SECOND MIDTERM

Nataly Phawllyn Neira Parra Cod: 614212782
Fundación Universitaria Konrad Lorenz

27 de octubre de 2024

Computación científica II

---

1. Pseudorandom number generators with linear structures.

consideremos la secuencia aleatoria de números

$$[137, 553, 990, 881, 646, 618, 323, 832, 897, 230, 181, 432, 44, 925, 525, 695, 367, 711, 974, 274, ...]$$

generados por un **LCG** con parámetros (m, a,c, x_0). Los números de la secuencia son generados por una ecuación de recurrencia.

$$[ax_n + c = x_{n+1}]modm$$

cuya semilla (x_0) es el primer parámetro, $x_0 = 137$.

Para encontrar el parámetro m, se divide la secuencia en arreglos de cuatro entradas de números consecutivos de la secuencia, cada parte es procesada por un núcleo en programación en paralelo (punto1.py). De cada arreglo $[a_0, a_1, a_2, a_3]$ se definen tres vectores $\alpha = (a_0, a_1)$, $\beta = (a_1, a_2)$, $\gamma = (a_2, a_3)$ y se calcula el volumen del paralelepípedo formado por dichos vectores. Con la lista de los volúmenes se calcula el máximo común divisor encontrando 3 valores que son los candidatos de m.$(33759, 1023, 1023)$

Para cada uno de los valores de m se soluciona con GNU parallel el sistema de ecuaciones (punto1CL.py)

$$[ax_0 + c = x_1]modm$$
$$[ax_1 + c = x_2]modm$$

Los resultados se comprueban re generando la secuencia y comprobando si es la misma. Se obtiene como resultado correcto $a = 997$, $c = 23$ para $m = 1023$.

```
root@DESKTOP-7JUCAKM:/mnt/c/Users/NATALY NEIRA/Documents/parcial2CCI# echo 33759
 1023 1023 | tr ' ' '\n' | parallel -j 3 python3 punto1CL.py {}
m=1023,x0=137,a=997,c=23
los parametros encontrados  NO son  correctos m=33759,x0=137,a=22480,c=26621
m=1023,x0=137,a=997,c=23
root@DESKTOP-7JUCAKM:/mnt/c/Users/NATALY NEIRA/Documents/parcial2CCI#
```

Figura 1: Solución de LCG

2. A simple Markov Chain Monte Carlo (MCMC)

1. Se escribe un programa para para generar cadenas de Markov con el algoritmo de Metropolis-Hastings (punto2.py) cuya función de distribución de probabilidad objetivo es la distribución escalada y desplazada de T-student.

2. Se corre el programa para una cadena de longitud $N = 10$ con distintos pasos de $\alpha = (0.01, 0.1, 1, 10)$ para la distribución propuesta, la distribución gaussiana. se gráfica la traza(punto2_traza.py) (Fig.2) de las cadenas de las cuales se puede inferir que los mejores valores de $\alpha = (1, 10)$, pues exploran un rango mayor del espacio sin quedar estancados en zonas de mayor probabilidad.

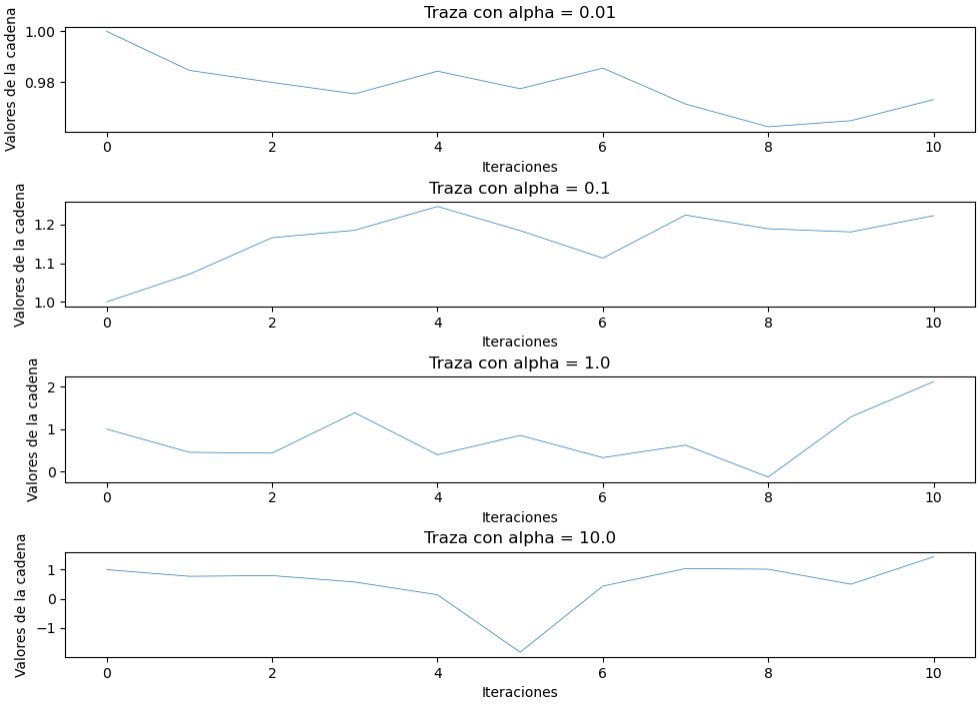

Figura 2: visualización de cadenas

3. Se genera una cadena mas grande $N = 1000$ para cada valor de α en GNU parallel, y se gráfica el histograma de los valores obtenidos (punto2_histograma.py)(Fig.3). Podemos ver que la cadena que se ajusta mejor a la distribución propuesta es para $\alpha = 1$.

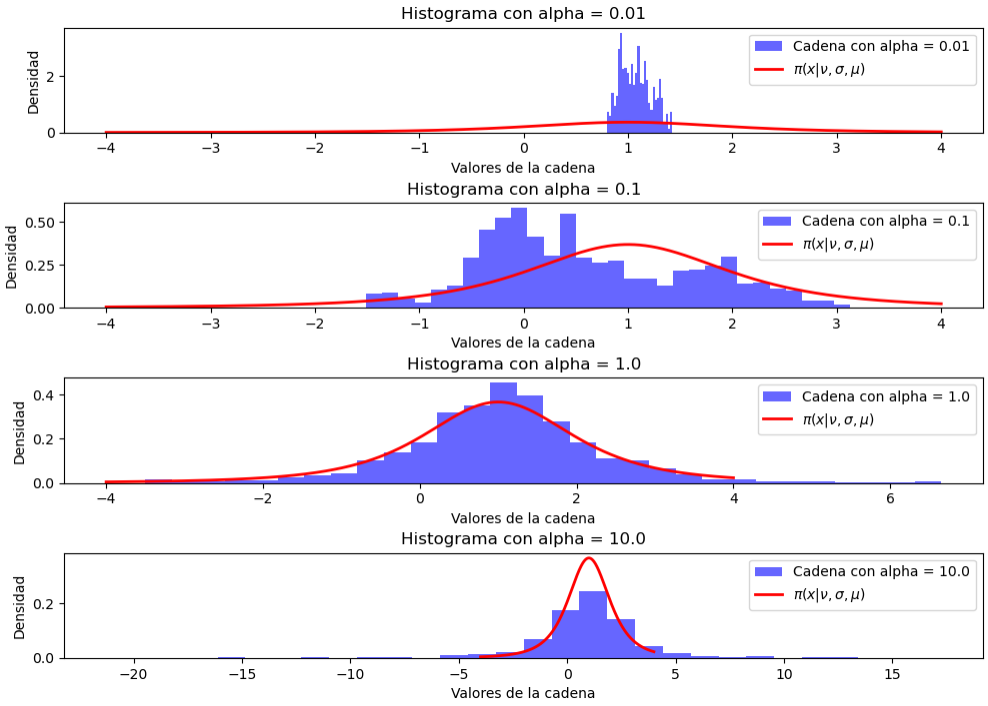

Figura 3: cadenas de Markov vs distribución T-student

4. Para cada *alpha* obtenido se calcula la fracción de aceptación (*F=valores aceptados/ valores propuestos*) obteniendo porcentajes muy altos para los alfas pequeños, y porcentajes muy pequeños para el alfa mas grande (fig.4)

```
la fracción de aceptación para 0.01 es de 99.60%
la fracción de aceptación para 0.1 es de 96.99%
la fracción de aceptación para 1.0 es de 73.37%
la fracción de aceptación para 10.0 es de 18.02%
```

Figura 4: Fracciones de aceptación

Se concluye que el valor mas optimo para generar la muestras aleatorias que se asemejan mas a la distribución de T-student es con $\alpha = 1$

3. Parameter estimation

1. Con $\alpha = 1$ se genera una cadena de $N = 100$ datos. Se escribe un programa (punto3.py) que lee dichos datos y crea una cadena de Markov para los parámetros ν, μ, δ , con una distribución objetivo el nominador del teorema de Bayes, que lo constituye la función de verosimilitud por la multiplicación de las distribuciones uniformes sobre cada parámetro en los intervalos dados. Se genera con GNU parallel cadenas de tripletas (ν,δ, μ) de tamaños $N = (10, 20, 50, 100, 500)$ para comprobar el funcionamiento del código.

2. Con la muestra de $N = 100$ y $N = 500$ se gráfica los histogramas marginales para cada parámetro (punto3_histograma.py) (Fig.5, Fig.6)

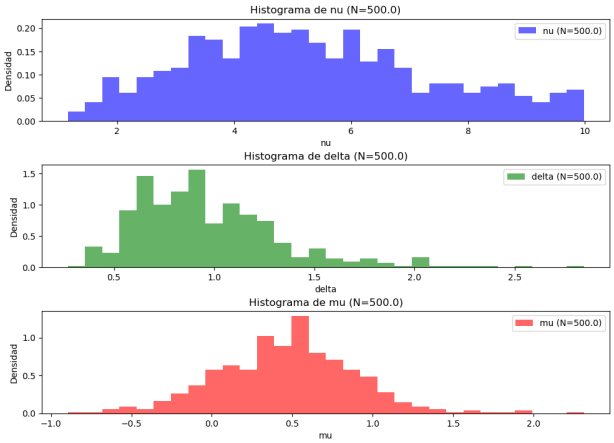

Figura 5: histogramas N=100

Los valores de los parámetros que se pueden inferir de la muestra $N = 100$ son: $\nu \approx 8$, $\delta \approx 0.8$, $\mu \approx 0.5$

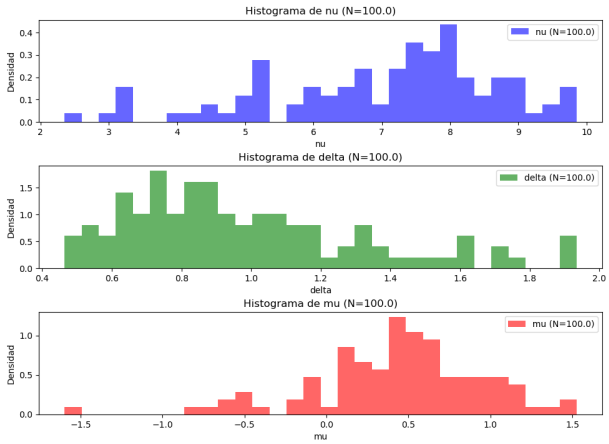

Figura 6: Histogramas N=500

Los valores de los parámetros que se pueden inferir de la muestra $N = 500$ son: $\nu \approx 5$, $\delta \approx 1$, $\mu \approx 0.5$

3. las gráficas de correlación de los parámetros nos indican (μ vs ν, δ vs ν, μ vs δ) no parece hacer una correlación significativa. es decir son parametros independientes.

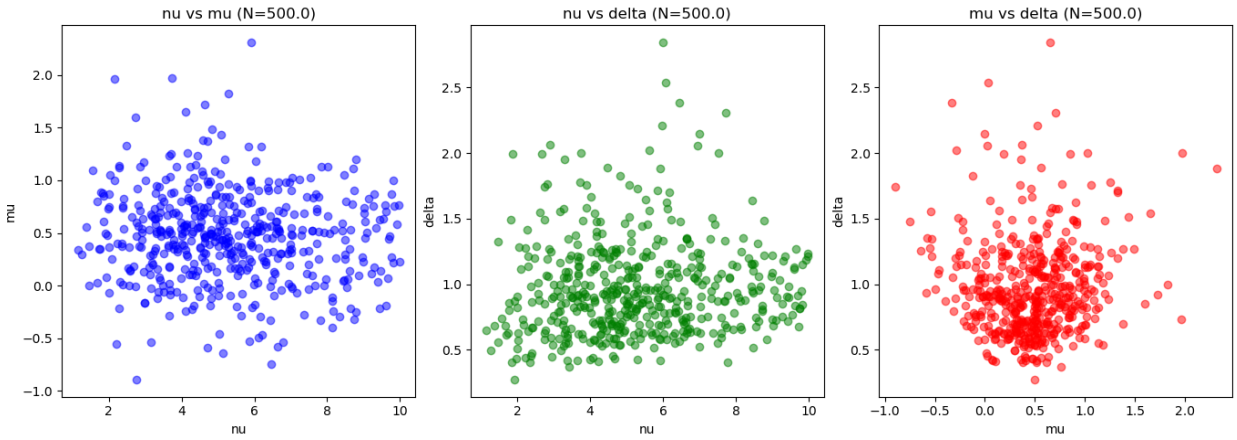

Figura 7: graficas de correlación

Nota: Los Scripts donde se solucionan los puntos se encuentran adjuntas, en archivo .zip donde se encontraba este documento.

.