% area-error - Cálculo del área de un polígono con simulación de errores gaussianos individuales (Monte Carlo)

% Incluye exportación a CSV e intervalo de confianza del 95%

% ======================== Entrada de datos ========================

% Solicitar el número de vértices

N = input('Ingresar el número de vértices: ');

% Ingresar coordenadas de los vértices

x = input('Ingresar vector de las coordenadas X: ');

y = input('Ingresar vector de las coordenadas Y: ');

% Validar que los vectores tengan la longitud correcta

if length(x) ~= N || length(y) ~= N

error('Error: los vectores X e Y deben tener exactamente %d elementos.', N);

endif

% Ingresar errores gaussianos individuales por vértice

error\_x = input('Ingresar vector de errores en X (desv. estándar): ');

error\_y = input('Ingresar vector de errores en Y (desv. estándar): ');

% Validar que los vectores de error tengan la longitud correcta

if length(error\_x) ~= N || length(error\_y) ~= N

error('Error: los vectores de errores deben tener exactamente %d elementos.', N);

endif

% ======================== Cálculo del área original ========================

% Cerrar el polígono agregando el primer punto al final

xc = [x, x(1)];

yc = [y, y(1)];

% Calcular el área usando la fórmula de Shoelace

suma1 = sum(xc(1:N) .\* yc(2:N+1));

suma2 = sum(yc(1:N) .\* xc(2:N+1));

A = 0.5 \* abs(suma1 - suma2);

% Mostrar el área original

fprintf('El área del polígono original es: %.4f\n', A);

% Graficar el polígono original

figure;

plot(xc, yc, '-o', 'LineWidth', 2, 'MarkerSize', 8, 'MarkerFaceColor', 'r');

grid on;

axis equal;

xlabel('X');

ylabel('Y');

title(['Polígono original - Área = ', num2str(A)]);

% Agregar etiquetas a los vértices

hold on;

for k = 1:N

text(x(k), y(k), sprintf(' (%.2f, %.2f)', x(k), y(k)), ...

'VerticalAlignment', 'bottom', 'HorizontalAlignment', 'right', 'FontSize', 10);

end

hold off;

% ======================== Simulación Monte Carlo ========================

P = input('Ingresar el número de simulaciones Monte Carlo (P): ');

areas\_MC = zeros(1, P); % Vector para almacenar áreas simuladas

for i = 1:P

% Desplazamientos aleatorios gaussianos por vértice

dx = randn(1, N) .\* error\_x;

dy = randn(1, N) .\* error\_y;

% Coordenadas perturbadas

x\_pert = x + dx;

y\_pert = y + dy;

% Cerrar el polígono perturbado

xcp = [x\_pert, x\_pert(1)];

ycp = [y\_pert, y\_pert(1)];

% Calcular área del polígono perturbado

s1 = sum(xcp(1:N) .\* ycp(2:N+1));

s2 = sum(ycp(1:N) .\* xcp(2:N+1));

area\_pert = 0.5 \* abs(s1 - s2);

areas\_MC(i) = area\_pert;

% Graficar los primeros 10 polígonos perturbados

if i <= 10

hold on;

plot(xcp, ycp, '--', 'Color', [0.6, 0.6, 0.6], 'LineWidth', 1);

endif

endfor

hold off;

% ======================== Estadística y resultados ========================

media\_area = mean(areas\_MC);

mediana\_area = median(areas\_MC);

desv\_est\_area = std(areas\_MC);

min\_area = min(areas\_MC);

max\_area = max(areas\_MC);

% Intervalo de confianza del 95% para la media del área simulada

z = 1.96; % Valor z para 95%

margen\_error = z \* desv\_est\_area / sqrt(P);

IC\_inf = media\_area - margen\_error;

IC\_sup = media\_area + margen\_error;

fprintf('\n--- Resultados estadísticos ---\n');

fprintf('Media del área simulada: %.4f\n', media\_area);

fprintf('Mediana del área simulada: %.4f\n', mediana\_area);

fprintf('Desviación estándar: %.4f\n', desv\_est\_area);

fprintf('Mínimo: %.4f\n', min\_area);

fprintf('Máximo: %.4f\n', max\_area);

fprintf('Intervalo de confianza del 95%%: [%.4f, %.4f]\n', IC\_inf, IC\_sup);

% ======================== Histograma ========================

figure;

hist(areas\_MC, 50);

xlabel('Área simulada');

ylabel('Frecuencia');

title('Distribución de áreas por simulación Monte Carlo (gaussiana por vértice)');

% ======================== Exportación ========================

nombre\_archivo = input('Ingresar nombre del archivo CSV para exportar resultados (sin extensión): ', 's');

csvwrite([nombre\_archivo, '.csv'], areas\_MC);

fprintf('Resultados exportados en: %s.csv\n', [nombre\_archivo, '.csv']);

**1. Entrada de datos**

* El usuario ingresa:
  + Cantidad de vértices N.
  + Coordenadas X e Y de los vértices del polígono.
  + Errores individuales (desviación estándar) en X e Y para cada vértice.
* Se valida que todos los vectores tengan longitud N.

**2. Cálculo del área original**

* El polígono se **cierra automáticamente** (repitiendo el primer punto al final).
* Se calcula el **área original** con la fórmula de Shoelace.

**3. Gráfico del polígono original**

* Se dibuja el polígono con sus vértices marcados.
* Se etiquetan los vértices con sus coordenadas.

**4. Simulación Monte Carlo**

* Se realizan P simulaciones.
* En cada simulación:
  + Se generan desplazamientos **aleatorios gaussianos** por vértice (con randn() escalado por los errores dados).
  + Se obtienen nuevas coordenadas perturbadas.
  + Se calcula el área del nuevo polígono perturbado.
  + Se guarda esa área.
  + Si es una de las primeras 10 simulaciones, se grafica ese polígono.

**5. Análisis estadístico**

* Se calcula:
  + Media, mediana, desviación estándar, mínimo y máximo del área simulada.
  + Un **intervalo de confianza del 95%** para la **media del área simulada**, asumiendo distribución normal.

**6. Histograma**

* Se grafica un histograma con 50 barras para ver la **distribución de las áreas simuladas**.

**7. Exportación a CSV**

* El usuario indica el nombre del archivo y se guarda un archivo .csv con todos los valores simulados.