Práctica 0

Pablo Arranz Ropero
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***.

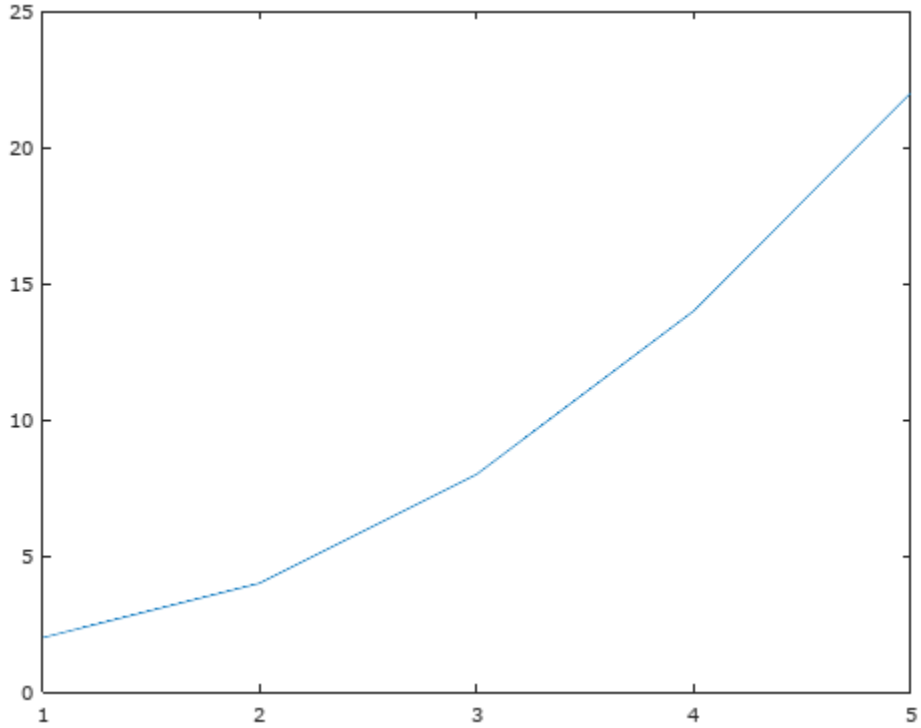
Una vez generados dichos puntos el cálculo se hace dividiendo los puntos generados aleatoriamente que quedan por debajo de la ecuación y los puntos generados en total

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

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

<u>Algoritmo generado</u>

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 = 5, 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:

```
function [valor] = mcint(fun, a, b,num_puntos)

  numAlX = rand(1, num_puntos) * (b - 1) + a;

  X=[a:0.001:b];
  res = fun(X);
  M = max(res);

  numAlY = rand(1, num_puntos) * M;

        resultados = fun(numAlX);
        ndebajo = 0;
        for i= 1: num_puntos

          if(numAlY(i) < resultados(i))
            ndebajo = ndebajo + 1;
          endif

        endfor

        valor = M * (b - a) * (ndebajo/num_puntos);


      endfunction
```

- Segunda solución con operaciones de vectores:

```
      function [valor] = mcintvector(fun, a, b,num_puntos)

  numAlX = rand(1, num_puntos) * (b - 1) + a;

  X=[a:0.001:b];
  res = fun(X);
  M = max(res);

  numAlY = rand(1, num_puntos) * M;

        resultados = fun(numAlX);
        resta = numAlY - resultados;
        ndebajo = sum(resta(:) < 0);

        valor = M * (b - a) * (ndebajo/num_puntos);


      endfunction

```