UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

Escuela de Ciencias

Departamento de Actuaría, Física y Matemáticas

“Desempeño de estimadores no paramétricos de densidad de probabilidad: un estudio comparativo entre los algoritmos KDE y RU”

Tesis que, para completar los requisitos del Programa de Honores presenta la estudiante

Ariana Coyopol Figueroa

170525

Licenciatura en Actuaría

Dr. Miguel Ángel Reyes Cortés

San Andrés Cholula, Puebla. **Verano I, 2024**

Hoja de firmas

Tesis que, para completar los requisitos del Programa de Honores presenta el estudiante Ariana Coyopol Figueroa 170525

Director de Tesis

_____

Dr. Miguel Ángel Reyes Cortés

Presidente de Tesis

_____

Dra. Daniela Cortés Toto

Secretario de Tesis

_____

Dr. Rubén Blancas Rivera

Agradecimientos

Esta tesis no habría sido posible sin el apoyo, en primer lugar, del Dr. Miguel Ángel Reyes Cortés, quien a lo largo de dos años trabajó conmigo, explicándome una y otra vez los conceptos para que me quedaran claros. Quiero agradecer además a los profesores del Departamento de Actuaría, Física y Matemáticas de la universidad; a mis sinodales Daniela Cortés Toto y Rubén Blancas Rivera, además del profesor Freddy Palma Mancilla.

Por otro lado, quiero agradecerle a mi mamá por haber creído en mi desde el momento en el que apliqué a la universidad, por traerme el desayuno mientras escribía este trabajo, cuidar que no me desvelara y apoyarme en los momentos de mayor estrés. A mis amigos Brenda, Georgette, Sofia, Marian y Emilio, por darme ánimos en los momentos en los que sentía que ya no podía y hacerme reír para relajarme.

Finalmente, quiero agradecerle a Ali, mi novio, por apoyarme desde los 17 y crecer conmigo. Gracias por desvelarte conmigo en videollamada, por proporcionarme café innumerables veces y por quererme tanto.

Índice

Capítulo 1

Introducción

El concepto de *función de densidad* es una herramienta esencial para el análisis de datos. La estimación de la función de densidad de probabilidad f (llamada simplemente densidad o función de densidad a lo largo del texto) es un problema fundamental en la estadística, pues se busca modelar la distribución de un conjunto de datos observados. En términos simples, implica estimar la función de densidad parental f de donde provienen las observaciones muestrales. Esta función proporciona información sobre la estructura probabilística de los datos, lo que permite realizar inferencias estadísticas, simular escenarios o muestras aleatorias, así como realizar predicciones.

La relevancia de la estimación de densidad f se extiende a una amplia gama de campos:

- Inferencia estadística: Resulta fundamental para comprender la distribución de los datos y realizar inferencias sobre parámetros desconocidos. Por ejemplo, en el análisis de datos en el área de salud, la estimación de la densidad de probabilidad puede ayudar a modelar la distribución de variables como la presión arterial o los niveles de glucosa (Banzi et al., 2021)
- Aprendizaje automático: Para esta área, la estimación de densidad se utiliza para construir modelos predictivos robustos. Por ejemplo, en la clasificación de imágenes, comprender la distribución de características como la intensidad de los pixeles puede ayudar a la clasificación de los objetos (Arai, 2020).

- Ciencia de datos: La estimación de densidad f se utiliza para explorar y entender conjuntos de datos complejos. Por ejemplo, en datos de índole financiera, la estimación de la función de densidad puede revelar patrones ocultos en los precios de las acciones o las tasas de rendimiento (Das et al., 2024).

Éstos tres ejemplos son áreas donde puede ser de interés la estimación de la densidad, pero existen muchas más, tales como ciencias biológicas, psicología, educación o economía (Efron & Tibshirani, 1995).

En las ciencias actuariales, un ejemplo concreto del uso de la densidad f surge en la administración de riesgos, pues el *Valor en el Riesgo,* también llamado VaR, se define como las pérdidas potenciales de un portafolio en un horizonte de tiempo con cierto nivel de confianza (Jorion, 1996). Su cálculo puede hacerse desde distintos enfoques, tales como: datos históricos, simulación Montecarlo o metodología de la varianza y covarianza.

De acuerdo con John C. Hull (2003), para calcular el VaR con datos históricos, basta con tomar los precios de los activos del portafolio en los últimos 501 días, con los cuales se consigue el rendimiento diario. Una vez obtenido, se ordenan los valores de menor a mayor, lo que puede considerarse como una aproximación empírica (también llamada *probability distribution)* de la función de densidad (pdf) proyectada por medio de un histograma (debido a que las observaciones son discretas). Con dicha función de densidad es posible computar el valor al riesgo, que es el valor de la pérdida en el percentil correspondiente al nivel de confianza deseado, durante el periodo de tiempo usado.

Otro ejemplo consiste en que, para las aseguradoras, es deseable minimizar la cantidad de siniestros con sumas aseguradas elevadas dentro del total de siniestros. Es decir, se busca que la distribución de pérdidas tenga la mayor densidad en 0, pues se espera pagar lo menos posible. Es por esto que resulta atractivo en el modelo colectivo la creación de una distribución por partes, o también llamada mixta, compuesta por una parte discreta y una continua (Dickson, 2005). En el estudio de este modelo es necesario el cálculo de primas p, con un total de n asegurados, el cual, siguiendo el principio de la prima pura, resulta que $E[Y] = np$, y posteriormente es modificado con la intención de garantizar ganancias para la institución. En este caso, para poder calcular el valor esperado que la aseguradora deberá pagar, es necesario conocer la función de densidad de las reclamaciones (o pérdidas).

Un último ejemplo proviene del área de la demografía. En su artículo *The form of the regional density function,* John B. Parr (2007) demostró que la función de densidad de una distribución log-normal era capaz de reproducir las características poblacionales de un área metropolitana, por medio del análisis del desarrollo de la estructura espacial de regiones altamente urbanizadas.

De lo anterior, queda claro que la función de densidad representa una herramienta esencial para el análisis de datos, destacando la importancia crítica de su estimación a partir de una muestra aleatoria.

Para estimar la densidad de probabilidad f existen dos enfoques: el paramétrico y el no paramétrico. El enfoque paramétrico asume una forma funcional específica para la distribución parental f, y se estiman los parámetros de esta distribución a partir de la

muestra. Por ejemplo, asumiendo que los datos siguen una distribución normal, se estiman la media y la desviación estándar, ya sea con algoritmos máximo-verosímiles o por el método de los momentos.

Proceder paramétricamente tiene sus ventajas y desventajas. En cuanto a las primeras, puede decirse que al asumir una forma funcional de la densidad se tiene mayor eficiencia computacional y mayor interpretabilidad. En cuanto a las desventajas, puede mencionarse que la asunción de un modelo paramétrico puede resultar algo rígido, y si el supuesto distribucional es incorrecto, eso puede llevar a inferencias erróneas (Deshpande et al, 2017).

Por otra parte, el enfoque no paramétrico consiste en no asumir una forma funcional específica de la distribución de la que provengan los datos, entonces estiman la densidad directamente a partir de los datos observados, sin imponer restricciones sobre la forma de la distribución. Lo anterior deriva en mayor flexibilidad en la estimación, así como mayor robustez, además de que, por su naturaleza, pueden manejar adecuadamente información categórica, agregando más elementos a su versatilidad y aplicabilidad (Abdulazeez, 2014); sin embargo, tienen como desventaja una mayor complejidad computacional y al no haber parámetros específicos, la interpretación de los resultados se vuelve algo más desafiante.

Este documento tiene como objetivo investigar y comparar el desempeño de dos algoritmos no paramétricos de estimación de la densidad de probabilidad: la estimación tipo núcleo (*Kernel Density Estimation,* en adelante KDE, por sus siglas en inglés), y el algoritmo *Root-Unroot* (en adelante RU, por sus siglas en inglés).

Ambos algoritmos pueden ser ampliamente utilizados en diversos campos debido a su capacidad para modelar la distribución de datos de manera flexible y sin hacer suposiciones específicas sobre su forma funcional. Sin embargo, aunque ambos métodos comparten el enfoque no paramétrico, es crucial comprender cómo difieren en términos de exactitud y precisión bajo diferentes escenarios de tamaño muestral y rugosidad de la distribución parental f. Así, esta investigación se centrará en una comparación de estos dos algoritmos, con el objetivo de proporcionar una comprensión mas profunda de sus fortalezas y limitación en la estimación de la densidad f.

Justificación

Este trabajo se fundamenta en la necesidad de comprender a fondo el desempeño de algoritmos no paramétricos de estimación de densidad de probabilidad en diferentes escenarios. Ya que, en el campo de la estadística, el aprendizaje automático, la ciencia de datos, riesgos financieros y del sector asegurador, la estimación de densidad de probabilidad es crucial para una variedad de tareas, como la modelización de datos, hasta la toma de decisiones informadas, la elección del algoritmo de estimación adecuado puede ser un desafío, dada la variedad de métodos disponibles. Por lo tanto, esta investigación se justifica por la necesidad de proporcionar una comparación rigurosa entre dos algoritmos no paramétricos de estimación de densidad de probabilidad: el KDE y el RU.

Al abordar esta comparación, esta tesis representa una comprensión más profunda de las fortalezas y limitaciones de cada algoritmo en términos de precisión y eficiencia. Además, esta investigación tiene el potencial de informar la selección de algoritmos para

futuros análisis de datos, contribuyendo así al avance en la aplicación de métodos de estimación de densidad de probabilidad en la práctica estadística y científica.

Objetivos

Objetivos Generales

Comparar el desempeño de los algoritmos KDE y RU, en términos de exactitud y precisión en diferentes escenarios de tamaño muestral y rugosidad de la densidad parental.

Objetivos Específicos

- Analizar la teoría y fundamentos de los algoritmos KDE y RU.
- Implementar los algoritmos de KDE y RU en un entorno de simulación controlado, generando conjuntos de datos simulados con diferentes tamaños muestrales y niveles de rugosidad de densidad parental.
- Establecer métricas de evaluación para medir la exactitud y precisión de las estimaciones de densidad generadas por cada algoritmo.
- Interpretar y analizar los resultados obtenidos para identificar las fortalezas y limitaciones de cada algoritmo en la estimación de densidad.
- Proporcionar recomendaciones sobre el uso adecuado de cada algoritmo en función de las características específicas del conjunto de datos y del problema en cuestión.
- Discutir las implicaciones de los hallazgos para la aplicación práctica de los algoritmos de estimación de densidad en diferentes campos de estudio.

Capítulo 2

Marco Teórico

Función de densidad de probabilidad: conceptos fundamentales

Un variable aleatoria es una función que asigna un valor numérico a los posibles resultados de cierto experimento aleatorio, representando una forma de cuantificar los resultados. Pueden ser de dos tipos, discretas y continuas. Las variables aleatorias discretas son aquellas que pueden tomar un numero contable de valores (tienen un rango finito o un rango infinito contable), a diferencia de las continuas, en las que los valores que pueden tomar son incontables (con un rango no numerable) (Ross, 2010)

En lo subsecuente, se refiere a variables aleatorias continuas. Sea X una variable aleatoria, se define la probabilidad de que X se encuentre en el intervalo $(-\infty, a]$ como

$$F_x(a) = P\ (\ X \leq a) = \int_{-\infty}^{a} f_x(x)\, dx$$

donde f_x representa la función de densidad y F_x , conocida como la función de distribución (Ross, 2010).

En particular, para que una función f pueda considerarse como una función de densidad, debe satisfacer las siguientes propiedades:

- $\int_{-\infty}^{\infty} f_x(x)\, dx = 1$
- $P(a < X < b) = \int_{a}^{b} f_x(x)\, dx$

Estimación no paramétrica

Cuando se recurre a métodos paramétricos de estimación, la selección acertada de una distribución resulta crucial para asegurar la precisión del algoritmo. De lo contrario, los resultados podrían ser imprecisos o incluso contraproducentes. Esta dificultad cobra mayor importancia por el hecho de que, en el mundo real, los datos no necesariamente se ajustan a distribuciones paramétricas predefinidas. Es especialmente relevante señalar que, en campos como la educación o las ciencias sociales, donde la información suele ser categórica, este enfoque resulta poco apropiado.

En respuesta a esta limitación, desde la década de 1950 se popularizó la estadística no paramétrica. En su artículo *Nonparametric Statistics,* Savage (1957) señaló que para el año 1952 había aproximadamente 999 artículos donde se mencionaba la estimación paramétrica, y que, en un lapso de solo 5 años, ese número incrementó a 1500. Esto refleja la creciente necesidad de nuevos y más flexibles métodos para abordar la complejidad de los datos disponibles. Como respuesta a esta demanda, se han desarrollado diversos métodos no paramétricos que ofrecen alternativas flexibles para el análisis de datos en una variedad de contextos.

Histograma

El histograma es posiblemente la herramienta más popular para la estimación de la densidad de probabilidad no paramétrica, gracias a su sencillez y facilidad de comprensión.

Para crear un histograma, se divide la recta real en intervalos de la misma longitud, que representan las categorías en las que se clasificarán los datos. Luego, se determina la

altura de cada intervalo en el eje *Y*, basándose en la proporción de datos que caen dentro de ese intervalo, dividido por la longitud del intervalo en el eje *X*.

Esta herramienta ofrece al usuario el control total sobre la elección, cantidad y ubicación de los intervalos sobre los cuales se dividirá la muestra. Sin embargo, un cambio en el ancho del intervalo puede tener un impacto significativo en todos los demás intervalos. Además, pequeñas modificaciones en la posición de los intervalos pueden resultar en estimaciones muy diferentes de la densidad Scott (2010). Dada la naturaleza subjetiva de esta elección, la representación visual del histograma puede variar considerablemente, lo que puede influir en las estimaciones realizadas (Wand & Jones, 1995).

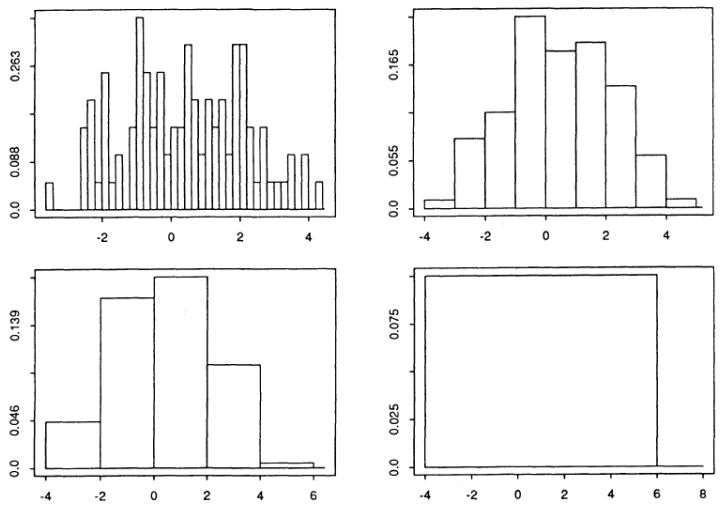

Figura 1. Variación de histogramas según número de intervalos.
Nota: Adaptado de Smoothing techniques (p. 11) por W. Härdle. 1991. Springer-science+business media, b.v.

Estimación tipo núcleo (KDE)

Para abordar la subjetividad que supone la creación de histogramas, surge el método de *Histogramas Desplazados Promedio* (ASH, por sus siglas en inglés) planteado por Scott (1985) en su artículo *Averaged Shifted Histograms: Effective Nonparametric Density Estimators in Several Dimensions*. Este enfoque consiste en promediar *m* posibles histogramas generados al modificar el origen de los intervalos. Como resultado, la estimación obtenida es más suave y no depende del punto de origen de los intervalos (Härdle, 1991).

A partir del concepto de ASH, se desarrolla el método de *Promedio Ponderado de Puntos Redondeados* (WARPing, por sus siglas en inglés), que es esencialmente una generalización de ASH. Según lo propuesto por Härdle (1991), una estimación de densidad realizada mediante WARP se aproxima al método KDE con el kernel de Epanechnikov.

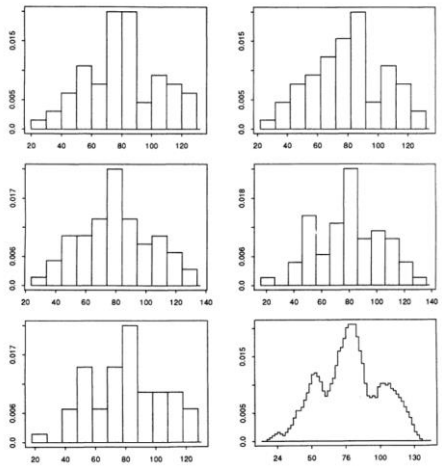

Figura 2. Histogramas por medio de WARPing.
Nota: Adaptado de Smoothing techniques (p. 28) por W. Härdle. 1991. Springer-science+business media, b.v.

Como señala Wergieluk (2021) en su artículo "Histograms vs. KDEs Explained", la función de estimación calculada mediante KDE se asemeja a una versión suavizada de los

histogramas. En lugar de utilizar bloques rectangulares para representar la frecuencia de observaciones, parece que se esparcen granos de arena, lo que realza una estructura más realista de la distribución de los datos.

Los primeros indicios del algoritmo KDE datan de la década de 1950. En algunos campos como el procesamiento de señales y la econometría, también se le conoce como el método de ventana de Parzen-Rosenblatt, en honor a Emanuel Parzen y Murray Rosenblatt, a quienes generalmente se les atribuye la creación independiente de este método en su forma actual (Parzen, E. 1962)

Sea $X_1, X_2, \dots, X_n$ una muestra aleatoria con función de densidad f. El estimador tipo núcleo, KDE, se define como:

$$\hat{f}(x; h) = \frac{1}{nh} \sum_{i=1}^{n} K \left\{ \frac{(x - X_i)}{h} \right\},$$

donde x representa el punto donde se quiere estimar la densidad f, n representa el tamaño muestral, h es un número positivo llamado ventana, y *K* representa una función no negativa, simétrica, llamada núcleo o kernel, cuyo objetivo es ponderar la influencia de las observaciones de la muestra en la estimación de la densidad en el punto x. Además, debe cumplirse que $\int K\,(x)dx = 1$.

En cuanto a h, ésta representa la amplitud que tendrá el kernel; un valor pequeño de h provoca que el kernel sólo tome valores muy cercanos a x, mientras que un valor mayor permitirá que el kernel tome valores lejanos a x, por lo cual su valor es muy importante para la correcta estimación de f.

Cabe mencionar que existen distintos tipos de núcleos, entre los cuales se destacan el de Epanechnikov, el uniforme, el Gaussiano (o normal) y triangular (García-Portugués et al., 2024).

Kernel	$K(u)$
Uniform	$\frac{1}{2} I(\|u\| \leq 1)$
Triangle	$(1 - \|u\|)I(\|u\| \leq 1)$
Epanechnikov	$\frac{3}{4}(1 - u^2)I(\|u\| \leq 1)$
Quartic	$\frac{15}{16}(1 - u^2)^2 I(\|u\| \leq 1)$
Triweight	$\frac{35}{32}(1 - u^2)^3 I(\|u\| \leq 1)$
Gaussian	$\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}u^2\right)$
Cosinus	$\frac{\pi}{4} \cos\left(\frac{\pi}{2}u\right) I(\|u\| \leq 1)$

Tabla 1. Tipos de Kernel.
Nota: Adaptado de Smoothing techniques (p. 45) por W. Härdle. 1991. Springer-science+business media, b.v.

Puede plantearse la pregunta sobre cómo varía el desempeño del algoritmo KDE en función del núcleo elegido. Es posible demostrar que, desde el punto de vista del error cuadrático medio integrado (MISE, por sus siglas en inglés), el kernel óptimo es el de Epanechnikov (Kolassa, 2020).

No obstante, como se puede ver en la Tabla 2, la eficiencia relativa del kernel de Epanechnikov con respecto a los otros núcleos es cercana a uno, lo que significa que, en la práctica, la elección del núcleo es secundaria, no así la elección de h, que es la que principalmente determina el buen funcionamiento del estimador.

Table 2.1. *Efficiencies of several kernels compared to the optimal kernel.*

Kernel	$\{C(K^*)/C(K)\}^{5/4}$
Epanechnikov	1.000
Biweight	0.994
Triweight	0.987
Normal	0.951
Triangular	0.986
Uniform	0.930

Tabla 2. Rendimiento de Kernels.
Nota: Adaptado de Kernel Smoothing (p. 31) por M.P. Wand & M.C Jones. 1991. Springer-science+business media, b.v.

Considere a K* como el núcleo de Epanechnikov. En este contexto, la eficiencia de K en relación con K* se mide por la proporción de datos que K* necesitaría para alcanzar la misma medida de error que se obtendría usando K (Wand & Jones, 1991).

Medidas de Error Local y Global

El error cuadrático medio (MSE, por sus siglas en inglés a partir de *Mean Square Error*) es una medida comúnmente utilizada para evaluar la precisión de un modelo de predicción o estimación. De manera general, si θ es un parámetro que se desea estimar y $\hat{\theta}$ es el algoritmo de estimación, se define el MSE por

$$MSE[\hat{\theta}] = E\left[\left(\hat{\theta} - \theta \right)^2 \right]. \quad (2.1)$$

Considerando que

$$Var(x) = E[x^2] - E[x]^2 , \quad (2.2)$$

se tiene que

$$E[x^2] = E[x]^2 + Var(x), \quad (2.3)$$

con lo cual

$$MSE\left[\hat{\theta}\right] = E\left[\left(\hat{\theta} - \theta\right)^2\right] = E\left[\left(\hat{\theta} - \theta\right)\right]^2 + Var\left(\left(\hat{\theta} - \theta\right)\right). \qquad (2.4)$$

El primer término del lado derecho de la igualdad en (2.4) representa el cuadrado del sesgo de $\hat{\theta}$, mientras que el segundo término representa la varianza de $\hat{\theta}$; es decir,

$$MSE\left[\hat{\theta}\right] = E\left[\left(\hat{\theta} - \theta\right)^2\right] = \mathbb{B}\left(\hat{\theta} - \theta\right)^2 + Var\left(\left(\hat{\theta} - \theta\right)\right). \qquad (2.5)$$

En el contexto de estimación de funciones de densidad, considere a $\hat{f}(x)$ un estimador de la función de densidad f en un punto x, con $x \in \mathbb{R}$. Una medida del desempeño local del estimador $\hat{f}$ está dado por la ecuación (2.5), mediante

$$MSE\left[\hat{f}(x)\right] = \mathbb{B}\left(\widehat{f(x)}\right)^2 + Var\left(\widehat{f(x)}\right). \qquad (2.6)$$

En lugar de estimar f en un solo punto, es común necesitar la estimación de f a lo largo de todo el soporte de la densidad f. Una forma de evaluar el desempeño global de $\hat{f}$ es mediante el error cuadrático integrado (ISE, por sus siglas en inglés, derivado de *Integrated Squared Error*),

$$ISE\left[\hat{f}\right] = \int \left(\hat{f}(x) - f(x)\right)^2 dx \, . \qquad (2.7)$$

La ecuación anterior es una medida global del desempeño del estimador $\hat{f}$ para una realización (es decir, para una muestra). Sin embargo, si lo que se busca es evaluar el performance promedio de $\hat{f}$, debe considerarse el valor esperado de la ecuación anterior; es decir,

$$MISE[\hat{f}] = E\left[ISE[\hat{f}]\right] = E\left[\int \left(\hat{f}(x) - f(x)\right)^2 dx\right]. \qquad (2.8)$$

Nótese que, al cambiar el orden de integración, es posible expresar lo anterior como

$$MISE\left[\hat{f}\right] = \int E\left[\left(\hat{f}(x) - f(x)\right)^2\right] dx = \int MSE\left[\hat{f}\right] dx = \int \mathbb{B}(\widehat{f(x)})^2 dx + \int Var(\widehat{f(x)}) dx \quad (2.9)$$

Criterio de selección de ventana

Un elemento esencial del algoritmo KDE es la elección del parámetro de suavizado o ventana h. En secciones anteriores se mencionó que en el algoritmo KDE, la elección del kernel es secundaria, no así la selección de la ventana h. Este parámetro es crucial, pues determina la suavidad de la estimación: una ventana h muy pequeña genera estimaciones muy rugosas (sobreajuste), mientras que una ventana h muy grande, genera estimaciones muy planas (subajuste). A partir de estos fenómenos, surge la idea de seleccionar la ventana h en su medida adecuada; es decir, óptima desde algún punto de vista.

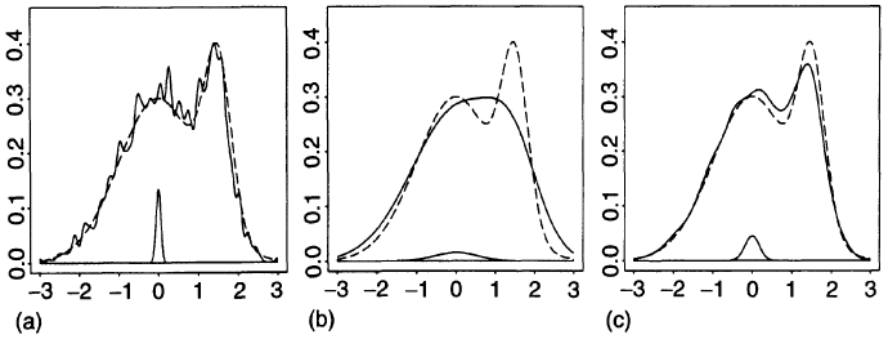

Figura 3. Variación de la ventana
Nota: Adaptado de Kernel Smoothing (p. 13) por M.P. Wand & M.C Jones. 1991. Springer-science+business media, b.v.

A las primeras estrategias que se utilizaron para seleccionar la ventana h se les conoce como reglas empíricas (*rules of thumb*) basadas en supuestos de normalidad. Estas

reglas resultan útiles cuando la estructura probabilística de la muestra se asemeja a una distribución normal. Sin embargo, al aplicar este método en diferentes muestras de una misma distribución, el valor de la ventana puede variar considerablemente (Hall et al., 1991).

Posteriormente, surgieron otras estrategias entre las que destacan métodos automáticos de selección tipo enchufe (*plug-in*) y validación cruzada (*Cross-validation*). Estos enfoques tienen como objetivo minimizar alguna versión efectiva del $MISE\left[\hat{f}\right]$ o de su versión asintótica, el error cuadrático medio asintótico, $AMISE\left[\hat{f}\right]$ (Wand & Jones, 1995).

Históricamente ha existido una discusión sobre cuál de los dos enfoques, *plug-in* o validación cruzada, es el más adecuado. Aunque no existe consenso en la comunidad estadística, se sabe que cualquiera de los dos enfoques puede dar buenos resultados, dependiendo del tipo de estructura probabilística de donde provengan los datos. De acuerdo con algunos estudios de simulación, parece haber evidencia de que los métodos *plug-in* ofrecen mayor estabilidad en la selección de la ventana h (Wand & Jones, 1995).

Selectores *plug-in*

La idea de los selectores *plug-in* se basa en encontrar la ventana h óptima desde el punto de vista del $AMISE$. Para el algoritmo KDE, es posible demostrar que

$$Var\left(\hat{f}(.;h)\right) = (nh)^{-1}R(K)f(x) + o[\,(nh)^{-1}\,], \qquad (2.10)$$

$$AMISE\left[\,\hat{f}(.;h)\right] = (nh)^{-1}R(K) + \frac{1}{4}\,h^4\mu_2(K)^2R(f''), \qquad (2.11)$$

donde $R(K) = \int f^{(2)}(x)f(x)dx$ y $\mu_2(K) = \int z^2K(z)dz$.

Derivando de la ecuación (2.11), igualando a cero y resolviendo para h, es posible obtener una expresión para la ventana optima desde el punto de vista del $AMISE$:

$$h_{AMISE} = \left[\frac{R(K)}{\mu_2(K)^2\psi_4 n}\right]^{\frac{1}{5}}, \qquad (2.12)$$

donde $\psi_r = \int f^{(r)}(x)f(x)dx = E\left[f^{(r)}(X)\right]$. El reto de la ecuación (2.12) es enchufar una estimación adecuada para el funcional ψ_4.

Una solución consiste en proponer un estimador tipo KDE ya no para f, sino para el funcional ψ_4, con posiblemente otro núcleo y otra ventana diferentes de las usadas en el algoritmo KDE para la estimación de la densidad f. Una solución a este problema es la llamada *direct-plug-in* (DPI) o selector *plug-in* de dos etapas, la cual ha resultado ser una de las más populares dentro de este tipo de selectores (Sheather y Jones, 1991).

Existe una variante del algoritmo DPI, conocida como *solve-the-equation* (STE). En este enfoque, se considera que la ventana óptima para estimar el funcional es una función de la ventana óptima para estimar la densidad $f(\gamma(h))$ (Scott, Tapia, Thompson,1977; Sheather, 1986; Park y Marron, 1990; Sheather y Jones, 1991; Hermann et al., 1995). En la práctica, ambos algoritmos proporcionan valores de la ventana razonables para una amplia gama de situaciones, y la elección práctica entre ambos algoritmos puede radicar en elementos de tipo computacional.

Algoritmo RU

El algoritmo RU consiste en transformar el problema de estimación de la densidad f en un problema de regresión. Supongamos que $X_1, \ldots, X_n \sim F$ con densidad $f = F'$. Para simplificar, supongamos que los datos están en $[0, 1]$. Dividamos el intervalo $[0, 1]$ en k intervalos de igual longitud, donde $k \approx \frac{n}{10}$ (Wasserman 2006).

En este contexto, nótese que el número de observaciones en el intervalo j-ésimo, N_j, se puede considerar como una variable aleatoria que se distribuye según una distribución de Poisson de parámetro λ.

Por definición, la probabilidad de que la observación x esté en el j-ésimo intervalo (B_j) está dada por

$$P(x \in B_j) = \int_{B_j} f(x)dx, \tag{2.12}$$

y considerando que $N_j \sim Poisson(\lambda)$,

$$E[N_j] = \lambda = n \int_{B_j} f(x)dx\,; \tag{2.13}$$

es decir, $N_j \sim Poisson\ (\ n \int_{B_j} f(x)dx\ )$. La ecuación (2.13) puede aproximarse mediante:

$$n \int_{B_j} f(x)dx \approx n \left(\frac{1}{k}\right) f(t_j), \tag{2.14}$$

donde k es el número total de intervalos y n el número de observaciones en el j-ésimo intervalo. Esto resulta en que $n_j \sim Poisson \left(n \left(\frac{1}{k}\right) f(t_j)\right)$. Por lo tanto,

$$E[n_j] = Var(n_j) = \lambda \approx n\left(\frac{1}{K}\right) f(t_j). \qquad (2.15)$$

En este caso, es notable que la varianza depende de n y k. Cuando *n* incrementa, la varianza (λ), aumentan consigo, con lo cual es necesario aplicar una transformación que estabilice la varianza.

Para la elección de la transformación estabilizadora existen antecedentes (Hoyle, 1973), (Efon, 1982). Especialmente, para una distribución Poisson, Bartlett (1936) propone $\sqrt{x}$ como la transformación óptima cuando se tenga un modelo lineal homocedástico donde $X \sim Poisson(\lambda)$. Más adelante, Anscombe (1948) propone $\sqrt{x + \frac{3}{8}}$. De manera general, $\sqrt{x + c}$ tiene la propiedad de normalizar a *x*.

Para el algoritmo *root-unroot*, se pretende encontrar una constante c , tal que la media de $\sqrt{x + c}$ sea lo más cercana a $\sqrt{\lambda}$. Con esto se logra tomar control sobre el sesgo y la varianza (Brown et al., 2009)

Según el resultado de Anscombe (1948) y Brown et al. (2009) usando expansiones de Taylor, se obtiene lo siguiente:

Lemma 1 Sea $X \sim Poisson(\lambda)$ con $\lambda > 0$ y $c \geq 0$ una constante.

$$E[\sqrt{x + c}] = \lambda^{\frac{1}{2}} + \frac{4c - 1}{8}\lambda^{-\frac{1}{2}} - \frac{16c^2 - 24c + 2}{128}\lambda^{-\frac{3}{2}} + O\left(\lambda^{-\frac{5}{2}}\right)$$

$$Var(\sqrt{x + c}) = \frac{1}{4} + \frac{3 - 8c}{32}\lambda^{-1} - \frac{32c^2 - 52c + 17}{128}\lambda^{-2} + O(\lambda^{-3}).$$

Con $c = \frac{1}{4}$,

$$E\left[\sqrt{x+\frac{1}{4}}\right] = \lambda^{\frac{1}{2}} - \frac{1}{64}\lambda^{-\frac{3}{2}} + \ O\left(\lambda^{-\frac{5}{2}}\right)$$

$$Var\left(\sqrt{x+\frac{1}{4}}\right) = \frac{1}{4} + \frac{1}{32}\lambda^{-1} + \frac{3}{64}\lambda^{-2} + \ O(\lambda^{-3}),$$

lo que muestra que cuando $c = \frac{1}{4}$ se elimina el sesgo (en primer orden) y se alcanza una varianza casi constante.

En este caso, x representaría el recuento de observaciones en cada intervalo obtenido, entonces la transformación puede verse de la siguiente forma:

$$\sqrt{x+c} = \sqrt{\frac{k}{n}\left(n_j + \frac{1}{4}\right)} = \sqrt{\frac{k}{n}}\sqrt{n_j + \frac{1}{4}} = Y_j \ = \ RU_{transformation.} \quad (2.16)$$

Después se aplica una regresión no paramétrica en la que la variable predictora corresponde a los puntos medios de los intervalos t_j y la variable de respuesta a la transformación Y_j, con la finalidad de obtener una estimación de $\widehat{\sqrt{f(x)}}$.

El algoritmo RU no especifica qué regresión no paramétrica utilizar. Simplemente, se indica que se use la regresión no paramétrica "preferida". En este trabajo se ha decidido utilizar *Splines* de Suavizado (*Smoothing Splines*), una de las técnicas de regresión no paramétrica más populares hoy en día. Esta elección se basa en las numerosas ventajas que ofrece. En primer lugar, por su propia naturaleza, permite capturar relaciones complejas entre variables de manera precisa y sin asumir una forma específica para la relación, lo que los hace muy flexibles y adaptables a diferentes tipos de datos.

Además, los *splines* de suavizado son conocidos por su capacidad para balancear de manera efectiva el ajuste del modelo, asegurando que capture las tendencias principales de los datos sin sobreajustarse a las variaciones menores o el ruido. Esta característica es crucial para mantener la generalización del modelo y su aplicabilidad a nuevos datos.

Otra ventaja significativa de los *splines* de suavizado es su robustez y versatilidad, lo que permite su aplicación en una amplia variedad de contextos y tipos de datos. Este enfoque asegura que el algoritmo sea más confiable y eficiente en diversas situaciones.

Finalmente, solo basta con aplicar la transformada inversa

$$\hat{f} = \left(\widehat{\sqrt{f(x)}}\right)^2, \quad \text{pag 140 wasserman} \tag{2.17}$$

junto con una normalización del estimador $\hat{f}$ obtenida mediante

$$\tilde{f} = \frac{\hat{f}(t)}{\int_0^1 \hat{f}(t)dt}, \tag{2.18}$$

lo que garantiza que la estimación final, $\tilde{f}$, sea efectivamente una densidad.

Rugosidad de las Densidades Parentales

La primera derivada de una función representa la pendiente de la recta tangente a la curva de la función en algún punto, *x,* mientras que la segunda derivada representa la rapidez de cambio de dicha pendiente.

Por otra parte, la integral del cuadrado de la segunda derivada de una función *f*, representada matemáticamente como

$$\int_{sop(x)} f''(x)^2 dx, \qquad (2.19)$$

es una medida global de su curvatura o rugosidad. Esta integral cuantifica cómo cambia la pendiente de la función a lo largo de su dominio $(sop(x))$. Específicamente, la segunda derivada, *f'',* indica la tasa de cambio de la pendiente de la función, es decir, cómo se curva o se dobla la función en cada punto. Al elevar al cuadrado esta cantidad, se asegura que tanto las curvaturas hacia arriba como hacia abajo contribuyan positivamente a la medida total, sin cancelar las variaciones opuestas. Al integrar este cuadrado a lo largo de todo el dominio, se obtiene una suma acumulada de estas contribuciones locales, proporcionando una medida global de la curvatura.

En términos de teoría de probabilidad y estadística, considerar *f* como una densidad de probabilidad nos da una perspectiva interesante sobre la rugosidad de la función. Un valor alto en esta integral indica que la función tiene grandes cambios en la pendiente en muchas partes de su dominio, lo que sugiere una función con mucha oscilación o variación rápida, es decir, una función rugosa. En cambio, un valor bajo

de esta integral indica que la función es más suave, con menos cambios abruptos en su pendiente. Por lo tanto, esta integral no sólo mide la curvatura en términos de magnitud, sino también la frecuencia y distribución de los cambios en la pendiente a lo largo del dominio de la función

Capítulo 3

Metodología

En este apartado se describe el funcionamiento de los algoritmos KDE y RU, así como los criterios empleados para llevar a cabo la comparación de su desempeño. Las simulaciones y el análisis estadístico de los datos se llevaron a cabo utilizando el software R (versión 4.4.0; R Core Team, 2024), un lenguaje de programación específicamente diseñado para computo estadístico, ampliamente reconocido en la comunidad científica por su flexibilidad y potencia en el análisis de datos. Además, se usan los paquetes *splines* (R Core Team, 2023) y *nor1mix* (Maechler, 2019).

Diseño experimental

Considérese una densidad parental g, por simplicidad, con soporte en $[0,1]$. Defínase una rejilla equidistante de valores $\varsigma = (x_1, x_2, \dots x_r)$ en el soporte $[0,1]$. Concretamente, en estas simulaciones se considerará $r = 512$ y un número de repeticiones $B = 1000$.

Caso 1: KDE

En la implementación de las simulaciones con el algoritmo KDE se usó la función *density()* para estimar la densidad de probabilidad. En esta función, se especificó el uso de un kernel gaussiano y el selector de ventana de Sheather y Jones con la variante *Solve the Equation* (STE).

1.- Generar una muestra aleatoria de tamaño n proveniente de g.

2.- Estimar la densidad con el algoritmo KDE (función *density()*), en cada punto de la rejilla $ç$.

3.- Repetir los pasos anteriores B veces.

Como resultado de lo anterior, se obtendrán B estimaciones de la densidad g en cada punto de la rejilla. Para comparar el desempeño de KDE, será necesario comparar, punto a punto, las B estimaciones de la densidad g con los valores de g en la rejilla $ç$.

Caso 2: RU

1.- Dado un tamaño muestral n, dividir el soporte $[0,1]$ en K intervalos, donde $K \approx \frac{n}{10}$.

2.- Generar una muestra aleatoria de tamaño n proveniente de g.

3.- Obtener el número de observaciones n_j en cada intervalo y considerar el punto medio de cada intervalo t_j.

4.- Aplicar la transformación *Root,* según la ecuación (2.16).

5.- Realizar la regresión no paramétrica mediante *splines* de suavizado al conjunto de observaciones $t_i, Y_i, i \in \{1,2, \dots, K\}$ y obtener el valor de la regresión en cada punto de la rejilla $ç$.

6.- Aplicar la transformación inversa, *Unroot.*

En cuanto al paso 5, la regresión mediante *smoothing splines* se realizó con la función *smooth.splines()* de R. El parámetro de suavizado se obtuvo mediante *cross validation*, según la recomendación de algunos autores (Hastie, Tibshirani & Friedman, 2009).

Los siguientes son los parámetros que se consideraron para realizar las simulaciones:

1. Numero de simulaciones

- Se realizarán B simulaciones independientes de muestras aleatorias dada una densidad parental g.

2. Tamaño muestral

- Para evaluar el impacto del tamaño muestral en el desempeño de los algoritmos KDE y RU, se consideraron cuatro tamaños muestrales: $n = (100, 300, 900, 2700)$.

3. Rugosidades

Para evaluar el impacto de la curvatura de la densidad parental g en el desempeño de los algoritmos KDE y RU, se consideraron tres densidades con distinto nivel de rugosidad. Las mixturas de normales de Marron y Wand (1992) fueron diseñadas para evaluar y mejorar los métodos no paramétricos de estimación de la densidad de probabilidad. Proporcionar como referencia densidades con propiedades bien definidas, permite una evaluación rigurosa y comparativa de diferentes técnicas estadísticas. En este trabajo se consideraron tres de las mixturas propuestas por Marron y Wand: normal, bimodal y trimodal. El soporte de las mixturas fue reescalado al intervalo $[0,1]$.

	Media	Varianza	Pesos
Normal	$\frac{1}{2}$	$\frac{1}{8}$	–
Bimodal	$\frac{1}{2}, \frac{11}{16}$	$\frac{1}{8}, \frac{1}{24}$	$\frac{3}{4}, \frac{1}{4}$
Trimodal	$\frac{7}{20}, \frac{13}{20}, \frac{1}{2},$	$\frac{3}{40}, \frac{3}{40}, \frac{1}{32},$	$\frac{9}{20}, \frac{9}{20}, \frac{1}{10},$

Tabla 3. Escala de mixturas. Elaboración propia

- La representación visual de las mixturas

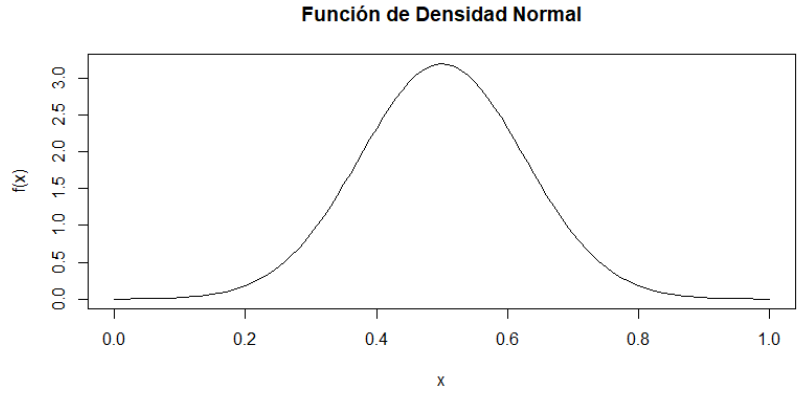

Figura 4. Función de densidad normal. Elaboración propia.

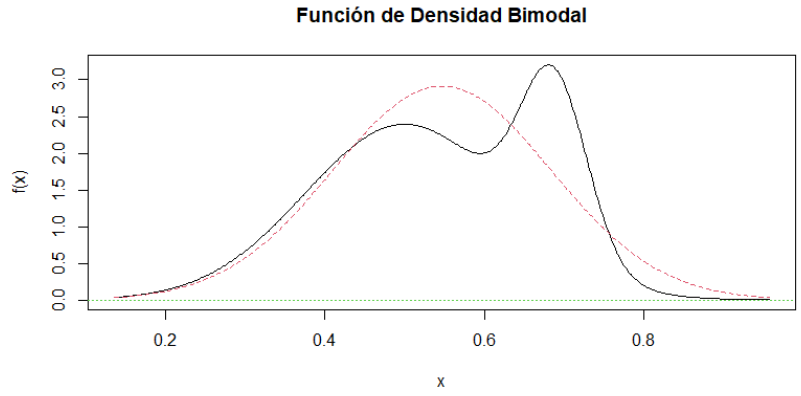

Figura 5. Función de densidad bimodal. Elaboración propia.

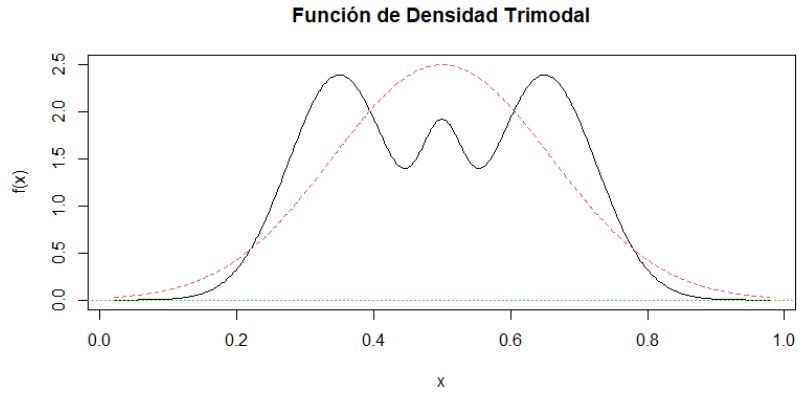

Figura 6. Función de densidad trimodal. Elaboración propia.

Medidas de Rendimiento

Debido a que el soporte de las funciones de densidad está compuesto de una rejilla de 512 puntos sobre el intervalo [0,1], es necesaria una versión aproximada del MSE y el MISE.

En ese sentido, siendo B el número de simulaciones realizadas, para cada simulación, la medida de rendimiento local se aproxima por

$$E\left[\,\hat{f}(x_i)\,\right] \approx \frac{1}{B}\sum_{j=1}^{B}\widehat{f}_j(x_i), \tag{3.1}$$

$$\mathbb{B}\left(\,\hat{f}(x_i)\,\right) \approx \frac{1}{B}\sum_{j=1}^{B}\widehat{f}_j(x_i) - f(x_i). \tag{3.2}$$

La varianza del estimador en el punto i-ésimo puede expresarse como

$$Var\left(\hat{f}(x_i)\,\right) = E\left[\,\hat{f}^2(x_i)\,\right] + \left(E\left[\,\hat{f}^2(x_i)\,\right]\right)^2, \tag{3.3}$$

con lo cual, considerando la ecuación (3.1), una aproximación numérica de la ecuación (3.3) consiste

$$Var\left(\hat{f}(x_i)\right) \approx \frac{1}{B}\sum_{j=1}^{B} \widehat{f}_j^{\,2}(x_i) + \left[\frac{1}{B}\sum_{j=1}^{B} \widehat{f}_j(x_i)\right]^2. \qquad (3.4)$$

De lo anterior se deriva que el MSE del estimador en el punto i-ésimo se puede expresar como

$$MSE\left[\hat{f}(x_i)\right] = \left[\frac{1}{B}\sum_{j=1}^{B} \widehat{f}_j(x_i) - f(x_i)\right]^2 + \left(\frac{1}{B}\sum_{j=1}^{B} \widehat{f}_j^{\,2}(x_i) + \left[\frac{1}{B}\sum_{j=1}^{B} \widehat{f}_j(x_i)\right]^2\right). \qquad (3.5)$$

Las ecuaciones (3.2), (3.4) y (3.5) proporcionan medidas de exactitud, precisión y performance local, respectivamente. Sin embargo, se tiene la necesidad de evaluar el rendimiento de los estimadores de manera global; es decir, a lo largo de todo el soporte. Usando el teorema del valor medio en su forma integral, y considerando la ecuación (3.2), podemos aproximar la ecuación (2.8) como

$$\int_a^b \mathbb{B}^2\left[\widehat{f}(x)\right] dx \approx (b-a)\,\frac{1}{\mathrm{r}}\sum_{i=1}^{c}\left[\frac{1}{B}\sum_{j=1}^{B} \widehat{f}_j(x_i) - f(x_i)\right]^2, \qquad (3.6)$$

mientras que la varianza integrada se aproxima mediante

$$\int_a^b Var\left(\widehat{f}(x)\right) dx \approx (b-a)\,\frac{1}{\mathrm{c}}\sum_{i=1}^{c}\left[\frac{1}{B}\sum_{j=1}^{B} \widehat{f}_j^{\,2}(x_i) + \left[\frac{1}{B}\sum_{j=1}^{B} \widehat{f}_j(x_i)\right]^2\right], \qquad (3.7)$$

con *c* siendo el tamaño de la rejilla.

Finalmente, como el intervalo es $[0,1]$, $(b-a)=1$, con lo cual, la versión aproximada del MISE es

$$MISE\left[\hat{f}(xi)\right] \approx \frac{1}{\mathrm{c}} \sum_{i=1}^{c} \left[\left[\frac{1}{B} \sum_{j=1}^{B} \widehat{f}_j(x_i) - f(x_i) \right]^2 + \frac{1}{B} \sum_{j=1}^{B} \widehat{f}_J^{\,2}(x_i) + \left[\frac{1}{B} \sum_{j=1}^{B} \widehat{f}_j(x_i) \right]^2 \right]. \quad (3.8)$$

Capítulo 4

Resultados

Análisis de Medidas de Rendimiento Global

A continuación, se muestra el error cuadrático medio global (es decir, la aproximación del error cuadrático medio integrado mostrado en la ecuación (3.8)) de los algoritmos KDE y RU en los distintos escenarios de tamaño muestral y rugosidad de la densidad parental.

MISE	**100**		**300**		**900**		**2700**	
	KDE	RU	KDE	RU	KDE	RU	KDE	RU
NORMAL	0.0543	0.0477	0.0219	0.0201	0.0093	0.0066	0.0040	0.0024
BIMODAL	0.0861	0.1243	0.0395	0.0469	0.0173	0.0169	0.0072	0.0064
TRIMODAL	0.0784	0.0959	0.0370	0.0432	0.0163	0.0177	0.0071	0.0072

Tabla 4. MISE. Elaboración propia

.

Visualizaciones Complementarias

- $n = 100$

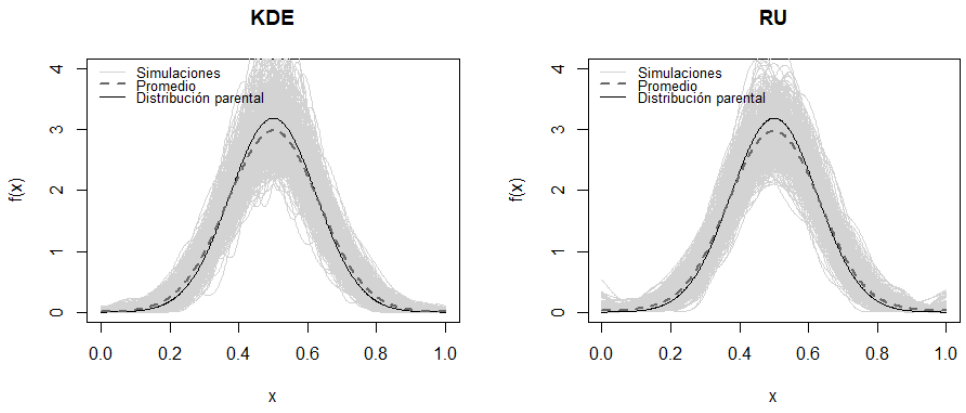

Figura 7. Comparación de algoritmos: caso normal, n=100. Elaboración propia

En general, parece que en el caso RU hay una mayor cercanía de la curva de estimación promedio con la curva de la densidad parental, así como una menor variabilidad de las estimaciones a lo largo del soporte. Lo anterior permite afirmar que el algoritmo RU muestra una superioridad en la estimación de la densidad de probabilidad en este tamaño muestral, relativamente pequeño, y en el caso en el que las muestras provienen de una densidad parental normal.

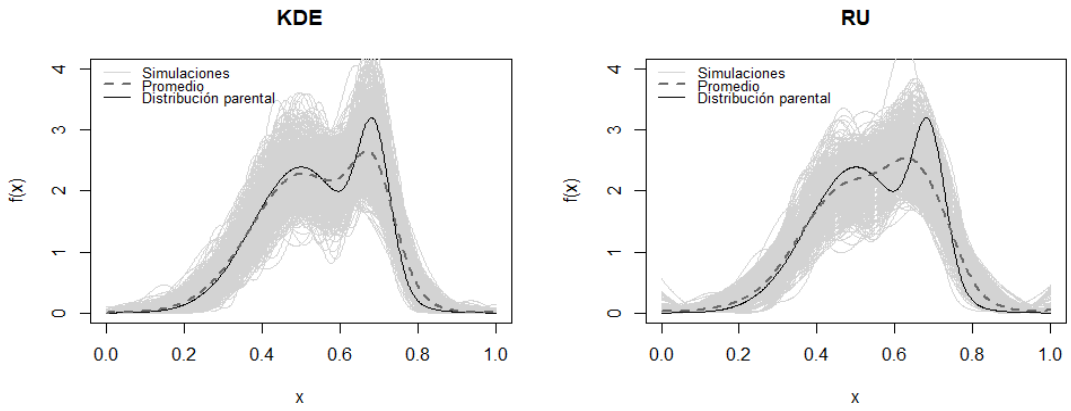

Figura 8. Comparación de algoritmos: caso bimodal, n=100. Elaboración propia.

En el caso de la distribución bimodal considerada, el algoritmo KDE muestra una mayor capacidad para detectar la estructura bimodal en comparación con el algoritmo RU. Este fenómeno es evidente en la silueta formada por las B estimaciones, donde KDE tiende a captar con mayor claridad las dos modas de la distribución.

La curva promedio generada por el algoritmo KDE destaca esta ventaja al presentar los picos correspondientes a la bimodalidad de la estructura probabilística de manera más evidente. En contraste, el algoritmo RU tiende a subajustar esta característica, lo que resulta en una representación muy simplificada de las dos modas presentes en la distribución. En

resumen, aunque el algoritmo RU exhibe ventajas en otros aspectos, KDE demuestra una superioridad específica en la detección y representación de la estructura bimodal.

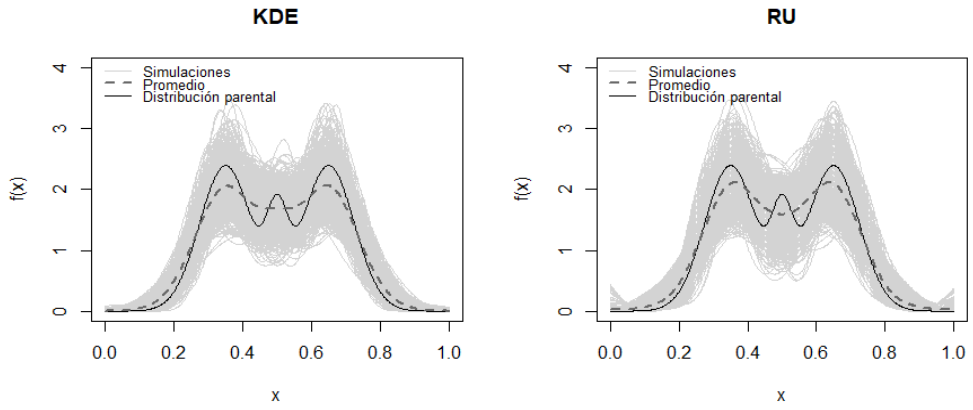

Figura 9. Comparación de algoritmos: caso trimodal, n=100. Elaboración propia.

En el caso de la distribución trimodal, se observa que tanto KDE como RU no logran, en promedio, detectar adecuadamente la estructura trimodal. Esto se refleja en una sobresimplificación de la parte central de la densidad, donde ambos algoritmos tienden a suavizar excesivamente la variación entre las modas. Sin embargo, la silueta de las B estimaciones de la densidad parecen darle una mínima ventaja a KDE sobre RU, ya que, dentro de toda la variabilidad existente en estas estimaciones, KDE detecta levemente la moda central. Todo lo anterior se confirma numéricamente al analizar la tabla 4, donde efectivamente, el algoritmo RU tiene un mejor desempeño (menor MISE) que KDE en el caso normal, pero en los otros dos casos, KDE es el que tiene el mejor desempeño global.

- $n = 300$

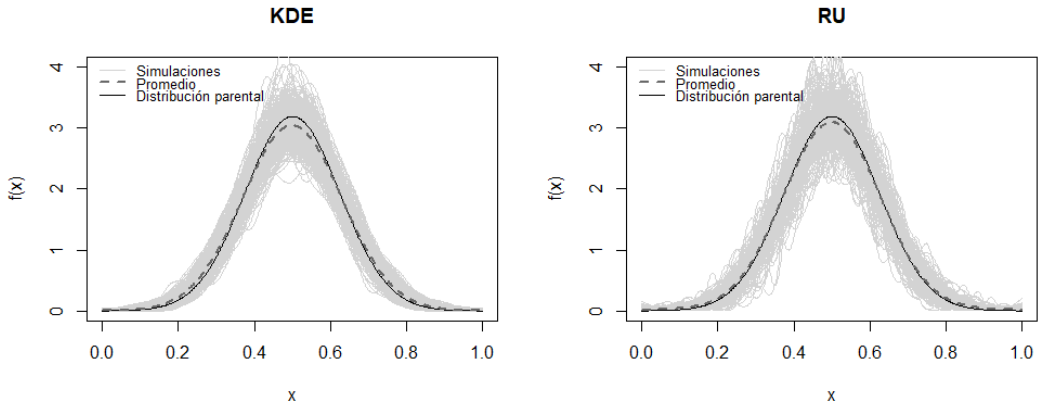

Figura 10. Comparación de algoritmos: caso normal, n=300. Elaboración propia.

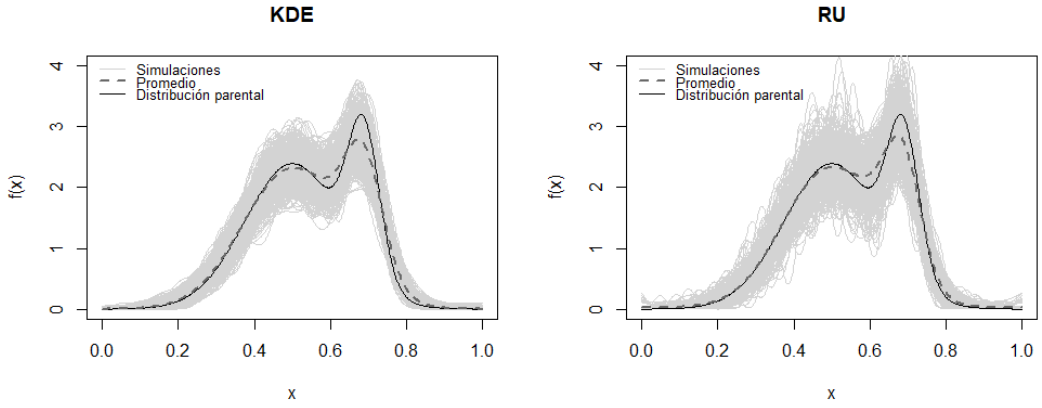

Figura 11. Comparación de algoritmos: caso bimodal, n=300. Elaboración propia.

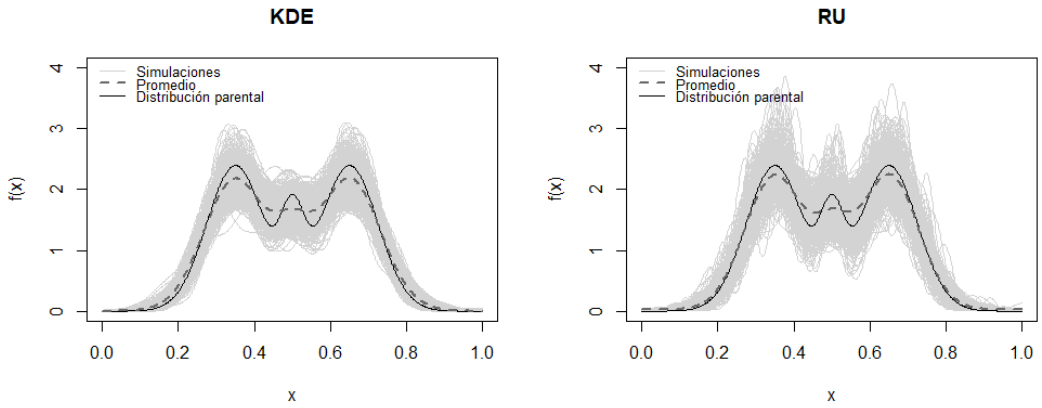

Figura 12. Comparación de algoritmos: caso trimodal, n=300. Elaboración propia.

En el caso de que $n = 300$, se observa un fenómeno similar al anterior. Mientras que RU tiene un mejor desempeño estimando la densidad parental normal, KDE se destaca por estimar mejor las distribuciones bimodal y trimodal. Puede notarse también que la varianza de las estimaciones de KDE es menor que la de RU, y si bien ambos algoritmos tienen dificultad en rescatar la estructura bimodal y principalmente la trimodal, la silueta de las B estimaciones de KDE reproducen de manera más exacta (menor sesgo) y más precisa (menor varianza) dichas estructuras. Nuevamente, este fenómeno tiene sentido al observar el MISE en estas situaciones, comprobándose en la Tabla 4 que el MISE de KDE es menor en los casos bimodal y trimodal, pero es mayor en el caso normal, donde RU se desempeña mejor.

- $n = 900$

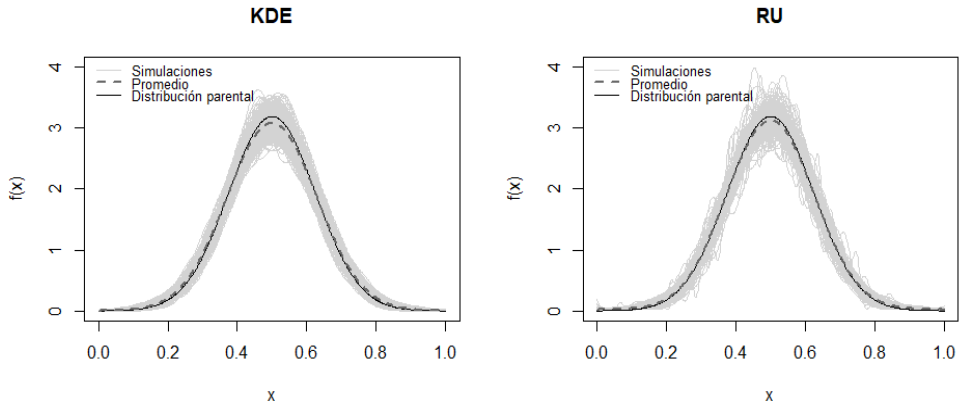

Figura 13. Comparación de algoritmos: caso normal, n=900. Elaboración propia.

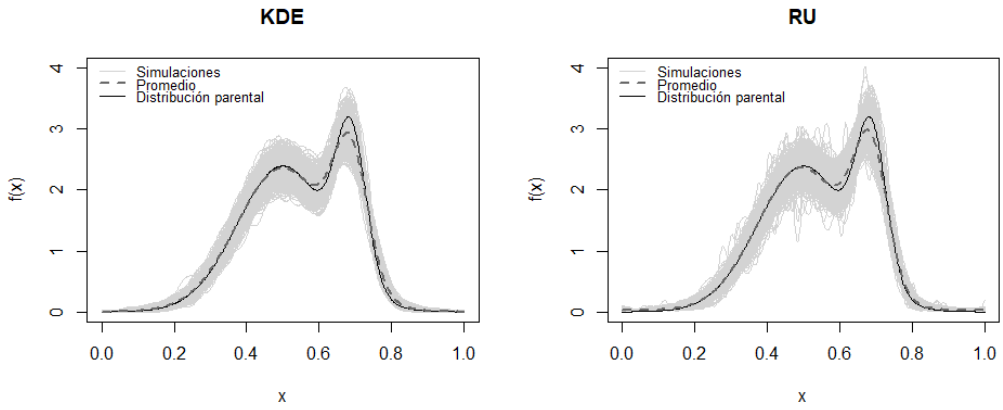

Figura 14. Comparación de algoritmos: caso bimodal, n=900. Elaboración propia.

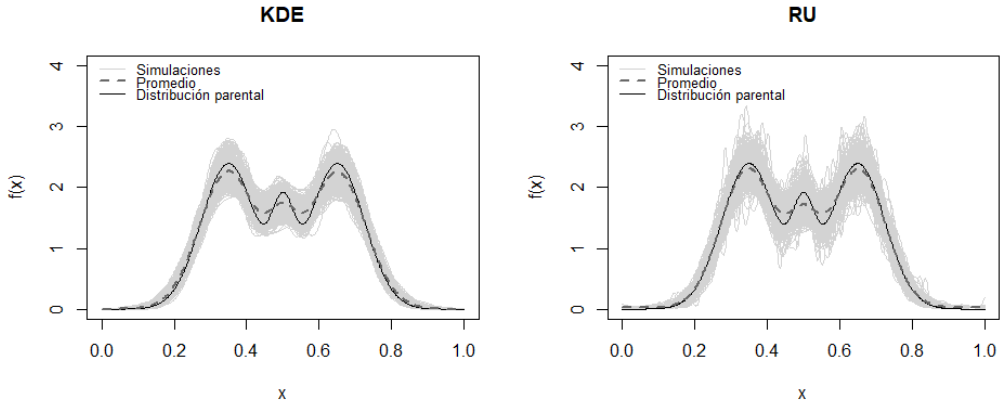

Figura 15. Comparación de algoritmos: caso trimodal, n=900. Elaboración propia.

En este tamaño muestral comienza a no distinguirse visualmente el desempeño de ambos estimadores. Si bien RU muestra algo más de ruido en las estimaciones, en promedio, ambos algoritmos parecen detectar muy bien la densidad parental normal. Estrictamente hablando, RU obtuvo un mejor desempeño en el caso normal al obtener un menor MISE, pero al cambiar a densidades más complejas, se pasa de un prácticamente empate en el caso bimodal, a un mejor desempeño de KDE en el caso trimodal.

En este sentido, es importante destacar que en este tamaño muestral ambos algoritmos ya son capaces de detectar algo mejor la estructura bimodal y trimodal, tanto en promedio como en la silueta de las B estimaciones, teniendo KDE más estabilidad en las estimaciones.

- $n = 2700$

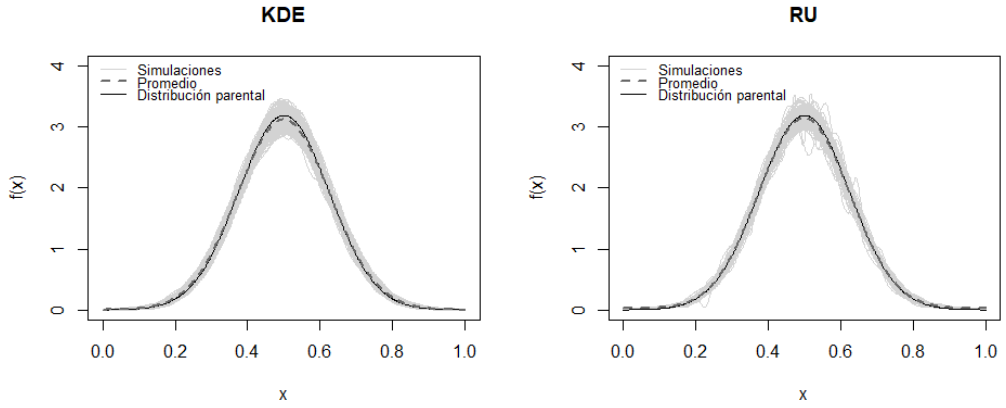

Figura 16. Comparación de algoritmos: caso normal, n=2700. Elaboración propia.

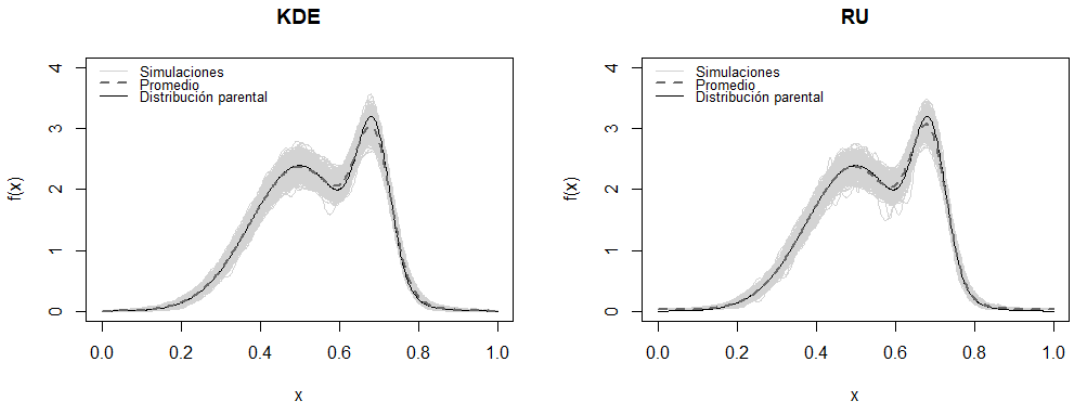

Figura 17. Comparación de algoritmos: caso bimodal, n=2700. Elaboración propia.

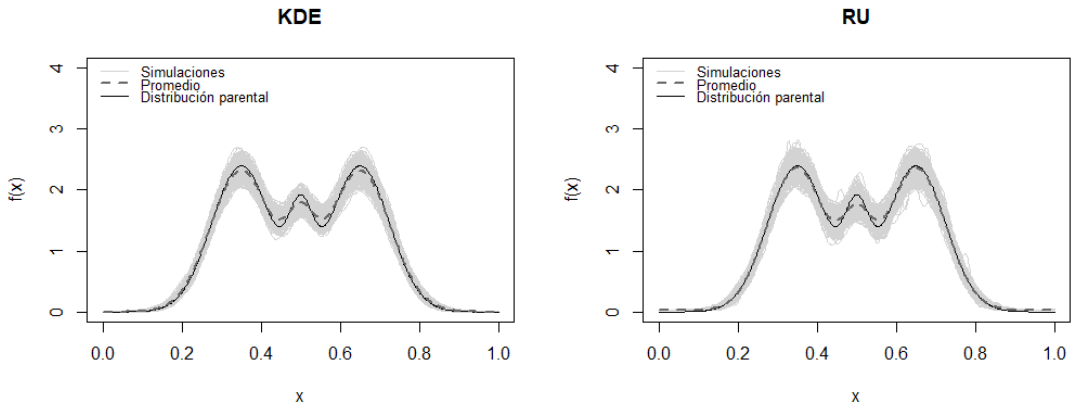

Figura 18. Comparación de algoritmos: caso trimodal, n=2700. Elaboración propia.

En este tamaño muestral, ambos algoritmos se desempeñan notablemente bien, reproduciendo correctamente la distribución unimodal y la estructura más compleja de la bimodal y la trimodal, aunque destacan las estimaciones de KDE por lucir más estables. Dada la escala en que se encuentra el MISE, podemos decir que en el caso normal tiene mejor desempeño RU, y en los otros dos es prácticamente un empate técnico entre KDE y RU. Lo que se observa en los últimos dos tamaños muestrales, 900 y 2700, es esencialmente el fenómeno de consistencia: al aumentar n , tanto el sesgo (al cuadrado e integrado en este caso) como la varianza (integrada en este caso), se desvanecen.

Discusión

Interpretación de Resultados

En el escenario donde la densidad parental es normal, se observa que, para cualquier tamaño muestral, el algoritmo RU tiene un mejor desempeño que KDE. Por lo tanto, se recomienda utilizar RU bajo condiciones de normalidad o similares. Sin embargo, para muestras pequeñas y medianas-pequeñas, KDE sigue siendo competitivo en comparación con RU. Específicamente, en muestras pequeñas, RU supera a KDE en un 14%. Al aumentar el tamaño muestral a una categoría mediana-pequeña, la ventaja de RU sobre KDE se reduce al 9%. Sin embargo, el rendimiento de KDE empeora drásticamente con el aumento del tamaño muestral. En muestras medianas-grandes, KDE es un 40% menos eficiente que RU, y en muestras grandes, el desempeño de KDE es un 67% inferior al de RU.

En el caso de la distribución bimodal, KDE muestra un mejor desempeño que RU, independientemente del tamaño muestral. Para muestras pequeñas y medianas-pequeñas, KDE supera a RU en un 31% y un 16%, respectivamente. Sin embargo, es notable que el rendimiento de RU mejora a medida que aumenta el tamaño muestral. Esto se evidencia en muestras medianas-grandes y grandes, donde la diferencia en desempeño se reduce significativamente: para muestras medianas-grandes, RU es solo un 2% menos eficiente que KDE, lo que puede considerarse prácticamente un empate. Así, parece que, al aumentar el tamaño muestral, el rendimiento de RU tiende a converger con el de KDE.

Para la distribución parental trimodal, KDE presenta el mejor desempeño. Similar al caso bimodal, se observa que el rendimiento de RU mejora a medida que aumenta el

tamaño muestral y converge al comportamiento de KDE en muestras medianas-grandes y grandes. Específicamente, RU es un 18% menos eficiente que KDE para muestras pequeñas, un 14% para muestras medianas-pequeñas, un 8% para muestras medianas-grandes, y solo un 1% para muestras grandes.

El algoritmo RU tiene, en principio, elementos para pensar que podría desempeñarse mejor que KDE en general; sin embargo, no ha sido así. La inclusión de una regresión no paramétrica intermedia en el algoritmo RU invita a pensar que esa flexibilidad se podría heredar al volver al espacio de densidad, dotando de flexibilidad local a la estimación de la densidad, pero no fue así. Sospechamos que uno de los elementos que están entorpeciendo el potencial buen desempeño de RU en tamaños muestrales medianos y grandes, y especialmente en contextos de alta curvatura parental, radica en el número de intervalos seleccionado, lo que pone en entredicho la efectividad del criterio de selección del número de intervalos usado en esta investigación.

Limitaciones del estudio

En esta investigación se realizó una simulación en la que sólo se han abarcado tres casos particulares tratando de representar tres niveles o contextos de rugosidad parental: suave o curvatura baja (normal), suavidad o curvatura media (bimodal) y curvatura alta (trimodal). Debido a esta limitación, la simulación da sólo una idea del comportamiento de los algoritmos a la hora de reproducir características locales específicas de las densidades.

Otra limitación de esta investigación radica en la forma en que se ha abordado la multiplicidad de parámetros de la que depende el algoritmo RU. En primer lugar, el número de intervalos se ha obtenido mediante la relación $K \approx \frac{n}{10}$, sugerida por Wasserman (2006).

En palabras del propio Wasserman, este criterio es relativamente arbitrario, con lo cual se podrían explorar otros criterios de selección del número de intervalos.

Continuando con la complejidad del algoritmo RU, otra limitación es que sólo se ha usado una técnica no paramétrica de regresión consistente en *splines* de suavizado; sin embargo, el desempeño de RU podría verse afectado si se elige otra técnica de regresión no paramétrica.

Otro elemento de complejidad es la selección del parámetro de suavizado del que dependen todas las técnicas no paramétricas de estimación de curvas, y concretamente *splines* de suavizado. En el presente caso se recurrió, por estabilidad computacional, a validación cruzada generalizada, pero pueden plantearse otros criterios para seleccionar el parámetro de suavizado óptimo, lo que podría provocar un desempeño distinto del algoritmo RU.

Finalmente, KDE también plantea algunas limitaciones, pues se ha implementado el algoritmo considerando como selector de ventana el de Sheater y Jones, versión STE; sin embargo, planteando otros escenarios de rugosidad parental, podría estudiarse el desempeño de KDE usando otros criterios de selección automática de la ventana, mismos que podrían provocar un mejor desempeño.

Sugerencias para investigaciones futuras

Es el algoritmo RU el que plantea los retos más complejos por los parámetros de los que depende. Concretamente, investigaciones futuras deberían considerar lo siguiente:

1. Explorar otros criterios para determinar el número (quizá óptimo según el tamaño muestral y la curvatura) de intervalos en los que dividir el soporte.
2. Explorar el desempeño de RU considerando otros algoritmos de regresión no paramétrica, como *Wavelets*, o regresión tipo núcleo, entre otros.
3. Considerar diferentes métodos de selección del parámetro de suavizado más allá de los considerados en esta investigación, tanto para RU como para KDE.

Capítulo 5

Conclusiones

Resumen de hallazgos

En este texto se compararon dos métodos de estimación no paramétrica de densidad: KDE (*Kernel Density Estimation*) y RU (*Root-Unroot + Splines*), evaluando su rendimiento bajo distintas condiciones de tamaño muestral y tipos de distribuciones parentales (normal, bimodal y trimodal). Los hallazgos principales son:

- **Distribución Normal**: El método RU mostró un mejor desempeño general en todos los tamaños muestrales comparado con KDE. La diferencia de rendimiento se acentuó con el incremento del tamaño muestral.
- **Distribución Bimodal**: KDE superó consistentemente a RU en la detección de la rugosidad de la función, especialmente en tamaños muestrales pequeños y medianos. Con tamaños muestrales grandes, la diferencia se redujo, aunque KDE mantuvo una ligera ventaja.
- **Distribución Trimodal**: KDE también tuvo un mejor desempeño en la detección de rugosidad en distribuciones trimodales. Sin embargo, con tamaños muestrales grandes, RU mostró una tendencia a converger en rendimiento con KDE.

Cumplimiento de los objetivos

El objetivo principal de la investigación era comparar el rendimiento de los métodos KDE y RU en diferentes contextos de rugosidad y tamaño muestral. Este objetivo se cumplió de la siguiente manera:

- **Comparación de rendimiento**: Se logró una comparación detallada y cuantitativa del rendimiento de KDE y RU, destacando las condiciones en las que cada método es más efectivo.
- **Evaluación de diferentes tamaños muestrales**: Se analizaron los métodos bajo cuatro tamaños muestrales distintos, proporcionando una visión completa de cómo cada método se comporta con muestras pequeñas, medianas y grandes.
- **Análisis de distintos tipos de rugosidad**: Se evaluaron los métodos en tres tipos de distribuciones parentales, lo que permitió entender mejor sus fortalezas y debilidades en contextos con diferentes niveles de rugosidad.

Importancia de la investigación

La importancia de esta investigación radica en proporcionar una comparación exhaustiva y detallada entre dos métodos de estimación de densidad ampliamente utilizados, lo que tiene implicaciones significativas para el análisis de datos en múltiples disciplinas. Los resultados podrían ayudar a los practicantes de la estadística aplicada a seleccionar el método más adecuado según las características de sus datos, optimizando así la precisión de sus estimaciones.

Implicaciones prácticas y teóricas

- **Implicaciones Prácticas**: Los hallazgos sugieren que, para distribuciones normales o no tan alejadas de esta característica, el método RU es preferible debido a su mayor precisión en todos los tamaños muestrales. Para distribuciones bimodales y trimodales, o bien, con mayor complejidad en términos de curvatura parental, KDE es más adecuado, especialmente con tamaños muestrales pequeños y medianos.

- **Implicaciones Teóricas**: Los resultados contribuyen a la literatura existente sobre métodos de estimación de densidad, proporcionando evidencia empírica-computacional sobre las condiciones bajo las cuales cada método es más efectivo. Esto puede influir en futuras investigaciones y desarrollos analíticos en técnicas de estimación de densidad.

Cierre del trabajo

En conclusión, este estudio proporciona una comparación detallada entre KDE y RU, resaltando las fortalezas y debilidades de cada método en diferentes contextos. Los resultados obtenidos no sólo ayudan a seleccionar el método de estimación adecuado para diferentes tipos de datos, sino que también abren nuevas vías para futuras investigaciones en el campo de la estimación de densidad. El trabajo realizado amplía nuestro conocimiento y ofrece una base sólida para desarrollos posteriores en esta área de estudio.

Capítulo 6

Referencias

Abdulazeez, A. (2014, febrero). Differences and Similarities between Parametric and Non-Parametric Statistics. *GEO 8304: QUALITATIVE AND QUANTITATIVE TECHNIQUES*. Recuperado 9 de abril de 2024, de https://www.academia.edu/14662671/Differences_and_Similarities_between_Parametric_and_Non_Parametric_Statistics

Anscombe, F. J. (1948). The Transformation of Poisson, Binomial and Negative-Binomial Data. Biometrika, 35(3/4), 246. https://doi.org/10.2307/2332343

Arai, K. (2020). Image Classification Considering Probability Density Function based on Simplified Beta Distribution. International Journal Of Advanced Computer Science And Applications/International Journal Of Advanced Computer Science & Applications, 11(4). https://doi.org/10.14569/ijacsa.2020.0110464

Banzi, W., Kambutse, I., Dusabejambo, V., Rutaganda, E., Minani, F., Niyobuhungiro, J., Mpinganzima, L., & Ntaganda, J. M. (2021). Mathematical Modelling of Glucose-Insulin System and Test of Abnormalities of Type 2 Diabetic Patients. International Journal Of Mathematics And Mathematical Sciences, 2021, 1-12. https://doi.org/10.1155/2021/6660177

Bartlett, M. S. (1936). The Square Root Transformation in Analysis of Variance. Supplement To The Journal Of The Royal Statistical Society, 3(1), 68. https://doi.org/10.2307/2983678

Berger, J. O. (1980). Statistical decision theory. En *Springer series in statistics*. https://doi.org/10.1007/978-1-4757-1727-3

Brown, L., Cai, T., Zhang, R., Zhao, L., & Zhou, H. (2009). The root–unroot algorithm for density estimation as implemented via wavelet block thresholding. *Probability Theory And Related Fields*, *146*(3-4). https://doi.org/10.1007/s00440-008-0194-2

Das, T., Halder, A., & Saha, G. (2024). Application of Density-Based clustering approaches for stock market analysis. Applied Artificial Intelligence, 38(1). https://doi.org/10.1080/08839514.2024.2321550

Deshpande, J. V., Naik-Nimbalkar, U., & Dewan, I. (2017). *Nonparametric Statistics: Theory and Methods*. https://openlibrary.org/books/OL28615222M/Nonparametric_Statistics

Dickson, D. C. M. (2005). *Insurance Risk and Ruin*. https://doi.org/10.1017/cbo9780511624155

Efron, B. (1982). The Jackknife, the Bootstrap and Other Resampling Plans. https://doi.org/10.1137/1.9781611970319

Efron, B., & Tibshirani, R. J. (1995). An Introduction to the Bootstrap. *Journal Of The Royal Statistical Society. Series A. Statistics In Society/Journal Of The Royal Statistical Society. Series A, Statistics In Society*, *158*(2), 347. https://doi.org/10.2307/2983304

García-Portugués, E., De Micheaux, P. L., Meintanis, S. G., & Verdebout, T. (2024). Nonparametric Tests of Independence for Circular Data Based on Trigonometric Moments. *Statistica Sinica*. https://doi.org/10.5705/ss.202021.0416

Hall, P., Sheather, S. J., Jones, M. C., & Marron, J. S. (1991). On Optimal Data-Based Bandwidth Selection in Kernel Density Estimation. *Biometrika*, *78*(2), 263. https://doi.org/10.2307/2337251

Härdle, W. (1991). Smoothing techniques. En *Springer series in statistics*. https://doi.org/10.1007/978-1-4612-4432-5

Hastie, T, Tibshirani, R.,, Friedman, J. (2001). The Elements of Statistical Learning. New York, NY, USA: Springer New York Inc..

Herrmann, E., Wand, M. P., Engel, J., & Gasser, T. (1995). A Bandwidth Selector for Bivariate Kernel Regression. Journal Of The Royal Statistical Society. Series B, Statistical Methodology, 57(1), 171-180. https://doi.org/10.1111/j.2517-6161.1995.tb02022.x

Hoyle, M. H. (1973). Transformations: An Introduction and a Bibliography. International Statistical Review, 41(2), 203. https://doi.org/10.2307/1402836

Hull, J. (2003). *Options, futures & other derivatives*. http://ci.nii.ac.jp/ncid/BA42935800

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning. En Springer texts in statistics. https://doi.org/10.1007/978-1-4614-7138-7

Jorion, P. (1996). Risk2: Measuring the Risk in Value at Risk. Financial Analysts Journal, 52(6), 47–56. http://www.jstor.org/stable/4479959

Kolassa, J. E. (2020). *An Introduction to Nonparametric Statistics*.

Lowther, J., Keller, G., & Warwick, B. (1997). Statistics for Management and Economics. Journal Of The Operational Research Society, 48(9), 963. https://doi.org/10.2307/3010500

M. S. Bartlett, The Square Root Transformation in Analysis of Variance, *Supplement to the Journal of the Royal Statistical Society*, Volume 3, Issue 1, January 1936, Pages 68–78, https://doi.org/10.2307/2983678

Maechler M (2019). _nor1mix: Normal aka Gaussian (1-d) Mixture Models (S3 Classes and Methods)_. R package version 1.3-0, <https://CRAN.R-project.org/package=nor1mix>.

Marron, J. S., & Wand, M. P. (1992). Exact Mean Integrated squared error. Annals Of Statistics, 20(2). https://doi.org/10.1214/aos/1176348653

Park, B. U., & Marron, J. S. (1990). Comparison of Data-Driven Bandwidth Selectors. Journal Of The American Statistical Association, 85(409), 66. https://doi.org/10.2307/2289526

Parr, J. B. (1985). The form of the regional density function. *Regional Studies*, *19*(6), 535-546. https://doi.org/10.1080/09595238500185521

Parzen, E. (1962). On Estimation of a Probability Density Function and Mode. Annals Of Mathematical Statistics, 33(3), 1065-1076. https://doi.org/10.1214/aoms/1177704472

R Core Team (2023). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Ross, S. M. (2010). *A First Course in Probability*. Prentice Hall.

Savage, I. R. (1957). Nonparametric statistics. *Journal Of the American Statistical Association*, *52*(279), 331-344. https://doi.org/10.1080/01621459.1957.10501392

Scott, D. W. (1985). Averaged Shifted Histograms: Effective Nonparametric Density Estimators in Several Dimensions on JSTOR. *www.jstor.org*, *13*(3). http://www.jstor.org/stable/2241123

Scott, D. W. (2010). Averaged shifted histogram. *Wiley Interdisciplinary Reviews. Computational Statistics*, *2*(2), 160-164. https://doi.org/10.1002/wics.54

Scott, D. W., Tapia, R. A., & Thompson, J. R. (1977). Kernel density estimation revisited. Nonlinear Analysis, 1(4), 339-372. https://doi.org/10.1016/s0362-546x(97)90003-1

Sheather, S. J. (1986b). An improved data-based algorithm for choosing the window width when estimating the density at a point. Computational Statistics & Data Analysis, 4(1), 61-65. https://doi.org/10.1016/0167-9473(86)90026-5

Sheather, S. J., & Jones, M. C. (1991). A Reliable Data-Based Bandwidth Selection Method for Kernel Density Estimation. Journal Of The Royal Statistical Society. Series B. Methodological, 53(3), 683-690. https://doi.org/10.1111/j.2517-6161.1991.tb01857.x

Silverman, B. W., & Jones, M. C. (1989). E. Fix and J.L. Hodges (1951): An Important Contribution to Nonparametric Discriminant Analysis and Density Estimation: Commentary on Fix and Hodges (1951). *International Statistical Review*, *57*(3), 233. https://doi.org/10.2307/1403796

Wand, M. P., & Jones, M. C. (1995). Kernel smoothing. En *Springer eBooks*. https://doi.org/10.1007/978-1-4899-4493-1

Wasserman, L. (2006). All of Nonparametric Statistics. En *Springer eBooks*. https://doi.org/10.1007/0-387-30623-4

Wegman, E. J. (1972). Nonparametric Probability Density Estimation: I. A Summary of Available Methods. *Technometrics*, *14*(3), 533. https://doi.org/10.2307/1267282

Wergieluk, J. (2021, 14 diciembre). Histograms vs. KDEs Explained - Towards Data Science. *Medium*. https://towardsdatascience.com/histograms-vs-kdes-explained-ed62e7753f12

Capítulo 7

Anexos

Descripción de funciones en R

Para la función *density*, los parámetros son:

- "x": Muestra sobre la cual se estimará la densidad
- "bw": Criterio con el cual se escoge la ventana optima
- "kernel": El kernel de preferencia
- "from": El límite a la izquierda de la rejilla/soporte de la estimación
- "to": El límite a la derecha de la rejilla/soporte de la estimación

Para la regresión (por medio de *splines*) se usan la función *smooth.spline* cuyos parámetros son:

- "x": vector que contiene los valores de la variable predictora
- "y": vector que contiene los valores de la variable de respuesta
- "cv": método para seleccionar el parámetro de suavizado

y, la función *predict,* cuyos parámetros son:

- "object": el objecto sobre el cual se va a realizar la predicción
- "x": vector que contiene el soporte/rejilla sobre el cual se realizará la predicción

Creación de mixturas

Para las distribuciones bimodal y trimodal se crean mixturas en R con ayuda de la función *norMix,* cuyos parámetros son:

- “name”: nombre del objeto (mixtura)
- “mu”: vector de valores de la media
- “sigma”: vector de valores de la desviación estándar
- “w”: vector de pesos de los componentes

Para la distribución bimodal se asigna el nombre *AM.8* y para la distribución trimodal *AM.9*

Muestras aleatorias

Para la realización de muestras aleatorias normales en R se usa la función *rnorm,* cuyos parámetros son:

- “m”: número de observaciones de la muestra
- “mean”: valor de la media
- “sd”: valor de la desviación estandar

Para la realización de muestras aleatorias bimodales y trimodales en R se usa la función *rnorMix,* cuyos parámetros son:

- “n”: número de observaciones de la muestra
- “obj”: objeto del tipo norMix

Almacenamiento de resultados

El almacenamiento se compone de 6 arreglos de matrices con dimensión 500x512x3, cubriendo todas las combinaciones de tamaño muestral y tipo de distribución.

Splines

Los *splines* de regresión (*Regression Splines*, en inglés) son una técnica de regresión que permite modelar relaciones no lineales entre variables. A diferencia de los modelos de regresión lineal simple, que suponen una relación lineal entre las variables dependientes e independientes, los *splines* de regresión dividen el rango de los datos en segmentos. Estos segmentos se determinan mediante puntos llamados "nodos" (*knots*, en inglés), y en cada segmento se ajusta un polinomio de cierto grado *d*. Por lo general, el grado *d* del polinomio no es muy alto (*d*=3 o menor), lo cual contrasta con el ajuste que se realizaría con, por ejemplo, una regresión polinómica, en la que se logra una flexibilidad global notable a cambio de un ajuste de un polinomio de grado muy alto.

Para garantizar que los polinomios ajustados en cada segmento están conectados suavemente en los nodos, se establecen condiciones de frontera para que coincidan en los puntos de unión; concretamente, se pide continuidad en la función y en sus derivadas hasta orden *d*-1, lo cual asegura que el modelo global resultante sea continuo y suave. La elección del número y la ubicación de los nodos es crucial, ya que puede afectar significativamente el ajuste del modelo. Un número excesivo de nodos puede llevar a un sobreajuste, mientras que muy pocos pueden oscurecer o simplificar en exceso la relación de dependencia entre la predictora y la respuesta.

La selección automática del número de nodos y su ubicación se puede realizar mediante varios métodos, y muchos de estos están incorporados en las funciones de R. Algunos de estos métodos son:

- División Cuantilada: Coloca los nodos en posiciones que dividen los datos según sus cuantiles.

- Validación Cruzada: Prueba diferentes configuraciones de nodos y selecciona la que minimiza el error de validación cruzada, proporcionando un buen equilibrio entre ajuste y generalización.
- Algoritmos de Optimización: Emplea algoritmos como el gradiente descendente para encontrar la ubicación óptima de los nodos minimizando una función de pérdida.

En R, estos métodos están implementados en funciones específicas como ns() para natural *splines* y bs() para B-*splines* del paquete *splines*, o a través de paquetes más avanzados como mgcv para *splines* de suavizado. Estas funciones y paquetes automatizan el proceso de selección de nodos, facilitando su aplicación y mejorando la precisión y capacidad de generalización del modelo.

Por otro lado, los *splines* de suavizado son una técnica que busca encontrar un balance óptimo entre el ajuste (cercanía) del modelo a los datos y la suavidad del ajuste. A diferencia de los *splines* de regresión, que requieren una elección explícita de los nodos, los *splines* de suavizado utilizan todos los puntos de datos como posibles nodos, pero aplican una penalización a la curva ajustada para evitar el sobreajuste.

El balance entre ajuste y suavidad se controla mediante un parámetro de suavizado λ. Un λ grande implica una mayor penalización a la variabilidad, produciendo una curva más suave y lineal. En cambio, un λ pequeño permite que la curva siga más de cerca a los datos, potencialmente capturando más variaciones locales. La selección del valor óptimo de λ es clave y, generalmente, se realiza mediante métodos como la validación cruzada.

La principal diferencia entre los *splines* de regression y *splines* de suavizado radica en cómo se maneja la flexibilidad del modelo y la complejidad computacional. Los *splines* de regresión son más intuitivos y directos cuando se sabe aproximadamente dónde se necesitan los nodos. Sin embargo, se requieren decisiones explícitas sobre la ubicación de estos nodos y pueden ser menos eficientes computacionalmente en presencia de grandes cantidades de datos. En cambio, los *splines* de suavizado, aunque más automatizados en su ajuste debido al uso de todos los puntos de datos como nodos potenciales, implican resolver un problema de optimización para encontrar el mejor parámetro de suavización. Este método es particularmente útil cuando no se tiene un conocimiento claro de dónde deberían estar los nodos o cuando se quiere un modelo más robusto que evite que el modelo persiga a los datos (es decir, más robusto contra el ruido en los datos).

Finalmente, ambos métodos ofrecen soluciones avanzadas para la modelización de relaciones no lineales, siendo los *splines* de regresión más adecuados para situaciones donde se puede anticipar la estructura subyacente de los datos y los *splines* de suavizado se prefieren cuando se busca un enfoque más flexible y automatizado para el ajuste suave de los datos. Por lo anterior, en este trabajo se decidió usar *splines* de suavizado, para lo cual existen paquetes y funciones de R que facilitan su implementación.

Para detalles más técnicos sobre el tema, véase (Hastie, T. et al, 2001)

Conformación de MISE

La medida de error global MISE está compuesta por la suma del sesgo cuadrado integrado (SCi) con la varianza integrada (VARi). Las tablas 5 y 6 cuantifican las características observadas en los gráficos 7 a 18 del capítulo 4.

$\mathbb{B}^2$ ***integrado***	**100**		**300**		**900**		**2700**	
	KDE	RU	KDE	RU	KDE	RU	KDE	RU
NORMAL	0.0083	0.0074	0.0040	0.0017	0.0019	0.0007	0.0009	0.0004
BIMODAL	0.0256	0.0699	0.0134	0.0106	0.0051	0.0036	0.0019	0.0014
TRIMODAL	0.0284	0.0279	0.0132	0.0089	0.0059	0.0042	0.0023	0.0015

Tabla 5. Sesgo al cuadrado integrado. Elaboración propia

var integrada	**100**		**300**		**900**		**2700**	
	KDE	RU	KDE	RU	KDE	RU	KDE	RU
NORMAL	0.0460	0.0403	0.0179	0.0184	0.0074	0.0059	0.0031	0.0020
BIMODAL	0.0605	0.0543	0.0262	0.0364	0.0123	0.0132	0.0054	0.0049
TRIMODAL	0.0499	0.0680	0.0239	0.0342	0.0104	0.0135	0.0048	0.0056

Tabla 6. Varianza integrada. Elaboración propia.

Puede verse que, en el caso normal, RU tiene un menor sesgo cuadrado integrado y varianza integrada sin importar el tamaño muestral, lo cual se refleja en un menor MISE en comparación con KDE, tal como se verifica en la tabla 4 del capítulo 4. Sin embargo, en los casos de mayor curvatura parental, independientemente del tamaño, es KDE el algoritmo que tiende a tener menor varianza integrada, lo que concuerda con la descripción realizada

en la página 38, capítulo 4 en relación con la varianza de las estimaciones de RU. Así, las tablas de sesgo cuadrado integrado y varianza integrada ayudan a entender con más profundidad la razón por la que en los casos de mayor curvatura parental, KDE tiende a tener mejor desempeño que RU.

Código en R

```
####### KDE & ROOT-UNROOT PERFORMANCE #######

#librerías
  library(ISLR)
  library(splines)
  library(nor1mix)

#Mixturas
  #1. normal escalada
    #mu_normal=0.5
    #sigma_normal=1/8

  #2.-Mixtura MW.8 escalada
    mu_8=c(1/2,11/16)
    sigma_8=c(1/8,1/24)
    w_8=c(3/4,1/4)
    AM_8<- norMix(name= "AM.8", mu=mu_8, sigma=sigma_8, w=w_8)

  #3.-Mixtura Mw.9 escalada
    mu_9=c(7/20,13/20,1/2)
    sigma_9=c(0.075,0.075,0.03125)
    w_9=c(0.45,0.45,0.10)
    AM_9<- norMix(name= "AM.9", mu=mu_9, sigma=sigma_9, w=w_9)

    e<-seq(0,1, by=0.01)
    plot(e,dnorm(e, mean=0.5, sd=(1/8)), type="l", lwd=1.5, main = "Función de Densidad
Normal", xlab="x", ylab="f(x)")
    plot(AM_8, main = "Función de Densidad Bimodal" , xlab="x", ylab="f(x)")
    plot(AM_9, main = "Función de Densidad Trimodal" , xlab="x", ylab="f(x)")

#Datos
  set.seed(1806)
  n<- c(100,300,900,2700)
  B <- 1000
  xgrid<-seq(0,1, length.out=512)
  ref_normal<-dnorm(xgrid, mean=0.5, sd=(1/8))

#Almacenamiento (6 matrices 500X512)
  n_kde <- array(NA,dim=c(B,length(xgrid), length(n)))
  b_kde <- array(NA,dim=c(B,length(xgrid), length(n)))
  t_kde <- array(NA,dim=c(B,length(xgrid), length(n)))
  n_ru  <- array(NA,dim=c(B,length(xgrid), length(n)))
  b_ru  <- array(NA,dim=c(B,length(xgrid), length(n)))
  t_ru  <- array(NA,dim=c(B,length(xgrid), length(n)))

#Procedimiento: toma de muestras

  #Caso normal
    for(j in 1:length(n))
    {
      for(i in 1:B)
      {
        #obteniendo y limpiando la muestra
        sampl <- rnorm(n[j], mean=0.5, sd=(1/8))
        sample <- sampl[sampl >= 0 & sampl<= 1]

        #KDE
        obj_2 <- density(x = sample, from = 0, to = 1,
                         kernel = "gaussian", bw = "SJ")
        n_kde[i,,j] <- obj_2$y

        ?density
        #RU
        brks<-seq(0,1, length.out= ((n[j]/10)+1) )
```

```
      hist<-hist(sample,breaks=brks,plot=FALSE)
      hist<-hist(sample,breaks=brks,plot=FALSE)
      ni<-hist$counts
      ti<-hist$mids

      yi<- sqrt((n[j]/10)/n[j])*sqrt(ni+0.25)

      fit_smooth<-smooth.spline(ti, yi, cv=FALSE)

      h_hat<-predict(fit_smooth, x=xgrid)
      h_hat$y[h_hat$y<0]<-0
      y<-h_hat$y

      f_hat<-(y^2)/mean(y^2)

      n_ru[i,,j] <- f_hat
    }
  }

#Caso bimodal
  for(j in 1:length(n))
  {
    for(i in 1:B)
    {
      #obteniendo y limpiando la muestra
      sampl <- rnorMix(n[j],AM_8)
      sample<- sampl[sampl >= 0 & sampl<= 1]

      #KDE
      obj_2 <- density(x = sample, from = 0, to = 1,
                       kernel = "gaussian", bw = "SJ")
      b_kde[i,,j] <- obj_2$y

      #RU
      brks<-seq(0,1, length.out= (n[j]/10)+1)
      hist<-hist(sample,breaks=brks,plot=FALSE)
      ni<-hist$counts
      ti<-hist$mids

      yi<- sqrt((n[j]/10)/n[j])*sqrt(ni+0.25)

      fit_smooth<-smooth.spline(ti, yi, cv=FALSE)
      h_hat<-predict(fit_smooth, x=xgrid)
      h_hat$y[h_hat$y<0]<-0
      y<-h_hat$y

      f_hat<-(y)^2/mean((y)^2)

      b_ru[i,,j] <- f_hat
    }
  }

#Caso trimodal
  for(j in 1:length(n))
  {
    for(i in 1:B)
    {
      #obteniendo y limpiando la muestra
      sampl <- rnorMix(n[j],AM_9)
      sample<- sampl[sampl >= 0 & sampl<= 1]

      #KDE
      obj_2 <- density(x = sample, from = 0, to = 1,
                       kernel = "gaussian", bw = "SJ")
      t_kde[i,,j] <- obj_2$y
```

```
        #RU
        brks<-seq(0,1, length.out= (n[j]/10)+1)
        hist<-hist(sample,breaks=brks,plot=FALSE)
        ni<-hist$counts
        ti<-hist$mids

        yi<- sqrt((n[j]/10)/n[j])*sqrt(ni+0.25)

        fit_smooth<-smooth.spline(ti, yi, cv=FALSE)
        h_hat<-predict(fit_smooth, x=xgrid)
        h_hat$y[h_hat$y<0]<-0
        y<-h_hat$y

        f_hat<-(y)^2/mean((y)^2)

        t_ru[i,,j] <- f_hat
      }
    }

#Procedimiento: Obtención de error cuadrático medio global
  #caso normal
    #KDE
      b2_kde<- ( colMeans(n_kde) - ref_normal  )^2
      var_kde_l<-apply(n_kde, c(2,3), var)
      mse_kde_l<-b2_kde + var_kde_l
      mse_kde_gl<-colMeans(mse_kde_l)
    #RU
      b2_ru<- ( colMeans(n_ru) - ref_normal  )^2
      var_ru_l<-apply(n_ru, c(2,3), var)
      mse_ru_l<-b2_ru + var_ru_l
      mse_ru_gl<-colMeans(mse_ru_l)

  #caso bimodal
    #KDE
      b2_kde_b<- ( colMeans(b_kde) - dnorMix(xgrid, AM_8) )^2
      var_kde_l_b<-apply(b_kde, c(2,3), var)
      mse_kde_l_b<-b2_kde_b + var_kde_l_b
      mse_kde_gl_b<-colMeans(mse_kde_l_b)
    #RU
      b2_ru_b<- ( colMeans(b_ru) - dnorMix(xgrid, AM_8) )^2
      var_ru_l_b<-apply(b_ru, c(2,3), var)
      mse_ru_l_b<-b2_ru_b + var_ru_l_b
      mse_ru_gl_b<-colMeans(mse_ru_l_b)

  #caso trimodal
      #KDE
      b2_kde_t<- ( colMeans(t_kde) - dnorMix(xgrid, AM_9) )^2
      var_kde_l_t<-apply(t_kde, c(2,3), var)
      mse_kde_l_t<-b2_kde_t + var_kde_l_t
      mse_kde_gl_t<-colMeans(mse_kde_l_t)
      #RU
      b2_ru_t<- ( colMeans(t_ru) - dnorMix(xgrid, AM_9) )^2
      var_ru_l_t<-apply(t_ru, c(2,3), var)
      mse_ru_l_t<-b2_ru_t + var_ru_l_t
      mse_ru_gl_t<-colMeans(mse_ru_l_t)

#Resultados
      nombres_columnas <- c("100", "300", "900","2700")
      nombres_filas <- c("normal KDE", "normal RU", "bimodal KDE", "bimodal RU", "trimodal
KDE", "trimodal RU")
      # Crear la tabla y asignar nombres a las columnas
      resultados <- as.table(rbind
                             (mse_kde_gl,mse_ru_gl,
                              mse_kde_gl_b,mse_ru_gl_b,
```

```
                              mse_kde_gl_t,mse_ru_gl_t ))
      colnames(resultados) <- nombres_columnas
      rownames(resultados) <- nombres_filas

      round(resultados*10000,0)

      round(resultados, 4)

#Comparando gráficamente
      par(mfrow = c(1, 2))

      ########CASO NORMAL
      plot(xgrid, dnorm(xgrid, mean=0.5, sd=(1/8)), type="l", ylim=c(0,4),
main="KDE",xlab="x", ylab="f(x)" )
      for (i in 1:B){
        lines(xgrid,n_kde[i,,4], type="l", col="lightgray")
      }
      lines(xgrid, dnorm(xgrid, mean=0.5, sd=(1/8)), type="l", lwd=1.5,
main="KDE",xlab="x", ylab="f(x)" )
      media_kde <- apply(n_kde[,,4], 2, mean)
      lines(xgrid,media_kde, lty=2, col="dimgray", lwd=2)
      legend("topleft", legend=c("Simulaciones", "Promedio", "Distribución parental"),
             col=c("lightgray", "dimgray", "black"), lty=c(1, 2, 1), lwd=c(1, 2, 1),
bty="n", cex=0.8)

      plot(xgrid, dnorm(xgrid, mean=0.5, sd=(1/8)), type="l", ylim=c(0,4),
main="RU",xlab="x", ylab="f(x)" )
      for (i in 1:B){
        lines(xgrid,n_ru[i,,4], col="lightgray")
      }
      lines(xgrid, dnorm(xgrid, mean=0.5, sd=(1/8)), type="l", lwd=1.5,
main="RU",xlab="x", ylab="f(x)" )
      media_ru <- apply(n_ru[,,4], 2, mean)
      lines(xgrid,media_ru, lty=2, col="dimgray", lwd=2)
      legend("topleft", legend=c("Simulaciones", "Promedio", "Distribución parental"),
             col=c("lightgray", "dimgray", "black"), lty=c(1, 2, 1), lwd=c(1, 2, 1),
bty="n", cex=0.8)

      ########CASO BIMODAL
      plot(xgrid, dnorMix(xgrid, AM_8), type="l", ylim=c(0,4), main="KDE",xlab="x",
ylab="f(x)" )
      for (i in 1:B){
        lines(xgrid,b_kde[i,,4], type="l", col="lightgray")
      }
      lines(xgrid, dnorMix(xgrid, AM_8), type="l", lwd=1.5)
      media_kde <- apply(b_kde[,,4], 2, mean)
      lines(xgrid,media_kde, lty=2, col="dimgray", lwd=2)
      legend("topleft", legend=c("Simulaciones", "Promedio", "Distribución parental"),
             col=c("lightgray", "dimgray", "black"), lty=c(1, 2, 1), lwd=c(1, 2, 1),
bty="n", cex=0.8)

      plot(xgrid, dnorMix(xgrid, AM_8), type="l", ylim=c(0,4), main="RU", xlab="x",
ylab="f(x)")
      for (i in 1:B){
        lines(xgrid,b_ru[i,,4], col="lightgray")
      }
      lines(xgrid, dnorMix(xgrid, AM_8), type="l", lwd=1.5)
      media_ru <- apply(b_ru[,,4], 2, mean)
      lines(xgrid,media_ru, lty=2, col="dimgray", lwd=2)
      legend("topleft", legend=c("Simulaciones", "Promedio", "Distribución parental"),
             col=c("lightgray", "dimgray", "black"), lty=c(1, 2, 1), lwd=c(1, 2, 1),
bty="n", cex=0.8)


      ########CASO TRIMODAL
```

```
      plot(xgrid, dnorMix(xgrid, AM_9), type="l", ylim=c(0,4), main="KDE",xlab="x",
ylab="f(x)" )
      for (i in 1:B){
        lines(xgrid,t_kde[i,,4], type="l", col="lightgray")
      }
      lines(xgrid, dnorMix(xgrid, AM_9), type="l", lwd=1.5)
      media_kde <- apply(t_kde[,,4], 2, mean)
      lines(xgrid,media_kde, lty=2, col="dimgray", lwd=2)
      legend("topleft", legend=c("Simulaciones", "Promedio", "Distribución parental"),
             col=c("lightgray", "dimgray", "black"), lty=c(1, 2, 1), lwd=c(1, 2, 1),
bty="n", cex=0.8)

      plot(xgrid, dnorMix(xgrid, AM_9), type="l", ylim=c(0,4), main="RU", xlab="x",
ylab="f(x)")
      for (i in 1:B){
        lines(xgrid,t_ru[i,,4], col="lightgray")
      }
      lines(xgrid, dnorMix(xgrid, AM_9), type="l", lwd=1.5)
      media_ru <- apply(t_ru[,,4], 2, mean)
      lines(xgrid,media_ru, lty=2, col="dimgray", lwd=2)
      legend("topleft", legend=c("Simulaciones", "Promedio", "Distribución parental"),
             col=c("lightgray", "dimgray", "black"), lty=c(1, 2, 1), lwd=c(1, 2, 1),
bty="n", cex=0.8)
```