Tarea 1 Métodos Numéricos para la Ciencia y la Ingeniería

Javier Silva Lafaurie

Septiembre 2015

1. Introducción

Se busca analizar un conjunto de datos del espectro del Sol, en este caso su flujo en función de la longitud de onda con el objetivo de obtener un gráfico de este. Además se desea integrar numéricamente el espectro como también la solución teórica del cuerpo negro para una temperatura del 5778 K implementando el método del trapecio. Con lo anterior se determina el radio efectivo del Sol. Finalmente se vuelven a integrar el espectro como el cuerpo negro, esta vez con rutinas ya determinadas en Python y se compara la velocidad de ejecución de estas con los métodos implementados.

2. Procedimiento

Para graficar el espectro del Sol con los datos otorgados se procede a leer estos importando la librería *numpy* de Python a través del comando *numpy.loadtxt* como procede:

```
datos=numpy.loadtxt('sun_AM0.dat')
long_onda=datos[:,0]
flujo=datos[:,1]
```

Una vez indentificadas las columnas de la longitud de onda y del flujo se guardan en arreglos, como ya se hizo, para luego transformar a las unidades convencionales en Astronomía (Esto es del sistema MKS al CGS para el flujo y a micrones para la longitud de onda). Para esto usamos la libreria *astropy* e importamos *constants* y *units*.

Finalmente para graficar estos datos de procede a importar la libreria *matplotlib.pyplot*, con la cual podemos agregar ejes, título, rango de ejes, etc.

Para integrar estos datos se procede a usar el método del trapecio.

2.1. Método del trapecio

Con este algoritmo se aproxima el área bajo la curva o integral a través de áreas de pequeños trapecios de vertices x_i, x_{i+1}, $f(x_i)$ y $f(x_{i+1})$ donde x es un arreglo de la variable independiente y $f(x)$ es un arreglo de la variable dependiente. En el caso del espectro del Sol los x vendrían a ser las longitudes de onda y los $f(x)$ corresponden a los flujos. Luego se procede a calcular el área de estos trapecios recordando que es igual al promedio de las bases por la altura:

$$Area = \frac{f(x_i)+f(x_{i+1})}{2}(x_{i+1}-x_i)$$

En el caso del flujo del Sol se programa la siguiente rutina en Python:

```
Integral=0
for i in range(0,len(flujo)-2):
    Integral=Integral+(flujo[i+1]+flujo[i])*(long_onda[i+1]-long_onda[i])/2
```

Al reescalar el cuerpo negro e integrar el área bajo esta aparece la integral:

$$I = \int_0^\infty \frac{x^3}{e^x-1}dx$$

Lo cual el problemático pues no se puede discretizar entre 0 e infinito, por lo que se toma el cambio de variable:

$$y = arctan(x)$$
$$x = tan(y)$$
$$dx = \frac{dy}{cos^2(y)}$$

Con lo cual nuestra integral nos queda:

$$I = \int_0^{\pi/2} \frac{tan^3(y)}{cos^2(y)(e^{tan(y)}-1)} dy$$

El problema ahora es que la función dentro de la integral se indetermina en 0 y en $\pi/2$, sin embargo tomando el límite por la derecha para el primero y por la izquierda para el segundo, vemos que estos límites existen y son iguales a cero. Para solucionar este problema empezaremos la discretización en un valor mayor a 0 y la terminaremos en un valor menor a $\pi/2$ este aumento y disminución deben ser ínfimas con el fin de no afectar el valor de la integral. Una vez planteado en esto podemos ver que se cumple en la siguiente rutina:

```
a=0.05 #partimos en 0.05 y no en 0
b=m.pi/2-0.05 #temrinamos en pi/2 - 0.05 y no en pi/2
n=2000 #precisión del vector
dx=(b-a)/n #discretización
x=np.linspace(a,b,n) #vector equispaciado
x=np.array(x)
y=fint(x) #función de la integral
integral=0
for i in range(0,len(y)-1):
    integral=integral+(y[i+1]+y[i])*dx/2 #integramos usando trapecios
```

Se comparan las dos integrales anteriores (escaladas a las constantes y en los valores de unidades CGS) para conocer el radio efectivo del Sol. Sea $I1$ la integral del espectro de los datos del Sol e $I2$ la integral del cuerpo negro del Sol, entonces se tiene la siguiente relación:

$$4\pi R^2 = I1/I2$$
$$\Rightarrow R = (\frac{I1/I2}{4\pi})^{1/2}$$

Finalmente se implementan los métodos de integración ya definidos en Python y se procede a comparar los valores de estos con los obtenidos con las rutinas ya definidas. Además se toman los tiempos de cada algoritmo para ver la eficiencia de estos.

3. Resultados

La lectura de datos y consiguiente gráfico se puede apreciar en la figura 1

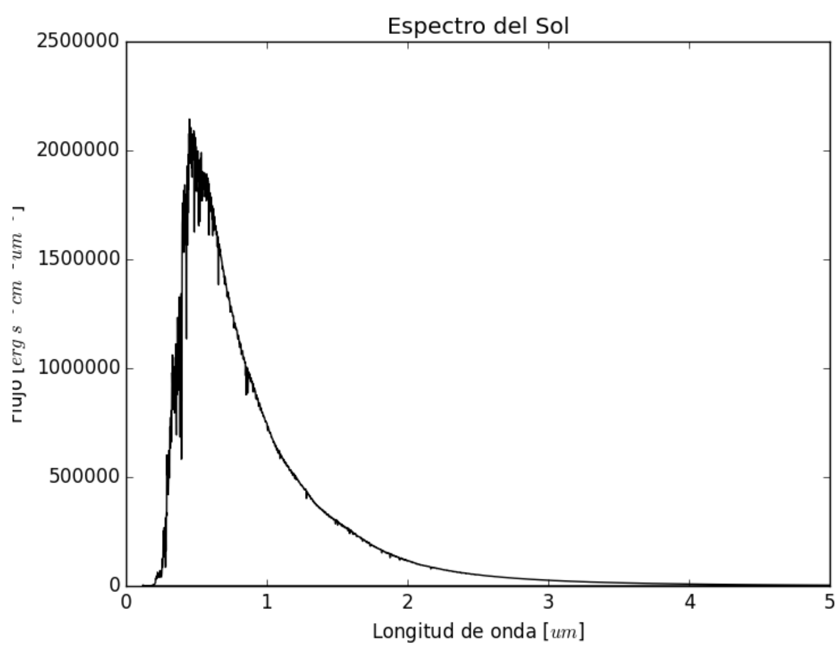

Figura 1. Espectro del Sol con los datos del archivo sun_AM0.dat

Los valores obtenidos tanto con las rutinas implementadas como con las librerias de Python se muestran en la tabla 1.

Integral	Implementación [erg s-1 cm-2]	rutina Python [erg s-1 cm-2]
Espectro del Sol	1366090,76552	1366090,7968
Cuerpo Negro	63168490023,4	63200090068,4

Tabla 1. Comparación precisión métodos programados versus rutinas de Python

De igual forma se procede a mostrar los diferentes tiempos de ejecución para estos algoritmos en la tabla 2.

Integral	Tiempo en Implementación [s]	Tiempo en Rutina Python [s]
Espectro del Sol	0,012420091136712	0,00130105018616
Cuerpo Negro	0,00457501411438	0,000392913818359

Tabla 1. Comparación eficiencia de tiempo métodos programados versus rutinas de Python

Ahora que obtuvimos las integrales podemos calcular el radio efectivo usando la ecuación de la sección de procedimiento, con lo que obtenemos un valor de 0,00131 UA.

4. Conclusiones

Dentro de lo obtenido anteriormente hacemos notar que tanto el gráfico como las rutinas obtenidas y junto a los datos calculados hacen de Python una plataforma ideal para muchas aplicaciones dentro del campo científico.

Para el caso de la integración por implementación versus las rutinas predefinidas en Python es posible notar que no presentan mayores diferencias en el valor de la integral. En el caso del espectro del Sol recien en el tercer decimal hay una diferencia entre los dos valores numéricos. Lo mismo para el Cuerpo Negro donde la primera diferencia es dos ordenes de magnitud más pequeños.

Para el tiempo de ejecución vemos una clara evidencia y es que los algoritmos de las librerias de Python son más rápidos en comparación a los implementados.

Finalmente se hace notar que el radio efectivo del Sol es unas 10 veces su radio aproximadamente.