

Práctica 0

Pablo Arranz Roperro
Juan Alberto Camino Sáez
Grupo 2

Práctica 0: Octave (Integral de Montecarlo)

La práctica consiste en calcular la integral de una función mediante el método de Montecarlo.

Este método consiste en generar puntos aleatorios cuya coordenada x estará entre a y b , siendo a y b los puntos en los que se define la integral y cuya coordenada y estará entre 0 y M siendo M el máximo de la función entre a y b .

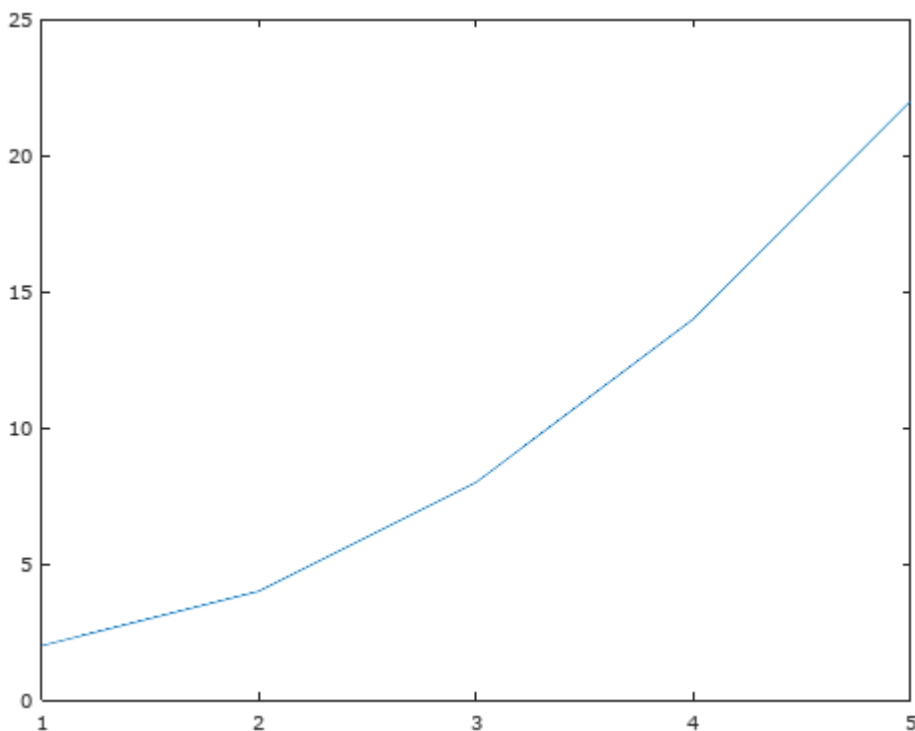
Una vez generados dichos puntos el cálculo se hace dividiendo los puntos generados aleatoriamente que quedan por debajo de la ecuación y los puntos generados en total

$$I \approx \frac{N_{debajo}}{N_{total}} (b - a)M$$

Cuanto más puntos generemos más aproximado será el resultado al valor de la integral.

Algoritmo generado

Para explicar nuestro código, usaremos como ejemplo la función $f(x) = x^2 - x + 2$ en el intervalo a - b siendo $a = 1$ y $b = 2$, representada con la siguiente gráfica:



Para calcular el máximo de la función en ese intervalo, hemos creado un array que vaya desde a hasta b con una diferencia de 0.001, y luego calculamos $f(x)$ de todos los valores del array y buscamos el máximo en ese array de resultados. De esta forma, obtenemos el valor M , que en este caso se alcanza con el valor $x = 5$.

Después, creamos un array de números aleatorios entre a y b (en el ejemplo, entre 1 y 5), tantos como se indican en la cabecera de la función. Una vez creados, se obtiene otro array obteniendo $f(x)$ de esos puntos aleatorios, y, mediante un bucle for (primera solución) o mediante operaciones con vectores (segunda solución) contamos los valores que quedan por debajo de la gráfica, y con ello ya tenemos todos los datos necesarios para la fórmula, cosa que nuestro código realiza y devuelve el resultado.

A continuación, se incluye el código generado:

- Primera solución con bucle for:

```
1 function [valor] = mcint(fun, a, b,num_puntos)
2
3     numAlX = rand(1, num_puntos) * (b - 1) + a;
4
5     X=[a:0.001:b];
6     res = fun(X);
7     M = max(res);
8
9     numAlY = rand(1, num_puntos) * M;
10
11     resultados = fun(numAlX);
12     ndebajo = 0;
13     for i= 1: num_puntos
14
15         if(numAlY(i) < resultados(i))
16             ndebajo = ndebajo + 1;
17         endif
18
19     endfor
20
21     valor = M * (b - a) * (ndebajo/num_puntos);
22
23
24     endfunction
```

- Segunda solución con operaciones de vectores:

```
1 function [valor] = mcintvector(fun, a, b,num_puntos)
2
3     numAlX = rand(1, num_puntos) * (b - 1) + a;
4
5     X=[a:0.001:b];
6     res = fun(X);
7     M = max(res);
8
9     numAlY = rand(1, num_puntos) * M;
10
11     resultados = fun(numAlX);
12     resta = numAlY - resultados;
13     ndebajo = sum(resta(:) < 0);
14
15     valor = M * (b - a) * (ndebajo/num_puntos);
16
17
18     endfunction
```