



Tarea 2: *Simulación y CMTD*

Integrantes

- ◇ **Integrante 1:** Vicente Lavagnino (Sección 2)
- ◇ **Integrante 2:** Jacinta Ortiz (Sección 4)

Instrucciones

- ◇ Esta tarea debe ser realizada **individualmente** o **en parejas** (pueden ser de diferentes secciones dentro de las que son coordinadas). Deben indicar en la primera plana del PDF los nombres de los/as integrantes, junto con las secciones respectivas de c/u.
- ◇ Tienen plazo hasta el **Martes 01 de octubre a las 23:59 hrs** para entregar sus soluciones.
- ◇ Deben entregar un archivo reporte, **en formato PDF**. En el archivo PDF deben entregar sus respuestas de los 3 problemas. Los códigos implementados deben adjuntarse como un archivo .zip aparte en el buzón. En el PDF deberá estar el desarrollo, resumen de resultados y correspondiente análisis. Se recomienda fuertemente utilizar \LaTeX . Sin embargo, estas también pueden ser escritas en algún otro editor de texto o a mano y escaneadas, pero **debe estar ordenado y legible**, de lo contrario, se descontarán 3 décimas a la nota final de la tarea. Cualquier código que utilicen de internet debe ser debidamente citado insertando el *link* asociado.
- ◇ Si realizan la tarea completa en \LaTeX , incluyendo todas las fórmulas y ecuaciones, tendrán **3 décimas extras**.
- ◇ La tarea tiene un total de 60 puntos. La nota final será calculada como la suma de las tres preguntas mediante la fórmula

$$Nota = \frac{P_1 + P_2 + P_3}{60} \cdot 6 + 1$$

Donde P_i corresponde al puntaje de la pregunta i . Se aproxima a dos decimales.

- ◇ Las preguntas se harán a través del foro de discusión asociado a la Tarea 1 en Canvas, por lo que les pedimos que **NO** manden dudas por mail sobre la tarea. De esta manera, todos tienen acceso a las respuestas.
- ◇ Seremos estrictos en que **no se aceptarán envíos por mail**. Evite problemas de internet, del sistema Canvas, de la calidad de la foto o cualquier otro que pudiese ocurrir al entregar sobre la hora. El/La alumno/a que envíe su tarea después de la hora límite, tendrá **nota 1.0**.

Problema 1 (20 puntos)

Parte 1

a)

- i) Considerando todo el horizonte de tiempo, se tiene que la media y la varianza de la cantidad de personas en el hospital, son las siguientes:

$$\bar{X} = 448,633 \quad \hat{V} = 14507,717 \quad (1)$$

- ii) Si graficamos los datos (Anexo a), podemos notar que desde el dato n°60 en adelante la muestra se empieza a estabilizar y por lo tanto podemos decir que entre los datos (1:59) el hospital se encuentra en el periodo de warm-up. Esto tiene sentido ya que el enunciado menciona que el hospital está en un periodo de ajuste donde la cantidad de personas en el hospital se está regulando respecto a las tasas de llegada de pacientes y atención de estos. Es por eso que si calculamos las varianzas y medias sin el periodo de warm up tenemos :

$$\bar{X} = 498,97 \quad \hat{V} = 131,503 \quad (2)$$

Estos resultados nos dicen que el promedio no cambia considerablemente respecto a los resultados en todo el horizonte de tiempo, pero sí la varianza. Podemos notar la gran diferencia entre ambas varianzas y esto nos demuestra que la inclusión del periodo de warm up afectaría en la interpretación de los resultados en el largo plazo. Debido a que, durante el warm-up, el sistema no ha alcanzado aún un equilibrio, incluir este periodo en el análisis provocaría que los resultados puedan estar sesgados por condiciones iniciales transitorias.

b)

- iii) Con la muestra de datos se realizó un gráfico para evidenciar el comportamiento de los datos (Anexo b) y así dividir correctamente en grupos aproximadamente independientes entre sí. De esta forma, se dividió a la muestra en 24 grupos (b=24) de 60 observaciones cada uno (m=60). Obteniendo la media cada uno de los batches, al buscar el promedio de estas se obtuvo que $\bar{Y} = 0,078788795$ y la varianza de estas como $\hat{V}_B = 0,009574671$.
- iv) Para definir un intervalo de confianza del 95 % con b=24, m=60, $\bar{Y} = 0,0787$ y $\hat{V}_B = 0,00957$ reemplazamos estos valores en lo siguiente:

$$\bar{Y} \pm t_{b-1, 1-\alpha/2} \sqrt{\frac{V_B}{b}} \quad (3)$$

Y como $t_{23, 0.025} = 2.0687$ (Anexo c), resulta:

$$0.0787 \pm 2.0687 \sqrt{\frac{0.00957}{24}} \quad (4)$$

Por lo que el intervalo de confianza es [0.0374 ; 0.12]. Un intervalo relativamente amplio como el obtenido sugiere que existe una variabilidad significativa en el rendimiento diario del fondo. Esto significa que, aunque el rendimiento promedio es positivo, puede variar bastante de un día a otro, lo que indica cierta inestabilidad y por lo tanto, un riesgo para los inversores.

Parte 2

c)

En primer lugar, tenemos que un proceso estocástico es una Cadena de Markov en tiempo discreto si cumple la propiedad markoviana y la propiedad estacionaria. Es decir, debiera cumplirse que:

$$P(Z_{n+1} = j | Z_n = i, Z_{n-1} = i-1, \dots, Z_0 = i_0) = P(Z_{n+1} = j | Z_n = i) \quad (5)$$

$$P(Z_{n+1} = j | X_n = i) \text{ depende sólo de } i \text{ y } j \quad (6)$$

Dado que Z_i es el i -ésimo valor del evento notable, según el enunciado, Z_i cambiará de estado en el momento n sólo si X_n es mayor a todos los valores de X hasta el momento. Esto quiere decir, que Z_i sólo depende del

máximo inmediatamente anterior y no del resto de los valores por lo que se cumple la propiedad markoviana. Por otro lado, dado que el evento notable no depende del tiempo n sino únicamente del valor del máximo observado hasta ese momento. En cada etapa del proceso, el valor de Z_{i+1} depende únicamente del valor actual de Z_i y de la probabilidad de que el siguiente X_n sea mayor a ese valor. Por lo que se cumple la propiedad estacionaria y efectivamente $Z_i : i > 1$ es una Cadena de Markov de Tiempo Discreto.

Sus probabilidades de transición son:

$$P_{z_i, z_j} = 0 \quad \text{si } j < i \quad (7)$$

$$P_{z_i, z_j} = P(X_i = Z_j) = a_j \quad \text{si } j > i \quad (8)$$

$$P_{z_i, z_j} = P(Z_{i+1} \geq Z_i) = P(Z_{i+1} = Z_i) = 0 \quad \text{si } j = i \quad (9)$$

d)

Dado lo planteado por el enunciado, queremos demostrar que la matriz de tiempos de ocupación se puede representar como:

$$M(n) = \sum_{r=0}^n P^r \quad (10)$$

Por definición, $m_{i,j}(n)$ es la cantidad esperada de visitas de j comenzando desde el estado i hasta el tiempo n . La probabilidad de estar en el estado j en el tiempo r , comenzando desde el estado i , se da por la entrada (i, j) de la matriz de transición P^r , lo llamamos $((P^r)_{i,j})$.

Por lo tanto, el tiempo de ocupación $m_{i,j}(n)$ se puede calcular como la suma de las probabilidades de estar en el estado j desde el tiempo $r = 0$ hasta n :

$$m_{i,j}(n) = \sum_{r=0}^n (P^r)_{i,j} \quad (11)$$

Por lo tanto, la matriz de tiempos de ocupación $M(n)$ se puede expresar como la suma de las potencias de la matriz de transición P desde $r = 0$ hasta $r = n$:

$$M(n) = \sum_{r=0}^n P^r \quad (12)$$

Demostrando así que la matriz de tiempos de ocupación $M(n)$ se puede representar como la suma de las potencias de la matriz de transición P desde $r = 0$ hasta $r = n$

Problema 2 (20 puntos)

a)

En primer lugar, dada la política de inventario $(\{s_x, s_y\}, \{S_Y, S_X\})$ y X e Y siendo las cantidades de gomitas y huevos de pascua respectivamente, tenemos el siguiente espacio de estados:

$$EE = \{(X, Y) : s_x \leq X \leq S_X, s_y \leq Y \leq S_Y\} \quad (13)$$

Luego, si definimos a G como la demanda de bolsas de gomitas en 1 hora que distribuye Binomial(n, p) y a H como la demanda de huevitos de pascua en 1 hora que distribuye Uniforme($0, m$). Entonces tenemos :

$$X_t = \begin{cases} X_{t-1} - G_{t-1} & \text{si } X_{t-1} - G_{t-1} \geq s_x \\ S_x & \text{si } X_{t-1} - G_{t-1} < s_x \end{cases} \quad (14)$$

$$Y_t = \begin{cases} Y_{t-1} - H_{t-1} & \text{si } Y_{t-1} - H_{t-1} \geq s_y \\ S_y & \text{si } Y_{t-1} - H_{t-1} < s_y \end{cases} \quad (15)$$

El sistema puede modelarse como una Cadena de Markov en Tiempo Discreto (CMTD) porque cumple la propiedad markoviana y estacionaria. Por un lado, el inventario en la hora t depende únicamente del estado del inventario en la hora $t - 1$ y de las demandas de Gomas y Huevitos en la hora t , que son independientes entre sí. No se requiere información de horas anteriores a $t - 1$ para predecir el estado en t lo que cumple con la propiedad markoviana. Además, las probabilidades de transición entre los estados son constantes a lo largo del tiempo. Es decir, las demandas de Gomas y Huevitos en cada hora se distribuyen siempre de acuerdo a las mismas distribuciones (Binomial y Uniforme), y el proceso de reposición es el mismo en cada ciclo.

b)

Buscamos :

$$P_{(i,j),(k,l)} = P((X_t, Y_t) = (k, l) | (X_{t-1}, Y_{t-1}) = (i, j)) \quad (16)$$

Lo cual lo podemos expresar de la siguiente forma:

$$P_{(i,j),(k,l)} = P(X_t = k | X_{t-1} = i) \cdot P(Y_t = l | Y_{t-1} = j) \quad (17)$$

Además, definimos lo siguiente:

◇ Probabilidad de que la demanda de gomas sea λ :

$$p_G(\lambda) = P(G = \lambda)$$

◇ Probabilidad de que la demanda de gomas sea mayor o igual a λ :

$$P_G(\lambda) = P(G \geq \lambda)$$

◇ Probabilidad de que la demanda de huevitos sea λ :

$$p_H(\lambda) = P(H = \lambda)$$

◇ Probabilidad de que la demanda de huevitos sea mayor o igual a λ :

$$P_H(\lambda) = P(H \geq \lambda)$$

Ahora para encontrar las probabilidades de transición, debemos hacerlo por casos.

Casos de X, asumiendo que se parte de un estado $s_x \leq i \leq S_X$:

◇ Caso 1: $k = S_X$. Eso ocurre en cualquier caso tal que se demande al menos la cantidad que hace disminuir de la reposición s_x . Entonces, según lo definido anteriormente, buscamos $P_G(i - s_x)$, como la demanda de gomas distribuye Binomial(n,p), resulta:

$$P_G(i - S_X) = 1 - \sum_{j=0}^{i-S_X} \binom{n}{j} p^j (1-p)^{n-j}$$

◇ Caso 2: cualquier estado k, tal que $k \geq s_x$. En este caso no se necesita reposición por lo que buscamos la probabilidad de que la demanda sea justamente $i-k$. Es decir, buscamos $p_G(i - k)$:

$$p_G(i - k) = \binom{n}{i-k} p^{i-k} (1-p)^{n-(i-k)}$$

De manera análoga tenemos los casos de Y, asumiendo que se parte de un estado $s_y \leq j \leq S_Y$:

◇ Caso 3: $l = S_Y$. Eso ocurre en cualquier caso tal que se demande al menos la cantidad que hace disminuir de la reposición s_y . Entonces, según lo definido anteriormente, buscamos $P_H(j - s_y)$, como la demanda de huevitos de pascua distribuye Uniforme(0,m), resulta:

$$P_H(j - s_y) = 1 - \frac{j - s_y}{m}$$

◇ Caso 4: cualquier estado l, tal que $l \geq s_y$. En este caso no se necesita reposición por lo que buscamos la probabilidad de que la demanda sea justamente $j-l$. Es decir, buscamos $p_H(j - l)$:

$$p_H(j - l) = \frac{1}{m}$$

Por último, definimos las probabilidades de transición mediante la fórmula 14 y las probabilidades previamente definidas, de la siguiente manera:

$$P_{(i,j),(k,l)} = \begin{cases} 1 - \sum_{j=0}^{i-S_X} \binom{n}{j} p^j (1-p)^{n-j} \cdot (1 - \frac{j-s_y}{m}), & \text{si } k = S_X, l = S_Y \\ \binom{n}{i-k} p^{i-k} (1-p)^{n-(i-k)} \cdot (\frac{1}{m}), & \text{si } k \geq s_X, l \geq s_y \\ \binom{n}{i-k} p^{i-k} (1-p)^{n-(i-k)} \cdot (1 - \frac{j-S_Y}{m}), & \text{si } k \geq S_X, l = S_Y \\ 1 - \sum_{j=0}^{i-S_X} \binom{n}{j} p^j (1-p)^{n-j} \cdot (\frac{1}{m}), & \text{si } k = S_X, l \geq S_Y \end{cases}$$

c)

El estado en el que el inventario de gomitas está lleno, es decir, $X = S_x$, es un estado recurrente positivo, debido a que es un problema de inventario con capacidades máximas, y porque siempre se regresa a este estado cada vez que el inventario cae por debajo de s_x y es repuesto. El mismo comportamiento ocurre con los huevitos de pascua; el estado $Y = S_y$ también es recurrente positivo porque, cada vez que la cantidad de huevitos cae por debajo de s_y , se repone el inventario a S_y . Esto implica que el estado (S_x, S_y) , donde ambos inventarios están en sus niveles máximos, es recurrente en el sistema.

En contraste, los estados donde el inventario de gomitas o huevitos de pascua está por encima de s_x o s_y , pero por debajo de sus niveles máximos, son transitorios. Esto se debe a que, eventualmente, las demandas disminuirán el inventario por debajo de s_x o s_y , lo que activará la reposición y llevará al sistema de vuelta a (S_x, S_y) .

d)

En este caso, se realizó la matriz de probabilidades de transición mediante código de Python (Anexo) y se traspaso a Excel para visualizar:

| Estados | (1, 2) | (1, 3) | (1, 4) | (1, 5) | (2, 2) | (2, 3) | (2, 4) | (2, 5) | (3, 2) | (3, 3) | (3, 4) | (3, 5) |
|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| (1, 2) | 0,03125 | 0,03125 | 0,03125 | 0,03125 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| (1, 3) | 0,03125 | 0,03125 | 0,03125 | 0,03125 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0,75 |
| (1, 4) | 0,03125 | 0,03125 | 0,03125 | 0,03125 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0,5 |
| (1, 5) | 0,03125 | 0,03125 | 0,03125 | 0,03125 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0,25 |
| (2, 2) | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,03125 | 0,03125 | 0,03125 | 0,03125 | 0 | 0 | 0 | 1 |
| (2, 3) | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,03125 | 0,03125 | 0,03125 | 0,03125 | 0 | 0 | 0 | 0,75 |
| (2, 4) | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,03125 | 0,03125 | 0,03125 | 0,03125 | 0 | 0 | 0 | 0,5 |
| (2, 5) | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,03125 | 0,03125 | 0,03125 | 0,03125 | 0 | 0 | 0 | 0,25 |
| (3, 2) | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,03125 | 0,03125 | 0,03125 | 0,875 |
| (3, 3) | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,03125 | 0,03125 | 0,03125 | 0,65625 |
| (3, 4) | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,03125 | 0,03125 | 0,03125 | 0,4375 |
| (3, 5) | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,09375 | 0,03125 | 0,03125 | 0,03125 | 0,21875 |

e)

Buscamos $T_{(3,4)}^{(6)}$. Mediante código de python (Anexo) se calculó la probabilidad de retorno en 6 pasos resultando : 0.02007.

f)

Buscamos:

$$E(T(\{2,2\}, \{3,4\})) \quad (18)$$

Esta ecuación la podemos escribir como:

$$1 + \sum_{l \neq \{3,4\}} P_{\{2,2\},l} \cdot E(T(l, \{3,4\})) \quad (19)$$

Mediante código de python (Anexo) se obtuvo que el valor esperado de horas que transcurrieran para que Horace tenga 3 bolsas de gomitas Acidas y 4 paquetes de Huevitos de Pascua son 85.242 horas.

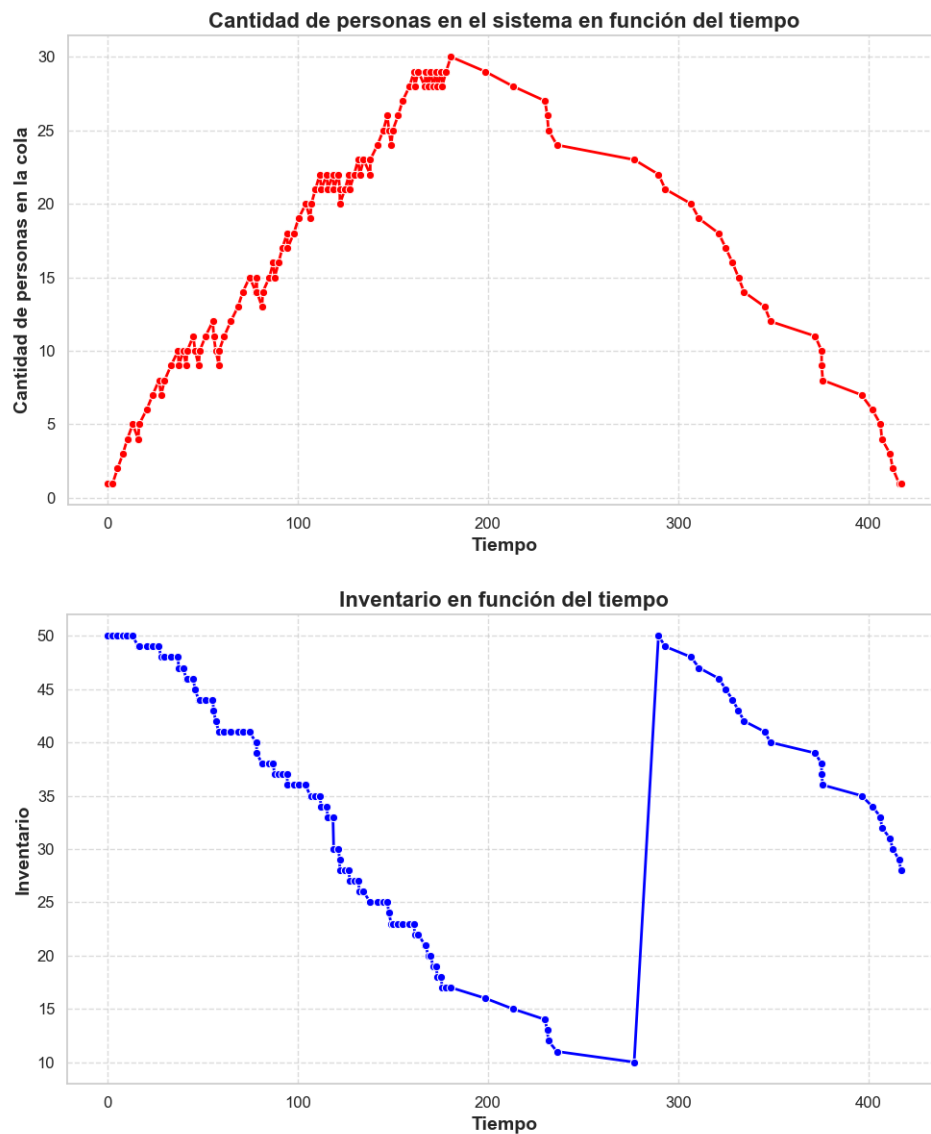
Problema 3 (20 puntos)

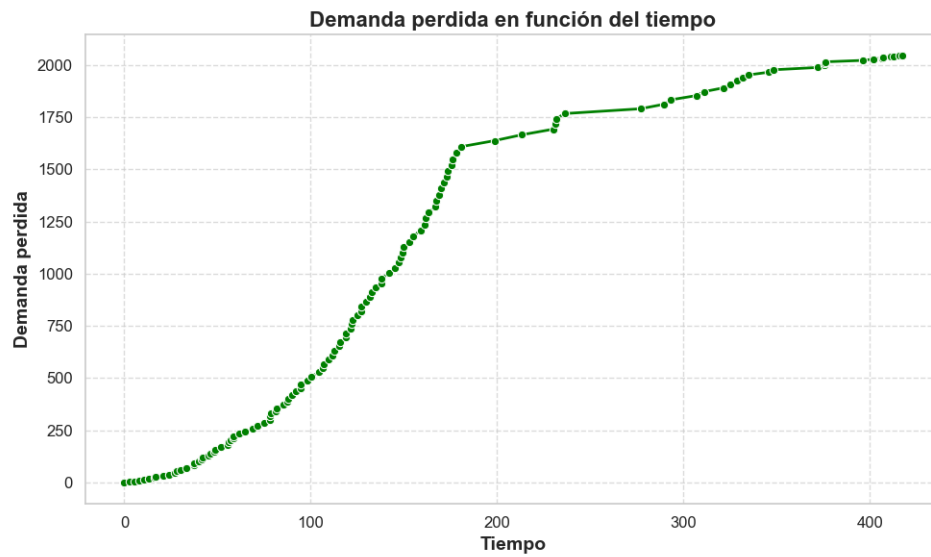
a)

Para el desarrollo de este ejercicio analizaremos cada intervalo de tiempo - distribución por separado para ver más detalladamente el comportamiento del inventario.

Horario entre las 08:00 y 11:00

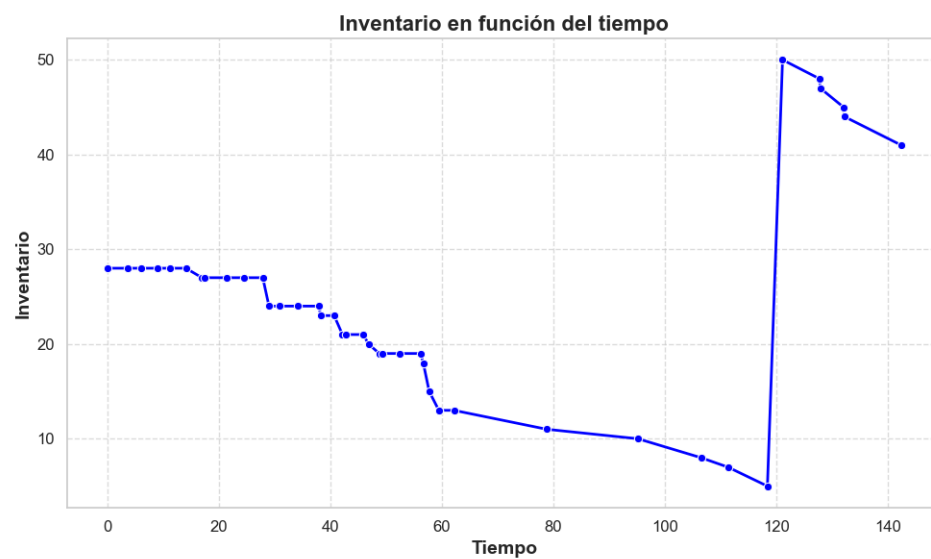
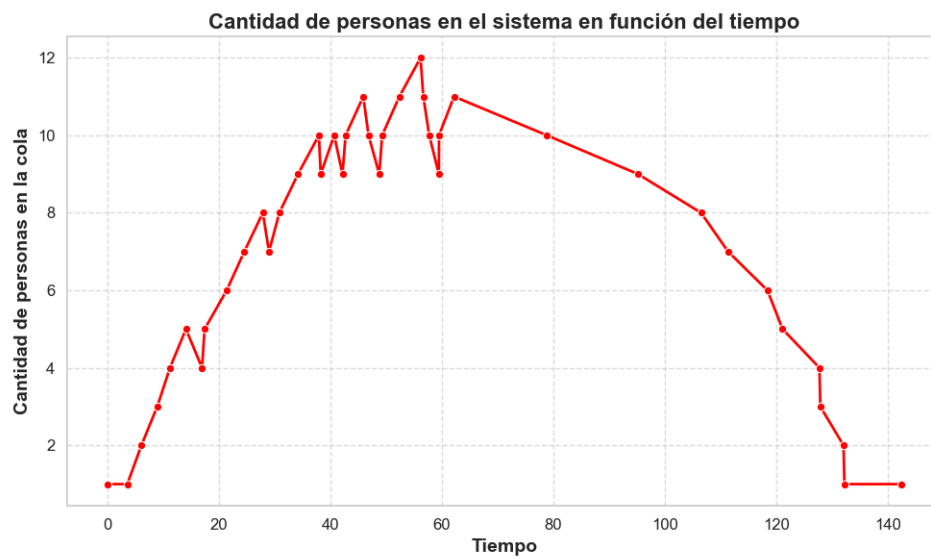
$n = 4, p = 0.1, \text{Uniforme}(2, 4)$

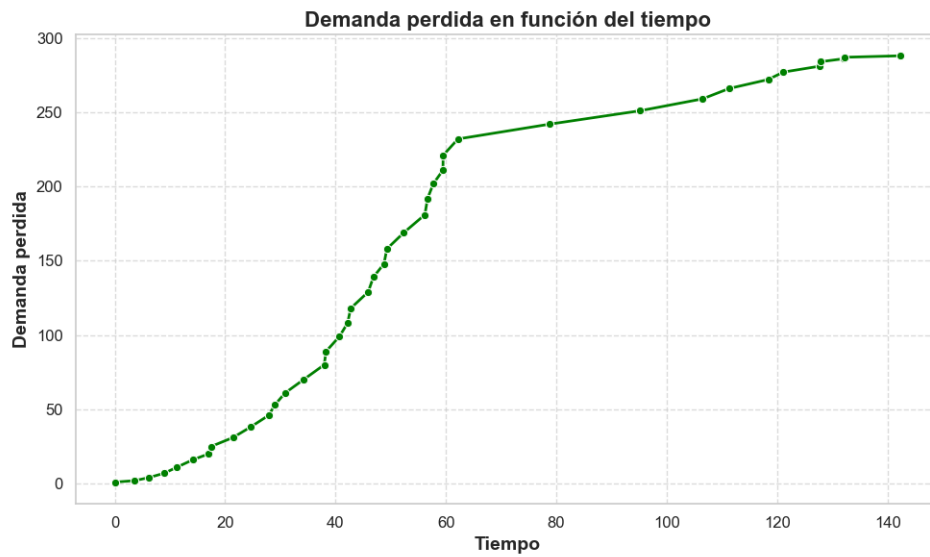




Ya que nuestro stock final es de 28, iniciaremos con eso para la siguiente fase.

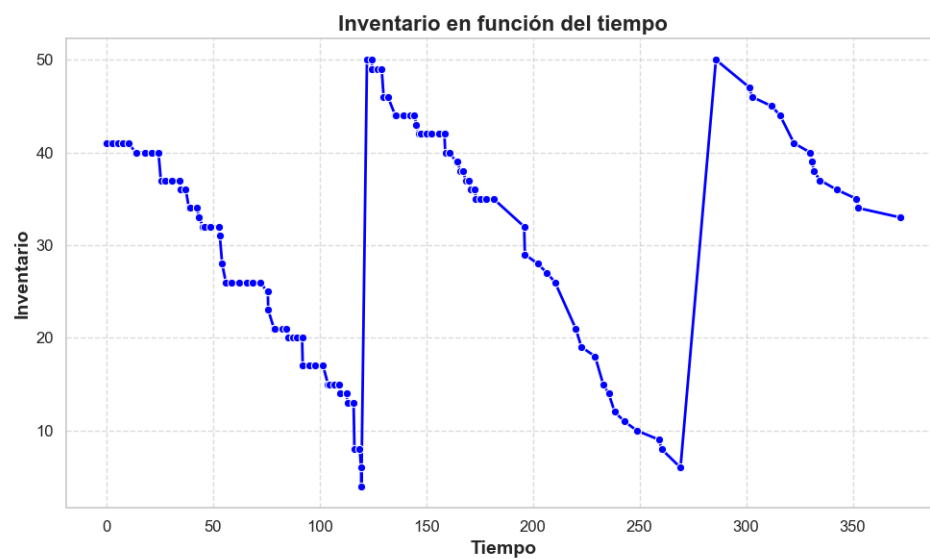
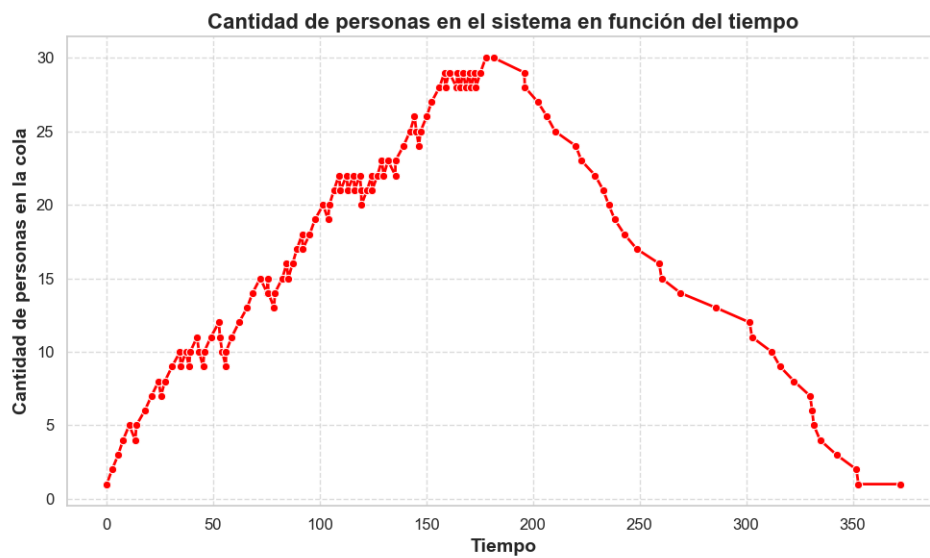
Horario entre las 11:00 y 12:00
 $n = 7, p = 0.2, \text{Uniforme}(2, 4)$

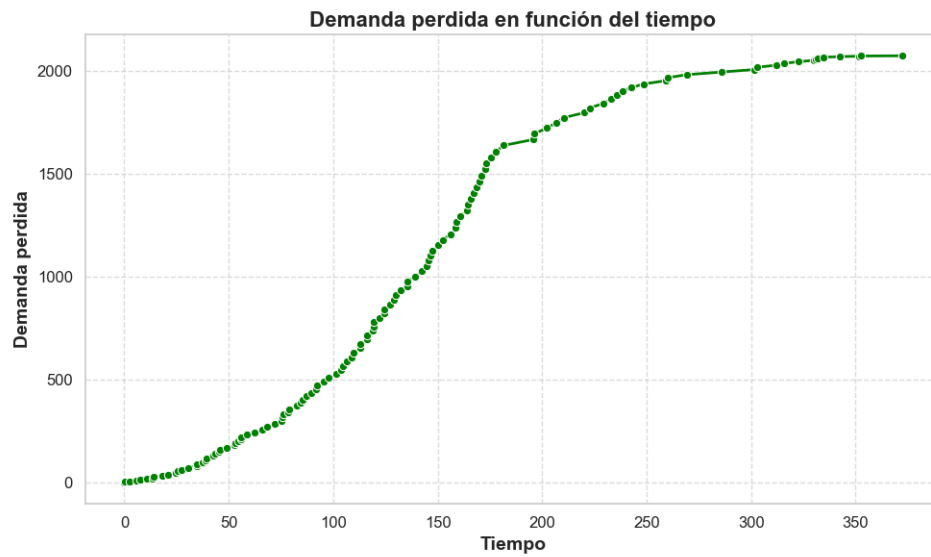




Ya que nuestro stock final es de 41, iniciaremos con eso para la siguiente fase.

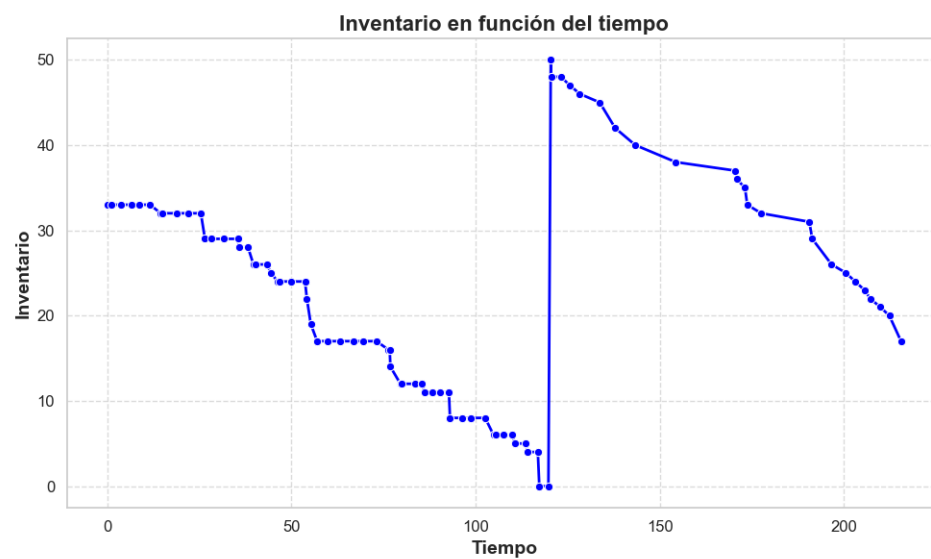
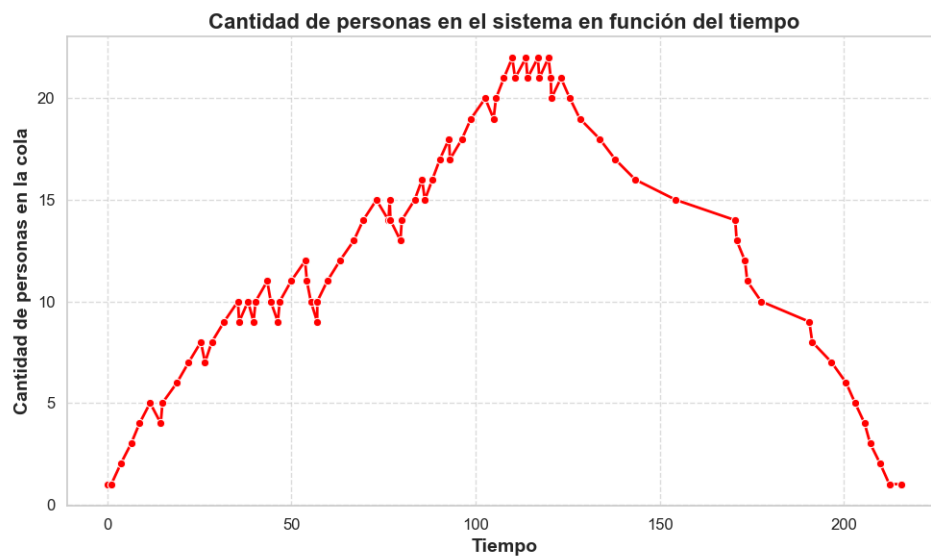
Horario entre las 12:00 y 15:00
 $n = 7, p = 0.2, Exponencial(3)$

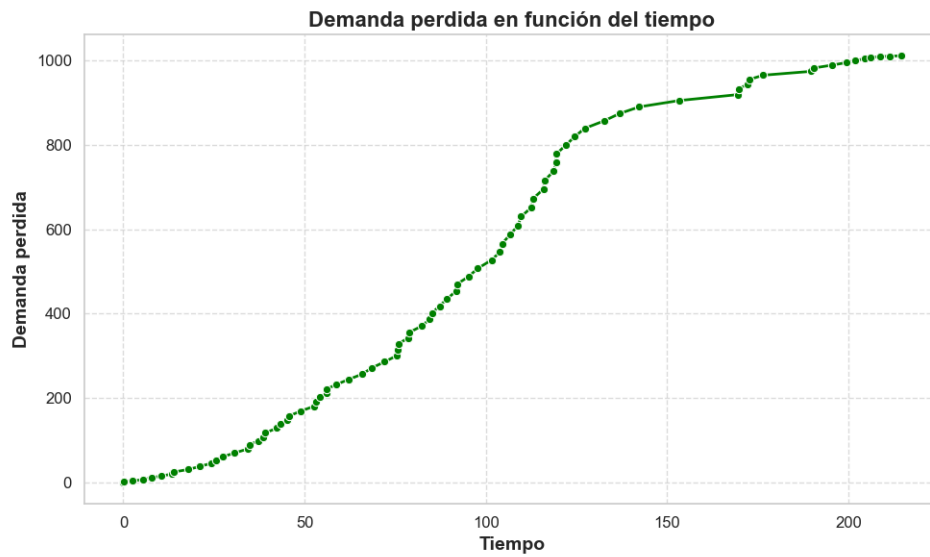




Ya que nuestro stock final es de 33, iniciaremos con eso para la siguiente fase.

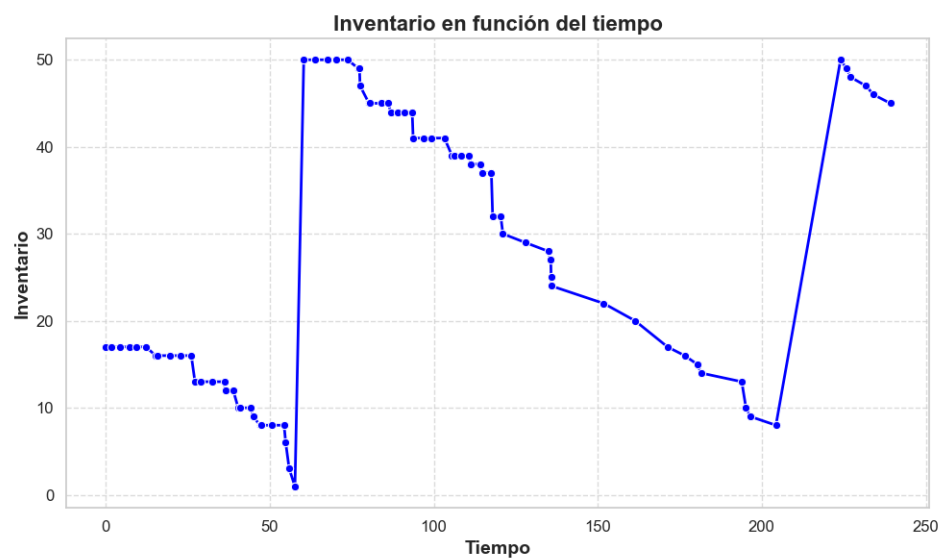
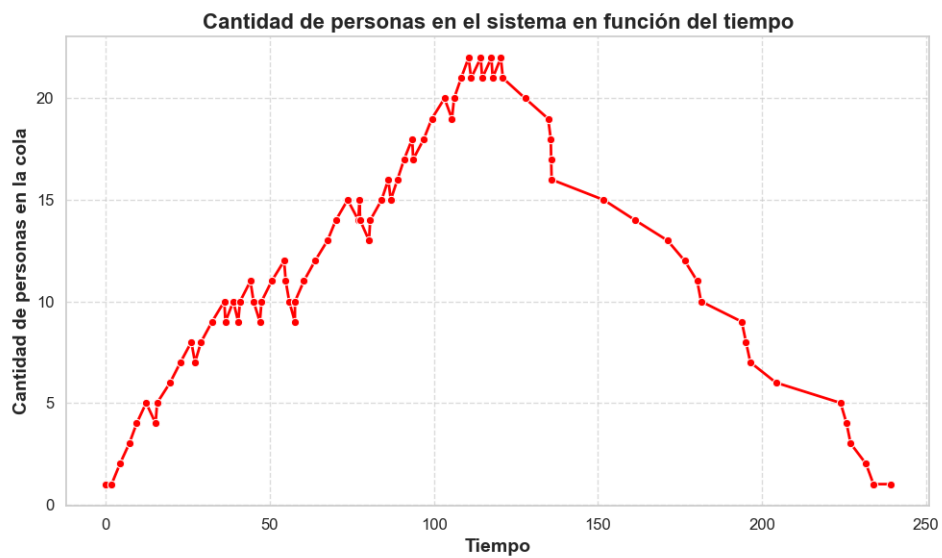
Horario entre las 15:00 y 17:00
 $n = 5, p = 0.3, Exponencial(3)$

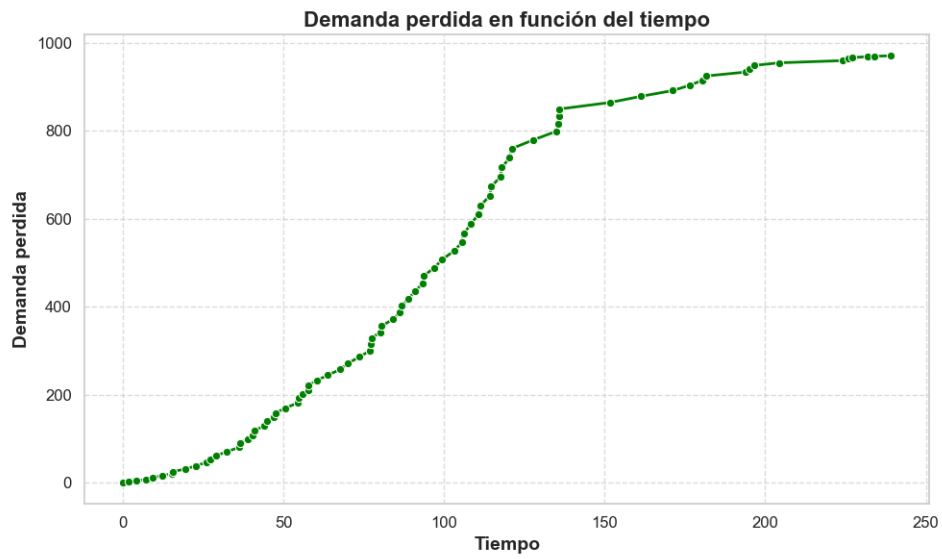




Ya que nuestro stock final es de 17, iniciaremos con eso para la siguiente fase.

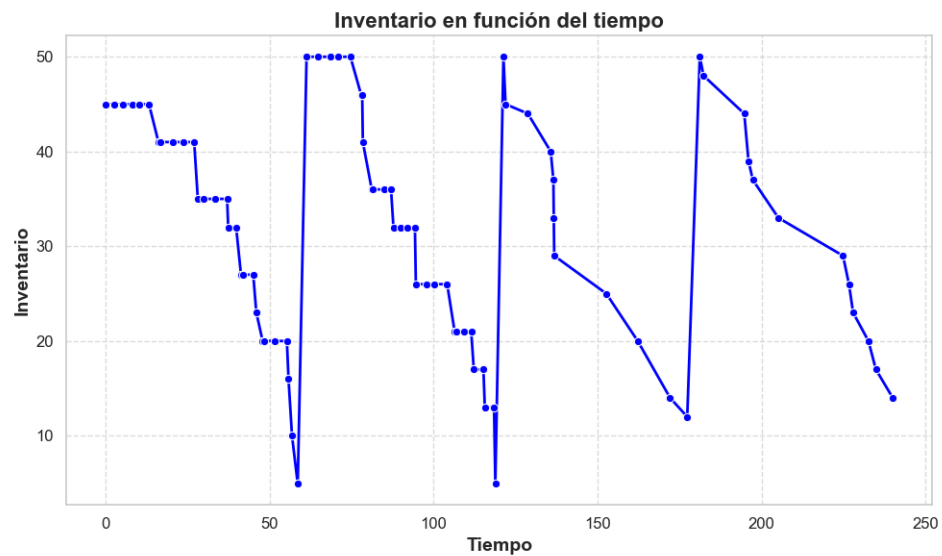
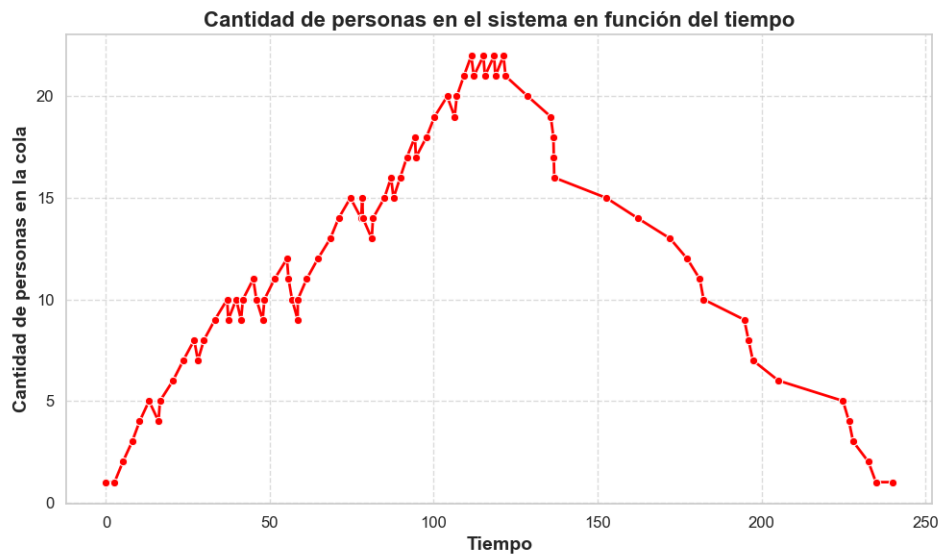
Horario entre las 17:00 y 19:00
 $n = 5, p = 0.3, \text{Rayleigh}(3)$

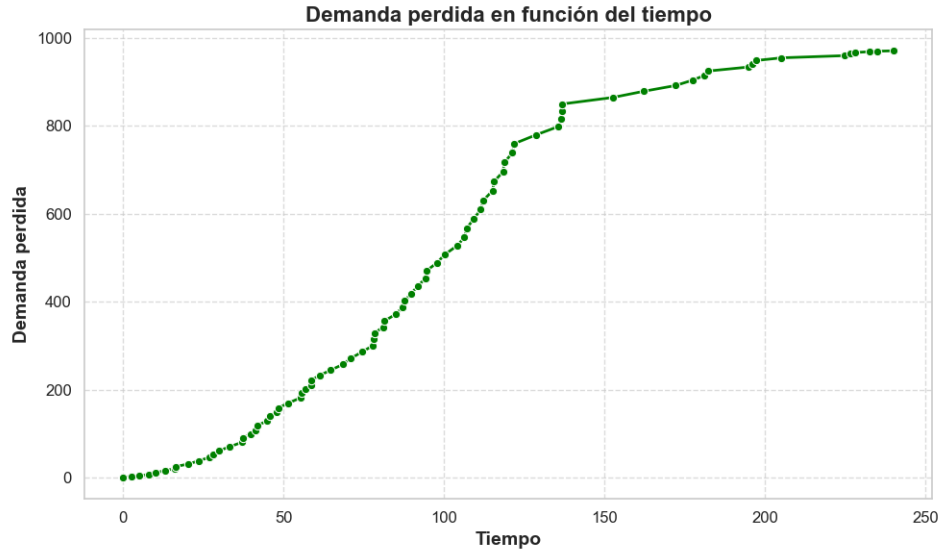




Ya que nuestro stock final es de 45, iniciaremos con eso para la siguiente fase.

Horario entre las 19:00 y 00:00
 $n = 8, p = 0.5, \text{Rayleigh}(3)$





Quedando con un stock final de 14.

b)

Las políticas evaluadas son las siguientes:

- ◊ **Política Base** $(s, S, \tau, K) = (10, 50, 30, 30)$
- ◊ **Política Alternativa** $(s, S, \tau, K) = (20, 45, 60, 15)$

Se llevaron a cabo 100 simulaciones por réplica, y un total de 100 réplicas para cada política. A continuación, se presenta el análisis de los resultados.

Ganancias promedio

- ◊ **Política Base:** La ganancia promedio obtenida para la política base fue de $\bar{X}_{\text{base}} = 15000.0$.
- ◊ **Política Alternativa:** La ganancia promedio obtenida para la política alternativa fue de $\bar{X}_{\text{alt}} = 13500.0$.
- ◊ **Política Base:** El intervalo de confianza para la política base es $IC_{\text{base}} = (15000.0, 15000.0)$.
- ◊ **Política Alternativa:** El intervalo de confianza para la política alternativa es $IC_{\text{alt}} = (13500.0, 13500.0)$.

Diferencia de ganancias promedio

La diferencia de las ganancias promedio entre la política alternativa y la política base se denota como:

$$\Delta X = \bar{X}_{\text{alt}} - \bar{X}_{\text{base}}$$

El valor de ΔX obtenido es $\Delta X = -1500.0$.

Intervalo de Confianza del 95 % para la diferencia de las ganancias promedio

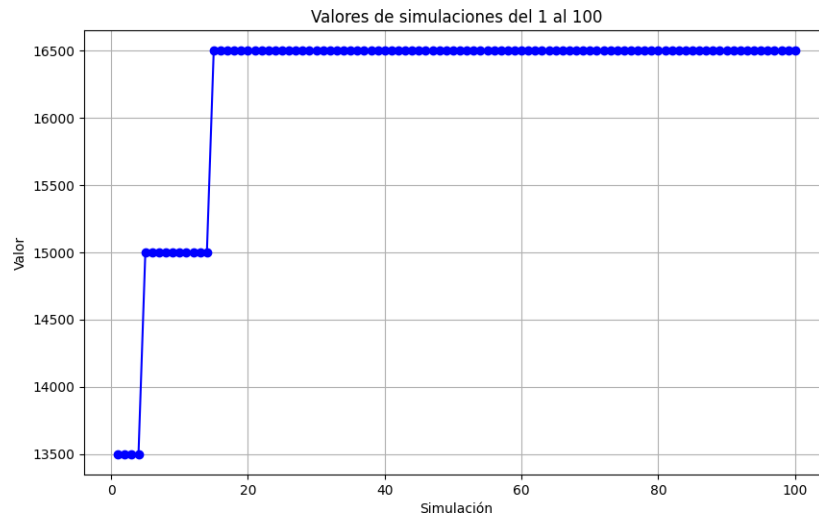
El intervalo de confianza del 95 % para la diferencia de las ganancias promedio es:

$$IC_{\Delta X} = (-1500.0, -1500.0)$$

Este intervalo de confianza es un número cerrado, lo que implica que las réplicas no tienen variabilidad en las ganancias estimadas.

c)

En este caso vemos que de las 100 simulaciones, obtenemos el valor más cercano desde el caso 14 y desde ahí en adelante se mantiene como el mejor resultado.



d)

Para disminuir en el sesgo de esta implementación, nos tenemos que enfocar en ciertas áreas claves:

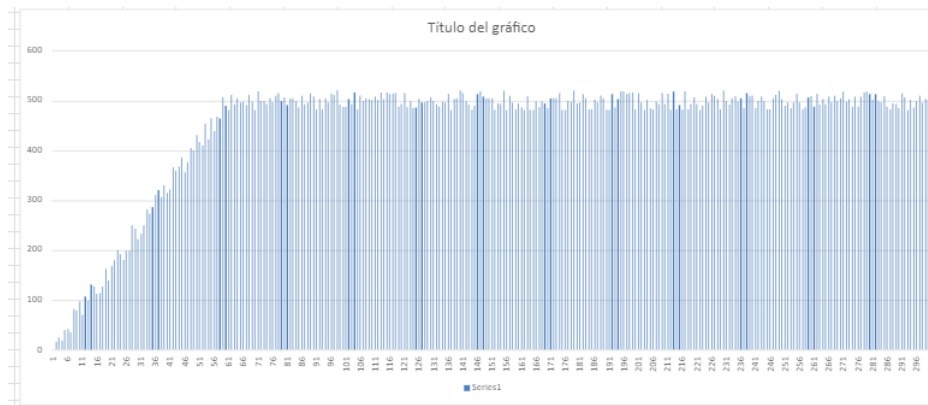
Asegurar la aleatoriedad adecuada, utilizar semillas diferentes en cada ejecución del código, puede reducir este tipo de sesgo.

Mayor número de réplicas, incrementar la cantidad de réplicas de simulación realizadas es fundamental para mejorar la precisión de los intervalos de confianza.

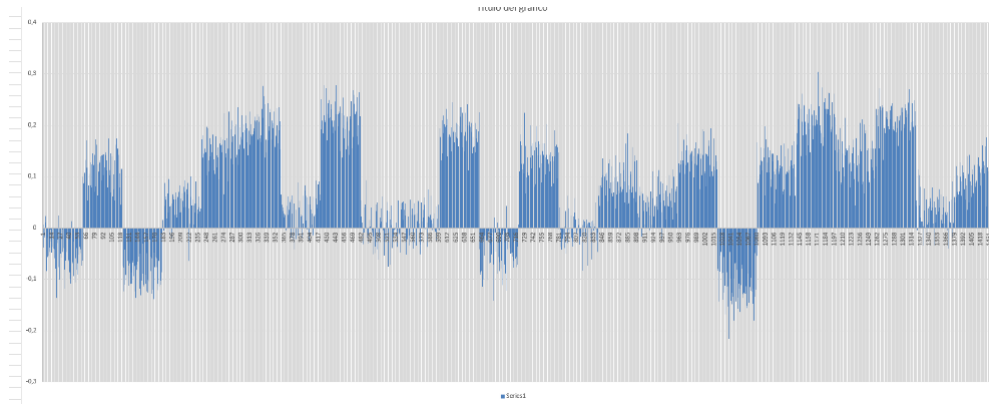
Además, revisar los supuestos clave del modelo (como la capacidad del sistema, la distribución de la demanda, y los tiempos de servicio) es esencial para asegurarse de que no haya errores o simplificaciones excesivas y lejanas a la realidad.

Anexo

(a) Gráfico Problema 1 Parte 1 a




(b) Gráfico Problema 1 parte 1 b



(c) Tabla T-Student

Tabla t-Student



| Grados de libertad | 0.25 | 0.1 | 0.05 | 0.025 | 0.01 | 0.005 |
|--------------------|--------|--------|--------|---------|---------|---------|
| 1 | 1.0000 | 3.0777 | 6.3137 | 12.7062 | 31.8210 | 63.6599 |
| 2 | 0.8165 | 1.8856 | 2.9200 | 4.3027 | 6.9645 | 9.2500 |
| 3 | 0.7649 | 1.6377 | 2.3534 | 3.1824 | 4.5407 | 5.8409 |
| 4 | 0.7407 | 1.5332 | 2.1318 | 2.7765 | 3.7469 | 4.8041 |
| 5 | 0.7267 | 1.4759 | 2.0150 | 2.5708 | 3.3649 | 4.0221 |
| 6 | 0.7176 | 1.4368 | 1.9432 | 2.4469 | 3.1427 | 3.7074 |
| 7 | 0.7111 | 1.4149 | 1.8946 | 2.3646 | 2.9979 | 3.4995 |
| 8 | 0.7064 | 1.3984 | 1.8605 | 2.3060 | 2.8965 | 3.3554 |
| 9 | 0.7027 | 1.3850 | 1.8331 | 2.2622 | 2.8214 | 3.2498 |
| 10 | 0.6998 | 1.3722 | 1.8125 | 2.2311 | 2.7638 | 3.1693 |
| 11 | 0.6974 | 1.3604 | 1.7959 | 2.2070 | 2.7181 | 3.1058 |
| 12 | 0.6955 | 1.3502 | 1.7823 | 2.1788 | 2.6810 | 3.0545 |
| 13 | 0.6938 | 1.3402 | 1.7709 | 2.1604 | 2.6503 | 3.0123 |
| 14 | 0.6924 | 1.3340 | 1.7613 | 2.1448 | 2.6245 | 2.9768 |
| 15 | 0.6912 | 1.3308 | 1.7531 | 2.1315 | 2.6025 | 2.9467 |
| 16 | 0.6901 | 1.3286 | 1.7459 | 2.1199 | 2.5835 | 2.9206 |
| 17 | 0.6892 | 1.3264 | 1.7396 | 2.1098 | 2.5669 | 2.8982 |
| 18 | 0.6884 | 1.3244 | 1.7341 | 2.1009 | 2.5524 | 2.8784 |
| 19 | 0.6876 | 1.3227 | 1.7291 | 2.0930 | 2.5395 | 2.8609 |
| 20 | 0.6870 | 1.3213 | 1.7247 | 2.0860 | 2.5280 | 2.8453 |
| 21 | 0.6864 | 1.3202 | 1.7207 | 2.0796 | 2.5176 | 2.8314 |
| 22 | 0.6858 | 1.3192 | 1.7171 | 2.0739 | 2.5083 | 2.8188 |
| 23 | 0.6853 | 1.3185 | 1.7139 | 2.0687 | 2.4999 | 2.8073 |
| 24 | 0.6848 | 1.3178 | 1.7109 | 2.0639 | 2.4922 | 2.7970 |
| 25 | 0.6844 | 1.3173 | 1.7081 | 2.0595 | 2.4851 | 2.7874 |
| 26 | 0.6840 | 1.3169 | 1.7056 | 2.0555 | 2.4786 | 2.7787 |
| 27 | 0.6837 | 1.3167 | 1.7033 | 2.0518 | 2.4727 | 2.7707 |
| 28 | 0.6834 | 1.3165 | 1.7011 | 2.0484 | 2.4671 | 2.7633 |
| 29 | 0.6832 | 1.3164 | 1.6991 | 2.0452 | 2.4620 | 2.7564 |
| 30 | 0.6829 | 1.3164 | 1.6973 | 2.0423 | 2.4573 | 2.7500 |
| 31 | 0.6825 | 1.3165 | 1.6955 | 2.0395 | 2.4528 | 2.7440 |
| 32 | 0.6822 | 1.3166 | 1.6939 | 2.0369 | 2.4487 | 2.7385 |
| 33 | 0.6820 | 1.3167 | 1.6924 | 2.0345 | 2.4448 | 2.7333 |
| 34 | 0.6818 | 1.3169 | 1.6909 | 2.0322 | 2.4411 | 2.7284 |
| 35 | 0.6816 | 1.3170 | 1.6895 | 2.0301 | 2.4377 | 2.7238 |
| 36 | 0.6814 | 1.3170 | 1.6883 | 2.0281 | 2.4345 | 2.7195 |
| 37 | 0.6812 | 1.3170 | 1.6871 | 2.0262 | 2.4314 | 2.7154 |
| 38 | 0.6810 | 1.3170 | 1.6860 | 2.0244 | 2.4286 | 2.7116 |
| 39 | 0.6808 | 1.3170 | 1.6849 | 2.0227 | 2.4258 | 2.7079 |
| 40 | 0.6807 | 1.3171 | 1.6839 | 2.0211 | 2.4233 | 2.7045 |
| 41 | 0.6805 | 1.3172 | 1.6829 | 2.0195 | 2.4208 | 2.7012 |
| 42 | 0.6804 | 1.3172 | 1.6820 | 2.0181 | 2.4185 | 2.6981 |
| 43 | 0.6802 | 1.3172 | 1.6811 | 2.0167 | 2.4163 | 2.6951 |
| 44 | 0.6801 | 1.3172 | 1.6802 | 2.0154 | 2.4141 | 2.6923 |
| 45 | 0.6800 | 1.3172 | 1.6794 | 2.0141 | 2.4121 | 2.6896 |
| 46 | 0.6799 | 1.3172 | 1.6787 | 2.0129 | 2.4102 | 2.6870 |
| 47 | 0.6797 | 1.3172 | 1.6779 | 2.0117 | 2.4083 | 2.6846 |
| 48 | 0.6796 | 1.3172 | 1.6772 | 2.0106 | 2.4066 | 2.6822 |
| 49 | 0.6795 | 1.3172 | 1.6766 | 2.0096 | 2.4049 | 2.6800 |

(d) Problema 2 d)

```

sx, sy = 1, 2
Sx, Sy = 3, 5
n, m, p = 3, 4, 0.5

# Función para calcular las probabilidades
def P_trans(i, j, k, l):
    if k == Sx and l == Sy:
        return (1 - np.sum([comb(n, j) * p**j * (1 - p)**(n - j) for j in range(i - Sx + 1)])) * (1 - (j - sy) / m)
    elif k == Sx and l == Sy:
        return comb(n, i - k) * p**(i - k) * (1 - p)**(n - (i - k)) * (1 / m)
    elif k == Sx and l == Sy:
        return comb(n, i - k) * p**(i - k) * (1 - p)**(n - (i - k)) * (1 - (j - sy) / m)
    elif k == Sx and l == Sy:
        return (1 - np.sum([comb(n, j) * p**j * (1 - p)**(n - j) for j in range(i - Sx + 1)])) * (1 / m)

# Crear las combinaciones de estados solo dentro de los rangos de x=1, Sx=2, Sy=3, Sy=5
states = [(i, j) for i in range(sx, Sx + 1) for j in range(sy, Sy + 1)]
num_states = len(states)

# Crear la matriz de transición
transition_matrix = np.zeros((num_states, num_states))

# Llenar la matriz de transición
for index, (i, j) in enumerate(states):
    for target_index, (k, l) in enumerate(states):
        transition_matrix[index, target_index] = P_trans(i, j, k, l)

# Crear un DataFrame para visualizar la matriz de transición
transition_df = pd.DataFrame(transition_matrix, index=states, columns=states)
import ace_tools as tools; tools.display_dataframe_to_user(name="Matriz de Probabilidades de Transición (Ajustada)", dataframe=transition_df)
file_path = "C:\\Users\\jacin\\OneDrive\\Escritorio\\python\\matriz_transicion_ajustada.csv"
transition_df.to_csv(file_path)

print(f"Matriz de transición guardada en: {file_path}")

file_path_csv = r"C:\\Users\\jacin\\OneDrive\\Escritorio\\python\\matriz_transicion_ajustada.csv"
df = pd.read_csv(file_path_csv)

```

(e) Problema 2 e)

```

# Crear la matriz de transición a la posición 6
P_6 = np.linalg.matrix_power(transition_matrix, 6)

# Definir el estado inicial (3, 4)
initial_state_index = states.index((3, 4))

# Probabilidad de volver al estado (3, 4) después de 6 horas
prob_6_hours = P_6[initial_state_index, initial_state_index]

# Mostrar el resultado
print(f"La probabilidad de estar en el estado (3, 4) después de 6 horas es: {prob_6_hours}")

```

(f) Problema 2 f)

```

states = [(1, 2), (1, 3), (1, 4), (1, 5), (2, 2), (2, 3), (2, 4), (2, 5), (3, 2), (3, 3), (3, 4), (3, 5)]
target_state = (3, 4)
initial_state = (2, 2)

# Índices de los estados
state_indices = {state: i for i, state in enumerate(states)}

# Matriz de probabilidades de transición (que ya tienes)
# Usamos 'transition_matrix' que ya has generado antes

# Tiempos de primer paso para cada estado (inicialmente indefinidos)
T = np.zeros(len(states))

# Configurar el sistema de ecuaciones excluyendo el estado objetivo
A = np.zeros((len(states) - 1, len(states) - 1))
b = np.ones(len(states) - 1) # El término constante en cada ecuación es 1

# Llenar el sistema de ecuaciones
row = 0
for i, state in enumerate(states):
    if state == target_state:
        continue # No necesitas una ecuación para el estado objetivo

    col = 0 # Reiniciar el índice de columna para cada fila
    for j, next_state in enumerate(states):
        if next_state == target_state:
            continue # No incluimos el estado objetivo en las columnas

        A[row, col] = -transition_matrix[i, j] # Probabilidades de transición
        col += 1 # Avanzar al siguiente índice de columna

    A[row, row] += 1 # Coeficiente de la variable actual
    row += 1 # Avanzar a la siguiente fila

# Resolvemos el sistema de ecuaciones A * T = b
T_solution = np.linalg.solve(A, b)

# Encontrar el índice del estado inicial en la lista reducida (sin el estado objetivo)
initial_state_index = state_indices[initial_state]
if initial_state_index > state_indices[target_state]:
    initial_state_index -= 1 # Ajustar el índice porque removimos el estado objetivo

# Mostrar el resultado para el estado inicial (2, 2)
T_initial = T_solution[initial_state_index]

print(f"El valor esperado del tiempo para ir de {initial_state} a {target_state} es {T_initial} horas.")

```