por favor, introduzca datos validos')
                fprintf('\n')
            end
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
     end

 end

%%%%%%%%%%%%%%%%%%%%%%%
%DATOS DE LOS OBJETOS %
%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%OBSERVACION 1
%TIEMPO: 1
     variable=0;
     while variable==0
     obj1_time(i)=input('Introduzca la primera fecha de observacion
("DD/MM/YYYY HH:MM:SS"): ');
     fprintf('\n')
     if isstring(obj1_time(i))==1
            variable=1;
     else
            disp('Por favor, introduzca datos validos')
            fprintf('\n')
     end
     end
%AZIMUT: 1
     variable=0;
```

```
     while variable==0
     obj1_az(i)=input('Introduzca el azimut de la primera observacion
en grados [0, 360): ');
     fprintf('\n')
     if isnumeric(obj1_az(i))==1
        sz=size(obj1_az(i));
        if sz(1)==1 && sz(2)==1 && obj1_az(i)<360 && obj1_az(i)>=0
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
     end
%ELEVACION: 1
     variable=0;
     while variable==0
     obj1_h(i)=input('Introduzca la elevacion de la primera
observacion en grados [0, 90]: ');
     fprintf('\n')
     if isnumeric(obj1_h(i))==1
        sz=size(obj1_h(i));
        if sz(1)==1 && sz(2)==1 && obj1_h(i)<=90 && obj1_h(i)>=0
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
     end
%DISTANCIA: 1
     variable=0;
     while variable==0
     obj1_rho(i)=input('Introduzca la distancia de la primera
observacion en km: ');
     fprintf('\n')
     if isnumeric(obj1_rho(i))==1
        sz=size(obj1_rho(i));
        if sz(1)==1 && sz(2)==1 && obj1_rho(i)>0
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
     end

%%%%%%%%%%%%%%%%%%%%%%%%%OBSERVACION 2
%TIEMPO: 2
     variable=0;
     while variable==0
```

```
     obj2_time(i)=input('Introduzca la segunda fecha de observacion
("DD/MM/YYYY HH:MM:SS"): ');
     fprintf('\n')
     if isstring(obj2_time(i))==1

delta_t_DateTime=between(datetime(obj1_time(i),'InputFormat','dd/MM/yy
yy HH:mm:ss'),datetime(obj2_time(i),'InputFormat','dd/MM/yyyy
HH:mm:ss'));
            delta_t(i)=seconds(split(delta_t_DateTime,'Time'));
            if delta_t(i)>0
            variable=1;
            else
            disp('Por favor, introduzca datos validos')
            fprintf('\n')
            end
     else
            disp('Por favor, introduzca datos validos')
            fprintf('\n')
     end
     end
%AZIMUT: 2
     variable=0;
     while variable==0
     obj2_az(i)=input('Introduzca el azimut de la segunda observacion
en grados [0, 360]: ');
     fprintf('\n')
     if isnumeric(obj2_az(i))==1
        sz=size(obj2_az(i));
        if sz(1)==1 && sz(2)==1 && obj2_az(i)<=360 && obj2_az(i)>=0
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
     end
%ELEVACION: 2
     variable=0;
     while variable==0
     obj2_h(i)=input('Introduzca la elevacion de la segunda
observacion en grados [0, 90]: ');
     fprintf('\n')
     if isnumeric(obj2_h(i))==1
        sz=size(obj2_h(i));
        if sz(1)==1 && sz(2)==1 && obj2_h(i)<=90 && obj2_h(i)>=0
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
     end
%DISTANCIA: 2
     variable=0;
     while variable==0
```

```
     obj2_rho(i)=input('Introduzca la distancia de la segunda
observacion en km: ');
     fprintf('\n')
     if isnumeric(obj2_rho(i))==1
        sz=size(obj2_rho(i));
        if sz(1)==1 && sz(2)==1 && obj2_rho(i)>0
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
     end

%%%%%%%%%%%%%%% DETERMINACIÓN DEL CAMINO A CALCULAR (CORTO O LARGO)
    variable=0;
     while variable==0
    value=input('Indique si la orbita es POSIGRADA o RETROGRADA:
','s'); %ESCRIBIR CORTO O LARGO EN MAYÚSCULAS
    fprintf('\n')
     if ischar(value)==1
        if strcmp(value,'POSIGRADA')||strcmp(value,'posigrada')
            camino(i)=1;
            variable=1;
        elseif strcmp(value,'RETROGRADA')||strcmp(value,'retrograda')
            camino(i)=-1;
            variable=1;
        else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
        end
     else
              disp('Por favor, introduzca datos validos')
              fprintf('\n')
     end
     end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%

%%%%%%%%%%%%%%%   FIN DE LA INTRODUCCIÓN DE DATOS
%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%

%%                    RESOLUCIÓN DEL PROBLEMA
%%
tic;
for i=1:Num_problemas
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
%% OBTENCION DE POSICIONES DEL OBJETO EN SISTEMA GEOCENTRICO
ECUATORIAL (0)
if Modo_GS(i)==0
```

```
[r_1,r_obs_1,GST_1]=posicion_Modo_GS_0(obj1_time(i),obj1_az(i),obj1_h(
i),obj1_rho(i),GS_1(i),GS_2(i),GS_3(i),R_e,R_p);

[r_2,r_obs_2,GST_2]=posicion_Modo_GS_0(obj2_time(i),obj2_az(i),obj2_h(
i),obj2_rho(i),GS_1(i),GS_2(i),GS_3(i),R_e,R_p);

else


[r_1,r_obs_1,GST_1]=posicion_Modo_GS_1(obj1_time(i),obj1_az(i),obj1_h(
i),obj1_rho(i),GS_1(i),GS_2(i),GS_3(i),R_e,R_p);

[r_2,r_obs_2,GST_2]=posicion_Modo_GS_1(obj2_time(i),obj2_az(i),obj2_h(
i),obj2_rho(i),GS_1(i),GS_2(i),GS_3(i),R_e,R_p);

end
%Para pintar las posiciones de las estaciones i>1 con el GMST_1 de i=1
if i>1
    angle=(GST-GST_1)*pi/180;
    r_obs_1=[cos(angle) -sin(angle) 0; sin(angle) cos(angle) 0; 0 0
1]*r_obs_1;
    r_obs_2=[cos(angle) -sin(angle) 0; sin(angle) cos(angle) 0; 0 0
1]*r_obs_2;
else
    GST=GST_1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
%% RESOLUCIÓN DE PROBLEMA DE LAMBERT %%

[z,v_1,v_2,err]=SolveLambert(r_1,r_2,delta_t(i),mu,camino(i),tol,N);
v_1=real(v_1);%Elimina un posible remanente imaginario fruto de
errores numéricos
v_2=real(v_2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%
%% CÁLCULO DE PARÁMETROS ORBITALES %%%

[p,h,e,a,T,anomalia_verdadera_1,anomalia_verdadera_2,r_p,r_a,RAAN,Incl
inacion,Argumento_perigeo,u_1,u_2,Longitud_perigeo,Lambda_T_1,Lambda_T
_2,MAT_GIRO,theta_graf_min,theta_graf_max]=Parametros_orbitales(r_1,v_
1,r_2,v_2,mu,i);

%% ALMACENAMIENTO DE RESULTADOS

R_OBS(:,:,i)=[r_obs_1 r_obs_2];
R(:,:,i)=[r_1 r_2];
V(:,:,i)=[v_1 v_2];
P(i)=p;
H(:,i)=h;
EXCENTRICIDAD(i)=e;
SEMIEJE_MAYOR(i)=a;
ANOMALIA_VERDADERA(:,i)=[anomalia_verdadera_1; anomalia_verdadera_2];
RADIO_PERIGEO(i)=r_p;
RADIO_APOGEO(i)=r_a;
LONGITUD_NODO_ASCENDENTE(i)=RAAN;
INCLINACION(i)=Inclinacion;
```

```
ARGUMENTO_PERIGEO(i)=Argumento_perigeo;
U(:,i)=[u_1; u_2];
LONGITUD_PERIGEO(i)=Longitud_perigeo;
LAMBDA_T(:,i)=[Lambda_T_1; Lambda_T_2];
M_GIRO(:,:,i)=MAT_GIRO;
THETA_GRAF(:,i)=[theta_graf_min; theta_graf_max];
GMST_GLOB(:,i)=[GST_1; GST_2];

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%

%%%%%%%%%%%%%%%%   FIN DE LA RESOLUCION DEL PROBLEMA
%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%

%%                                      GRÁFICOS
%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%

figure('Color','k');%Establece el fondo negro
hold on;
set(gca,'NextPlot','add', 'Visible','off');%Quita los ejes normales
axis equal;axis auto;%Establece ejes por defecto
view(GST+90,0);%Establece vista inicial
axis vis3d;%Especifciacion de graficado 3d

%light('Position',[1 1 1]);
[X_Earth,Y_Earth,Z_Earth]=ellipsoid(0,0,0,R_e,R_e,R_p,180);%Crea un
elipsoide centrado
globe=surf(X_Earth,Y_Earth,-Z_Earth,'FaceColor','none','EdgeColor',
0.5*[0.5 1 1]);%Lo establece como superficie 3d
shading interp
lightangle(GST+90,0)
globe.FaceLighting = 'gouraud';
globe.AmbientStrength = 0.05;
globe.DiffuseStrength = 0.95;
hgx = hgtransform;
set(hgx,'Matrix',makehgtform('zrotate',pi*GST/180)); %Rota la tierra
GMST_1 grados
set(globe,'Parent',hgx);
cdata=imread(image_file);%Carga la imagen de la tierra
set(globe,'FaceColor','texturemap','CData',cdata,'FaceAlpha',Opacidad_
tierra,'EdgeColor','none');%Set image as color data. Set face color to
indicate texture map. Turn off mesh edges.

for i=1:Num_problemas
theta_graf=linspace(THETA_GRAF(1,i),THETA_GRAF(2,i),Precision_orbita);

%Distancia al Foco en funcion del argumento
r(i,:)=P(i)./(1+EXCENTRICIDAD(i)*cos(theta_graf));

%Puntos para gráfica 3d
```

```
r_3d=[r(i,:); r(i,:); r(i,:)].*(M_GIRO(:,:,i)*[cos(theta_graf);
sin(theta_graf); zeros(1,length(theta_graf))]);

graficos(i)=1;
for k=1:Precision_orbita
    value=(r_3d(1,k)/R_e)^2+(r_3d(2,k)/R_e)^2+(r_3d(3,k)/R_p)^2;
    if value<1
        graficos(i)=0;
        break
    end
end

if graficos(i)==1

    plot3(r_3d(1,:),r_3d(2,:),r_3d(3,:),'linewidth',1)
    hold on
    scatter3(R_OBS(1,1,i),R_OBS(2,1,i),R_OBS(3,1,i),'filled')
    hold on
    text(R_OBS(1,1,i),R_OBS(2,1,i),R_OBS(3,1,i),"ESTACION
P"+(i),'Color','white');
    hold on
    scatter3(R(1,1,i),R(2,1,i),R(3,1,i),'filled')
    hold on
    scatter3(R(1,2,i),R(2,2,i),R(3,2,i),'filled')

else
    disp('-------------------------------------------------------------
---------')
    disp("La solucion del problema "+i+" no es viable para un
satelite")
end

end

for i=1:Num_problemas
    if graficos(i)==1

%Puntos para gráfica 2d
r_2d=r(i,:).*[cos(theta_graf); sin(theta_graf)];
R_1_2d=M_GIRO(:,:,i)\R(:,1,i);
R_2_2d=M_GIRO(:,:,i)\R(:,2,i);

    figure(i+1)
    title("Orbita del problema "+i+" adimensionalizada con el radio
del ecuador")
    hold on
    xlabel('r/R_e_c_u_a_d_o_r')
    hold on
    ylabel('r/R_e_c_u_a_d_o_r')
    plot(r_2d(1,:)./R_e,r_2d(2,:)./R_e,'k')
    hold on
    scatter(R_1_2d(1)/R_e,R_1_2d(2)/R_e,'filled')
    hold on
    text(R_1_2d(1)/R_e,R_1_2d(2)/R_e,"r1 de P"+(i));
    hold on
    scatter(R_2_2d(1)/R_e,R_2_2d(2)/R_e,'filled')
    hold on
    text(R_2_2d(1)/R_e,R_2_2d(2)/R_e,"r2 de P"+(i));
    hold on
```

```
    viscircles([0 0],1,'linewidth',1,'color','blue');
    set(gca,'DataAspectRatio',[1 1 1])

    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%

%%%%%%%%%%%%%%%%%%%%    FIN DEL APARTADO DE GRAFICOS
%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%

t_ejecucion=toc;

disp('=================================================================
=====')
fprintf('Tiempo de ejecucion t = %.6f segundos\n',t_ejecucion);
disp('=================================================================
=====')
```

AI.II. *Posicion_Modo_GS_0*

```
function
[r,r0_obs,GST]=posicion_Modo_GS_0(Date,Az,h,rho,LAT,LON,H,R_e,R_p)

%calculo de GST0
date=datetime(Date,'InputFormat','dd/MM/yyyy HH:mm:ss');

%J0=juliandate(date); %dias julianos a las 00:00 UT
J0=(367*year(date))-
floor((7*(year(date)+(floor((month(date)+9)/12)))/4))+floor((275*month
(date))/9)+day(date)+1721013.5;
T0=(J0-2451545)/36525;
GST0=100.4606184+(36000.77004*T0)+(0.000387933*T0^2)-(2.583*10^-
8*T0^3);

%Calculo de GST
UT=hour(date)+(minute(date)/60)+(second(date)/3600);
GST=GST0+360.98564724*UT/24;%DEG
GST=GST-floor(GST/360)*360;
gst=GST*pi/180;


LAT=LAT*pi/180;
LON=LON*pi/180;

%COORDENADAS DEL OBSERVADOR EN SISTEMA GEOGRAFICO (1)
f=(R_e-R_p)/R_e;
x1_obs=(H+R_e/sqrt(1-f*(2-f)*sin(LAT)^2))*cos(LAT)*cos(LON);
y1_obs=(H+R_e/sqrt(1-f*(2-f)*sin(LAT)^2))*cos(LAT)*sin(LON);
z1_obs=(H+R_e*(1-f)^2/sqrt(1-f*(2-f)*sin(LAT)^2))*sin(LAT);

%MATRIZ DE GIRO DE SISTEMA GEOGRAFICO (1) A SISTEMA GEOCENTRICO
ECUATORIAL (0)
```

```
GIRO_10=[cos(gst) -sin(gst) 0;sin(gst) cos(gst) 0; 0 0 1];

%COORDENADAS DEL OBSERVADOR EN SISTEMA GEOCENTRICO ECUATORIAL (0)
r0_obs=GIRO_10*[x1_obs; y1_obs; z1_obs];

%VECTORES BASE DE SITEMA TOPOCENTRICO (2)
e3_2=2.*[x1_obs/R_e^2; y1_obs/R_e^2; z1_obs/R_p^2];
e3_2=e3_2/norm(e3_2); %normal a superficie de elipsoide
e1_2=cross([0; 0; 1],e3_2);%perpendicular a 0XY y a normal a la
superficie
e2_2=cross(e3_2,e1_2);

%%MATRIZ DE GIRO DE SITEMA TOPOCENTRICO (2) A SISTEMA GEOGRAFICO (1)
GIRO_21=[e1_2 e2_2 e3_2];

%POSICION DE OBJETO OBSERVADO EN SISTEMA TOPOCENTRICO (2)
Az=Az*pi/180;
h=h*pi/180;

x2_obj=rho*cos(h)*sin(Az);
y2_obj=rho*cos(h)*cos(Az);
z2_obj=rho*sin(h);

%POSICION DE OBJETO OBSERVADO EN SISTEMA GEOCENTRICO ECUATORIAL (0)

r=r0_obs+GIRO_10*GIRO_21*[x2_obj; y2_obj; z2_obj];

end
```

AI.III. *Posicion_Modo_GS_1*

```
function
[r,r0_obs,GST]=posicion_Modo_GS_1(Date,Az,h,rho,X,Y,Z,R_e,R_p);
%calculo de GST0
date=datetime(Date,'InputFormat','dd/MM/yyyy HH:mm:ss');

%J0=juliandate(date); %dias julianos a las 00:00 UT
J0=(367*year(date))-
floor((7*(year(date)+(floor((month(date)+9)/12)))/4))+floor((275*month
(date))/9)+day(date)+1721013.5;
T0=(J0-2451545)/36525;
GST0=100.4606184+(36000.77004*T0)+(0.000387933*T0^2)-(2.583*10^-
8*T0^3);

%Calculo de GST
UT=hour(date)+(minute(date)/60)+(second(date)/3600);
GST=GST0+360.98564724*UT/24;%DEG
GST=GST-floor(GST/360)*360;
gst=GST*pi/180;
%calculo de posicion de observacion

%MATRIZ DE GIRO DE SISTEMA GEOGRAFICO (1) A SISTEMA GEOCENTRICO
ECUATORIAL (0)
GIRO_10=[cos(gst) -sin(gst) 0;sin(gst) cos(gst) 0; 0 0 1];
%COORDENADAS DEL OBSERVADOR EN SISTEMA GEOCENTRICO ECUATORIAL (0)
r0_obs=GIRO_10*[X; Y; Z];
```

```
%VECTORES BASE DE SITEMA TOPOCENTRICO (2)
e3_2=2.*[X/R_e^2; Y/R_e^2; Z/R_p^2];
e3_2=e3_2/norm(e3_2); %normal a superficie de elipsoide
e1_2=cross([0; 0; 1],e3_2);%perpendicular a 0XY y a normal a la
superficie
e2_2=cross(e3_2,e1_2);
%%MATRIZ DE GIRO DE SITEMA TOPOCENTRICO (2) A SISTEMA GEOGRAFICO (1)
GIRO_21=[e1_2 e2_2 e3_2];
%POSICION DE OBJETO OBSERVADO EN SISTEMA TOPOCENTRICO (2)
Az=Az*pi/180;
h=h*pi/180;

x2_obj=rho*cos(h)*sin(Az);
y2_obj=rho*cos(h)*cos(Az);
z2_obj=rho*sin(h);
%POSICION DE OBJETO OBSERVADO EN SISTEMA GEOCENTRICO ECUATORIAL (0)
%[x0_obj; y0_obj; z0_obj]
r=r0_obs+GIRO_10*GIRO_21*[x2_obj; y2_obj; z2_obj];
end
```

AI.IV. *SolveLambert*

```
function [z,v_1,v_2,err]=SolveLambert(r_1,r_2,delta_t,mu,camino,tol,N)
r_3=cross(r_1,r_2);

r1=norm(r_1);
r2=norm(r_2);

delta_theta=acos(dot(r_1,r_2)/r1/r2);

if r_3(3)<0 && camino==1
    delta_theta=2*pi-delta_theta;
elseif r_3(3)>0 && camino==-1
    delta_theta=2*pi-delta_theta;
end

%CÁLCULO DE CONSTANTE A
A=sin(delta_theta)*sqrt(r1*r2/(1-cos(delta_theta)));

%ESTIMACION DE VALOR INICIAL DE PHI
z=-100;
F=-1;
while F<0
    if z==0
        C2=1/2;
        C3=1/6;
    elseif z>0
        C2=(1-cos(sqrt(z)))/z;
        C3=(sqrt(z)-sin(sqrt(z)))/z^(3/2);
    else
        C2=(cosh(sqrt(-z))-1)/(-z);
        C3=(sinh(sqrt(-z))-sqrt(-z))/(-z)^(3/2);
    end

    y=r1+r2+A*(z*C3-1)/sqrt(C2);

    F=(y/C2)^(3/2)*C3+A*sqrt(y)-sqrt(mu)*delta_t;
```

```
    z=z+1;
end

%RESOLUCIÓN NEWTON
num_it=0;
err=1;

while err>tol && num_it<N
    if z==0
        C2=1/2;
        C3=1/6;
    elseif z>0
        C2=(1-cos(sqrt(z)))/z;
        C3=(sqrt(z)-sin(sqrt(z)))/z^(3/2);
    else
        C2=(cosh(sqrt(-z))-1)/(-z);
        C3=(sinh(sqrt(-z))-sqrt(-z))/(-z)^(3/2);
    end

    y=r1+r2+A*(z*C3-1)/sqrt(C2);

    F=(y/C2)^(3/2)*C3+A*sqrt(y)-sqrt(mu)*delta_t;

    if z==0
        dF_dphi=sqrt(2)*y^(3/2)/40+A*(sqrt(y)+A/sqrt(2*y))/8;
    else
        dF_dphi=(y/C2)^(3/2)*((C2-
3*C3/2/C2)/2/z+3*C3^2/4/C2)+A*(3*C3*sqrt(y)/C2+A*sqrt(C2/y))/8;
    end

    z=z-F/dF_dphi;

    err=abs(F);
    num_it=num_it+1;
end

%CÁLCULO COEFICIENTES DE LAGRANGE
f=1-y/r1;
g=A*sqrt(y/mu);
dg_dt=1-y/r2;

%CÁLCULO DE VECTORES DE VELOCIDAD
v_1=(r_2-f*r_1)/g;
v_2=(dg_dt*r_2-r_1)/g;

end
```

AI.V. *Parametros_orbitales*

```
function
[p,h,e,a,T,anomalia_verdadera_1,anomalia_verdadera_2,r_p,r_a,RAAN,Incl
inacion,Argumento_perigeo,u_1,u_2,Longitud_perigeo,Lambda_T_1,Lambda_T
_2,MAT_GIRO,theta_graf_min,theta_graf_max]=Parametros_orbitales(r_1,v_
1,r_2,v_2,mu,i)
%DEVUELVE PARÁMETROS ORBITALES.
%VALORES ANGULARES EN GRADOSA
```

```
disp('================================================================
=====')
fprintf('\n')
disp("RESULTADOS PARA EL PROBLEMA "+i)
fprintf('\n')

h1=cross(r_1,v_1);
h2=cross(r_2,v_2);
h=(h1+h2)/2;
p=norm(h)^2/mu;
E1=cross(v_1,h)/mu-r_1/norm(r_1);
E2=cross(v_2,h)/mu-r_2/norm(r_2);
E=(E1+E2)/2;
e=norm(E);
e1=E/e;
e3=h/norm(h);
e2=cross(e3,e1);
%MAT_GIRO es un triedro formado por los vectores unitarios Ê, ?xÊ, ?
MAT_GIRO=[e1 e2 e3];

I=acos(h(3)/norm(h));
Inclinacion=I*180/pi;

r_p=p/(1+e);
%Valores espeiales por defecto
delta=NaN; %deflexion de la trayectoria
T=NaN;
r_a=NaN;
a=NaN;
theta=pi;

%Tolerancia para ser considerada orbita parabolica
tol=1e-8;

if abs(e-1)>tol

    a=p/(1-e^2);
    r_a=abs(p/(1-e));
    theta_graf_max=2*pi;
    theta_graf_min=0;

    if e<1 && e>tol

        tipo_de_orbita='eliptica'; %órbita elíptica
        T=2*pi*sqrt(a^3/mu);

    elseif e>=0 && e<=tol

        tipo_de_orbita='circular'; %órbita circular
        T=2*pi*sqrt(a^3/mu);

    elseif e>1

        tipo_de_orbita='hiperbolica'; %órbita hiperbólica
        theta=acos(-1/e);
        theta_graf_max=theta-0.3;
        theta_graf_min=-theta_graf_max;
```

```
        theta=theta*180/pi;
        delta=2*asin(1/e)*180/pi;

    end

else

    tipo_de_orbita='parabolica'; %órbita parabólica
    theta_graf_max=pi-0.3;
    theta_graf_min=-pi+0.3;

end

disp(['Tipo de órbita: ',tipo_de_orbita])
fprintf('\n')

%Clasificacion orbitas especiales

Argumento_perigeo=NaN;
RAAN=NaN;
Longitud_perigeo=NaN;
anomalia_verdadera_1=NaN;
anomalia_verdadera_2=NaN;
u_1=NaN;
u_2=NaN;
Lambda_T_1=NaN;
Lambda_T_2=NaN;

r1_unitario=r_1/norm(r_1);
r2_unitario=r_2/norm(r_2);

if Inclinacion==0 || Inclinacion==180

    if strcmp(tipo_de_orbita,'circular')

        Lambda_T_1=180*acos(r1_unitario(1))/pi;
        if r_1(2)<0 && h(3)>0
            Lambda_T_1=360-Lambda_T_1;
        end
        if r_1(2)>0 && h(3)<0
            Lambda_T_1=360-Lambda_T_1;
        end

        Lambda_T_2=180*acos(r2_unitario(1))/pi;
        if r_1(2)<0 && h(3)>0
            Lambda_T_2=360-Lambda_T_2;
        end
        if r_1(2)>0 && h(3)<0
            Lambda_T_2=360-Lambda_T_2;
        end

        MAT_GIRO=eye(3);
        if h(3)<0
        MAT_GIRO(3,3)=-1;
        end

    else
```

```
        Longitud_perigeo=180*acos(e1(1))/pi;
        if e1(2)<0 && h(3)>0
            Longitud_perigeo=360-Longitud_perigeo;
        end
        if e1(2)>0 && h(3)<0
            Longitud_perigeo=360-Longitud_perigeo;
        end

        anomalia_verdadera_1=180*acos(dot(r1_unitario,e1))/pi;
        if dot(r_1,v_1)<0
            anomalia_verdadera_1=360-anomalia_verdadera_1;
        end

        anomalia_verdadera_2=180*acos(dot(r2_unitario,e1))/pi;
        if dot(r_2,v_2)<0
            anomalia_verdadera_2=360-anomalia_verdadera_2;
        end

    end

else

    n=cross([0 0 1]',h);
    n=n/norm(n);

    RAAN=180*acos(n(1))/pi;
    if n(2)<0
    RAAN=360-RAAN;
    end

    if strcmp(tipo_de_orbita,'circular')

        e2=cross(e3,n);
        MAT_GIRO=[n e2 e3];

        u_1=180*acos(dot(r1_unitario,n))/pi;
        if r1_unitario(3)<0
            u_1=360-u_1;
        end

        u_2=180*acos(dot(r2_unitario,n))/pi;
        if r2_unitario(3)<0
            u_2=360-u_2;
        end

    else

        Argumento_perigeo=180*acos(dot(n,e1))/pi;
        if e1(3)<0
            Argumento_perigeo=360-Argumento_perigeo;
        end

        anomalia_verdadera_1=180*acos(dot(r1_unitario,e1))/pi;
        if dot(r_1,v_1)<0
            anomalia_verdadera_1=360-anomalia_verdadera_1;
        end

        anomalia_verdadera_2=180*acos(dot(r2_unitario,e1))/pi;
```

```
        if dot(r_2,v_2)<0
            anomalia_verdadera_2=360-anomalia_verdadera_2;
        end

    end

end

disp('Los parámetros orbitales, en km y °, son: ')
fprintf('\n')
if strcmp(tipo_de_orbita,'parabolica')
    if Inclinacion==0 || Inclinacion==180
        disp(table(e,Inclinacion,Longitud_perigeo,r_p))
        fprintf('\n')
        fprintf('RAAN y argumento de perigeo indefinidos\n');
        fprintf('Longitud del periastro = RAAN + argumento del
perigeo\n');
    else
        disp(table(e,Inclinacion,RAAN,Argumento_perigeo,r_p))
    end
    fprintf('\n')
    disp('El semieje mayor tiende a infinito')
    value=table(anomalia_verdadera_1,anomalia_verdadera_2);
elseif strcmp(tipo_de_orbita,'hiperbolica')
    if Inclinacion==0 || Inclinacion==180
        disp(table(a,e,Inclinacion,Longitud_perigeo,r_p))
        fprintf('\n')
        fprintf('RAAN y argumento de perigeo indefinidos\n');
        fprintf('Longitud del periastro = RAAN + argumento del
perigeo\n');
    else
        disp(table(a,e,Inclinacion,RAAN,Argumento_perigeo,r_p))
    end
    fprintf('\n')
    fprintf('La deflexion de la trayectoria es de %.4f°\n',delta);
    fprintf('Los angulos de las asintotas son %.4f° y %.4f°\n',theta,-
theta);
    value=table(anomalia_verdadera_1,anomalia_verdadera_2);
elseif strcmp(tipo_de_orbita,'circular')
    if Inclinacion==0 || Inclinacion==180
        disp(table(a,e,Inclinacion,T))
        fprintf('\n')
        fprintf('RAAN, argumento de perigeo y anomalia verdadera
indefinidos\n');
        fprintf('Lambda_T = RAAN + argumento del perigeo + anomalia
verdadera\n');
        value=table(Lambda_T_1,Lambda_T_2);
    else
        disp(table(a,e,Inclinacion,RAAN,T))
        fprintf('\n')
        fprintf('Argumento de perigeo y anomalia verdadera
indefinidos\n');
        fprintf('u = argumento del perigeo + anomalia verdadera\n');
        value=table(u_1,u_2);
    end
    fprintf('\n')
    fprintf('El radio de la orbita es de %.4f Km\n',r_p);
elseif strcmp(tipo_de_orbita,'eliptica')
    if Inclinacion==0 || Inclinacion==180
        disp(table(a,e,Inclinacion,Longitud_perigeo,r_p,r_a,T))
```

```
        fprintf('\n')
        fprintf('RAAN y argumento de perigeo indefinidos\n');
        fprintf('Longitud del periastro = RAAN + argumento del
perigeo\n');
    else
        disp(table(a,e,Inclinacion,RAAN,Argumento_perigeo,r_p,r_a,T))
    end
    value=table(anomalia_verdadera_1,anomalia_verdadera_2);
end
disp('-----------------------------------------------------------------
---')
fprintf('\n')
disp('Vector posición [km] y velocidad [km/s] en el punto inicial (1)
y final (2)')
fprintf('\n')
disp(table(r_1,v_1,r_2,v_2,h))
disp('-----------------------------------------------------------------
---')
fprintf('\n')
disp(value)
end
```