Integración Del Trapecio

Tiffany López Nicholson
Miriam Martín Jacinto
Sergio Vega García

16 de mayo de 2013

Resumen

A continuación se presentará como se ha implementado con *Python* un algoritmo capaz de resolver la integral definida $\int_1^6 \frac{1}{1+e^x}$

Índice

1. Motivaciones y objetivos

1.1. Objetivo principal

Debido a que realizar la Regla del Trapecio manualmente es muy tedioso y se pierde mucho tiempo, se ha decidido implementar un programa en *Python* que realice la Regla del Trapecio para $f(x)$.

1.2. Objetivo específico

El algoritmo que se desarrollará a lo largo del proyecto será capaz de resolver la integración del trapecio para $f(x) = \frac{1}{1+e^x}$, x E [1, 6]. Esto significaría una mayor rapidez para la obtención de resultados al igual que una mayor presición.

2. Fundamentos teóricos

La regla del trapecio es un método de integración numérica que se basa en aproximar el valor de la integral definida de $f(x)$ por el de la función lineal que pasa a través de esta, formándose una figura: un trapecio. Para obtener esta aproximación, debemos calcular el área de los trapecios.

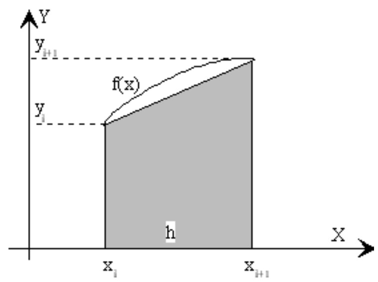


Figura 1: Ejemplo

Para justificar este método, deberemos "aproximar" de una buena manera nuestra función $f(x)$. Esto lo haremos gracias a la *interpolación polinomial.*

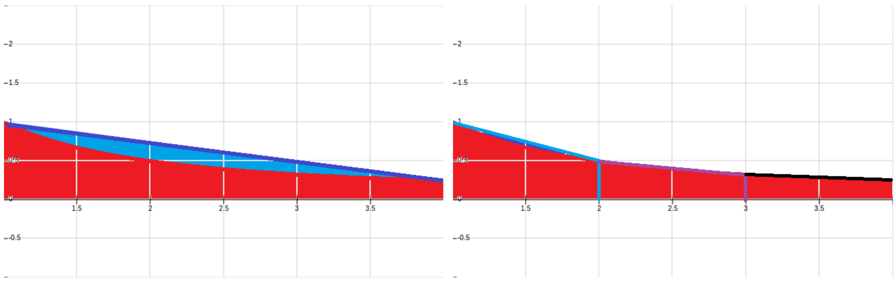
2.1. Interpolación polinomial

La interpolación polinomial nos dice que para hacer una "buena aproximación" de $f(x)$, que querremos integrar, por otra función $g(x)$, en los puntos x_i (con $i = 1, 2, 3, ..., n$); o lo que es lo mismo, $\int_{x_1}^{x_n} f(x)dx \approx \int_{x_1}^{x_n} g(x)dx$, $\forall x_i, coni = 1, 2, 3, ..., n$. Estas dos funciones deben ser continuas en el intervalo $[x_1, x_n]$.
Pero el problema que se presenta es como buscar estas funciones. Hay varios teoremas que nos ayudan a resolverlo, como el *teorema aproximación de Weierstrass*[1] o el *polinomio de interpolación de Lagrange*[2].

2.2. Integración del trapecio

Para la utilización del método del trapecio partimos de una función, la cual dividiremos en n trozos iguales. Cuanto mayor sea el número de particiones, mayor precisión tendrá el método.

[1]http://es.wikipedia.org/wiki/Teorema_de_aproximación_de_Weierstrass
[2]http://es.wikipedia.org/wiki/Interpolación_polinómica_de_Lagrange



Se puede apreciar que el área tomada por exceso (área azul), es decir, la que supera a la función, se reduce según aumenta el número de particiones.

La función general es:

$$\int_a^b f(x)dx \approx \frac{h}{2}[f(a) - 2f(a+h) + 2f(a+2h) + ... + f(b)]$$

Donde $h = \frac{b-a}{n}$ y n es el número de divisiones.

La expresión anterior también se puede escribir como:

$$\int_a^b f(x)dx \approx \frac{b-a}{n}(\frac{f(a)+f(b)}{2} + \sum_{k=1}^{n-1} f(a + k\frac{b-a}{n}))$$

2.3. Integración del trapecio aplicado

La ecuación específica del proyecto es $f(x)\frac{1}{1+e^x}$, cuya representación gráfica es la siguiente:

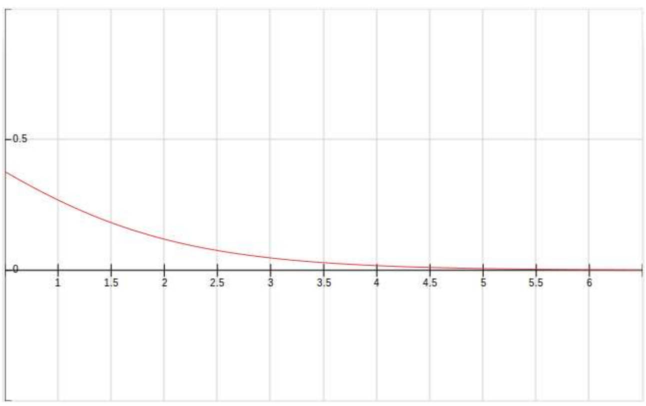


Figura 2: Prueba

El intervalo de la integral es [1,6]. Existe función en todos los puntos, por lo que también existe la integral. Al igual que en el caso anterior, por existir una curva, habría que realizar varias particiones para conseguir un resultado preciso.

3. Procedimiento experimental

Para poder realizar la integración del trapecio primero se necesita analizar los datos iniciales.

- La función $f(x) = \frac{1}{1+e^x}$ estaba definida en el intervalo [1,6].
- La función definida en [1,6] existe en $\mathbb{R}$, números reales, por tanto se puede integrar.
- En la fórmula de la integración del trapecio intervienen bastantes variables.

Con estos datos a priori se pueden sacar dos conclusiones:

1. Realizar la integración por la regla del trapecio manualmente es muy laborioso.
2. Se necesita implementar un programa en *Python* para agilizar la tarea.

Se ha creado en *Python* un programa que permita elegir el número de particiones y para así poder comparar resultados. La función principal es comprobar que cuantas más particiones se realicen, más precisa es la integración.

3.1. Descripción de los experimentos

Se ha implementado un programa en *Python* capaz de realizar la regla del trapecio, en el cual se puede elegir las particiones mínimas y máximas. Al ejecutarlo e introducir los datos se nos muestra los resultados de cada partición.

Se han comprobado algunas particiones manualmente para comprobar que realmente el programa está funcionando correctamente.

Se ha ejecutado dos veces el bucle con las siguientes caracteristicas:

1. De 1 a 10 particiones, con intervalo de 1 unidad.
2. De 10 a 100 particiones, con intervalo de 10 unidades.
3. De 100 a 1000 particiones, con intervalo de 100 unidades.

3.2. Descripción de las computadoras utilizadas

- S.O.: Linux - Ubuntu.
- Procesador: Pentium(R) Dual-Core CPU T4400 @ 2.20GHz.
- Velocidad: 1200.000 MHz.
- RAM: 4 GB
- Versión de *Python*: 2.7.3

3.3. Resultados obtenidos

Partición	Resultado	Partición	Resultado
1	0.6908982271	10	0.316070258216
2	0.418729690428	20	0.312415536652
3	0.360170903466	30	0.311647557367
4	0.339273533105	40	0.311347886591
5	0.329487501506	50	0.311195057792
6	0.324106360674	60	0.311104409514
7	0.320820024708	70	0.311045164508
8	0.31865911442	80	0.311003740508
9	0.317158073083	90	0.310973305356
10	0.316070258216	100	0.310950080829
Tiempo	0.000426054000854	Tiempo	0.00157999992371

Partición	Resultado
100	0.310950080829
200	0.310857929721
300	0.310831706878
400	0.310819438106
500	0.310812346491
600	0.310812346491
700	0.310804489423
800	0.310802088259
900	0.310800238521
1000	0.310798769966
Tiempo	0.0125889778137

3.4. Análisis de resultados

Analizando los resultados, se observa que no hay grandes cambios a partir de la octava división. Si solo se hicieran una o dos divisiones se tendría un resultado mucho mayor que el real, eso es porque hay una curva convexa pronunciada, sobre todo al principio. Esto se puede apreciar en la Figura 2. Si fuera una recta no habría variación en ninguna de las particiones.

Cuanto mayor es el número de particiones, más se reduce el error, por tanto, la curva que existe es decreciente.

El número de particiones influye en el tiempo que tarda en obtenerse el resultado, incluso en una computadora. Se puede observar como se eleva considerablemente cuanto mayor es el número de particiones.

4. Conclusiones

Después de pruebas y pruebas, se ha llegado a la conclusión de que el tipo de función determina la cantidad mínima de particiones que se deben hacer para conseguir una buena aproximación de la función.

El método del trapecio es muy útil cuando la integral es muy difícil, aunque su principal problema es el largo tiempo que toma para hacerlo. Sin embargo, mediante un programa en *Python* es todo mucho más sencillo.

Al utilizar un programa informático, el tiempo se reduce muchísimo, además, te permite realizar comparaciones entre varias particiones, ya que tendrías los resultados casi al instante.

Por otro lado, si la ecuación se puede integrar relativamente fácil sería mejor usar la tabla de las integrales inmediatas en vez de este método.

5. Apéndice

5.1. Algoritmo

NOMBRE: Trapecio.py

AUTORES:

- Tiffany López Nicholson
- Miriam Martín Jacinto
- Sergio Vega García

FECHA: 16 de mayo de 2013

DESCRIPCIÓN:

Se pide introducir por teclado la cantidad mínima de particiones, la cantidad máxima y la cantidad de intervalos.

Se asigna a una variable el tiempo actual.

Nos encontramos con dos bucles while anidados. En el primero, se asigna el valor de las particiones actuales; y en el segundo, se encuentra el sumatorio de la fórmula de la regla del trapecio.

Cada vez que se finaliza el primer bucle se imprime por pantalla el número de particiones y el valor de la integral.

Una vez ejecutado prácticamente todo el programa, se asigna otra variable para el tiempo final y se imprime la diferencia de tiempos, es decir, el tiempo que tarda en realizarse todas las particiones.

6. Bibliografía

Referencias

[1] Regla del Trapecio: http://es.wikipedia.org/wiki/Regla_del_trapecio

[2] Integrales: http://math2.org/math/integrals/es-tableof.htm

[3] Símbolos de LaTeX: http://web.ift.uib.no/Teori/KURS/WRK/TeX/symALL.html

[4] Interpolación de Lagrange: http://es.wikipedia.org/wiki/Interpolación_polinómica_de_Lagrange

[5] Teorema de Weirstrass: http://es.wikipedia.org/wiki/Teorema_de_aproximación_de_Weierstrass

[6] Integtración numérica: http://portales.puj.edu.co/objetosdeaprendizaje/Online/OA10/capitulo4/capitulo4_2.htm