Recurso Examen 2º Parcial 2021

En este recurso trataremos de resolver dos ejercicios bastante interesantes desde el punto de visto de la preparación de cara al examen parcial de enero, ya que vamos a resolver gran parte del 2º examen parcial de 2021. El primero de estos ejercicios, el ejercicio 1, es un ejercicio en el que entrenaremos el manejo de matrices y muchas de las funcionalidades que podemos obtener de estas, mientras que el segundo que haremos, el ejercicio 3. A), algo más complicado, se puede asemejar algo más a un algoritmo de interpolación, ya que se pide hallar la aproximación de una integral de un modo específico.

EJERCICIO 1 (3 puntos)
El famoso mago Mr. Dados dice que cuando se lanzan p dados m veces, es capaz de adivinar cuántas veces sale un número par y cuantas un número impar y además separar los valores pares y los impares. Sin embargo, por si falla el truco, te pide que realices un algoritmo que siga el siguiente proceso:

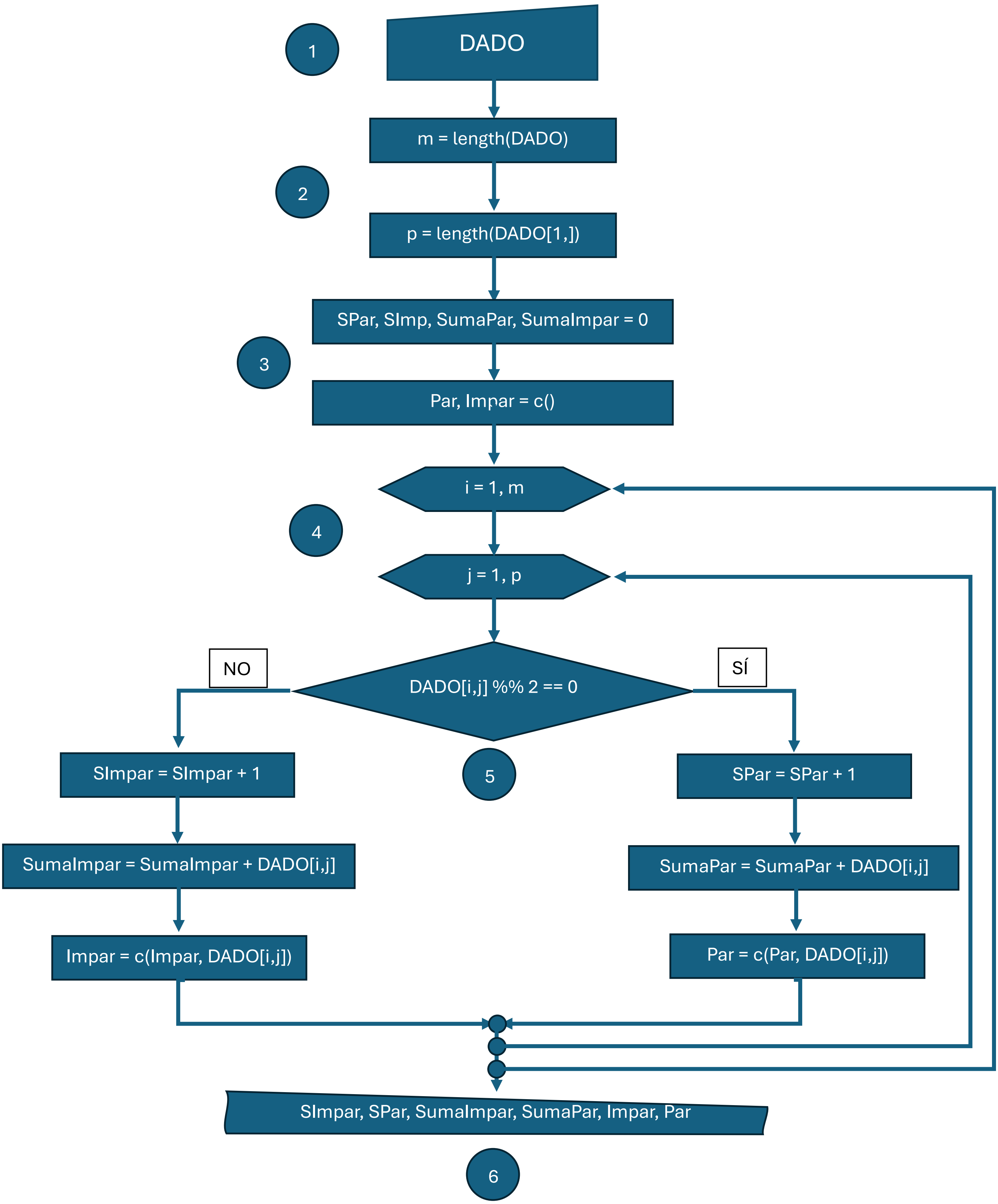
Los datos: una matriz DADO de m filas y p columnas que contiene los números que han salido al lanzar los dados (es decir, todos los valores serán números enteros entre 1 y 6).

Obtener una variable llamada **SPar** que contenga la cantidad de números pares que hay en **DADO**, otra variable **SumaPar** que contenga la suma de los valores pares y un vector llamado **Par** que contenga dichos números.

Obtener una variable llamada **SImp** que contenga la cantidad de números impares que hay en **DADO**, otra variable llamada **SumaImpar** que contenga la suma de los números impares y un vector llamado **Impar** que contenga dichos números.

En este primer ejercicio, se nos propone, a partir de una matriz dada, obtener dos variables enteras que contengan la cantidad de número pares e impares de la matriz, respectivamente. Por otro lado, se nos pide, guardar en un vector estos números por separado, pares en un vector e impares en otro, y acumular en dos variables la suma de los valores de cada uno de esos dos vectores. Por lo tanto, deberemos recorrer esta matriz, guardando toda esta información en nuestras nuevas variables, para luego devolverlas. Por último, se requiere que devolvamos una matriz con los dos vectores mencionados anteriormente como las columnas de esta matriz.

Presentamos ahora el algoritmo que hará todas estas operaciones.


1
DADO
m = length(DADO)
2
p = length(DADO[1,])
SPar, SImp, SumaPar, SumaImpar = 0
3
Par, Impar = c()
i = 1, m
4
j = 1, p
NO
DADO[i,j] %% 2 == 0
SÍ
SImpar = SImpar + 1
5
SPar = SPar + 1
SumaImpar = SumaImpar + DADO[i,j]
SumaPar = SumaPar + DADO[i,j]
Impar = c(Impar, DADO[i,j])
Par = c(Par, DADO[i,j])
SImpar, SPar, SumaImpar, SumaPar, Impar, Par
6

1. En primer lugar, recibimos como parámetro de entrada la matriz DADO que vamos analizar.
2. Guardamos en dos variables las dimensiones de esta matriz, m (filas) para el número de veces que se tiran los dados y p (columnas) para el número de dados que se lanzan en cada tirada. Posteriormente, utilizaremos estas variables para recorrer la matriz con los bucles.
3. Inicializamos las variables que necesitamos devolver a 0 en caso de ser enteros y a vectores vacíos en caso de ser vectores.
4. Entramos en un primer bucle for que recorre las filas de la matriz para después recorrer cada elemento de cada fila mediante otro bucle for que va desde 1 hasta p.
5. Preguntamos si el elemento de la fila i y columna j de la matriz DADO es par con un condicional if.
    a. Si es par, nos vamos por la derecha, añadimos 1 a la variable SPar (lleva el conteo de números pares), añadimos el valor del elemento a la variable SumaPar (que va acumulando la suma de números pares que tiene la matriz) y añadimos el valor de dicho elemento al final del vector Par.
    b. Si es impar, nos vamos por la izquierda, añadimos 1 a la variable SImpar (lleva el conteo de números impares), añadimos el valor del elemento a la variable SumaImpar (que va acumulando la suma de números impares que tiene la matriz) y añadimos el valor de dicho elemento al final del vector Impar.
6. Devolvemos el valor de todas las variables que nos piden como salida: Simpar, SPar, SumaImpar, SumaPar, Impar y Par.

EJERCICIO 3 (4 puntos)
PARTE A) (3 puntos). Se considera que la acumulación de cierto fármaco en el organismo está dada por la integral:

$$\int_a^b f(x)\,dx$$

siendo x el tiempo transcurrido entre dos instantes: a, b. Se desea obtener el valor de la integral mediante una fórmula de **Gauss compuesta**, con un número creciente de sub-intervalos. El proceso comenzará considerando todo el intervalo [a,b] y se irá duplicando el número de sub-intervalos mientras la diferencia entre el valor exacto de la integral y el aproximado sea mayor que cierta tolerancia prefijada.

Para ello, se realizará un **ALGORITMO** tal que, dados:

- una variable f que contiene la expresión de la función que representa la acumulación del fármaco,
- las variables a, b que son los extremos del intervalo de integración,
- una variable valex, que contiene el valor exacto de la integral (supuesto conocido),
- una variable tol que es la tolerancia admisible en el proceso iterativo desarrollado,

se aplique una fórmula de Gauss compuesta, que emplee un soporte de dos puntos en cada sub-intervalo, de manera que el número de sub-intervalos se vaya duplicando mientras la diferencia en valor absoluto entre el valor exacto y el aproximado (variable **dif**) sea mayor que una tolerancia prefijada (**tol**).

La fórmula de Gauss compuesta con dos puntos de soporte en cada sub-intervalo viene dada por:

$$\int_a^b f(x)\,dx \approx \frac{h}{2}\sum_{i=1}^{N} f\left(\frac{s_i + s_{i+1}}{2} - \frac{h\sqrt{3}}{3}\right) + f\left(\frac{s_i + s_{i+1}}{2} + \frac{h\sqrt{3}}{3}\right)$$

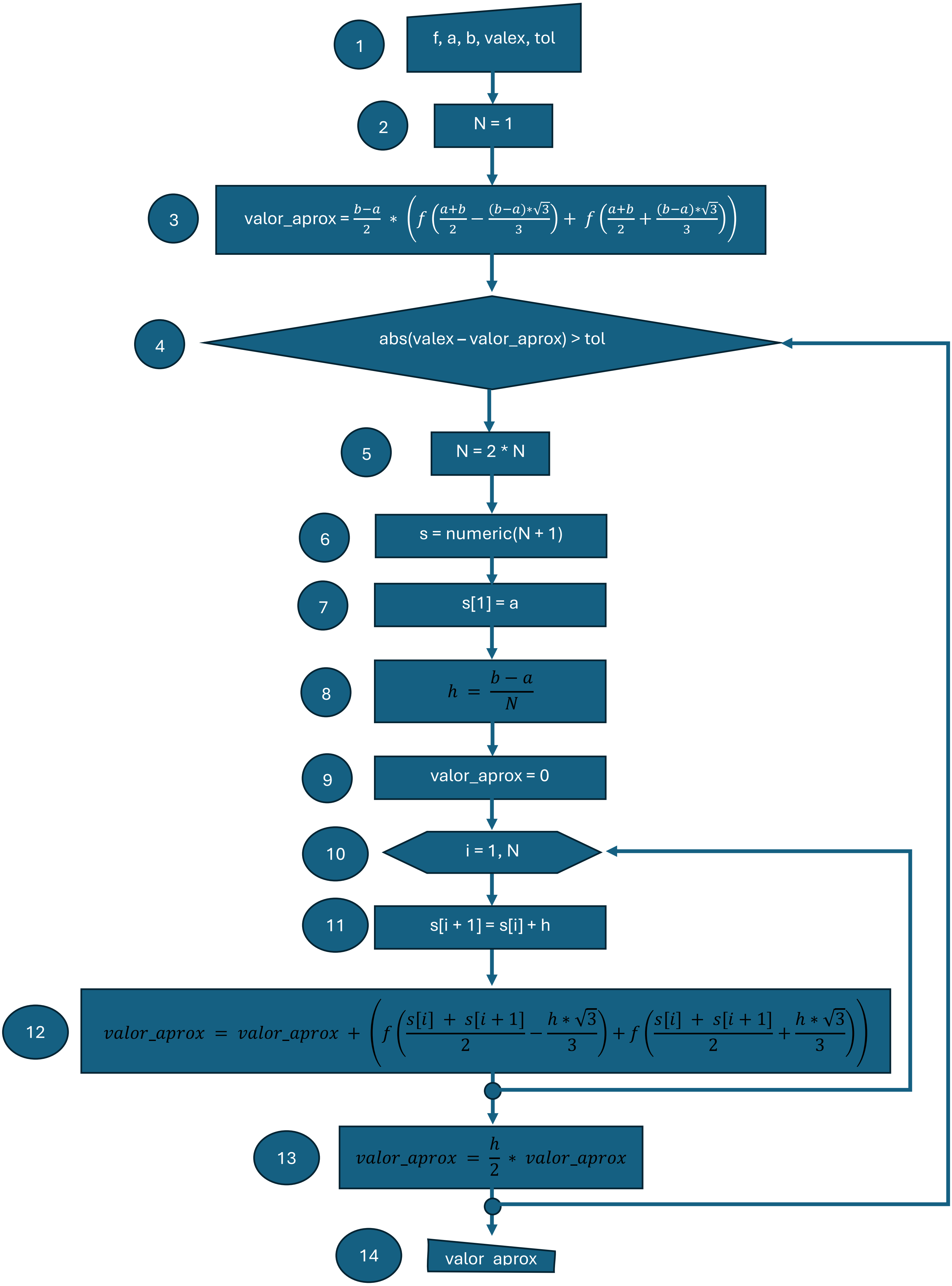
Siendo

$$h = s_{i+1} - s_i$$

Donde **N** es el número de sub-intervalos que se van considerando durante las iteraciones. El vector **s** contiene las coordenadas de los puntos que constituyen los extremos de cada sub-intervalo y que habrá que calcular dentro del proceso iterativo.

Al iniciar el proceso iterativo se tomará N=1 (se irá duplicando durante el proceso iterativo).

Para resolver este segundo ejercicio de examen, utilizaremos la estructura iterativa del bucle while, ya que debemos seguir haciendo iteraciones duplicando el número de intervalos que tomamos hasta alcanzar una aproximación suficientemente cercana al valor real. Para ello, en cada iteración comprobaremos que el valor absoluto del valor real que nos dan menos el valor aproximado que obtenemos nosotros mediante la fórmula es menor que la tolerancia dada. Si no lo es, hacemos una nueva iteración en la que aumentamos el número de puntos del soporte y sacamos el valor aproximado de la integral con un bucle for para recrear el sumatorio de la fórmula. Planteamos el algoritmo para explicarlo más tarde.


1
f, a, b, valex, tol
2
N = 1
3
valor_aprox = $\frac{b-a}{2} * \left(f\left(\frac{a+b}{2} - \frac{(b-a)*\sqrt{3}}{3}\right) + f\left(\frac{a+b}{2} + \frac{(b-a)*\sqrt{3}}{3}\right)\right)$
4
abs(valex – valor_aprox) > tol
5
N = 2 * N
6
s = numeric(N + 1)
7
s[1] = a
8
$h = \frac{b-a}{N}$
9
valor_aprox = 0
10
i = 1, N
11
s[i + 1] = s[i] + h
12
$valor\_aprox = valor\_aprox + \left(f\left(\frac{s[i] + s[i+1]}{2} - \frac{h*\sqrt{3}}{3}\right) + f\left(\frac{s[i] + s[i+1]}{2} + \frac{h*\sqrt{3}}{3}\right)\right)$
13
$valor\_aprox = \frac{h}{2} * valor\_aprox$
14
valor_aprox

1. Recibimos como parámetros de entrada:
    a. f: función que vamos a integrar
    b. a: límite inferior de integración
    c. b: límite superior de integración
    d. valex = valor real de la integral que vamos a estimar
    e. tol = error máximo que puede tener nuestra estimación
2. Establecemos que el soporte inicial sólo va a tener dos puntos (a y b) y por tanto sólo va a haber un intervalo (N = 1) sobre el que vamos a aplicar la fórmula.
3. Calculamos el valor aproximado de la integral mediante la fórmula de Gauss completa para un solo intervalo.
4. Comparamos en el while la diferencia entre el valor real y el valor aproximado con la tolerancia máxima que nos piden que esta diferencia no supere. Si la diferencia es mayor que la tolerancia, seguimos entrando en el cuerpo del bucle, para intentar realizar una aproximación más exacta, y así, hasta que consigamos un valor aproximado con error con respecto al valor real menor que la tolerancia.
5. Al entrar al bucle, duplicamos el número de intervalos que tenemos (N = 2*N), y, por tanto, tendremos 2*N + 1 puntos en el soporte.
6. Así, inicializamos el vector que contendrá los puntos del soporte, con N + 1 ceros (N ya está actualizado).
7. El primer punto del soporte siempre será el punto a.
8. Calculamos el paso h (distancia entre los puntos del soporte), que será el intervalo total (b-a) entre el número total de intervalos.
9. Inicializamos el valor aproximado a 0, ya que este valor se obtendrá mediante un sumatorio, que calcularemos en el siguiente bucle for.
10. Entramos en un bucle for, cuyo índice i recorrerá los números enteros de 1 hasta N (número de intervalos) y en el que iremos añadiendo los sumandos del sumatorio, los cuales dependen de los puntos del soporte i e i+1.
11. Calculamos los valores del soporte sobre la marcha, en función del h calculado anteriormente. En cada iteración i, calcularemos el punto s[i+1] del soporte que necesitamos en la fórmula.
12. Hallamos la parte del sumatorio de la fórmula de Gauss compuesta para empezar a estimar el valor de la integral.
13. Multiplicamos el valor del sumatorio por h/2 para obtener el valor de la fórmula por completo y volvemos a realizar la pregunta del while. Si se cumple la condición, volvemos a entrar en el cuerpo del bucle, y, en caso contrario, salimos del bucle while.
14. Una vez que hemos salido, esto significa que el error absoluto de la aproximación es menor que la tolerancia, y por tanto, que podemos devolver el valor aproximado de la integral.