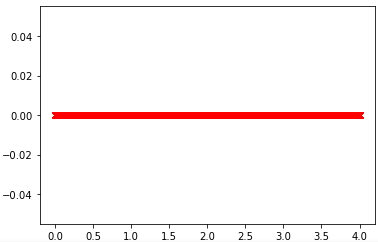
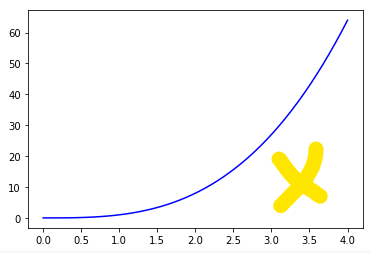
**Práctica 0 (Vectorización)**

¿Cómo he planteado la práctica?

Primero he creado un vector de puntos aleatorios en el eje X, estos puntos van desde A hasta B



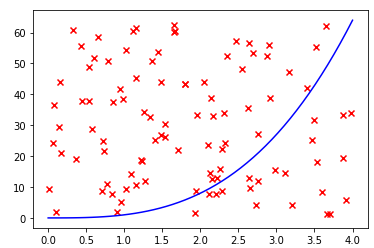
Queremos calcular la integral con los números que tenemos ***debajo*** de nuestra ***función***



Así que la ecuación dada para aproximarnos a la integral es:

Ahora obtenemos M, que es el máximo valor de la función en cualquier valor de X que hemos generado aleatoriamente.

Posteriormente, añadimos a nuestro vector el eje Y de la misma forma, aunque ahora los límites son de 0 a M.



A partir de aquí vamos a hacerlo de dos formas:

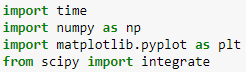
* **Bucles**: Recorremos todos los elementos del eje X y les aplicamos la función, este resultado lo comparamos con el punto aleatorio generado correspondiente en el eje Y, si es menor, está dentro del área y por lo tanto sumamos 1, si es mayor, no hacemos nada
* **Vectorización**: Comparamos directamente los resultados, de forma que nos devuelva un vector de booleanos, a este vector le aplicamos un sumatorio.

Tras ello, en ambas formas, aplicamos la fórmula que nos permite aproximar la integral.

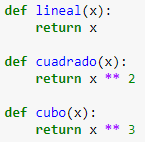
**NOTA**: Todo ello se podría usar en una única función para evitar repetir código, pero he preferido hacerlo en dos funciones distintas en pro de un aprendizaje académico.

Código comentado y gráficas

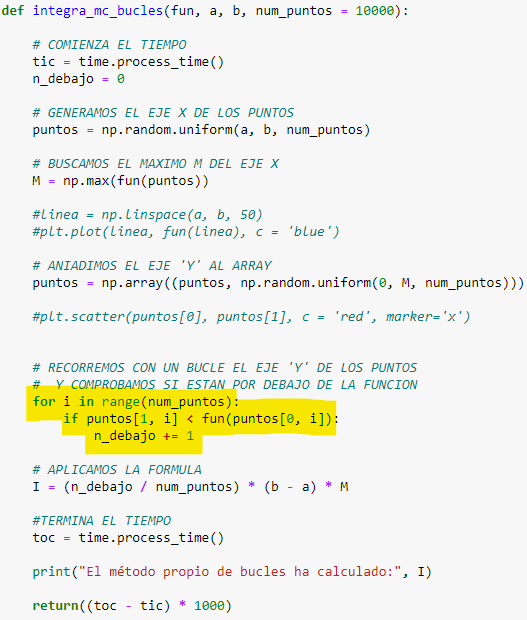
1. Importar las librerías

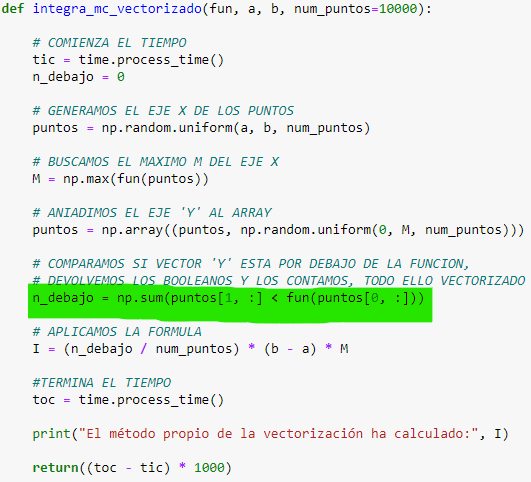


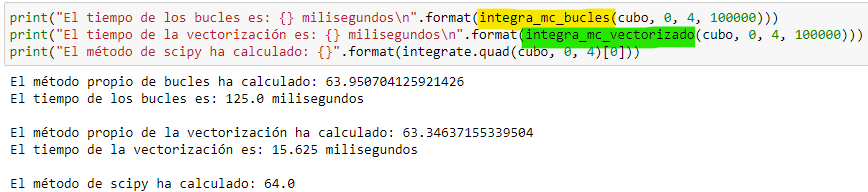
1. Crear funciones (Podría implementarse con funciones lambda, he preferido implementarlo en otras funciones con fines didácticos)



1. Función de bucles y función vectorizada

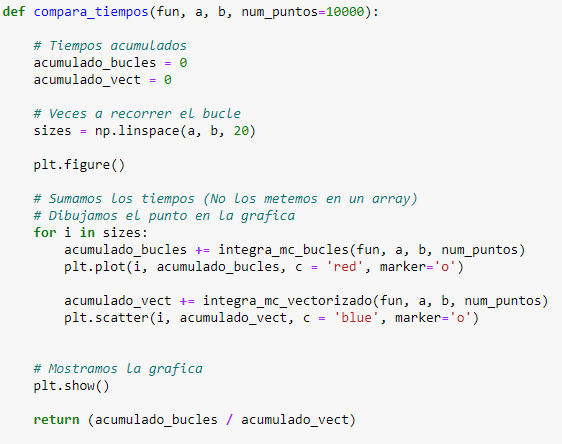




1. Ahora usamos las dos funciones anteriores y comparamos el resultado con la librería scipy. Como es lógico el tiempo de bucles es superior al tiempo de vectorización.

**NOTA:** Los resultados de ambos métodos propios difieren, esto es lógico porque en ambas funciones se crean puntos aleatorios, por lo que son distintos, aunque la diferencia es relativamente despreciable.

1. Ahora solo queda comparar los tiempos, en este caso, a través de una gráfica y numéricamente, ¿cuántas veces más es eficiente?



1. Y llamar a la función anterior, en este caso, como habíamos creado 3 funciones, vamos a probarlas.
2. Resultados

