Ingeniería en Seguridad Informática y Redes

Ingeniería en Sistemas Computacionales

Algoritmos de Solución Numérica

Función de repaso

SI727576 - Edgar Guzmán Claustro

IS727272 - Marco Ricardo Cordero Hernández

Jal., 19 de septiembre de 2023

Índice

Contexto

En el análisis de funciones de una variable, usualmente se sigue un recorrido marcado partiendo desde la introducción al álgebra y sus incógnitas, pasando por las cónicas, series, límites, derivadas y finalmente integrales. Sin involucrar al cálculo multivariable, esta última disciplina es quizás una de las más complicadas en retrospectiva con la obtención de una derivada, su comprensión e interpretación.

Para atender los fundamentos de este campo del cálculo, un concepto introductorio es el método de las sumas para obtener el área bajo una curva, mejor conocido como *sumas de Riemann* (Jin, 2020). En su más pura y simplista esencia, el funcionamiento de este método consta en la suma del área de múltiples rectángulos, en donde su base está dada por un ancho infinitesimalmente diminuto y su altura se determina por la evaluación de alguna función en cada elemento del dominio correspondiente. La definición formal es la siguiente:

$$\lim_{n \to \infty} \sum_{i=1}^{n} f(x_i)\Delta x_i = \int_a^b f(x)dx$$

En la actualidad, existen diversas tecnologías que permiten calcular el área de manera exacta, sin embargo, es posible aproximar dicho valor tan precisamente como se desee al modificar el límite de la sumatoria a un valor finito para un rango determinado → [a, b].

Para realizar el cálculo de lo descrito previamente, así como la visualización de los rectángulos mencionados, se hará uso de la herramienta *Matlab* (The MathWorks Inc., 2023), en donde es posible manipular con relativa facilidad los parámetros necesarios para lograr el objetivo ya mencionado.

También, cabe mencionar en este punto que las sumas de Riemann tienen algunas variantes, sin embargo, para este ejercicio se ha utilizada la denominada *suma de Riemann* ***por la derecha***, en donde cada evaluación de la función para un subintervalo $[x_i , x_{i+s}]$ es evaluada para el extremo *derecho* del mismo rango. El uso de este método dará como resultado distintos valores según se especifique la afinidad del espacio entre los intervalos (s), pudiendo terminar con valores muy acercados al área real, o con resultados sustancialmente desfasados. La alternativa y resolución para este problema evidentemente es la integral u otras variantes como sumas por punto medio o la regla del trapecio (Khan Academy, s.f.), no obstante, la revisión de estos métodos está fuera del alcance de este desarrollo.

Resultados

El siguiente conjunto de códigos realiza el cálculo de la sumatoria de Riemann y lo despliega visualmente a través de una gráfica, respectivamente.

Datos de entrada

Dentro de *summation.m* (archivo de la función) se reciben los siguientes parámetros:

- *f* : Función sobre la cual se realizará la sumatoria
- *li* : Límite inferior (extremo izquierdo de sumatoria)
- *ls* : Límite superior (extremo derecho de sumatoria)
- *s* : De ***stepping***; paso entre intervalo de suma (dx de integral)

Cuando se invoca correctamente esta función, el resultado es un arreglo unidimensional de todas las evaluaciones en un punto x_i determinado internamente por *s*, con la suma final anexada al final del arreglo como último elemento → $[f(x_0), f(x_1), \ldots, f(x_{(|li| + |ls|)/s}), \Sigma f(x_i)]$; Caso contrario, devolverá un 0.

Dentro de *summationPlot.m* (archivo que invoca y grafica los resultados de la función anterior), las siguientes variables pueden ser modificadas para obtener el resultado deseado:

- *x* : Dominio de la gráfica a mostrar (límite izquierdo : paso : límite derecho)
- *y* : Función a graficar (probada con ecuaciones de hasta segundo grado)
- *s* : De ***stepping***; paso entre intervalo de suma (dx de integral)
- *li* : Límite inferior (extremo izquierdo de sumatoria)
- *ls* : Límite superior (extremo derecho de sumatoria)

Con la correcta y única definición de estas variables, el programa desarrollado debería mostrar la gráfica y el resultado deseado.

Método de sumatoria

```
function sum = summation(f, li, ls, s)
%{
   f  = Función a evaluar
   li = Límite inferior
   ls = Límite superior
   s  = Paso entre evaluaciones
%}

% Variables locales
helper = 0;                 % Variable de ayuda
res = 0;                    % Contenedor de resultado (suma
actual)
sum = [];                   % Elementos
% Comprobación de límites
if (li == ls)
  sum = 0;
```

```
  return
elseif (li > ls)            % Asegurar orden de límites
   helper = li;
   li = ls;
   ls = helper;
end

% Realizar n sumas hasta recorrer todo el rango
for i = li:s:ls-s
   helper = f(i + s);       % Evaluación de la función en el punto
actual
   sum(end + 1) = helper;   % Anexar evaluación a resultado final
   res = res + helper * s; % Acumular área (base por altura) a
suma final
end

sum(end + 1) = res;
```

Gráfica de sumatoria

```
% Restablecer entorno
clear, clc, close all

% Definición de parámetros
x = -10 : 0.01 : 10;                    % Dominio predefinido
y = @(x) x + 3;                         % Función con raíces
reales
s = 0.5;                                % Paso de sumatoria
[li, ls] = deal(-3, 3);                 % Límites de integración
cy = 0;                                 % Auxiliar para
rectángulos

% Graficar resultados
figure('name','Integral por sumatoria', ...
   'NumberTitle','off');               % Título de ventana
title('Integral por sumatoria')         % Título de la gráfica
xlabel('x')                             % Etiqueta del eje x
ylabel(strrep(char(y),'@(x)','y = '))   % Etiqueta del eje y
grid on                                 % Cuadrícula de la gráfica
hold on                                 % Conservar gráfica

% Visualización de la sumatoria
elements = summation(y, li, ls, s);
for h = 1:length(elements) - 1
   if (elements(h) < 0)
       cy = elements(h);
   else
```

```
        cy = 0;
    end
    rectangle('Position', [li+s*(h-1) cy s abs(elements(h))], ...
            'FaceColor', '#c7c7c7');
end

plot(x, y(x), 'k')                        % Gráfica de función
plot([0 0], ylim, 'k-')                   % Eje de abscisas
plot(xlim, [0 0], 'k-')                   % Eje de ordenadas
text(0, -1, strcat('Área = ', num2str(elements(end)))); %
Resultado

% Mostrar límite inferior
plot([li li], ylim, 'k--')
text(li, -1, strcat('x = ', num2str(li)));

% Mostrar límite superior
plot([ls ls], ylim, 'k--')
text(ls, -1, strcat('x = ', num2str(ls)));
```

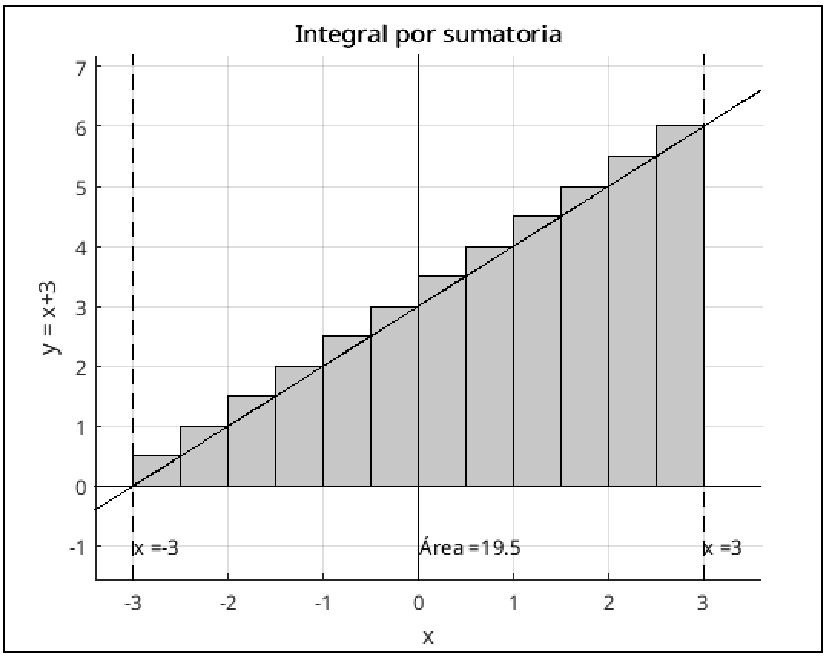
Casos de prueba y resultados

Caso 1

Integral por sumatoria					
	Función de entrada -> f(x) = x + 3			Límite inferior = -3	
	Límite superior = 3		Paso = 0.5	Resultado esperado = 18	
i	Li actual	Ls actual	f(i)	Área	Suma actual
1	-3	-2.5	0.5	0.25	0.25
2	-2.5	-2	1	0.5	0.75
3	-2	-1.5	1.5	0.75	1.5
4	-1.5	-1	2	1	2.5
5	-1	-0.5	2.5	1.25	3.75
6	-0.5	0	3	1.5	5.25
7	0	0.5	3.5	1.75	7
8	0.5	1	4	2	9
9	1	1.5	4.5	2.25	11.25
10	1.5	2	5	2.5	13.75

11	2	2.5	5.5	2.75	16.5
12	2.5	3	6	3	19.5

Resultado

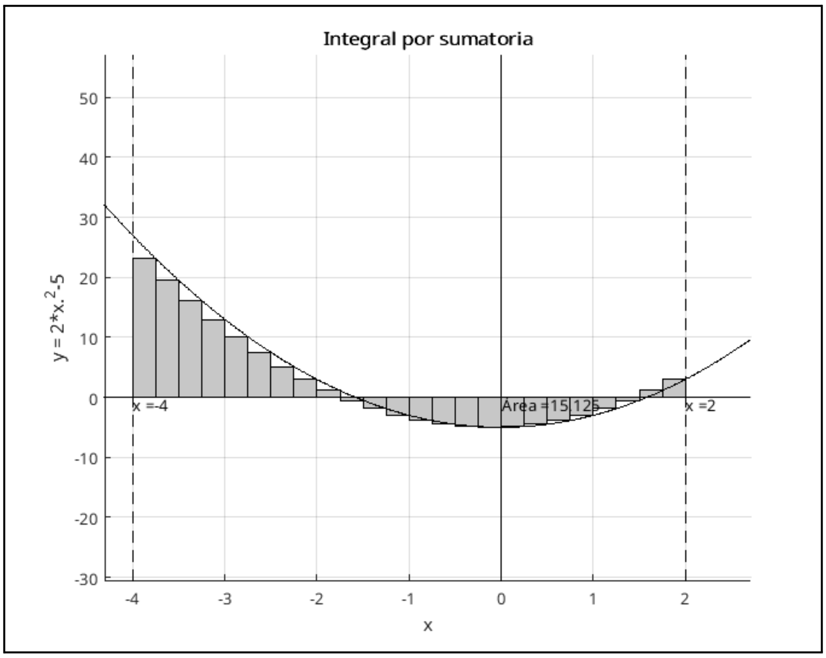


Caso 2

Integral por sumatoria					
Función de entrada -> f(x) = 2x² - 5				Límite inferior = -4	
Límite superior = 2			Paso = 0.25	Resultado esperado = 18	
i	Li actual	Ls actual	f(i)	Área	Suma actual
1	-4	-3.75	23.125	5.78125	5.78125
2	-3.75	-3.5	19.5	4.875	10.65625
3	-3.5	-3.25	16.125	4.03125	14.6875
4	-3.25	-3	13	3.25	17.9375
5	-3	-2.75	10.125	2.53125	20.46875
6	-2.75	-2.5	7.5	1.875	22.34375
7	-2.5	-2.25	5.125	1.28125	23.625
8	-2.25	-2	3	0.75	24.375

9	-2	-1.75	1.125	0.28125	24.65625
10	-1.75	-1.5	-0.5	-0.125	24.53125
11	-1.5	-1.25	-1.875	-0.46875	24.0625
12	-1.25	-1	-3	-0.75	23.3125
13	-1	-0.75	-3.875	-0.96875	22.34375
14	-0.75	-0.5	-4.5	-1.125	21.21875
15	-0.5	-0.25	-4.875	-1.21875	20
16	-0.25	0	-5	-1.25	18.75
17	0	0.25	-4.875	-1.21875	17.53125
18	0.25	0.5	-4.5	-1.125	16.40625
19	0.5	0.75	-3.875	-0.96875	15.4375
20	0.75	1	-3	-0.75	14.6875
21	1	1.25	-1.875	-0.46875	14.21875
22	1.25	1.5	-0.5	-0.125	14.09375
23	1.5	1.75	1.125	0.28125	14.375
24	1.75	2	3	0.75	15.125

Resultado



Conclusiones

Guzmán Claustro, Edgar

Este ejercicio sirvió para retomar conceptos ya antes aprendidos en materias de cálculo pasadas. También, es de gran apoyo para aprender a manejar matlab y darle una aplicación útil al método desarrollado en este trabajo. La realización de este tipo de funciones en código, puede servir para que en un futuro se pueda crear una recopilación de todas la que se han estado desarrollando en el curso y que funcionen en una sola interfaz.

Se nota el potencial que puede llegar a tener el lenguaje y sin duda cuenta con una sintaxis más fácil de comprender que algunos otros. Aunque sin duda, los pasos para desarrollar un algoritmo son los mismos sin importar la plataforma en la que se codifica.

Cordero Hernández, Marco R.

La realización de este ejercicio sirve como repaso no solo de la declaración, invocación y utilización de una función dentro de Matlab, sino que también ha sido útil para repasar conceptos básicos de lo que alguna vez se revisó en cálculo integral hace ya algunos ayeres. Guiados por el principio de simplicidad (mejor conocido como *KISS* en inglés), el deseo por realizar esta visualización viene del repaso de la esencia de las cosas, de volver a lo básico del todo, porque todo gran desarrollo comenzó de apenas unos cuantos conceptos unidos para hacer sentido.

En este caso, la comprensión de los conceptos detrás de la integral y su computación (en la más pura definición de la palabra) no es sino el primer paso a lo que puede llegar a convertirse en una herramienta informática de cálculo de integrales, la cual debería ser optimizada para realizar millones de operaciones en solo unos cuantos milisegundos.

Finalmente, haciendo referencia nuevamente a Matlab, repasar su sintaxis resulta una acción que cimentará las habilidades con el mismo software para futuros desarrollos en la revisión de métodos numéricos mucho más complejos. Sin duda alguna, realizar estos algoritmos es una actividad sumamente enriquecedora, lo cual mantiene la habilidad del diseño y desarrollo como algo fresco, como si se descubriera por primera vez en cada ocasión.

Bibliografía

Jin, E. (2020). *Integration and Summation*. Recuperado de https://www.mit.edu/~ehjin/files/Integration_And_Summation.pdf.

The MathWorks Inc. (2022). MATLAB version: 9.14.0 (R2023a), Natick, Massachusetts: The MathWorks Inc. https://www.mathworks.com.

Khan Academy. (s.f.). *Repaso de sumas de Riemann*. Recuperado de https://es.khanacademy.org/math/ap-calculus-ab/ab-integration-new/ab-6-2/a/riemann-sums-review#:~:text=Una%20suma%20de%20Riemann%20es,tales%20como%20rect%C3%A1ngulos%20o%20trapecios.