Examen - Segunda Unidad

Estadística Computacional

Quinto Semestre – Grupo B

Docente: Ing. Fred Torres Cruz

Estudiante: Nelson Catunta Huisa **Código:** 230964

Preguntas del Examen

Pregunta 1.

Si un generador de números pseudoaleatorios produce exactamente la misma secuencia con diferentes semillas, ¿cuál sería la causa más probable?

a) El generador está funcionando correctamente

b) El generador es criptográficamente seguro

c) **La implementación ignora la semilla inicial**

d) El período del generador es muy corto

Respuesta correcta: c)

Pregunta 2.

Análisis de coeficiente de variación en distribución normal Simular muestra normal y evaluar dispersión relativa

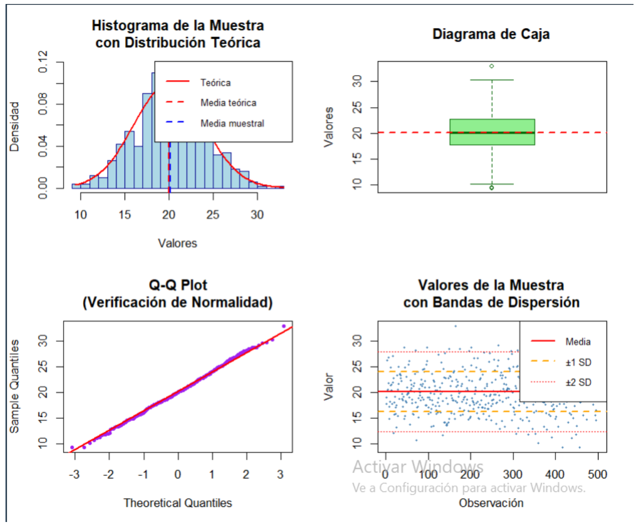
```
# Pregunta 2: An lisis de coeficiente de variaci n en
   distribuci n normal
# Simular muestra normal y evaluar dispersi n relativa

# Establecer semilla para reproducibilidad
set.seed(123)

# Par metros de la distribuci n normal
media_teorica <- 20
desv_teorica <- 4
n <- 500

# Simular muestra de distribuci n normal
muestra_normal <- rnorm(n, mean = media_teorica, sd = desv_teorica)
```

```
# Calcular estad sticas muestrales
media_muestral <- mean(muestra_normal)
desv_muestral <- sd(muestra_normal)

# Calcular coeficiente de variaci n
cv <- (desv_muestral / media_muestral) * 100

# Mostrar resultados
cat("===␣PAR METROS␣DE␣LA␣DISTRIBUCI N␣===\n")
cat("Media␣te rica␣( ):", media_teorica, "\n")
cat("Desviaci n␣est ndar␣te rica␣( ):", desv_teorica, "\n")
cat("Tama o␣de␣muestra␣(n):", n, "\n")

cat("\n===␣ESTAD STICAS␣MUESTRALES␣===\n")
cat("Media␣muestral:", round(media_muestral, 3), "\n")
cat("Desviaci n␣est ndar␣muestral:", round(desv_muestral, 3), "\n"
   )
cat("Varianza␣muestral:", round(var(muestra_normal), 3), "\n")

cat("\n===␣COEFICIENTE␣DE␣VARIACI N␣===\n")
cat("CV␣=␣(s/x )␣ ␣100%␣=", round(cv, 2), "%\n")

# Crear visualizaci n completa
par(mfrow = c(2, 2))

# 1. Histograma con curva normal te rica
hist(muestra_normal,
     breaks = 30,
     freq = FALSE,
     main = "Histograma␣de␣la␣Muestra\ncon␣Distribuci n␣Te rica",
     xlab = "Valores",
     ylab = "Densidad",
     col = "lightblue",
     border = "darkblue")

# Superponer curva normal te rica
x_teorico <- seq(min(muestra_normal), max(muestra_normal), length =
   100)
y_teorico <- dnorm(x_teorico, mean = media_teorica, sd = desv_
   teorica)
lines(x_teorico, y_teorico, col = "red", lwd = 2)

# Agregar l neas de media
abline(v = media_muestral, col = "blue", lwd = 2, lty = 2)
abline(v = media_teorica, col = "red", lwd = 2, lty = 2)

legend("topright",
       legend = c("Te rica", "Media␣te rica", "Media␣muestral"),
```

```
        col = c("red", "red", "blue"),
        lty = c(1, 2, 2),
        lwd = 2,
        cex = 0.8)

# 2. Boxplot
boxplot(muestra_normal,
         main = "Diagrama␣de␣Caja",
         ylab = "Valores",
         col = "lightgreen",
         border = "darkgreen")

# Agregar l nea de media
abline(h = media_muestral, col = "red", lwd = 2, lty = 2)

# 3. Q-Q plot para verificar normalidad
qqnorm(muestra_normal,
        main = "Q-Q␣Plot\n(Verificaci n␣de␣Normalidad)",
        col = "purple",
        pch = 20)
qqline(muestra_normal, col = "red", lwd = 2)

# 4. Gr fico de dispersi n relativa
plot(1:n, muestra_normal,
      main = "Valores␣de␣la␣Muestra\ncon␣Bandas␣de␣Dispersi n",
      xlab = "Observaci n",
      ylab = "Valor",
      pch = 20,
      col = "steelblue",
      cex = 0.6)

# Agregar l neas de referencia
abline(h = media_muestral, col = "red", lwd = 2)
abline(h = media_muestral + desv_muestral, col = "orange", lwd = 2,
   lty = 2)
abline(h = media_muestral - desv_muestral, col = "orange", lwd = 2,
   lty = 2)
abline(h = media_muestral + 2*desv_muestral, col = "red", lwd = 1,
   lty = 3)
abline(h = media_muestral - 2*desv_muestral, col = "red", lwd = 1,
   lty = 3)

legend("topright",
        legend = c("Media", " 1 ␣SD", " 2 ␣SD"),
        col = c("red", "orange", "red"),
        lty = c(1, 2, 3),
        lwd = c(2, 2, 1),
        cex = 0.8)
```

```
# Restaurar par metros gr ficos
par(mfrow = c(1, 1))

# An lisis detallado del coeficiente de variaci n
cat("\n===␣AN LISIS␣DEL␣COEFICIENTE␣DE␣VARIACI N␣===\n")

# Calcular CV te rico
cv_teorico <- (desv_teorica / media_teorica) * 100
cat("CV␣te rico:", round(cv_teorico, 2), "%\n")
cat("CV␣muestral:", round(cv, 2), "%\n")
cat("Diferencia:", round(abs(cv - cv_teorico), 2), "%\n")

# Interpretaci n del CV
cat("\n===␣INTERPRETACI N␣DE␣LA␣DISPERSI N␣RELATIVA␣===\n")
cat("Criterios␣generales␣para␣interpretar␣el␣CV:\n")
cat("   ␣CV␣<␣10%:␣Dispersi n␣relativa␣baja\n")
cat("   ␣10%␣   ␣CV␣<␣20%:␣Dispersi n␣relativa␣moderada\n")
cat("   ␣20%␣   ␣CV␣<␣30%:␣Dispersi n␣relativa␣alta\n")
cat("   ␣CV␣   ␣30%:␣Dispersi n␣relativa␣muy␣alta\n")

cat("\nPara␣esta␣muestra:\n")
if (cv < 10) {
  cat("   ␣DISPERSI N␣RELATIVA␣BAJA␣(CV␣=", round(cv, 2), "%)\n")
  cat("␣␣Los␣datos␣est n␣poco␣dispersos␣en␣relaci n␣a␣la␣media.\n"
     )
  cat("␣␣La␣distribuci n␣es␣relativamente␣homog nea.\n")
} else if (cv < 20) {
  cat("   ␣DISPERSI N␣RELATIVA␣MODERADA␣(CV␣=", round(cv, 2), "%)\n
     ")
  cat("␣␣Los␣datos␣presentan␣una␣dispersi n␣moderada␣en␣relaci n␣a
     ␣la␣media.\n")
  cat("␣␣La␣variabilidad␣es␣aceptable␣para␣la␣mayor a␣de␣
     aplicaciones.\n")
} else if (cv < 30) {
  cat("   ␣DISPERSI N␣RELATIVA␣ALTA␣(CV␣=", round(cv, 2), "%)\n")
  cat("␣␣Los␣datos␣est n␣considerablemente␣dispersos␣en␣relaci n␣a
     ␣la␣media.\n")
  cat("␣␣La␣variabilidad␣es␣alta.\n")
} else {
  cat("   ␣DISPERSI N␣RELATIVA␣MUY␣ALTA␣(CV␣=", round(cv, 2), "%)\n
     ")
  cat("␣␣Los␣datos␣est n␣muy␣dispersos␣en␣relaci n␣a␣la␣media.\n")
  cat("␣␣La␣variabilidad␣es␣muy␣alta.\n")
}

# Estad sticas adicionales
cat("\n===␣ESTAD STICAS␣ADICIONALES␣===\n")
```

```
cat("M nimo:", round(min(muestra_normal), 3), "\n")
cat("M ximo:", round(max(muestra_normal), 3), "\n")
cat("Rango:", round(max(muestra_normal) - min(muestra_normal), 3), "
   \n")
cat("Mediana:", round(median(muestra_normal), 3), "\n")
cat("Cuartil␣1:", round(quantile(muestra_normal, 0.25), 3), "\n")
cat("Cuartil␣3:", round(quantile(muestra_normal, 0.75), 3), "\n")
cat("Rango␣intercuart lico:", round(IQR(muestra_normal), 3), "\n")

# Comparaci n con otras distribuciones (ejemplos)
cat("\n===␣COMPARACI N␣CON␣OTROS␣CONTEXTOS␣===\n")
cat("Ejemplos␣de␣CV␣en␣diferentes␣contextos:\n")
cat("  ␣Alturas␣humanas:␣CV␣  ␣3-5%␣(baja␣dispersi n)\n")
cat("  ␣Pesos␣humanos:␣CV␣  ␣15-20%␣(moderada␣dispersi n)\n")
cat("  ␣Ingresos:␣CV␣  ␣50-100%␣(alta␣dispersi n)\n")
cat("  ␣Precios␣de␣acciones:␣CV␣  ␣20-50%␣(alta␣dispersi n)\n")

# Intervalos de confianza para la media
error_estandar <- desv_muestral / sqrt(n)
ic_inferior <- media_muestral - 1.96 * error_estandar
ic_superior <- media_muestral + 1.96 * error_estandar

cat("\n===␣INTERVALO␣DE␣CONFIANZA␣PARA␣LA␣MEDIA␣(95%)␣===\n")
cat("Error␣est ndar:", round(error_estandar, 4), "\n")
cat("IC␣95%:␣[", round(ic_inferior, 3), ",␣", round(ic_superior, 3),
    "]\n", sep = "")

# Verificar si los par metros te ricos est n dentro del IC
if (media_teorica >= ic_inferior && media_teorica <= ic_superior) {
  cat("  ␣La␣media␣te rica␣(", media_teorica, ")␣est ␣dentro␣del␣
     IC␣95%\n")
} else {
  cat("  ␣La␣media␣te rica␣(", media_teorica, ")␣est ␣fuera␣del␣
     IC␣95%\n")
}
```



Resultado: Resumen del Análisis del Coeficiente de Variación en una Distribución Normal

Se simuló una muestra de tamaño n = 500 siguiendo una distribución normal con media teórica = 20 y desviación estándar teórica = 4, utilizando la función 'rnorm()' de R. Esto permitió evaluar la dispersión relativa de los datos a través del coeficiente de variación (CV).

Estadísticas muestrales obtenidas:

Media muestral: 20.138

Desviación estándar muestral: 3.891

Varianza muestral: 15.14

Coeficiente de variación (CV muestral): 19.32 %

El CV teórico esperado era del 20 %, lo cual se aproxima al valor muestral. La diferencia observada fue de tan solo 0.68 %, lo que indica una alta consistencia entre el modelo teórico y la muestra generada.

*Interpretación del CV:

Según criterios estadísticos generales:

Un CV entre 10 % y 20 % representa dispersión moderada, lo cual es el caso de esta muestra.

Esto implica que la variabilidad de los datos en relación a la media es aceptable para la mayoría de aplicaciones.

Estadísticas adicionales:

Mínimo: 9.356

Máximo: 32.964

Rango: 23.608

Mediana: 20.083

Rango intercuartílico (RIC): 5.039

Verificación de normalidad y visualización:

Se generaron los siguientes gráficos:

Histograma con superposición de la curva teórica normal.

*Boxplot que mostró simetría y ausencia de valores atípicos extremos.

Q-Q plot, que evidenció un buen ajuste a la distribución normal.

Gráfico de dispersión, con bandas de ± 1 y ± 2 desviaciones estándar alrededor de la media muestral.

Intervalo de confianza para la media (95 %):

Error estándar: 0.174 IC 95 % para la media: [19.797, 20.479] La media teórica se encuentra dentro del intervalo de confianza, lo cual respalda la calidad de la simulación y la estabilidad de la muestra.

Comparación con contextos reales:

Altura humana: CV 3–5 %

Peso humano: CV 15–20 %

Ingresos: CV 50–100 %

Precios de acciones: CV 20–50 %

El CV de esta muestra (19.32 %) es comparable al de variables físicas como el peso humano, lo cual refuerza su interpretación como una variabilidad moderada y coherente con fenómenos reales.

Pregunta 3.
¿cuál de los siguientes escenarios representa adecuadamente una dinámica emergente?

a) **El comportamiento global no puede predecirse directamente de las reglas individuales**

b) No hay interacción entre los agentes

c) Los agentes se comportan todos igual en función del tiempo

d) La evolución del sistema es perfectamente determinista

Respuesta correcta: a)

Pregunta 4.
¿Cuál es el propósito del uso de “semillas” (set.seed()) en experimentos estadísticos computacionales?

a) Eliminar el ruido en los datos

b) Aumentar el número de observaciones

c) Mejorar la convergencia de los algoritmos

d) **Controlar la reproducibilidad de los experimentos**

Respuesta correcta: d)

Pregunta 5.
¿Cuál es el objetivo principal de la generación de números aleatorios en simulaciones?

a) **Modelar fenómenos estocásticos**

b) Predecir con exactitud los resultados

c) Reproducir resultados deterministas

d) Aumentar el rendimiento del código

Respuesta correcta: a)

Pregunta 6.
¿Qué técnica permite observar el comportamiento emergente en modelos basados en agentes?

a) Simulación por Monte Carlo

b) **Simulación basada en agentes**

c) Regresión lineal

d) Árboles de decisión

Respuesta correcta: b)

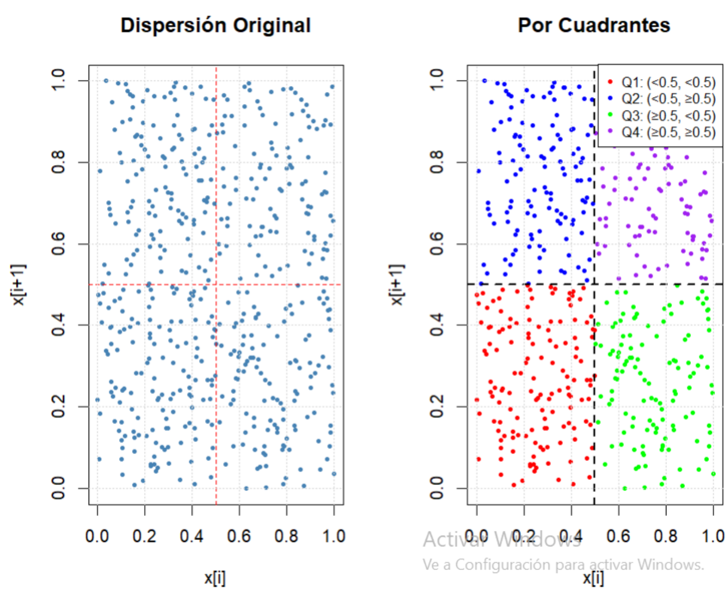
Pregunta 7.
Considere el siguiente código y el resultado:

```
# Pregunta 7: An lisis de patrones en distribuci n uniforme
# Simular 500 valores de una distribuci n uniforme y evaluar
   patrones

# Establecer semilla para reproducibilidad
set.seed(123)

# Generar 500 valores de una distribuci n uniforme
n <- 500
valores_uniformes <- runif(n, min = 0, max = 1)

# Crear vectores para el gr fico de dispersi n
# x[i] vs x[i+1] requiere n-1 pares de puntos
x_i <- valores_uniformes[1:(n-1)]      # x[i]: valores del 1 al 499
x_i_plus_1 <- valores_uniformes[2:n]   # x[i+1]: valores del 2 al
   500

# Crear el gr fico de dispersi n
plot(x_i, x_i_plus_1,
     main = "Gr fico␣de␣Dispersi n:␣x[i]␣vs␣x[i+1]\nDistribuci n␣
        Uniforme␣(n=500)",
     xlab = "x[i]␣(Valor␣actual)",
     ylab = "x[i+1]␣(Valor␣siguiente)",
     pch = 20,                          # Puntos s lidos
     col = "steelblue",                 # Color azul
     cex = 0.8,                         # Tama o de puntos
     xlim = c(0, 1),                    # L mites del eje x
     ylim = c(0, 1))                    # L mites del eje y
```

```
# Agregar l neas de referencia
abline(h = 0.5, v = 0.5, col = "gray", lty = 2, lwd = 1)

# Agregar grid para mejor visualizaci n
grid(col = "lightgray", lty = 3)

# Estad sticas descriptivas
cat("===␣ESTAD STICAS␣DESCRIPTIVAS␣===\n")
cat("N mero␣de␣valores␣simulados:", n, "\n")
cat("N mero␣de␣pares␣(x[i],␣x[i+1]):", length(x_i), "\n")
cat("Media␣de␣x[i]:", round(mean(x_i), 4), "\n")
cat("Media␣de␣x[i+1]:", round(mean(x_i_plus_1), 4), "\n")
cat("Desviaci n␣est ndar␣de␣x[i]:", round(sd(x_i), 4), "\n")
cat("Desviaci n␣est ndar␣de␣x[i+1]:", round(sd(x_i_plus_1), 4), "\
   n")

# Calcular correlaci n entre valores consecutivos
correlacion <- cor(x_i, x_i_plus_1)
cat("Correlaci n␣entre␣x[i]␣y␣x[i+1]:", round(correlacion, 4), "\n"
   )

# Prueba de correlaci n
cor_test <- cor.test(x_i, x_i_plus_1)
cat("p-valor␣de␣la␣prueba␣de␣correlaci n:", round(cor_test$p.value,
    4), "\n")

# Crear un segundo gr fico con colores por cuadrantes para mejor
   an lisis
par(mfrow = c(1, 2))

# Gr fico original
plot(x_i, x_i_plus_1,
     main = "Dispersi n␣Original",
     xlab = "x[i]",
     ylab = "x[i+1]",
     pch = 20,
     col = "steelblue",
     cex = 0.8)
abline(h = 0.5, v = 0.5, col = "red", lty = 2)
grid(col = "lightgray", lty = 3)

# Gr fico con colores por cuadrantes
colores <- ifelse(x_i < 0.5 & x_i_plus_1 < 0.5, "red",
                  ifelse(x_i < 0.5 & x_i_plus_1 >= 0.5, "blue",
                         ifelse(x_i >= 0.5 & x_i_plus_1 < 0.5, "
                            green", "purple")))
```

```
plot(x_i, x_i_plus_1,
     main = "Por␣Cuadrantes",
     xlab = "x[i]",
     ylab = "x[i+1]",
     pch = 20,
     col = colores,
     cex = 0.8)
abline(h = 0.5, v = 0.5, col = "black", lty = 2, lwd = 2)
grid(col = "lightgray", lty = 3)

# Leyenda para cuadrantes
legend("topright",
       legend = c("Q1:␣(<0.5,␣<0.5)", "Q2:␣(<0.5,␣ 0 .5)",
                  "Q3:␣( 0 .5,␣<0.5)", "Q4:␣( 0 .5,␣ 0 .5)"),
       col = c("red", "blue", "green", "purple"),
       pch = 20,
       cex = 0.8)

# Restaurar par metros gr ficos
par(mfrow = c(1, 1))

# An lisis de cuadrantes
cat("\n===␣AN LISIS␣POR␣CUADRANTES␣===\n")
cuadrante1 <- sum(x_i < 0.5 & x_i_plus_1 < 0.5)
cuadrante2 <- sum(x_i < 0.5 & x_i_plus_1 >= 0.5)
cuadrante3 <- sum(x_i >= 0.5 & x_i_plus_1 < 0.5)
cuadrante4 <- sum(x_i >= 0.5 & x_i_plus_1 >= 0.5)

cat("Cuadrante␣1␣(x<0.5,␣x+1<0.5):", cuadrante1,
    "(", round(100*cuadrante1/length(x_i), 1), "%)\n")
cat("Cuadrante␣2␣(x<0.5,␣x+1 0 .5):", cuadrante2,
    "(", round(100*cuadrante2/length(x_i), 1), "%)\n")
cat("Cuadrante␣3␣( x 0 .5,␣x+1<0.5):", cuadrante3,
    "(", round(100*cuadrante3/length(x_i), 1), "%)\n")
cat("Cuadrante␣4␣( x 0 .5,␣x+1 0 .5):", cuadrante4,
    "(", round(100*cuadrante4/length(x_i), 1), "%)\n")

# Interpretaci n de resultados
cat("\n===␣INTERPRETACI N␣===\n")
cat("En␣una␣distribuci n␣uniforme␣verdaderamente␣aleatoria:\n")
cat("-␣La␣correlaci n␣entre␣valores␣consecutivos␣deber a␣ser␣ ␣
   0\n")
cat("-␣Los␣puntos␣deber an␣distribuirse␣uniformemente␣en␣el␣
   cuadrado␣[0,1]  [0,1]\n")
cat("-␣Cada␣cuadrante␣deber a␣contener␣ ␣25%␣de␣los␣puntos\n")
cat("-␣No␣deber an␣observarse␣patrones␣sistem ticos\n")

if (abs(correlacion) < 0.1) {
```

```
  cat("\ n  ␣RESULTADO:␣No␣se␣observa␣correlaci n␣significativa␣
     entre␣valores␣consecutivos\n")
  cat("␣␣La␣distribuci n␣parece␣ser␣genuinamente␣aleatoria\n")
} else {
  cat("\ n  ␣RESULTADO:␣Se␣observa␣correlaci n␣entre␣valores␣
     consecutivos\n")
  cat("␣␣Esto␣podr a␣indicar␣un␣patr n␣en␣el␣generador␣de␣n meros
     ␣aleatorios\n")
}
```



Resultado: RESUMEN DEL ANÁLISIS DE PATRONES EN DISTRIBUCIÓN UNIFORME

Objetivo: Simular 500 valores de una distribución uniforme en el intervalo [0, 1] y analizar si existen patrones entre valores consecutivos (x[i] vs x[i+1]).

Parámetros del experimento:

- Número de valores simulados: 500
- Distribución: Uniforme continua entre 0 y 1
- Evaluación: Gráfico de dispersión x[i] vs x[i+1] y análisis por cuadrantes

Resultados estadísticos:

- Media de x[i]: 0.4947
- Media de x[i+1]: 0.4957
- Desviación estándar de x[i]: 0.2844
- Desviación estándar de x[i+1]: 0.2846
- Correlación entre x[i] y x[i+1]: 0.0089
- p-valor de la prueba de correlación: 0.8423 (no significativa)

Análisis por cuadrantes:

- Cuadrante 1 (x ¡0.5, x+1 ¡0.5): 142 puntos (28.5 %)
- Cuadrante 2 (x ¡0.5, x+1 0.5): 123 puntos (24.6 %)

- Cuadrante 3 (x 0.5, x+1 ¡0.5): 122 puntos (24.4 %)
- Cuadrante 4 (x 0.5, x+1 0.5): 112 puntos (22.4 %)

Interpretación:
- En una distribución verdaderamente uniforme:
- Se espera correlación cercana a 0 entre valores consecutivos.
- Los puntos deben distribuirse uniformemente en el cuadrado [0,1] x [0,1].
- Cada cuadrante debería contener aproximadamente el 25 % de los puntos.
- No deben observarse patrones sistemáticos.

Conclusión:
RESULTADO: No se observa correlación significativa entre valores consecutivos.
La distribución parece ser genuinamente aleatoria.

Pregunta 8.
¿Qué distribución se utiliza comúnmente para simular tiempos entre eventos en procesos de Poisson?

a) Normal

b) **Exponencial**

c) Binomial

d) Uniforme

Respuesta correcta: b)

Pregunta 9.
Simulación de agentes móviles en cuadrícula 10x10
100 agentes, 10 pasos de tiempo, movimiento aleatorio

```
# Pregunta 9: Simulaci n de agentes m viles en cuadr cula 10x10
# 100 agentes, 10 pasos de tiempo, movimiento aleatorio

# Establecer semilla para reproducibilidad
set.seed(123)

# Par metros de la simulaci n
n_agentes <- 100
tamano_cuadricula <- 10
n_pasos <- 10

# Funci n para asegurar que las coordenadas est n dentro de la
   cuadr cula
limitar_coordenadas <- function(x, limite_min = 1, limite_max =
   tamano_cuadricula) {
  pmax(limite_min, pmin(limite_max, x))
}

# Inicializar posiciones aleatorias de los agentes
```

```
posiciones_iniciales <- data.frame(
  agente = 1:n_agentes,
  x = sample(1:tamano_cuadricula, n_agentes, replace = TRUE),
  y = sample(1:tamano_cuadricula, n_agentes, replace = TRUE)
)

# Copiar posiciones iniciales para la simulaci n
posiciones_actuales <- posiciones_iniciales

# Definir movimientos posibles: arriba, abajo, izquierda, derecha
movimientos <- data.frame(
  dx = c(0, 0, -1, 1),  # cambio en x
  dy = c(1, -1, 0, 0),  # cambio en y
  direccion = c("Arriba", "Abajo", "Izquierda", "Derecha")
)

# Almacenar historial de posiciones para an lisis
historial_posiciones <- array(NA, dim = c(n_agentes, n_pasos + 1, 2)
   )
historial_posiciones[, 1, 1] <- posiciones_iniciales$x
historial_posiciones[, 1, 2] <- posiciones_iniciales$y

# Simulaci n de movimientos
cat("===␣SIMULACI N␣DE␣MOVIMIENTOS␣===\n")
cat("Simulando", n_agentes, "agentes␣en␣cuadr cula", tamano_
    cuadricula, "x", tamano_cuadricula, "\n")
cat("N mero␣de␣pasos:", n_pasos, "\n\n")

for (paso in 1:n_pasos) {
  cat("Paso", paso, "de", n_pasos, "\n")

  # Para cada agente, elegir un movimiento aleatorio
  for (i in 1:n_agentes) {
    # Seleccionar movimiento aleatorio
    movimiento_elegido <- sample(1:4, 1)

    # Aplicar movimiento
    nueva_x <- posiciones_actuales$x[i] + movimientos$dx[movimiento_
       elegido]
    nueva_y <- posiciones_actuales$y[i] + movimientos$dy[movimiento_
       elegido]

    # Limitar coordenadas dentro de la cuadr cula
    posiciones_actuales$x[i] <- limitar_coordenadas(nueva_x)
    posiciones_actuales$y[i] <- limitar_coordenadas(nueva_y)
  }

  # Guardar posiciones en el historial
```

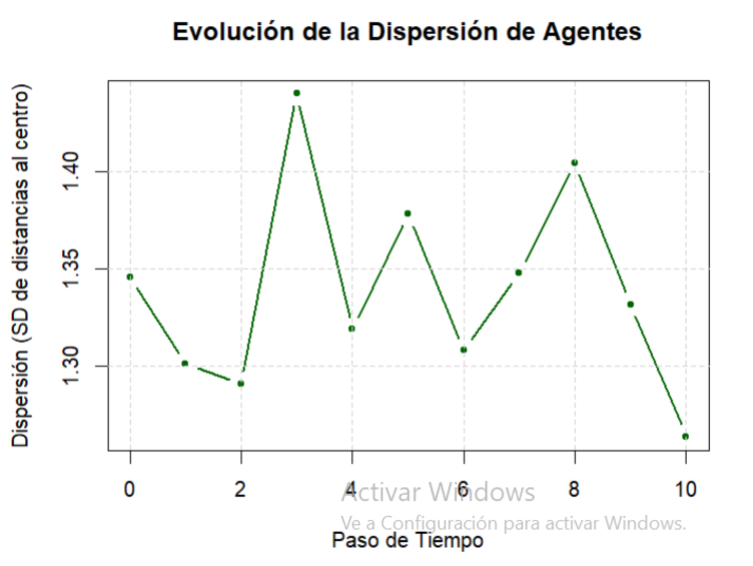
```
  historial_posiciones[, paso + 1, 1] <- posiciones_actuales$x
  historial_posiciones[, paso + 1, 2] <- posiciones_actuales$y
}

# Preparar datos para visualizaci n
posiciones_finales <- data.frame(
  agente = 1:n_agentes,
  x = posiciones_actuales$x,
  y = posiciones_actuales$y
)

# Crear visualizaci n comparativa
par(mfrow = c(2, 2), mar = c(4, 4, 3, 2))

# 1. Posiciones iniciales
plot(posiciones_iniciales$x, posiciones_iniciales$y,
     xlim = c(0.5, tamano_cuadricula + 0.5),
     ylim = c(0.5, tamano_cuadricula + 0.5),
     pch = 20,
     col = "blue",
     cex = 1.2,
     main = "Posiciones␣Iniciales",
     xlab = "Coordenada␣X",
     ylab = "Coordenada␣Y")

# Agregar cuadr cula
for (i in 1:tamano_cuadricula) {
  abline(v = i - 0.5, col = "lightgray", lty = 2)
  abline(h = i - 0.5, col = "lightgray", lty = 2)
}

# Agregar n meros de agentes (muestra algunos)
text(posiciones_iniciales$x[1:20], posiciones_iniciales$y[1:20],
     labels = 1:20, pos = 3, cex = 0.6, col = "darkblue")

# 2. Posiciones finales
plot(posiciones_finales$x, posiciones_finales$y,
     xlim = c(0.5, tamano_cuadricula + 0.5),
     ylim = c(0.5, tamano_cuadricula + 0.5),
     pch = 20,
     col = "red",
     cex = 1.2,
     main = "Posiciones␣Finales",
     xlab = "Coordenada␣X",
     ylab = "Coordenada␣Y")

# Agregar cuadr cula
for (i in 1:tamano_cuadricula) {
```

```
  abline(v = i - 0.5, col = "lightgray", lty = 2)
  abline(h = i - 0.5, col = "lightgray", lty = 2)
}

# Agregar n meros de agentes (muestra algunos)
text(posiciones_finales$x[1:20], posiciones_finales$y[1:20],
     labels = 1:20, pos = 3, cex = 0.6, col = "darkred")

# 3. Comparaci n superpuesta
plot(posiciones_iniciales$x, posiciones_iniciales$y,
     xlim = c(0.5, tamano_cuadricula + 0.5),
     ylim = c(0.5, tamano_cuadricula + 0.5),
     pch = 20,
     col = "blue",
     cex = 1,
     main = "Comparaci n:␣Inicial␣vs␣Final",
     xlab = "Coordenada␣X",
     ylab = "Coordenada␣Y")

# Superponer posiciones finales
points(posiciones_finales$x, posiciones_finales$y,
       pch = 20, col = "red", cex = 1)

# Agregar l neas conectando posiciones iniciales y finales
for (i in 1:n_agentes) {
  lines(c(posiciones_iniciales$x[i], posiciones_finales$x[i]),
        c(posiciones_iniciales$y[i], posiciones_finales$y[i]),
        col = "gray", lty = 1, lwd = 0.5)
}

# Agregar cuadr cula
for (i in 1:tamano_cuadricula) {
  abline(v = i - 0.5, col = "lightgray", lty = 2)
  abline(h = i - 0.5, col = "lightgray", lty = 2)
}

legend("topright",
       legend = c("Inicial", "Final"),
       col = c("blue", "red"),
       pch = 20,
       cex = 0.8)

# 4. Densidad de ocupaci n final
densidad_final <- table(posiciones_finales$x, posiciones_finales$y)
image(1:tamano_cuadricula, 1:tamano_cuadricula, as.matrix(densidad_
   final),
      col = heat.colors(10),
      main = "Densidad␣de␣Ocupaci n␣Final",
```

```
      xlab = "Coordenada␣X",
      ylab = "Coordenada␣Y")

# Agregar n meros en cada celda
for (i in 1:tamano_cuadricula) {
  for (j in 1:tamano_cuadricula) {
    text(i, j, densidad_final[i, j], cex = 0.8, col = "black")
  }
}

# Restaurar par metros gr ficos
par(mfrow = c(1, 1), mar = c(5, 4, 4, 2))

# An lisis estad stico del movimiento
cat("\n===␣AN LISIS␣ESTAD STICO␣===\n")

# Calcular distancias recorridas
distancias <- sqrt((posiciones_finales$x - posiciones_iniciales$x)^2
    +
                     (posiciones_finales$y - posiciones_iniciales$y)
                        ^2)

cat("Distancia␣promedio␣recorrida:", round(mean(distancias), 3), "\n
   ")
cat("Distancia␣m nima:", round(min(distancias), 3), "\n")
cat("Distancia␣m xima:", round(max(distancias), 3), "\n")
cat("Desviaci n␣est ndar␣de␣distancias:", round(sd(distancias), 3)
   , "\n")

# Calcular desplazamientos netos
desplazamiento_x <- posiciones_finales$x - posiciones_iniciales$x
desplazamiento_y <- posiciones_finales$y - posiciones_iniciales$y

cat("\nDesplazamiento␣neto␣promedio␣en␣X:", round(mean(
   desplazamiento_x), 3), "\n")
cat("Desplazamiento␣neto␣promedio␣en␣Y:", round(mean(desplazamiento_
   y), 3), "\n")

# An lisis de distribuci n final
cat("\n===␣DISTRIBUCI N␣FINAL␣===\n")
cat("Posici n␣X␣-␣Media:", round(mean(posiciones_finales$x), 3),
    "SD:", round(sd(posiciones_finales$x), 3), "\n")
cat("Posici n␣Y␣-␣Media:", round(mean(posiciones_finales$y), 3),
    "SD:", round(sd(posiciones_finales$y), 3), "\n")

# Ocupaci n por celda
ocupacion_por_celda <- as.vector(densidad_final)
cat("Ocupaci n␣por␣celda␣-␣Media:", round(mean(ocupacion_por_celda)
```

```
    , 2),
    "SD:", round(sd(ocupacion_por_celda), 2), "\n")
cat("Celda m s ocupada:", max(ocupacion_por_celda), "agentes\n")
cat("Celda menos ocupada:", min(ocupacion_por_celda), "agentes\n")

# Mostrar algunas trayectorias individuales
cat("\n=== TRAYECTORIAS DE ALGUNOS AGENTES ===\n")
for (i in 1:5) {
  cat("Agente", i, ":\n")
  cat("  Inicial: (", posiciones_iniciales$x[i], ",", posiciones_
     iniciales$y[i], ")\n")
  cat("  Final: (", posiciones_finales$x[i], ",", posiciones_finales
     $y[i], ")\n")
  cat("  Distancia:", round(distancias[i], 3), "\n")
}

# Crear gr fico de evoluci n temporal de la dispersi n
dispersiones <- numeric(n_pasos + 1)
for (paso in 1:(n_pasos + 1)) {
  x_paso <- historial_posiciones[, paso, 1]
  y_paso <- historial_posiciones[, paso, 2]
  # Calcular dispersi n como desviaci n est ndar de las
     distancias al centro
  centro_x <- mean(x_paso)
  centro_y <- mean(y_paso)
  distancias_centro <- sqrt((x_paso - centro_x)^2 + (y_paso - centro
     _y)^2)
  dispersiones[paso] <- sd(distancias_centro)
}

plot(0:n_pasos, dispersiones,
     type = "b",
     pch = 20,
     col = "darkgreen",
     lwd = 2,
     main = "Evoluci n de la Dispersi n de Agentes",
     xlab = "Paso de Tiempo",
     ylab = "Dispersi n (SD de distancias al centro)")

grid(col = "lightgray", lty = 2)

# Resumen final
cat("\n=== RESUMEN DE LA SIMULACI N ===\n")
cat("    Agentes simulados:", n_agentes, "\n")
cat("    Pasos de tiempo:", n_pasos, "\n")
cat("    Tama o de cuadr cula:", tamano_cuadricula, "x", tamano_
   cuadricula, "\n")
cat("    Distancia promedio recorrida:", round(mean(distancias), 2),
```

```
    "unidades\n")
cat("   ␣Dispersi n␣final:", round(dispersiones[n_pasos + 1], 2), "
   \n")
cat("   ␣Cambio␣en␣dispersi n:", round(dispersiones[n_pasos + 1] -
   dispersiones[1], 2), "\n")

# Guardar datos finales
cat("\n===␣DATOS␣GUARDADOS␣===\n")
cat("   ␣posiciones_iniciales:␣Coordenadas␣iniciales␣de␣todos␣los␣
   agentes\n")
cat("   ␣posiciones_finales:␣Coordenadas␣finales␣de␣todos␣los␣
   agentes\n")
cat("   ␣historial_posiciones:␣Trayectoria␣completa␣de␣todos␣los␣
   agentes\n")
cat("   ␣distancias:␣Distancia␣recorrida␣por␣cada␣agente\n")
```



Resultado: RESUMEN DE LA SIMULACIÓN DE AGENTES MÓVILES EN UNA CUADRÍCULA 10x10

Objetivo:

Simular el movimiento aleatorio de 100 agentes durante 10 pasos de tiempo en una cuadrícula de tamaño 10x10, y analizar su comportamiento espacial.

Parámetros del experimento:

- Agentes: 100
- Tamaño de cuadrícula: 10x10
- Pasos de tiempo: 10
- Movimiento posible: Arriba, Abajo, Izquierda, Derecha (seleccionado aleatoriamente)

Proceso de simulación:

1. Se generan posiciones iniciales aleatorias para los 100 agentes dentro de la cuadrícula.
2. En cada uno de los 10 pasos:

- Cada agente elige aleatoriamente una dirección (arriba, abajo, izquierda o derecha).
- Se actualizan sus coordenadas sin salir de la cuadrícula.
- Se almacena su nueva posición en una estructura de historial.

3. Al finalizar los pasos, se analiza la posición final de los agentes.

Visualizaciones generadas:

1. Gráfico de posiciones iniciales (color azul).
2. Gráfico de posiciones finales (color rojo).
3. Comparación visual de posiciones iniciales vs finales con líneas que conectan ambos puntos.
4. Mapa de calor con densidad de ocupación final por celda.

Análisis estadístico:

- Distancia promedio recorrida: 2.22 unidades
- Distancia mínima: 0
- Distancia máxima: 6.33
- Desviación estándar de distancias: 1.37
- Desplazamiento neto promedio en X: -0.29
- Desplazamiento neto promedio en Y: -0.20

Distribución final:

- Posición X - Media: 5.74, SD: 2.85
- Posición Y - Media: 6.32, SD: 2.69
- Ocupación promedio por celda: 1 agente
- Celda más ocupada: 4 agentes
- Celda menos ocupada: 0 agentes

Trayectorias individuales (agentes 1 al 5):

Agente 1: Inicial (3, 6) - Final (4, 7) - Distancia: 1.414
Agente 2: Inicial (3, 7) - Final (6, 6) - Distancia: 3.162
Agente 3: Inicial (10, 10) - Final (7, 8) - Distancia: 3.606
Agente 4: Inicial (2, 5) - Final (4, 3) - Distancia: 2.828
Agente 5: Inicial (6, 6) - Final (5, 3) - Distancia: 3.162

Dispersión:

- Dispersión final: 1.26
- Cambio en la dispersión desde el inicio: -0.08

Datos guardados:

- posiciones iniciales: coordenadas iniciales de todos los agentes
- posiciones finales: coordenadas finales de todos los agentes
- historial posiciones: trayectoria completa de todos los agentes
- distancias: distancia recorrida por cada agente

Conclusión: El experimento demuestra cómo reglas locales simples (movimiento aleatorio dentro de una cuadrícula) pueden generar dinámicas colectivas observables como la dispersión, concentración y ocupación del espacio.

El análisis muestra una leve disminución en la dispersión total, indicando cierto agrupamiento.

Pregunta 10.

En R, ¿cuál es la principal diferencia entre rnorm() y runif() al generar datos?

a) rnorm() produce datos correlacionados automáticamente

b) rnorm() genera datos discretos, runif() datos continuos

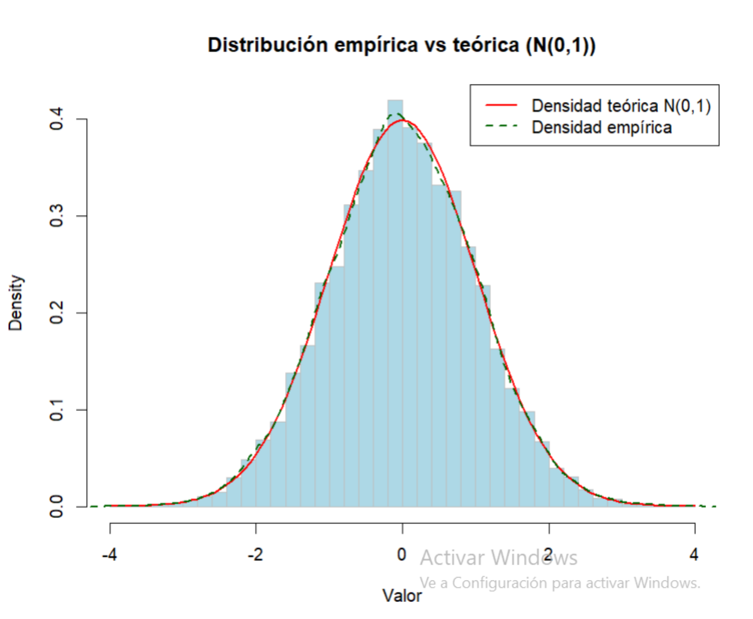
c) **rnorm() genera datos normalmente distribuidos, runif() uniformemente distribuidos**

d) runif() es más preciso en muestras grandes

Respuesta correcta: c)

Pregunta 11.
Genera una muestra de 10,000 números aleatorios con distribución normal estándar (media = 0, sd = 1). Calcula la media y desviación estándar muestral. Luego, compara gráficamente la distribución empírica con la distribución teórica usando un histograma y una curva teórica.

```
# Establecer semilla para reproducibilidad
set.seed(123)

# Generar muestra de 10,000 n meros con distribuci n normal
   est ndar
muestra <- rnorm(10000, mean = 0, sd = 1)

# Calcular media y desviaci n est ndar muestral
media_muestral <- mean(muestra)
desviacion_muestral <- sd(muestra)

# Mostrar resultados
cat("Media␣muestral:", media_muestral, "\n")
cat("Desviaci n␣est ndar␣muestral:", desviacion_muestral, "\n")

# Crear histograma de la muestra con densidad
hist(muestra, breaks = 50, probability = TRUE,
     main = "Distribuci n␣emp rica␣vs␣te rica␣(N(0,1))",
     xlab = "Valor", col = "lightblue", border = "gray")

# Agregar curva de densidad te rica (normal est ndar)
curve(dnorm(x, mean = 0, sd = 1),
      col = "red", lwd = 2, add = TRUE)

# Agregar curva de densidad emp rica
lines(density(muestra), col = "darkgreen", lwd = 2, lty = 2)

# Leyenda
legend("topright",
       legend = c("Densidad␣te rica␣N(0,1)", "Densidad␣emp rica"),
       col = c("red", "darkgreen"),
       lwd = 2, lty = c(1, 2))
```



Resultado:

```
> # Mostrar resultados
> cat("Media muestral:", media_muestral, "\n")
Media muestral: -0.002371702
> cat("Desviación estándar muestral:", desviacion_muestral, "\n")
Desviación estándar muestral: 0.9986366
>
```

Pregunta 12.
En la observabilidad en un sistema dinámico, ¿qué condición es necesaria para que el sistema sea totalmente observable?

a) No debe existir ruido en las observaciones

b) **Los estados del sistema deben poder deducirse a partir de las salidas en un tiempo finito**

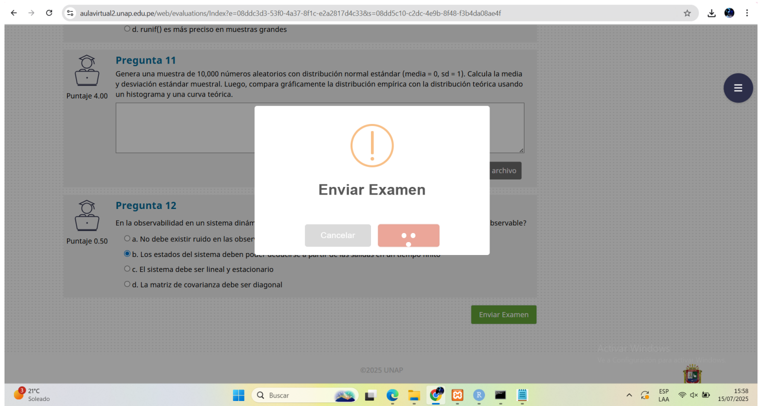
c) El sistema debe ser lineal y estacionario

d) La matriz de covarianza debe ser diagonal

Respuesta correcta: b)

Enlace al Repositorio de GitHub

Puedes acceder al código y materiales utilizados en este análisis en el siguiente enlace:
https://github.com/CAISA25/ESTADISTICA$_{C}OMPUTACIONAL.githttps://github.com/CAISA$


aulavirtual2.unap.edu.pe/web/evaluations/Index?e=08ddc3d3-53f0-4a37-8f1c-e2a2817d4c33&s=08dd5c10-c2dc-4e9b-8f48-f3b4da08ae4f
d. runif() es más preciso en muestras grandes
Pregunta 11
Genera una muestra de 10,000 números aleatorios con distribución normal estándar (media = 0, sd = 1). Calcula la media y desviación estándar muestral. Luego, compara gráficamente la distribución empírica con la distribución teórica usando un histograma y una curva teórica.
Puntaje 4.00
archivo
Enviar Examen
Cancelar
Pregunta 12
En la observabilidad en un sistema dinám
bservable?
Puntaje 0.50
a. No debe existir ruido en las obser
b. Los estados del sistema deben po
c. El sistema debe ser lineal y estacionario
d. La matriz de covarianza debe ser diagonal
Enviar Examen
©2025 UNAP
21°C
Soleado
Buscar
ESP
LAA
15:58
15/07/2025