**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.

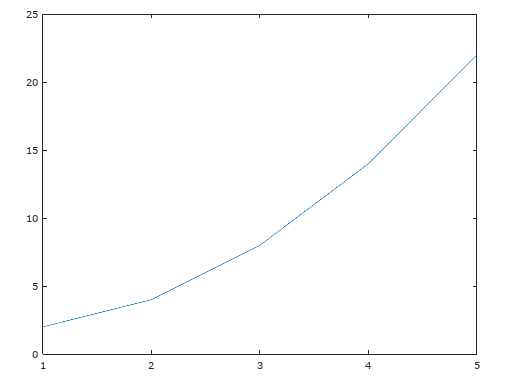
Este método consiste en generar puntos aleatorios cuya coordenada ***x*** estará entre ***a*** y ***b***, siendo ***a*** y ***b*** los puntos en los que se define la integral y cuya coordenada ***y*** estará entre 0 y ***M*** siendo ***M*** el máximo de la función entre ***a*** y ***b***.

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

Cuantos 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 = 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:

1. **function** [valor] **=** mcint(fun, a, b,num\_puntos)
3. numAlX **=** [rand](http://octave.sourceforge.net/octave/function/rand.html)(1, num\_puntos) **\*** (b **-** 1) **+** a;
5. X**=**[a:0.001:b];
6. res **=** fun(X);
7. M **=** [max](http://octave.sourceforge.net/octave/function/max.html)(res);
9. numAlY **=** [rand](http://octave.sourceforge.net/octave/function/rand.html)(1, num\_puntos) **\*** M;
11. resultados **=** fun(numAlX);
12. ndebajo **=** 0;
13. **for** [**i**](http://octave.sourceforge.net/octave/function/i.html)**=** 1: num\_puntos
15. **if**(numAlY([**i**](http://octave.sourceforge.net/octave/function/i.html)) **<** resultados([**i**](http://octave.sourceforge.net/octave/function/i.html)))
16. ndebajo **=** ndebajo **+** 1;
17. **endif**
19. **endfor**
21. valor **=** M **\*** (b **-** a) **\*** (ndebajo**/**num\_puntos);

24. **endfunction**

* Segunda solución con operaciones de vectores:

1. **function** [valor] **=** mcintvector(fun, a, b,num\_puntos)
3. numAlX **=** [rand](http://octave.sourceforge.net/octave/function/rand.html)(1, num\_puntos) **\*** (b **-** 1) **+** a;
5. X**=**[a:0.001:b];
6. res **=** fun(X);
7. M **=** [max](http://octave.sourceforge.net/octave/function/max.html)(res);
9. numAlY **=** [rand](http://octave.sourceforge.net/octave/function/rand.html)(1, num\_puntos) **\*** M;
11. resultados **=** fun(numAlX);
12. resta **=** numAlY **-** resultados;
13. ndebajo **=** [**sum**](http://octave.sourceforge.net/octave/function/sum.html)(resta(:) **<** 0);
15. valor **=** M **\*** (b **-** a) **\*** (ndebajo**/**num\_puntos);

18. **endfunction**