Práctica 8
Modelo de urnas

Edson Edgardo Samaniego Pantoja **Materia:** Simulación computacional
Fecha 21 de abril de 2021

1. Introducción

La coalescencia y fragmentación se representa como partículas uniéndose para formar cúmulos y estos cúmulos se pueden volver a descomponer en fragmentos menores. Esto es relevante en muchos campos de química, como por ejemplo en el filtrado de aguas residuales, donde solamente los cúmulos de suficiente tamaño serán capturadas por el filtro y hay que buscar formas para facilitar que crezcan los cúmulos de residuos para lograr su filtrado adecuado.

2. Metodología

Cada código explicado en clase de la obtención paso por paso de los cúmulos y como manipularlos se pueden consultar en el repositorio de Schaeffer [2]. Como parte de la metodología utilizada y explicada puede ser vista en la pagina de Schaeffer [3].

Se tiene una cantidad de `n` partículas que al inicio el tamaño de los `k` cúmulos existentes sigue la distribución normal. Suponiendo que la mediana de los tamaños iniciales corresponde al tamaño crítico `c:`, cúmulos menores a `c` solamente pueden pegarse uno al otro y quedarse como son, pero tamaños $\geq c$ pueden además fragmentarse.

La fragmentación se discretiza de tal forma que si un cúmulo se rompe, siempre resulta en dos pedazos no vacíos, cuyos tamaños se determinan uniformemente al azar.

Para la probabilidad de que un cúmulo con menos de `c` partículas quiera unirse con otro, se realiza una curva de distribución exponencial, para que cúmulos muy pequeños quieran juntarse con más atracción, pero que sea más difícil cuando son de tamaños mayores.

3. Objetivo

Se hace la suposición de cúmulos con `c` o más partículas (haciendo referencia al tamaño crítico c) son suficientemente grandes para filtrar. Tal caso se requiere graficar para `k=1000`, $n \in \{16k, 32k, 64k, 128k\}$ en cada iteración el porcentaje de las partículas que se logra filtrar.

4. Simulación

La simulación modificada para la práctica puede ser consultada en el repositorio de Samaniego [1], los primeros cambios realizados son hacer el cambio de la variable `n` en un ciclo `for` que varía de `16k` hasta `128k` a partir de este ciclo ya comienza el resto de la programación.

```
k = 1000
porcentaje=[]
for n in (16000, 32000, 64000, 128000):
```

```
orig = np.random.normal(size = k)
cumulos = orig - min(orig)
cumulos += 1
cumulos = cumulos / sum(cumulos)
```

Antes del ciclo `for` que tiene duración de cincuenta iteraciones se declaran dos variables que acumularan datos de las partículas más grandes y la iteración en la que va.

```
grandes=[]
itera=[]
for paso in range(duracion):
```

En cada iteración se realiza un ciclo que revisa la lista de tamaño de `cumulos`, para después de esto tomar una decisión con la función `if`, donde sí el tamaño del cumulo en la variable `valor` es mayor a `c` (el valor crítico) entonces la variable `grandes` acumula este dato que es mayor a `c`.

```
for CML in range(total):
        valor=cumulos[CML]
        if valor > c:
            grandes.append(cumulos[CML])# acumula los mas grandes

    cantidad=len(cumulos)
    restantes= len(grandes)
    PR=((restantes*100)/cantidad)# porcentaje de filtrados
    porcentaje.append(PR) # se acumulan los porcentajes
    itera.append(paso)
    grandes.clear()
```

Mas adelante después del ciclo `if` pero dentro del mismo `for`, las variables `cantidad y restantes` van a tomar valor del número de datos que hay en `cumulos` y en `grandes` para poder obtener el porcentaje de cuántos cúmulos restantes son los que se retuvieron en el filtro. Este porcentaje se acumula en una lista ya declarada como `porcentaje`.

5. Resultados

La manera que se obtuvieron los resultados gráficos se pueden consultar en github [1]. La primer gráfica que se obtiene en la figura 1 se observa el comportamiento del porcentaje para `16k y 32k` por iteración y se marca en una linea vertical el punto más alto de cada gráfico.

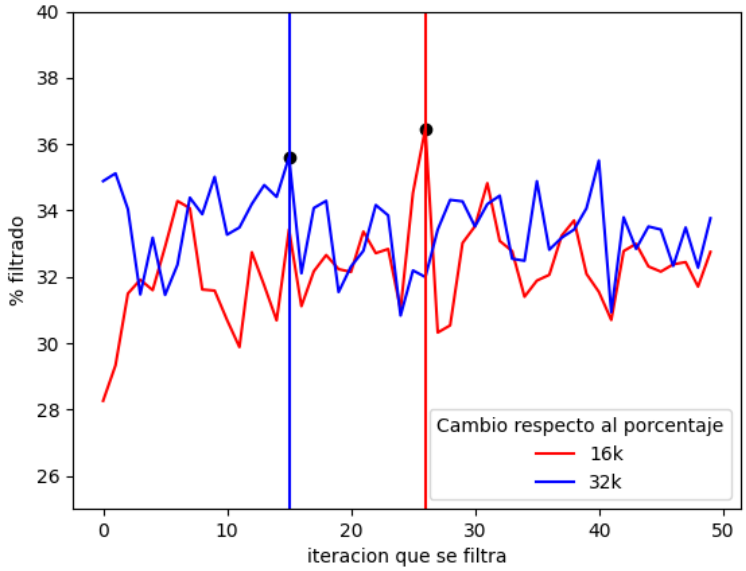

Figura 1: Gráfica de 16k y 32k de porcentaje filtrado por iteración.

En la figura 2 se obtiene el comportamiento para `64k y 128k`, se observa que el punto mas alto de porcentaje de cúmulos más grandes no tiene una relación de cambio con respecto a la figura 1 ya que los puntos dan al azar debido al comportamiento de partirse y aglomerarse de las partículas.

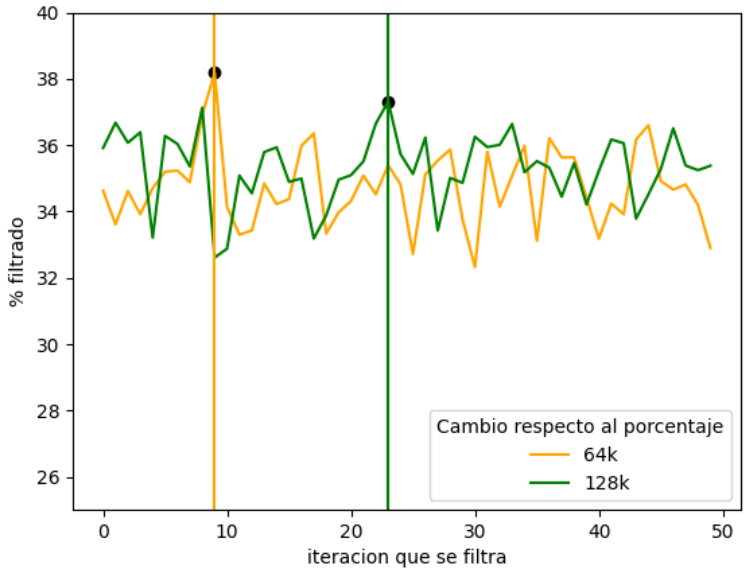

Figura 2: Gráfica de 64k y 128k de porcentaje filtrado por iteración.

6. Conclusión

En base a los resultados obtenidos se concluye que lo que se busca en ver como se comporta el porcentaje de partículas filtradas y ver su punto más alto no tiene una relación con respecto a la `n` que se varía debido a que los datos son muy aleatorios y generan diversos picos en la gráfica y los puntos altos están de igual manera aleatoriamente, lo que si tiene una cierta relación es que el porcentaje de las cuatro gráficas se mantuvo en rangos de entre veintiocho a treinta y ocho de porcentaje.

Referencias

[1] Samaniego E. Práctica 8, 2021. URL https://github.com/edson-samaniego/simulation-2021/tree/main/Pr%C3%A1ctica-8.

[2] Schaeffer E. simulación, 2021. URL https://github.com/satuelisa/Simulation/blob/master/UrnModel/aggrFrag.py.

[3] Schaeffer E. *Práctica 8*, 2021. URL https://elisa.dyndns-web.com/teaching/comp/par/p8.html.