**Taller de Matemática Computacional**

**Nombre**: Agustín

**Apellido**: Diaz Gargiulo

**DNI**: 38524932

**2017.**

**Facultad de Ciencias Exactas, UNCPBA.**

**TUDAI.**

**Introducción:**

Se nos planteó calcular la probabilidad de obtener dos 0 seguidos como resultado del llamado a una función dada, que mediante un número de documento devolvía aleatoriamente un valor entre 1 y 0. Se implementara el uso de un motor de Montecarlo, el cual calculará las probabilidad de que esto ocurra. Para la implementación de este motor de montecarlo utilizaremos Octave.

**Desarrollo**:

Para solucionar este problema, lo que hice fue implementar un Script , que lo llame **Script\_Trabajo\_Especial**, el cual utiliza principalmente una función **Converge** la cual es la que me permite controlar la que la resta entre la probabilidad Actual y la Probabilidad anterior sea menor o mayor a un epsilon dado. A su vez, esta funcion tambien verifica que se hayan realizado una cierta cantidad mínima ( yo implemente un mínimo de 60) de llamados a la funcion my\_mex\_service.mex (que es la que me devuelve 0 y 1) lo cual me permite que la probabilidad se pueda estabilizar, ya que al principio suele obtener resultados no precisos. Esta función converge me devuelve si la las probabilidades convergen o no.

Esta función es utilizada en un proceso al cual lo llame **Calcular\_Probabilidad**, en la cual lo que hacemos ir calculando la probabilidad en cada ciclo de del while controlado por la función previamente descrita. Para ello lo que hacemos es contar si el caso en el que nos encontramos es favorable o no. De serlo lo que hacemos es sumarlo a una variable que cuenta la cantidad de veces que ocurrió el caso de obtener dos ceros seguidos. Todos los demás casos no nos interesa, por eso simplemente lo que hacemos es decir que ese caso no fue exitoso y sumamos en un contador la cantidad de veces que vamos realizando esta prueba. Ahora sí podremos obtener la probabilidad actual, la cual la obtenemos de la división entre los casos favorables y el total de ciclos acumulados en ese momento. Esa probabilidad la guardaremos en un arreglo para luego ser mostrada por pantalla.

Cada vez que entramos en el while lo primero que hacemos es guardar el valor de esa probabilidad para así luego puedes comprarla con la que obtendremos así cuando llamamos a la función Converge nos dirá si la diferencia entre ambos valores es menor a epsilon o no.

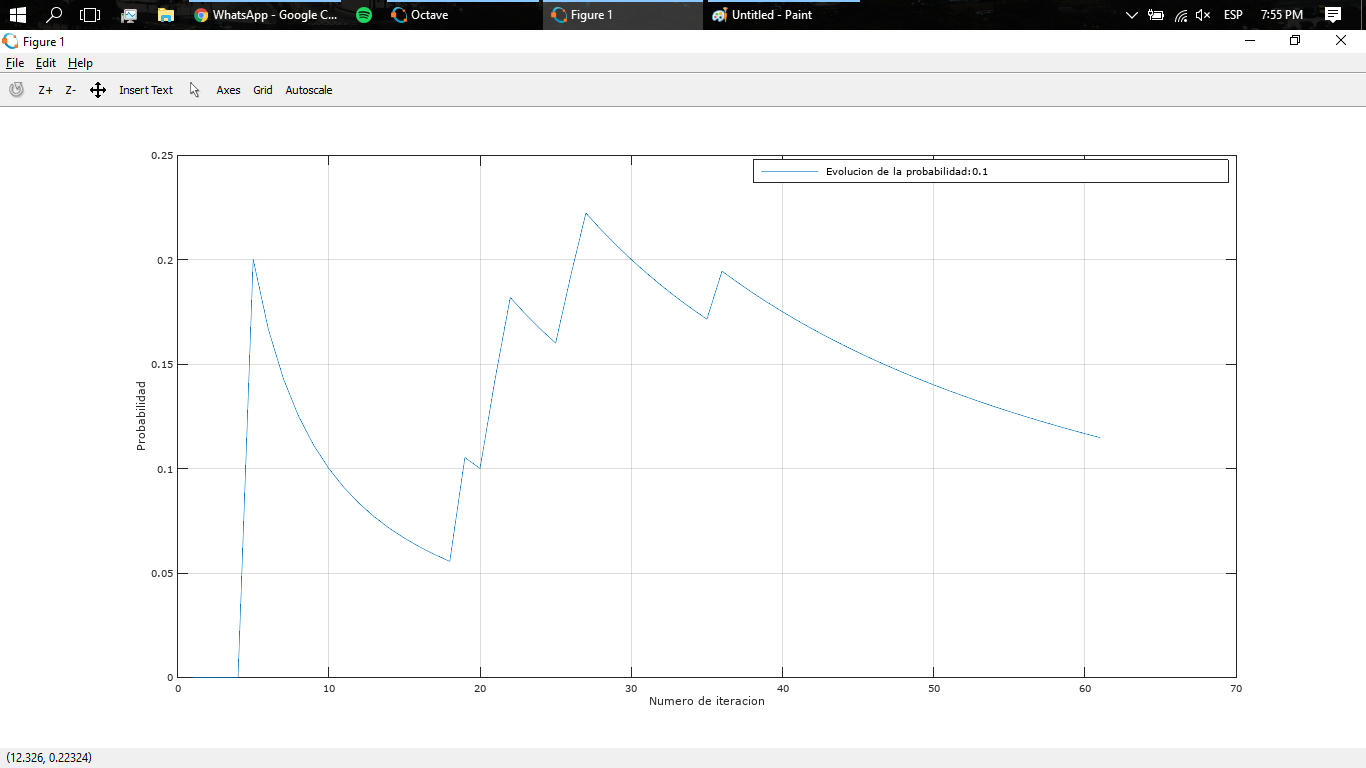
**Resultados:**

Lo que se puede ver en los gráficos es como varía bruscamente la probabilidad al principio durante los primeros casos de test hasta que luego se estabiliza obteniendo así un resultado más preciso.

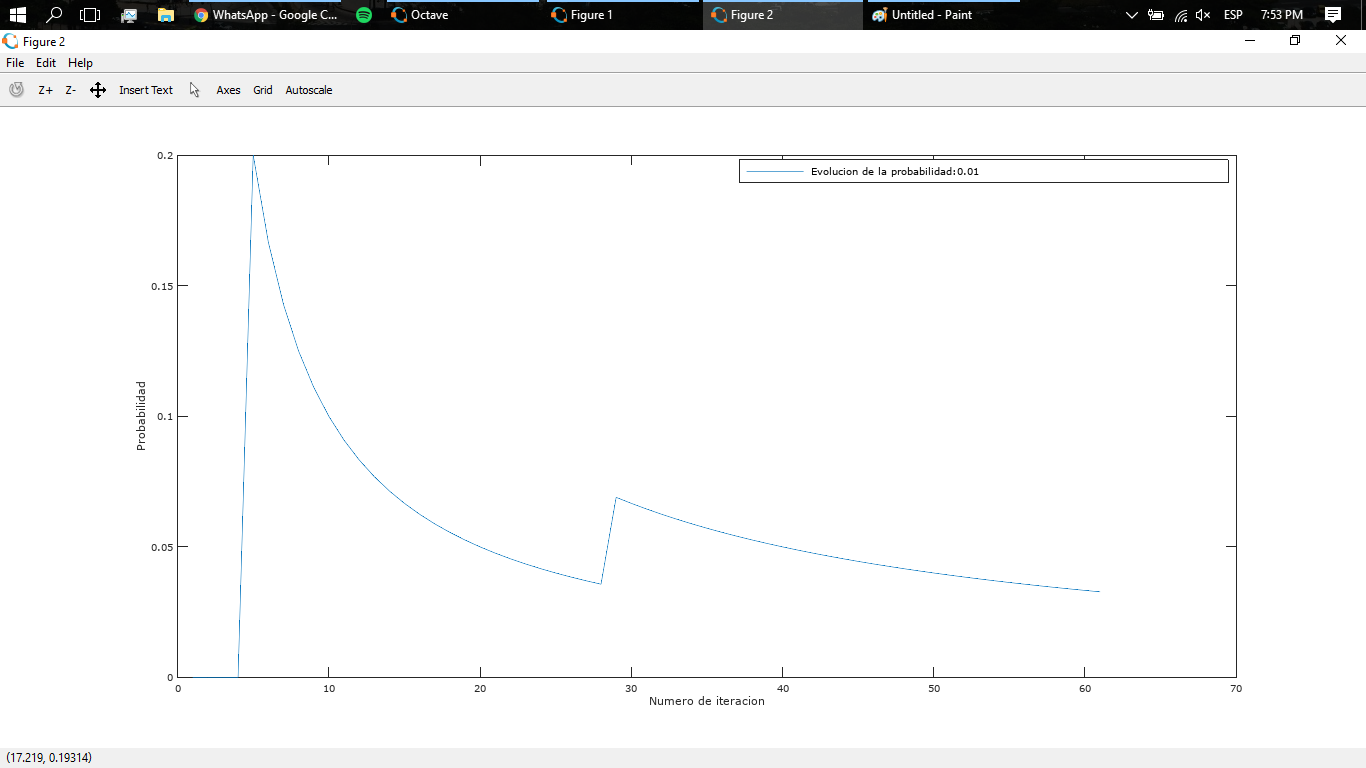
A medida que vamos bajando el epsilon lo que pasa es que el programa necesita realizar más iteraciones, por lo que un montecarlo con epsilon = 0.1 el cual necesitaba 70 ejecuciones del while, uno con un epsilon de 0.001 necesita 700 iteraciones del while. Por eso decimos que cuanto menor sea el valor del epsilon, obtendremos un valor de probabilidad más cercano a la realidad.

En mi caso los valores obtenidos fueron:

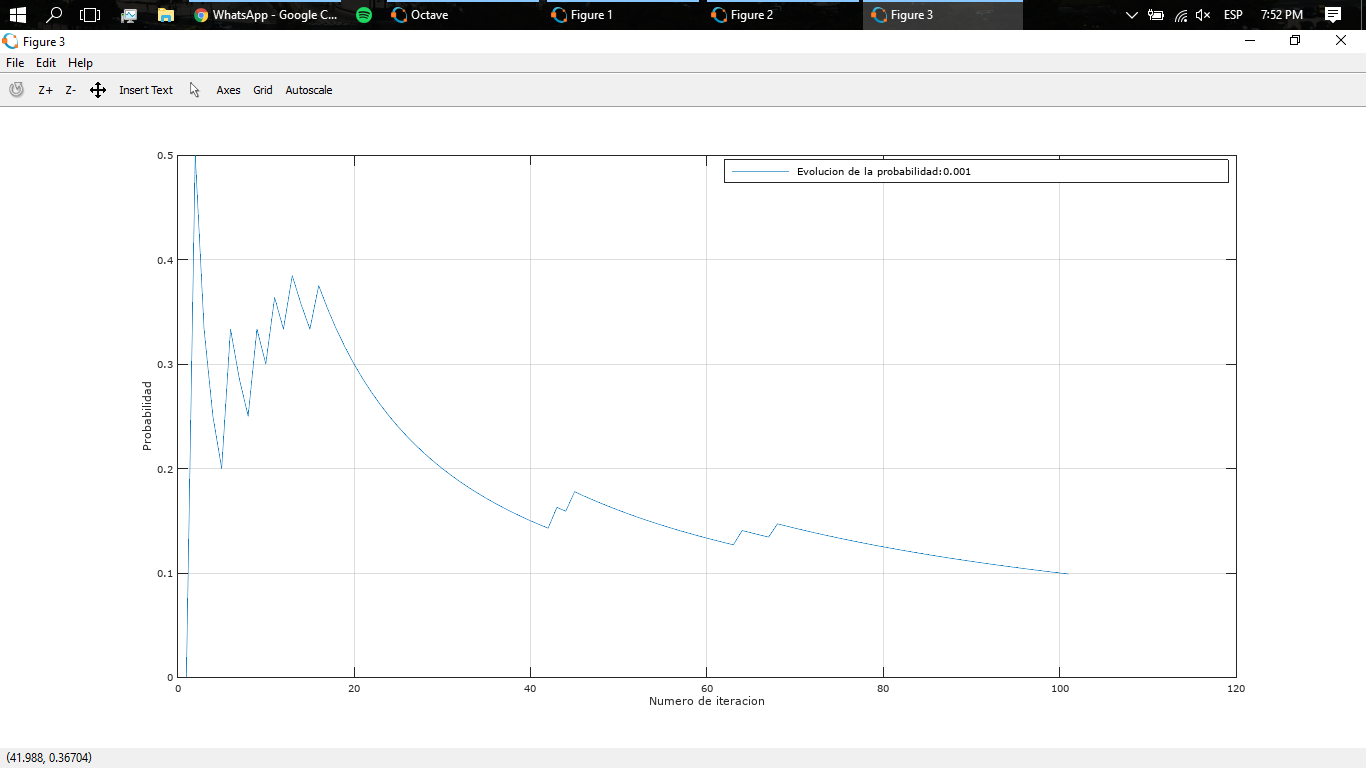
* Con epsilon 0.1 la probabilidad de que la función me diera dos 0 seguidos fue de: **0.01345**



* Con epsilon 0.01 la probabilidad de que la función me diera dos 0 seguidos fue de: **0.0481803**



* Con epsilon 0.001 la probabilidad de que la función me diera dos 0 seguidos fue de:  **0.1**



Otra cosa que pude observar fue que el valor de epsilon de 0.1 es el que mas variaba en relación a las distintas veces que corría el montecarlo con ese valor, mientras que el epsilon de 0.001 no solía variar mucho entre los distintos llamados.

**Conclusiones**:

Cuando utilizamos valores altos de epsilon como por ejemplo 0,1 obtenemos un resultado no tan preciso como cuando usamos un epsilon 0,01. Estos últimos valores de epsilon nos permiten acercarnos a una diferencia mucho más chica entre las diferentes probabilidades que estábamos calculando (actual y anterior) y realizan mucha más casos de test por lo que los valores que obtenemos en estos casos son más exactos y no varían tanto entre diferentes llamados al script.