Informe Tarea 1: Métodos Numéricos

Ignacio Andrés Sánchez Barraza

September 20, 2015

1 Pregunta 1

1.1 Introducción

- Se busca graficar a partir del archivo $sun\_AM0.dat$, el espectro del sol (en unidades de energía por unidad de aréa, de tiempo y de longitud de onda) vs. longitud de onda, es decir el flujo por unidad de longitud de onda vs. longitud de onda (todo en el sistema CGS y para longitudes de onda en μm)

1.2 Procedimiento y resultados

- Aquí dado el archivo $sun\_AM0.dat$, se procedió a leer el contenido del archivo el comando $np.loadtxt('sun\_AM0.dat')$ en python y guardar lo leído en la variable $datos$, siendo este último una lista múltiple de la forma $datos = [[londa_0, fsol_0],$
$[londa_1, fsol_1], ...]$, desde donde se obtendrían una lista o arreglo para la longitud de onda y el flujo. Terminado el arreglo para cada variable ($fsol$ y $londa$), como se pedia para el flujo, unidades del sistema CGS, y para la longitud de onda, unidades de μm, se hizo la conversión de unidades apropiada quedando el flujo en unidades de $\frac{erg}{s\ cm^2\ \mu m}$ y la longitud de onda en μm. Terminado esto se procedió a realizar el gráfico con cada variable con el comando $plt.plot(londa, fsol, 'r-')$ que realizaría el gráfico flujo vs long. de onda, obteniéndose el siguiente gráfico:

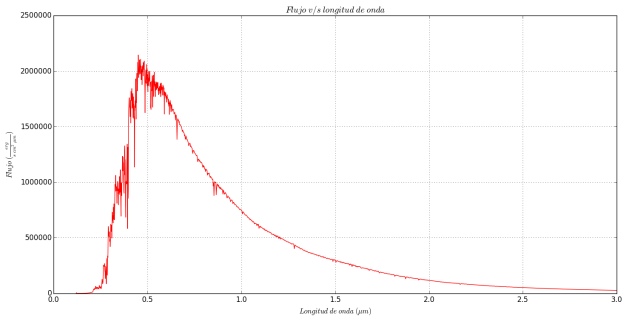

Figure 1: Gráfico flujo vs longitud de onda.

2 Pregunta 2

2.1 Introducción

- Se pide elegir un método apropiado para integrar numéricamente el espectro, o flujo por unidad de longitud de onda, en longitud de onda, es decir obtener la energía por unidad de área y de tiempo, para luego a partir de ello calcular la luminosidad del sol (energía por unidad de tiempo)

2.2 Procedimiento

- Para esto se procedió de dos formas, la primera con un método de integración numérica compuesto por el método de simpson y de trapezoides, y el segundo por un método netamente de trapezoides.

Para el método compuesto se procedió a hacer una segmentación de los datos de acuerdo a sus "coeficiente de paso" o diferencia entre los distintos valores de la variable λ (longitud de onda), formando así segmentos con un número de intervalos pares, dado que así lo exige la regla de Simpson, por lo que se arreglaron los segmentos de tal forma que hubieran la mayor cantidad de segmentos que cumplieran esa regla. Pero como no todos podían cumplirla, dichos segmentos eran integrados usando la regla de los trapezoides, asi como las uniones entre los segmentos que permitían el uso de la regla de Simpson, puesto que al ser intervalos simples (solo un intervalo) no se pueden integrar con dicha regla. Dicho esto, se procedió a integrar numéricamente por ambos métodos y a sumar los resultados de ambos métodos, abarcando al final, una integral sobre toda la variable λ.

- Para el método de trapezoides se definió una función $trap()$ que para cada valor de j en un rango de 0 a 1696 (ie, 1697 índices), asignaba un valor a $x_f = j + 1$ y a $x_0 = j$, con lo que tomaba intervalos individuales para asi aplicar la integración numérica dpor trapezoides tradicional. Este método consiste en multiplicar la suma de la función evaluada en x_0, en x_f y en valores intermedios (en este caso ninguno, dado que no hay datos intermedios) por la mitad del "paso" o diferencia entre x_f y x_0, es decir en el código descrito como $\frac{londa[x_f]-londa[x_0]}{2}$. Así para cada j en el rango mencionado anteriormente y sumando cada una de esas multiplicaciones se obtiene la integral numérica pedida.

2.3 Resultados

- Como resultado de la integración compuesta entre la regla de Simpson y el métodos de los trapezoides se obtuvo que el valor para el flujo del sol a una distancia de 1 UA era $F_{sol} = 1370144.79919[\frac{erg}{scm^2}]$
- Como resultado de la integración numérica por el métodos de los trapezoides, se obtuvo un valor para el flujo del sol a una distancia de 1 UA de $F_{sol} = 1366090.79684[\frac{erg}{s\ cm^2}]$, y con ello como se tiene la expresión que relaciona la luminosidad con el flujo de una fuente: $F = \frac{L}{4\pi r^2}$ (con L: luminosidad y r: distancia a la fuente), se obtuvo que la luminosidad del sol es $L_{sol} = 3.87399315194e + 33[\frac{erg}{s}]$

3 Pregunta 3

3.1 Introducción

- En esta pregunta se pide integrar numéricamente la función de Planck:

$$B_\lambda(T) = \frac{2\pi hc^2/\lambda^5}{e^{hc/\lambda k_B T}-1}$$

(donde h es la constante de Planck, c es la velocidad de la luz en el vacio, k_B es la constante de Bolzmann, T es la temperatura del cuerpo negro y λ es la longitud de onda) para estimar la energía total por unidad de área emitida por un cuerpo negro con la temperatura efectiva del sol $T_{eff} = 5778[K]$, y compararla con la energía total calculada en la pregunta número 2 para estimar así el radio efectivo del sol, es decir, el radio al cual el sol emite como cuerpo negro con una temperatura igual a T_{eff}.

3.2 Procedimiento

- Como la radiacion de un cuerpo negro en unidades de energia por unidad de tiempo por unidad de area por unidad de longitud de onda esta dada por la funcion de Planck, la expresión a integrar numéricamente es:

$$I = \int_0^\infty B_\lambda(T) d\lambda$$

Pero el enunciado proporciona dicha integral escrita de otra forma como:

$$P = \frac{2\pi h}{c^2}\left(\frac{k_B T}{h}\right)^4 \int_0^\infty \frac{x^3}{e^x-1} dx$$

Entonces ahora lo que se debe hacer solamente es integrar numéricamente P y hacer las conversiones pertinentes de unidades para cada constante con el módulo *astropy.constansts* y redefinir cada constante como $A = constants.a.cgs$ para asi convertir sus unidades a el sistema CGS. Así es como ahora integrando P mediante el método de Simpson, se tiene que

$$P = \frac{(\frac{(x_f - x_0)}{n})}{3}(S(x_0) + 2\Sigma S(x_i) + 6\Sigma S(x_j) + S(x_f)$$

con $S = \frac{x^3}{e^x - 1}$, con $\Sigma S(x_i)$ la suma sobre índices pares, $\Sigma S(x_j)$ la suma sobre índices impares y con x_0 y x_f los extremos del intervalo de integración. Ahora bien como no se puede evaluar la función directamente en los extremos 0 e ∞, se usa el cambio de variable propuesto $y = \arctan(x) \rightarrow x = \tan(y) \rightarrow dx = \sec(y)^2 dy$ con y entre $[0, \frac{\pi}{2}]$. Con este cambio de variable la función S cambia a $S = \frac{\tan(y)^3}{e^{(\tan(y))} - 1}\sec(y)^2$ y ahora definiendo una función que realice lo descrito anteriormente se obtiene lo buscado, teniendo especial cuidado en el valor de $S(x_0)$ y $S(x_f)$, ie. en los límites, que para fines prácticos en el código se han colocado los límites numéricos para cada extremo, es decir, $S(x_0) = 6$ y $S(x_f) = 0$, obtenidos a través de la regla de L'Hopital.

3.3 Resultados

- A raíz de este proceder se obtiene que la integral de $\int_0^{\frac{\pi}{2}} S = \int_0 \frac{\pi}{2} \frac{\tan(y)^3}{e^{(\tan(y))} - 1}\sec(y)^2 \approx \frac{\pi^4}{15}$ como se dice en el enunciado sobre el resultado teórico de dicha integral, y finalmente se obtiene que $I = P = 63231254495.9[\frac{erg}{scm^2}]$. Ahora con este último resultado, se puede obtener el radio efectivo del sol a través de la relación:

$$F_{sol} = \frac{L_{sol}}{4\pi R_s ol^2}$$

con $R_s ol$: radio del sol, de donde se obtiene que $R_{sol_{eff}} = 69824621987.4[cm]$

4 Pregunta 4

4.1 Introducción

- En esta pregunta se pide comparar el algoritmos creado para poder resolver las preguntas 2 y 3 con los algoritmo propios del módulo scipy, *scipy.trapz* y *scipy.quad*, y estimar el tiempo de ejecución de todos los procesos mencionados y compararlos, intentando explicar el por qué de las diferencias.

4.2 Procedimiento

- Para lograr esto solo hace falta tomar una función que posee python para poder calcular el tiempo de ejecución de un proceso, en este caso $mgc = get\_ipython().magic$, el cual se usa de la forma $mgc('\%timeit f(x)')$ con $f(x)$ la función a la cual se quiere medir el tiempo de ejecución.

4.3 Resultados

- Usando lo dicho anteriormente se obtienen los siguientes resultados:
  Para el caso del algoritmo propio para la pregunta 2 (que en este caso fueron 2) el tiempo de ejecución para el método compuesto fue de $t = 60.8[ns]$ y para el método netamente de trapezoides fue de $t = 4[ms]$ y para la pregunta 3 el tiempo de ejecución del algoritmo propio fue de $t = 78.8[ns]$. Ahora para el caso del algoritmo *scipy.trapz* (pregunta 2) el tiempo de ejecución fue de $t = 78[ns]$ y para *sciipy.quad* (pregunta 3) fue de$t = 61.2[ns]$.

4.4 Conclusiones

- Del item anterior se ve claramente que el método *scipy.trapz* fue más rapido que el algoritmo propio de trapezoide (pero menor al compuesto, esto último debido a que correponde mas que nada a obtener los datos en intervalos específicos dados por el programador, y el programa no debe buscar los intervalos o reconocerlos por si mismo por lo que incurre en menos tiempo de ejecución), ya que éste estaba definido para intervalos individuales, recorriendo intervalo por intervalo, mientras que el del módulo de scipy probablemente lo hace de manera más generalizada evitando iteraciones innecesarias en el código. Ahora para el caso de la pregunta número 3 el tiempo de ejecución fue menor en el proceso *scipy.quad*, probablemente debido a iteraciones innecesarias o aproximaciones en los límites ímplicitas como el caso en que en el algoritmo propio se procedió a usar el cálculo diferencial, específicamente la regla de L'Hopital para obtenerlos, pero no para valores de la variable independiente cercanos a dichos límites.