Informe Examen 1: Astrofísica Computacional

Estudiante: Bryan Camilo Restrepo Arcila

Aclaración inicial para una correcta ejecución

- Para una ejecución correcta de los comandos debes de estar parado en tu consola en la carpeta donde se encuentran los archivos con el codigo fuente

Punto 1: Epsilon de la maquina de ejecución

El algoritmo comienza con un valor de epsilon igual a 1.0. Luego, entra en un bucle que reduce epsilon a la mitad en cada iteración. En cada paso, el algoritmo verifica si el valor actual de epsilon, cuando se suma a 1.0, resulta en un valor diferente a 1.0. Si al sumar la mitad del epsilon actual a 1.0 el resultado sigue siendo 1.0, significa que epsilon aún no es lo suficientemente pequeño como para ser considerado el épsilon de la máquina. El bucle continúa hasta que se encuentra el valor más pequeño de epsilon que, al sumarse a 1.0, produce un valor detectable como diferente de 1.0.

El valor de epsilon encontrado para mi maquina fue de: 2.22045×10^{-16}

Para ejecutar el codigo fuente, utiliza el siguiente comando:

```
gcc 1_epsilon_de_mi_maquina.c -o 1_epsilon_de_mi_maquina_ejecutable.out ;
./1_epsilon_de_mi_maquina_ejecutable.out
```

Punto 2: Numero de elementos de la serie de leibniz necesarios para calcular π con 10 cifras significativas

La primera parte del programa calcula una aproximación de π. Este cálculo se basa en la serie de Leibniz, que es una serie infinita donde π se puede aproximar como 4 veces la suma de una serie de fracciones con términos alternantes positivos y negativos.

La lógica realiza los siguientes pasos:

- Inicia variables para la sumatoria, el valor real de π, el valor numérico de π dividido por 4 y el error.
- La serie alterna signos y divide 1 entre los números impares sucesivos (1, 3, 5, …).
- Suma estos términos hasta que el error absoluto en la aproximación de π es menor a `1e-10`.
- Al final del bucle, imprime la cantidad de términos necesarios para alcanzar la precisión deseada.

El numero de terminos necesarios para conseguir el valor de pi con un error menor a `1e-10` es: 1581043254

Según el punto anterior: La precisión que puede conseguir mi computador es: 2.22045×10^{-16}

El número máximo de cifras significativas con las que mi computador puede calcular pi es 16

Para ejecutar el codigo fuente, utiliza el siguiente comando:

```
gcc 2_calculo_de_pi.c -o 2_calculo_de_pi -lm ; ./2_calculo_de_pi_ejecutable.out
```

Punto 3: Funcion de Bessel de orden 0, 1 y 2

Este programa calcula las funciones de Bessel de primera especie de orden alpha (J_α) para un rango de valores x.

El cálculo se realiza mediante la fórmula de la serie de Taylor de las funciones de Bessel, sumando términos hasta el 150, para asegurar la convergencia.

Se generan archivos de datos para los órdenes alpha de 0 a 2, en los cuales se almacena el valor de x y el correspondiente valor de J_α.

El gráfico obtenido para las funciones de bessel de orden 0, 1 y 2, es el siguiente:

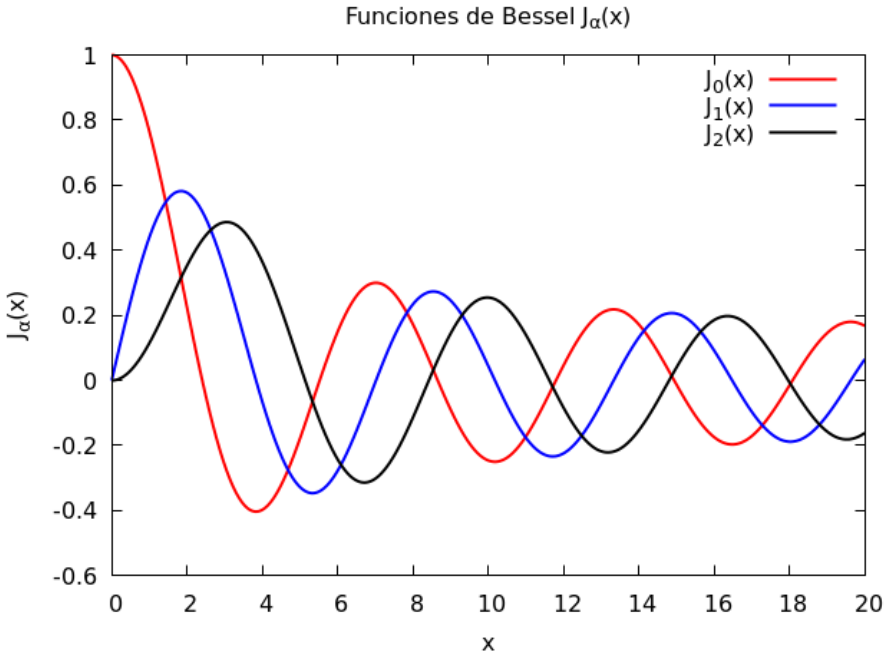

Figure 1: Funciones de Bessel

En el gráfico se puede observar una convergencia muy buena (Respecto al comportamiento esperado para esta función) y rapida a pesar es estar trabajando

con factores factoriales que tienen a generar un desbordamiento rapido de los valores que puede procesar una computadora convencional.

Para ejecutar el codigo fuente, y obtener el conjunto de datos para cada orden de las funciones de Bessel utiliza el siguiente comando:

```
gcc 3_funcion_de_bessel.c -o 3_funcion_de_bessel_ejecutable.out -lm ;
./3_funcion_de_bessel_ejecutable.out
```

Para la ejecución del script de graficación, y obtener el gráfico según los datos obtenidos de la ejecución del codigo fuente, puedes usar el siguiente comando:

```
gnuplot 3_script_graficacion_funcion_bessel.gp
```

Punto 4: Integración con Leap-Frog para un movimiento parabólico

En este punto se simula el movimiento de un objeto en caída libre y calcula el error en la energía del sistema usando dos métodos de integración numérica: el método de Euler y el método Leap Frog. El objetivo es evaluar y comparar la precisión de estos métodos para modelar sistemas físicos.

Para ejecutar el codigo fuente, y obtener el conjunto de datos para la graficación:

```
gcc 4_integracion_lead_frog.c  -o 4_integracion_lead_frog_ejecutable.out -lm ;
./4_integracion_lead_frog_ejecutable.out
```

Para el sistema se hizo un analisis del movimiento con diferente cantidad de puntos con ambos métodos de integración, gráficamente esto fue lo que se obtuvo:

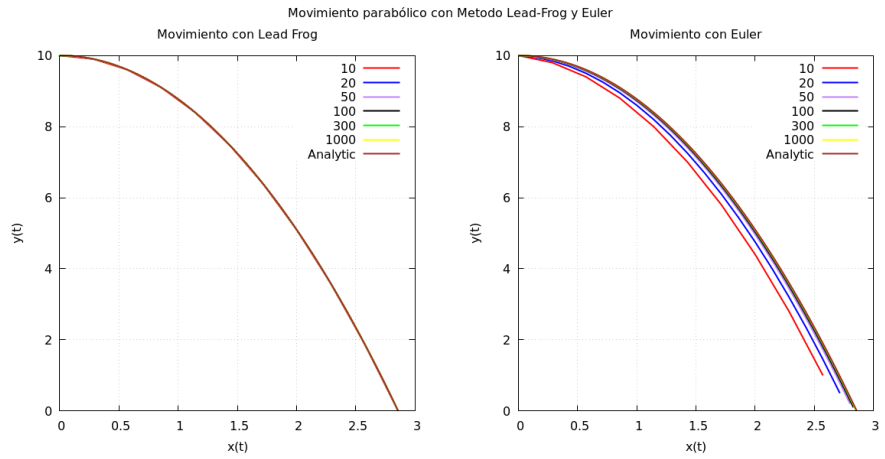

Figure 2: Movimiento con Leap-Frog y Euler

Como se observa en la imagen, el método de Leap-Frog es superior al de Euler en terminos de precisión, ya que, incluso con apenas 10 puntos se consigue la

aproximación al movimiento analítico es bastante buena a diferencia que con el método de Euler que para una aproximación buena necesita alrededor de unos 100 puntos de integración.

Para la ejecución del script de graficación, y obtener el gráfico según los datos obtenidos de la ejecución del codigo fuente, puedes usar el siguiente comando:

```
gnuplot 4_script_graficacion_movimiento.gp
```

Ahora haciendo un analisis de la conservacion de la energía por ambos métodos de integración para una diferente cantidad de puntos, podemos obtener la siguiente gráfica:

Se puede observar en el gráfico que si bien el método de euler va obteniendo errores en la energía con respecto a la energía inicial cada vez más pequeños, el método de Leap-Frog consigue una conservación de la energía sin importar el número de puntos, así que el método de Euler también consigue mejores resultados en terminos de la conservación de la energía.

El factor en el que el método de Euler es mejor que el de Leap-Frog es en el hecho de que no tiene que calcular una velocidad intermedia para hallar la actualización en las posiciones por cada iteración, y esto hace para una misma cantidad de puntos el Método de Euler tiene un menor tiempo de ejecución aunque tenga una menor precisión.

Para la ejecución del script de graficación, y obtener el gráfico según los datos obtenidos de la ejecución del codigo fuente, puedes usar el siguiente comando:

```
gnuplot 4_script_graficacion_errores_energia.gp
```

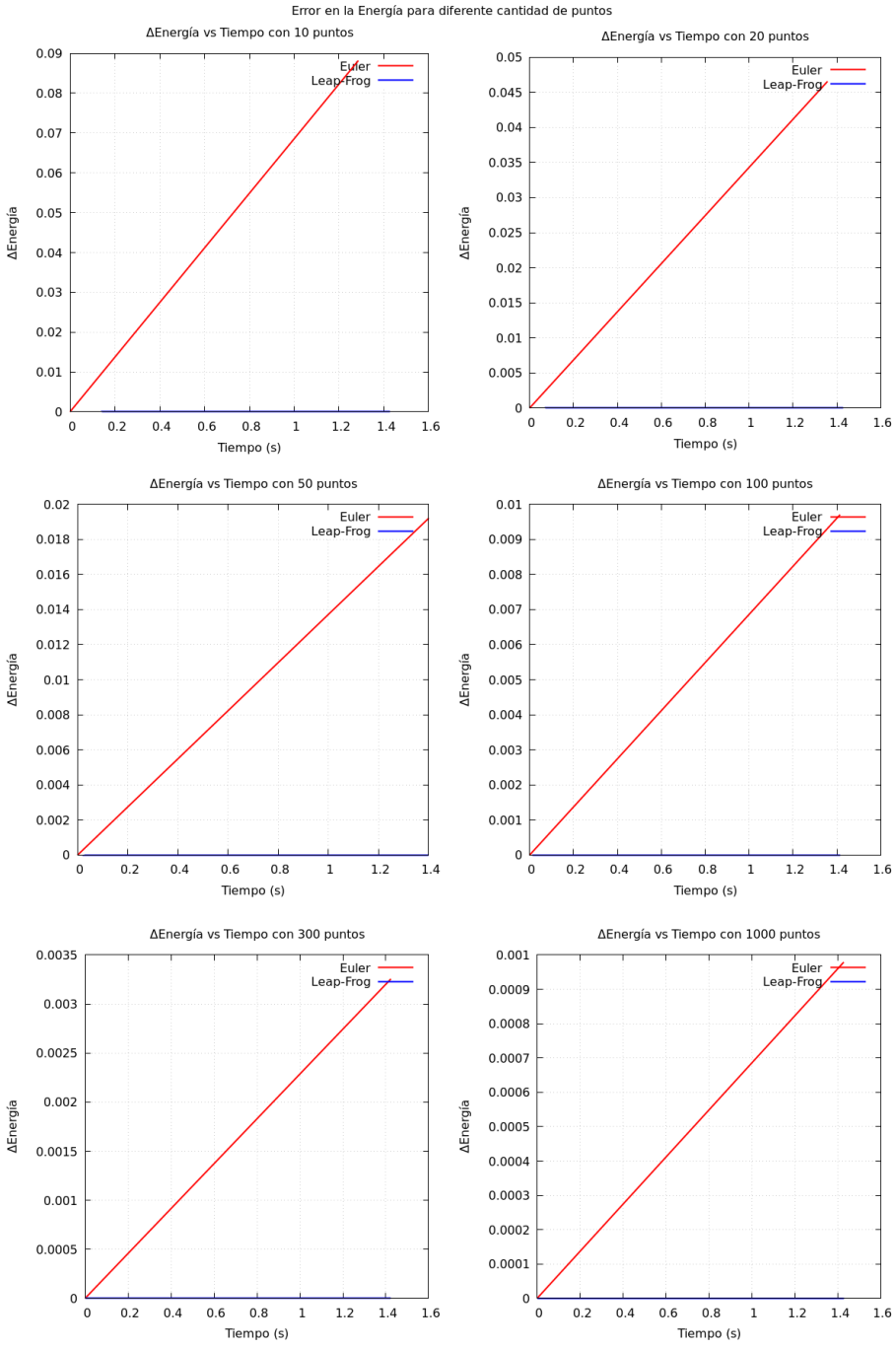

Figure 3: Error en la energía con Leap-Frog y Euler