

UACM

Universidad Autónoma
de la Ciudad de México

NADA HUMANO ME ES AJENO

COLEGIO DE CIENCIA Y TECNOLOGÍA

LICENCIATURA EN MODELACIÓN MATEMÁTICA

Solución Numérica al Problema de los tres y N cuerpos

TESIS

QUE PARA OPTAR POR EL TÍTULO DE

LICENCIADO EN MODELACIÓN MATEMÁTICA

PRESENTA

ROBERTO SAMUEL SANCHEZ RUBIO Y AGUILERA

DIRECTOR

M. EN C. ENRIQUE CRUZ MARTÍNEZ

Ciudad de México, diciembre de 2025.

SISTEMA BIBLIOTECARIO DE INFORMACIÓN Y DOCUMENTACIÓN



UNIVERSIDAD AUTÓNOMA DE LA CIUDAD DE MÉXICO COORDINACIÓN ACADÉMICA

RESTRICCIONES DE USO PARA LAS TESIS DIGITALES

DERECHOS RESERVADOS[©]

La presente obra y cada uno de sus elementos está protegido por la Ley Federal del Derecho de Autor; por la Ley de la Universidad Autónoma de la Ciudad de México, así como lo dispuesto por el Estatuto General Orgánico de la Universidad Autónoma de la Ciudad de México; del mismo modo por lo establecido en el Acuerdo por el cual se aprueba la Norma mediante la que se Modifican, Adicionan y Derogan Diversas Disposiciones del Estatuto Orgánico de la Universidad de la Ciudad de México, aprobado por el Consejo de Gobierno el 29 de enero de 2002, con el objeto de definir las atribuciones de las diferentes unidades que forman la estructura de la Universidad Autónoma de la Ciudad de México como organismo público autónomo y lo establecido en el Reglamento de Titulación de la Universidad Autónoma de la Ciudad de México.

Por lo que el uso de su contenido, así como cada una de las partes que lo integran y que están bajo la tutela de la Ley Federal de Derecho de Autor, obliga a quien haga uso de la presente obra a considerar que solo lo realizará si es para fines educativos, académicos, de investigación o informativos y se compromete a citar esta fuente, así como a su autor ó autores. Por lo tanto, queda prohibida su reproducción total o parcial y cualquier uso diferente a los ya mencionados, los cuales serán reclamados por el titular de los derechos y sancionados conforme a la legislación aplicable.

*A mi mamá, por su amor incondicional y por todo lo que hizo por mí.
Hoy ya no estás conmigo; no llegaste a verme titularme como hubieses
querido, pero al final lo logré. Lo logré en parte gracias a ti. Se cumplió
nuestro sueño. Gracias por tu apoyo y por tu amor. Siempre estarás en mi
corazón.*

*A mi papá, gracias por estar conmigo, por animarme a alcanzar mis
sueños, por tus consejos y por todo lo que me has enseñado.*

Agradecimientos

Quiero agradecer al profesor Enrique Cruz Martines, por haberme acompañado en este proceso de aprendizaje, por su paciencia, guía, motivación y por todo el apoyo que me brindó en este proceso. Me llevo de él muchas lecciones tanto académicas como personales. Le agradezco al profesor Agustín González Villanueva por su manera tan particular de enseñar y su dedicación. Gracias profesores por darme un espacio donde estudiar y desarrollarme, el laboratorio Laceci y Lamat.

Le agradezco a la Universidad Autónoma de la Ciudad de México la formación académica que tuve, las oportunidades y apoyos que me brindó. El apoyo para el congreso fue enriquecedor para mi formación al igual que la beca de tesis fue de mucho apoyo.

A mi primo Ivan Cruz Aguilera que me impulsó a entrar en esta escuela. A mi tía Elena Cruz Aguilera quien fue mi primera maestra de matemáticas. En general gracias a los dos por siempre estar conmigo.

A mi novia Bianca Agosto Nino gracias por compartir conmigo en esta etapa de la vida. Gracias por tu apoyo y todo lo que haces por mi.

Índice general

1. Historia del estudio del cielo	8
1.0.1. Los griegos	8
1.0.2. Nicolás Copérnico(1473-1543)	9
1.0.3. Galileo Galilei(1564-1642)	11
1.0.4. Johannes Kepler	13
1.0.5. Issac Newton (1643-1727)	15
2. Problema de kepler	17
2.1. Problema Inicial	19
2.1.1. Explicación	19
2.1.2. Formulación Matemática sobre las Leyes del movimien- to celeste	19
2.1.3. Resolución del problema por métodos numéricos	21
2.2. Normalización	22
2.3. Aplicación	25
2.3.1. Órbitas de la estación espacial internacional	25
2.3.2. Normalización de los datos	26
2.3.3. Programa y Simulaciones	27
2.3.4. Análisis por medio de la Ley de conservación de la Energía	30
2.3.5. Buscando Órbitas para una distancia entre la Tierra y la Luna	31
2.4. Órbitas con luna en movimiento	35
3. Gravitación Universal	39
3.1. Introducción	39
3.2. Deducción analítica de las ecuaciones de movimiento gravita- cional	40

3.3. Solución Numérica	45
4. El problema de los 3 cuerpos restringido	48
4.1. Introducción	48
4.2. Deducción de las Ecuaciones de Movimiento para 3 cuerpos. .	49
4.3. Aplicando el método de Euler	51
4.4. Aplicando el método Runge kutta de 4to orden	55
4.5. Simulaciones numéricas y resultados	57
5. Problema de N cuerpos	61
5.1. Introducción	61
5.2. Simulaciones y Resultados para 4 cuerpos	61
5.3. Simulación para N cuerpos	63
5.3.1. Simulaciones del Big Bang y Resultados	64
6. Buscando el Sistema solar usando el método de Montecarlo	66
6.1. Múltiples Simulaciones (Método de montecarlo)	66
6.2. Lenguaje Script y organización de archivos	67
6.3. Probando múltiples condiciones con Montarlo	69
7. Conclusiones	73
A. Códigos en Fortran	76
B. Códigos Scripts	103

Introducción

Desde la antigüedad el ser humano ha observado el cielo nocturno tratando de dar una explicación de lo que ve, en diferentes lugares del mundo y en diferentes épocas; ha estudiado los cielos y ha registrado sus observaciones. Llamando su primera atención objetos que se movían diferente a las demás estrellas, estos objetos que parecían vagar sin sentido aparente es lo que denominó como “planetas” del griego planētēs que significa “errantes”

El primer modelo que explicó la relación de nuestro planeta con el universo fue el modelo geocéntrico el cual coloca al planeta Tierra en el centro de este. Junto con los demás planetas, el Sol y la Luna orbitando alrededor. De este modo el planeta Tierra es el centro del universo. Este modelo tuvo sus raíces en las observaciones astronómicas realizadas por antiguas civilizaciones como los mesopotámicos, egipcios, griegos e hindúes; así como las civilizaciones mesoamericanas.

Estas observaciones les ayudaron a identificar los ciclos de naturaleza, principalmente para conocer el tiempo conveniente para sembrar y cosechar; el crear un calendario que les permitiera dar un orden a las actividades a realizar. A través de la observación a simple vista, notaron que los cuerpos celestes parecían moverse en el cielo y trataron de dar una explicación a estos movimientos. El astrónomo griego Claudio Ptolomeo fue uno de los principales exponentes del geocentrismo en su obra *Almagesto* en el siglo II D.C. Este modelo sitúa la tierra como centro del universo con los planetas, el sol y la luna girando en torno suyo en órbitas circulares.

En el siglo XVI, el astrónomo polaco Nicolás Copérnico propuso un modelo heliocéntrico, donde el Sol ocupaba el centro del sistema y los planetas giraban a su alrededor. Esta teoría fue desarrollada y perfeccionada por otros

astrónomos como Johannes Kepler, quien en el siglo XVII formuló las leyes del movimiento planetario.

Las leyes de Kepler establecieron que los planetas describen órbitas elípticas alrededor del Sol. Estas leyes proporcionaron una descripción matemática del movimientos planetarios, pero no se ajustaban a los datos que se tenían entonces. Las Leyes son:

Primera Ley: Los planetas orbitan alrededor del Sol en trayectorias elípticas, con el Sol ubicado en uno de los focos de la elipse. Esto significa que la órbita de un planeta no es un círculo perfecto, sino una elipse en la cual el Sol ocupa una posición ligeramente descentrada.

Segunda Ley: Los planetas barren áreas iguales en tiempos iguales durante su movimiento orbital. Esto significa que un planeta se mueve más rápido cuando está más cerca del Sol (en su perihelio) y más lento cuando está más alejado (en su afelio). Esta ley implica que la velocidad orbital de un planeta varía a lo largo de su órbita.

Tercera Ley: El cuadrado del período orbital de cualquier planeta es proporcional al cubo del radio de la órbita.

Matemáticamente:

$$\frac{T^2}{a^3} = K$$

Donde;

T es el tiempo o período orbital

a es el radio o semieje mayor

K es la constante

Fue Isaac Newton quien en el siglo XVII desarrolló la teoría de la gravitación universal. Para entender el movimiento de los cuerpos celestes es necesario entender la mecánica de Newton la cual podemos resumir en tres leyes;

La primera ley nos dice: “Todo cuerpo permanecerá inerte o a velocidad constante, a menos una fuerza externa actúe sobre él”

La Segunda Ley de Newton se expresa matemáticamente como: $F = ma$ ó también como: $\vec{F} = \frac{d(m\vec{v})}{dt} = \frac{d\vec{p}}{dt}$. Donde $\vec{p} = m\vec{v}$. Esta formulación nos permite comprender cómo los planetas orbitan alrededor del Sol, cómo las lunas orbitan alrededor de los planetas y cómo los satélites artificiales se mantienen en órbita alrededor de la Tierra. Nos permite calcular las trayectorias y velocidades de los cuerpos celestes, así como predecir sus posiciones en el tiempo. Esta ley es fundamental para la mecánica porque permite comprender cómo es el movimiento de los cuerpos.

Newton postuló que todos los cuerpos se atraen entre sí con una fuerza proporcional a sus masas e inversamente proporcional al cuadrado de la distancia que los separa, esto se puede expresar matemáticamente como:

$$F \sim \frac{m * M}{r^2}$$

Donde:

- M y m son las masas de los cuerpos
- r La distancia que los separa.

Esta teoría unificó los movimientos celestiales con los fenómenos terrestres, y permitió explicar la órbita de los planetas y la Luna.

Más adelante se abordarán los siguientes temas para comprender este fenómeno:

- Leyes de Newton
- Problemas de n cuerpos
- Ecuaciones diferenciales de Orden Superior
- Métodos de solución
- Gráficas de la solución
- Interpretación física de la solución
- Gravitación

Capítulo 1

Historia del estudio del cielo

1.0.1. Los griegos

El estudio del cielo nocturno existe desde quizá los inicios de la civilización humana: un gran número de civilizaciones en la antigüedad han dejado registro de esta evidencia. Es posible que desde que el ser humano ha existido, se haya dedicado a observar, registrar y tratar de entender el mundo que lo rodea, en especial lo relativo a los cuerpos celestes. Estudiar las posiciones de estos objetos tuvo para los antiguos un contexto mágico; se creía que en las estrellas estaba escrito el destino Silvia [10]. La mitología y la arqueología dan evidencia de cómo estas civilizaciones fueron influenciadas por el movimiento de los astros.

Al observar noche tras noche el cielo, los seres humanos de la antigüedad dejaron registro del movimiento de los planetas que pudieron identificar. Es claro que no disponían más que de su vista y una gran paciencia para darse cuenta de estos movimientos. En un principio la Luna, el Sol, Mercurio, Venus, Marte, Júpiter y Saturno fueron considerados como planetas Silvia [10].

Para Aristóteles, la Tierra estaba en reposo y los planetas giraban circularmente alrededor de ella; las estrellas estaban fijas a una esfera que contenía todo este movimiento lo explicó con su teoría del “primer motor” con la cual atribuía el movimiento a fuerzas metafísicas más allá de las estrellas. Estas ideas fueron aceptadas por la Iglesia, la cual supuso estas fuerzas místicas como ángeles. Hawking [5]

Ptolomeo, de igual modo, concibió el centro de la Tierra como centro del universo, atribuyendo que todo tiende a caer a este centro y por ende, la Tierra estaba inmóvil. “El Universo era un sistema en donde los cuerpos celestes se movían en circunferencias, alrededor de la circunferencia de sus epiciclos (un círculo en que el planeta se mueve y cuyo centro se desplaza simultáneamente a lo largo de un círculo de radio mayor).” Hawking [5].

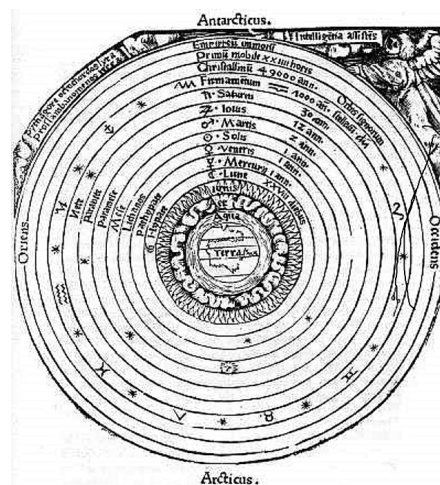


Figura 1.1: Modelo Aristotélico.

“La cristiandad tuvo pocos problemas con el modelo geocéntrico ptolemaico, que dejaba espacio en el universo, más allá de las estrellas fijas, para acomodar un cielo y un infierno, de manera que la Iglesia adoptó este modelo del universo como una verdad establecida.”. Hawking [5].

Las ideas de Aristóteles y Ptolomeo acerca del cosmos, que apoyaban la teoría geocéntrica del Universo, estuvieron vigentes durante más de mil años. Esto no significa que no existiera una teoría heliocéntrica; El astrónomo y matemático griego Aristarco de Samos (310-230 A.C) fue el primero en proponer esta visión. Sin embargo sus ideas no fueron aceptadas sino hasta 1514, cuando un sacerdote polaco empezó a considerar sus ideas.

1.0.2. Nicolás Copérnico(1473-1543)

Nicolás Copérnico, sacerdote, matemático y fundador de la astronomía moderna, retomó el modelo heliocéntrico del universo, el cual coloca al Sol en el centro del sistema solar, con los demás planetas a su alrededor. Sin embargo, este modelo enfrentó grandes críticas. Quizás porque quitaba del centro del universo a la creación, lo cual no le gustaba a la Iglesia que tenía mucha influencia sobre la cultura de aquel entonces.

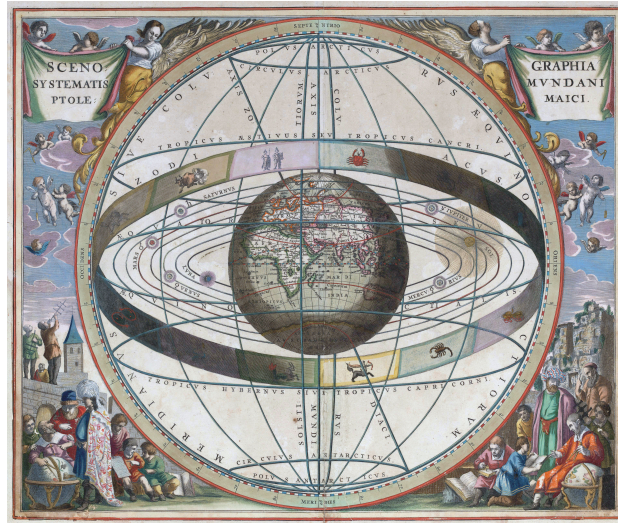


Figura 1.2: Extracto de Harmonia Macrocosmica de Andreas Cellarius, 1660/61. Figura que muestra los signos del zodiaco y el sistema solar con la Tierra en el centro, según el sistema de Ptolomeo.

“Copérnico lo propuso meramente como un modelo para calcular las posiciones de los planetas porque temía que la Iglesia le condenara de hereje si lo proponía como una auténtica descripción de la realidad. A través de sus estudios de los movimientos planetarios, Copérnico llegó a convencerse de que la Tierra era un planeta más y que el Sol estaba en el centro del universo, hipótesis que se conoce como modelo heliocéntrico.” . Copérnico desarrolló su teoría casi en secreto en su obra: “De revolutionibus orbium coelestium.” (Sobre las revoluciones de las esferas celestes), la cual al ser publicada, fue alterada. “Andreas Osiander. Este, esperando apaciguar a los partidarios de la teoría geocéntrica, introdujo algunas alteraciones sin el conocimiento y consentimiento de Copérnico: introdujo la palabra «hipótesis» en la portada, borró párrafos importantes y añadió frases que diluían el impacto y la certeza de la obra.” Hawking [5]. Se dice que Copérnico recibió un ejemplar de la publicación de su obra. La teoría no fue retomada sino mucho después. “Sus ideas permanecieron en una relativa oscuridad durante casi cien años, pero el siglo XVII vio cómo gente de la talla de Galileo Galilei, Johannes Kepler e Isaac Newton construían teorías de universos heliocéntricos, apartando definitivamente las ideas aristotélicas.” Hawking [5].

1.0.3. Galileo Galilei(1564-1642)

Galileo Galilei retomó el modelo heliocéntrico del universo. Expuso sus ideas, apoyadas en sus investigaciones a pesar de estar en contra de lo establecido, en una época donde la Iglesia tenía un control absoluto sobre el conocimiento científico y por lo cual era peligroso para Galileo. Por lo que no fue de esperar que tuviera problemas.

“En 1633, noventa años después de la muerte de Copérnico, el astrónomo y matemático italiano Galileo Galilei fue llevado a Roma para ser juzgado por herejía ante la Inquisición. Los cargos procedían de la publicación de su libro: 'Diálogo sobre los dos principales sistemas del mundo: ptolemaico y copernicano'. En este libro, Galileo declaró enérgicamente, desafiando un edicto de 1616 contra la difusión de la doctrina copernicana, que el sistema heliocéntrico no era una simple hipótesis, sino que correspondía a la verdad.” Hawking [5].

Por esto fue juzgado por herejía y obligado a retractarse, pero continuó exponiendo sus investigaciones cautivando durante siglos a científicos y estudiosos a buscar la verdad a pesar de las circunstancias. Hawking [5].

“Después de todo, lo que había llevado a Galileo ante la Inquisición fue su publicación de 'Diálogo sobre los dos principales sistemas del mundo: ptolemaico y copernicano' dos máximos sistemas del mundo, un desafío frontal al edicto eclesiástico de 1616 que prohibía enseñar la teoría copernicana de la Tierra en movimiento alrededor del Sol como más que una hipótesis.” Hawking [5].

Galileo Galilei nació el 15 de febrero de 1564 en Pisa. A temprana edad mostró gran interés por las matemáticas y la mecánica. Su padre lo impulsó a estudiar medicina en 1581 en la Universidad de Pisa. Realmente él no tenía ningún interés en estudiar medicina, así que empezó a estudiar matemáticas.

“En 1586, había empezado a cuestionar la ciencia y la filosofía de Aristóteles, prefiriendo reexaminar los trabajos del gran matemático Arquímedes.” Hawking [5], “Arquímedes también obtuvo una gran reputación por su invención de numerosos artefactos utilizados como máquinas bélicas, como catapultas gigantes para lanzar piedras contra ejércitos en marcha y grandes grúas para volcar barcos. Galileo se inspiró sobre todo en el genio matemático

de Arquímedes.” Hawking [5].

En 1589 fue contratado como profesor de matemáticas en la Universidad de Pisa, donde tenía que enseñar sobre la teoría de Ptolomeo y Aristóteles. También empezó a refutar a Aristóteles en su escrito: “Sobre el movimiento”.

“En 1604, una supernova observada en el cielo nocturno de Padua renovó sus dudas acerca del modelo aristotélico de un cielo inmutable. Galileo se lanzó a la primera línea del debate, con conferencias provocadoras, pero dudó si debía o no publicar sus teorías.” Hawking [5].

“En octubre de 1608, un holandés llamado Hans Lipperhey solicitó una patente para un catalejo que hacía que objetos lejanos parecieran próximos.” Hawking [5]. Galileo se dedicó a mejorar este artilugio, con el cual abrió camino en la observación del cielo, descubriendo que la luna no era perfectamente redonda como proponía Aristóteles sino que tenía cráteres, montañas, etc. De igual manera se dedicó a observar y estudiar otros cuerpos celestes. “Avistó cuatro lunas alrededor de Júpiter, un descubrimiento que tenía implicaciones tremendas para los partidarios del geocentrismo, que defendían que todos los cuerpos celestes giraban exclusivamente alrededor de la Tierra. Aquel mismo año publicó el Nuncio sidereo (Sidereus Nuncius) en que explicaba sus descubrimientos, que le situaron al frente de la astronomía contemporánea.” Hawking [5].

Galileo Galilei cuestionó el conocimiento de su época, en los libros: “Los dos máximos sistemas del mundo presenta la forma de una polémica entre un partidario de Aristóteles y Ptolomeo y uno de Copérnico, que intentan ganar para sus respectivas filosofías a un hombre ilustrado corriente.”, Hawking [5] y “Dos nuevas ciencias representó una contribución tan grande a la física, que los estudiosos afirman que el libro se anticipó a las leyes del movimiento de Isaac Newton.” Hawking [5], revolucionó la forma de estudiar las estrellas con su telescopio, dando pruebas científicas de que la Tierra gira alrededor del sol, junto con los demás planetas, descubrió la existencia de las lunas de Júpiter, estudió la luna, sentando las bases de la astronomía y la física modernas.

1.0.4. Johannes Kepler

Johannes Kepler(1571-1630), Fue un astrónomo alemán con una gran obsesión por las mediciones, “Llegó a dar las tablas astronómicas más exactas de su tiempo” Hawking [5]. Inspirado por el trabajo de Copérnico se dedicó al estudio del movimiento planetario. Kepler dedicó su vida al estudio del movimiento de los planetas, en un tiempo en el que la filosofía, la religión, la superstición y la ciencia eran vistos como lo mismo. En ese entonces, no había distinción entre la astrología y la astronomía; Kepler vivía de esta ciencia, la astrología era una profesión muy respetada con una buena fuente de ingresos.

“Una de las obligaciones de Kepler como matemático en Graz era componer un calendario astrológico completo con predicciones, práctica muy común en aquel tiempo.” Hawking [5]. Además utilizaba la astronomía junto con las matemáticas para poder entender el movimiento de los planetas.

Retomando viejas ideas de Aristóteles, Kepler vislumbró un modelo del movimiento, en el cual se conjugaba tanto una visión religiosa, filosófica, como también científica.

“Un día, mientras daba una lección de geometría en Graz, Kepler tuvo una súbita revelación que le lanzó a un viaje apasionado y que cambiaría el curso de su vida. Se trataba, creyó él, de la llave secreta para la comprensión del universo. Se le ocurrió que la razón de los círculos indicaría la razón de las órbitas de Saturno y de Júpiter. Inspirado por esta revelación, supuso que los seis planetas conocidos en aquel tiempo estaban dispuestos alrededor del Sol de tal manera que ciertas figuras geométricas ajustaran perfectamente entre ellas. Inicialmente ensayar esta hipótesis sin éxito, utilizando figuras planas bidimensionales como el pentágono, el cuadrado y el triángulo. Entonces acudió a los sólidos pitagóricos, utilizados por los antiguos griegos, que descubrieron que tan sólo cinco poliedros podían ser construidos mediante figuras geométricas regulares. Para Kepler, aquello explicaba por qué sólo podía haber seis planetas (Mercurio, Venus, la Tierra, Marte, Júpiter y Saturno) con cinco espacios entre ellos, y por qué sus separaciones no eran uniformes. Esta teoría geométrica sobre las órbitas y distancias planetarias impulsó a Kepler a redactar el "Misterio del Cosmos"(Mysterium Cosmographicum), publicado en 1596. Tardó un año en escribirlo, y aunque el esquema resultaba razonablemente preciso, estaba muy seguro de que sus

teorías se demostrarían finalmente correctas.

“Nunca podré expresar con palabras cuán intenso fue mi placer al efectuar este descubrimiento. Ya no lamenté el tiempo que me había costado. Consumía días y noches en cálculos, para comprobar si esta idea concordaba con las órbitas copernicanas, o si mi alegría sería llevada por el viento. Al cabo de unos pocos días todo encajaba, y vi cómo un planeta tras otro se situaban con precisión en su lugar.” Hawking [5].

Kepler pasó el resto de su vida intentando obtener la demostración matemática que justificara su teoría y sus observaciones científicas. “Misterio del cosmos” fue la primera obra decididamente copernicana publicada desde el tratado de Copérnico “De revolutionibus”. Hawking [5].

En ese tiempo no era nada fácil acceder a datos de observaciones astronómicas. Por lo que tuvo que trabajar para Tycho Brahe. Un gran observador astronómico que dedicó muchos años a catalogar y medir el comportamiento de los cuerpos celestes, Tycho Brahe poseía una gran base de datos de la época, poder acceder a datos observacionales era muy difícil en aquel entonces, Brahe tenía los datos pero “no tenía las aptitudes matemáticas ni analíticas necesarias para comprender el movimiento planetario”. Hawking [5].

Brahe contrató a Kepler para poder estudiar sus observaciones sobre el movimiento de Marte, que había confundido a los astrónomos durante años. Muchos habían tratado de ajustar tal movimiento a una circunferencia, sin tener éxito, Kepler pudo ajustar los datos a una elipse rompiendo con la idea de las órbitas circulares del modelo aristotélico. Esto se establece como lo que conocemos hoy como La Primera Ley de Kepler.

“En 1605, Kepler anunció su primera ley, la ley de las elipses, que establece que los planetas se mueven en elipses que tienen el Sol en uno de sus focos. La Tierra, dijo Kepler, está más cerca del Sol en enero y más lejos en julio durante su viaje por su órbita elíptica. Su segunda ley, la ley de la igualdad de las áreas, establece que los planetas corren menos cuando están más lejos. Kepler lo demostró arguyendo que una línea imaginaria entre un planeta cualquiera y el Sol debería barrer áreas iguales en tiempos iguales. Publicó ambas leyes en 1609 en su libro Nueva astronomía (Astronomía nova).” Hawking [5].

La Tercera Ley es una relación entre el periodo orbital y la distancia. A diferencia de las leyes anteriores esto no es tan fácil de visualizar. Se puede entender gracias a su expresión matemática.

$$\frac{T^2}{a^3} = \text{constante}$$

donde T es el período orbital y a es el semieje mayor de la órbita.

“Las armonías del mundo el 27 de mayo de 1618. En esta serie de cinco libros, extendió su teoría de la armonía a la música, la astrología, la geometría y la astronomía. La serie incluyó su tercera ley de los movimientos planetarios, que unos sesenta años más tarde inspirará a Isaac Newton, y que establece que los cubos de las distancias medias de los planetas al Sol son proporcionales a los cuadrados de sus períodos de revolución. En síntesis, Kepler descubrió cómo orbitan los planetas, y al conseguirlo, allanó el camino para que Newton descubriera el porqué.” Hawking [5].

Kepler pudo determinar cómo es el movimiento de los cuerpos celestes relacionándolo con las cónicas; más tarde Newton determinó el origen de las fuerzas causantes del movimiento.

1.0.5. Issac Newton (1643-1727)

Issac Newton nació en la Navidad de 1642, mismo año de la muerte de Galileo Galilei, en la ciudad de Woolsthorpe. Su infancia fue difícil por la ausencia de su padre, quien murió antes de su nacimiento. Además de tener una relación distante con su madre y no le iba muy bien en la escuela, pero desde temprana edad mostró interés por la mecánica. “Newton pasó incontables horas construyendo relojes, cometas llameantes, relojes de sol y molinos en miniatura (movidos por ratoncitos), además de dibujar detallados esbozos de animales y barcos.” Hawking [5].

Como no le iba bien en la escuela, lo pusieron a trabajar en el campo; afortunadamente su tío William Ayscough, que era un clérigo muy reconocido, influyó para que Newton volviera a la escuela en Grantham y continuara sus estudios, lo cual lo llevó eventualmente a la Universidad de Cambridge.

“Esta vez, Newton vivió con el director de la escuela libre de gramática, John Stokes, cosa que supuso un cambio de rumbo en la educación. Se dice que, de alguna manera, un golpe en la cabeza que le propinó un matón del patio le iluminó, y le permitió corregir el rumbo negativo de sus perspectivas escolares. Mostrando ahora aptitudes intelectuales y curiosidad, Newton empezó a prepararse para proseguir los estudios en una universidad. En Trinity, Newton recibía una ayuda para pagar el coste de su educación a cambio de hacer diversos trabajos, como servir las mesas y limpiar habitaciones para la facultad. Pero en 1664 fue admitido como becario, lo que le garantizó apoyo económico y le permitió liberarse de las tareas domésticas. Cuando la universidad cerró a causa de la epidemia de fiebre bubónica en 1665, Newton se retiró a Lincolnshire. En los dieciocho meses que pasó en casa durante la epidemia se dedicó por su cuenta a la mecánica y las matemáticas, y empezó a concentrarse en óptica y gravitación. Este «*annus mirabilis*» («año milagroso»), como Newton lo llamó, fue uno de los períodos más productivos y fértiles de su vida.” Hawking [5] Isaac Newton se desarrolló en varias áreas, como la teoría de la luz y el color, en mecánica, óptica y matemáticas; es considerado como padre del cálculo infinitesimal. Lo que aseguró su lugar en la historia fue la formulación de la gravitación universal. Hawking [5]. Las Leyes del Movimiento y la Ley de Gravitación Universal cambiaron para siempre nuestra comprensión del movimiento y las fuerzas en el Universo, permitieron explicar fenómenos que anteriormente eran mal comprendidos y unificó los movimientos celestiales con los fenómenos terrestres permitiendo explicar las órbitas de los planetas y la Luna. Anteriormente Kepler respondió la pregunta de ¿cómo es el movimiento de los cuerpos celestes?, Newton pudo responder ¿por qué es el movimiento de los cuerpos celestes?. Todo esto lo plasmó en su obra maestra Principios Matemáticos de Filosofía Natural (*Philosophiae Naturalis Principia Mathematica*), conocida generalmente como los Principia. “En ella, fundió las contribuciones científicas de Copérnico, Galileo, Kepler y otros en una gran sinfonía dinámica. Los Principia, el primer libro de física teórica, es unánimemente considerado como la obra más importante de la historia de la ciencia y el fundamento científico de la moderna visión del mundo.” Hawking [5].

Capítulo 2

Problema de kepler

El problema de Kepler radica en cómo es el comportamiento de un sistema en el cual interactúan dos cuerpos en el espacio. Podemos considerar el centro de un planeta como el punto donde se concentra la fuerza de atracción gravitacional, esto para poder expresarlo de forma vectorial. La forma en que interactúan los cuerpos, lo podemos entender usando La Segunda Ley de Newton y La Ley de Gravitación Universal. Esto por supuesto lo ignoraba Kepler y sus contemporáneos, los cuales solo poseían la observación. La contribución de Kepler, sus tres leyes fueron deducidas gracias a la observación, especulación y análisis de datos, lo que le tomó años de arduo trabajo. "Kepler fue el primero en proponer las leyes que definen las órbitas a partir de las observaciones empíricas del movimiento de Marte apoyadas, en gran parte en las observaciones astronómicas realizadas por Tycho Brahe" (Francisco García). Posteriormente Newton al introducir sus ideas, dio un panorama más completo, pudo describir cómo es este movimiento, desde una perspectiva mecánica. "Unificando el movimiento de los cuerpos terrestres con el movimiento de los cuerpos celestes" Hawking [5], dando lugar a lo que llamamos Mecánica Celeste.

La fuerza de gravitación actúa entre dos cuerpos. Por ejemplo; dos planetas, uno mil veces la masa del otro, la luna y la Tierra, una nave espacial y la Tierra, etc. Cuando se pretende trabajar con un sistema de 3 cuerpos, se debe considerar interacciones de dos a dos. Por ejemplo, sean tres planetas: p_1 , p_2 , p_3 . Existirá una fuerza f_1 de p_1 a p_2 , una fuerza f_2 de p_2 a p_1 , una fuerza f_3 de p_1 a p_3 , una fuerza f_4 de p_3 a p_1 , una fuerza f_5 de p_2 a p_3 , y una fuerza f_6 de p_3 a p_2 (Figura 2.1)

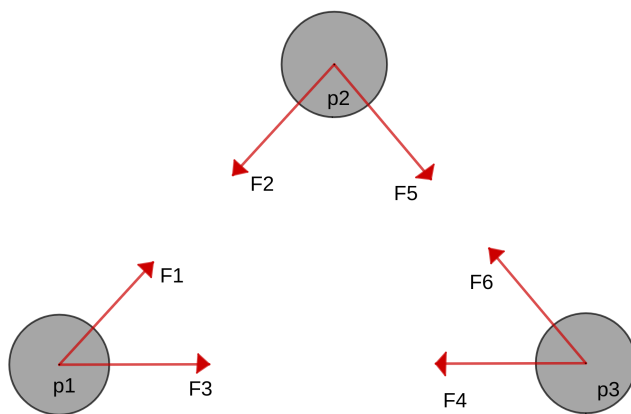


Figura 2.1: La fuerza de gravitación entre dos cuerpos es de dos a dos.

Considere un sistema con un planeta y una estrella, en un tiempo inicial el planeta posee una velocidad inicial, estas son las condiciones iniciales es decir para un tiempo $t=0$. El planeta tendrá uno de tres posibles comportamientos, podrá describir una trayectoria parabólica tal que medio rodeará al Sol y después seguirá disparado al infinito; podrá terminar dando vueltas alrededor del Sol de tal manera que la distancia entre los dos sea cada vez más pequeña hasta colisionar o podría terminar en una trayectoria, tal vez circular o elíptica de forma continua. Este mismo comportamiento lo veremos en una nave que es afectada por la fuerza gravitacional de un planeta. Sea el sistema Tierra-Luna, Tierra-Luna y una nave, etc. La cuestión es: ¿Como son las trayectorias?

Sabemos que un sistema de dos cuerpos, tres cuerpos, n -cuerpos estarán sometidos a la segunda Ley de Newton y la Ley de Gravitación Universal. Como veremos más adelante de estas leyes y sus expresiones matemáticas podemos construir ecuaciones diferenciales y al resolver estas ecuaciones, ya sea con Métodos Numéricos o analíticamente, nos darán fórmulas de cómo son las trayectorias, las velocidades, si hay aceleración o no la hay; nos ayudará a entender el movimiento. En lo que más nos vamos a enfocar en este trabajo es en la parte de Métodos Numéricos. Al solucionar las ecuaciones con Métodos Numéricos podremos visualizar una simulación meramente numérica del comportamiento.

Lo interesante aquí es como tomando dos leyes que rigen un comportamiento en la naturaleza, es posible replicarlo en una computadora.

¿Para que nos serviría todo esto? La principal aplicación es poner un satélite en órbita; este caso sería el de dos cuerpos, donde uno es masivo y el otro pequeño. La simulación numérica nos ayudará a saber las condiciones iniciales necesarias para que, una vez que la nave haya conseguido salir del planeta, este pueda mantenerse en órbita; Para ello necesitamos conocer su masa, su posición inicial y un cálculo lo suficientemente preciso de la velocidad inicial que requiere la nave para mantenerse en órbita.

2.1. Problema Inicial

2.1.1. Explicación

Se tiene una nave espacial a una distancia inicial de la Tierra r_0 . En un Tiempo $t = 0$ la nave enciende sus motores obteniendo una velocidad inicial v_0 . La nave está sometida a la fuerza gravitacional de la tierra, definida por la Ley de Gravitación Universal. Calcule la distancia y la velocidad inicial de la nave para que esta se mantenga en una órbita cerrada alrededor de la tierra.

2.1.2. Formulación Matemática sobre las Leyes del movimiento celeste

La Fuerza de gravitación universal, es dos a dos. Sea F la fuerza que experimenta la nave. Como está sometida a la fuerza de atracción de la Tierra y la Luna F sera expresada como;

$$\vec{F} = \vec{F}_M + \vec{F}_E \quad (2.1)$$

Donde:

$$\begin{aligned} \vec{F} &= \text{Fuerza total que experimenta la nave,} \\ \vec{F}_M &= \text{Fuerza ejercida por la Luna,} \\ \vec{F}_E &= \text{Fuerza ejercida por la Tierra.} \end{aligned}$$

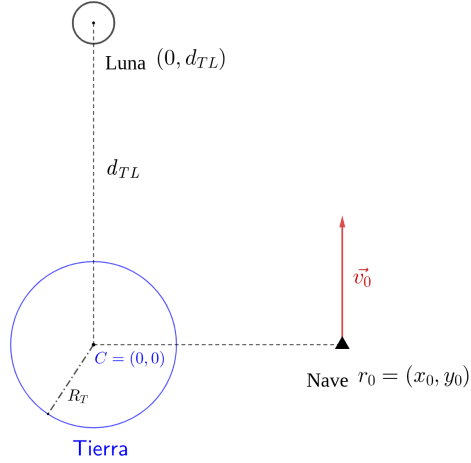


Figura 2.2: Sistema Tierra-Luna-Nave con los datos del problema. El sistema está en coordenadas cartesianas, el centro del sistema es el centro de la Tierra.

Por la Segunda Ley de Newton $\vec{F} = m\vec{a}$ y por la Ley de Gravitación Universal;

$$\vec{F} = m\vec{a} = -G \frac{mm_e}{|\vec{r} - \vec{r}_e|^3} (\vec{r} - \vec{r}_e) - G \frac{mm_m}{|\vec{r} - \vec{r}_m|^3} (\vec{r} - \vec{r}_m) \quad (2.2)$$

Donde;

- m : masa de la nave
- m_e : masa de la Tierra
- m_m : masa de la Luna
- $\vec{r}$: vector posición de la nave
- $\vec{r}_e$: vector posición de la Tierra
- $\vec{r}_m$: vector posición de la Luna
- G : constante de Gravitación Universal

De la ecuación (2) se eliminará la masa de la nave m . Reescribiéndola en componentes cartesianas (x,y);

$$\ddot{x} = -Gm_e \frac{(x - x_e)}{d_e^3} - Gm_m \frac{(x - x_m)}{d_m^3}, \quad \ddot{y} = -Gm_e \frac{(y - y_e)}{d_e^3} - Gm_m \frac{(y - y_m)}{d_m^3} \quad (2.3)$$

Donde d_e es la distancia entre la Nave y la Tierra. d_m es la distancia entre la Nave y la Luna;

$$d_e = \sqrt{(x - x_e)^2 + (y - y_e)^2}$$

$$d_m = \sqrt{(x - x_m)^2 + (y - y_m)^2}$$

Se considerarán los siguientes datos.

Símbolo	Descripción
$d_{tl} = 3.844 \times 10^8$ m	Distancia entre la Tierra y la Luna
$R_L = 1337$ km	Radio de la Luna
$R_T = 6371$ km	Radio de la Tierra
$m_e = 5.9742 \times 10^{24}$ kg	Masa de la Tierra
$m_m = 7.35 \times 10^{22}$ kg	Masa de la Luna
$G = 6.6726 \times 10^{-11}$ N·m ² /kg ²	Constante de gravitación universal
$T = 27.322$ días	Período orbital de la Luna (tiempo en completar una órbita alrededor de la Tierra)

2.1.3. Resolución del problema por métodos numéricos

El siguiente sistema de ecuaciones describe como es la aceleración, en componentes cartesianas de una masa puntual m sometida a las fuerzas gravitacionales de la Tierra y la Luna.

$$\ddot{x} = -Gm_e \frac{(x - x_e)}{d_e^3} - Gm_m \frac{(x - x_m)}{d_m^3}, \quad \ddot{y} = -Gm_e \frac{(y - y_e)}{d_e^3} - Gm_m \frac{(y - y_m)}{d_m^3}$$

Para resolver este sistema de ecuaciones por métodos numéricos, primero se realizó el cambio de variable $\ddot{r} = \dot{v}$. De esta forma, en lugar de tener un sistema de ecuaciones de segundo orden con dos incógnitas, se obtiene un sistema de ecuaciones de primer orden. Este sistema se resolverá utilizando el método numérico de Runge-Kutta de cuarto orden.

Entonces se resolverá el sistema;

$$\begin{aligned}\ddot{x} = \dot{v}_x &= f_1(x, y, v_x, v_y, t) \\ \ddot{y} = \dot{v}_y &= f_2(x, y, v_x, v_y, t) \\ \dot{x} &= f_3(x, y) \\ \dot{y} &= f_4(x, y)\end{aligned}$$

Con las condiciones iniciales:

$$x(0) = x_0, \quad y(0) = y_0, \quad v_x(0) = v_{x0}, \quad v_y(0) = v_{y0}.$$

Para encontrar las condiciones iniciales de posición y velocidad que debe poseer la nave, para que su trayectoria sea una órbita. Se planteó probar varias condiciones iniciales, hasta encontrar las condiciones idóneas para las cuales se observa la órbita. Esto gracias a la capacidad de la computación para resolver operaciones aritméticas con gran velocidad.

2.2. Normalización

La normalización es un proceso de re-escalamiento de un conjunto de datos relacionados entre sí. Homólogo a la conversión de unidades pero a diferencia de la conversión de unidades, que se aplica a un solo dato, la normalización implica re-escalar un conjunto completo de datos relacionados junto con un sistema de ecuaciones. Este proceso tiene como objetivo facilitar el análisis o en nuestro caso el cálculo numérico, al ajustar todos los elementos del sistema a una escala común. Esta escala es elegida según las necesidades del problema.

El objetivo de normalizar un sistema físico es facilitar el cálculo numérico a la computadora. Por ejemplo no es lo mismo utilizar la masa de la Tierra $Mt = 5,97 \times 10^{24}$, a usar $Mt = 1$. Estos 24 ceros harán más pesado el cálculo numérico considerando que la computadora opera en Binario. Sin embargo el utilizar la masa de la Tierra igual a 1 implica que en el sistema todas las masas se normalicen de igual forma. Es decir todas las masas del sistema, serán divididas entre la masa de la tierra.

Este proceso será fundamentalmente necesario para los sistemas físicos que se estudiarán más adelante, pues aparte de utilizar constantes muy grandes, el cálculo numérico es demasiado pesado.

Antes de comenzar con la normalización, es crucial entender las unidades de medición que se utilizarán. Para este problema se emplea el sistema MKS (metros, kilogramos y segundos). Además es importante tener en cuenta que el re-escalamiento se realizará utilizando las unidades más fundamentales, es decir normalizamos las unidades de distancia, tiempo y masa.

Primero se analizaron las unidades más fundamentales de la constante de Gravitación Universal. Esto por que la normalización partirá desde esta.

Analizando las Unidades fundamentales de G

Si partimos solo de las Unidades que la conforman, esta se puede escribir:

$$[G] = N \frac{m^2}{kg^2}$$

Donde $N = kg \frac{m}{s^2}$, entonces:

$$[G] = kg \frac{m}{s^2} \frac{m^2}{kg^2}$$

$$[G] = \frac{m^3}{s^2 kg}$$

Es decir la constante G se puede representar como:

$G = 6,6726 \times 10^{-11} [\frac{m^3}{s^2 kg}]$ en sus unidades fundamentales.

Observe que G es una cantidad compuesta de aceleración por metro cuadrado entre kilogramo;

$$[\frac{m^3}{s^2 kg}] = \frac{m}{s^2} \frac{m^2}{1} \frac{1}{kg}$$

Vamos a normalizar.

Sea τ un tiempo característico, es decir una escala de tiempo tal que $G = 1$.

Como estamos cambiando la constante de gravitación. Vamos a escribir $G^* = 1$. Diferenciándola del valor real y teniendo en cuenta que G^* es un valor normalizada.

Definiendo:

$$t^* = \frac{t}{\tau}, \quad M^* = \frac{M}{M_T}, \quad r^* = \frac{r}{R_T}$$

De esta forma se esta re-escalando las unidades básicas. Las unidades con * son las unidades normalizadas, sin el * son valores reales. Entonces usando τ como unidad de tiempo, la masa de la Tierra M_T como unidad de masa y el radio de la Tierra R_T como la unidad de distancia.

Es decir:

$$G^* = G \left(\frac{\tau^2}{R_T} \right) \left(\frac{1}{R_T^2} \right) M_T$$

$$G^* = G \left(\frac{\tau^2 M_T}{R_T^3} \right)$$

Si $G^* = 1$ entonces se obtiene el valor del tiempo característico τ .

$$G^* = 1 = G \left(\frac{\tau^2 M_T}{R_T^3} \right) \text{ Entonces } \tau^2 = \frac{R_T^3}{M_T G}.$$

Sustituyendo los valores de G, R_T, M_T ;

$$\begin{aligned} G &= 6.6726 \times 10^{-11} \text{ N} \cdot \text{m}^2 \cdot \text{kg}^{-2} \\ R_T &= 6.731 \times 10^6 \text{ m} \\ M_T &= 5.9742 \times 10^{24} \text{ kg} \end{aligned}$$

$$\tau = 805.715$$

Para el nuevo sistema

De la misma forma vamos a normalizar los demás datos:

$$\begin{aligned} d_{TL}^* &= \frac{d_{TL}}{R_T} = \frac{3.8444 \times 10^8}{6.371 \times 10^6} = 60.33, \\ R_L^* &= \frac{R_L}{R_T} = \frac{1.3337 \times 10^6}{6.371 \times 10^6} = 0.2093, \\ M_M^* &= \frac{M_M}{M_T} = \frac{7.35 \times 10^{22}}{5.9742 \times 10^{24}} = 0.0123 \\ T^* &= \frac{T}{\tau} = \frac{2360448}{805.715} = 2929.6 \end{aligned}$$

2.3. Aplicación

2.3.1. Orbitas de la estación espacial internacional

Retomando el problema 1. Se ha considerado un sistema de tres cuerpos, el cual consiste de la Tierra, una Luna estática, y una nave espacial. Considerando los datos reales de velocidad y algunos rangos de distancia de la Estación Espacial Internacional, se pretende simular las trayectorias de la Estación, con el fin de responder los cuestionamientos de a qué velocidad y distancia se puede garantizar alcanzar una órbita estable para una nave en el problema.

2.3.2. Normalización de los datos

En una búsqueda rápida en internet se han encontrado los siguientes datos de la Estación Espacial Internacional. La velocidad

$$v = 273 \times 10^3 \frac{km}{h}$$

También se encontró que la altura a la que se encuentra esta dentro de un rango de $335km$ a $460km$.

Para normalizar estos datos primero se han de pasar al sistema MKS. (Metros, kilogramos, Segundos.)

$$v = \left(27 \times 10^3 \frac{km}{h} \right) \left(\frac{1000 m}{1 km} \right) \left(\frac{1 h}{3600 s} \right)$$

$$v = 7500 \frac{m}{s}$$

Para la distancia se ha considerado como:

$$D = h + R_t$$

Donde h es la altura de la Estación Espacial Internacional, R_t es el radio de la tierra.

Dado que $h \in [335, 460] km$.

Para $h = 3.35 \times 10^5 m$ resulta $D_0 = 70.66 \times 10^5 m$

Para $h = 4.6 \times 10^5 m$ resulta $D_1 = 71.92 \times 10^5 m$

Normalizando

Sea v^* y d^* los datos normalizados.

$$v^* = v \times (\tau) \times \left(\frac{1}{R_t}\right)$$

$$v^* = \left(7500\frac{m}{s}\right) \times (805.715s) \times \left(\frac{1}{6.73 \times 10^6}\right)$$

$$v^* = 0.8978$$

Para la distancia

$$D_0 = D_0 \times \frac{1}{R_t}$$

entonces $D_0 = 1.0498$ y $D_1 = 1.0685$

2.3.3. Programa y Simulaciones

Simulaciones

A continuación, se muestran los resultados de la simulación (Programa 1, Apéndice A). Este código en fortran 90 al igual que los demás puede modificarse para encontrar más condiciones que generen órbitas.

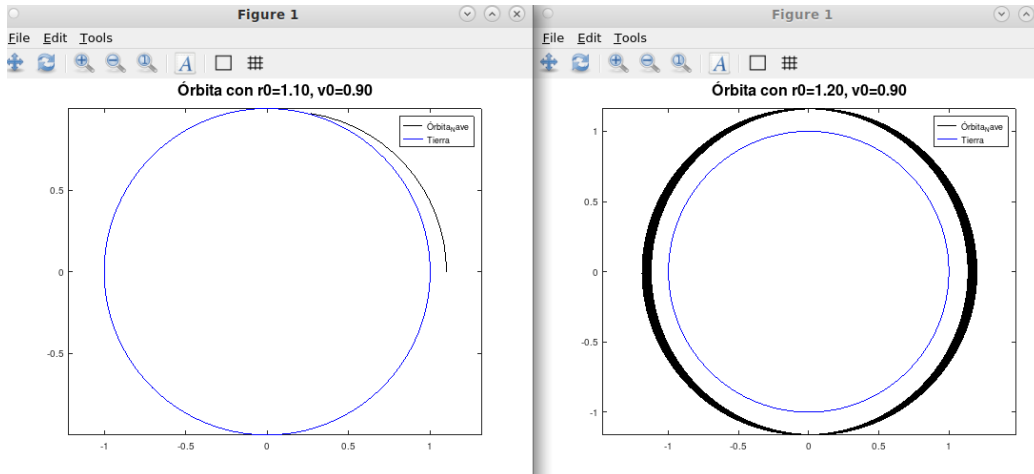
Ejecución del programa desde la terminal Bash en Fedora:

```
[samuelsanchezrubio@fedora]$ f95 orbitas.f90
[samuelsanchezrubio@fedora]$ ./a.out
colisión con la tierra en r=    1.10000002    v=  0.897800028
Archivo generado: datos        1                1 .dat
Posible órbita en r=    1.20000005    v=  0.897800028
Archivo generado: datos        1                2 .dat
Posible órbita en r=    1.29999995    v=  0.897800028
Archivo generado: datos        1                3 .dat
Posible órbita en r=    1.39999998    v=  0.897800028
Archivo generado: datos        1                4 .dat
Posible órbita en r=    1.50000000    v=  0.897800028
Archivo generado: datos        1                5 .dat
Posible órbita en r=    1.60000002    v=  0.897800028
Archivo generado: datos        1                6 .dat
```

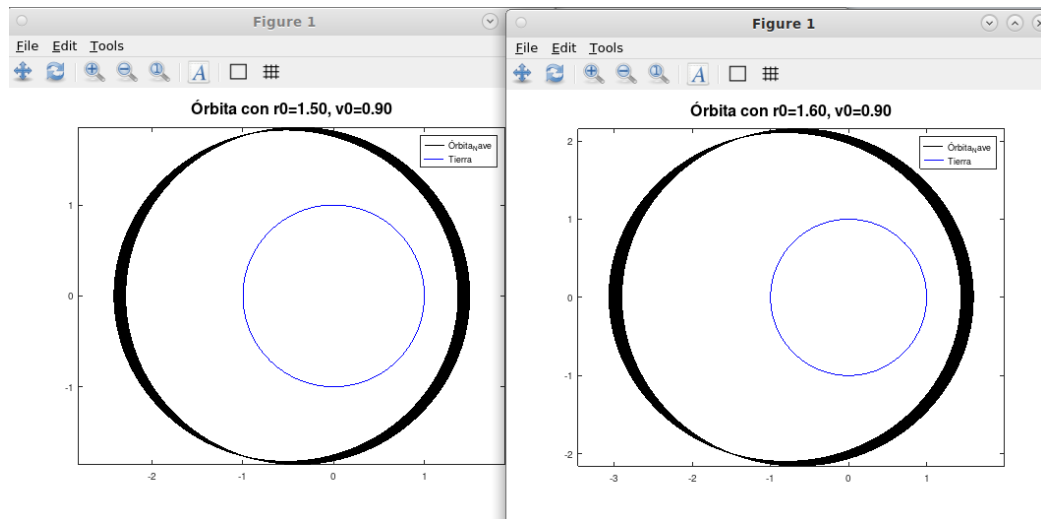
Posible órbita en $r=$	1.70000005	$v=$	0.897800028
Archivo generado: datos	1		7 .dat
Posible órbita en $r=$	1.79999995	$v=$	0.897800028
Archivo generado: datos	1		8 .dat
Posible órbita en $r=$	1.90000010	$v=$	0.897800028
Archivo generado: datos	1		9 .dat
Posible órbita en $r=$	2.00000000	$v=$	0.897800028
Archivo generado: datos	2		10 .dat

Gráficas

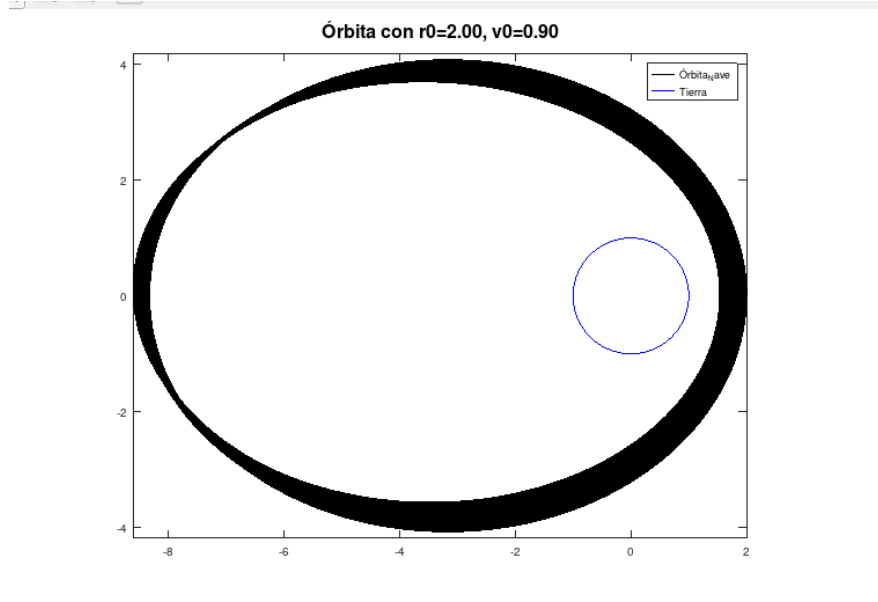
Algunas gráficas de estas simulaciones de mostraran a continuación



Gráficas de los resultados. Trayectoria en negro. Alrededor de la Tierra representada como una circunferencia azul con radio normalizado.



Gráficas de los resultados. Trayectoria en negro. Alrededor de la Tierra representada como una circunferencia azul con radio normalizado.



Gráficas de los resultados. Trayectoria en negro. Alrededor de la Tierra representada como una circunferencia azul con radio normalizado. Aquí claramente se ha alcanzado una órbita elíptica.

2.3.4. Análisis por medio de la Ley de conservación de la Energía

La Ley de conservación de la Energía Mecánica dice que en un sistema sin fricción ni otra fuerza externa, la suma entre la energía cinética y potencial es igual a una constante.

Matemáticamente la energía cinética se expresa como:

$$k = \frac{1}{2} * m * v^2$$

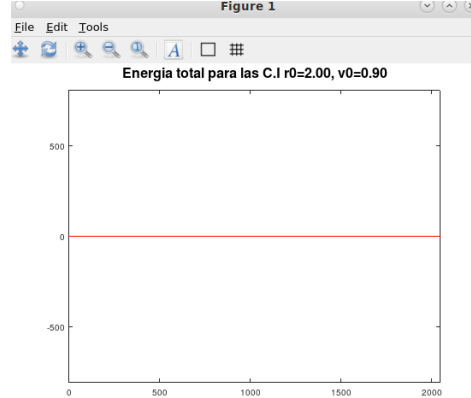
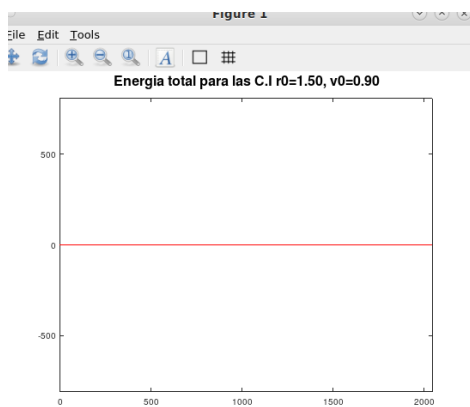
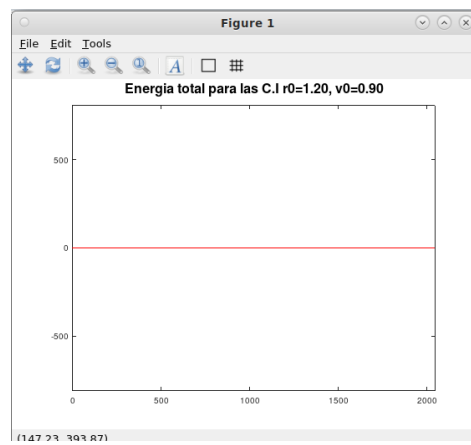
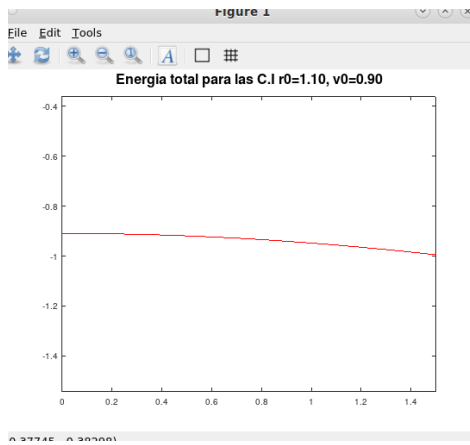
y la energía potencial se expresa como:

$$u = -\frac{G * m_1 * m_2}{r}$$

La Ley de conservación de la energía se expresa matemáticamente como:

$$E_t = k + u = cte$$

Para hacer este análisis de conservación de la energía mecánica, se ha creado una subrutina en fortran, la cual calcula la energía mecánica total de la nave en cada punto calculado.



Gráficas de Energía total en función del tiempo, (Programa 1, subrutina Energía, Apéndice A.)

2.3.5. Buscando Órbitas para una distancia entre la Tierra y la Luna

A continuación, se planteó la búsqueda de la condición de velocidad necesaria para que la nave alcanzara una órbita, partiendo desde una posición inicial situada entre la Tierra y la Luna. Las condiciones iniciales de posición fueron: $x = 30.165$, $y = 0$, $vx = 0$, variando vy desde 0.001 a 0.50 con un incremento de 0.001.

Ejecución desde terminal:

```
[samuelsanchezrubio@fedora P3C(LE)]$ time ./a.out
```

colisión con la tierra en r=	30.1650009	v=	9.99999978E-03	
Archivo generado: datos		30		1 .dat
colisión con la tierra en r=	30.1650009	v=	1.99999996E-02	
Archivo generado: datos		30		2 .dat
colisión con la tierra en r=	30.1650009	v=	2.99999993E-02	
Archivo generado: datos		30		3 .dat
colisión con la tierra en r=	30.1650009	v=	3.99999991E-02	
Archivo generado: datos		30		4 .dat
Posible órbita en r=	30.1650009	v=	4.99999970E-02	
Archivo generado: datos		30		5 .dat
Posible órbita en r=	30.1650009	v=	5.99999987E-02	
Archivo generado: datos		30		6 .dat
Posible órbita en r=	30.1650009	v=	7.00000003E-02	
Archivo generado: datos		30		7 .dat
Posible órbita en r=	30.1650009	v=	7.99999982E-02	
Archivo generado: datos		30		8 .dat
Posible órbita en r=	30.1650009	v=	8.99999961E-02	
Archivo generado: datos		30		9 .dat
Posible órbita en r=	30.1650009	v=	9.99999940E-02	
Archivo generado: datos		30		10 .dat
Posible órbita en r=	30.1650009	v=	0.109999999	
Archivo generado: datos		30		11 .dat
Posible órbita en r=	30.1650009	v=	0.119999997	
Archivo generado: datos		30		12 .dat
Posible órbita en r=	30.1650009	v=	0.129999995	
Archivo generado: datos		30		13 .dat
Posible órbita en r=	30.1650009	v=	0.140000001	
Archivo generado: datos		30		14 .dat
Posible órbita en r=	30.1650009	v=	0.149999991	
Archivo generado: datos		30		15 .dat
Posible órbita en r=	30.1650009	v=	0.159999996	
Archivo generado: datos		30		16 .dat
Posible órbita en r=	30.1650009	v=	0.170000002	
Archivo generado: datos		30		17 .dat
Posible órbita en r=	30.1650009	v=	0.179999992	
Archivo generado: datos		30		18 .dat
Posible órbita en r=	30.1650009	v=	0.189999998	
Archivo generado: datos		30		19 .dat

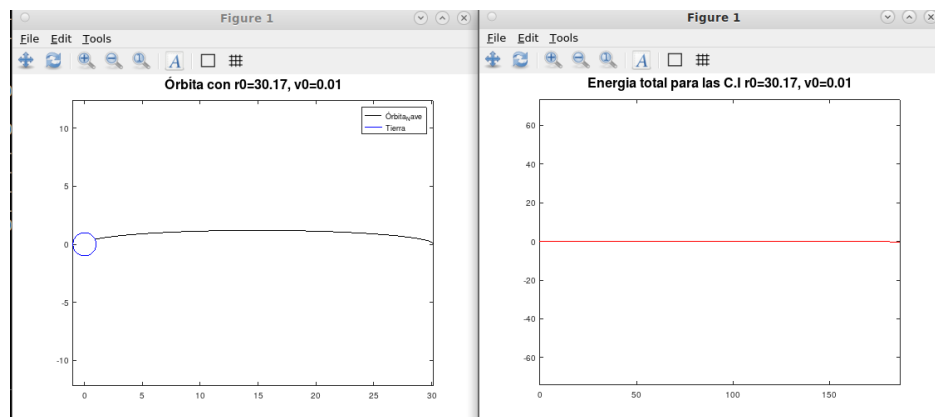
```

Posible órbita en r=    30.1650009    v=  0.199999988
Archivo generado: datos                                30    20 .dat

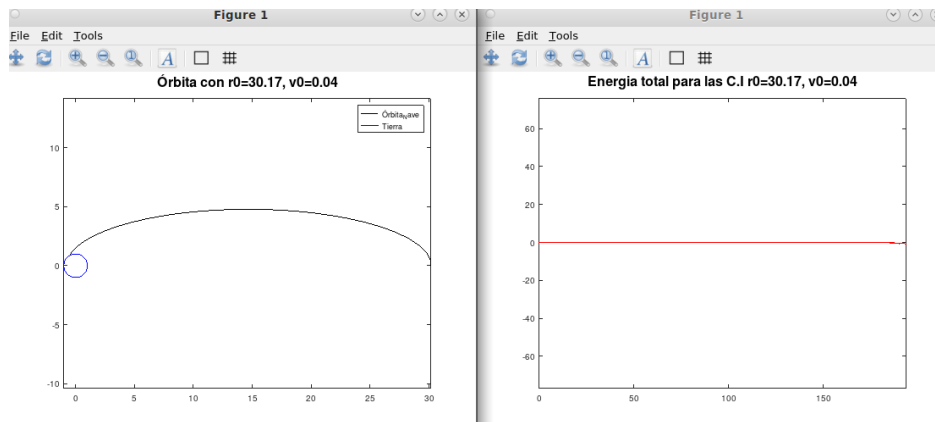
real 18m48.850s
user 18m1.613s
sys  0m4.295s

```

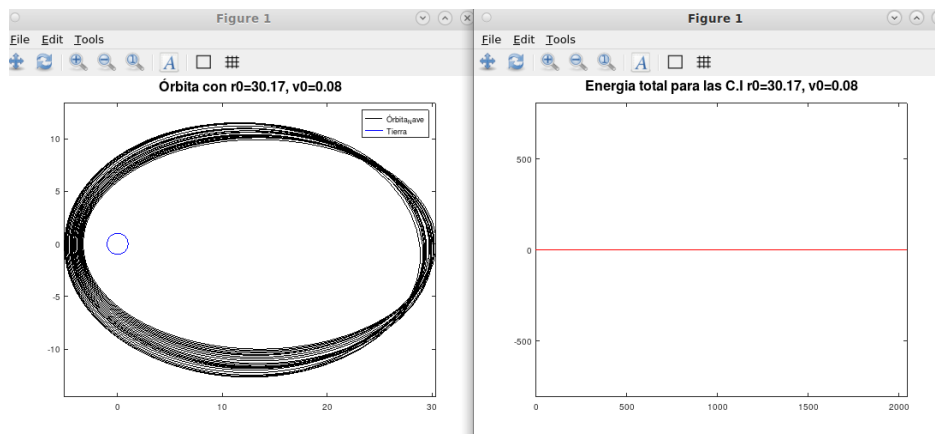
Algunas gráficas de las orbitas y la energía total



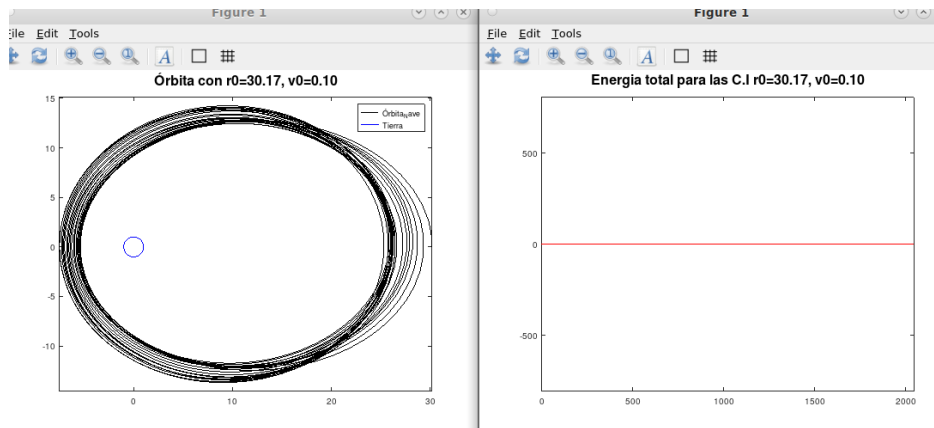
A la derecha. Trayectoria de colisión con la Tierra la cual es representada como una circunferencia de color azul. A la izquierda gráfica de análisis de energía, energía total en función del tiempo de la trayectoria.



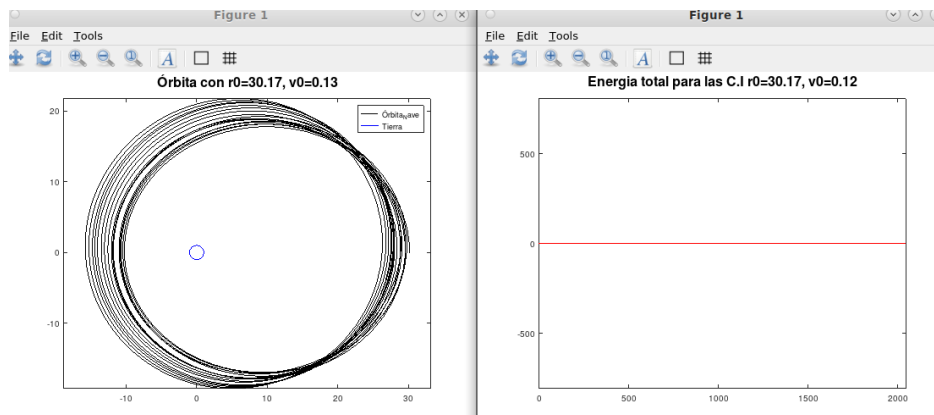
A la derecha. Trayectoria de colisión con la Tierra la cual es representada como una circunferencia de color azul. A la izquierda gráfica de análisis de energía, energía total en función del tiempo de la trayectoria.



A la derecha. Órbita al rededor de la Tierra la cual es representada como una circunferencia de color azul. A la izquierda gráfica de análisis de energía, energía total en función del tiempo de la trayectoria.



A la derecha. Órbita al rededor de la Tierra la cual es representada como una circunferencia de color azul. A la izquierda, gráfica de análisis de energía, energía total en función del tiempo de la trayectoria.



A la derecha. Órbita al rededor de la tierra la cual es representada como una circunferencia de color azul. A la izquierda gráfica de análisis de energía, energía total en función del tiempo de la trayectoria.

2.4. Órbitas con luna en movimiento

Se ha agregado una subrutina para simular el movimiento de la Luna (Programa 1, subrutina Movluna). Se ha supuesto que la Luna se mueve con una trayectoria circular uniforme alrededor de la Tierra. Usando el dato del

periodo de la Luna primero procedemos a racionalizarlo.

Dado que el periodo Lunar es $P = 29.53$ días, necesitamos pasarlo primero a segundos. Para esto multiplicamos por 24, obteniéndolo en horas, luego volvemos a multiplicar por 60, obtenido el periodo en minutos y posteriormente multiplicamos por 60 obteniendo así el periodo en segundos.

$$P = 29.53 * 24 * 60 * 60 = 3166.6s$$

Se ha agregado una subrutina para simular el movimiento de la Luna. Se ha supuesto que la Luna se mueve con una trayectoria circular uniforme alrededor de la Tierra. Usando el dato del periodo de la Luna se ha racionalizado.

Dado que la Luna está variando su posición, es necesario modificar el algoritmo. A continuación se muestra el programa modificado, también se puede observar la subrutina que hace los cálculos de energía.

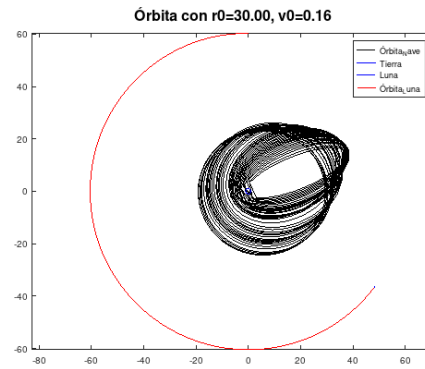
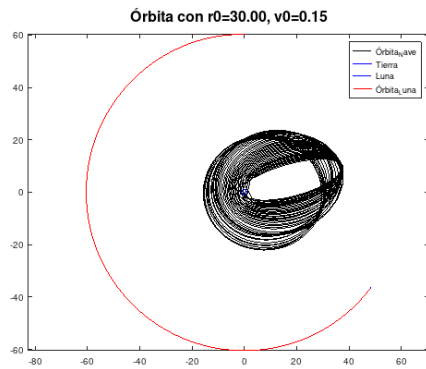
El programa anterior se uso para las condiciones iniciales; $r_0 = 30$ que es la mitad de la distancia entre la Tierra y la Luna y v_0 variando desde 15 a 17.

Este programa tardó más dado la precisión que se esta usando ($h = 1 \times 10^{-4}$).

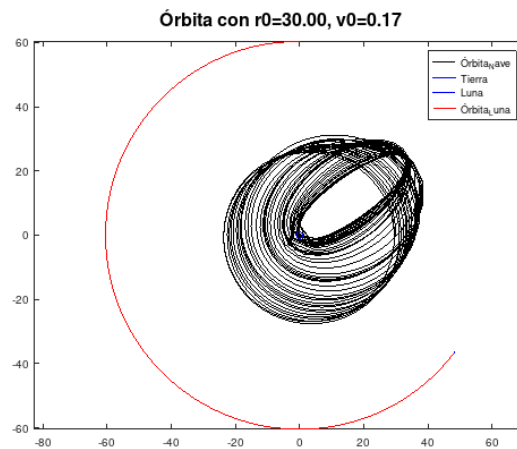
Cuando analizamos la energía total, esta tiende a una constante. **Ejecución desde terminal:**

```
[samuelsanchezrubio@fedora orbitasL]$ time ./a.out
colisión con la tiera en r=    30.0000000    v=  0.149999991
Archivo generado: datos                30                                15 .dat
colisión con la tiera en r=    30.0000000    v=  0.159999996
Archivo generado: datos                30                                16 .dat
colisión con la tiera en r=    30.0000000    v=  0.170000002
Archivo generado: datos                30                                17 .dat

real 18m29.765s
user 11m54.633s
sys 0m1.563s
```



En negro, trayectoria de una nave alrededor de la Tierra bajo la fuerza gravitacional de la tierra y la luna. En rojo, trayectoria de la Luna, simulada con movimiento circular alrededor de la Tierra.



En negro trayectoria de una nave alrededor de la Tierra bajo la fuerza gravitacional de la Tierra y la Luna. En rojo trayectoria de la Luna, simulada con movimiento circular alrededor de la Tierra.

Variando más la velocidad.

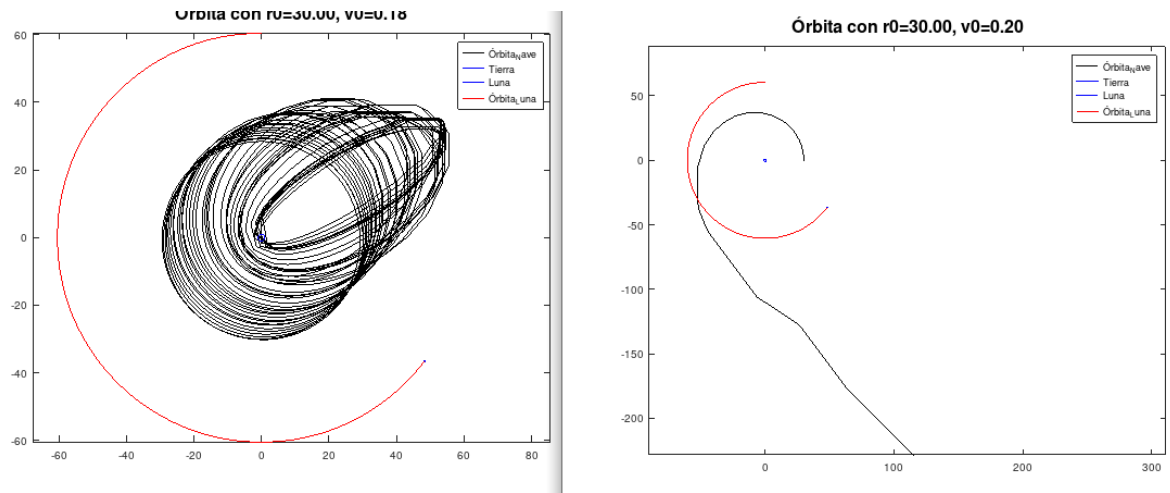


Figura 2.3: En negro trayectoria de una nave alrededor de la Tierra bajo la fuerza gravitacional de la Tierra y la Luna. En rojo trayectoria de la Luna, simulada con movimiento circular alrededor de la Tierra.

Capítulo 3

Gravitación Universal

3.1. Introducción

Las Leyes de Kepler sobre el movimiento de los cuerpos, junto con la Ley de Gravitación Universal, permitieron dar una mejor comprensión del movimiento y las fuerzas involucradas. Esto permitió explicarle mejor muchos fenómenos que antes eran mal entendidos, acerca de las órbitas de los planetas y la Luna.

Newton postuló que todos los cuerpos se atraen entre sí con una fuerza proporcional a sus masas e inversamente proporcional al cuadrado de la distancia que los separa, la formulación matemática de esta ley es la siguiente:

$$F \sim \frac{m_1 * m_2}{r^2}$$

Donde:

- m_1 y m_2 son las masas de los cuerpos.

- r La distancia que los separa.

En su forma vectorial:

$$\vec{F} = \frac{-Gm_1m_2}{|\vec{r}|^2} \frac{\vec{r}}{|\vec{r}|}$$

- F produce una aceleración sobre m
- Por segunda ley de Newton:

$$\vec{F} = m\vec{r}$$

Partiendo de la segunda Ley de Newton y la Ley de Gravitación Universal, podremos deducir el sistema de ecuaciones diferenciales que junto con un método numérico nos ayudarán a hacer las simulaciones ya vistas en el capítulo anterior. Estas simulaciones son una aplicación del método numérico para resolver el sistema de ecuaciones para ciertas condiciones iniciales.

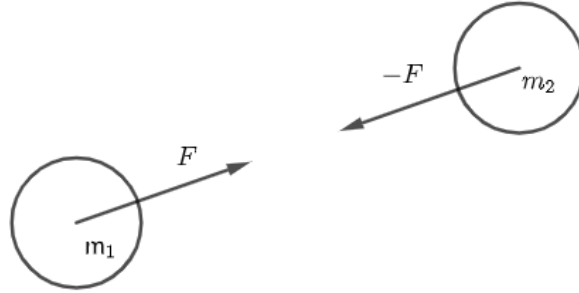


Figura 3.1: Dos cuerpos sometidos a la fuerza de gravitación, las dos masas experimentan la misma fuerza pero en diferente sentido.

3.2. Deducción analítica de las ecuaciones de movimiento gravitacional

Partiendo de la segunda ley de Newton y la Ley de Gravitación Universal.

$$\vec{F} = \frac{-Gm_1m_2}{|\vec{r}|^2}, \vec{F} = m_1\vec{r}$$

Recordemos que $\vec{F}$ es la fuerza con la que se atraen dos masas puntuales m_1 y m_2 . Por segunda ley de Newton esta fuerza genera una aceleración sobre los cuerpos, podremos suponer la segunda ley para un cuerpo, entonces obtendremos como es afectado este cuerpo(m_1).

Iguálamos

$$m_1 \ddot{\vec{r}} = \frac{-Gm_1 m_2}{|\vec{r}|^2} \frac{\vec{r}}{|\vec{r}|}$$

Para describir la posición de un objeto en el espacio, suponiendo que su masa está concentrada en un punto. Lo haremos mediante el vector;

$$\vec{r} = r \hat{e}_r \quad (*)$$

También recordemos que en coordenadas polares

$$\hat{e}_r = (\cos(\theta), \sin(\theta))$$

$$\dot{\hat{e}}_r = (-\sin(\theta), \cos(\theta)) \dot{\theta}$$

$$\dot{\hat{e}}_\theta = \dot{\theta} \hat{e}_\theta \text{ donde } \hat{e}_\theta = (-\sin(\theta), \cos(\theta))$$

Ahora sabemos que la derivada, o el cambio de la posición como función del tiempo, puede representarse matemáticamente como la derivada, en este caso la derivada de r .

$$\frac{dr}{dt} = \dot{r}$$

De (*)

$$\dot{\vec{r}} = \dot{r} \hat{e}_r + r \dot{\theta} \hat{e}_\theta$$

$$\ddot{\vec{r}} = \ddot{r} \hat{e}_r + \dot{r} (\dot{\theta} \hat{e}_\theta) + (\dot{r} \dot{\theta} + r \ddot{\theta}) \hat{e}_\theta + r \dot{\theta} (-\dot{\theta} \hat{e}_r)$$

Usando La segunda Ley de Newton y Gravitación Universal.

$$m_1 \vec{a} = (\ddot{r} - r\dot{\theta}^2)\hat{e}_r + (2\dot{r}\dot{\theta} + r\ddot{\theta})\hat{e}_\theta = \frac{-Gm_1m_2}{r^2}\hat{e}_r + 0\hat{e}_\theta$$

$$m\vec{a} = \frac{-Gm_1m_2}{r^2}$$

Por igualdad vectorial

$$\vec{e}_r : m(\ddot{r} - r\dot{\theta}^2) = \frac{-Gm_1m_2}{r^2} \quad (\text{Ec 2.1})$$

$$\vec{e}_\theta : \frac{r}{r}(2\dot{r}\dot{\theta} + r\ddot{\theta}) = \frac{(2r\dot{r}\dot{\theta} + r^2\ddot{\theta})}{r} = 0 \quad (\text{Ec 2.2})$$

De (2,1). Si $\mu = Gm_1$ entonces;

$$\ddot{r} = r\dot{\theta}^2 - \mu r^{-2}$$

Multiplicando por r^2 y despejando μ

$$\mu = r^3\dot{\theta}^2 - \ddot{r}r^2 \quad (3.1)$$

Reescribiendo la ecuación (2.2)

$$2r\dot{r}\dot{\theta} + r^2\ddot{\theta} = \frac{d}{dt}(r^2\dot{\theta}) = 0$$

Entonces

$r^2\dot{\theta} = h$ tal que h es una constante

Sustituyendo en (3,1)

$$\dot{\theta} = \frac{h}{r^2}, \quad \dot{\theta}^2 = \frac{h^2}{r^4}$$

$$\mu = \frac{h^2}{r} - r^2 \ddot{r}$$

Aplicando el cambio de variable $u(t) = \frac{1}{r(t)}$

$$\rightarrow r = \frac{1}{u} \rightarrow \dot{r} = \frac{-\dot{u}}{u^2} \rightarrow \ddot{r} = \frac{2u\dot{u}^2 - \ddot{u}u^2}{u^4}$$

Sustituyendo en (3.1)

$$\mu = r^2 \dot{\theta}^2 - \ddot{r} r^2$$

$$\mu = uh^2 - u^{-2}(2u^{-3}\dot{u}^2 - \ddot{u}u^{-2})$$

$$\mu = uh^2 - u^{-5}\dot{u}^2 + \ddot{u}u^{-4}$$

Multiplicando por u^2

$$u^2\mu = u^3h^2 - 2u^{-3}\dot{u}^2 + \ddot{u}u^{-2} \quad (3.2)$$

Ahora podemos escribir esta ecuación en función de $U(\theta)$

siendo $u(t) = U(\theta(t))$ y Aplicando regla de la cadena;

$$\dot{U} = \frac{du}{d\theta} \frac{d\theta}{dt}. \text{ Recuerde que } \dot{\theta} = \frac{h}{r^2} = U^2h$$

$$\dot{U} = U_{\theta} h U^2$$

$$\ddot{U} = \dot{\theta} \frac{d}{d\theta} (U_{\theta} h U^2)$$

Simplificando

$$\ddot{U} = h^2 U^3 (U_{\theta\theta} U + 2U_{\theta}^2)$$

Sustituyendo en (3.2) y simplificando

$$\boxed{\frac{\mu}{h^2} = U + U_{\theta\theta}}$$

La ecuación diferencial lineal de segundo orden tiene la solución;

$$U(\theta) = \lambda_1 \cos(\theta - \lambda_2) + \frac{\mu}{h^2}$$

Donde λ_1 y λ_2 determinados por las condiciones iniciales del problema.

Esta ecuación esta conformada por: la solución a la homogénea más la solución particular.

$$U(\theta) = U_{homogenea} + U_{particular}$$

Regresando a términos de $r(\theta)$;

$$r(\theta) = \frac{1}{\frac{\mu}{h^2} + \lambda_1 \cos(\theta - \lambda_2)}$$

Esta ecuación describe un movimiento Elíptico en función de θ .

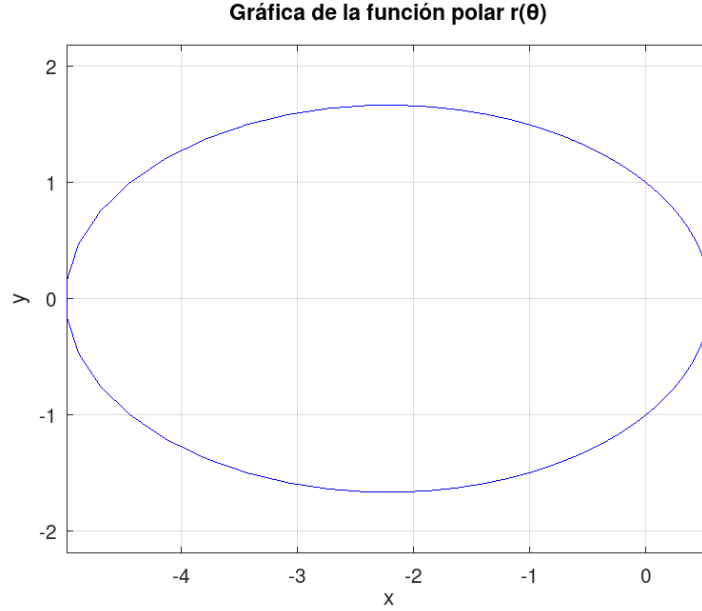


Figura 3.2: Gráfica de la función $U(\theta) = \frac{\mu}{h^2} + \lambda_1 \cos(\theta - \lambda_2)$ con los parámetros: $\mu = 1$, $h = 1$, $\lambda_1 = 0,8$, y $\lambda_2 = 0,8$.

3.3. Solución Numérica

Anteriormente se ha estudiado la solución analítica del problema. En este capítulo se estudiará la solución numérica. Esta solución requiere el desarrollo de un sistema de ecuaciones, condiciones iniciales y la aplicación de un método numérico para resolver el sistema.

El punto de partida será la ecuación de Gravitación Universal junto con la segunda ley de Newton.

$$\vec{F} = m_1 \ddot{\vec{r}}_1 = -G \frac{m_1 m_2}{|\vec{r}_1 - \vec{r}_2|^3} (\hat{r}_1 - \hat{r}_2)$$

Esta ecuación describe como es la aceleración de m_1 sometida a la fuerza gravitacional de m_2

$$m_1 \ddot{\vec{r}}_1 = -G \frac{m_1 m_2}{|\vec{r}_1 - \vec{r}_2|^3} (\hat{r}_1 - \hat{r}_2) \quad (3.3)$$

Este sistema describe como es la aceleración de los cuerpos 1 $\ddot{\vec{r}}_1$ y como es la aceleración del cuerpo 2 $\ddot{\vec{r}}_2$

$$\ddot{\vec{r}}_1 = -G \frac{m_2 (\vec{r}_1 - \vec{r}_2)}{|\vec{r}_1 - \vec{r}_2|^3} (\hat{r}_1 - \hat{r}_2) \quad \ddot{\vec{r}}_2 = -G \frac{m_1 (\vec{r}_2 - \vec{r}_1)}{|\vec{r}_1 - \vec{r}_2|^3} (\hat{r}_1 - \hat{r}_2) \quad (3.4)$$

Junto con las condiciones iniciales de velocidad

$$\dot{\vec{r}}_1 = \vec{v}_1 \quad \dot{\vec{r}}_2 = \vec{v}_2 \quad (3.5)$$

y posición

$$\vec{r}_1 = (x_1, y_1) \quad \vec{r}_2 = (x_2, y_2) \quad (3.6)$$

Considerando solo el cuerpo 1. Reescribiendo el sistema en su forma cartesiana.

$$\begin{cases} \dot{x} = v_x \\ \dot{y} = v_y \\ \dot{v}_x = -\frac{Gm_2(x - x_2)}{(x^2 + y^2)^{3/2}} \\ \dot{v}_y = -\frac{Gm_2(y - y_2)}{(x^2 + y^2)^{3/2}} \end{cases}$$

Donde cada ecuación es una función dependiente del tiempo, posición, velocidad y aceleración.

$$\begin{aligned}
\ddot{x} &= f_1(t, x, y, v_x, v_y) \\
\ddot{y} &= f_2(t, x, y, v_x, v_y) \\
\dot{x} &= f_3(t, x, y, v_x, v_y) \\
\dot{y} &= f_4(t, x, y, v_x, v_y) \\
x &= f_5(t) \\
y &= f_6(t)
\end{aligned}$$

Para resolver este sistema, aplicaremos un cambio de variable (C.V.).

$$\begin{aligned}
\dot{x} &= v_x \\
\dot{y} &= v_y
\end{aligned}$$

Entonces, el sistema se transforma en:

$$\begin{aligned}
\dot{v}_x &= f_1(x, y, v_x, v_y, t) \\
\dot{v}_y &= f_2(x, y, v_x, v_y, t) \\
v_x &= f_3(x, y) \\
v_y &= f_4(x, y)
\end{aligned}$$

Con las condiciones iniciales dadas para v_x , v_y , x y y .

$$\begin{aligned}
v_{x_0} \\
v_{y_0} \\
x_0 \\
y_0
\end{aligned}$$

Para condiciones iniciales dadas de Posición (x_0, y_0) y Velocidad Inicial (v_{x_0}, v_{y_0}) aplicando el método tendremos diferentes resultados sobre la posición del cuerpo 1 en función del tiempo.

Capítulo 4

El problema de los 3 cuerpos restringido

4.1. Introducción

El primer problema que motivó este trabajo, es conocer el movimiento de una nave espacial, que en un momento inicial $t = 0$, se encuentra a una cierta distancia de la Tierra r_0 y cuyos motores imprimen una fuerza que dota a la nave de una v_0 . La cuestión es obtener la velocidad inicial de la nave, para que esta se mantenga en una órbita alrededor de la Tierra. Utilizando la ley de gravitación universal de Newton, la segunda ley, es posible llegar a un sistema de ecuaciones que junto con el método numérico Runge kutta de cuarto orden se pudo resolver, probando varias condiciones iniciales de v_0 , a una distancia dada r_0 . Para este primer problema, fue relativamente fácil obtener varias condiciones que nos daban órbitas iniciales; posteriormente se complicó el problema agregando la Luna. Con este problema se puede entender cómo es un sistema de dos cuerpos (El sistema Nave-Tierra) y ayuda a entender cómo es un sistema de tres cuerpos (Nave-Tierra-Luna). Este sistema de tres cuerpos, es un sistema restringido, dado que la masa de la nave es prácticamente despreciable.

El problema de dos cuerpos tiene solución, pero el problema con tres cuerpos no tiene como tal una solución general; se dice que Newton pasó muchos años buscando una solución general a este problema sin tener éxito. Es un problema aún estudiado hoy en día.

El problema se soluciona cuando se hacen ciertas restricciones. En este capítulo se estudiarán algunos casos usando métodos numéricos.

4.2. Deducción de las Ecuaciones de Movimiento para 3 cuerpos.

Partiendo de la segunda ley de Newton, $F = ma$ y de la fuerza de gravitación universal,

$$\vec{F} = \frac{-Gm_1m_2}{|\vec{r}|^2} \frac{\vec{r}}{|\vec{r}|}$$

Para dos masas puntuales, m_1 , m_2 , que distan entre si $\vec{r}$. Se determinará como son las ecuaciones de movimiento para un sistema de tres cuerpos.

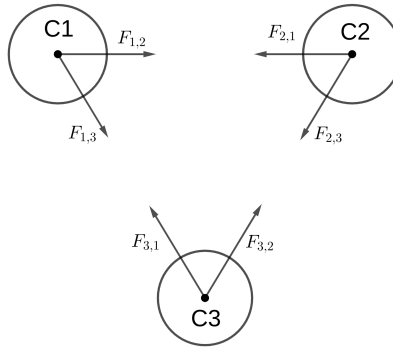


Figura 4.1: Tres cuerpos y sus fuerzas de gravitación

La fuerza de gravitación universal de Newton. Se calcula para dos cuerpos de masas puntuales. De modo que para encontrar la fuerza total sobre uno de estos cuerpos será necesario hacer una suma vectorial, la cual involucra cada fuerza.

Entonces la fuerza neta sobre el cuerpo $C1$, Será expresada como $F_{c1} = F_{1,2} + F_{1,3}$. En lenguaje natural se puede expresar: "La fuerza total sobre el cuerpo 1, $C1$, es la suma vectorial de la fuerza que ejerce el cuerpo 2, sobre

el cuerpo 1, $F_{1,2}$ más la fuerza que ejerce el cuerpo 3 sobre el cuerpo 1, $F_{1,3}$ ”.

Sustituyendo las fuerzas, tendremos la fuerza neta sobre $C1$ como:

$$F_{c1} = m_1 \ddot{\vec{r}} = -\frac{Gm_1m_2}{|\vec{r}_1 - \vec{r}_2|^3}(\vec{r}_1 - \vec{r}_2) - \frac{Gm_1m_3}{|\vec{r}_1 - \vec{r}_3|^3}(\vec{r}_1 - \vec{r}_3)$$

Eliminando m_1 y en notación cartesiana, se puede expresar:

$$\begin{aligned}\ddot{x}_{c1} &= -\frac{Gm_2(x_1 - x_2)}{(d_{1,2})^3} - \frac{Gm_3(x_1 - x_3)}{(d_{1,3})^3} \\ \ddot{y}_{c1} &= -\frac{Gm_2(y_1 - y_2)}{(d_{1,2})^3} - \frac{Gm_3(y_1 - y_3)}{(d_{1,3})^3}\end{aligned}$$

Con

$$\begin{aligned}d_{1,2} &= \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \\ d_{1,3} &= \sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2}\end{aligned}$$

De igual forma para $C2$

$$\begin{aligned}\ddot{x}_{c2} &= -\frac{Gm_1(x_2 - x_1)}{(d_{1,2})^3} - \frac{Gm_3(x_3 - x_1)}{(d_{2,3})^3} \\ \ddot{y}_{c2} &= -\frac{Gm_1(y_2 - y_1)}{(d_{1,2})^3} - \frac{Gm_3(y_3 - y_1)}{(d_{2,3})^3}\end{aligned}$$

De igual forma para $C3$

$$\begin{aligned}\ddot{x}_{c3} &= -\frac{Gm_1(x_3 - x_1)}{(d_{1,3})^3} - \frac{Gm_2(x_3 - x_2)}{(d_{3,2})^3} \\ \ddot{y}_{c3} &= -\frac{Gm_1(y_3 - y_1)}{(d_{1,3})^3} - \frac{Gm_2(y_3 - y_2)}{(d_{3,2})^3}\end{aligned}$$

De forma mas explícita:

$$\ddot{x}_{c1} = -\frac{Gm_2(x_1 - x_2)}{(\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2})^3} - \frac{Gm_3(x_1 - x_3)}{(\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2})^3}$$

$$\ddot{y}_{c1} = -\frac{Gm_2(y_1 - y_2)}{(\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2})^3} - \frac{Gm_3(y_1 - y_3)}{(\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2})^3}$$

$$\ddot{x}_{c2} = -\frac{Gm_1(x_2 - x_1)}{(\sqrt{(x_2 - x_3)^2 + (y_1 - y_2)^2})^3} - \frac{Gm_3(x_2 - x_3)}{(\sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2})^3}$$

$$\ddot{y}_{c2} = -\frac{Gm_1(y_2 - y_1)}{(\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2})^3} - \frac{Gm_3(y_2 - y_3)}{(\sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2})^3}$$

$$\ddot{x}_{c3} = -\frac{Gm_1(x_3 - x_1)}{(\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2})^3} - \frac{Gm_2(x_3 - x_2)}{(\sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2})^3}$$

$$\ddot{y}_{c3} = -\frac{Gm_1(y_3 - y_1)}{(\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2})^3} - \frac{Gm_2(y_3 - y_2)}{(\sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2})^3}$$

4.3. Aplicando el método de Euler

Suponga que existen tres cuerpos con masas puntuales, en cualquier lugar del plano euclidiano y con una velocidad inicial. Cada uno es atraído por la fuerza gravitacional de los otros, igualmente se considera la segunda ley de Newton. Utilizando las ecuaciones que se han planteado previamente se pueden probar múltiples condiciones iniciales de posición y velocidad con el fin de visualizar las trayectorias de los cuerpos. Considerando todas las posibles condiciones iniciales deben existir trayectorias tales que los cuerpos siempre están sometidos a estas fuerzas, esto es lo que en un principio se está buscando, se puede decir que estas son algunas soluciones al problema de los tres cuerpos.

El sistema de ecuaciones que describe estas trayectorias es:

$$\begin{aligned}
\dot{v}_{x_1} = \ddot{x}_1 &= -\frac{Gm_2(x_1 - x_2)}{\left(\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}\right)^3} - \frac{Gm_3(x_1 - x_3)}{\left(\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2 + (z_1 - z_3)^2}\right)^3} \\
\dot{v}_{y_1} = \ddot{y}_1 &= -\frac{Gm_2(y_1 - y_2)}{\left(\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}\right)^3} - \frac{Gm_3(y_1 - y_3)}{\left(\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2 + (z_1 - z_3)^2}\right)^3} \\
\dot{v}_{z_1} = \ddot{z}_1 &= -\frac{Gm_2(z_1 - z_2)}{\left(\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}\right)^3} - \frac{Gm_3(z_1 - z_3)}{\left(\sqrt{(x_1 - x_3)^2 + (y_1 - y_3)^2 + (z_1 - z_3)^2}\right)^3} \\
\dot{v}_{x_2} = \ddot{x}_2 &= -\frac{Gm_1(x_2 - x_1)}{\left(\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}\right)^3} - \frac{Gm_3(x_2 - x_3)}{\left(\sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2 + (z_2 - z_3)^2}\right)^3} \\
\dot{v}_{y_2} = \ddot{y}_2 &= -\frac{Gm_1(y_2 - y_1)}{\left(\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}\right)^3} - \frac{Gm_3(y_2 - y_3)}{\left(\sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2 + (z_2 - z_3)^2}\right)^3} \\
\dot{v}_{z_2} = \ddot{z}_2 &= -\frac{Gm_1(z_2 - z_1)}{\left(\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}\right)^3} - \frac{Gm_3(z_2 - z_3)}{\left(\sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2 + (z_2 - z_3)^2}\right)^3} \\
\dot{v}_{x_3} = \ddot{x}_3 &= -\frac{Gm_1(x_3 - x_1)}{\left(\sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2 + (z_3 - z_1)^2}\right)^3} - \frac{Gm_2(x_3 - x_2)}{\left(\sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2 + (z_3 - z_2)^2}\right)^3} \\
\dot{v}_{y_3} = \ddot{y}_3 &= -\frac{Gm_1(y_3 - y_1)}{\left(\sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2 + (z_3 - z_1)^2}\right)^3} - \frac{Gm_2(y_3 - y_2)}{\left(\sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2 + (z_3 - z_2)^2}\right)^3} \\
\dot{v}_{z_3} = \ddot{z}_3 &= -\frac{Gm_1(z_3 - z_1)}{\left(\sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2 + (z_3 - z_1)^2}\right)^3} - \frac{Gm_2(z_3 - z_2)}{\left(\sqrt{(x_3 - x_2)^2 + (y_3 - y_2)^2 + (z_3 - z_2)^2}\right)^3}
\end{aligned}$$

Condiciones iniciales, posiciones y velocidades Usando notación vectorial donde P_i son las coordenadas (x, y, z) para el cuerpo i

$$\mathbf{P}_1 = (-0,5, 0, 0), \quad \mathbf{P}_2 = (0,5, 0, 0), \quad \mathbf{P}_3 = (0, 0,866025404, 0)$$

Se probaron las siguientes velocidades iniciales:

$$\mathbf{V}_1 = (0, -1, 0), \quad \mathbf{V}_2 = (0, 1, 0), \quad \mathbf{V}_3 = (-1, 0, 0)$$

Para esta simulación se usaron los siguientes datos

Tres masas iguales; $m_1 = m_2 = m_3 = 1$

Constante de Gravitación Universal; $G = 1$

Variables del método;

$$\Delta t = 0.00001, \text{ Pasos} = 1000000$$

Algoritmo

1. Dadas las posiciones iniciales calcular las aceleraciones.

$$a_1 = -G * m_2 * \frac{(P_1 - P_2)}{r_{12}^3} - G * m_3 * \frac{(P_1 - P_3)}{r_{13}^3}$$

$$a_2 = -G * m_3 * \frac{(P_2 - P_3)}{r_{23}^3} - G * m_1 * \frac{(P_2 - P_1)}{r_{12}^3}$$

$$a_3 = -G * m_1 * \frac{(P_3 - P_1)}{r_{13}^3} - G * m_2 * \frac{(P_3 - P_2)}{r_{23}^3}$$

Donde las Distancias entre cuerpos están definidas como:

$$r_{12} = \sqrt{\sum (p_1 - p_2)^2}$$

$$r_{13} = \sqrt{\sum (p_1 - p_3)^2}$$

$$r_{23} = \sqrt{\sum (p_2 - p_3)^2}$$

2. Actualizar velocidades y posiciones. Desde $i = 1$, hasta $Pasos - 1$

$$P_1 = p_1 + v_1 \cdot \Delta t$$

$$V_1 = v_1 + a_1 \cdot \Delta t$$

$$P_2 = p_2 + v_2 \cdot \Delta t$$

$$V_2 = v_2 + a_2 \cdot \Delta t$$

$$P_3 = p_3 + v_3 \cdot \Delta t$$

$$V_3 = v_3 + a_3 \cdot \Delta t$$

Obteniendo para cada i una posición diferente

El algoritmo es implementado en (Apéndice A, programa 2)

Resultados

Utilizando las mismas condiciones solo variando velocidades iniciales:

$$\mathbf{V}_1 = \left(0, \frac{-1}{10}, 0\right), \quad \mathbf{V}_2 = \left(0, \frac{1}{10}, 0\right), \quad \mathbf{V}_3 = \left(\frac{-1}{10}, 0, 0\right)$$

$$\mathbf{V}_1 = \left(0, \frac{-2}{10}, 0\right), \quad \mathbf{V}_2 = \left(0, \frac{2}{10}, 0\right), \quad \mathbf{V}_3 = \left(\frac{-2}{10}, 0, 0\right)$$

$$\mathbf{V}_1 = \left(0, \frac{-3}{10}, 0\right), \quad \mathbf{V}_2 = \left(0, \frac{3}{10}, 0\right), \quad \mathbf{V}_3 = \left(\frac{-3}{10}, 0, 0\right)$$

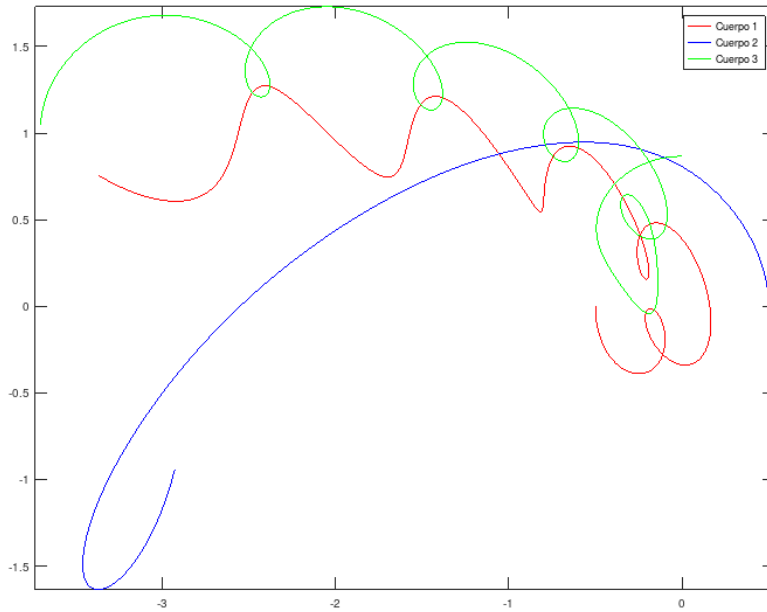


Figura 4.2: Gráfica xy para las trayectorias obtenidas.

$$\mathbf{v}_1 = \left(0, \frac{-4}{10}, 0\right), \quad \mathbf{v}_2 = \left(0, \frac{4}{10}, 0\right), \quad \mathbf{v}_3 = \left(\frac{-4}{10}, 0, 0\right)$$

Además se aumentó la precisión.

4.4. Aplicando el método Runge kutta de 4to orden

El método numérico Runge kutta de cuarto orden ya lo hemos aplicado para resolver el problema inicial. Nuevamente será aplicado, ahora para resolver una variación del problema de los tres cuerpos.

Igual que en la subsección anterior donde aplicamos el método de Euler, aplicaremos ahora el método Runge kutta de cuarto orden a condiciones iniciales similares.

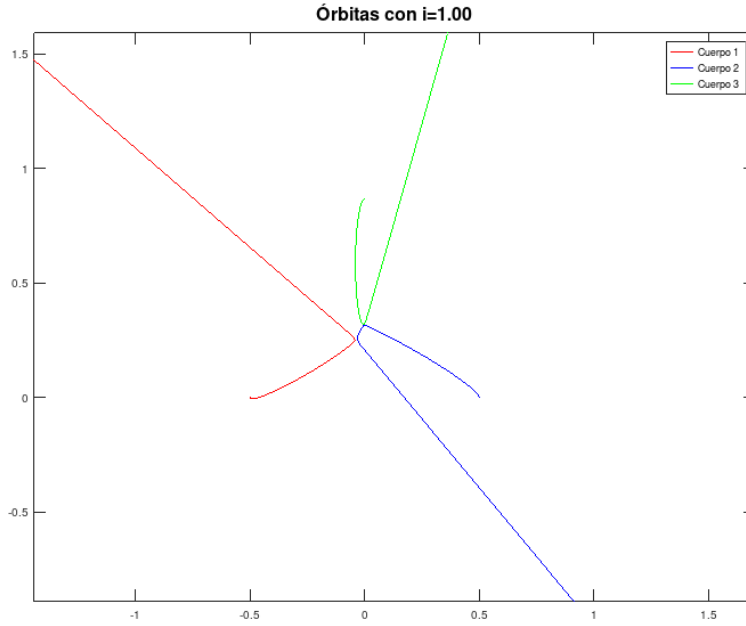


Figura 4.3: Gráfica x,y para las trayectorias obtenidas.

Condiciones iniciales, posiciones y velocidades Usando notación vectorial donde P_i son las coordenadas (x, y, z) para el cuerpo i

$$\mathbf{P}_1 = (-0.5, 0, 0), \quad \mathbf{P}_2 = (0.5, 0, 0), \quad \mathbf{P}_3 = (0, 0.866025404, 0)$$

Se probaron las siguientes velocidades iniciales:

$$\mathbf{V}_1 = (0, -0.5, 0), \quad \mathbf{V}_2 = (0, 0.5, 0), \quad \mathbf{V}_3 = (-0.5, 0, 0)$$

Para esta simulación se usaron los siguientes datos

Tres masas iguales; $m_1 = m_2 = m_3 = 1$

Constante de Gravitación Universal; $G = 1$

Variables del método:

$$\Delta t = 0,0001, \text{ Pasos} = 100000$$

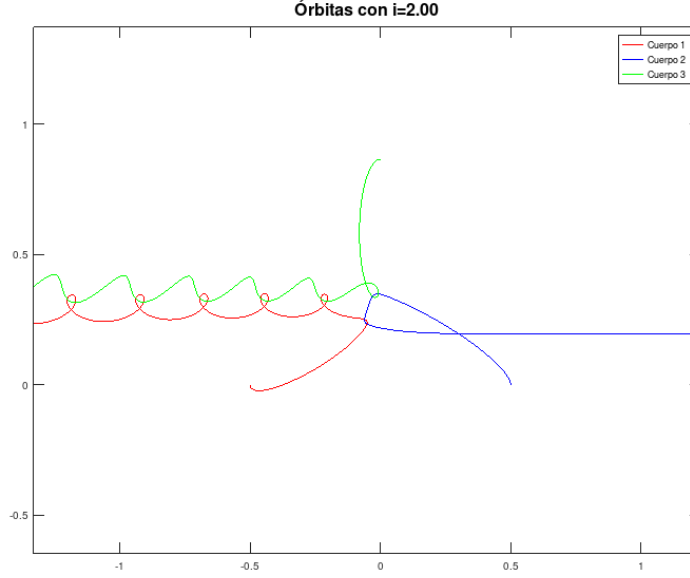


Figura 4.4: Gráfica x,y para las trayectorias obtenidas.

4.5. Simulaciones numéricas y resultados

Variado las C.I:

$$\mathbf{V}_1 = (0, -1, 0), \quad \mathbf{V}_2 = (0, 1, 0), \quad \mathbf{V}_3 = (-1, 0, 0)$$

La implementación de este programa fue importante para poder generalizarlo a N cuerpos. El programa principal tiene la misma estructura que en la implementación del método de Euler, solo que ahora se aplica el método Runge kutta de cuarto orden, el programa tiene muchas líneas de código. Considere que se está trabajando en el espacio $\mathbb{R}^3$, donde hay arreglos a de tres dimensiones para las posiciones, (x, y, z) y para las velocidades (v_x, v_y, v_z) .

Para el caso de dos cuerpos en $\mathbb{R}^2$ el sistema de ecuaciones a resolver es:

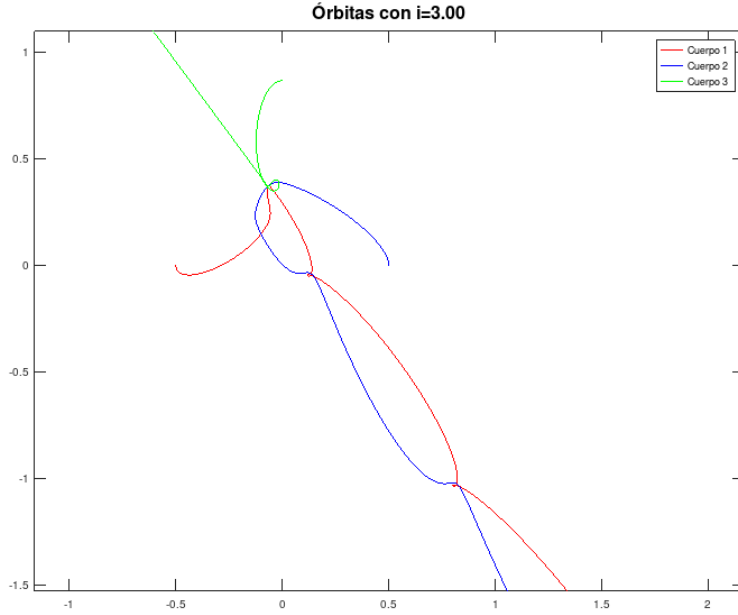


Figura 4.5: Gráfica x,y para las trayectorias obtenidas.

$$\begin{aligned}
 f_1(t, x, y) &= \dot{v}_x = \frac{-Gm_1(x_2 - x_1)}{\sqrt{(x_2 - x_1)^3}} \\
 f_2(t, x, y) &= v_x \\
 f_3(t, x, y) &= \dot{v}_y = \frac{-Gm_1(y_2 - y_1)}{\sqrt{(y_2 - y_1)^3}} \\
 f_4(t, x, y) &= v_y
 \end{aligned}$$

Ahora tendremos por ende un sistema de ecuaciones 6×6 , donde tendremos 3 ecuaciones de posición y 3 ecuaciones de velocidad. Considerando solo el movimiento de un cuerpo, como fue el caso del Problema Inicial visto en el capítulo 2. Dado que estamos resolviendo para los 3 cuerpos simultáneamente, tendremos un sistema de 18 ecuaciones de movimiento. Porque tenemos un sistema 6×6 , por cada cuerpo, para 3 cuerpos.

El uso de vectores de tres dimensiones en la aplicación a fortran que se

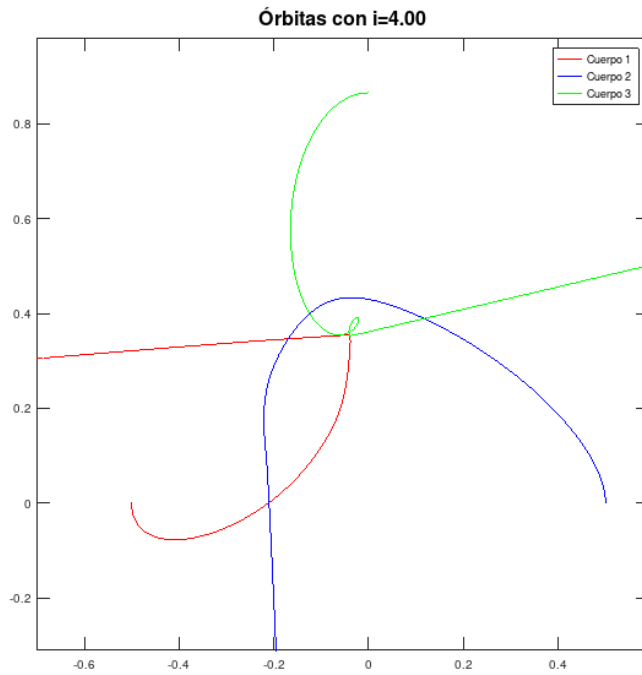


Figura 4.6: Gráfica x,y para las trayectorias obtenidas.

ha estado utilizando resulta ya obsoleto ya que tendremos que definir las 18 ecuaciones dentro del método, volviéndose demasiado robusto y confuso. Con el fin de entender bien la estructura y cómo organizarlo en la aplicación se decidió hacer el código, lo cual ayudó a entender el orden de las variables. (Parte de la implementación Apéndice, Programa 3).

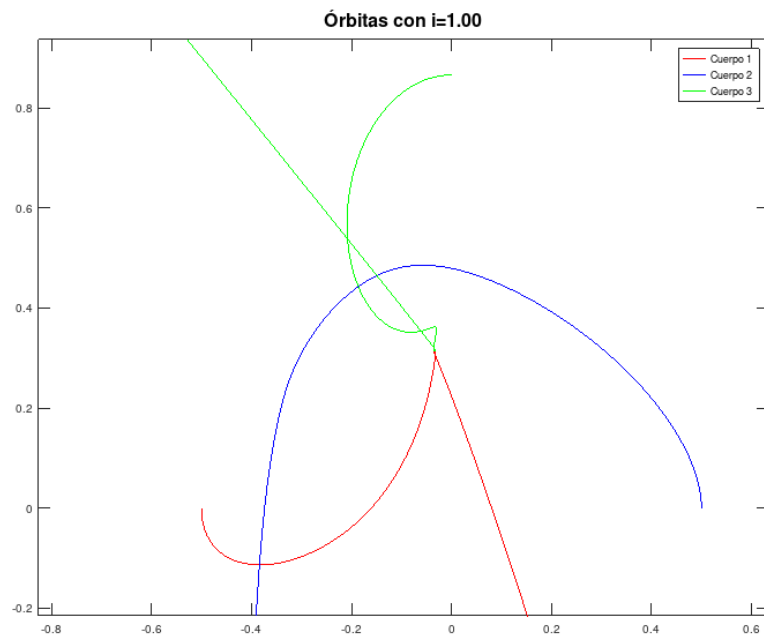


Figura 4.7: Gráfica x,y para las trayectorias obtenidas.

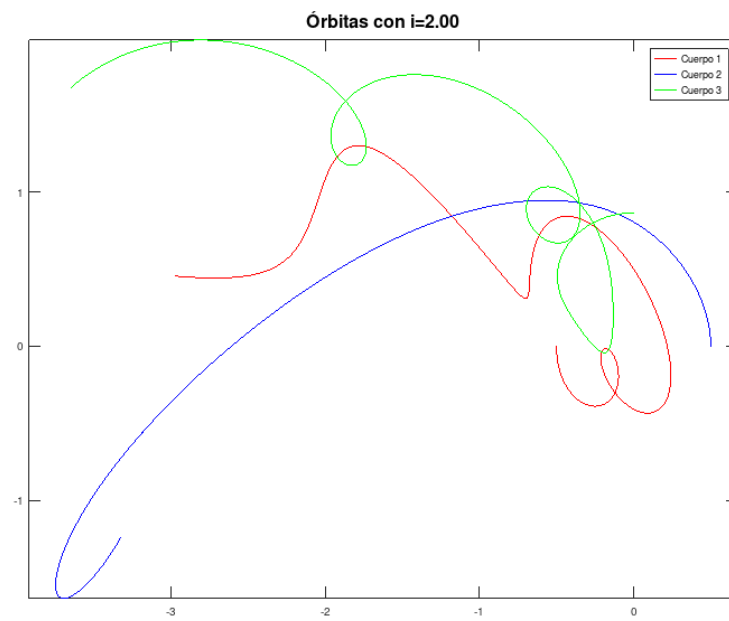


Figura 4.8: Gráfica x,y para las trayectorias obtenidas

Capítulo 5

Problema de N cuerpos

5.1. Introducción

Ya se han realizado simulaciones con 3 cuerpos, ahora para mejorar el algoritmo y simplificarlo se han utilizado arreglos bidimensionales o matrices. Esto ha permitido simplificar mucho el código (Programa 3). Las condiciones iniciales son guardadas en matrices, al aplicar el método numérico este opera con las mismas matrices actualizándose en cada paso y guardando los datos. Para ello se tuvo que rehacer el método pero utilizando arreglos unidimensionales para los parámetros del método, ahora N , que es el número de cuerpos es un parámetro que puede ser modificado.

5.2. Simulaciones y Resultados para 4 cuerpos

Mejorando el algoritmo implementado en el Programa 3, ahora con el uso de matrices para guardar los datos de posición, velocidad y aceleración se ha mejorado el algoritmo, siendo más eficiente para un número mayor a 3 cuerpos. Este algoritmo mejorado se ha implementado en el programa 4 en el apéndice A. Este programa utiliza el parámetro N como número de cuerpos, por lo cual puede ser modificado para hacer simulaciones con cualquier número de cuerpos, observe que también debe ser modificada la subrutina dentro del módulo.

Ahora se planteó un nuevo caso para cuatro cuerpos. Haciendo múltiples pruebas se llegaron a las siguientes Condiciones iniciales para las cuales se

encuentran órbitas.

Condiciones Iniciales de Posición:

$$\mathbf{P}_1 = (0,5, 0, 0), \quad \mathbf{P}_2 = (0, 0,8660, 0), \quad \mathbf{P}_3 = (-0,5, 0, 0), \quad \mathbf{P}_4 = (0, -0,8660, 0)$$

Condiciones Iniciales de velocidad:

$$\mathbf{V}_1 = (0, 0,01, 0), \quad \mathbf{V}_2 = (-0,01, 0, 0), \quad \mathbf{V}_3 = (0, -0,01, 0), \quad \mathbf{V}_4 = (0,01, 0, 0)$$

Igual que en las simulaciones de 3 cuerpos. Se consideraron cuatro cuerpos con masas iguales; $m_1 = m_2 = m_3 = m_4 = 1$

Constante de Gravitación Universal; $G = 1$

Variables del método;

$$\Delta t = 0,0001, \text{ Pasos} = 1000000$$

Resultado

Para guardar los datos. El programa crea varios archivos con el nombre "BODYi.dat" donde i corresponde al cuerpo. De esta manera los archivos; "BODY1.dat", "BODY2.dat", "BODY3.dat", "BODY4.dat" almacenan los datos de las coordenadas x , y y z en cada paso. Siendo la primer columna para las x , segunda columna para y y la tercera columna para z .

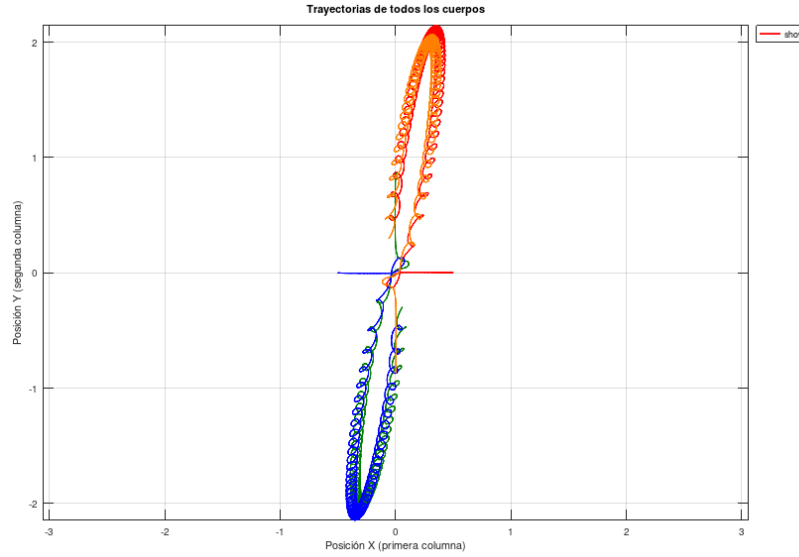


Figura 5.1: Gráfica plano x,y para las trayectorias de 4 cuerpos

5.3. Simulación para N cuerpos

Para crear una simulación de N cuerpos usando el código ya usado para las simulaciones de 4 cuerpos (Programa 4, Apéndice A). Modificando el parámetro N , además de definir las masas y las condiciones iniciales las cuales incluyen posiciones y velocidades.

Lo primero que motivó una simulación para N cuerpos, fue la idea de simular un Big Bang. Considere definir un conjunto de masas. Estas masas pueden ser definidas con variables aleatorias o simplemente $M(:) = 1,0$. Las posiciones de cada una, definidas en la matriz P , se distribuyen de manera uniforme dentro de una circunferencia, con un radio definido. Dice la teoría del Big Bang que en un inicio toda la masa del universo, estaba concentrada en un punto. Dentro de esta simulación tendremos una pequeña circunferencia con muchas masas puntuales ya que concentrar todas las posiciones en un mismo punto, generará un error numérico, el radio de esta circunferencia podrá ser tan pequeño como se desee. La forma en que simularemos la gran explosión será darles a estas masas puntuales una velocidad inicial.

Primero se probó con velocidades aleatorias, los resultados no fueron sa-

tisfactorios.

Después se probó con velocidades paralelas al radio de la circunferencia. Las distancias entre el centro y los puntos crecieron sin que se viera algún comportamiento de nuestro interés.

Por último se probó con velocidades tangenciales al radio de la circunferencia. Esto provocó un comportamiento en las partículas. Las gráficas obtenidas son como bolas de estambre, donde se puede apreciar un comportamiento caótico, con trayectorias semicirculares, en múltiples direcciones.

5.3.1. Simulaciones del Big Bang y Resultados

Para mejorar los tiempos de ejecución se mejoró el código (Programa 4, Apéndice A). Usando las herramientas de programación paralela de openmp se ha creado el código (Programa 5, Apéndice A). Este código en paralelo, nos proporcionará un mejor rendimiento dividiendo el cálculo para que más de un núcleo de una computadora pueda trabajar en el cálculo de la matriz P.

Parámetros de la Simulación

Número de cuerpos (N): 500

Constante gravitacional (G): 1.0

Paso temporal (Δt): 0.0001

Número de pasos (steps): 10000

Radio máximo de distribución: 10.0 unidades

Magnitud de velocidad: 10.0 unidades

Condiciones Iniciales

Masas: Distribución aleatoria uniforme en el rango $[0,0, 100,0)$

Posiciones: Distribución uniforme dentro de un disco de radio máximo
 $R = 10,0$ en el plano xy

Velocidades: Magnitud constante $v = 10,0$ con dirección tangencial
la circunferencia en el plano xy

Resultados

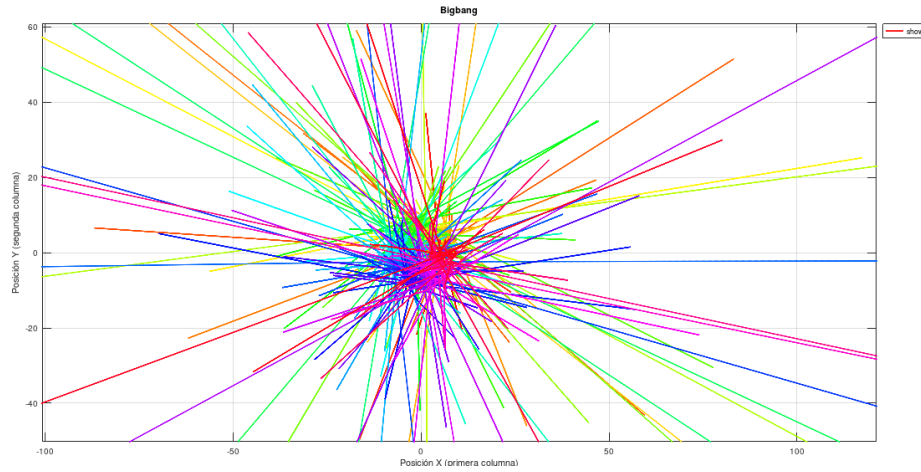


Figura 5.2: Simulación del Big Bang con 500 masas puntuales.

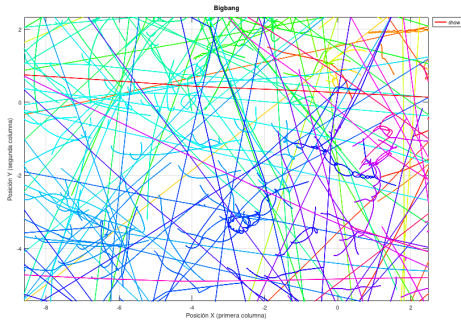


Figura 5.3: Cuando acercamos la imagen se pueden observar caos y algunas trayectorias semi-parabólicas

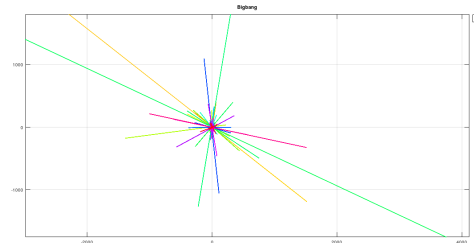


Figura 5.4: Al alejar la imagen se puede observar que algunos cuerpos se alejan de su posición inicial.

Capítulo 6

Buscando el Sistema solar usando el método de Montecarlo

6.1. Múltiples Simulaciones (Método de montecarlo)

En el código (programa 5, del Apéndice A) ya hemos paralelizado el código para poder mejorar los tiempos de ejecución usando openmp. Ahora probaremos múltiples condiciones iniciales (posición y velocidad). Igual se probará con masas diferentes. A este método se le conoce como método de Montecarlo, el cual funciona con variables aleatorias. De esta manera podremos analizar una gran variedad de posibles eventos. Para ello se hará uso del lenguaje Bash Script. Esto nos permitirá organizar las simulaciones, ejecutando el código múltiples veces, probando diferentes condiciones para un sistema de N cuerpos, bajo la acción de la fuerza gravitacional.

Lo que motivó este capítulo, es poder simular un sistema solar estable como en el que nos encontramos. Para lograr esto se ha modificado el código (programa 5). Se han hecho estas simulaciones con $N = 10$, es decir para 10 cuerpos.

En el centro del plano se ha colocado el cuerpo 1 con $M_1 = 10000$, esta masa representaría un sol central. Las masas de los demás cuerpos serán mucho más pequeñas. Generadas con variables aleatorias estarán dentro de este rango $(10, 99)$, para las posiciones, estarán distribuidas dentro del círculo

de una forma aleatoria y uniforme, las velocidades serán perpendiculares al radio entre el centro y el cuerpo. Para las primeras simulaciones se dio el mismo valor para todas las velocidades, $V = 10$, después de estuvo variando este parámetro dentro de un rango entre (10, 99) Se podría decir que al final se utilizó la misma idea para la simulación del bigbang, solo cambió la existencia de una masa central.

6.2. Lenguaje Script y organización de archivos

Dado que la idea es probar múltiples condiciones iniciales, resulta fácil pensar en rangos. Por ejemplo, para las condiciones iniciales ya descritas, las masas oscilan entre 10 y 99. Esto se debe a la facilidad que otorga fortran para utilizar variables aleatorias y el cómo trabajar estas variables para escalarlas.

Una breve explicación. Sea x una variable aleatoria, usando la función de fortran `call random_number(x)`. Donde x estará dentro del rango (0, 1). Para trabajar en un rango como (0, 10) simplemente multiplicamos por 10. Para asignar posiciones o velocidades convendría este rango (−10, 10).

Pensando en una forma más general sea el rango (a, b) , para escalar nuestra variable aleatoria dentro de este rango usaremos.

$$x_{reesc} = a + (b - a) * x$$

De esta forma se buscó re-escalar nuestras variables, teniendo en cuenta que los valores serán del mismo orden.

Cada que ejecutamos el código este se ejecutará con valores diferentes para las masas, posiciones y velocidades. Se puede decir que cada se ejecuta el programa se está probando una serie de condiciones diferentes. A excepción del cuerpo 1 el cual siempre tendrá la masa definida, la posición siempre será fija en (0,0,0) y la velocidad igual permanecerá en cero, este representa el sol, por ende su masa es mayor a los demás y suponemos que estará inerte en el resto del cálculo de la simulación.

El programa será ejecutado con el `script 1` en el apéndice B. Además de ejecutarlo un número determinado de veces. Para las primeras pruebas que se exponen se ejecuta 100 veces. Al ejecutar el script, este ordena los datos

de cada simulación, al ejecutar el programa se crea un directorio llamado **Simulación i** , donde i varía de 1 al número de simulaciones dado, posteriormente guarda los datos generados por el programa, el cual genera 10 bases de datos para cada planeta.

Para visualizar los resultados, se ha creado un segundo script (script 2, apéndice B), el cual copia dos programas de graficación; `|graficasnbody.m|` y `nbody2.m` en cada directorio **simulación i** , después hace una modificación en cada uno esto con el fin de guardar los resultados con el nombre de la simulación al que corresponde, al ejecutar el primer programa este genera una imagen que no la visualiza, la guarda en un directorio de resultados. La idea es solo visualizar las imágenes en un directorio, al encontrar un resultado de interés como órbitas estables este se podrá estudiar a fondo pues la imagen nos da el nombre de la carpeta. Por ejemplo podría ser **simulación33**, al entrar en este directorio el programa `graficasnbody2.m` nos generara la imagen la cual podremos ajustar a conveniencia. De igual forma el programa principal que ejecuta el script, se le ha echo una modificación, para que al momento de ejecutarlo guarde en un archivo independiente las condiciones iniciales del sistema, de esta forma podremos volver a reproducir dicho resultado.

6.3. Probando múltiples condiciones con Montecarlo

Una vez compilado el programa, podremos proceder a ejecutar el script. En la siguiente imagen se muestra la compilación del programa con openmp y lo que hay en el directorio de trabajo. El ejecutable es guardado con el nombre `sistema_s`, este programa será ejecutado 100 veces por el script `montecarlo`.

```
[Montecarlo] gfortran -fopenmp Sistema_solar.f90 -o sistema_s
Sistema_solar.f90:221:16:

  221 |     write(100,*), "Planeta", i
      |           1
Aviso: Extensión antigua: Comma before i/o item list at (1)
[Montecarlo] ls
graficasnbody2.m  graficasnbody.m  graficas_s.sh  limpiar.sh  montecarlo.sh  resultados  rk4_mod.mod  sistema_s  Sistema_solar.f90
```

Ejecución del script `montecarlo.sh`

```
[Montecarlo] gfortran -fopenmp Sistema_solar.f90 -o sistema_s
Sistema_solar.f90:221:16:

  221 |     write(100,*), "Planeta", i
      |           1
Aviso: Extensión antigua: Comma before i/o item list at (1)
[Montecarlo] ./montecarlo.sh
Ejecutando simulación 1
Ejecutando simulación 2
Ejecutando simulación 3
Ejecutando simulación 4
Ejecutando simulación 5
Ejecutando simulación 6
Ejecutando simulación 7
Ejecutando simulación 8
Ejecutando simulación 9
Ejecutando simulación 10
Ejecutando simulación 11
Ejecutando simulación 12
Ejecutando simulación 13
Ejecutando simulación 14
Ejecutando simulación 15
Ejecutando simulación 16
Ejecutando simulación 17
Ejecutando simulación 18
Ejecutando simulación 19
Ejecutando simulación 20
Ejecutando simulación 21
Ejecutando simulación 22
Ejecutando simulación 23
Ejecutando simulación 24
```

Ejecución del script `graficas_s.sh`

```
[Montecarlo] ./graficas_s.sh
Iniciando graficación para simulación1
■ Se encontraron 10 archivos de planetas
Leyendo archivos...
✓ Archivo Planeta1.dat cargado: 1000 puntos
✓ Archivo Planeta10.dat cargado: 1000 puntos
✓ Archivo Planeta2.dat cargado: 1000 puntos
✓ Archivo Planeta3.dat cargado: 1000 puntos
✓ Archivo Planeta4.dat cargado: 1000 puntos
✓ Archivo Planeta5.dat cargado: 1000 puntos
✓ Archivo Planeta6.dat cargado: 1000 puntos
✓ Archivo Planeta7.dat cargado: 1000 puntos
✓ Archivo Planeta8.dat cargado: 1000 puntos
✓ Archivo Planeta9.dat cargado: 1000 puntos
warning: legend: ignoring extra arguments after "show"
warning: called from
    legend>parse_opts at line 694 column 7
    legend at line 206 column 8
    graficasnbody at line 56 column 1

🖨 Guardando gráfica como simulacion1.png...
✓ Gráfica guardada exitosamente como: simulacion1.png
📄 Resumen: 10 planetas graficados con 1000 puntos cada uno
🔍 Escala fija: X [-15, 15], Y [-15, 15]
⏱ Proceso completado en segundo plano
Fin para grafica simulación1
Iniciando graficación para simulación2
■ Se encontraron 10 archivos de planetas
```

Al visualizar los resultados, se pretende buscar los de interés, estos son en los que visualizamos órbitas para todos los cuerpos. A continuación se muestra uno de los mejores resultados.

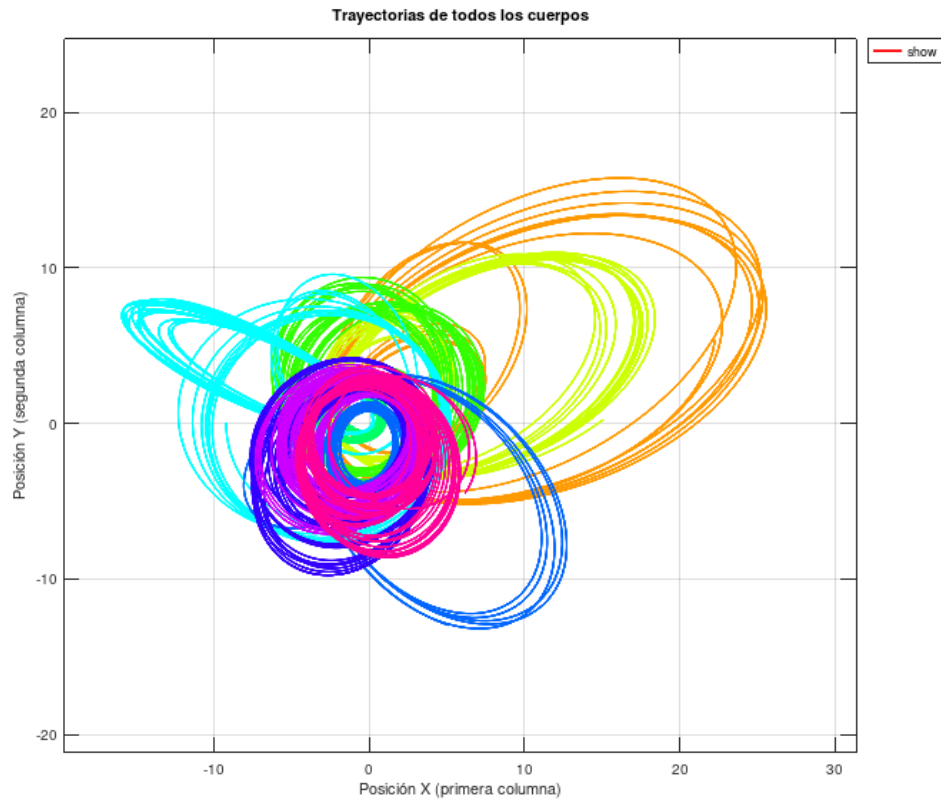


Figura 6.1: Simulación de un sistema solar Se exploraron miles de condiciones iniciales, variando tanto las posiciones como las velocidades de los cuerpos. Además, se asignaron masas aleatorias a todos los objetos, excepto a un cuerpo ubicado en el origen, el cual se mantuvo fijo y con una masa significativamente mayor que la de los demás.

Cuadro 6.1: **Condiciones iniciales del sistema planetario.**

Planeta	Posición			Velocidad			Masa
	x	y	z	v_x	v_y	v_z	
1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	100000.0000
2	6.7591	4.9108	0.0000	-33.4873	46.0913	0.0000	0.0192
3	2.1968	6.7610	0.0000	-75.3154	24.4715	0.0000	22.7229
4	-2.3060	7.0972	0.0000	-86.4440	-28.0874	0.0000	80.2730
5	-3.0761	2.2349	0.0000	-47.9065	-65.9376	0.0000	34.4655
6	-9.2039	0.0000	0.0000	0.0000	-86.9741	0.0000	12.2750
7	-7.6388	-5.5499	0.0000	30.9509	-42.6002	0.0000	62.7696
8	-2.0058	-6.1733	0.0000	79.1443	-25.7156	0.0000	7.1260
9	1.6547	-5.0926	0.0000	91.1835	29.6273	0.0000	44.1896
10	6.1686	-4.4818	0.0000	46.8776	64.5214	0.0000	97.4005

Capítulo 7

Conclusiones

- Para más de 2 cuerpos, las trayectorias tienden a comportarse con movimiento en espirales semejantes a elipses, circunferencias o con movimiento parabólico. Esto ocurre cuando hay una influencia clara de la fuerza gravitacional, aunque son resultados difíciles de obtener, por lo que es necesario realizar múltiples pruebas.
- En las simulaciones se buscó uno de tres posibles resultados:
 - Los cuerpos del sistema colisionan cuando la velocidad no es suficiente para escapar de la fuerza gravitacional.
 - Los cuerpos escapan cuando la velocidad es demasiada y logran superar la fuerza gravitacional.
 - En el caso de interés, la velocidad es suficiente para que el cuerpo no quede atrapado (caso 1) pero tampoco logre escapar (caso 2), resultando en una órbita circular o elíptica.
- Los casos de colisión y escape pueden ser confusos. Al no modelarse el fenómeno de colisión, cuando un cuerpo es atraído hacia otro con mayor masa, su trayectoria aparece gráficamente como una línea recta. Lo que ocurre es que la distancia disminuye tendiendo a cero (sin alcanzarlo por error numérico) y luego aumenta, representando efectivamente una colisión. La diferencia entre ambos casos radica en el comportamiento de las distancias durante los primeros pasos de la simulación.
- La búsqueda de trayectorias elípticas estables resultó en casos muy escasos. En su mayoría, los cuerpos escapan o colisionan, visualizándose

en las gráficas como rectas o parábolas.

- La mayoría de las órbitas encontradas eran elipses formadas por espirales que eventualmente desembocaban en trayectorias rectilíneas.
- Para obtener un sistema solar estable (sin planetas escapando), fue necesario realizar miles de pruebas, variando los parámetros de masa, posición y velocidad.
- Es necesario trabajar con rangos específicos para masas, posiciones y velocidades. La precisión del método numérico implica un costo computacional que puede compensarse reduciendo el número de pasos de simulación. Para pruebas iniciales, conviene usar hasta 1/1000 del número de pasos original.
- Las simulaciones del big bang resultaron meramente caóticas, observándose trayectorias parabólicas, en forma de moños, completas. No se observó la formación de un sistema solar estable, y al variar las condiciones de velocidad (perpendicular o paralela) se obtuvieron gráficas muy similares.
- Los comportamientos parabólicos, en moños y semicirculares en sistemas de N cuerpos fueron igualmente escasos, requiriendo variar masas, velocidades y distribución de posiciones para obtenerlos (ver tabla 1 y tabla 2).

Nº Cuerpos	Nº Órbitas (cerradas, espirales o listones)	Condiciones Iniciales
2	5	$r_0 = 1.20, v_0 = 0.90$ $r_0 = 1.50, v_0 = 0.90$ $r_0 = 1.60, v_0 = 0.90$ $r_0 = 2.00, v_0 = 0.90$ $r_0 = 30.17, v_0 = 0.08$
3	7	Configuración 1: $P_1 = (-0.5, 0, 0), P_2 = (0.5, 0, 0), P_3 = (0, 0.8660, 0)$ $V_1 = (0, -1, 0), V_2 = (0, 1, 0), V_3 = (-1, 0, 0)$ $V_1 = (0, -0.1, 0), V_2 = (0, 0.1, 0), V_3 = (-0.1, 0, 0)$ $V_1 = (0, -0.2, 0), V_2 = (0, 0.2, 0), V_3 = (-0.2, 0, 0)$ $V_1 = (0, -0.3, 0), V_2 = (0, 0.3, 0), V_3 = (-0.2, 0, 0)$ $V_1 = (0, -0.4, 0), V_2 = (0, 0.4, 0), V_3 = (-0.4, 0, 0)$ Configuración 2: $P_1 = (-0.5, 0, 0), P_2 = (0.5, 0, 0), P_3 = (0, 0.8660, 0)$ $V_1 = (0, -0.5, 0), V_2 = (0, 0.5, 0), V_3 = (-0.5, 0, 0)$ $V_1 = (0, -1, 0), V_2 = (0, 1, 0), V_3 = (-1, 0, 0)$
4	1	$P_1 = (0.5, 0, 0), P_2 = (0, 0.8660, 0),$ $P_3 = (-0.5, 0, 0), P_4 = (0, -0.8660, 0),$ $V_1 = (0, 0.01, 0), V_2 = (-0.01, 0, 0),$ $V_3 = (0, -0.01, 0), V_4 = (0.01, 0, 0)$
500	3	Posiciones: Distribución uniforme dentro de un disco de radio máximo $R = 10$. en el plano xy .
9	1	Posiciones: Distribución uniforme dentro de un disco de radio máximo $R = 10$. en el plano xy . A excepción del cuerpo 1, $M_1 = 1000, P_1 = (0, 0, 0),$ $V_1 = (0, 0, 0)$, tal que toda $m < M_1$

Cuadro 7.1: Órbitas cerradas, espirales o listones

Apéndice A

Códigos en Fortran

Programa 1

```
1  program sol_doscuerpos
2  !* * * * *
3  ! El siguiente programa soluciona el sistema de
4  ! ecuaciones que describen el movimiento de una
5  ! nave espacial, sometido a las fuerzas gravita-
6  ! cionales de la luna y la tierra, aplicando el
7  ! metodo Runge-Kutta de cuarto orde.
8  !
9  ! Considerando: una luna en movimiento circular
10 ! alrededor de la tierra
11 !
12 ! Imprimiendo en pantalla las condiciones
13 ! de velocidad y distancia para que la nave
14 ! consiga una orbita estable
15 !
16 !* * * * *
17
18 implicit none
19 !var posicion y velocidad
20 real::x,y,vx,vy,r,v,t,r0
21 real::mm,xm,ym,dm,theta0
22 !metodo kutta
23 real::k1,k2,k3,k4,l1,l2,l3,l4,kk1,kk2,kk3,kk4,l11,l12,l13,l14
24 real::h,dt,pasos
```

```

25  !funciones
26  real:: f1,f2,f3,f4
27  !Variables de archivos
28  integer*8::unit,dr,dv,n,n2,n3,i,ii,iii
29  character(len=20)::filename
30  !Variables de, son para escribir de 100 en 100 puntos en archivo
31  real::v1,v0,cont,de,de2,de3,ri,vi
32  real::STOP,ra
33  real::orbita, or
34  real::alfa=1.5327E18 !alfa=me*G*dtl
35  !alfa variable para reescalar, considerando que el programa
36  !resuelve para un sistema normalizado
37
38  !Energia cinetica y Potencial
39  real::u,k,Et,Etant,Etantc,tol
40      !Etant==Energia anterior, Etantc==contador de constante
41  n=1.E10 ! Numero de pasos
42  dt=1.E6 !Tiempo
43
44
45  !Apicando el metodo
46  !h=1.e-03
47  h=dt/n
48  tol=1.E-15 !tolerancia para verificar conservacion de la energia
49
50
51  Do i=30, 30
52  Do ii=15, 20
53  !**** POSICION****
54  x=i
55  y=0.
56  r=sqrt(x**2+y**2)
57  r0=r
58  ra=r!Guardamos r anterior
59  !Datos
60  t=0.
61  !****velocidad****
62  vx=0.
63  vy=ii*(1./100.)
64  v=sqrt(vx**2+vy**2 )

```

```

65  v0=sqrt(vx**2+vy**2)
66  xm=0.
67  ym=60.33
68  theta0=atan2(ym, xm)
69
70  !Distancia Tierra Luna Reescalada
71  dm=sqrt((x-xm)**2+(y-ym)**2) !Distancia Nave_Luna
72
73  de=100000.
74
75  !de=100.
76  STOP=0.
77  Etantc=0.
78  cont=0.
79
80  !Calculando Energia total a un t=0
81  !call Energia(x,y,vx,vy,xm,ym,k,u,Et)
82  ! Definir el nombre del archivo
83  write(filename, '(a, i0, i0, a)') 'datos', i, ii, '.dat'
84  ! Abrir el archivo para escritura
85  open(unit=25, file=filename, status='unknown')
86  open(unit=10, file='Moondat.txt', status='unknown', action='write')
87
88
89  !write(25,'(10(F13.7))') t, x, y, r, vx, vy, v, k, u, Et
90  write(25,*) t, x, y, r, vx, vy, v, k, u, Et
91  write(10,*) t, xm, ym
92
93  !APLICANDO EL METODO
94  Do iii=1, (n)
95  ra=r
96  call Runge_Kutta(x,y,vx,vy,xm,ym,h)
97  call Movluna(t,xm,ym,theta0)
98  !thetaant=atan2(ym, xm)
99  Etant=Et
100  call Energia(x,y,vx,vy,xm,ym,k,u,Et)
101
102  t=t+h
103  r=sqrt(x**2+y**2)
104  v=sqrt(vx**2+vy**2 )

```

```

105  !*****
106  !Escritura de datos en archivo, se escribirán
107  !de 10 en 10 o de 100 en 100 con variable "de"
108  if(iii==de )then
109  !write(25,'(10(F13.7))') t, x, y, r, vx, vy, v, k, u, Et
110  write(25,*) t, x, y, r, vx, vy, v, k, u, Et
111  write(10,*) t, xm, ym
112
113  de=de+100000.
114
115
116
117
118  end if
119  !*****
120  !*****
121  !      Analisis punto a punto, posibles casos
122  !
123  !      a) Colision con la tierra
124  !      b) Colision con la luna
125  !      c) Escape al espacio
126
127  if(r<1.0)then
128  write(*,*) "colisión con la tierra en r= ",r0 ,"v=",v0
129  STOP=1. ! Ya choco, no tiene caso seguir calculando
130  end if
131
132  !Colisión Luna
133  !Distancia entre nave luna
134  dm=sqrt((x-xm)**2.+(y-ym)**2.)
135  if(dm<0.2093)then
136  write(*,*)"colisión con la luna en r0= ",r0,"v0=",v0
137  STOP=1. ! Ya choco, no tiene caso seguir calculando
138  end if
139  !*****
140  !Si STOP es igual a 0 ha este momento tenemos una posible orbita
141  if(STOP==1.)exit
142
143  if(iii==(n))then
144  write(*,*)"Posible órbita en r= ",r0,"v=",v0

```

```

145 end if
146
147 end do
148
149 write(*,*)"Archivo generado: datos", i, ii, ".dat"
150 close(25)
151 close(10)
152 end do
153 end do
154
155 end program sol_doscuerpos
156
157 real function f1(x,y,vx,vy,xm,ym)
158 real::x,y,vx,vy
159 !constantes
160 real::G,me,xe,ye
161 real::xm,ym,mm
162 G=1. !Nm2/kg2
163 me=1. ! kg
164 xe=0.
165 ye=0.
166 mm=(7.35E22/5.972E24)
167 f1=(-G*me*(x-xe))/((sqrt((x-xe)**2 + (y-ye)**2 ))**3 )+&
168 (-G*mm*(x-xm))/((sqrt((x-xm)**2 + (y-ym)**2 ))**3 )
169 end function
170 real function f2(x,y,vx,vy)
171 real::x,y,vx,vy
172 f2=vx
173 end function
174 real function f3(x,y,vx,vy,xm,ym)
175 real::x,y,vx,vy
176 !constantes
177 real::G,me,xe,ye
178 real::xm,ym,mm
179 G=1. !Nm2/kg2
180 me=1. ! kg
181 xe=0.
182 ye=0.
183 mm=(7.35E22/5.972E24)
184 f3=(-G*me*(y-ye))/((sqrt((x-xe)**2 + (y-ye)**2 ))**3 )+&

```

```

185  (-G*mm*(y-ym))/((sqrt((x-xm)**2 + (y-ym)**2 ))**3 )
186  end function
187
188  real function f4(x,y,vx,vy)
189  real::x,y,vx,vy
190  f4=vy
191  end function
192
193  subroutine Runge_Kutta(x,y,vx,vy,xm,ym,h)
194  real::x,y,vx,vy,xm,ym,h
195  real::k1,k2,k3,k4,l1,l2,l3,l4,kk1,kk2,kk3,kk4,l11,l12,l13,l14
196
197  k1=h*f1(x,y,vx,vy,xm,ym)
198  l1=h*f2(x,y,vx,vy)
199  kk1=h*f3(x,y,vx,vy,xm,ym)
200  ll1=h*f4(x,y,vx,vy)
201
202  k2=h*f1((x+(l1/2.)),(y+(l11/2.)),(vx+(k1/2.)),(vy+(kk1/2. )),xm,ym)
203  l2=h*f2((x+(l1/2.)),(y+(l11/2.)),(vx+(k1/2.)),(vy+(kk1/2. )))
204  kk2=h*f3((x+(l1/2.)),(y+(l11/2.)),(vx+(k1/2.)),(vy+(kk1/2. )),xm,ym)
205  ll2=h*f4((x+(l1/2.)),(y+(l11/2.)),(vx+(k1/2.)),(vy+(kk1/2. )))
206
207  k3=h*f1((x+(l2/2.)),(y+(l12/2.)),(vx+(k2/2.)),(vy+(kk2/2. )),xm,ym)
208  l3=h*f2((x+(l2/2.)),(y+(l12/2.)),(vx+(k2/2.)),(vy+(kk2/2. )))
209  kk3=h*f3((x+(l2/2.)),(y+(l12/2.)),(vx+(k2/2.)),(vy+(kk2/2. )),xm,ym)
210  ll3=h*f4((x+(l2/2.)),(y+(l12/2.)),(vx+(k2/2.)),(vy+(kk2/2. )))
211
212  k4=h*f1((x+l3),(y+l13),(vx+k3),(vy+kk3),xm,ym)
213  l4=h*f2((x+l3),(y+l13),(vx+k3),(vy+kk3))
214  kk4=h*f3((x+l3),(y+l13),(vx+k3),(vy+kk3),xm,ym)
215  ll4=h*f4((x+l3),(y+l13),(vx+k3),(vy+kk3))
216
217  vx=vx+((1./6.)*(k1+(2.*k2)+(2.*k3)+k4))
218  x=x+((1./6.)*(l1+(2.*l2)+(2.*l3)+l4))
219
220  vy=vy+((1./6.)*(kk1+(2.*kk2)+(2.*kk3)+kk4))
221  y=y+((1./6.)*(ll1+(2.*ll2)+(2.*ll3)+ll4))
222  end subroutine Runge_Kutta
223
224  subroutine Movluna(t,xm,ym,theta0)

```

```

225  real::t, xm, ym
226  real::w, periodo, theta, r, theta0
227  pi=3.1415926
228  periodo=3166.6
229  r=sqrt(xm**2.+ym**2.)
230  w=(2.*pi)/periodo
231  theta=(w*t)+theta0
232  xm=cos(theta)*r
233  ym=sin(theta)*r
234  end subroutine
235
236  subroutine Energia(x,y,vx,vy,xm,ym,k,u,Et)
237  real::x,y,vx,vy,k,u,Et !datos nave
238  real::G,m,me,mm,d,xe,ye,xm,ym,v
239  real::r,re,rm
240  !d Distancia entre el centro de masa del sistema
241  !      tierra luna
242  G=1.
243  me=1.
244  mm=(7.35E22/5.972E24)
245  m=(1000./5.972E24) !Aprox masa de una nave espacial
246  !xm=0.
247  !ym=1.0211
248  xe=0.
249  ye=0.
250  ! Velocidad de la nave
251  v=sqrt(vx**2.+vy**2.)
252  !Calculando Energia Cinetica
253  k=0.5*m*(v**2.)
254  !Calculando Energia Potencial
255  r=sqrt(x**2.+y**2)
256  re=sqrt(xe**2.+ye**2)
257  rm=sqrt(xm**2.+ym**2)
258  u=((-G*me)/(abs(r-re)))+((-G*mm)/(abs(r-rm)))
259  !Energian Total Et
260  Et=k+u
261  end subroutine Energia

```

Programa 2

```
1 program three_body_problem
2 implicit none
3 ! Constantes
4 real(8), parameter :: G = 1.
5 real(8), parameter :: m_1 = 1., m_2 = 1., m_3 = 1.
6 ! Condiciones iniciales
7 real(8), dimension(3) :: p1 = [-0.5, 0.0, 0.0]
8 real(8), dimension(3) :: p2 = [0.5, 0.0, 0.0]
9 real(8), dimension(3) :: p3 = [0., 0.866025404, 0.0]
10 real(8), dimension(3) :: v1 = [0.0, 0.0, 0.0]
11 real(8), dimension(3) :: v2 = [0.0, 0.0, 0.0]
12 real(8), dimension(3) :: v3 = [0.0, 0.0, 0.0]
13 real(8), dimension(3) :: a1, a2, a3
14 ! Parámetros de simulación
15 real(8), parameter :: delta_t = 0.00001d0
16 integer, parameter :: steps = 1000000
17 integer :: i, ii, unit, de
18 character(len=20) :: filename
19 integer(kind=8) :: j
20 unit=25
21
22 !Probando multiples condiciones
23 Do i=1, 5
24 de=100
25 write(filename, '(A, I0, A)') 'trayectoriaseu', i, '.dat'
26 open(unit=25, file=filename, status='unknown')
27 p1 = [-0.5, 0.0, 0.0]
28 p2 = [0.5, 0.0, 0.0]
29 p3 = [0., 0.866025404, 0.0]
30 !Condiciones Iniciales velocidad
31 v1 = [0.0, -1.0, 0.0]
32 v2 = [0.0, 1.0, 0.0]
33 v3 = [-1.0, 0.0, 0.0]
34 write(*,*)"Iniciando simulación, i= ", i
35 write(25, *) p1, p2, p3
36 ! Bucle principal de simulaciones
37 do ii = 1, steps - 1
38 ! Calcular aceleraciones
```

```

39  call accelerations(p1, p2, p3, a1, a2, a3)
40  ! Actualizar velocidades y posiciones usando el método de Euler
41  p1 = p1 + v1 * delta_t
42  v1 = v1 + a1 * delta_t
43  p2 = p2 + v2 * delta_t
44  v2 = v2 + a2 * delta_t
45  p3 = p3 + v3 * delta_t
46  v3 = v3 + a3 * delta_t
47  if (de==ii)then !Escribiendo datos de 100 en 100
48  write(25, *) p1, p2, p3
49  de=de+100.
50  end if
51  end do
52  close(25)
53  write(*,*) " Resultados guardados en trayectoriaseu", i, ".dat"
54  end do
55  end program three_body_problem
56
57  subroutine accelerations(p1, p2, p3, a1, a2, a3)
58  implicit none
59  real(8), dimension(3), intent(in) :: p1, p2, p3
60  real(8), dimension(3), intent(out) :: a1, a2, a3
61  real(8) :: r12, r13, r23
62  real(8):: G, m_1, m_2, m_3
63  m_1=1.0
64  m_2=1.0
65  m_3=1.0
66  G=1.0
67  ! Distancias entre cuerpos
68  r12 = sqrt(sum((p1 - p2)**2))
69  r13 = sqrt(sum((p1 - p3)**2))
70  r23 = sqrt(sum((p2 - p3)**2))
71  ! Aceleraciones
72  a1 = -G * m_2 * (p1 - p2) / r12**3 - G * m_3 * (p1 - p3) / r13**3
73  a2 = -G * m_3 * (p2 - p3) / r23**3 - G * m_1 * (p2 - p1) / r12**3
74  a3 = -G * m_1 * (p3 - p1) / r13**3 - G * m_2 * (p3 - p2) / r23**3
75  end subroutine accelerations

```

Programa 3

A continuación se muestra la subrutina y las funciones de la implementación en fortran del método Runge kutta de cuarto orden, utilizando arreglos de tres dimensiones. Las k 's se relacionan con la velocidad, mientras que las l 's se relacionan con la posición, el primer subíndice especifica el cuerpo para el que se relaciona

```
1  subroutine Runge_Kutta(p1,p2,p3,v1,v2,v3,h)
2  real(8), dimension(3)::p1,p2,p3
3  real(8), dimension(3)::v1,v2,v3
4  real(8)::h
5  real(8)::x1,y1,z1,vx1,vy1,vz1
6  real(8)::k1_1,k1_2,k1_3,k1_4,l1_1,l1_2,l1_3,l1_4,&
7  kk1_1,kk1_2,kk1_3,kk1_4,ll1_1,ll1_2,ll1_3,ll1_4
8  real(8)::kkk1_1,kkk1_2,kkk1_3,kkk1_4,lll1_1,lll1_2,lll1_3,lll1_4
9  ! -----
10 real(8)::x2,y2,z2,vx2,vy2,vz2
11 real(8)::k2_1,k2_2,k2_3,k2_4,l2_1,l2_2,l2_3,l2_4,&
12 kk2_1,kk2_2,kk2_3,kk2_4,ll2_1,ll2_2,ll2_3,ll2_4
13 real(8)::kkk2_1,kkk2_2,kkk2_3,kkk2_4,lll2_1,lll2_2,lll2_3,lll2_4
14 ! -----
15 real(8)::x3,y3,z3,vx3,vy3,vz3
16 real(8)::k3_1,k3_2,k3_3,k3_4,l3_1,l3_2,l3_3,l3_4,&
17 kk3_1,kk3_2,kk3_3,kk3_4,ll3_1,ll3_2,ll3_3,ll3_4
18 real(8)::kkk3_1,kkk3_2,kkk3_3,kkk3_4,&
19 lll3_1,lll3_2,lll3_3,lll3_4
20 x1 = p1(1); x2 = p2(1); x3 = p3(1)
21 y1 = p1(2); y2 = p2(2); y3 = p3(2)
22 z1 = p1(3); z2 = p2(3); z3 = p3(3)
23 vx1 = v1(1); vx2 = v2(1); vx3 = v3(1)
24 vy1 = v1(2); vy2 = v2(2); vy3 = v3(2)
25 vz1 = v1(3); vz2 = v2(3); vz3 = v3(3)
26 !cuerpo 1 bajo la acción del cuerpo 2 y cuerpo 3
27 k1_1=h*f1_1(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,&
28 x3,y3,z3,vx3,vy3,vz3)
29 l1_1=h*f1_2(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,&
30 x3,y3,z3,vx3,vy3,vz3)
31 kk1_1=h*f1_3(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,&
32 x3,y3,z3,vx3,vy3,vz3)
```

```

33  l11_1=h*f1_4(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,&
34  x3,y3,z3,vx3,vy3,vz3)
35  kkk1_1=h*f1_5(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,&
36  x3,y3,z3,vx3,vy3,vz3)
37  l111_1=h*f1_6(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,&
38  x3,y3,z3,vx3,vy3,vz3)
39
40  k1_2=h*f1_1((x1+(l1_1/2.)),(y1+(l11_1/2.)),(z1+(l111_1/2.)), &
41  (vx1+(k1_1/2.)),(vy1+(kk1_1/2.)),(vz1+(kkk1_1/2.)), &
42  x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
43  l1_2=h*f1_2((x1+(l1_1/2.)),(y1+(l11_1/2.)),(z1+(l111_1/2.)), &
44  (vx1+(k1_1/2.)),(vy1+(kk1_1/2.)),(vz1+(kkk1_1/2.)), &
45  x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
46  kk1_2=h*f1_3((x1+(l1_1/2.)),(y1+(l11_1/2.)),(z1+(l111_1/2.)), &
47  (vx1+(k1_1/2.)),(vy1+(kk1_1/2.)),(vz1+(kkk1_1/2.)), &
48  x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
49  l11_2=h*f1_4((x1+(l1_1/2.)),(y1+(l11_1/2.)),(z1+(l111_1/2.)), &
50  (vx1+(k1_1/2.)),(vy1+(kk1_1/2.)),(vz1+(kkk1_1/2.)), &
51  x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
52  kkk1_2=h*f1_5((x1+(l1_1/2.)),(y1+(l11_1/2.)),(z1+(l111_1/2.)), &
53  (vx1+(k1_1/2.)),(vy1+(kk1_1/2.)),(vz1+(kkk1_1/2.)), &
54  x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
55  l111_2=h*f1_6((x1+(l1_1/2.)),(y1+(l11_1/2.)),(z1+(l111_1/2.)), &
56  (vx1+(k1_1/2.)),(vy1+(kk1_1/2.)),(vz1+(kkk1_1/2.)), &
57  x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
58  k1_3=h*f1_1((x1+(l1_2/2.)),(y1+(l11_2/2.)),(z1+(l111_2/2.)), &
59  (vx1+(k1_2/2.)),(vy1+(kk1_2/2.)),(vz1+(kkk1_2/2.)), &
60  x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
61  l1_3=h*f1_2((x1+(l1_2/2.)),(y1+(l11_2/2.)),(z1+(l111_2/2.)), &
62  (vx1+(k1_2/2.)),(vy1+(kk1_2/2.)),(vz1+(kkk1_2/2.)), &
63  x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
64  kk1_3=h*f1_3((x1+(l1_2/2.)),(y1+(l11_2/2.)),(z1+(l111_2/2.)), &
65  (vx1+(k1_2/2.)),(vy1+(kk1_2/2.)),(vz1+(kkk1_2/2.)), &
66  x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
67  l11_3=h*f1_4((x1+(l1_2/2.)),(y1+(l11_2/2.)),(z1+(l111_2/2.)), &
68  (vx1+(k1_2/2.)),(vy1+(kk1_2/2.)),(vz1+(kkk1_2/2.)), &
69  x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
70  kkk1_3=h*f1_5((x1+(l1_2/2.)),(y1+(l11_2/2.)),(z1+(l111_2/2.)), &
71  (vx1+(k1_2/2.)),(vy1+(kk1_2/2.)),(vz1+(kkk1_2/2.)), &
72  x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)

```

```

73 l1l1_3=h*f1_6((x1+(l1_2/2.)),(y1+(l1l1_2/2.)),(z1+(l1l1_2/2.)), &
74 (vx1+(k1_2/2.)),(vy1+(kk1_2/2.)),(vz1+(kkk1_2/2.)), &
75 x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
76 k1_4=h*f1_1((x1+(l1_3)),(y1+(l1l1_3)),(z1+(l1l1_3)), &
77 (vx1+(k1_3)),(vy1+(kk1_3)),(vz1+(kkk1_3)), &
78 x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
79 l1_4=h*f1_2((x1+(l1_3)),(y1+(l1l1_3)),(z1+(l1l1_3)), &
80 (vx1+(k1_3)),(vy1+(kk1_3)),(vz1+(kkk1_3)), &
81 x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
82 kk1_4=h*f1_3((x1+(l1_3)),(y1+(l1l1_3)),(z1+(l1l1_3)), &
83 (vx1+(k1_3)),(vy1+(kk1_3)),(vz1+(kkk1_3)), &
84 x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
85 l1l1_4=h*f1_4((x1+(l1_3)),(y1+(l1l1_3)),(z1+(l1l1_3)), &
86 (vx1+(k1_3)),(vy1+(kk1_3)),(vz1+(kkk1_3)), &
87 x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
88 kkk1_4=h*f1_5((x1+(l1_3)),(y1+(l1l1_3)),(z1+(l1l1_3)), &
89 (vx1+(k1_3)),(vy1+(kk1_3)),(vz1+(kkk1_3)), &
90 x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
91 l1l1_4=h*f1_6((x1+(l1_3)),(y1+(l1l1_3)),(z1+(l1l1_3)), &
92 (vx1+(k1_3)),(vy1+(kk1_3)),(vz1+(kkk1_3)), &
93 x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
94 !-----
95 !Similarmente para Cuerpo 2 bajo la accion de cuerpo 1 y
96 !cuerpo 3
97 !-----
98 !Similarmente para cuerpo 3 bajo la accion de cuerpo 1 y
99 !cuerpo 2
100
101 ! Actualización de posiciones y velocidades
102 vx1=vx1+((1./6.)*(k1_1+(2.*k1_2)+(2.*k1_3)+k1_4))
103 x1= x1+((1./6.)*(l1_1+(2.*l1_2)+(2.*l1_3)+l1_4))
104
105 vy1=vy1+((1./6.)*(kk1_1+(2.*kk1_2)+(2.*kk1_3)+kk1_4))
106 y1 =y1+((1./6.)*(l1l1_1+(2.*l1l1_2)+(2.*l1l1_3)+l1l1_4))
107
108 vz1=vz1+((1./6.)*(kkk1_1+(2.*kkk1_2)+(2.*kkk1_3)+kkk1_4))
109 z1=z1+((1./6.)*(l1l1_1+(2.*l1l1_2)+(2.*l1l1_3)+l1l1_4))
110
111 vx2=vx2+((1./6.)*(k2_1+(2.*k2_2)+(2.*k2_3)+k2_4))
112 x2=x2+((1./6.)*(l2_1+(2.*l2_2)+(2.*l2_3)+l2_4))

```

```

113
114 vy2=vy2+((1./6.)*(kk2_1+(2.*kk2_2)+(2.*kk2_3)+kk2_4))
115 y2 =y2+((1./6.)*(l12_1+(2.*l12_2)+(2.*l12_3)+l12_4))
116
117 vz2=vz2+((1./6.)*(kkk2_1+(2.*kkk2_2)+(2.*kkk2_3)+kkk2_4))
118 z2=z2+((1./6.)*(l112_1+(2.*l112_2)+(2.*l112_3)+l112_4))
119
120 vx3=vx3+((1./6.)*(k3_1+(2.*k3_2)+(2.*k3_3)+k3_4))
121 x3=x3+((1./6.)*(l3_1+(2.*l3_2)+(2.*l3_3)+l3_4))
122
123 vy3=vy3+((1./6.)*(kk3_1+(2.*kk3_2)+(2.*kk3_3)+kk3_4))
124 y3=y3+((1./6.)*(l13_1+(2.*l13_2)+(2.*l13_3)+l13_4))
125
126 vz3=vz3+((1./6.)*(kkk3_1+(2.*kkk3_2)+(2.*kkk3_3)+kkk3_4))
127 z3=z3+((1./6.)*(l113_1+(2.*l113_2)+(2.*l113_3)+l113_4))
128 !Actualización a los arreglos
129 p1(1)=x1; p2(1)=x2; p3(1)=x3
130 p1(2)=y1; p2(2)=y2; p3(2)=y3
131 p1(3)=z1; p2(3)=z2; p3(3)=z3
132
133 v1(1)=vx1; v2(1)=vx2; v3(1)=vx3
134 v1(2)=vy1; v2(2)=vy2; v3(2)=vy3
135 v1(3)=vz1; v2(3)=vz2; v3(3)=vz3
136
137 end subroutine Runge_Kutta
138
139 !***** Funciones para cuerpo 1 sometido a la gravitación del cuerpo 2 y 3
140 REAL function f1_1(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
141 real(8)::x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3
142 real(8)::G
143 real(8)::m1,m2,m3
144 m1= 1.0D0
145 m2= 1.0D0
146 m3= 1.0D0
147 G = 1.0D0
148 f1_1=(-G*m2*(x1-x2))/((sqrt((x1-x2)**2+(y1-y2)**2+(z1-z2)**2)+(1e-10))**3) + &
149      (-G*m3*(x1-x3))/((sqrt((x1-x3)**2+(y1-y3)**2+(z1-z3)**2)+(1e-10))**3)
150 END function
151 REAL function f1_2(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
152 real(8)::x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3

```

```

153  f1_2=vx1
154  END function
155  REAL function f1_3(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
156  real(8)::x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3
157  real(8)::G
158  real(8)::m1,m2,m3
159  m1= 1.0D0
160  m2= 1.0D0
161  m3= 1.0D0
162  G = 1.0D0
163  f1_3=(-G*m2*(y1-y2))/((sqrt((x1-x2)**2+(y1-y2)**2+(z1-z2)**2)+(1e-10))**3)+ &
164      (-G*m3*(y1-y3))/((sqrt((x1-x3)**2+(y1-y3)**2+(z1-z3)**2)+(1e-10))**3)
165  END function
166  REAL function f1_4(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
167  real(8)::x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3
168  f1_4=vy1
169  END function
170
171  REAL function f1_5(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
172  real(8)::x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3
173  real(8)::G
174  real(8)::m1,m2,m3
175  m1= 1.0D0
176  m2= 1.0D0
177  m3= 1.0D0
178  G = 1.0D0
179  f1_5=(-G*m2*(z1-z2))/((sqrt((x1-x2)**2+(y1-y2)**2+(z1-z2)**2)+(1e-10))**3)+ &
180      (-G*m3*(z1-z3))/((sqrt((x1-x3)**2+(y1-y3)**2+(z1-z3)**2)+(1e-10))**3)
181  END function
182
183  REAL function f1_6(x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3)
184  real(8)::x1,y1,z1,vx1,vy1,vz1,x2,y2,z2,vx2,vy2,vz2,x3,y3,z3,vx3,vy3,vz3
185  f1_6=vz1
186  END function
187  !*****
188  !Similarmente para; las
189  !Funciones del cuerpo 2,
190  !sometido a la gravitación
191  !del cuerpo 1 y 3. y las
192  !Funciones del cuerpo 3

```

```

193  !sometido a la gravitación
194  !del cuerpo 1 y 2.

```

Programa 4

```

1  program N_body_problem
2  !*****
3  !Modificanco three_body_problem para N>3
4  !*****
5  use RK4_mod, only:Runge_kutta
6  implicit none
7  integer, parameter :: N = 4
8  real(8), dimension (N,3) :: P, V, A ! Matriz Posiciones, Velocidades, Aceleraciones
9  real(8),dimension (N)::M= 1.0d0 !masas
10 character(len=50)::filename
11 integer::unit
12 real(8) :: G = 1.0d0
13 real(8), parameter :: delta_t = 0.0001d0
14 integer, parameter :: steps = 1000000
15 integer :: i, ii, de, j
16 integer::archivo
17 unit=2.
18 j=1.
19 !Bucle para probar múltiples condiciones iniciales
20 Do i=1, 1
21 de=100
22 Do archivo=1, 4
23 write(filename, '(A, I0, A)') 'BODY', archivo, '.dat'
24 open(archivo, file=filename, status='unknown', position='append')
25 end do
26 !Todas las masas =1
27 !M(:)=1.0d0
28 ! ***** Condiciones Iniciales *****
29 ! Posiciones Iniciales
30 !call random_number(P)
31 !P(1,:)= [1.0, 2.0, 3.0]
32 !call random_number(V)
33 P(1,:) = [ 0.5, 0.0, 0.0]
34 P(2,:) = [ 0.0, 0.866025404, 0.0]

```

```

35 P(3,:) = [-0.5, 0.0, 0.0]
36 P(4,:) = [ 0.0,-0.866025404, 0.0]
37 !Velocidades Iniciales
38 v(1,:) = [0.0, (0.01*real(i)), 0.0]
39 v(2,:) = [(-0.01*real(i)), 0.0, 0.0]
40 v(3,:) = [0.0,(-0.01*real(i)), 0.0]
41 v(4,:) = [(0.01*real(i)), 0.0, 0.0]
42
43 A(:, :) = 0.0
44 !*****
45 Do Archivo=1, 4
46 write(archivo,*) P(Archivo,:)
47 end do
48
49 ! Bucle principal de la simulación
50 do ii = 1, (steps-1-900000)
51
52 call Runge_kutta(P, V, A, M, G, N, delta_t)
53
54 if(ii==de )then
55 Do Archivo=1, 4
56 write(archivo,*) P(Archivo,:)
57 end do
58 de=de+100
59 end if
60 end do
61 Do archivo=1, N
62 close(archivo)
63 end do
64 end do
65 end program N_body_problem

```

Modulo, Aplicación del método numérico

```

1 module RK4_mod
2 implicit none
3 contains
4 Function aceleracion(P,V,M) result(A)
5 integer, parameter:: N=4

```

```

6  real(8), intent(in):: P(N,3), V(N,3), M(N)
7  real(8) ::A(N,3), rij(N), d, G
8  integer::i,j,cuerpoi
9  ! Calculando la Aceleración del cuerpo i
10 ! sometido a la fuerza de los N cuerpos
11 G=1.0d0
12 A(:, :) = 0.0
13 Do i=1, N
14 Do j=1, N
15 if(i==j) cycle
16 ! Distancias entre los cuerpos
17 ! rij = sqrt(sum((P(i,:) - P(j,:))**2))!Diferencia vectore (rj-ri)
18 !Aceleraciones
19 A(i,:)=A(i,:) -G * M(j) * ( (P(i,:) - P(j,:) ) /(sqrt(sum((P(i,:) - P(j,:))**2)) )**3)
20 end Do
21 end Do
22 end Function aceleracion
23
24 subroutine Runge_kutta(P, V, A, M, G, N, h)
25 real(8):: h, G
26 integer :: N
27 real(8), dimension(N, 3) :: P, V, A ! Matriz Posiciones y Velocidades
28 real(8), dimension(N):: M !masas
29 real(8), dimension(N, 3)::k1,k2,k3,k4,l1,l2,l3,l4
30 real(8), dimension(N, 3)::P_tem, V_tem
31
32 A = aceleracion(P,V,M)
33 k1 = h*A
34 l1 = h*V
35
36 P_tem = P + 0.5 * l1
37 V_tem = V +0.5 * k1
38 A = aceleracion(P_tem,V_tem,M)
39
40 k2 = h*A
41 l2 = h*V_tem
42
43 P_tem = P + 0.5 * l2
44 V_tem = V +0.5 * k2
45 A = aceleracion(P_tem,V_tem,M)

```

```

46
47 k3 = h*A
48 l3 = h*V_tem
49
50 P_tem = P + l3
51 V_tem = V + k3
52 A = aceleracion(P_tem,V_tem,M)
53
54 k4 = h*A
55 l4 = h*V_tem
56 ! --- Actualización final ---
57 P = P + (l1 + 2.0*l2 + 2.0*l3 + l4)/6.0
58 V = V + (k1 + 2.0*k2 + 2.0*k3 + k4)/6.0
59 end subroutine Runge_kutta
60 end module RK4_mod

```

Programa 5

```

1 ! MÓDULO AL PRINCIPIO
2 !=====
3 ! VERSIÓN SIMPLIFICADA - PARALELIZACIÓN DEL BUCLE PRINCIPAL
4 !=====
5 module RK4_mod
6 implicit none
7 contains
8
9 function aceleracion_para_cuerpo(i, P, M, N) result(accel)
10 integer, intent(in) :: i, N
11 real(8), intent(in) :: P(N,3), M(N)
12 real(8) :: accel(3)
13 real(8) :: dx, dy, dz, dist, dist3, G
14 integer :: j
15
16 G = 1.0d0
17 accel = 0.0d0
18
19 do j = 1, N
20     if (i == j) cycle
21

```

```

22     dx = P(i,1) - P(j,1)
23     dy = P(i,2) - P(j,2)
24     dz = P(i,3) - P(j,3)
25
26     dist = sqrt(dx*dx + dy*dy + dz*dz)
27
28     if (dist > 1.0d-10) then
29         dist3 = dist * dist * dist
30         acel(1) = acel(1) - G * M(j) * dx / dist3
31         acel(2) = acel(2) - G * M(j) * dy / dist3
32         acel(3) = acel(3) - G * M(j) * dz / dist3
33     end if
34 end do
35
36 end function aceleracion_para_cuerpo
37
38 subroutine evolucion_paralela(P, V, M, G, N, h)
39 real(8), intent(in) :: h, G
40 integer, intent(in) :: N
41 real(8), dimension(N,3), intent(inout) :: P, V
42 real(8), dimension(N), intent(in) :: M
43 real(8), dimension(N,3) :: A, k1, k2, k3, k4, l1, l2, l3, l4
44 real(8), dimension(N,3) :: P_temp, V_temp
45 integer :: i
46
47 ! PARALELIZACIÓN POR CUERPOS - CADA THREAD CALCULA UN GRUPO DE CUERPOS
48 !$OMP PARALLEL DO PRIVATE(i) SHARED(P, V, A, M, G, N)
49 do i = 1, N
50     A(i,:) = aceleracion_para_cuerpo(i, P, M, N)
51 end do
52 !$OMP END PARALLEL DO
53
54 !$OMP PARALLEL DO PRIVATE(i) SHARED(P, V, A, k1, l1, h)
55 do i = 1, N
56     k1(i,:) = h * A(i,:)
57     l1(i,:) = h * V(i,:)
58 end do
59 !$OMP END PARALLEL DO
60
61 !$OMP PARALLEL DO PRIVATE(i) SHARED(P, V, k1, l1, P_temp, V_temp)

```

```

62  do i = 1, N
63      P_temp(i,:) = P(i,:) + 0.5d0 * l1(i,:)
64      V_temp(i,:) = V(i,:) + 0.5d0 * k1(i,:)
65  end do
66  !$OMP END PARALLEL DO
67
68  !$OMP PARALLEL DO PRIVATE(i) SHARED(P_temp, V_temp, A, M, G, N)
69  do i = 1, N
70      A(i,:) = aceleracion_para_cuerpo(i, P_temp, M, N)
71  end do
72  !$OMP END PARALLEL DO
73  ! ... continuar con los otros pasos de Runge-Kutta ...
74  !$OMP PARALLEL DO PRIVATE(i) SHARED(P, V, A, k2, l2, h)
75  do i = 1, N
76      k2(i,:) = h * A(i,:)
77      l2(i,:) = h * V_temp(i,:)
78  end do
79  !$OMP END PARALLEL DO
80
81  !$OMP PARALLEL DO PRIVATE(i) SHARED(P, V, k2, l2, P_temp, V_temp)
82  do i = 1, N
83      P_temp(i,:) = P(i,:) + 0.5d0 * l2(i,:)
84      V_temp(i,:) = V(i,:) + 0.5d0 * k2(i,:)
85  end do
86  !$OMP END PARALLEL DO
87
88  !$OMP PARALLEL DO PRIVATE(i) SHARED(P_temp, V_temp, A, M, G, N)
89  do i = 1, N
90      A(i,:) = aceleracion_para_cuerpo(i, P_temp, M, N)
91  end do
92  !$OMP END PARALLEL DO
93
94  !$OMP PARALLEL DO PRIVATE(i) SHARED(P, V, A, k3, l3, h)
95  do i = 1, N
96      k3(i,:) = h * A(i,:)
97      l3(i,:) = h * V_temp(i,:)
98  end do
99  !$OMP END PARALLEL DO
100
101  !$OMP PARALLEL DO PRIVATE(i) SHARED(P, V, k3, l3, P_temp, V_temp)

```

```

102  do i = 1, N
103      P_temp(i,:) = P(i,:) + l3(i,:)
104      V_temp(i,:) = V(i,:) + k3(i,:)
105  end do
106  !$OMP END PARALLEL DO
107
108  !$OMP PARALLEL DO PRIVATE(i) SHARED(P_temp, V_temp, A, M, G, N)
109  do i = 1, N
110      A(i,:) = aceleracion_para_cuerpo(i, P_temp, M, N)
111  end do
112  !$OMP END PARALLEL DO
113
114  !$OMP PARALLEL DO PRIVATE(i) SHARED(P, V, A, k4, l4, h)
115  do i = 1, N
116      k4(i,:) = h * A(i,:)
117      l4(i,:) = h * V_temp(i,:)
118  end do
119  !$OMP END PARALLEL DO
120  ! .....
121  ! Actualización final
122  !$OMP PARALLEL DO PRIVATE(i) SHARED(P, V, l1, l2, l3, l4, k1, k2, k3, k4)
123  do i = 1, N
124      P(i,:) = P(i,:) + (l1(i,:) + 2.0d0*l2(i,:) + 2.0d0*l3(i,:) + l4(i,:)) / 6.0d0
125      V(i,:) = V(i,:) + (k1(i,:) + 2.0d0*k2(i,:) + 2.0d0*k3(i,:) + k4(i,:)) / 6.0d0
126  end do
127  !$OMP END PARALLEL DO
128
129  end subroutine evolucion_paralela
130
131  end module RK4_mod
132
133  program N_body_problem_paralelo
134  use RK4_mod, only: evolucion_paralela
135  use omp_lib
136  implicit none
137
138  ! Parámetros
139  integer, parameter :: N = 500
140  real(8), parameter :: G = 1.0d0
141  real(8), parameter :: delta_t = 0.0001d0

```

```

142 integer, parameter :: steps = 10000!0000
143 real(8), parameter :: radio_maximo = 10.0d0 ! Radio de la circunferencia
144 real(8), parameter :: velocidad_magnitud = 10.0!1d0! Magnitud de la velocidad
145 ! Variables
146 real(8), dimension(N,3):: P, V, A
147 real(8), dimension(N) :: M
148 real(8), dimension(N):: ang_rand, radio
149 character(len=50) :: filename
150 integer :: i, ii, archivo
151 real(8) :: angulo, start_time, end_time
152 !integer :: num_threads
153 ! Abriendo Archivos
154 Do archivo=1, N
155 write(filename, '(A, IO, A)') 'Big_BODY', archivo, '.dat'
156 open(archivo, file=filename, status='unknown', position='append')
157 end do
158 ! Configurar threads
159 !call omp_set_num_threads(4)
160
161 ! Inicialización
162 ! ***** Condiciones iniciales *****
163 call random_number(M)
164 M=M*100.0
165 ! ***** Condiciones Iniciales *****
166 ! 1. POSICIONES: Distribución circular en el plano XY
167 ! Posiciones aleatorias dentro de un círculo
168 ! Generando angulos aleatorios
169 call random_number(ang_rand)
170 ang_rand = 2.0d0 * 3.141592653589793d0 * ang_rand
171 ! Genera radios aleatorios
172 call random_number(radio)
173 radio= radio_maximo * sqrt(radio)
174 call random_number(P)
175 do i = 1, N
176 !Calculando coordenadas
177 !P(i,1) = radio(i)*cos(ang_rand(i))
178 !P(i,2) = radio(i)*sin(ang_rand(i))
179 !P(i,3) = 0.0d0
180
181 angulo = 2.0d0 * 3.141592653589793d0 * (i-1) / real(N,8)

```

```

182      !call random_number(P)
183      !P=P*10.0
184      P(i,1) = radio(i) * cos(angulo)  ! Coordenada X
185      P(i,2) = radio(i) * sin(angulo)  ! Coordenada Y
186      P(i,3) = 0.0d0                  ! Coordenada Z (plano)
187      !distancia al origen
188      ! dis_o(i) = sqrt(P(i,1)**2 + P(i,2)**2 + P(i,3)**2)
189  end do
190  ! 2. VELOCIDADES: Tangentes a la circunferencia (perpendiculares al radio)
191
192  do i = 1, N
193      angulo = 2.0d0 * 3.141592653589793d0 * (i-1) / real(N,8)
194      V(i,1) = -velocidad_magnitud * sin(angulo)  ! Perpendicular al radio
195      V(i,2) = velocidad_magnitud * cos(angulo)   ! Perpendicular al radio
196      !V(i,1) = velocidad_magnitud * cos(angulo)  ! Paralelo al radio
197      ! V(i,2) = velocidad_magnitud * sin(angulo)  ! Paralelo al radio
198      V(i,3) = 0.0d0                               ! Componente Z
199  end do
200  ! Inicializar aceleraciones
201  A(:, :) = 0.0d0
202  24
203  ! Escribiendo condiciones iniciales
204  Do archivo=1, N
205  write(archivo,*) P(Archivo,:)
206  end do
207
208
209
210
211  ! BUCLE PRINCIPAL PARALELIZADO
212  start_time = omp_get_wtime()
213
214
215  do ii = 1, 1000 ! ((steps-1)/8)
216      ! ESTA LLAMADA ES PARALELA INTERNAMENTE
217      call evolucion_paralela(P, V, M, G, N, delta_t)
218
219      if (mod(ii, 1) == 0) then
220          print *, 'Paso:', ii, '/', steps
221

```

```

222         ! Escritura paralela de archivos
223         !$OMP PARALLEL DO PRIVATE(archivo) SHARED(P)
224         do archivo = 1, N
225             write(archivo, *) P(archivo,:)
226         end do
227         !$OMP END PARALLEL DO
228     end if
229 end do
230
231 end_time = omp_get_wtime()
232 print *, 'Tiempo total:', end_time - start_time, 'segundos'
233
234 Do archivo=1, N
235 close(Archivo)
236 end do
237 end program N_body_problem_paralelo

```

Programa 6

Modificación del programa 5, para simular el sistema solar. A diferencia del programa 5 este tiene 9 cuerpos, distribuidos de forma uniforme, pero con radios aleatorios, los radios o distancia al origen son aleatorios entre (1, 9.99), siendo igual para las velocidades. Además se guardan las condiciones iniciales del sistema, para poder reproducirlas de nuevo si se desea.

```

1  program Sistema_solar
2  use RK4_mod, only: evolucion_paralela
3  use omp_lib
4  implicit none
5
6  ! Parámetros
7  integer, parameter :: N = 10
8  real(8), parameter :: G = 1.0d0
9  real(8), parameter :: delta_t = 0.0001d0
10 integer, parameter :: steps = 100000
11 real(8), parameter :: radio_maximo = 10.0d0 ! Radio de la circunferencia
12 !real(8), parameter :: velocidad_magnitud = 10.0!1d0! Magnitud de la velocidad
13 ! Variables
14 real(8), dimension(N,3):: P, V, A

```

```

15 real(8), dimension(N) :: M, velocidad_magnitud
16 real(8), dimension(N):: ang_rand, radio
17 character(len=50) :: filename, filename2
18 integer :: i, ii, archivo
19 real(8) :: angulo, start_time, end_time
20 !integer :: num_threads
21 ! Abriendo Archivos
22 filename2="condiciones_iniciales.dat"
23 Do archivo=1, N
24 write(filename, '(A, I0, A)') 'Planeta', archivo, '.dat'
25 open(archivo, file=filename, status='unknown', position='append')
26 end do
27 open(unit=100, file=filename2, status='unknown', position='append')
28 ! Configurar threads
29 !call omp_set_num_threads(4)
30
31 !      Inicialización
32 ! ***** Condiciones iniciales *****
33 !Masas
34 call random_number(M)
35 M=M*100.0
36 M(1)=10000
37 P(1,:)= [0.0, 0.0, 0.0]
38 V(1,:)= [0.0, 0.0, 0.0]
39 1. POSICIONES: Distribución circular en el plano XY
40     !Posiciones aleatorias dentro de un círculo
41     !Generando angulos aleatorios
42     call random_number(ang_rand)
43     ang_rand = 2.0d0 * 3.141592653589793d0 * ang_rand
44     !Genera radios aleatorios
45     call random_number(radio)
46     radio= radio_maximo * sqrt(radio)
47     call random_number(P)
48 do i = 1, N
49     !Calculando coordenadas
50     !P(i,1) = radio(i)*cos(ang_rand(i))
51     !P(i,2) = radio(i)*sin(ang_rand(i))
52     !P(i,3) = 0.0d0
53
54     angulo = 2.0d0 * 3.141592653589793d0 * (i-1) / real(N,8)

```

```

55      !call random_number(P)
56      !P=P*10.0
57      P(i,1) = radio(i) * cos(angulo)  ! Coordenada X
58      P(i,2) = radio(i) * sin(angulo)  ! Coordenada Y
59      P(i,3) = 0.0d0                    ! Coordenada Z (plano)
60      !distancia al origen
61      ! dis_o(i) = sqrt(P(i,1)**2 + P(i,2)**2 + P(i,3)**2)
62  end do
63  ! 2. VELOCIDADES: Tangentes a la circunferencia (perpendiculares al radio)
64  call random_number(velocidad_magnitud)
65  velocidad_magnitud=velocidad_magnitud*10.0
66  do i = 1, N
67      angulo = 2.0d0 * 3.141592653589793d0 * (i-1) / real(N,8)
68      V(i,1) = -velocidad_magnitud(i) * sin(angulo)  ! Perpendicular al radio
69      V(i,2) = velocidad_magnitud(i) * cos(angulo)  ! Perpendicular al radio
70      !V(i,1) = velocidad_magnitud * cos(angulo)  ! Paralelo al radio
71      ! V(i,2) = velocidad_magnitud * sin(angulo)  ! Paralelo al radio
72      V(i,3) = 0.0d0                                ! Componente Z
73  end do
74  ! Inicializar aceleraciones
75  P(1,:)= [0.0, 0.0, 0.0]
76  V(1,:)= [0.0, 0.0, 0.0]
77  A(:,:) = 0.0d0
78
79  ! Escribiendo condiciones iniciales
80  Do archivo=1, N
81  write(archivo,*) P(Archivo,:)
82  end do
83
84  ! guardando en otro archivo condiciones iniciales
85  do i = 1, N
86      !write(100,*), "Planeta", i, ": Pos =", P(i,:), "Vel =", V(i,:)
87      write(100,*) "Condiciones Iniciales"
88      write(100,*), "Planeta", i
89      write(100,*) "P =", P(i,:)
90      write(100,*) "V =", V(i,:)
91      write(100,*) "M =", M(i)
92
93  end do
94

```

```

95  ! BUCLE PRINCIPAL PARALELIZADO
96  start_time = omp_get_wtime()
97
98  do ii = 1, ((steps-1)/10)
99      ! ESTA LLAMADA ES PARALELA INTERNAMENTE
100     call evolucion_paralela(P, V, M, G, N, delta_t)
101     P(1,:)= [0.0, 0.0, 0.0]
102     V(1,:)= [0.0, 0.0, 0.0]
103
104
105     if (mod(ii, 20) == 0) then
106         print *, 'Paso:', ii, '/', steps
107
108         ! Escritura paralela de archivos
109         !$OMP PARALLEL DO PRIVATE(archivo) SHARED(P)
110         do archivo = 1, N
111             write(archivo, *) P(archivo,:)
112         end do
113         !$OMP END PARALLEL DO
114     end if
115 end do
116
117 end_time = omp_get_wtime()
118 print *, 'Tiempo total:', end_time - start_time, 'segundos'
119
120 Do archivo=1, N
121 close(Archivo)
122 end do
123 end program Sistema_solar

```

Apéndice B

Códigos Scripts

Script 1

```
1  #!/bin/bash
2  *****
3  # Script: Montecarlo.sh
4  # Descripción: Ejecuta n veces el programa principal,
5  # donde n es el número de simulaciones a realizar.
6  # Guarda los resultados de cada simulación en un
7  # directorio único con el nombre simulacióni,
8  # donde i es una variable que varía según el número
9  # de simulación ejecutada.
10 *****
11
12
13 num_simulaciones=100
14
15 #Antes de ejecutar el script compilar el programa principal
16 #gfortran -fopenmp Sistema_solaral.f90 -o sistema_s
17
18 programa="./sistema_s"
19
20 for i in $(seq 1 $num_simulaciones); do
21     echo "Ejecutando simulación $i"
22
23     #Crear directorio donde se guardaran los datos
24
```

```

25 mkdir -p "simulación2$i"
26
27 #Ejecutar el programa
28 $programa
29
30 #_-----
31 #cd simulación$i
32 #touch exito.txt ##Crea un directorio vacio
33 #cd ..
34 #_-----
35
36 mv *.dat simulación$i/
37
38
39 done

```

Script 2

```

1  #!/bin/bash
2  #*****
3  # Script: graficas_s.sh
4  # Descripción: Copia los programas en octave
5  # en cada directorio Simulacióni. Entra en
6  # cada uno de ellos, hace una modificación
7  # en los archivos de octave ejecuta el
8  # graficador 1, y guarda la imagen generada
9  # en el directorio resultados.
10 #*****
11
12 num_simulaciones=100
13
14 programa="./sistema_s"
15
16 for i in $(seq 1 $num_simulaciones); do
17 echo "Ejecutando simulación $i"
18
19 #Crear directorio donde se guardaran los datos
20
21 mkdir -p "simulación2$i"

```

```

22
23 #Ejecutar el programa
24 $programa
25
26 #_-----
27 #cd simulación$i
28 #touch exito.txt ##Crea un directorio vacio
29 #cd ..
30 #_-----
31
32 mv *.dat simulación$i/
33
34
35 done

```

Graficador 1 en octave

Nombre: graficasnbody.m Función: Este programa lee las bases de datos de cada cuerpo, gráfica estos datos en el plano cartesiano, guardando la gráfica como una imagen, Con una escala de $[-15, 15]$ para ambos ejes.

```

1 % Graficador de trayectorias N-Body en segundo plano
2 clear all; close all; clc;
3
4 % Configuración
5 num_puntos_graficar = 1000; % Número definido de puntos a graficar
6
7 % Detectar automáticamente cuántos archivos de planetas existen
8 archivos_planetas = dir('Planeta*.dat');
9 num_archivos = length(archivos_planetas);
10
11 if num_archivos == 0
12     error(' No se encontraron archivos Planeta*.dat en el directorio actual');
13 end
14
15 fprintf(' Se encontraron %d archivos de planetas\n', num_archivos);
16
17 % Definir colores distintos para cada cuerpo
18 colores = lines(num_archivos);

```

```

19
20 % Leer todos los datos primero
21 datos_completos = cell(num_archivos, 1);
22
23 fprintf('Leyendo archivos...\n');
24 for i = 1:num_archivos
25     filename = archivos_planetas(i).name;
26     try
27         datos_completos{i} = load(filename);
28         fprintf(' Archivo %s cargado: %d puntos\n', filename,...
29                 size(datos_completos{i}, 1));
30     catch error
31         fprintf(' Error cargando %s: %s\n', filename, error.message);
32     end
33 end
34
35 % Crear figura EN SEGUNDO PLANO (sin mostrarse)
36 fig = figure('Position', [100, 100, 1200, 800], 'Visible', 'off');
37 hold on;
38 grid on;
39 title(sprintf('Trayectorias - %d Planetas', num_archivos));
40 xlabel('Posición X');
41 ylabel('Posición Y');
42
43 % Graficar cada planeta con el número definido de puntos
44 for i = 1:num_archivos
45     if ~isempty(datos_completos{i})
46         datos = datos_completos{i};
47         % Tomar solo el número definido de puntos
48         puntos_usar = min(num_puntos_graficar, size(datos, 1));
49         x = datos(1:puntos_usar, 1);
50         y = datos(1:puntos_usar, 2);
51
52         plot(x, y, 'LineWidth', 1.5, 'Color', colores(i, :), ...
53              'DisplayName', sprintf('Planeta %d', i));
54     end
55 end
56
57 legend('show', 'Location', 'bestoutside');
58 axis equal;

```

```

59 box on;
60
61 % ESTABLECER ESCALA FIJA DE -15 A 15 EN AMBOS EJES
62 xlim([-15 15]);
63 ylim([-15 15]);
64
65 % Guardar la gráfica SIN MOSTRAR LA FIGURA
66 nombre_archivo = 'simulacion1.png';
67 fprintf(' Guardando gráfica como %s...\n', nombre_archivo);
68 print(fig, nombre_archivo, '-dpng', '-r300');
69
70 % Cerrar la figura para liberar memoria
71 close(fig);
72
73 fprintf('\n Gráfica guardada exitosamente como: %s\n', nombre_archivo);
74 fprintf(' Resumen: %d planetas graficados con %d puntos cada uno\n', ...
75         num_archivos, num_puntos_graficar);
76 fprintf(' Escala fija: X [-15, 15], Y [-15, 15]\n');
77 fprintf(' Proceso completado en segundo plano\n');

```

Graficador 2 en octave

Nombre: graficasnbody2.m Función: Este programa lee las bases de datos de cada cuerpo, gráfica estos datos en el plano cartesiano.

```

1 % Graficador de trayectorias N-Body con pausa
2 clear all; close all; clc;
3 % Configuración
4 num_archivos = 10; % Cambia según tus archivos
5 colores = hsv(num_archivos); % Genera colores diferentes para cada cuerpo
6 %colores = repmat([1, 0, 0], num_archivos, 1); % [R, G, B] = [1, 0, 0]
7 % Crear figura
8 figure('Position', [100, 100, 1200, 800]);
9 hold on;
10 grid on;
11 title('Trayectorias de todos los cuerpos');
12 xlabel('Posición X (primera columna)');
13 ylabel('Posición Y (segunda columna)');
14 set(gcf, 'Name', 'Simulación N-Body'); % Título de ventana

```

```

15 % Definir colores distintos para cada cuerpo
16 %colores = [
17 %     1.0 0.0 0.0; % Rojo - Cuerpo 1
18 %     0.0 0.5 0.0; % Verde - Cuerpo 2
19 %     0.0 0.0 1.0; % Azul - Cuerpo 3
20 %     1.0 0.5 0.0 % Naranja - Cuerpo 4
21 %     1.0 0.5 0.0 % Naranja - Cuerpo 5
22 %
23 %];
24
25
26 % Leer y graficar cada archivo
27 for i = 1:num_archivos
28     filename = sprintf('Planeta%d.dat', i);
29     % Verificar si el archivo existe
30     if ~exist(filename, 'file')
31         fprintf(' Archivo %s no encontrado\n', filename);
32         continue;
33     end
34     % Cargar datos
35     try
36         datos = load(filename);
37         fprintf(' Archivo %s cargado: %d puntos\n', filename, size(datos, 1));
38         % Extraer columnas X e Y
39         x = datos(:, 1); % Primera columna: X
40         y = datos(:, 2); % Segunda columna: Y
41         % Graficar con color único para cada cuerpo
42         plot(x, y, 'LineWidth', 1.5, 'Color', colores(i, :), ...
43             'DisplayName', sprintf('Cuerpo %d', i));
44     catch error
45         fprintf(' Error cargando %s: %s\n', filename, error.message);
46     end
47 end
48
49 % Ajustar gráfica
50 legend('show', 'Location', 'bestoutside');
51 axis equal;
52 box on;
53
54 % ===== AQUÍ VA EL CÓDIGO DE GUARDADO =====

```

```

55 % GUARDAR LA GRÁFICA COMO PNG
56 nombre_archivo_png = 'simulacion1.png';
57 fprintf('\nGuardando gráfica como %s...\n', nombre_archivo_png);
58
59 % Guardar con alta resolución (300 DPI)
60 print(nombre_archivo_png, '-dpng', '-r300');
61 % Alternativa: saveas(gcf, nombre_archivo_png, 'png');
62
63 %fprintf(' Gráfica guardada exitosamente como %s\n', nombre_archivo_png);
64 %fprintf(' Archivo guardado en: %s/%s\n', pwd, nombre_archivo_png);
65
66 % Mantener abierto para visualización
67 %fprintf('\nLa ventana permanecerá abierta para visualización...\n');
68 %fprintf('Puedes cerrarla cuando desees.\n\n');
69
70 try
71     while ishandle(gcf)
72         pause(1);
73     end
74     fprintf('Ventana cerrada. Programa terminado.\n');
75 catch
76     fprintf('Programa interrumpido.\n');
77 end

```

Bibliografía

- [1] Benjamin Badger. *threebody: N-body gravitational system simulator*. <https://github.com/blbadger/threebody>. n.d.
- [2] Paul L. DeVries. *A First Course in Computational Physics*. John Wiley & Sons, 1994.
- [3] E. Díaz-Herrera, L. O. Pimentel y T. Roque Ramírez. “Estudiando trayectorias para ir a la luna: solución numérica del problema de Kepler”. En: *Latin American Journal of Physics Education* (2022).
- [4] Frida Gleisner. “Three solutions to the two body problem”. Bachelor’s thesis. Linnaeus University, 2013.
- [5] Stephen Hawking. *A hombros de gigantes*. Crítica, 2004.
- [6] Francisco López García. *Apuntes Mecánica Celeste Clásica*. Universidad Nacional de San Juan, 2018.
- [7] Kjartan Karígardarsson Mýrdal. “Some theoretical and numerical aspects of the n-body problem”. Master’s thesis. Lund University, 2013.
- [8] Shoichiro Nakamura. *Métodos Numéricos Aplicados con Software*. Pearson Educación, 1992.
- [9] Carlos M. Naón, Raúl D. Rossignolli y Eve M. Santangelo. *Ecuaciones Diferenciales en Física*. Universidad Nacional de La Plata, 2014.
- [10] Bravo Silvia. *Encuentro con una estrella*. Fondo de Cultura Económica, 2012.
- [11] Dennis G. Zill, Raúl D. Rossignolli y Eve M. Santangelo. *Ecuaciones Diferenciales con aplicaciones de modelado*. Cengage Learning, 2014.