Máster en Estadística Aplicada

Departamento de Estadística e Investigación Operativa

Universidad de Granada

Trabajo fin de máster

Estimación de Densidades Multivariantes Mediante Funciones de Tipo Núcleo

Daniel Bacaicoa Barber

Granada, septiembre de 2021

Máster en Estadística Aplicada

Departamento de Estadística e Investigación Operativa

Universidad de Granada

Trabajo de investigación presentado por Don. Daniel Bacaicoa Barber y dirigido por la profesora Dra. Dña. María Dolores Martinez Miranda

VºBº

Maria dolores Martinez Miranda

Daniel Bacaicoa Barber

Índice

Índice de figuras

1. Introducción

Cuando pensamos en el análisis exploratorio de una función de densidad de una variable desconocida, el primer método en el que normalmente pensamos, es el de construir un histograma. Un método que según (Scott, 2015), se remonta al S.XVII.

En este trabajo sin embargo, estamos interesados en indagar en otro método, si bien muy relacionado con el del histograma tal como pone de manifiesto (Chacón y Duong, 2018). Nos centraremos en el estudio de el estimador de tipo Kernel de funciones de densidad. Este estimador propuesto inicialmente por (Parzen, 1962; Rosenblatt, 1956) ha ido evolucionando y atrayendo el interés investigador de muchos autores.

Así, desde el estimador con un objetivo meramente exploratorio que (Silverman, 1986), el estimador y sus aplicaciones han crecido exponencialmente. El estimador univariante es bien conocido a día de hoy pero el caso multivariante es más desafiante.

Desde el estimador (Wand y Jones, 1994) con matrices restringidas a los selectores de banda más modernos como (Chacón y Duong, 2010) ha habido un desarrollo importante del estimador y de sus propiedades. Así como de los selectores de ancho de banda, la estimación de derivadas o la aplicación de la estimación de densidades al mundo de la estadística, las matemáticas u otras como las ciencias de la computación o la ciencia de datos.

Investigar sobre el la estimación de densidades de tipo Kernel tiene un interés importante. Hoy en día es un área de investigación que genera artículos novedosos. En este sentido, mucha de la investigación sobre estimación de densidades multivariantes es reciente y está impulsada por los motores computacionales más potentes.

Asimismo, las aplicaciones derivadas de la estimación de densidades sigue dando muchos artículos de investigación como demuestran, (Casa y col., 2019) que estudian la selección del ancho de banda para realizar agrupamientos basados en la densidad.

Uno de los objetivo principales que me impulsaron a adentrarme en este trabajo fue el de suplir un vacío que había, al menos hasta donde alcanza mi conocimiento, en la estimación de densidades mediante funciones de tipo núcleo en Python. Así como R sí que tiene implementados paquetes como (Duong y col., 2007), Python únicamente tiene implementadas alguna función que permite la estimación de densidades de tipo núcleo, pero no ofrece la opción de seleccionar ni la función de tipo Kernel a emplear ni el método de selección del ancho de banda. El primer problema no resulta especialmente grave ya que diversos autores señalan que la importancia al seleccionar la función de tipo Kernel no influye sustancialmente en la estimación, sin embargo, sí que lo hace el ancho de banda. Es por esto que resulta de interés crear funciones que permitan una selección del ancho de banda a realizar la estimación de densidad de tipo Kernel.

El otro, el hecho de adentrarme de forma profunda y rigurosa en un tema del que, había oido hablar tanto desde el punto de vista matemático como en el de aprendizaje máquina, mucho más de moda en los días que corren.

El presente trabajo se estructura de la siguiente manera. En el Capítulo 2 se realiza un estudio del estimador de densidad de tipo núcleo univariante. En la sección 2.2 se muestran las principales propiedades del estimador, en la sección 2.3 se muestran los principales estimadores del ancho de banda del estimador y en la sección 2.5 se realiza una comparación de dichos estimadores sobre una muestra sintética.

En el Capítulo 3 se realiza el estudio del estimador de densidad de tipo núcleo multivariante. Como en el caso univariante, en la sección 3.2 se muestran las principales propiedades del estimador, en la sección 3.3 se muestran los principales estimadores del ancho de banda del estimador y en la sección 3.5 se realiza una comparación de dichos estimadores sobre una muestra bivariante sintética.

En el capítulo 4, se muestra brevemente la estimación de derivadas de la función de densidad, así como sus principales propiedades. En el 5º, se muestra la estimación de densidades mediante el algoritmo de vecinos más próximos. Mostramos la similitud que presenta este método con el estimador de tipo kernel que ocupa este trabajo.

Por último, en el capítulo 6, realizamos la estimación de tipo núcleo sobre datos reales tanto de manera univariante como bivariante. Finalizamos el trabajo con una sección en la que se detallan las conclusiones obtenidas durante la realización de este trabajo y las posibles líneas de investigación futuras que podrían seguirse en el futuro.

Adicionalmente, en los Anexos II y III se puede encontrar el código empleado en el trabajo con los ejemplos que se presentan en las secciones 2.5 y 3.5. El código está escrito en un *Notebook* de *Jupyter* para realzar su interactividad y facilitar su seguimiento al mostrar tanto código de entrada como salidas.

2. Estimación de la densidad unidimensional tipo núcleo

El objetivo de la estimación no paramétrica de densidades es el de estimar la función de densidad generadora de la muestra f estableciendo el mínimo número posible de hipótesis sobre f.

2.1. Definición del estimador

Sea $X_1, \ldots, X_n$ una muestra aleatoria procedente de una distribución de probabilidad (univariante) continua con función de densidad f.

El estimador de densidad de tipo núcleo se define como:

$$\hat{f}(x;h) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right) \tag{2.1}$$

donde K, el núcleo, es una función que satisface $\int_{\mathbb{R}} K(x)dx = 1$ y h, el ancho de banda, es un parámetro positivo, $h > 0$.

La función núcleo se puede reescalar, obteniendo su versión reescalada, $K_h(u) = \frac{1}{h}K(\frac{u}{h})$. Se puede comprobar que esta versión reescalada del núcleo, en efecto, es una función núcleo.

$$\int_{\mathbb{R}} K_h(u)du = \int_{\mathbb{R}} \frac{1}{h}K\left(\frac{u}{h}\right)du = \int_{\mathbb{R}} \frac{1}{h}K(x)hdx = \int_{\mathbb{R}} K(x)dx = 1$$

Con esto, se puede reescribir el estimador de densidad de tipo núcleo (2.1) como:

$$\hat{f}(x;h) = \frac{1}{n}\sum_{i=1}^{n} K_h\left(x - X_i\right) \tag{2.2}$$

Nótese que no se impone que $K(x) \geq 0$, es decir, no se impone que $K(x)$ sea una función de densidad si bien esto es deseable, ya que, de no imponer esta propiedad al núcleo, no podemos garantizar que el estimador sea una función de densidad.

Una interpretación del estimador de densidad de tipo núcleo como apunta (Wand y Jones, 1994) "Uno puede pensar que el núcleo está extendiendo una 'masa de probabilidad' de tamaño $1/n$ asociada a cada observación sobre su entorno". Gráficamente, esta idea puede representarse de la siguiente manera:

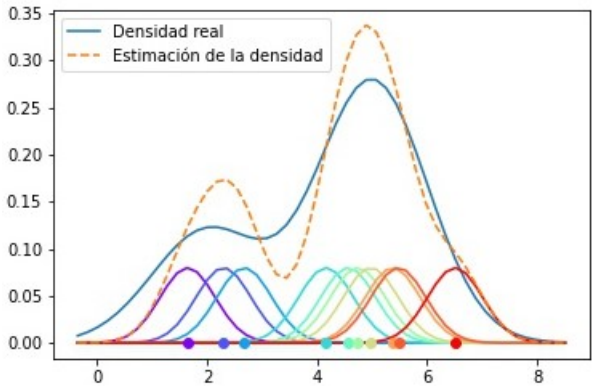


Figura 1: Representación de la función de densidad de una mixtura de normales $0{,}3N(2,1) + 0{,}7N(5,1)$ y su estimación de tipo núcleo para una muestra aleatoria simple de tamaño 10 generada de dicha mixtura. Asimismo, se muestran las observaciones y las funciones de tipo núcleo $N(0,0{,}5)$ reescaladas a un área de $1/10$ centradas en cada observación. La estimación de la densidad resulta de la suma de las funciones núcleo.

Algunas de las funciones de tipo núcleo univariantes más utilizadas son:

- Núcleo uniforme: $K_h(x) = \frac{1}{2h}, \quad |x| \leq h$
- Núcleo triangular: $K_h(x) = \frac{1}{h} - \frac{|x|}{h^2}, \quad |x| \leq h$
- Núcleo normal: $K_h(x) = \phi_h(x) = \frac{1}{h\sqrt{2\pi}} e^{-\frac{x^2}{2h^2}}$. $\phi_\sigma(x)$ corresponde a la función de densidad de una variable aleatoria $N(0,\sigma)$
- Núcleo de Epanechnikov: $K_h(x) = \frac{3}{4h}\left(1 - \frac{x^2}{h^2}\right), \quad |x| \leq h$
- Núcleo Biweight: $K_h(x) = \frac{15}{16h}\left(1 - \frac{x^2}{h^2}\right)^2, \quad |x| \leq h$
- Núcleo Triweight: $K_h(x) = \frac{35}{32h}\left(1 - \frac{x^2}{h^2}\right)^3, \quad |x| \leq h$
- Núcleo del coseno: $K_h(x) = \frac{\pi}{4h}\cos\left(\frac{\pi x}{2h}\right), \quad |x| \leq h$
- Núcleo de distribución lognormal: $K_h(x) = \frac{1}{hx\sqrt{2\pi}} e^{-\frac{log^2(x)}{2s^2}}$
- Núcleo de orden superior: $K_h(x) = \frac{1}{h}\phi_h(x)\left(\frac{3}{h} - \frac{x^2}{2h^2}\right)$

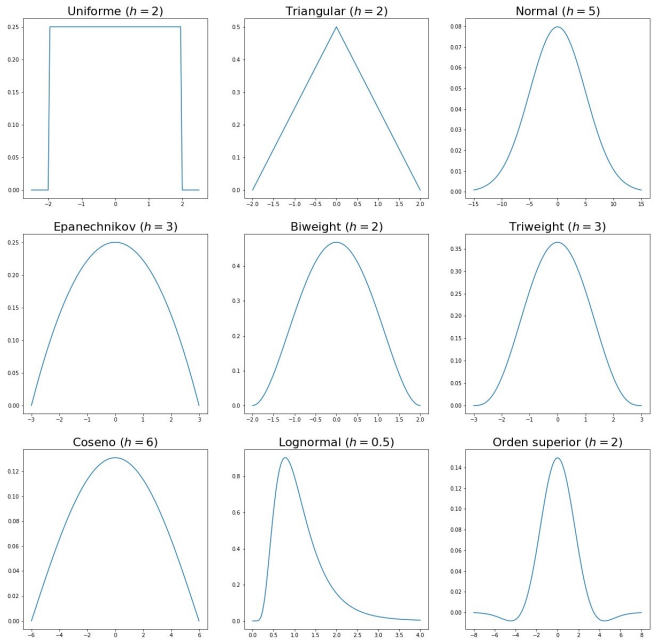


Figura 2: Representación de distintas funciones de tipo núcleo. Las siete primeras representan funciones de densidad simétricas. La octava representa una función de densidad no simétrica y la última representa una función que no es estrictamente positiva.

Algunas de estas funciones núcleo, como apunta (Duong, 2015), se pueden sintetizar en la familia de funciones núcleo basadas en la distribución beta. Así, la función núcleo beta de orden r se define como

$$K(x;r) = c_r(1-x^2)^r \mathbf{1}_{\{x\in[-1,1]\}}(x) \tag{2.3}$$

donde $c_r = \frac{(2r+1)!}{2^{2r+1}(r!)^2} = \frac{1}{\mathcal{B}(r+1,1/2)}$ siendo $\mathcal{B}(a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$ la función beta. Podemos considerar la función núcleo de la familia beta en su forma reescalada,

$$K_h(x;r) = \frac{c_r}{h}\left(1-\left(\frac{x}{h}\right)^2\right)^r \mathbf{1}_{\{x\in[-h,h]\}}(x) \tag{2.4}$$

.

Con las definiciones mostradas anteriormente, podemos comprobar como la función núcleo $K_h(x;r)$ para $r = 0, 1, 2, 3$ se corresponde a las funciones de tipo núcleo uniforme, Epanechnikov, Biweight y Triweight respectivamente.

2.2. Propiedades teóricas

Para evaluar las propiedades teóricas del estimador es preciso establecer medidas de error tanto de forma puntual como en la recta real.

Antes de comenzar con el desarrollo de las propiedades del estimador se realizan unas consideraciones previas.

En lo restante de capítulo se abusará de la notación al representar la integral sobre la recta real sin especificar el conjunto de integración. Se establecerá $\int f(x)dx$ para referirse a $\int_{\mathbb{R}} f(x)dx$.

Además, con el fin de aliviar la notación, se define la norma L^2 de una función de cuadrado integrable g como:

$$R(g) = \int g(x)^2 dx$$

Por otra parte, de aquí en adelante, se considerará que la función núcleo, $K(x)$, es una función de densidad simétrica respecto al origen.

Como es habitual al estudiar las propiedades de los estimadores paramétricos, la evaluación de la cercanía de un estimador al parámetro real se suele realizar considerando el error cuadrático medio.

En el caso del estimador de densidad de tipo núcleo, el error cuadrático medio (MSE)[a] de un estimador en un punto se define como

$$MSE\left[\hat{f}(x;h)\right] = \mathbb{E}\left[\left(\hat{f}(x;h) - f(x)\right)^2\right]$$

que puede descomponerse mediante el sesgo y la varianza del estimador de la siguiente manera

$$\begin{aligned} MSE\left[\hat{f}(x;h)\right] &= \left(\mathbb{E}\left[\hat{f}(x;h)\right] - f(x)\right)^2 + Var\left(\hat{f}(x;h)\right) \\ &= sesgo^2\left(\hat{f}(x;h)\right) + Var\left(\hat{f}(x;h)\right) \end{aligned}$$

Se realiza el cálculo del sesgo y la varianza por separado. Para el sesgo del estimador en un punto, se debe calcular la esperanza del estimador en dicho punto, es decir,

$$\begin{aligned} \mathbb{E}\left[\hat{f}(x;h)\right] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} K_h(x - X_i)\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[K_h(x - X_i)\right] = \\ = \mathbb{E}\left[K_h(x - X_1)\right] = \int K_h(x-y)f(y)dy = (K_h * f)(x) \end{aligned} \tag{2.5}$$

En el tercer paso se ha tenido en cuenta que $\{X_i\}_{i=1}^n$ son variables idénticamente distribuidas. La esperanza se reduce a la convolución entre el núcleo y la función de densidad que ha generado la muestra. Por tanto, el sesgo puede escribirse como,

$$sesgo\left(\hat{f}(x;h)\right) = (K_h * f)(x) - f(x)$$

[a] De sus siglas en inglés: Mean Squared Error

Por otra parte, la varianza del estimador puede desarrollarse como

$$\begin{aligned} Var\left(\hat{f}(x;h)\right) &= Var\left(\frac{1}{n}\sum_{i=1}^{n} K_h(x-X_i)\right) = \frac{1}{n^2} Var\left(\sum_{i=1}^{n} K_h(x-X_i)\right) = \\ &= \frac{1}{n^2}\sum_{i=1}^{n} Var\left(K_h(x-X_i)\right) = \frac{1}{n} Var\left(K_h(x-X_i)\right) = \\ &= \frac{1}{n}\left(\mathbb{E}\left[K_h(x-X_i)^2\right] - \mathbb{E}\left[K_h(x-X_i)^2\right]^2\right) = \\ &= \frac{1}{n}\left(\int K_h(x-y)^2 f(y)dy - (K_h * f)(x)^2\right) = \\ &= \frac{1}{n}\left((K_h^2 * f)(x) - (K_h * f)(x)^2\right) \end{aligned} \tag{2.6}$$

en el tercer paso se ha tenido en cuenta que las variables de la muestra son independientes y en el cuarto que provienen de la misma distribución de probabilidad.

Por tanto, el error cuadrático medio puede expresarse como

$$MSE\left(\hat{f}(x;h)\right) = \frac{1}{n}\left((K_h^2 * f)(x) - (K_h * f)(x)^2\right) + \left((K_h * f)(x) - f(x)\right)^2 \tag{2.7}$$

Puede que el interés resida no solo en el error puntual sino en toda la recta real. Para ello, se integra el error cuadrático medio en la recta real y obteniendo una medida del error del estimador global.

Así, el error cuadrático medio integrado (MISE)[b] del estimador de se define como:

$$MISE\left(\hat{f}(\cdot;h)\right) = \int MSE\left(\hat{f}(x;h)\right) dx$$

así, de (2.7) podemos integrar la expresión (2.7) del error cuadrático medio

$$\begin{aligned} MISE\left(\hat{f}(\cdot;h)\right) &= \frac{1}{n}\int (K_h^2 * f)(x)dx - \frac{1}{n}\int (K_h * f)(x)^2 dx + \int \left((K_h * f)(x) - f(x)\right)^2 dx = \\ &= \frac{1}{n}\int (K_h^2 * f)(x)dx - \frac{1}{n}\int (K_h * f)(x)^2 dx \\ &\quad + \int (K_h * f)(x)^2 dx - 2\int (K_h * f)(x) f(x) dx + \int f(x)^2 dx = \\ &= \frac{1}{n}\int (K_h^2 * f)(x)dx + \left(1 - \frac{1}{n}\right)\int (K_h * f)(x)^2 dx \\ &\quad - 2\int (K_h * f)(x) f(x) dx + R(f) = \end{aligned}$$

El primer término puede reescribirse de la siguiente manera teniendo en cuenta la conmutatividad de la convolución

$$\begin{aligned} \frac{1}{n}\int (K_h^2 * f)(x)dx &= \frac{1}{n}\int\int K_h(x-y)^2 f(y)dydx = \frac{1}{n}\int\int K_h(y)^2 f(x-y)dxdy = \\ &= \frac{1}{n}\int K_h(y)^2 \int f(x-y)dxdy = \frac{1}{n}\int K_h(y)^2 dy = \frac{1}{n}\int \frac{1}{h^2} K\left(\frac{y}{h}\right)^2 dy \underset{y=zh}{=} \\ &= \frac{1}{nh^2}\int K(z)^2 h dz = \frac{1}{nh}\int K(z)^2 dz = \frac{1}{nh} R(K) \end{aligned}$$

[b] De sus siglas en inglés: Mean Integrated Squared Error

por lo que el error cuadrático medio integrado es:

$$MISE\left(\hat{f}(\cdot;h)\right) = \frac{1}{nh}R(K) + \left(1-\frac{1}{n}\right)\int (K_h * f)(x)^2 dx - 2\int (K_h * f)(x) f(x) dx + R(f) \quad (2.8)$$

Las medidas de discrepancia mencionadas anteriormente, tanto a nivel local como global, ofrecen un valor no aleatorio sobre el comportamiento del estimador. En ciertos casos sin embargo es deseable obtener un valor estocástico que devuelva una medida no en media sino en la muestra disponible. Para ello se prescinde del operador esperanza en el error cuadrático medio integrado obteniendo el error cuadrático integrado (ISE)[c]. Esta medida resulta de calcular la distancia L^2 al cuadrado entre el estimador y la función de densidad f.

$$\begin{aligned} ISE\left(\hat{f}(\cdot;h)\right) &= \int \left(\hat{f}(x;h) - f(x)\right)^2 dx = \\ &= \int \hat{f}(x;h)^2 dx - 2\int \hat{f}(x;h) f(x) dx + \int f(x)^2 dx \end{aligned} \quad (2.9)$$

2.2.1. Propiedades asintóticas

Una vez desarrolladas las propiedades de estimador, se realiza un estudio de sus propiedades asintóticas. Así, desarrollando los primeros momentos del estimador se pueden obtener expresiones que dependen del ancho de banda de una manera más sencilla para muestras grandes.

En este sentido, se requieren ciertas propiedades de regularidad en la función de densidad f, de la función núcleo y de la secuencia de ancho de banda para poder realizar los correspondientes desarrollos.

- f es de cuadrado integrable, derivable dos veces con derivada de segundo orden acotada, continua y cuadrado integrable.
- El ancho de banda h_n es una secuencia de números positivos tal que

$$h_n \to 0, \quad nh_n \to \infty \quad \text{cuando} \quad n \to \infty$$

- K es una función de densidad simétrica respecto del origen y acotada con momento de segundo orden finito.

Teniendo en cuenta la definición del estimador de tipo núcleo (2.2), su esperanza se puede expresar de la siguiente manera partiendo de (2.5),

$$\begin{aligned} \mathbb{E}\left[\hat{f}(x;h)\right] &= \int K_h(x-y) f(y) dy \underset{y=x-hz}{=} \int K_h(hz) f(x-hz) h dz = \\ &= \int \frac{1}{h} K(z) f(x-hz) h dz = \int K(z) f(x-hz) dz \end{aligned}$$

[c] De sus siglas en inglés: Integrated Squared Error

Ahora, desarrollando $f(x-hz)$ en torno a x

$$f(x-hz)=\sum_{i=1}^{2}\frac{(-1)^i}{i!}f^{(i)}(x)(hz)^i+o(h^2)=f(x)-hzf'(x)+\frac{1}{2}h^2z^2f''(x)+o(h^2)$$

se obtiene

$$\begin{aligned}\mathbb{E}\left[\hat{f}(x;h)\right]&=\int K(z)\left(f(x)-hzf'(x)+\frac{1}{2}h^2z^2f''(x)+o(h^2)\right)dz=\\&=f(x)\int K(z)dz-hf'(x)\int zK(z)dz+\\&\quad+\frac{1}{2}h^2f''(x)\int z^2K(z)dz+o(h^2)=\\&=f(x)+\frac{1}{2}h^2\mu_2(K)f''(x)+o(h^2)\end{aligned}$$

en el último paso se ha tenido en cuenta que $\int K(z)dz=1$ por ser una función de densidad de probabilidad y $\int zK(z)dz=0$ por ser simétrica y estar centrada en el origen.

De este desarrollo es inmediato establecer una formulación simple para el sesgo del estimador ya que

$$\text{sesgo}(\hat{f}(x;h))=\mathbb{E}\left[\hat{f}(x;h)\right]-f(x)=\frac{1}{2}h^2\mu_2(K)f''(x)+o(h^2) \tag{2.10}$$

De la misma manera que en el caso del primer momento, se puede también considerar el desarrollo de la varianza del estimador. Como en el caso anterior, se puede reescribir la varianza de la siguiente manera:

$$\begin{aligned}Var\left(\hat{f}(x;h)\right)&=\frac{1}{n}\int K_h^2(x)f(x-y)dy-\frac{1}{n}\mathbb{E}^2\left[\hat{f}(x;h)\right]=\\&=\frac{1}{n}\int K_h^2(hz)f(x-hz)hdz-\frac{1}{n}\mathbb{E}^2\left[\hat{f}(x;h)\right]=\\&=\frac{1}{n}\int \frac{1}{h^2}K^2(z)f(x-hz)hdz-\frac{1}{n}\mathbb{E}^2\left[\hat{f}(x;h)\right]=\\&=\frac{1}{nh}\int K^2(z)f(x-hz)dz-\frac{1}{n}\mathbb{E}^2\left[\hat{f}(x;h)\right]=\\&=\frac{1}{nh}\int K^2(z)(f(x)+o(1))dz-\frac{1}{n}(f(x)+o(1))=\\&=\frac{1}{nh}f(x)\int K^2(z)dz+o\left(\frac{1}{nh}\right)=\\&=\frac{1}{nh}f(x)R(K)+o\left(\frac{1}{nh}\right)\end{aligned}$$

hemos tenido en cuenta que $\frac{1}{n}f(x)+o\left(\frac{1}{n}\right)$ es de orden menor a $o\left(\frac{1}{nh}\right)$

Con todo esto, el error cuadrático medio del estimador se puede formular de la siguiente manera teniendo en cuenta los desarrollos de la esperanza y la varianza del estimador.

$$\begin{aligned}MSE\left(\hat{f}(x;h)\right)&=\frac{1}{4}h^4\mu_2(K)^2f''(x)^2+o(h^4)+\frac{1}{nh}f(x)R(K)+o\left(\frac{1}{nh}\right)=\\&=\frac{1}{nh}f(x)R(K)+\frac{1}{4}h^4\mu_2(K)^2f''(x)^2+o\left(\frac{1}{nh}+h^4\right)\end{aligned} \tag{2.11}$$

De igual mantera, integrando esta expresión se obtiene una formulación distinta para el error cuadrático medio integrado

$$\begin{aligned} MISE\left(\hat{f}(\cdot;h)\right) &= \frac{1}{nh}\int f(x)\int K^2(z)dzdx + \frac{1}{4}h^4\mu_2(K)^2\int f''(x)^2dx + o\left(\frac{1}{nh}+h^4\right) = \\ &= \frac{1}{nh}\int K^2(z)dz\int f(x)dx + \frac{1}{4}h^4\mu_2(K)^2\int f''(x)^2dx + o\left(\frac{1}{nh}+h^4\right) = \\ &= \frac{1}{nh}\int K^2(z)dz + \frac{1}{4}h^4\mu_2(K)^2\int f''(x)^2dx + o\left(\frac{1}{nh}+h^4\right) = \\ &= \frac{1}{nh}R(K) + \frac{1}{4}h^4\mu_2(K)^2R(f'') + o\left(\frac{1}{nh}+h^4\right) \end{aligned} \tag{2.12}$$

Con esto, podemos definir el error de forma asintótica tanto puntualmente mediante el error cuadrático medio asintótico (AMSE)[d] como el error cuadrático medio integrado asintótico (AMISE)[e]

$$AMSE\left(\hat{f}(x;h)\right) = \frac{1}{nh}f(x)R(K) + \frac{1}{4}h^4\mu_2(K)^2f''(x)^2 \tag{2.13}$$

$$AMISE\left(\hat{f}(\cdot;h)\right) = \frac{1}{nh}R(K) + \frac{1}{4}h^4\mu_2(K)^2R(f'') \tag{2.14}$$

2.3. El problema de selección del ancho de banda y ancho de banda óptimo

La selección del parámetro del ancho de banda h depende del criterio seleccionado para minimizar el error. Así, como se ha observado en (2.8), encontrar el ancho de banda óptimo para minimizar el error cuadrático medio o el error cuadrático medio integrado es complicado ya que estos dependen de h de una forma complicada. En la expresión (**??**) para el error cuadrático medio integrado, se ve como el MISE depende de h^4 y $1/nh$.

Sin embargo, para muestras grandes, la expresión (2.14) depende de h en una manera sencilla. Así, se puede obtener el valor de h óptimo que minimiza el error cuadrático medio integrado asintótico.

$$h^*_{AMISE} = \underset{h>0}{\operatorname{argmin}} AMISE\left(\hat{f}(\cdot;h)\right)$$

que se obtiene derivando e igualando a 0 la expresión 2.14.

$$\frac{dAMISE\left(\hat{f}(\cdot;h)\right)}{dh} = -\frac{1}{nh^2}R(K) + h^3\mu_2(K)^2R(f'') = 0$$

por tanto, despejando h

$$h^*_{AMISE} = \left(\frac{R(K)}{n\mu_2(K)^2R(f'')}\right)^{\frac{1}{5}} \tag{2.15}$$

[d] De sus siglas en inglés: Asymptotic Mean Squared Error
[e] De sus siglas en inglés: Asymptotic Mean Integrated Squared Error

Con el desarrollo anterior se ha obtenido el ancho de banda óptimo respecto al error cuadrático medio integrado asintótico. Ha de notarse que este ancho de banda óptimo es aplicable a muestras grandes.

Por otra parte, hay que subrayar que h^*_{AMISE} depende de $R(f'')$ de manera inversa. $R(f'')$ nos describe la norma L_2 de la variación de la función de densidad subyacente a la muestra. Si la muestra es muy variable $R(f'')$ será grande y necesitaremos h^*_{AMISE} pequeño que capte las asperezas de de la densidad subyacente, f. Por el contrario, si la densidad f es suave, $R(f'')$ será pequeña y para describir de forma adecuada f necesitamos un ancho de banda amplio.

En cualquier caso, la selección del ancho de banda proporcionado por el criterio que acabamos de mencionar no es útil en la selección del mismo en la práctica ya que depende directamente de f que es desconocida.

No obstante, se puede calcular h^*_{AMISE} haciendo presunciones paramétricas sobre la distribución de f. Así, (Silverman, 1986) y (Wand y Jones, 1994) entre otros, estudian el caso en el que f es normal y estudian la cota superior para el ancho de banda proporcionado por (Terrell, 1990).

2.4. Métodos de selección automática del ancho de banda

Existe la posibilidad de, sin realizar asunciones sobre la distribución de f, calcular el ancho de banda óptimo sobre la estimación del error cuadrático medio integrado (asintótico).

$$\hat{h} = \underset{h>0}{\operatorname{argmin}}(\widehat{AMISE}\left(\hat{f}(\cdot;h)\right)$$

Aunque existen diversas técnicas para calcular el ancho de banda óptimo, este capitulo ce centrará en las de validación cruzada tanto ligadas al error cuadrático medio como al error cuadrático medio asintótico y *el plug-in*.

2.4.1. Validación cruzada

A la hora de realizar la selección de un parámetro mediante validación cruzada podemos tomar diversos enfoques. Uno de los mas conocidos, *leave-one-out*, se centra en el estimador de tipo núcleo empleando toda la muestra salvo uno de los elementos como se muestra a continuación. Esta técnica de validación cruzada se llama también Validación cruzada insesgada o Validación cruzada por mínimos cuadrados.

Validacion cruzada insesgada En esencia, este método trata de proporcionar un estimador que permita la minimización con respecto al parámetro h del error cuadrático integrado.

Para ello, se desarrolla el error cuadrático integrado (2.9)

$$ISE\left(\hat{f}(\cdot;h)\right) = \int \hat{f}(x;h)^2 dx - 2\int \hat{f}(x;h)f(x)dx + \int f(x)^2 dx$$

como puede observarse, el último término de la ecuación que se puede escribir como $R(f)$ no depende del ancho de banda por lo que, para obtener el ancho de banda que proporciona el menor error cuadrático integrado, podemos obviarlo. Por otra parte, el segundo término representa la esperanza del estimador condicionada a las observaciones, es decir,

$$\int \hat{f}(x;h)f(x)dx = \mathbb{E}\left[\hat{f}(X;h)|X_1,\ldots,X_n\right]$$

Se define el estimador de densidad *leave-one-out*. Se construye como el estimador de tipo núcleo pero obviando el i-ésimo elemento de la muestra,

$$\hat{f}_{-i}(x;h) = \frac{1}{n-1}\sum_{\substack{j=1\\j\neq i}}^{n} K_h(x - X_i) \tag{2.16}$$

Empleando el estimador de densidad *leave-one-out* se define un estimador insegado para $\mathbb{E}\left[\hat{f}(x;h)|X_1,\ldots,X_n\right]$

$$\frac{1}{n}\sum_{i=1}^{n}\hat{f}_{-i}(X_i;h)$$

que, efectivamente, es un estimador insesgado de la esperanza condicionada anteriormente mencionada,

$$\begin{aligned}
\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}\hat{f}_{-i}(X_i;h)\right] &= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\hat{f}_{-i}(X_i;h)\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\frac{1}{n-1}\sum_{\substack{j=1\\j\neq i}}^{n}K_h(X_i - X_j)\right] = \\
&= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n-1}\sum_{\substack{j=1\\j\neq i}}^{n}\mathbb{E}\left[K_h(X_i - X_j)\right] = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n-1}(n-1)(K_h * f)(X_i) = \\
&= \frac{1}{n}\sum_{i=1}^{n}(K_h * f)(X_i) = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\hat{f}(X,h)|X_i] = \\
&= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\hat{f}(X,h)|X_1] = \int \hat{f}(x,h)f(x)dx
\end{aligned}$$

Con esto, se obtiene un estimador insesgado para $ISE\left(\hat{f}(x;h)\right) - f(x)$. Así, se define el criterio de validación cruzada insesgada (UCV)[f] como

$$UCV(h) = \int \hat{f}(x;h)^2dx - 2\frac{1}{n}\sum_{i=1}^{n}\hat{f}_{-i}(X_i;h) \tag{2.17}$$

que efectivamente es un estimador insesgado de $ISE\left(\hat{f}(x;h)\right) - f(x)$, de se puede obtener un ancho de banda tal que minimice el error cuadrático integrado,

$$h^*_{UCV} = \underset{h>0}{\operatorname{argmin}}\ UCV(h)$$

[f]De sus siglas en inglés: Unbiased Cross Validation

Es necesario definir el criterio de validación cruzada insesgada de una manera más adecuada para su facilitar su cálculo en la práctica. Para ello, se desarrollan los dos términos de (2.17) por separado,

$$\begin{aligned}\int \hat{f}(x;h)^2 dx &= \int \left(\frac{1}{n}\sum_{i=1}^{n} K_h(x-X_i)\right)\left(\frac{1}{n}\sum_{j=1}^{n} K_h(x-X_j)\right) dx = \\ &= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\int K_h(x-X_i)K_h(x-X_j)dx = \\ &= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}\int K_h(x-X_i)K_h(x-X_j)dx + \frac{1}{n^2}\sum_{i=1}^{n}\int K_h(x-X_i)^2 dx\end{aligned}$$

el segundo sumatorio se puede reescribir de la siguiente manera

$$\frac{1}{n^2}\sum_{i=1}^{n}\int K_h(x-X_i)^2 dx \underset{x-X_i=hz}{=} \frac{1}{n^2}\sum_{i=1}^{n}\int K_h(hz)^2 h dz = \frac{1}{n^2} hn \int \frac{1}{h^2}K(z)^2 dz = \frac{1}{nh}R(K)$$

y el primero,

$$\begin{aligned}&\frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}\int K_h(x-X_i)K_h(x-X_j)dx \underset{x-X_j=z}{=} \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}\int K_h(z-X_i+X_j)K_h(z)dz = \\ &= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}\int K_h(X_i-X_j-z)K_h(z)dz = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}(K_h * K_h)(X_i-X_j)\end{aligned}$$

en el segundo paso se ha empleado la hipótesis sobre la simetría de núcleo, $K_h(x) = K_h(-x)$. Así,

$$\int \hat{f}(x;h)^2 dx = \frac{1}{nh}R(K) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}(K_h * K_h)(X_i-X_j)$$

Por otra parte, se puede desarrollar el segundo término de (2.17),

$$2\frac{1}{n}\sum_{i=1}^{n}\hat{f}_{-i}(X_i;h) = 2\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n-1}\sum_{\substack{j=1\\ j\neq i}}^{n}K_h(X_i-X_j) = 2\frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}K_h(X_i-X_j)$$

Por tanto, el estimador de validación cruzada insesgado puede reescribirse de la siguiente manera,

$$\begin{aligned}UCV(h) &= \frac{1}{nh}R(K) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}(K_h * K_h)(X_i-X_j) - 2\frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}K_h(X_i-X_j) = \\ &= \frac{1}{nh}R(K) + \frac{(n-1)}{n^2(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}(K_h * K_h)(X_i-X_j) - \frac{2}{n(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}K_h(X_i-X_j) = \\ &= \frac{1}{nh}R(K) + \frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}\left\{\left(\frac{n-1}{n}\right)(K_h * K_h) - 2K_h\right\}(X_i-X_j)\end{aligned} \tag{2.18}$$

Es habitual considerar $1 - \frac{1}{n} \sim 1$. Aunque se pierda la insesgadez del estimador, no se pierde la insesgadez asintótica y el estimador tiene una forma más sencilla

$$UCV(h) = \frac{1}{nh}R(K) + \frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\left\{(K_h * K_h) - 2K_h\right\}(X_i - X_j) \tag{2.19}$$

En este sentido, no habría diferencias significativas entre la expresión (2.18) y (2.19) al tratar con muestras grandes.

Por otra parte, como muestran (Hall y Marron, 1991), el ancho de banda óptimo obtenido mediante validación cruzada insesgada, h^*_{UCV} presenta la existencia de varios mínimos locales y una tendencia al infrasuavizamiento de la curva de densidad. Asimismo, mencionan que el ancho de banda óptimo es altamente variable. Se realizara un estudio empírico de estas cuestiones en el capitulo 7.1.

Validación cruzada sesgada Anteriormente se ha considerado el ancho de banda que minimiza el error cuadrático (medio) integrado. En este caso, se empleara la expresión (2.14) del error cuadrático medio integrado asintótico.

$$AMISE\left(\hat{f}(\cdot;h)\right) = \frac{1}{nh}R(K) + \frac{1}{4}h^4\mu_2(K)^2R(f'')$$

donde hay que estimar la expresión $R(f'')$ que es desconocida. Para ello, debemos tener en cuenta que

$$\hat{f}(x;h) = \frac{1}{n}\sum_{i=1}^{n}K_h(x - X_i) \Rightarrow \hat{f}^{r)}(x;h) = \frac{1}{n}\sum_{i=1}^{n}K_h^{r)}(x - X_i) \tag{2.20}$$

$$K_h(x) = \frac{1}{h}K\left(\frac{x}{h}\right) \Rightarrow K_h'(x) = \frac{1}{h^2}K'\left(\frac{x}{h}\right) \Rightarrow \cdots \Rightarrow K_h^{r)}(x) = \frac{1}{h^{r+1}}K^{r)}\left(\frac{x}{h}\right) \tag{2.21}$$

La aproximación natural al problema, consistiría en estimar $R(f'')$ mediante $R(\hat{f}'')$ como proponen (Scott y col., 1977).

$$\begin{aligned} R(\hat{f}'') &= \int \hat{f}''(x;h)^2dx = \frac{1}{n^2}\int\left(\sum_{i=1}^{n}K_h''(x - X_i)\right)^2 dx = \\ &= \frac{1}{n^2}\int\sum_{i=1}^{n}K_h''(x - X_i)^2dx + \frac{1}{n^2}\int\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}K_h''(x - X_i)K_h''(x - X_j)dx \end{aligned}$$

donde el primer sumatorio puede reescribirse como,

$$\frac{1}{n^2}\int\sum_{i=1}^{n}K_h''(x - X_i)^2dx \underset{x-X_i=hz}{=} \frac{1}{n^2}\int\sum_{i=1}^{n}K_h''(hz)^2hdz = \frac{1}{n^2h^6}nh\int K''(z)^2dz = \frac{1}{nh^5}R(K'')$$

y el segundo,

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\int K_h''(x-X_i)K_h''(x-X_j)dx \underset{x-X_j=z}{=} \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\int K_h''(z-X_i+X_j)K_h''(z)dz =$$
$$= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\int K_h''(X_i-X_j-z)K_h''(z)dz = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}(K_h''*K_h'')(X_i-X_j)$$

notamos que, bajo ciertas condiciones de regularidad de K,

$$\int K''(x)K''(x)dx \underset{\substack{K''(x)=u\\K''(x)dx=dv}}{=} K''(x)K'(x)\big|_{-\infty}^{\infty} - \int K'(x)K^{(3)}(x)dx = -\int K'(x)K^{(3)}(x)dx =$$
$$\underset{\substack{K^{(3)}(x)=u\\K'(x)dx=dv}}{=} -K^{(3)}(x)K(x)\Big|_{-\infty}^{\infty} + \int K^{(4)}(x)K(x)dx = \int K(x)K^{(4)}(x)dx$$

por lo tanto, el segundo sumando lo podemos expresar también como,

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\int K_h^{(4)}(x-X_i)K_h(x-X_j)dx \underset{x-X_j=z}{=} \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\int K_h^{(4)}(z-X_i+X_j)K_h(z)dz =$$
$$= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\int K_h^{(4)}(X_i-X_j-z)K_h(z)dz = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}(K_h^{(4)}*K_h)(X_i-X_j)$$

por lo que,

$$\begin{aligned} R(\hat{f}'') &= \frac{1}{nh^5}R(K'') + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}(K_h''*K_h'')(X_i-X_j) = \\ &= \frac{1}{nh^5}R(K'') + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}(K_h^{(4)}*K_h)(X_i-X_j) \end{aligned} \tag{2.22}$$

Sin embargo, como se ve en (2.22) y como muestran (Scott y Terrell, 1987) en la demostración del Lema 3.2, que generalizamos en (), este no es un buen estimador ya que, bajo ciertas condiciones de regularidad de f y de K,

$$\mathbb{E}\left[R(\hat{f}'')\right] = \frac{(n-1)}{n}R(f'') + \frac{1}{nh^5}R(K'') + o(h^2) \tag{2.23}$$

como podía era esperable de (2.22).

Por tanto, proponen como estimador de $R(f'')$

$$\widetilde{R(f'')} = R(\hat{f}'') - \frac{1}{nh^5}R(K'') = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}(K_h''*K_h'')(X_i-X_j)$$

que si bien sesgado es asintóticamente insesgado.

Por tanto,

$$BCV_0(h)^{g} = \frac{1}{nh}R(K) + \frac{1}{4}h^4\mu_2^2(K)\widetilde{R(f'')} \tag{2.24}$$

[g]Le ponemos el subíndice 0 a esta función BCV porque preferiremos el estimador que resulta en la función $BCV_1(h)$.

Siguiendo esta idea, (Hall y Marron, 1987) proponen dos estimadores para $R(\hat{f}'')$, que también analizan (Jones y Kappenman, 1992). El primero basado en 2.22 pero teniendo en cuenta el término $\frac{n-1}{n}$ del que Scott y Terrell se deshacen,

$$\widehat{R(f'')} = \frac{n-1}{n} R(\hat{f}'') - \frac{1}{nh^5} R(K'') = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} (K_h'' * K_h'')(X_i - X_j) \tag{2.25}$$

notamos que podemos escribir el estimador como,

$$\widehat{R(f'')} = \frac{n-1}{n} R(\hat{f}'') - \frac{1}{nh^5} R(K'') = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} (K_h^{(4)} * K_h)(X_i - X_j) \tag{2.26}$$

$$BCV_1(h) = \frac{1}{nh} R(K) + \frac{1}{4} h^4 \mu_2^2(K) \widehat{R(f'')} \tag{2.27}$$

El segundo estimador que proponen (Hall y Marron, 1987), se basa en la idea de que

$$R(f'') = \int f''(x)^2 dx \underset{\substack{f''(x)=u \\ f''(x)dx=dv}}{=} f''(x)f'(x)|_{-\infty}^{\infty} - \int f'(x)f^{(3)}(x)dx = -\int f'(x)f^{(3)}(x)dx =$$

$$\underset{\substack{f^{(3)}(x)=u \\ f'(x)dx=dv}}{=} - f^{(3)}(x)f(x)\Big|_{-\infty}^{\infty} + \int f^{(4)}(x)f(x)dx = \int f(x)f^{(4)}(x)dx = \mathbb{E}\left[f^{(4)}(X)\right]$$

así, proponen el empleo de la cuarta derivada del estimador (2.16),

$$\widehat{\widehat{R(f'')}} = \frac{1}{n} \sum_{i=1}^{n} \hat{f}_{-i}^{(4)}(X_i) = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} K_h^{(4)}(X_i - X_j) \tag{2.28}$$

$$BCV_2(h) = \frac{1}{nh} R(K) + \frac{1}{4} h^4 \mu_2^2(K) \widehat{\widehat{R(f'')}} \tag{2.29}$$

Y minimizando la función adecuada para cada estimador, se obtiene el ancho de banda óptimo

$$h_{BCV}^* = \underset{h>0}{\operatorname{argmin}}\ BCV_i(h), \quad i = 1, 2$$

2.4.2. Plug-in

El método de selección del ancho de banda *plug-in*, se desarrolla del ancho de banda óptimo del error cuadrático medio integrado asintótico (2.15). En este caso, como en el anterior, se desarrolla una estimación para $R(f'')$ que es desconocido.

Para ello, se realiza el desarrollo de la norma L_2 de las derivadas de una función de densidad f a la que le suponemos la regularidad suficiente. Esto no es estrictamente necesario para el caso de estudio aquí presentado pero resulta de interés a la hora de obtener una expresión general.

Integrando por partes r veces se obtiene,

$$R(f^{(r)}) = \int f^{(r)}(x)^2 dx \underset{\substack{f^{(r)}(x)=u \\ f^{(r)}(x)dx=dv}}{=} f^{(r)}(x)f^{(r-1)}(x)\Big|_{-\infty}^{\infty} - \int f^{(r-1)}(x)f^{(r+1)}(x)dx =$$

$$= -\int f^{(r-1)}(x)f^{(r+1)}(x)dx \underset{\substack{f^{(r+1)}(x)=u \\ f^{(r-1)}(x)dx=dv}}{=}$$

$$= -f^{(r+1)}(x)f^{(r-2)}(x)\Big|_{-\infty}^{\infty} + \int f^{(r-2)}(x)f^{(r+2)}(x)dx = \cdots = (-1)^r \int f(x)f^{(2r)}(x)dx$$

que se corresponde a la esperanza del la $2r-$ésima derivada de f. Por lo tanto, cuando r es par,

$$\int f(x)f^{(r)}(x)dx = \mathbb{E}[f^{(r)}(X)] = \psi_r$$

Teniendo en cuenta el caso particular $r = 2$ en el que $R(f'') = \psi_4$, (2.15) se puede reescribir como,

$$h_{AMISE} = \left(\frac{R(K)}{\mu_2^2(K)\psi_4 n}\right)^{\frac{1}{5}} \tag{2.30}$$

Como se ha comentado $\psi_4 = \mathbb{E}[f^{(4)}(X)]$ por lo que el estimador natural de ψ_4 sería

$$\hat{\psi}_4 = \frac{1}{n}\sum_{i=1}^{n} \hat{f}^{4)}(X_i; g) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} L_g^{4)}(X_i - X_j) \tag{2.31}$$

donde L es una función de tipo núcleo posiblemente distinta de K y g es un ancho de banda piloto. Nosotros consideraremos que L es una función de densidad simétrica. El caso más general en el que L es un núcleo distinto de K simétrico respecto del origen de orden superior $r = 2n$ para $n > 1$ puede consultarse en (Wand y Jones, 1994) en la sección 3.5.

Impondremos las siguientes condiciones sobre el kernel y el ancho de banda pilotos.

- L es un kernel simétrico de orden k, $k = 2, 4, \ldots$, derivable r veces tal que

$$(-1)^{\frac{r+k}{2}+1} L^{(r)}(0)\mu_k(L) > 0$$

- Para $p > k$, f es una función de clase $\mathcal{C}^p$ con $\lim_{x\to\pm\infty} f^{(p)}(x) = 0$

- El ancho de banda piloto g_n es una secuencia de números positivos tal que

$$g_n \to 0, \quad n g_n^{2r+1} \to \infty \quad \text{cuando} \quad n \to \infty$$

En este punto, problema radica en la elección del ancho de banda óptimo. Una opción para ello resulta de la minimización del error cuadrático medio asintótico de la estimación $\hat{\psi}_4$ ya que estamos estimando un escalar y no una función.

En este punto como en la sección (2.2.1) se requieren ciertas propiedades para poder calcular el error cuadrático medio asintótico de $\hat{\psi}_4$.

Teniendo en cuenta que

$$\hat{\psi}_r = \frac{1}{n^2}\sum_{i=1}^{n} L_g^{(r)}(X_i - X_i) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}} L_g^{(r)}(X_i - X_j) = \frac{1}{n}L_g^{(r)}(0) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}} L_g^{(r)}(X_i - X_j)$$

por lo que,

$$\mathbb{E}[\hat{\psi}_r] = \frac{1}{n}L_g^{(r)}(0) + \frac{1}{n^2}\mathbb{E}\Big[\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}} L_g^{(r)}(X_i - X_j)\Big] = \frac{1}{n}L_g^{(r)}(0) + \frac{1}{n^2}n(n-1)\mathbb{E}\Big[L_g^{(r)}(X_1 - X_2)\Big] =$$
$$= \frac{1}{ng^{r+1}}L^{(r)}(0) + \frac{1}{n^2}n(n-1)\mathbb{E}\Big[L_g^{(r)}(X_1 - X_2)\Big]$$

Así calculando la esperanza del término estocástico en la ecuación anterior,

$$\mathbb{E}\Big[L_g^{(r)}(X_1 - X_2)\Big] = \int\int L_g^{(r)}(x-y)f(x)f(y)dxdy = \int f(x)\int L_g^{(r)}(x-y)f(y)dydx =$$
$$= \int f(x)\int L_g(x-y)f^{(r)}(y)dydx \underset{\substack{x=v+gu\\ y=v}}{=} \int\int L_g(gu)f(v+gu)f^{(r)}(v)gdudv =$$
$$= \int\int L(u)f(v+gu)f^{(r)}(v)dudv = \int\int L(u)f^{(r)}(v)\left(f(v) + \frac{1}{2}g^2u^2f''(v) + o(g^4)\right)dudv =$$
$$= \int f^{(r)}(v)f(v)\int L(u)dudv + \frac{1}{2}g^2\int f^{(r)}(v)f''(v)\int u^2L(u)dudv + O(g^4) =$$
$$= \int f^{(r)}(v)f(v)dv + \frac{1}{2}g^2\mu_2(L)\int f^{(r)}(v)f''(v)dv + O(g^4) = \psi_r + \frac{1}{2}g^2\mu_2(L)\psi_{r+2} + O(g^4)$$

Por tanto, el sesgo de $\hat{\psi}_r$ es

$$\begin{aligned}\mathbb{E}[\hat{\psi}_r] - \psi_r &= \frac{1}{n}L_g^{(r)}(0) + \frac{1}{n}\psi_r + \left(1 - \frac{1}{n}\right)\frac{1}{2}g^2\mu_2(L)\psi_{r+2} + O(g^4) = \\ &= \frac{1}{n}L_g^{(r)}(0) + \frac{1}{2}g^2\mu_2(L)\psi_{r+2} + O(g^4)\end{aligned} \tag{2.32}$$

Ahora es necesario el cálculo de la varianza de $\hat{\psi}_r$. Para ello, se desarrolla la varianza de la siguiente manera,

$$Var(\hat{\psi}_r) = \frac{1}{n^2}Var\Big(\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n} L_g^{(r)}(X_i - X_j)\Big) = \frac{2}{n^4}Var\Big(\sum_{i=1}^{n}\sum_{j>i}^{n} L_g^{(r)}(X_i - X_j)\Big) =$$
$$= \frac{2}{n^4}\sum_{i=1}^{n}\sum_{j>i}^{n} Var\big(L_g^{(r)}(X_i - X_j)\big) + 2\frac{2}{n^2}\sum_{i=1}^{n}\sum_{j>i}\sum_{k>j} Cov\big(L_g^{(r)}(X_i - X_j), L_g^{(r)}(X_j - X_k)\big) =$$
$$= \frac{2}{n^4}n(n-1)Var\big(L_g^{(r)}(X_1 - X_2)\big) + \frac{4}{n^4}n(n-1)(n-2)Cov\big(L_g^{(r)}(X_1 - X_2), L_g^{(r)}(X_2 - X_3)\big) =$$
$$= \frac{2}{n^3}(n-1)Var\big(L_g^{(r)}(X_1 - X_2)\big) + \frac{4}{n^3}(n-1)(n-2)Cov\big(L_g^{(r)}(X_1 - X_2), L_g^{(r)}(X_2 - X_3)\big)$$

Entonces,

$$\begin{aligned}
\mathbb{E}\big[L_g^{(r)}(X_1-X_2)^2\big] &= \int\int L_g^{(r)}(X_1-X_2)^2 f(x)f(y)dxdy = \int\int L_g^{(r)}(x-y)^2 f(x)f(y)dxdy \underset{\substack{x=v+gu\\ y=v}}{=} \\
&= \int\int L_g^{(r)}(gu)^2 f(v)f(v+gu)gdudv = \frac{1}{g^{2r+1}}\int\int L^{(r)}(u)^2 f(v+gu)f(v)dudv = \\
&= \frac{1}{g^{2r+1}}\int\int L^{(r)}(u)^2\big(f(v)+o(1)\big)f(v)dudv = \\
&= \frac{1}{g^{2r+1}}\int\int L^{(r)}(u)^2 f(v)f(v)dudv + o(\frac{1}{g^{2r+1}}) = \\
&= \frac{1}{g^{2r+1}}\int L^{(r)}(u)^2 du\int f(v)f(v)dv + o(\frac{1}{g^{2r+1}}) = \frac{1}{g^{2r+1}}R(L^{(r)})R(f) + o(\frac{1}{g^{2r+1}}) = \\
&= \frac{1}{g^{2r+1}}R(L^{(r)})\psi_0 + o(\frac{1}{g^{2r+1}})
\end{aligned}$$

Por otra parte,

$$\begin{aligned}
\mathbb{E}\big[L_g^{(r)}(X_1-X_2)L_g^{(r)}(X_2-X_2)\big] &= \int\int L_g^{(r)}(x-y)L_g^{(r)}(y-z)f(x)f(y)f(z)dxdydz = \\
&= \int f(y)\int L_g^{(r)}(x-y)f(x)dx\int L_g^{(r)}(y-z)f(z)dzdy = \\
&= \int f(y)\int L_g(x-y)f^{(r)}(x)dx\int L_g(y-z)f^{(r)}(z)dzdy = \\
&= \int f(y)\int L_g(x-y)f^{(r)}(x)dx\int L_g(y-z)f^{(r)}(z)dzdy = \\
&= \int\int\int L_g(x-y)L_g(y-z)f^{(r)}(x)f^{(r)}(z)f(y)dxdydz \underset{\substack{x=w+u\\ y=w\\ z=w-v}}{=} \\
&= \int\int\int L_g(u)L_g(v)f^{(r)}(w+u)f^{(r)}(w-v)f(w)dudvdw = \\
&= \int\int\int L_g(u)L_g(v)(f^{(r)}(w)+o(1))(f^{(r)}(w)+o(1))f(w)dudvdw = \\
&= \int L_g(v)dv\int L_g(u)du\int f^{(r)}(w)^2 f(w)dw + o(1) = \\
&= \int f^{(r)}(w)^2 f(w)dw + o(1)
\end{aligned}$$

$$\mathbb{E}\big[L_g^{(r)}(X_1-X_2)L_g^{(r)}(X_2-X_2)\big] = \int\int L_g^{(r)}(x-y)L_g^{(r)}(y-z)f(x)f(y)f(z)dxdydz =$$
$$= \int f(y)\int L_g^{(r)}(x-y)f(x)dx \int L_g^{(r)}(y-z)f(z)dzdy =$$
$$= \int f(y)\int L_g(x-y)f^{(r)}(x)dx \int L_g(y-z)f^{(r)}(z)dzdy =$$
$$= \int f(y)\int L_g(x-y)f^{(r)}(x)dx \int L_g(y-z)f^{(r)}(z)dzdy =$$
$$= \int\int\int L_g(x-y)L_g(y-z)f^{(r)}(x)f^{(r)}(z)f(y)dxdydz \underset{\substack{x=w+u\\ y=w\\ z=w-v}}{=}$$
$$= \int\int\int L_g(u)L_g(v)f^{(r)}(w+u)f^{(r)}(w-v)f(w)dudvdw =$$
$$= \int\int\int L_g(u)L_g(v)(f^{(r)}(w)+o(1))(f^{(r)}(w)+o(1))f(w)dudvdw =$$
$$= \int L_g(v)dv \int L_g(u)du \int f^{(r)}(w)^2 f(w)dw + o(1) =$$
$$= \int f^{(r)}(w)^2 f(w)dw + o(1)$$

Resumiendo los cálculos anteriores se obtiene,

$$Var(\hat{\psi}_r) = \frac{2}{n^3}(n-1)Var\big(L_g^{(r)}(X_1-X_2)\big) + \frac{4}{n^3}(n-1)(n-2)Cov\big(L_g^{(r)}(X_1-X_2), L_g^{(r)}(X_2-X_3)\big) =$$
$$= \frac{2}{n^3}(n-1)\left(\mathbb{E}\big[L_g^{(r)}(X_1-X_2)^2\big] - \mathbb{E}\big[L_g^{(r)}(X_1-X_2)^2\big]^2\right) +$$
$$+ \frac{4}{n^3}(n-1)(n-2)\left(\mathbb{E}\big[L_g^{(r)}(X_1-X_2)L_g^{(r)}(X_2-X_3)\big] - \mathbb{E}\big[L_g^{(r)}(X_1-X_2)^2\big]^2\right) =$$
$$= \frac{2}{n^3}(n-1)\left(\frac{R(L^{(r)})\psi_0}{g^{2r+1}} - \psi_r^2\right) + \frac{4}{n^3}(n-1)(n-2)\left(\int f^{(r)}(x)f(x)dx - \psi_r^2\right) =$$
$$= 2\left(\frac{1}{n^2}-\frac{1}{n^3}\right)\frac{R(L^{(r)})\psi_0}{g^{2r+1}} + 4\left(\frac{1}{n}-\frac{3}{n^2}+\frac{2}{n^3}\right)\int f^{(r)}(x)f(x)dx -$$
$$- 4\left(\frac{1}{n}-\frac{3}{n^2}+\frac{2}{n^3}+\frac{1}{2n^2}-\frac{1}{2n^3}\right)\psi_r^2 =$$
$$= \frac{2}{g^{2r+1}n^2}R(L^{(r)})\psi_0 + \frac{4}{n}\int f^{(r)}(x)f(x)dx - \frac{4}{n}\psi_r^2 + o(\frac{2}{g^{2r+1}n^2}+\frac{1}{n})$$

Con todo esto, podemos escribir el error cuadrático medio de $\hat{\psi}_r$

$$MSE(\hat{\psi}_r) = \left(\frac{1}{n}L_g^{(r)}(0) + \frac{1}{2}g^2\mu_2(L)\psi_{r+2}\right)^2 + \frac{2}{g^{2r+1}n^2}R(L^{(r)})\psi_0 + \\ + \frac{4}{n}\left(\int f^{(r)}(x)f(x)dx - \psi_r^2\right) + o(\frac{2}{g^{2r+1}n^2}+\frac{1}{n}) + O(g^8) \tag{2.33}$$

Por tanto, el error cuadrático medio asintótico, (AMSE)[h]

$$AMSE(\hat{\psi}_r) = \left(\frac{1}{n}L_g^{(r)}(0) + \frac{1}{2}g^2\mu_2(L)\psi_{r+2}\right)^2 + \frac{2}{g^{2r+1}n^2}R(L^{(r)})\psi_0 + \frac{4}{n}\left(\int f^{(r)}(x)f(x)dx - \psi_r^2\right) \tag{2.34}$$

[h] De sus siglas en inglés: Asymptotic Mean Squared Error

El ancho de banda óptimo resulta de minimizar el término del sesgo en (2.34). Así,

$$\frac{1}{ng^{r+1}}L^{(r)}(0)+\frac{1}{2}g^2\mu_2(L)\psi_{r+2}=0\Rightarrow-\frac{1}{n}L^{(r)}(0)=\frac{1}{2}g^{r+1+2}\mu_2(L)\psi_{r+2}$$
$$\Rightarrow-\frac{2}{n\mu_2(L)\psi_{r+2}}L^{(r)}(0)=g^{r+1+2}$$

Por tanto,

$$g^*_{AMSE}=\left(-\frac{2L^{(r)}(0)}{n\mu_2(L)\psi_{r+2}}\right)^{\frac{1}{r+1+2}} \tag{2.35}$$

Por lo tanto, el problema que resulta de este cálculo, es el siguiente: Si se quiere estimar ψ_r mediante $\hat{\psi}_r(g^*_{AMSE})$ resulta necesario estimar ψ_{r+2}. La solución a este problema se basa en insertar *to plug-in* una estimación simple de ψ_r en un punto.

Así se genera el estimador plug in en l estapas, dependiendo en qué l estimemos suponiendo que f sigue una distribución normal para estimar ψ_{4+2l}. Así, siendo $f(x)=\phi_\sigma(x-\mu)$ la función de densidad de una variable aleatoria $N(\mu,\sigma)$, para r par,

$$\begin{aligned}\psi_{2r}=R(f^{(r)})&=\int\phi^{(r)}_\sigma(x-\mu)\phi^{(r)}_\sigma(x-\mu)dx\underset{x=y+\mu}{=}\\&=\int\phi^{(r)}_\sigma(y)\phi^{(r)}_\sigma(y-\mu+\mu)dy=\int\phi^{(r)}_\sigma(y)\phi^{(r)}_\sigma(\mu-\mu-y)dy=\\&=(\phi^{(r)}_\sigma*\phi^{(r)}_\sigma)(0)\end{aligned}$$

Teniendo en cuenta que $(\phi^{(r)}_\sigma*\phi^{(r)}_\sigma)(x)=\phi^{(2r)}_{\sqrt{2\sigma^2}}(x)$ y que $\phi^{(2r)}_\sigma=(-1)^r\frac{1}{\sqrt{2\pi}\sigma^{2r-1}}\frac{(2r)!}{2^r r!}$ cuyas demostraciones se pueden encontrar en (Aldershof y col., 1995), se obtiene

$$\psi_r=\frac{(-1)^{\frac{r}{2}}r!}{\pi^{\frac{1}{2}}(2\sigma)^{r+1}(\frac{r}{2})!}$$

El problema radica en la estimación de σ. Como apuntan (Silverman, 1986; Wand y Jones, 1994) o más recientemente (García-Portugués, 2021) empleamos como estimador el mínimo entre la desviación estándar de la muestra, s, y su rango intercuartílico estandarizado para evitar la posible influencia de valores atípicos.

$$\hat{\sigma}=\text{mín}\left\{s,\frac{X_{[0{,}75n]}-X_{[0{,}5n]}}{\Phi^{-1}(0{,}75)-\Phi^{-1}(0{,}25)}\right\}$$

Así, obtenemos distintos estimadores de tipo *plug-in*.

***Plug in* de 0 etapas: *'Rule of thumb'* o regla general.** Este caso resulta de estimar $R(f'')=\psi_4$ asumiendo que f es la densidad de una distribución $N(\mu,\sigma)$. Es decir, inertamos

$$\hat{\psi_4}=\frac{4!}{\pi^{\frac{1}{2}}(2\hat{\sigma})^5 2!}=\frac{3}{8\pi^{\frac{1}{2}}\hat{\sigma}^5}$$

que incluido en (2.30), se obtiene el ancho de banda óptimo *plug-in* de 0 etapas o el *rule of thumb*

$$h^*_{PI_0} = \left(\frac{8\pi^{\frac{1}{2}}R(K)}{3\mu_2(K)^2 n}\right)^{\frac{1}{5}} \hat{\sigma} \tag{2.36}$$

***Plug in* de 2 etapas: *'Direct plug-in'*.** Es el caso presentado por (Sheather y Jones, 1991). En este caso se estima ψ_8 suponiendo normalidad en f de forma que,

$$\hat{\psi}_8 = \frac{8!}{\pi^{\frac{1}{2}}(2\hat{\sigma})^9 4!} = \frac{105}{32\pi^{\frac{1}{2}}\hat{\sigma}^9}$$

por lo que podemos obtener g_1 para estimar ψ_6 mediante

$$\hat{\psi}_6(g_1) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} K^{(6)}_{g_1}(X_i - X_j) \qquad \text{donde} \qquad g_1 = \left(-\frac{2K^{(6)}(0)}{n\mu_2(K)\hat{\psi}_8}\right)^{\frac{1}{9}}$$

así, obtenemos g_2 para estimar ψ_4 mediante

$$\hat{\psi}_4(g_2) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} K^{(4)}_{g_1}(X_i - X_j) \qquad \text{donde} \qquad g_2 = \left(-\frac{2K^{(4)}(0)}{n\mu_2(K)\hat{\psi}_6(g_1)}\right)^{\frac{1}{7}}$$

y por tanto el ancho de banda óptimo

$$h^*_{PI_2} = \left(\frac{R(K)}{\mu_2(K)^2\hat{\psi}_4(g_2)n}\right)^{\frac{1}{5}} \tag{2.37}$$

2.4.3. Otros selectores del ancho de banda

Hemos destacado dos métodos basados en datos como son la validación cruzada insesgada y sesgada. Aun así, hay otros métodos basados en los datos muestrales. (Silverman, 1986) propone un método de validación cruzada basada en la máxima verosimilitud que, salvo por una constante, es un estimador insesgado de la divergencia de Kullback-Leiber entre la densidad real y su estimación.

(Hall y col., 1992) el método de validación cruzada suavizada (*smoothed cross-validation*) que puede entenderse como un híbrido entre los métodos de validación cruzada y los métodos *plug-in*. Este método es equivalente al suavizado por *bootstrap* de (Faraway y Jhun, 1990; Taylor, 1989) como menciona (Duong, 2004).

Más recientemente (Mammen y col., 2011) desarrollan el método de validación cruzada *double-sided* que desarrolla una teoría asintótica para ancho de banda creados como combinaciones lineales de diferentes selectores de anchos de banda.

2.5. Comparación empírica de los estimadores del ancho de banda

Como hemos comentado anteriormente, en general la selección de h mediante validación cruzada insesgada resulta en densidades excesivamente abruptas. Mientras que otros métodos tienden a suavizar en exceso la función de densidad. En este apartado queremos comprobarlo de forma

empírica. Para comprobarlo empíricamente, se genera una muestra de $n = 1000$ observaciones que provienen de una mixtura de normales, $0{,}3N(4,1) + 0{,}7N(7,1)$ cuya función de densidad viene definida como

$$f(x) = 0{,}3\ \phi(x-4) + 0{,}7\ \phi(x-7) \tag{2.38}$$

Mediante la muestra generada se realiza la estimación de la densidad de tipo núcleo empleando distintos núcleos y para los estimadores del ancho de banda presentados anteriormente.

Como se indica en el capítulo 7, el código se detalla en los anexos II y III y es accesible y libre para descargar en la página de GitHub que se detalla el capítulo 7.

Como ya hemos mencionado anteriormente y como resaltan muchos autores, la elección del Kernel no es determinante a la hora de realizar la estimación. Sin embargo, mostraremos aquí, como realizar la estimación empleando un Kernel Gaussiano y un kernel de tipo beta de orden 3 que se corresponde con el núcleo *Triweight*. Cuando consideramos este segundo núcleo, en la estimación *plug-in* del ancho de banda, consideraremos un kernel piloto normal.

Consideraremos la densidad que hemos mencionado anteriormente y mostramos la muestra aleatoria que ha sido generada.

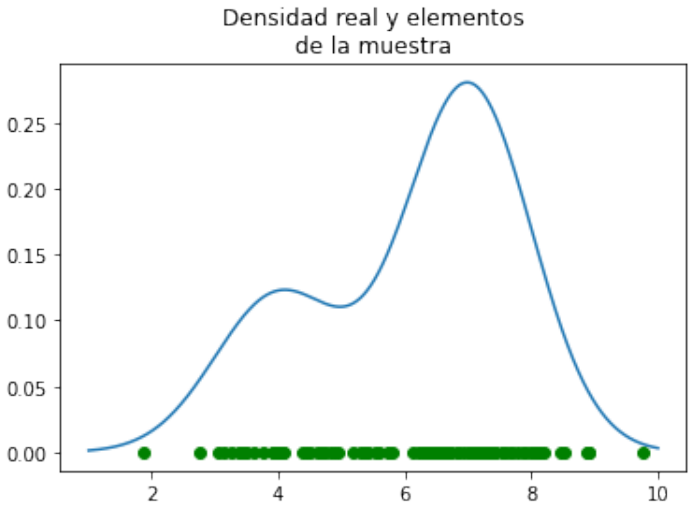


Figura 3: Representación de la función de densidad de (2.38) y la muestra aleatoria generada en forma de puntos de color verde

Ejemplo 1. Empleando un núcleo normal

Empleando un núcleo normal obtenemos las siguientes estimaciones de f,

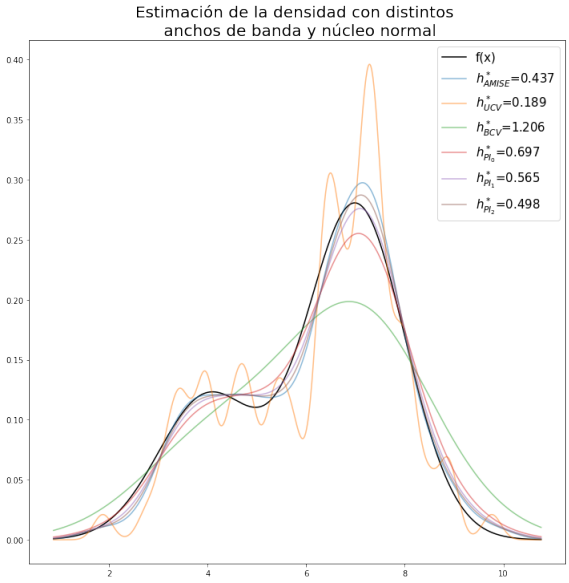


Figura 4: Representación de la función de densidad de (2.38) y la estimación de tipo núcleo mediante un núcleo normal para una muestra aleatoria simple de tamaño 100. Se ha realizado una estimación de f considerando los anchos de banda considerados en el capítulo.

Se puede comprobar cómo conociendo la densidad f se puede minimizar el error cuadrático medio integrado asintótico obteniendo la estimación más cercana. Por otro lado, vemos como el estimador del ancho de banda obtenido por validación cruzada insesgada es infraestima el valor de h^*_{AMISE} resultando en una función demasiado abrupta.

De la misma manera, se puede observar como el estimador de *plug-in* sobrestima el parámetro h sistemáticamente el ancho de banda óptimo haciéndose este mayor al considerar más etapas.

La ventaja clara de los métodos de *plug-in* es la rapidez en su cálculo. El hecho de que los métodos de validación cruzada tienen que estimar sus parámetros sobre una malla de valores posibles de h los hace computacionalmente muy pesados. Además, es recomendable realizar la minimización de dichas funciones mediante una malla pues al realizar su minimización se pueden hallar mínimos locales como se ha comentado. Las funciones BCV y UCV del ejemplo anterior son las que siguen

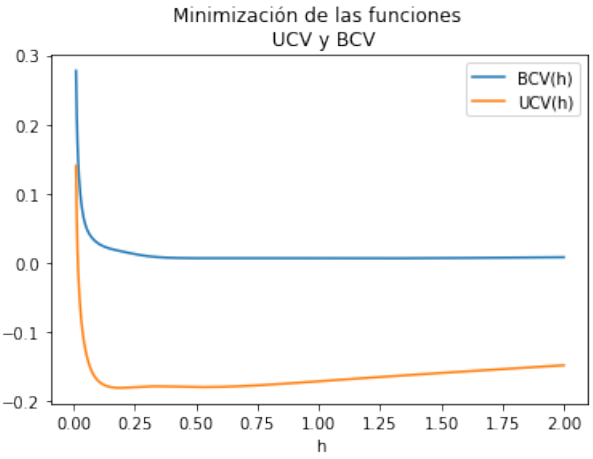


Figura 5: Representación de las funciones $BCV(h)$ y $UCV(h)$

Para indagar en este aspecto más en profundidad, se generan 10 muestras aleatorias de $n = 100$ de una distribución con función de densidad (2.38). Se realiza la selección del ancho de banda óptimo para cada muestra para poder comparar las distribuciones de los selectores del ancho de banda con respecto a h^*_{AMISE}

Así se puede dibujar el diagrama de cajas de $h^*_M - h^*_{AMISE}$ para cada uno de los métodos M calculados para cada una de las muestras.

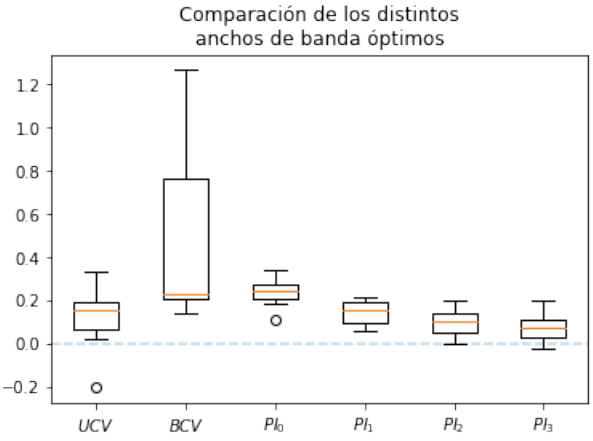


Figura 6: Diagrama de cajas para representar la distribución del ancho de banda óptimo obtenido mediante cada uno de los métodos expuestos anteriormente respecto del óptimo. Se han generado 100 muestras de tamaño $n = 100$ sobre las que se han estimado los anchos de banda óptimos con cada uno de los métodos.

Se puede comprobar que tanto la validación cruzada insegagada como el PI de 2 orden dan resultados mas cercanos al valor óptimo si bien PI resulta en una variabilidad de los datos muy inferior

al UCV. Esto, junto con la eficiencia computacional de PI pueden llevarnos a preferir el estimador PI.

Se puede observar como los estimadores de validación cruzada ofrecen una mayor varianza en cuanto a la estimación se refiere. Por otra parte, la variabilidad del selector es mayor cuantas más etapas se consideran en la estimación de *plu-in*.

Por otra parte, se puede comprobar cómo el selector de *plug-in* de menos etapas sobrestima sistemáticamente el ancho de banda óptimo, problema que se mitiga al considerar más etapas a cambio, eso sí, de aumentar la variabilidad de la estimación.

Por lo general y debido al coste computacional de los métodos de validación cruzada, escoger un método de *plug-in* de 2 etapas suele ser común.

Ejemplo 2. Empleando un núcleo *triweight*

Repetimos el mismo análisis empleando el núcleo *triweight* para la misma muestra generada por (2.38). Se obtienen las siguientes estimaciones de f,

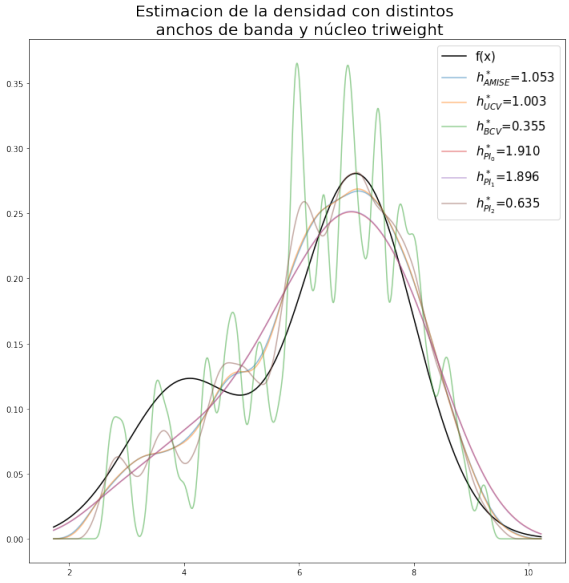


Figura 7: Representación de la función de densidad de (2.38) y la estimación de tipo núcleo mediante un núcleo de tipo triweight para la misma muestra aleatoria tamaño 100 generada en el ejemplo anterior. Se ha realizado una estimación de f considerando los anchos de banda considerados en el capítulo.

La interpretación de los anchos de banda es similar a la realizada en el caso normal. Obtenemos que el estimador por validación cruzada insesgada está infraestimando el valor del ancho de banda óptimo mientras que el resto están sobrestimándolo.

Para indagar en este aspecto más en profundidad, se generan 10 muestras aleatorias de n=100 de una distribución con función de densidad (2.38). Se realiza la selección del ancho de banda óptimo para cada muestra para poder comparar las distribuciones de los selectores del ancho de banda con respecto a h^*_{AMISE}

Así se puede dibujar el diagrama de cajas de $h^*_M - h^*_{AMISE}$ para cada uno de los métodos M calculados para cada una de las muestras.

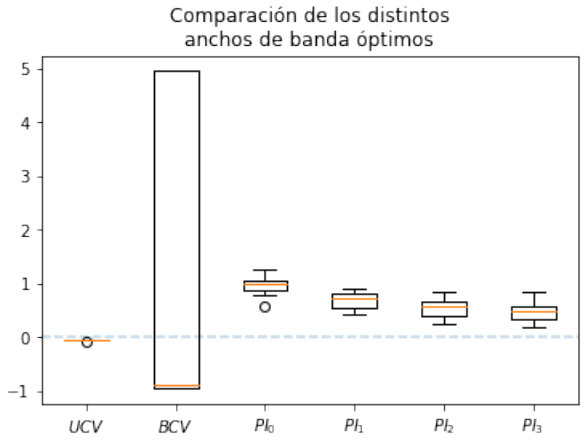


Figura 8: Representación de las funciones BCV y UCV

Se puede observar como el estimadores de validación cruzada sesgada ofrecen una mayor varianza en cuanto a la estimación se refiere. Esto es debido a que en la mayoría de casos, el estimador acababa en el límite inferior o superior de la malla considerada para su análisis. Por otra parte, la variabilidad del selector es mayor cuantas más etapas se consideran en la estimación de PI como habíamos mencionado en el caso anterior.

En este caso resulta llamativo que las estimaciones del ancho de banda PI siempre sobrestiman el ancho de banda óptimo y lo bien que ajusta el método de validación cruzada insesgada.

3. Estimación de la densidad multivariante tipo núcleo

En el caso de la estimación de la densidad multivariante de tipo núcleo es análoga a la presentada en el capítulo anterior con la diferencia de que las observaciones consideradas no son escalares, sino que son vectores de dimensión 2 o mayor. El objetivo es el mismo, haciendo el menor número posible de asunciones sobre la función de densidad de la distribución que ha generado la muestra, se trata de estimar dicha densidad. En este caso, f es una función vectorial.

3.1. Definición del estimador

Sea $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n$ una muestra aleatoria d-dimensional procedente de una distribución de probabilidad (multivariante) continua con función de densidad f.

El estimador de densidad de tipo núcleo se define como:

$$\hat{f}(\boldsymbol{x}; \boldsymbol{H}) = \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} \sum_{i=1}^{n} K\left(\boldsymbol{H}^{-\frac{1}{2}}(\boldsymbol{x} - \boldsymbol{X}_i)\right) \tag{3.1}$$

donde K, el núcleo, es una función integrable que satisface $\int K(\boldsymbol{x})d\boldsymbol{x} = 1$ y $\boldsymbol{H}$, es la matriz de ancho de banda. La matriz $\boldsymbol{H}$ es una matriz positiva definida simétrica de tamaño $d \times d$.

Se puede reescalar la función núcleo mediante el cambio de variable $u = hx$. Así se puede comprobar que el núcleo reescalado, $K_{\boldsymbol{H}}(\boldsymbol{x}) = |\boldsymbol{H}|^{-\frac{1}{2}} K(\boldsymbol{H}^{-\frac{1}{2}}\boldsymbol{x})$, sigue cumpliendo la propiedad de núcleo

$$\int K_{\boldsymbol{H}}(\boldsymbol{u})d\boldsymbol{u} = |\boldsymbol{H}|^{-\frac{1}{2}} \int K(\boldsymbol{H}^{-\frac{1}{2}}\boldsymbol{u})d\boldsymbol{u} \underset{\boldsymbol{u}=\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{x}}{=} |\boldsymbol{H}|^{-\frac{1}{2}} \int K(\boldsymbol{H}^{-\frac{1}{2}}\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{x})|\boldsymbol{H}|^{\frac{1}{2}}d\boldsymbol{x} = \int K(\boldsymbol{x})d\boldsymbol{x} = 1$$

Por tanto, se puede reescribir el estimador de densidad de tipo núcleo (3.1) de forma más compacta como:

$$\hat{f}(\boldsymbol{x}; \boldsymbol{H}) = \frac{1}{n} \sum_{i=1}^{n} K_{\boldsymbol{H}}\left(\boldsymbol{x} - \boldsymbol{X}_i\right) \tag{3.2}$$

Como en el caso univariante, en el multivariante, la única condición que se le exige a la función núcleo es que tenga integral unitaria. Sin embargo, en lo que sigue se considerará que K es una función de densidad esféricamente simétrica. Del teorema 4.1 en (Fourdrinier y col., 2018) vemos que esto se traduce en que la distribución condicional del vector aleatorio $\boldsymbol{x}$ dado un radio, $\|\boldsymbol{x}\| = r$, es la distribución uniforme sobre una esfera de radio r.

Como señala (Chacón y Duong, 2018), el estimador de densidad de tipo núcleo tiene dos posibles interpretaciones. Para un punto $\boldsymbol{x}$ el estimador puede ser considerado una media ponderada donde el peso de $\boldsymbol{X}_i$ disminuye cuando la distancia a $\boldsymbol{x}$ aumenta. Por otra parte podemos considerar que se está extendiendo una masa de probabilidad acorde al núcleo reescalado en los puntos donde está cada observación.

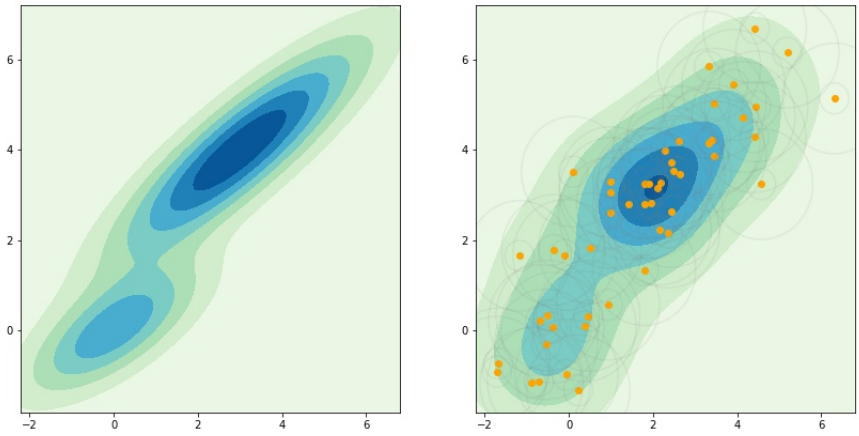


Figura 9: A la izquierda se puede ver la función de densidad de una mixtura de normales $_{0,3N}\left(\begin{pmatrix}0\\0\end{pmatrix},\begin{pmatrix}2 & 1\\1 & 1\end{pmatrix}\right)+_{0,7N}\left(\begin{pmatrix}3\\4\end{pmatrix},\begin{pmatrix}4 & 2{,}8\\2{,}8 & 2{,}6\end{pmatrix}\right)$. A la derecha en naranja los elementos de la muestra (n=50) obtenida de la distribución anterior, en gris las curvas de nivel asociadas a la densidad de una normal $N(\mathbf{0}, \boldsymbol{I})$ desplazada a cada uno de los elementos de la muestra y dividido por el número total de elementos de la muestra. Por último los niveles de color indican la suma de las funciones núcleo situadas en cada elemento de la muestra.

La construcción de las funciones de tipo núcleo multivariante se puede realizar de distintas formas como mencionan (Härdle y col., 2004). Así, la forma más sencilla de crear una función de kernel multidimensional resulta del producto de d núcleos univariantes

$$K^P(x_1, \ldots, x_d) = \prod_{i=1}^{d} K(x_i) \tag{3.3}$$

el problema que presenta está aproximación es que en el caso de funciones con soporte compacto, el soporte del núcleo es un hiperrectángulo.

Otra forma de obtener funciones de tipo kernel multivariantes de funciones univariantes es considerar

$$K^S(\boldsymbol{x}) = K(\|\boldsymbol{x}\|) \tag{3.4}$$

donde $\|\boldsymbol{x}\| = \sqrt{\boldsymbol{x}^T\boldsymbol{x}}$. Así, en el caso de funciones con soporte compacto, el soporte del núcleo es una hiperesfera y por tanto son funciones de núcleo esféricamente simétricas. Esto lo pone de manifiesto Langrené y Warin, 2019 ya que para el método de estimación de densidades multivariantes que proponen, es necesario que el soporte de la función núcleo sea un hiperrectángulo y no una hiperesfera.

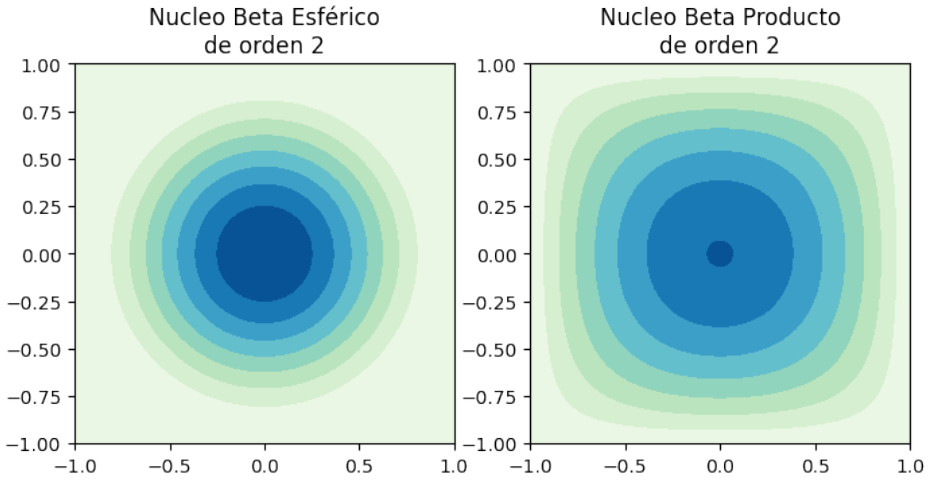


Figura 10: Se muestranlas curvas de nivel para el núcleo Beta bivariante de orden dos. A la izquierda, como núcleo esférico y a la derecha como producto.

Nótese que el núcleo creado a partir de la función de densidad normal multivariante, es esféricamente simétrico lo creemos de cualquiera de las dos formas ya que,

$$K(\boldsymbol{x}) = \prod_{i=1}^{d} K(x_i) = \prod_{i=1}^{d} \frac{1}{\sqrt{2\pi}} e^{-\frac{x_i^2}{2}} = \frac{1}{(2\pi)^{d/2}} \prod_{i=1}^{d} e^{-\frac{x_i^2}{2}} = \frac{1}{(2\pi)^{d/2}} e^{-\frac{1}{2}\sum_{i=1}^{d} x_i^2} = \frac{1}{(2\pi)^{d/2}} e^{-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{x}}$$

reescalando dicha función obtenemos la función núcleo más ampliamente utilizada, el núcleo normal, que se corresponde con la función de densidad normal en d dimensiones,

$$K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{X}_i) = \frac{1}{(2\pi)^{\frac{d}{2}} |\boldsymbol{H}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{X}_i)^T \boldsymbol{H}^{-1} (\boldsymbol{x} - \boldsymbol{X}_i)\right)$$

Teniendo esto en cuenta, podemos establecer la familia de funciones núcleo beta como producto de funciones núcleo marginales univariantes,

$$K^P(\boldsymbol{x}; r) = c_r^d \prod_{i=1}^{d} (1 - x_i^2)^r \mathbf{1}_{\{|x_i| \leq 1\}}(x_i) \tag{3.5}$$

donde, como en el caso univariante, $c_r = \frac{(2r+1)!}{2^{2r+1}(r!)^2} = \frac{1}{\mathcal{B}(r+1,1/2)}$ siendo $\mathcal{B}(a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$ la función beta.

De la misma manera, podemos obtener funciones de tipo núcleo esféricamente simétricas de la familia beta,

$$K(\boldsymbol{x}; r) = C_r (1 - \boldsymbol{x}^T\boldsymbol{x})^r \mathbf{1}_{\{\boldsymbol{x}^T\boldsymbol{x} \leq 1\}}(\boldsymbol{x}) \tag{3.6}$$

siendo el coeficiente $C_r = \frac{2}{dv_d \mathcal{B}(r+1,d/2)}$ tal como lo calcula (Duong, 2015) para dar una fórmula cerrada para la generalización del núcleo multivariante de la familia beta al caso esféricamente simétrico. $v_d = V_d(1, \mathbf{0}) = \frac{\pi^{\mathbf{d/2}}}{\mathbf{\Gamma}(\frac{\mathbf{d}}{\mathbf{2}}+\mathbf{1})}$ es el volumen de la hiperesfera de dimensión d y radio 1 centrada en $\mathbf{0}$.

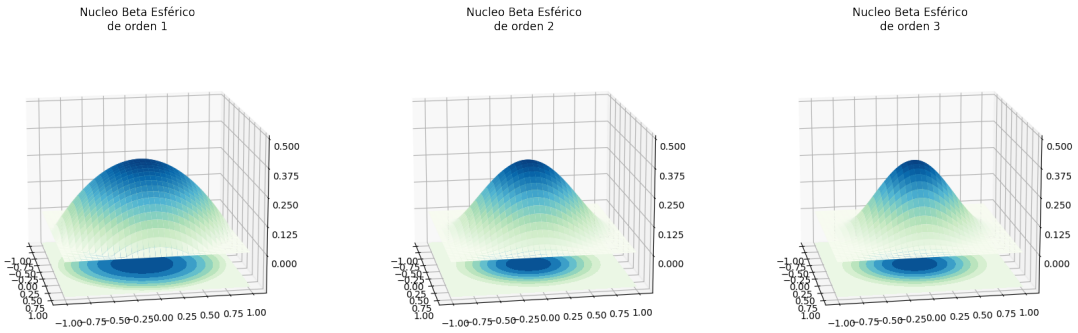


Figura 11: Se muestran tanto la superficie como las curvas de nivel para el núcleo Beta bivariante esférico.

Generalizando al caso multivariante, denotaremos las funciones núcleo esféricamente simétricas $K_{\boldsymbol{H}}(\boldsymbol{x}; r)$ para $r = 0, 1, 2, 3$ como las generalizaciones multivariantes uniforme, Epanechnikov, Biweight y Triweight respectivamente.

Nosotros, durante el resto del trabajo supondremos que el núcleo es esféricamente simétrico.

Además, podemos considerar otras de funciones de densidad asociadas a distribuciones que cumplen dicha propiedad como la distribución t de student multivariable o la distribución de Laplace multivariable entre otras.

En el caso multivariante, la matriz de ancho de banda ofrece una clase de posibilidades más rica que en el univariante. Así en este caso pueden considerarse distintos conjuntos de matices a considerar para la modelización del ancho de banda.

La opción más simple es la de considerar matrices diagonales. Así, considerando matrices del conjunto $\mathcal{A} = \{h^2 \boldsymbol{I}_d : h > 0\}$el estimador se reduce a

$$\hat{f}(\boldsymbol{x}; h) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\boldsymbol{x} - \boldsymbol{X}_i}{h}\right)$$

En este caso, como menciona (Duong, 2004), sería necesario una estandarizacion de los datos antes de realizar la estimación de tipo kernel, para después recuperar la original con la transformación inversa.

Una clase más flexible de de matrices diagonales permite valores distintos entre dimensiones si bien no permite correlación entre las mismas, $\mathcal{D} = \{\text{diag}(h_1^2, \ldots, h_d^2) : h_1, \ldots, h_d > 0\}$

La clase más flexible de matrices es la de matrices simétricas definidas positivas sin restricciones que se denotan por $\mathcal{F} = \{\boldsymbol{H} \in \mathcal{M}_d : \boldsymbol{H} > 0, \boldsymbol{H} = \boldsymbol{H}^T\}$

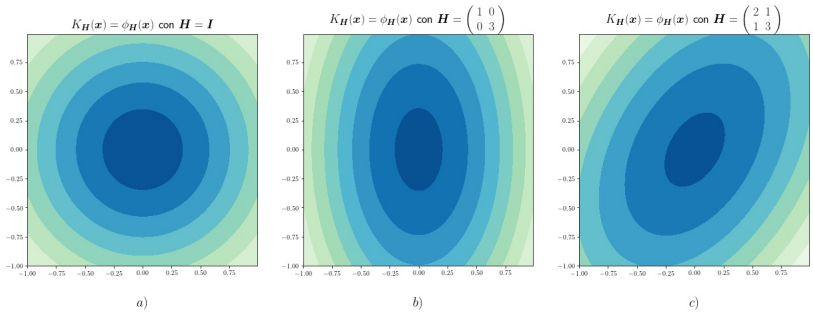


Figura 12: Efecto sobre la función núcleo de tipo normal de matrices de anchos de banda pertenecientes a las clases mencionadas anteriormente. a) $\boldsymbol{H} \in \mathcal{A}$, a) $\boldsymbol{H} \in \mathcal{D}$ y c) $\boldsymbol{H} \in \mathcal{F}$.

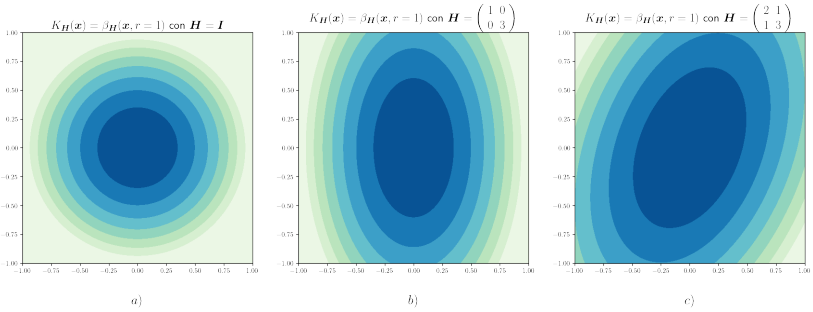


Figura 13: Efecto sobre la función núcleo de tipo beta esférico de orden r de matrices de anchos de banda pertenecientes a las clases mencionadas anteriormente. a) $\boldsymbol{H} \in \mathcal{A}$, a) $\boldsymbol{H} \in \mathcal{D}$ y c) $\boldsymbol{H} \in \mathcal{F}$.

3.2. Propiedades teóricas

El análisis de las propiedades teóricas exactas es completamente análogo al del caso univariante. La diferencia con el caso multivariante reside en el cálculo de las expresiones asintóticas, que si paralelas al del caso univariante, resultan más tediosas en cuanto a la notación y la intuición de las mismas.

Se desarrollan las expresiones como en (2.2) si bien los desarrollos como hemos comentado anteriormente varían únicamente en la notación vectorial de los argumentos utilizados.

Es necesario establecer ciertas consideraciones que conciernen a la notación empleada de aquí en adelante en lo que respecta al estimador de densidad de tipo núcleo multivariante.

Abusando de la notación, se emplea la expresión $\int f(\boldsymbol{x})d\boldsymbol{x}$ para hacer referencia a $\int_{\mathbb{R}^d} f(\boldsymbol{x})d\boldsymbol{x}$ como en (2). El espacio de integración queda claro en cada caso de la dimensión del integrando.

En el caso multivariante, la distancia L^2 de una función vectorial, $f : \mathbb{R}^d \to \mathbb{R}^p$, se define como la

matriz de dimensión $p \times p$

$$\boldsymbol{R}(f) = \int f(\boldsymbol{x}) f(\boldsymbol{x})^T d\boldsymbol{x}$$

mientras que en el caso de funciones $f : \mathbb{R}^d \to \mathbb{R}$ la notación resulta consistente como en la sección anterior,

$$R(f) = \int f(\boldsymbol{x})^2 d\boldsymbol{x}$$

Por otra parte en el desarrollo asintótico será necesaria la establecer de forma sistemática las derivadas parciales de orden r. Se seguirá la notación propuesta por (Chacón y Duong, 2018). Así, el operador diferencial de orden r se define a través del operador diferencial D,

$$D = \left(\frac{\partial}{\partial x_1}, \ldots, \frac{\partial}{\partial x_d} \right)^T$$

como el producto de Kronecker de D r veces consigo mismo, $D^{\otimes r}$. Así,

$$D^{\otimes 2} = D \otimes D = vec(DD^T) = vec(H)$$

donde H es la matriz Hessiana ya que esta puede expresarse como $H = DD^T$.

Con estas consideraciones, se pueden desarrollar las propiedades del estimador de densidad de tipo núcleo multivariantes.

En el caso multivariante, el error cuadrático medio del estimador de densidad de tipo núcleo, el error en un punto se define de la misma manera que en el caso univariante

$$MSE\left[\hat{f}(\boldsymbol{x}; \boldsymbol{H})\right] = \mathbb{E}\left[\left(\hat{f}(\boldsymbol{x}; \boldsymbol{H}) - f(\boldsymbol{x})\right)^2\right]$$

que puede descomponerse mediante el sesgo y la varianza del estimador de la siguiente manera

$$\begin{aligned} MSE\left[\hat{f}(\boldsymbol{x}; \boldsymbol{H})\right] &= \left(\mathbb{E}\left[\hat{f}(\boldsymbol{x}; \boldsymbol{H})\right] - f(\boldsymbol{x})\right)^2 + Var\left(\hat{f}(\boldsymbol{x}; \boldsymbol{H})\right) \\ &= sesgo^2\left(\hat{f}(\boldsymbol{x}; \boldsymbol{H})\right) + Var\left(\hat{f}(\boldsymbol{x}; \boldsymbol{H})\right) \end{aligned}$$

Para obtener el cálculo del error cuadrático medio en un punto, en primer lugar se debe calcular la esperanza del estimador en un punto. El cálculo es completamente análogo al caso unidimensional

$$\begin{aligned} \mathbb{E}\left[\hat{f}(\boldsymbol{x}; \boldsymbol{H})\right] &= \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{X}_i)\right] = \frac{1}{n}\mathbb{E}\left[\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{X}_i)\right] = \\ &= \frac{1}{n} n \mathbb{E}\left[K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{X}_1)\right] = \int K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{y}) f(\boldsymbol{y}) d\boldsymbol{y} = (K_{\boldsymbol{H}} * f)(\boldsymbol{x}) \end{aligned} \tag{3.7}$$

En el tercer paso se ha tenido en cuenta que $\{\boldsymbol{X}_i\}_{i=1}^n$ son vectores aleatorios idénticamente distribuidos de una distribución cuya función de densidad es f. Como puede observarse, la esperanza se reduce a la convolución entre el núcleo y la función de densidad generadora de la muestra. Así, el sesgo puede escribirse como,

$$sesgo\left(\hat{f}(\boldsymbol{x}; \boldsymbol{H})\right) = (K_{\boldsymbol{H}} * f)(\boldsymbol{x}) - f(\boldsymbol{x})$$

Por otra parte, el calculo de la varianza también es idéntico al caso univariante,

$$\begin{aligned} Var\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right) &= Var\left(\frac{1}{n}\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)\right) = \frac{1}{n^2} Var\left(\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)\right) = \\ &= \frac{1}{n^2}\sum_{i=1}^{n} Var\left(K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)\right) = \frac{1}{n} Var\left(K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)\right) = \\ &= \frac{1}{n}\left(\mathbb{E}\left[K^2_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)\right] - \mathbb{E}\left[K^2_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)\right]^2\right) = \\ &= \frac{1}{n}\left(\int K^2_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{y}) f(\boldsymbol{y}) d\boldsymbol{y} - (K_{\boldsymbol{H}} * f)^2(\boldsymbol{x})\right) = \\ &= \frac{1}{n}\left((K^2_{\boldsymbol{H}} * f)(\boldsymbol{x}) - (K_{\boldsymbol{H}} * f)^2(\boldsymbol{x})\right) \end{aligned} \tag{3.8}$$

en el tercer paso se ha tenido en cuenta que las variables de la muestra son independientes y en el cuarto que provienen de la misma distribución de probabilidad.

Por tanto, el error cuadrático medio quedaría como:

$$MSE\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right) = \frac{1}{n}\left((K^2_{\boldsymbol{H}} * f)(\boldsymbol{x}) - (K_{\boldsymbol{H}} * f)^2(\boldsymbol{x})\right) + \left((K_{\boldsymbol{H}} * f)(\boldsymbol{x}) - f(\boldsymbol{x})\right)^2 \tag{3.9}$$

Igual que en el caso unidimensional, puede que el interés resida en las propiedades globales del estimador más allá de las locales. Para ello, se integrar el error cuadrático medio en $\mathbb{R}^n$ y obteniendo una medida del error del estimador global.

Por tanto, el error cuadrático medio integrado del estimador de se define como:

$$\begin{aligned} MISE\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right) &= \int MSE\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right) d\boldsymbol{x} = \\ &= \int \text{sesgo}\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right)^2 d\boldsymbol{x} + \int Var\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right) d\boldsymbol{x} \end{aligned} \tag{3.10}$$

así, de (3.9) podemos integrar la expresión del error cuadrático medio

$$\begin{aligned} MISE\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right) &= \frac{1}{n}\int (K^2_{\boldsymbol{H}} * f)(\boldsymbol{x}) d\boldsymbol{x} - \frac{1}{n}\int (K_{\boldsymbol{H}} * f)^2(\boldsymbol{x}) d\boldsymbol{x} + \int \left((K_{\boldsymbol{H}} * f)(\boldsymbol{x}) - f(\boldsymbol{x})\right)^2 d\boldsymbol{x} = \\ &= \frac{1}{n}\int (K^2_{\boldsymbol{H}} * f)(\boldsymbol{x}) d\boldsymbol{x} - \frac{1}{n}\int (K_{\boldsymbol{H}} * f)^2(\boldsymbol{x}) d\boldsymbol{x} \\ &\quad + \int (K_{\boldsymbol{H}} * f)^2(\boldsymbol{x}) d\boldsymbol{x} - 2\int (K_{\boldsymbol{H}} * f)(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x} + \int f(\boldsymbol{x})^2 d\boldsymbol{x} = \\ &= \frac{1}{n}\int (K^2_{\boldsymbol{H}} * f)(\boldsymbol{x}) d\boldsymbol{x} + \left(1 - \frac{1}{n}\right)\int (K_{\boldsymbol{H}} * f)^2(\boldsymbol{x}) d\boldsymbol{x} \\ &\quad - 2\int (K_{\boldsymbol{H}} * f)(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x} + R(f) \end{aligned}$$

El primer término a su vez puede reescribirse de la siguiente manera:

$$\begin{aligned} \frac{1}{n}\int (K^2_{\boldsymbol{H}} * f)(\boldsymbol{x}) d\boldsymbol{x} &= \frac{1}{n}\int\int K^2_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{y}) f(\boldsymbol{y}) d\boldsymbol{y} d\boldsymbol{x} = \frac{1}{n}\int\int K^2_{\boldsymbol{H}}(\boldsymbol{y}) f(\boldsymbol{x}-\boldsymbol{y}) d\boldsymbol{x} d\boldsymbol{y} = \\ &= \frac{1}{n}\int K^2_{\boldsymbol{H}}(y)\int f(\boldsymbol{x}-\boldsymbol{y}) d\boldsymbol{x} d\boldsymbol{y} = \frac{1}{n}\int K^2_{\boldsymbol{H}}(\boldsymbol{y}) d\boldsymbol{y} = \\ &= \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}|\boldsymbol{H}|^{-\frac{1}{2}}\int K^2\left(\boldsymbol{H}^{-\frac{1}{2}}\boldsymbol{y}\right) d\boldsymbol{y} = \frac{1}{n|\boldsymbol{H}|}\int K^2(\boldsymbol{y}) d\boldsymbol{y} \underset{y=\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z}}{=} \\ &= \frac{1}{n|\boldsymbol{H}|}\int K^2(\boldsymbol{z})|\boldsymbol{H}|^{\frac{1}{2}} d\boldsymbol{z} = \frac{1}{n|\boldsymbol{H}|^{\frac{1}{2}}}\int K^2(\boldsymbol{z}) d\boldsymbol{z} = \frac{1}{n|\boldsymbol{H}|^{\frac{1}{2}}} R(K) \end{aligned}$$

por lo que el error cuadrático medio integrado es:

$$MISE\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right) = \frac{1}{n|\boldsymbol{H}|^{\frac{1}{2}}}R(K) + \left(1-\frac{1}{n}\right)\int (K_{\boldsymbol{H}} * f)^2(\boldsymbol{x})d\boldsymbol{x} - 2\int (K_{\boldsymbol{H}} * f)(\boldsymbol{x})f(\boldsymbol{x})d\boldsymbol{x} + R(f) \tag{3.11}$$

Como en el caso univariante se puede pretender obtener un valor aleatorio ligado a la muestra. En este caso podemos calcular el error cuadrático integrado prescindiendo de la esperanza en (3.10).

$$\begin{aligned} ISE\left(\hat{f}(\cdot;\boldsymbol{H})\right) &= \int \left(\hat{f}(\boldsymbol{x};\boldsymbol{H}) - f(\boldsymbol{x})\right)^2 d\boldsymbol{x} = \\ &= \int \hat{f}(\boldsymbol{x};\boldsymbol{H})^2 - 2\hat{f}(\boldsymbol{x};\boldsymbol{H})f(\boldsymbol{x}) + f(\boldsymbol{x})^2 d\boldsymbol{x} \end{aligned} \tag{3.12}$$

3.2.1. Propiedades asintóticas

Al igual que en el caso univariante, se realiza un estudio de sus propiedades asintóticas. Así, desarrollando los primeros momentos del estimador se pueden obtener expresiones que dependen del ancho de banda de una manera más sencilla para muestras grandes.

De la formulación de las derivadas mostrada anteriormente, es necesario establecer una formulación para la expansión de Taylor multivariante para una función derivable r veces

$$f(\boldsymbol{x}+\boldsymbol{a}) = \sum_{j=0}^{r} \frac{1}{j!} D^{\otimes j} f(\boldsymbol{x})^T \boldsymbol{a}^{\otimes j} + o(\|\boldsymbol{a}\|^r)$$

En este sentido, se requieren ciertas propiedades de regularidad en la función de densidad f, de la función núcleo y de la secuencia de matrices de ancho de banda para poder realizar los correspondientes desarrollos. Las condiciones son completamente análogas a las desarrolladas en la sección 2.2.1.

- f es cuadrado integrable y diferenciable dos veces con todas sus derivadas de segundo orden acotadas, continuas y cuadrado integrables.
- Las matrices de ancho de banda $\boldsymbol{H} = \boldsymbol{H}_n$ es una secuencia de matrices definidas positivos tal que
$$vec(H) \to 0, \quad n^{-1}|\boldsymbol{H}|^{-\frac{1}{2}} \to 0 \quad \text{cuado} \quad n \to \infty$$
- K es una función de densidad esféricamente simétrica con momento de orden dos finito. Esto se traduce en
$$\int \boldsymbol{z}K(\boldsymbol{z})d\boldsymbol{z} = 0 \quad \text{y} \quad \int \boldsymbol{z}^{\otimes 2}K(\boldsymbol{z}) = \mu_2(K)vec(\boldsymbol{I}_d)$$
donde $\mu_2(K) = \int z_i^2 K(\boldsymbol{z})d\boldsymbol{z}\ \forall i = 1,\dots,d$.

Teniendo en cuenta la definición del estimador de tipo núcleo (3.1), su esperanza se puede expresar

de la siguiente manera partiendo de (3.7),

$$\mathbb{E}\left[\hat{f}(\boldsymbol{x};\boldsymbol{H})\right] = \int K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{y})f(\boldsymbol{y})d\boldsymbol{y} \underset{\boldsymbol{y}=\boldsymbol{x}-\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z}}{=} \int K_{\boldsymbol{H}}(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z})f(\boldsymbol{x}-\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z})|\boldsymbol{H}|^{\frac{1}{2}}d\boldsymbol{z} =$$
$$= \int |\boldsymbol{H}|^{-\frac{1}{2}}K(\boldsymbol{H}^{-\frac{1}{2}}\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z})f(\boldsymbol{x}-\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z})|\boldsymbol{H}|^{\frac{1}{2}}d\boldsymbol{z} = \int K(\boldsymbol{z})f(\boldsymbol{x}-\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z})d\boldsymbol{z}$$

Ahora, desarrollando $f(\boldsymbol{x}-\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z})$ en torno a $\boldsymbol{x}$

$$f(\boldsymbol{x}-\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z}) = f(\boldsymbol{x}) - Df(\boldsymbol{x})^T\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z} + \frac{1}{2}D^{\otimes 2}f(\boldsymbol{x})^T(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z})^{\otimes 2} + o(\|vec\boldsymbol{H}\|)$$

se obtiene

$$\mathbb{E}\left[\hat{f}(\boldsymbol{x};\boldsymbol{H})\right] = \int K(\boldsymbol{z})\left(f(\boldsymbol{x}) - Df(\boldsymbol{x})^T\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z} + \frac{1}{2}D^{\otimes 2}f(\boldsymbol{x})^T(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z})^{\otimes 2} + o(\|vec\boldsymbol{H}\|)\right)d\boldsymbol{z} =$$
$$= f(\boldsymbol{x})\int K(\boldsymbol{z})d\boldsymbol{z} - Df(\boldsymbol{x})^T\boldsymbol{H}^{\frac{1}{2}}\int \boldsymbol{z}K(\boldsymbol{z})d\boldsymbol{z} +$$
$$+ \frac{1}{2}D^{\otimes 2}f(\boldsymbol{x})^T\boldsymbol{H}^{\frac{1}{2}^{\otimes 2}}\int z^{\otimes 2}K(\boldsymbol{z})d\boldsymbol{z} + o(\|vec\boldsymbol{H}\|) =$$
$$= f(\boldsymbol{x}) + \frac{1}{2}D^{\otimes 2}f(\boldsymbol{x})^T\boldsymbol{H}^{\frac{1}{2}^{\otimes 2}}\int z^{\otimes 2}K(\boldsymbol{z})d\boldsymbol{z} + o(\|vec\boldsymbol{H}\|) =$$
$$= f(\boldsymbol{x}) + \frac{1}{2}D^{\otimes 2}f(\boldsymbol{x})^T\boldsymbol{H}^{\frac{1}{2}^{\otimes 2}}\mu_2(K)vec(\boldsymbol{I}_d) + o(\|vec\boldsymbol{H}\|) =$$
$$= f(\boldsymbol{x}) + \frac{1}{2}D^{\otimes 2}f(\boldsymbol{x})^T\mu_2(K)vec(\boldsymbol{H}) + o(\|vec\boldsymbol{H}\|)$$

en el penúltimo paso se ha tenido en cuenta que $\int K(\boldsymbol{z})d\boldsymbol{z} = 1$ por ser una función de densidad de probabilidad y $\int \boldsymbol{z}K(\boldsymbol{z})d\boldsymbol{z} = 0$ por ser simétrica y estar centrada en el origen. Además se han empleado dos propiedades sobre el producto de Kronecker. En el paso 2, se ha empleado la propiedad $(A\otimes B)(C\otimes D) = AC\otimes BD$ para desarrollar, $(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z})\otimes(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z}) = (\boldsymbol{H}^{\frac{1}{2}}\otimes\boldsymbol{H}^{\frac{1}{2}})(\boldsymbol{z}\otimes\boldsymbol{z})$. Por otra parte, en el último paso se ha empleado la propiedad $vec(ABC) = (C^T\otimes A)vec(B)$ para reescribir $(\boldsymbol{H}^{\frac{1}{2}}\otimes\boldsymbol{H}^{\frac{1}{2}})vec(\boldsymbol{I}_d) = vec(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{I}_d\boldsymbol{H}^{\frac{1}{2}}) = vec(\boldsymbol{H})$. Las demostraciones de las propiedades del producto de Kronecker pueden consultarse entere otors, en el capítulo dos de (**Graham2018)**.

De este desarrollo es inmediato establecer una formulación simple para el sesgo del estimador ya que

$$\mathbb{E}\left[\hat{f}(\boldsymbol{x};\boldsymbol{H})\right] - f(\boldsymbol{x}) = \frac{1}{2}D^{\otimes 2}f(\boldsymbol{x})^T\mu_2(K)vec(\boldsymbol{H}) + o(\|vec\boldsymbol{H}\|) \tag{3.13}$$

Asimismo, para calcular el error cuadrático medio integrado, es necesario calcular el cuadrado del sesgo integrado.

$$\int \text{sesgo}\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right)^2 d\boldsymbol{x} = \frac{1}{4}\mu_2(K)^2\int vec(\boldsymbol{H})^T D^{\otimes 2}f(\boldsymbol{x})D^{\otimes 2}f(\boldsymbol{x})^T vec(\boldsymbol{H})d\boldsymbol{x} + o(\|vec(\boldsymbol{H})\|^2) =$$
$$= \frac{1}{4}\mu_2(K)^2 vec(\boldsymbol{H})^T\int D^{\otimes 2}f(\boldsymbol{x})D^{\otimes 2}f(\boldsymbol{x})^T d\boldsymbol{x}\ vec(\boldsymbol{H}) + o(\|vec(\boldsymbol{H})\|^2) =$$
$$= \frac{1}{4}\mu_2(K)^2 vec(\boldsymbol{H})^T R(D^{\otimes 2}f)vec(\boldsymbol{H}) + o(\|vec(\boldsymbol{H})\|^2)$$
(3.14)

Teniendo en cuenta que $Df(\boldsymbol{x}) \otimes Df(\boldsymbol{x}) = vec(Df(\boldsymbol{x})Df(\boldsymbol{x})^T) = vec(Hf(x))$ donde H representa la matriz Hessiana y que como $vec(A)^T vec(B) = tr(A^T B)$ entonces, tenemos que $vec(H)^T vec(D^{\otimes 2}) = tr(\boldsymbol{H}H)$ y por tanto, el sesgo al cuadrado integrado, se puede reescribir como,

$$\begin{aligned}\int \text{sesgo}\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right)^2 d\boldsymbol{x} &= \frac{1}{4}\mu_2(K)^2 \int vec(\boldsymbol{H})^T D^{\otimes 2} f(\boldsymbol{x}) D^{\otimes 2} f(\boldsymbol{x})^T vec(\boldsymbol{H}) d\boldsymbol{x} + o(\|vec(\boldsymbol{H})\|^2) = \\ &= \frac{1}{4}\mu_2(K)^2 \int tr(\boldsymbol{H}Hf(\boldsymbol{x}))^2 d\boldsymbol{x} + o(\|vec(\boldsymbol{H})\|^2)\end{aligned} \tag{3.15}$$

Alternativamente, como mencionan (Chacón y Duong, 2010), se puede dar una formulación alternativa a (3.14) y (3.15). Partiendo de (3.14) teniendo en cuenta que $vec(\boldsymbol{H})^T R(D^{\otimes 2} f) vec(\boldsymbol{H}) \in \mathbb{R}$,entonces, $vec(\boldsymbol{H})^T R(D^{\otimes 2} f) vec(\boldsymbol{H}) = vec(vec(\boldsymbol{H})^T R(D^{\otimes 2} f) vec(\boldsymbol{H}))$ y haciendo uso de la propiedad que se ha mencionado anteriormente $vec(ABC) = (C^T \otimes A) vec(B)$ tenemos,

$$\begin{aligned}\int \text{sesgo}\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right)^2 d\boldsymbol{x} &= \frac{1}{4}\mu_2(K)^2 vec(\boldsymbol{H})^T R(D^{\otimes 2} f) vec(\boldsymbol{H}) + o(\|vec(\boldsymbol{H})\|^2) = \\ &= \frac{1}{4}\mu_2(K)^2 (vec(\boldsymbol{H})^T)^{\otimes 2} vec(R(D^{\otimes 2} f)) + o(\|vec(\boldsymbol{H})\|^2)\end{aligned} \tag{3.16}$$

Como se ha hecho en el caso univariante, también se desarrolla el término de la varianza del estimador

$$\begin{aligned}Var\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right) &= \frac{1}{n}\int K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{y})^2 f(\boldsymbol{y}) d\boldsymbol{y} - \frac{1}{n}\mathbb{E}^2\left[\hat{f}(\boldsymbol{x};\boldsymbol{H})\right] \underset{\boldsymbol{y}=\boldsymbol{x}-\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z}}{=} \\ &= \frac{1}{n}\int K^2_{\boldsymbol{H}}(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z}) f(\boldsymbol{x}-\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z}) |\boldsymbol{H}|^{\frac{1}{2}} d\boldsymbol{z} - \frac{1}{n}(f(\boldsymbol{x})^2 + o(1)) = \\ &= \frac{1}{n}\int K^2_{\boldsymbol{H}}(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z}) f(\boldsymbol{x}-\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z}) |\boldsymbol{H}|^{\frac{1}{2}} d\boldsymbol{z} - \frac{1}{n}(f(\boldsymbol{x})^2 + o(1)) = \\ &= \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}|\boldsymbol{H}|^{-\frac{1}{2}}\int K^2(\boldsymbol{z}) f(\boldsymbol{x}-\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z}) |\boldsymbol{H}|^{\frac{1}{2}} d\boldsymbol{z} - \frac{1}{n}f(\boldsymbol{x})^2 + o\left(\frac{1}{n}\right) = \\ &= \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}\int K^2(\boldsymbol{z})(f(\boldsymbol{x}) + o(1)) d\boldsymbol{z} - \frac{1}{n}f(\boldsymbol{x})^2 + o\left(\frac{1}{n}\right) = \\ &= \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} f(\boldsymbol{x}) R(K) + o\left(\frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}\right)\end{aligned} \tag{3.17}$$

en la última igualdad se ha utilizado el hecho de que $\frac{1}{n}f(\boldsymbol{x})^2 + o\left(\frac{1}{n}\right)$ es de orden menor a $o(n^{-1}|\boldsymbol{H}|^{-\frac{1}{2}})$ ya que como hemos supuesto que $vec(\boldsymbol{H}) \to 0$ entonces $|\boldsymbol{H}| \to 0$.

En este punto, el cálculo de la varianza integrada necesaria para el cálculo del error cuadrático medio integrado es inmediato ya que solo $f(\boldsymbol{x})$ depende de $\boldsymbol{x}$,

$$\begin{aligned}\int Var\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right) d\boldsymbol{x} &= \int \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} f(\boldsymbol{x}) R(K) + o\left(\frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}\right) d\boldsymbol{x} = \\ &= \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} R(K)\int f(\boldsymbol{x}) d\boldsymbol{x} + o\left(\frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}\right) = \\ &= \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} R(K) + o\left(\frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}\right)\end{aligned} \tag{3.18}$$

Teniendo en cuenta la descomposición en la suma del cuadrado del sesgo y la varianza del error cuadrático medio, teniendo en cuenta (3.13) y (3.17), obtenemos el error cuadrático medio del estimador

$$\begin{aligned} MSE\left(\hat{f}(\boldsymbol{x};\boldsymbol{H})\right) = \frac{1}{4}\mu_2(K)^2 vec(\boldsymbol{H})^T D^{\otimes 2} f(\boldsymbol{x}) D^{\otimes 2} f(\boldsymbol{x})^T vec(\boldsymbol{H}) + \\ + \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} f(\boldsymbol{x}) R(K) + o\left(\frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} + \|vec(\boldsymbol{H})\|^2\right) \end{aligned} \tag{3.19}$$

Y de igual manera, con (3.10) y los desarrollos del sesgo integrado (3.14, 3.15 y 3.16) y la varianza integrada (3.18) del estimador, el error cuadrático medio integrado del estimador se puede formular de la siguiente manera

$$\begin{aligned} MISE\left(\hat{f}(\cdot;\boldsymbol{H})\right) &= \frac{1}{4}\mu_2(K)^2 \int tr(\boldsymbol{H}Hf(\boldsymbol{x}))^2 d\boldsymbol{x} + \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} R(K) + \\ &\quad + o\left(\frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} + \|vec(\boldsymbol{H})\|^2\right) = \\ &= \frac{1}{4}\mu_2(K)^2 vec(\boldsymbol{H})^T R(D^{\otimes 2} f) vec(\boldsymbol{H}) + \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} R(K) + \\ &\quad + o\left(\|vec(\boldsymbol{H})\|^2 + \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}\right) = \\ &= \frac{1}{4}\mu_2(K)^2 (vec(\boldsymbol{H})^T)^{\otimes 2} vec(R(D^{\otimes 2} f)) + \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} R(K) + \\ &\quad + o\left(\|vec(\boldsymbol{H})\|^2 + \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}\right) = \end{aligned} \tag{3.20}$$

Así, el error cuadrático medio integrado asintótico

$$\begin{aligned} AMISE\left(\hat{f}(\cdot;\boldsymbol{H})\right) &= \frac{1}{4}\mu_2(K)^2 \int tr(\boldsymbol{H}Hf(\boldsymbol{x}))^2 d\boldsymbol{x} + \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} R(K) = \\ &= \frac{1}{4}\mu_2(K)^2 vec(\boldsymbol{H})^T R(D^{\otimes 2} f) vec(\boldsymbol{H}) + \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} R(K) = \\ &= \frac{1}{4}\mu_2(K)^2 (vec(\boldsymbol{H})^T)^{\otimes 2} vec(R(D^{\otimes 2} f)) + \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} R(K) \end{aligned} \tag{3.21}$$

Cabe notar la similitud entre la primera igualdad de la ecuación anterior y (2.14). Mientras que en el caso univariante teníamos un término $R(f'')$, en el multivariante tenemos la integral del cuadrado de la traza de la matriz Hessiana multiplicada por $\boldsymbol{H}$.

3.3. El problema de selección del ancho de banda y ancho de banda óptimo

Como mencionan (Wand y Jones, 1994), a diferencia del caso univariante, en el caso multivariante no se dispone de expresiones explícitas para el calculo de la matriz de ancho de banda óptima AMISE en general.

Aun así, restringiendo la selección de la matriz de ancho de banda a $\mathcal{A}$ podemos obtener expresiones que simplifican (3.21).

Suponiendo que $\boldsymbol{H} = h^2\boldsymbol{I} \in \mathcal{A}$, podemos simplificar la expresión $vec(\boldsymbol{H})^T R(D^{\otimes 2} f) vec(\boldsymbol{H}) = \int vec(\boldsymbol{H})^T D^{\otimes 2} f(\boldsymbol{x}) D^{\otimes 2} f(\boldsymbol{x})^T vec(\boldsymbol{H}) d\boldsymbol{x}$ en (3.21).

Teniendo en cuenta que cada elemento del producto puede expresarse como $(D^{\otimes 2} f)^T vec(\boldsymbol{H}) = D^{\otimes 2} f^T vec(h^2 \boldsymbol{I}_d)$, y que $(D \otimes D)^T vec(\boldsymbol{I}) = D^T D = tr(H)$ donde H es la matriz Hessiana,

$$\begin{aligned} vec(\boldsymbol{H})^T R(D^{\otimes 2} f) vec(\boldsymbol{H}) &= \int vec(\boldsymbol{H})^T D^{\otimes 2} f(\boldsymbol{x}) D^{\otimes 2} f(\boldsymbol{x})^T vec(\boldsymbol{H}) d\boldsymbol{x} = \\ &= \int tr(Hf(\boldsymbol{x})) tr(Hf(\boldsymbol{x})) d\boldsymbol{x} = R(tr(Hf)) \end{aligned}$$

Con esto y teniendo en cuenta que $|\boldsymbol{H}|^{-\frac{1}{2}} = |h^2 \boldsymbol{I}_d|^{-\frac{1}{2}} = (h^{2d} |\boldsymbol{I}_d|)^{-\frac{1}{2}} = h^{-d}$, se puede reformular (3.21) como

$$AMISE\left(\hat{f}(\boldsymbol{x}; \boldsymbol{H})\right) = \frac{1}{4}\mu_2(K)^2 h^4 R(tr(Hf)) + \frac{1}{nh^d} R(K) \tag{3.22}$$

En este caso, restringiendo $\boldsymbol{H} = h^2 \boldsymbol{I} \in \mathcal{A}$ podemos obtener una matriz de ancho de banda óptima AMISE,

$$\boldsymbol{H}^*_{AMISE} = \underset{\boldsymbol{H} \in \mathcal{A}}{\operatorname{argmin}}\ AMISE\left(\hat{f}(\boldsymbol{x}; \boldsymbol{H})\right)$$

que se reduce a encontrar el valor óptimo de h. Entonces, derivando (3.24) con respecto a h e igualando a cero,

$$\frac{dAMISE\left(\hat{f}(x; h)\right)}{dh} = \mu_2(K)^2 R(tr(Hf)) h^3 + \frac{-d}{nh^{d+1}} R(K) = 0$$

por tanto, despejando h

$$h^*_{AMISE} = \left(\frac{d\ R(K)}{n\mu_2^2(K) R(tr(Hf))}\right)^{\frac{1}{d+4}} \tag{3.23}$$

Observamos que h en este caso aumenta con el tamaño de d, es decir la dimensionalidad del espacio. Según comenta (García-Portugués, 2021), este hecho puede entenderse intuitivamente como la necesidad de $\boldsymbol{H}_{AMISE}$ de agrandarse para considerar entornos más grandes alrededor de X_i a medida que aumenta el vacío del espacio $\mathbb{R}^p$.

Podemos considerar el caso más general en el que $\boldsymbol{H} = \operatorname{diag}\left(h_1^2, \ldots, h_d^2\right) \in \mathcal{D}$. Como en el caso anterior, simplificaremos $vec(\boldsymbol{H})^T R(D^{\otimes 2} f) vec(\boldsymbol{H}) = \int vec(\boldsymbol{H})^T D^{\otimes 2} f(\boldsymbol{x}) D^{\otimes 2} f(\boldsymbol{x})^T vec(\boldsymbol{H}) d\boldsymbol{x}$ en (3.21).

Teniendo en cuenta que cada elemento del producto puede expresarse como $(D^{\otimes 2} f)^T vec(\boldsymbol{H}) = D^{\otimes 2} f^T vec\left(\operatorname{diag}(h_1^2, \ldots, h_d^2)\right)$, y que $(D \otimes D)^T vec(\boldsymbol{H}) = D^T \boldsymbol{H} D = tr(\operatorname{diag}(h_1^2, \ldots, h_d^2) \odot H)$ donde H es la matriz Hessiana. Nótese que $tr(\operatorname{diag}(h_1^2, \ldots, h_d^2) \odot H) = \sum_{i=1}^d h_i^2 \frac{\partial^2}{\partial x_i^2}$. Por tanto,

$$
\begin{aligned}
vec(\boldsymbol{H})^T R(D^{\otimes 2} f) vec(\boldsymbol{H}) &= \int vec(\boldsymbol{H})^T D^{\otimes 2} f(\boldsymbol{x}) D^{\otimes 2} f(\boldsymbol{x})^T vec(\boldsymbol{H}) d\boldsymbol{x} = \\
&= \int (D^T \boldsymbol{H} Df(\boldsymbol{x}))^2 d\boldsymbol{x} = R(D^T \boldsymbol{H} Df) = \\
&= R(tr(\text{diag}(h_1^2, \ldots, h_d^2) \odot Hf))
\end{aligned}
$$

Con esto y teniendo en cuenta que $|\boldsymbol{H}|^{-\frac{1}{2}} = \left(\prod_{i=1}^d h_i^2\right)^{-\frac{1}{2}} = \prod_{i=1}^d h_i^{-1}$, se puede reformular (3.21) como

$$
AMISE\left(\hat{f}(\boldsymbol{x}; \boldsymbol{H})\right) = \frac{1}{4}\mu_2(K)^2 R(D^T \boldsymbol{H} Df) + \frac{1}{nh_1 \cdot \cdots \cdot h_d} R(K) \tag{3.24}
$$

Como en el caso univariante, obtener una forma para $\boldsymbol{H}$ que minimice el error cuadrático medio integrado asintótico, no supone una ventaja en el cálculo del mismo ya que este depende explícitamente en f que es desconocida.

(Chacón y Duong, 2018) calcula H^*_{AMISE} suponiendo distribución para f. Así, estudia el caso en el que f de se distribuye de forma normal y el caso en el que lo hace como una mixtura de normales.

3.4. Métodos de selección automática del ancho de banda

En el presente capítulo, se estudiarán los principales métodos empleados en la selección de la matriz del ancho de banda siguiendo el enfoque no paramétrico del estimador de densidades de tipo núcleo.

Como en el capítulo anterior, se presentarán las versiones multivariables de la validación cruzada, tanto sesgada como insesgada y el *plug-in*.

3.4.1. Validación cruzada

Se realiza el desarrollo de los estimadores de la matriz del ancho de banda de validación cruzada que tienen una interpretación análoga a los formulados en la sección (**??**).

Validación cruzada insesgada Como se hizo para el caso unidimensional, el objetivo es el de minimizar el error cuadrático integrado con respecto a la matriz $\boldsymbol{H}$.

Se desarrolla el error cuadrático integrado a partir de (3.12)

$$
ISE\left(\hat{f}(\cdot; \boldsymbol{H})\right) = \int \hat{f}^2(\boldsymbol{x}; \boldsymbol{H}) d\boldsymbol{x} - 2\int \hat{f}(\boldsymbol{x}; \boldsymbol{H}) f(\boldsymbol{x}) d\boldsymbol{x} + \int f^2(\boldsymbol{x}) d\boldsymbol{x}
$$

como puede observarse, el último término de la ecuación que se puede escribir como $R(f)$ no depende del ancho de banda por lo que, para obtener el ancho de banda que proporciona el menor error cuadrático integrado, podemos obviarlo. Por otra parte, el segundo término representa la

esperanza del estimador condicionada a las observaciones, es decir,

$$\int \hat{f}(\boldsymbol{x};\boldsymbol{H})f(\boldsymbol{x})d\boldsymbol{x} = \mathbb{E}\left[\hat{f}(\boldsymbol{X};\boldsymbol{H})|\boldsymbol{X}_1,\ldots,\boldsymbol{X}_n\right]$$

Se define el estimador de densidad *leave-one-out*. Se construye como el estimador de tipo núcleo pero obviando el i-ésimo elemento de la muestra,

$$\hat{f}_{-i}(\boldsymbol{x};\boldsymbol{H}) = \frac{1}{n-1}\sum_{\substack{i=1\\ i\neq j}}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)$$

Empleando el estimador de densidad *leave-one-out* se define un estimador insegado para $\mathbb{E}\left[\hat{f}(\boldsymbol{x};\boldsymbol{H})|\boldsymbol{X}_1,\ldots,\boldsymbol{X}_n\right]$

$$\frac{1}{n}\sum_{i=1}^{n}\hat{f}_{-i}(\boldsymbol{X}_i;\boldsymbol{H})$$

que, efectivamente, es un estimador insesgado de la esperanza condicionada anteriormente mencionada,

$$\begin{aligned}
\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}\hat{f}_{-i}(\boldsymbol{X}_i;\boldsymbol{H})\right] &= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\hat{f}_{-i}(\boldsymbol{X}_i;\boldsymbol{H})\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\frac{1}{n-1}\sum_{\substack{j=1\\ j\neq i}}^{n}K_{\boldsymbol{H}}(\boldsymbol{X}_i-\boldsymbol{X}_j)\right] = \\
&= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n-1}\sum_{\substack{j=1\\ j\neq i}}^{n}\mathbb{E}\left[K_{\boldsymbol{H}}(\boldsymbol{X}_i-\boldsymbol{X}_j)\right] = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n-1}(n-1)(K_{\boldsymbol{H}}*f)(\boldsymbol{X}_i) = \\
&= \frac{1}{n}\sum_{i=1}^{n}(K_{\boldsymbol{H}}*f)(\boldsymbol{X}_i) = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\hat{f}(\boldsymbol{X},\boldsymbol{H})|\boldsymbol{X}_i] = \mathbb{E}[\hat{f}(\boldsymbol{X},\boldsymbol{H})|\boldsymbol{X}_1,\ldots,\boldsymbol{X}_n]
\end{aligned}$$

Con esto, se obtiene un estimador insesgado para $ISE\left(\hat{f}(\cdot;\boldsymbol{H})\right) - f(\boldsymbol{x})$. Así, se define el criterio de validación cruzada insesgada (UCV) multivariante exactamente igual que en el caso univariante

$$UCV(\boldsymbol{H}) = \int \hat{f}^2(\boldsymbol{x};\boldsymbol{H})d\boldsymbol{x} - 2\frac{1}{n}\sum_{i=1}^{n}\hat{f}_{-i}(\boldsymbol{X}_i;\boldsymbol{H}) \tag{3.25}$$

que efectivamente es un estimador insegagado de $ISE\left(\hat{f}(\cdot;\boldsymbol{H})\right) - f(\boldsymbol{x})$. De aquí se puede obtener la matriz de ancho de banda que minimiza la función UCV

$$\boldsymbol{H}^*_{UCV} = \underset{\boldsymbol{H}\in\mathcal{F}}{\operatorname{argmin}}\ UCV(\boldsymbol{H})$$

Es necesario definir el criterio de validación cruzada insesgada de una manera más adecuada para su facilitar su cálculo en la práctica. Para ello, se desarrollan los dos términos de (3.25) por separado,

$$\begin{aligned}
\int \hat{f}^2(\boldsymbol{x};\boldsymbol{H})d\boldsymbol{x} &= \int\left(\frac{1}{n}\sum_{i=1}^{n}K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)\right)\left(\frac{1}{n}\sum_{j=1}^{n}K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_j)\right)d\boldsymbol{x} = \\
&= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\int K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_j)d\boldsymbol{x} = \\
&= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}\int K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_j)d\boldsymbol{x} + \frac{1}{n^2}\sum_{i=1}^{n}\int K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)^2d\boldsymbol{x}
\end{aligned}$$

el segundo sumatorio se puede reescribir de la siguiente manera

$$\frac{1}{n^2}\sum_{i=1}^{n}\int K_{\boldsymbol{H}}^2(\boldsymbol{x}-\boldsymbol{X}_i)dx \underset{\boldsymbol{x}=\boldsymbol{X}_i+\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z}}{=} \frac{1}{n^2}\sum_{i=1}^{n}\int K_{\boldsymbol{H}}(\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{z})^2|\boldsymbol{H}|^{\frac{1}{2}}d\boldsymbol{z} =$$
$$= n\frac{1}{n^2}|\boldsymbol{H}|^{-1}|\boldsymbol{H}|^{\frac{1}{2}}\int K^2(\boldsymbol{z})d\boldsymbol{z} = \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}R(K)$$

y el primero,

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\int K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i)K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_j)d\boldsymbol{x} \underset{\boldsymbol{x}=\boldsymbol{X}_j+\boldsymbol{z}}{=} \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\int K_{\boldsymbol{H}}(\boldsymbol{z}-\boldsymbol{X}_i+\boldsymbol{X}_j)K_{\boldsymbol{H}}(\boldsymbol{z})d\boldsymbol{z} =$$
$$= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\int K_{\boldsymbol{H}}(\boldsymbol{X}_i-\boldsymbol{X}_j-\boldsymbol{z})K_{\boldsymbol{H}}(\boldsymbol{z})d\boldsymbol{z} = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}(K_{\boldsymbol{H}}*K_{\boldsymbol{H}})(\boldsymbol{X}_i-\boldsymbol{X}_j)$$

en el segundo paso se ha tenido en cuenta que el núcleo es esféricamente simétrico, lo que implica que el núcleo escalado será elípticamente simétrico y por tanto los valores opuestos a z, $-z$, tendrán el mismo valor del núcleo al hallarse en la misma elipse que el elemento z. Por tanto, $K_{\boldsymbol{H}}(\boldsymbol{x}) = K_h(-x)$. Así,

$$\int \hat{f}^2(\boldsymbol{x};\boldsymbol{H})d\boldsymbol{x} = \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}R(K) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}(K_{\boldsymbol{H}}*K_{\boldsymbol{H}})(\boldsymbol{X}_i-\boldsymbol{X}_j)$$

Por otra parte, se puede desarrollar el segundo término de (3.25),

$$2\frac{1}{n}\sum_{i=1}^{n}\hat{f}_{-i}(\boldsymbol{X}_i;\boldsymbol{H}) = 2\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n-1}\sum_{\substack{j=1\\j\neq i}}^{n}K_{\boldsymbol{H}}(\boldsymbol{X}_i-\boldsymbol{X}_j) = 2\frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}K_{\boldsymbol{H}}(\boldsymbol{X}_i-\boldsymbol{X}_j)$$

Por tanto, el estimador de validación cruzada insesgado en un contexto multivariante puede reescribirse como,

$$\begin{aligned}
UCV(\boldsymbol{H}) &= \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}R(K) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}(K_{\boldsymbol{H}}*K_{\boldsymbol{H}})(\boldsymbol{X}_i-\boldsymbol{X}_j) - 2\frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}K_{\boldsymbol{H}}(\boldsymbol{X}_i-\boldsymbol{X}_j) = \\
&= \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}R(K) + \frac{(n-1)}{n^2(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}(K_{\boldsymbol{H}}*K_{\boldsymbol{H}})(\boldsymbol{X}_i-\boldsymbol{X}_j) - \frac{2}{n(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}K_{\boldsymbol{H}}(\boldsymbol{X}_i-\boldsymbol{X}_j) = \\
&= \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}R(K) + \frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\left\{\left(\frac{n-1}{n}\right)(K_{\boldsymbol{H}}*K_{\boldsymbol{H}}) - 2K_{\boldsymbol{H}}\right\}(\boldsymbol{X}_i-\boldsymbol{X}_j)
\end{aligned} \tag{3.26}$$

que es completamente análoga al caso univariante. Igual que en ese caso, es habitual considerar $1-\frac{1}{n}\sim 1$. Aunque se pierda la insesgadez del estimador, no se pierde la insesgadez asintótica y el estimador tiene una forma más sencilla

$$UCV(\boldsymbol{H}) = \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}R(K) + \frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1\\j\neq i}}^{n}\{(K_{\boldsymbol{H}}*K_{\boldsymbol{H}}) - 2K_{\boldsymbol{H}}\}(\boldsymbol{X}_i-\boldsymbol{X}_j) \tag{3.27}$$

Como menciona (Chacón y Duong, 2018) el problema al minimizar la función $UCV(\boldsymbol{H})$ resulta en que puede presentar varios mínimos locales y la tendencia a suavizar en exceso. Estos dos problemas generalmente se abordan restringiendo anchos de banda pequeños y con anchos de banda iniciales grandes.

Validación cruzada sesgada Como en el caso univariante, para desarrollar la validación cruzada sesgada, se partirá de la tercera expresión en (3.21) del error cuadrático medio integrado asintótico. Siguiendo a (Sain y col., 1994)..

$$AMISE\left(\hat{f}(\cdot;\boldsymbol{H})\right) = \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}R(K) + \frac{1}{4}\mu_2^2(K)vec(\boldsymbol{H})^T R(D^{\otimes 2}f)vec(\boldsymbol{H})$$

donde hay que estimar la expresión $R(D^{\otimes 2}f)$ que es desconocida.

Definimos $R(D^{\otimes 2}f) = \Psi_4$ y definimos $\boldsymbol{r}_1, r_2 \in \mathbb{N}^d$ donde $|\boldsymbol{r}_i| = \sum_{j=1}^d r_{i_j} = 2$ $i = 1, 2$. Así, cada elemento de Ψ_4 será de la forma,

$$\psi_{\boldsymbol{r}_1,\boldsymbol{r}_2} = \int \frac{\partial^2 f(\boldsymbol{x})}{\partial^2 x_1^{r_{1_1}} \cdots \partial x_1^{r_{1_d}}} \frac{\partial f(\boldsymbol{x})}{\partial x_1^{r_{2_1}} \cdots \partial x_1^{r_{2_d}}} d\boldsymbol{x} = \int f^{(\boldsymbol{r}_1)}(\boldsymbol{x}) f^{(\boldsymbol{r}_2)}(\boldsymbol{x}) d\boldsymbol{x}$$

con $|\boldsymbol{r}_1| = |\boldsymbol{r}_2| = 2$. Siguiendo el mismo desarrollo de (2.4.2),

$$\psi_{\boldsymbol{r}_1,\boldsymbol{r}_2} = \int f^{(\boldsymbol{r}_1)}(\boldsymbol{x}) f^{(\boldsymbol{r}_2)}(\boldsymbol{x}) d\boldsymbol{x} = \int f^{(\boldsymbol{r})}(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x} = \mathbb{E}\left[f^{(\boldsymbol{r})}(\boldsymbol{X})\right] = \psi_{\boldsymbol{r}}$$

con $|\boldsymbol{r}| = 4$. En la siguiente sección generalizaremos este resultado como lo hicimos en el capitulo anterior para desarrollar el método de *plug-in* multivariante. Así Ψ_4 se puede expresar de la siguiente manera.

$$\Psi_4 = \begin{pmatrix} \psi_{(4,0,\ldots,0)} & \cdots & \psi_{(2,0,\ldots,2)} \\ \vdots & \ddots & \vdots \\ \psi_{(2,0,\ldots,2)} & \cdots & \psi_{(0,0,\ldots,4)} \end{pmatrix}$$

por tanto, tenemos que estimar $\psi_{\boldsymbol{r}}$ con $|\boldsymbol{r}| = 4$.

Teniendo en cuenta esto, y que los elementos de Ψ_4 se comportan de la misma forma que lo hacen las funciones univariantes, podemos dar la generalización multivariante de los estimadores $\widehat{R(f'')}$ y $\widehat{\widehat{R(f'')}}$.

El primer estimador, como en (2.26) se puede escribir como

$$\hat{\psi}_{\boldsymbol{r}} = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j\neq i}}^{n} (K_{\boldsymbol{H}}^{(\boldsymbol{r})} * K_{\boldsymbol{H}})(\boldsymbol{X}_i - \boldsymbol{X}_j) \tag{3.28}$$

$$BCV_1(h) = \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}R(K) + \frac{1}{4}\mu_2(K)^2 vec(\boldsymbol{H})^T \hat{\Psi}_4 vec(\boldsymbol{H}) \tag{3.29}$$

Mientras que la generalización multivariante de el estimador $\widehat{\widehat{R(f'')}}$ presentado en (2.28) es

$$\hat{\hat{\psi}}_{\boldsymbol{r}} = \frac{1}{n}\sum_{i=1}^{n} \hat{f}_{-i}^{(\boldsymbol{r})}(\boldsymbol{X}_i, \boldsymbol{H}) = \frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n} K_{\boldsymbol{H}}^{(\boldsymbol{r})}(\boldsymbol{X}_i - \boldsymbol{X}_j) \tag{3.30}$$

$$BCV_2(h) = \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} R(K) + \frac{1}{4}\mu_2(K)^2 vec(\boldsymbol{H})^T \hat{\Psi}_4 vec(\boldsymbol{H}) \tag{3.31}$$

Y minimizando la función adecuada para cada estimador, se obtiene el ancho de banda óptimo

$$h_{BCV}^{*} = \underset{h>0}{\operatorname{argmin}}\ BCV_i(h), \quad i = 1, 2$$

Sin embargo, y como mencionan (Chacón y Duong, 2018) "Investigaciones posteriores han destacado la importancia de utilizar un ancho de banda piloto diferente para estimar la matriz de curvatura $\boldsymbol{R}\left(D^{\otimes 2} f\right)$, a pesar de una carga computacional adicional, por lo que el BCV no ha atraído suficiente interés desde su introducción".

3.4.2. Plug-in

Siguiendo los pasos de (Chacón y Duong, 2010; Wand y Jones, 1994) presentamos la generalización multivariante del método desarrollado en la sección 2.4.2.

Nos centraremos en el estimador PI para matrices sin restricciones presentado por (Chacón y Duong, 2010). Sin embargo, existen otras alternativas para abordar el problema de selección de la matriz del ancho de banda óptimo mediante el método *plug-in* multivariante tal y como señala (Chacón y Duong, 2010) en la Sección 3.

Como en el caso univariante objetivo es el de estimar $vec\left(\boldsymbol{R}(D^{\otimes 2} f)\right)$ en la expresión de 3.21. En este caso es más tratable emplear la tercera expresión para AMISE. Es por esto que llamaremos a $vec\left(\boldsymbol{R}(D^{\otimes 2} f)\right) = \Psi_4$.

Así podemos encontrar la matriz de ancho de banda $\boldsymbol{H}_{PI}$ minimizando la expresión de 3.21 donde sustituimos Ψ_4 por su una estimación $\hat{\Psi}_4$.

Como en el apartado anterior, podemos escribir $R(D^{\otimes s} f)$ y definimos $\boldsymbol{r}_1, \boldsymbol{r}_2 \in \mathbb{N}^d$ donde $|\boldsymbol{r}_i| = \sum_{j=1}^{d} r_{i_j} = s$ $i = 1, 2$. Así, cada elemento será de la forma,

$$\int \frac{\partial^s f(\boldsymbol{x})}{\partial x_1^{r_{1_1}} \cdots \partial x_1^{r_{1_d}}} \frac{\partial^s f(\boldsymbol{x})}{\partial x_1^{r_{2_1}} \cdots \partial x_1^{r_{2_d}}} d\boldsymbol{x} = \int f^{(\boldsymbol{r}_1)}(\boldsymbol{x}) f^{(\boldsymbol{r}_2)}(\boldsymbol{x}) d\boldsymbol{x}$$

con $|\boldsymbol{r}_1| = |\boldsymbol{r}_2| = s$. Siguiendo el mismo desarrollo de (2.4.2) y suponiendo las condiciones necesarias de regularidad de f,

$$\int f^{(\boldsymbol{r}_1)}(\boldsymbol{x}) f^{(\boldsymbol{r}_2)}(\boldsymbol{x}) d\boldsymbol{x} = (-1)^{|\boldsymbol{r}|} \int f^{(\boldsymbol{r})}(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x}$$

con $|\boldsymbol{r}| = 2s$.

Con todo esto y como señalan (Chacón y Duong, 2010), definiendo

$$\Psi_r = \int D^{\otimes r} f(\boldsymbol{x}) f(\boldsymbol{x}) d\boldsymbol{x} \tag{3.32}$$

se comprueba que $vec(\boldsymbol{R}(D^{\otimes s} f)) = (-1)^s \Psi_{2s}$. Además por definición, $\Psi_r = \mathbb{E}\left[D^{\otimes r} f(\boldsymbol{X})\right]$ por lo que el estimador natural para Ψ_r es,

$$\hat{\Psi}_r(\boldsymbol{G}) = \frac{1}{n}\sum_{i=1}^{n} D^{\otimes r} \tilde{f}(\boldsymbol{X}_i, \boldsymbol{G}) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} D^{\otimes r} L_{\boldsymbol{G}}(\boldsymbol{X}_i - X_j) \tag{3.33}$$

donde $\tilde{f}(\boldsymbol{X}_i, \boldsymbol{G})$ es un estimador de densidad de f con un kernel piloto L y una matriz de ancho de banda piloto $\boldsymbol{G}$ que pueden diferir de K y $\boldsymbol{H}$ respectivamente.

Impondremos las siguientes condiciones sobre el kernel y el ancho de banda pilotos.

- K es una función de densidad esféricamente simétrica tal que $\int \boldsymbol{z}^{\otimes 2} L(\boldsymbol{z}) = \mu_2(L) vec(\boldsymbol{I}_d)$ con todos los elementos de $D^{\otimes j} L$ acotados, continuos y cuadrado integrables para $0 \leq j \leq r$.
- Todos los elementos de $D^{\otimes j} f$ acotados, continuos y cuadrado integrables para $0 \leq j \leq r+2$
- La matriz de ancho de banda piloto $\boldsymbol{G}_n$ es una secuencia tal que

$$vec(\boldsymbol{G}_n) \to 0, \to \infty \quad \text{cuando} \quad n \to \infty$$

Como en (2.4.2) estamos interesados en minimizar el error cuadrático medio del estimador. Como ya hemos realizado anteriormente, el error cuadrático medio puede descomponerse como la suma del sesgo al cuadrado y la varianza del estimador, $MSE(\boldsymbol{G}) = \mathbb{E}\left[\|\hat{\Psi}_r(\boldsymbol{G}) - \Psi_r\|^2\right] = sesgo^2(\boldsymbol{G}) + V(\boldsymbol{G})$ donde $sesgo^2(\boldsymbol{G}) = \|\mathbb{E}\left[\hat{\Psi}_r(\boldsymbol{G})\right] - \Psi_r\|$ y $V(\boldsymbol{G}) = trVar(\hat{\Psi}_r(\boldsymbol{G}))$. Sin embargo, en este caso solo calcularemos el sesgo al cuadrado porque, al igual que en (2.4.2) el término principal de $\boldsymbol{G}$ en la varianza es de orden menor al del sesgo al cuadrado por lo que se puede obviar al minimizar el término de error cuadrático medio asintótico. Para una demostración completa de este punto véase el Teorema 1 en (Chacón y Duong, 2010).

Podemos reescribir $\hat{\Psi}_r(\boldsymbol{G})$ como,

$$\hat{\Psi}_r(\boldsymbol{G}) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} D^{\otimes r} L_{\boldsymbol{G}}(\boldsymbol{X}_i - \boldsymbol{X}_j) = \frac{1}{n} D^{\otimes r} L(0) + \frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n} D^{\otimes r} L_{\boldsymbol{G}}(\boldsymbol{X}_i - \boldsymbol{X}_j) \tag{3.34}$$

entonces,

$$\mathbb{E}\left[\hat{\Psi}_r\right] = \frac{1}{n} D^{\otimes r} L_{\boldsymbol{G}}(0) + \frac{1}{n^2}\mathbb{E}\Big[\sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}} D^{\otimes r} L_{\boldsymbol{G}}(X_i - X_j)\Big] =$$

$$= \frac{1}{n} D^{\otimes r} L_{\boldsymbol{G}}(0) + \frac{1}{n^2} n(n-1)\mathbb{E}\left[D^{\otimes r} L_{\boldsymbol{G}}(X_1 - X_2)\right]$$

calculando, la esperanza del segundo sumando,

$$\mathbb{E}\big[D^{\otimes r}L_{\boldsymbol{G}}(X_1-X_2)\big]=\int\int D^{\otimes r}L_{\boldsymbol{G}}(\boldsymbol{x}-\boldsymbol{y})f(\boldsymbol{x})f(\boldsymbol{y})d\boldsymbol{x}d\boldsymbol{y}=$$
$$=\int f(\boldsymbol{x})\int D^{\otimes r}L_{\boldsymbol{G}}(\boldsymbol{x}-\boldsymbol{y})f(\boldsymbol{y})d\boldsymbol{y}d\boldsymbol{x}=\int\int L_{\boldsymbol{G}}(\boldsymbol{x}-\boldsymbol{y})f(\boldsymbol{x})D^{\otimes r}f(\boldsymbol{y})d\boldsymbol{y}d\boldsymbol{x}\underset{\substack{\boldsymbol{x}=\boldsymbol{v}+\boldsymbol{G}^{\frac{1}{2}}\boldsymbol{u}\\ \boldsymbol{y}=\boldsymbol{v}}}{=}$$
$$=\int\int L_{\boldsymbol{G}}(\boldsymbol{G}^{\frac{1}{2}}\boldsymbol{u})f(\boldsymbol{v}+\boldsymbol{G}^{\frac{1}{2}}\boldsymbol{u})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}d\boldsymbol{u}=\int\int L(\boldsymbol{u})f(\boldsymbol{v}+\boldsymbol{G}^{\frac{1}{2}}\boldsymbol{u})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}d\boldsymbol{u}=$$
$$=\int\int L(\boldsymbol{u})f(\boldsymbol{v}+\boldsymbol{G}^{\frac{1}{2}}\boldsymbol{u})^TD^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}d\boldsymbol{u}=$$
$$=\int\int L(\boldsymbol{u})\Big(f(\boldsymbol{v})+\boldsymbol{u}^T\boldsymbol{G}^{1/2}Df(\boldsymbol{v})+\frac{1}{2}(\boldsymbol{u}^T\boldsymbol{G}^{1/2})^{\otimes 2}D^{\otimes r}f(\boldsymbol{v})+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r}\Big)D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}d\boldsymbol{u}=$$
$$=\int\int L(\boldsymbol{u})f(\boldsymbol{v})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}d\boldsymbol{u}+\int\int L(\boldsymbol{u})\boldsymbol{u}^T\boldsymbol{G}^{1/2}Df(\boldsymbol{v})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}d\boldsymbol{u}+$$
$$\int\int L(\boldsymbol{u})\frac{1}{2}(\boldsymbol{u}^T\boldsymbol{G}^{1/2})^{\otimes 2}D^{\otimes 2}f(\boldsymbol{v})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}d\boldsymbol{u}+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r}=$$
$$=\int f(\boldsymbol{v})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}\int L(\boldsymbol{u})d\boldsymbol{u}+\int\boldsymbol{G}^{1/2}Df(\boldsymbol{v})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}\int L(\boldsymbol{u})\boldsymbol{u}^Td\boldsymbol{u}+$$
$$\int L(\boldsymbol{u})\frac{1}{2}(\boldsymbol{u}^T\boldsymbol{G}^{1/2})^{\otimes 2}d\boldsymbol{u}\int D^{\otimes 2}f(\boldsymbol{v})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r}=$$
$$=\int f(\boldsymbol{v})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}+\frac{\mu_2(L)}{2}\int vec(\boldsymbol{I}_d)(\boldsymbol{G}^{1/2})^{\otimes 2}D^{\otimes 2}f(\boldsymbol{v})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r}=$$
$$=\int f(\boldsymbol{v})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}+\frac{\mu_2(L)}{2}\int vec(\boldsymbol{I}_d)^T(\boldsymbol{G}^{1/2})^{\otimes 2}D^{\otimes 2}f(\boldsymbol{v})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r}=$$
$$=\int f(\boldsymbol{v})D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}+\frac{\mu_2(L)}{2}\left(\int D^{\otimes r}f(\boldsymbol{v})D^{\otimes 2}f(\boldsymbol{v})^Td\boldsymbol{v}\right)vec(\boldsymbol{G})+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r}=$$
$$=\Psi_r+\frac{\mu_2(L)}{2}(vec(\boldsymbol{G})^T\otimes\mathbf{1}_{d^r})vec\left(\int D^{\otimes r}f(\boldsymbol{v})D^{\otimes 2}f(\boldsymbol{v})^Td\boldsymbol{v}\right)+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r}=$$
$$=\Psi_r+\frac{\mu_2(L)}{2}(vec(\boldsymbol{G})^T\otimes\mathbf{1}_{d^r})vec\left(\int D^{\otimes 2}f(\boldsymbol{v})\otimes D^{\otimes r}f(\boldsymbol{v})d\boldsymbol{v}\right)+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r}=$$
$$=\Psi_r+\frac{\mu_2(L)}{2}(vec(\boldsymbol{G})^T\otimes\mathbf{1}_{d^r})vec\left(\int D^{\otimes(r+2)}f(\boldsymbol{v})f(\boldsymbol{v})d\boldsymbol{v}\right)+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r}=$$
$$=\Psi_r+\frac{\mu_2(L)}{2}(vec(\boldsymbol{G})^T\otimes\mathbf{1}_{d^r})\Psi_{r+2}+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r}$$

Así, podemos desarrollar la esperanza de $\hat{\Psi}_r$ de la siguiente manera,

$$\mathbb{E}\left[\hat{\Psi}_r\right]=\frac{1}{n}D^{\otimes r}L_{\boldsymbol{G}}(0)+\frac{(n-1)}{n}\mathbb{E}\left[D^{\otimes r}L_{\boldsymbol{G}}(X_1-X_2)\right]=$$
$$=\frac{1}{n}D^{\otimes r}L_{\boldsymbol{G}}(0)+\frac{(n-1)}{n}\left(\Psi_r+\frac{\mu_2(L)}{2}(vec(\boldsymbol{G})^T\otimes\mathbf{1}_{d^r})\Psi_{r+2}+(o(tr(\boldsymbol{G}))\mathbf{1}_{d^r})\right)=$$
$$=\frac{1}{n}D^{\otimes r}L_{\boldsymbol{G}}(0)+\Psi_r+\frac{\mu_2(L)}{2}(vec(\boldsymbol{G})^T\otimes\mathbf{1}_{d^r})\Psi_{r+2}+(o(\frac{1}{n})+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r})=$$

por tanto, el sesgo de $\hat{\Psi}_r$

$$\text{sesgo}(\hat{\Psi}_r)=\left\|\frac{1}{n}D^{\otimes r}L_{\boldsymbol{G}}(0)+\frac{\mu_2(L)}{2}(vec(\boldsymbol{G})^T\otimes\mathbf{1}_{d^r})\Psi_{r+2}+(o(\frac{1}{n})+o(tr(\boldsymbol{G}))\mathbf{1}_{d^r})\right\| \tag{3.35}$$

y el sesgo asintótico, podemos escribirlo como,

$$\text{sesgo}_{\text{A}}(\hat{\Psi}_r)=\left\|\frac{1}{n}D^{\otimes r}L_{\boldsymbol{G}}(0)+\frac{\mu_2(L)}{2}(vec(\boldsymbol{G})^T\otimes\mathbf{1}_{d^r})\Psi_{r+2}\right\| \tag{3.36}$$

Como se ha mencionado con anterioridad, el término de la varianza dependiente de $\boldsymbol{G}$ es de orden menor que el del sesgo al cuadrado por lo que la matriz de ancho de banda piloto óptima se puede hallar minimizando el cuadrado del sesgo asintótico.

$$\boldsymbol{G}_{AMSE,r} = \underset{G\in\mathcal{F}}{\operatorname{argmin}}\ \operatorname{sesgo_A}^2(\boldsymbol{G}) \tag{3.37}$$

Por lo tanto, el problema que resulta de este cálculo, es el siguiente: Si se quiere estimar Ψ_r mediante $\Psi_r(\boldsymbol{G}_{AMSE,r})$ resulta necesario estimar Ψ_{r+2}. La solución a este problema, al igual que en el caso univariante, se basa en insertar *to plug-in* una estimación simple de ψ_r en un punto.

De esta forma, el objetivo será el de minimizar $PI(\boldsymbol{H},\boldsymbol{G})$ para $\boldsymbol{H}$ donde,

$$PI(\boldsymbol{H},\boldsymbol{G}) = \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}R(K) + \frac{1}{4}\mu_2^2(K)\hat{\Psi}_4(G)^T(vec(\boldsymbol{H}))^{\otimes 2} \tag{3.38}$$

Para la etapa inicial, es necesaria una referencia normal del estimador de Ψ_r, $\hat{\Psi}_r^{NR}$. Como muestran (Chacón y Duong, 2010),

$$\Psi_r^{NR} = \frac{(-1)^{\frac{r}{2}}r!}{2^{r+d}\frac{r}{2}!\pi^{\frac{d}{2}}}|\Sigma|^{-\frac{1}{2}}\mathcal{S}_{d,r}(vec\Sigma^{-1})^{\otimes(\frac{r}{2})}$$

donde $\mathcal{S}_{d,r}$ se refiere a la matriz simetrizadora d-variante de orden r. En la sección 3.3 de (Chacón y Duong, 2010) se pueden encontrar más referencias sobre esta matriz, aunque no es necesaria tenerla en cuenta ya que como demuestran en ese mismo artículo, en dicho caso la matriz de ancho de banda piloto óptima toma la forma

$$\boldsymbol{G}_{AMSE,r}^{NR} = \left(\frac{2}{r+d}\right)^{\frac{2}{r+d+2}} 2\Sigma n^{-\frac{2}{r+d+2}} \tag{3.39}$$

y cuya estimación se puede dar sustituyendo Σ por la matriz de covarianzas de la muestra $\boldsymbol{S}$.

$$\hat{\boldsymbol{G}}_{AMSE,r}^{NR} = \left(\frac{2}{r+d}\right)^{\frac{2}{r+d+2}} 2\boldsymbol{S} n^{-\frac{2}{r+d+2}} \tag{3.40}$$

En este punto podemos dar un algoritmo para el ancho de banda óptimo mediante el estimador *plug-in* de l-etapas.

Obtenemos $\hat{\boldsymbol{G}}_{AMSE,2l+2}^{NR}$. Con esto, podemos estimar

$$\hat{\Psi}_{2l+2}(\hat{\boldsymbol{G}}_{AMSE,2l+2}^{NR})$$

Para $j = 2l+2, 2l, \ldots, 6$, Introducimos en $\hat{\Psi}_j$ en $\operatorname{sesgo_A}^2(\boldsymbol{G})$ y con (3.37) minimizamos numéricamente para obtener $\hat{\boldsymbol{G}}_{AMSE,j-2}$.Con esto, podemos estimar

$$\hat{\Psi}_{j-2}(\hat{\boldsymbol{G}}_{AMSE,j-2})$$

Repetimos este proceso hasta obtener $\hat{\Psi}_4$.

Una vez que tenemos $\hat{\Psi}_4$, lo introducimos en 3.38 de forma que,

$$\boldsymbol{H}_{PI,l} = \underset{G\in\mathcal{F}}{\operatorname{argmin}} \left\{ \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} R(K) + \frac{1}{4}\mu_2^2(K)\hat{\Psi}_4(\hat{\boldsymbol{G}}_{AMSE,4})^T (vec(\boldsymbol{H}))^{\otimes 2} \right\} \tag{3.41}$$

***Plug in* de 2 etapas: *'Direct plug-in'*.** Obtenemos $\hat{\boldsymbol{G}}^{NR}_{AMSE,6} = \left(\frac{2}{6+d}\right)^{\frac{2}{6+d+2}} 2\boldsymbol{S}n^{-\frac{2}{6+d+2}}$. Con esto, podemos estimar

$$\hat{\Psi}_6(\hat{\boldsymbol{G}}^{NR}_{AMSE,6})$$

Ahora, introducimos en $\hat{\Psi}_6(\hat{\boldsymbol{G}}^{NR}_{AMSE,6})$ en $\text{sesgo}_{\text{A}}{}^2(\boldsymbol{G})$ y con (3.37) minimizamos numéricamente para obtener $\hat{\boldsymbol{G}}_{AMSE,4}$.Con esto, podemos estimar

$$\hat{\Psi}_4(\hat{\boldsymbol{G}}_{AMSE,4})$$

Una vez que tenemos $\hat{\Psi}_4$, lo introducimos en 3.38 de forma que,

$$\boldsymbol{H}_{PI,2} = \underset{G\in\mathcal{F}}{\operatorname{argmin}} \left\{ \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}} R(K) + \frac{1}{4}\mu_2^2(K)\hat{\Psi}_4(\hat{\boldsymbol{G}}_{AMSE,4})^T (vec(\boldsymbol{H}))^{\otimes 2} \right\} \tag{3.42}$$

3.5. Comparación empírica de los estimadores de la matriz de ancho de banda

El cálculo de derivadas de ordenes elevados resulta complicada en la práctica. En el caso de la distribución normal, existe una fórmula cerrada para el cálculo de $D^{\otimes r}\phi_{\boldsymbol{H}}$ pero esta tiene una aplicación complicada. Por esto, (Chacón y Duong, 2014) muestran un procedimiento recursivo para calcular las derivadas de la función de densidad normal de forma eficiente. La programación de este algoritmo tampoco resulta inmediato para su aplicación.

En este apartado, con el fin de comprobar empíricamente el comportamiento de los estimadores del ancho de banda, hemos simulado una muestra de tamaño 100 a partir de una mixtura de tres densidades normales, X, donde conocemos su función de densidad real,$f_X(x)$.

$$X \sim P(Y=0)N\left(\begin{pmatrix}3\\3\end{pmatrix}, \begin{pmatrix}2&1\\1&3\end{pmatrix}\right) + P(Y=1)N\left(\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}2&0\\0&1\end{pmatrix}\right) + P(Y=2)N\left(\begin{pmatrix}3\\2\end{pmatrix}, \begin{pmatrix}1&0\\0&3\end{pmatrix}\right)$$

donde $Y \sim Bin(2, 0{,}24)$

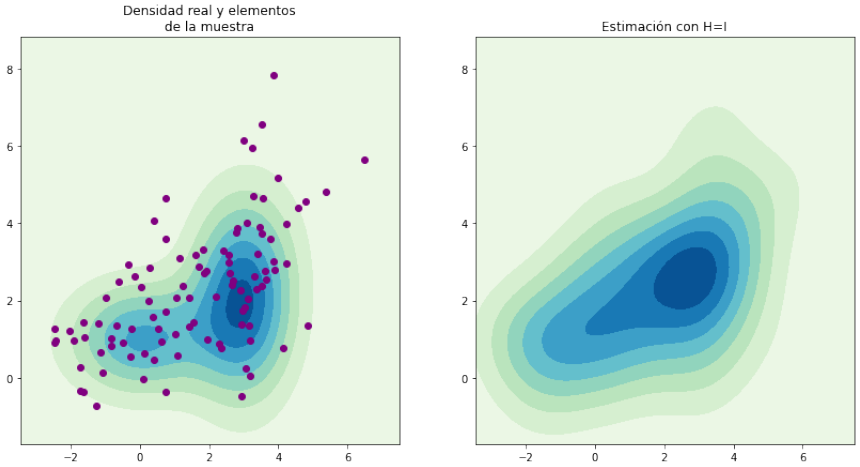


Figura 14: A la izquierda podemos ver las curvas de nivel de la función de densidad real generada de X. Además, se muestran sobre las curvas de nivel los elementos de la muestra generada aleatoriamente. A la derecha se puede visualizar una primera estimación de la densidad con el núcleo normal y la matriz de ancho de banda $\boldsymbol{I}_2$

Calculamos los anchos de banda óptimos para los métodos de validación cruzada insesgada y sesgada respectivamente, y para el estimador *plug-in* de 1 etapa. Como hemos mencionado, todos los detalles de cálculo están plasmados en el tercer Anexo.

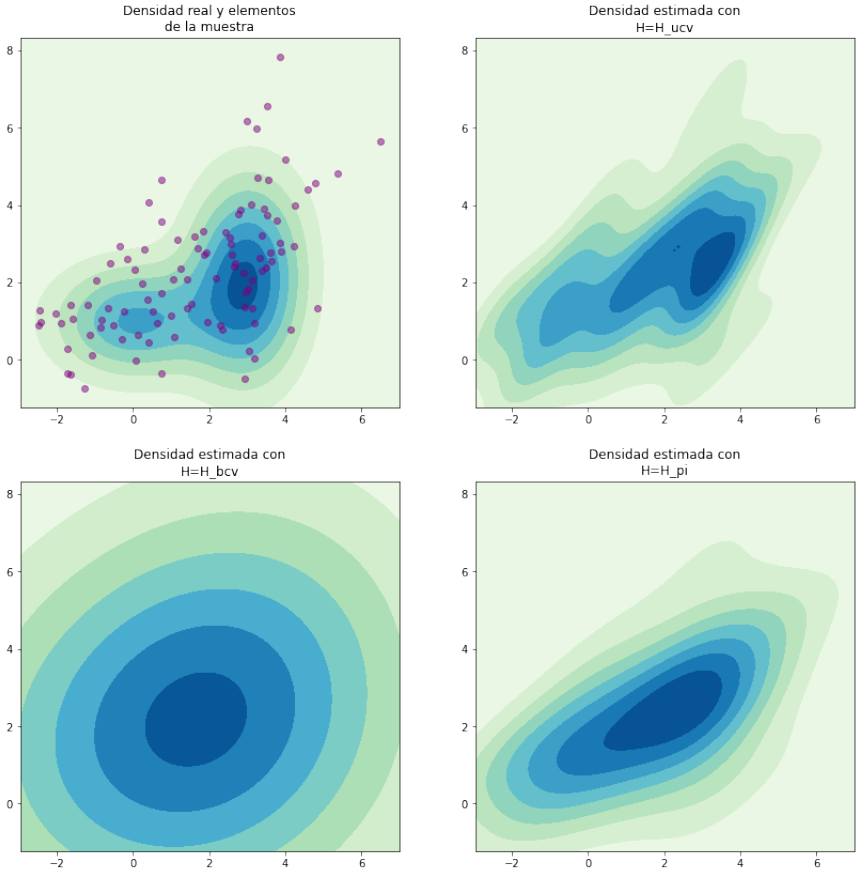


Figura 15: Podemos ver la estimación con los distintos anchos de banda óptimos

Como en el caso univariante, aunque en este caso no tenemos estadísticas sobre la distribución de las matrices de ancho de banda óptimas con cada uno de los modelos, podemos intuir un fenómeno muy similar al que teníamos en el caso unidimensional. Vemos cómo la matriz de ancho de banda óptima que nos proporciona el método BCV suaviza la función de densidad en exceso. En el otro extremo tenemos la que nos proporciona UCV que, como podemos observar sobreajusta la función de densidad estimada con la muestra. Sin embargo el que parece ofrece mejores prestaciones, además de su ventaja en eficiencia computacional, es el estimador de *plug-in*.

4. Estimación de derivadas

En este capítulo abordaremos la estimación de derivadas de una función de densidad multivariante, $D^{\otimes r} f(\boldsymbol{x})$, que como se ha mencionado antes, utilizando la formulación mediante el producto de Kronecker, es un vector de dimensión d^r conteniendo todas las derivadas parciales de orden r de $f(\boldsymbol{x})$.

Como menciona (Chacón y Duong, 2018), la adaptación de adaptación multivariante de la estimación inicialmente propuesta por (Bhattacharya, 1967), fue introducida por (Chacón y col., 2011) como

$$\widehat{D^{\otimes r} f}(\boldsymbol{x}; \boldsymbol{H}) = D^{\otimes r} \hat{f}(\boldsymbol{x}; \boldsymbol{H}) = \frac{1}{n} \sum_{i=1}^{n} D^{\otimes r} K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{X}_i)$$

donde, tenemos que tener en cuenta que

$$D^{\otimes r} K_{\boldsymbol{H}}(\boldsymbol{x}) = |\boldsymbol{H}|^{-\frac{1}{2}} (\boldsymbol{H}^{-\frac{1}{2}})^{\otimes r} D^{\otimes r} K(\boldsymbol{H}^{-\frac{1}{2}} \boldsymbol{x})$$

La estimación de derivadas de funciones de densidad resulta importante, como hemos podido comprobar en capítulos anteriores, para la obtención de anchos de banda o matrices de ancho de banda en un entorno multivariante. Sin embargo, existen multitud de aplicaciones en las que estimar las derivadas de funciones de densidad es igual de importante.

Si bien en lo que respecta al estudio del estimador de tipo núcleo el orden de derivadas necesario suele ser elevado, en el caso de emplear las derivadas para un análisis exploratorio o en ciertas aplicaciones, el gradiente o la matriz Hessiana suelen tener una relevancia especial. Por ejemplo, en el próximo capítulo puede verse el algoritmo *mean shift* que emplea el gradiente para aplicar un algoritmo de ascenso por gradiente.

Estimación del gradiente La estimación de gradientes se realiza con la siguiente fórmula,

$$D\hat{f}(\boldsymbol{x}; \boldsymbol{H}) = \frac{1}{n} \sum_{i=1}^{n} DK_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{X}_i) = \frac{1}{n} |\boldsymbol{H}|^{-\frac{1}{2}} (\boldsymbol{H}^{-\frac{1}{2}}) \sum_{i=1}^{n} DK(\boldsymbol{H}^{-\frac{1}{2}} (\boldsymbol{x} - \boldsymbol{X}_i)) \tag{4.1}$$

Evidentemente el gradiente de una función de densidad no es una función de densidad ya que el gradiente puede ser tanto positivo como negativo. Es por esto que puede emplearse como herramienta exploratoria. Así se pueden visualizar los contornos positivos y negativos por separado.

Empleando la notación que hemos expuesto anteriormente, $f_+^{(\boldsymbol{r})}(\boldsymbol{x}) = f^{(\boldsymbol{r})}(\boldsymbol{x}) \mathbf{1}_{\left\{f^{(\boldsymbol{r})}(\boldsymbol{x}) \geq 0\right\}}(\boldsymbol{x})$ y $f_-^{(\boldsymbol{r})}(\boldsymbol{x}) = f^{(\boldsymbol{r})}(\boldsymbol{x}) \mathbf{1}_{\left\{f^{(\boldsymbol{r})}(\boldsymbol{x}) < 0\right\}}(\boldsymbol{x})$, serían los contornos positivos y negativos respectivamente. En este caso $|\boldsymbol{r}| = \sum_{i=1}^{d} r_i = 1$, es decir $\boldsymbol{r}$ tiene todos sus componentes con valor 0 salvo el de la dirección en la que derivamos que sería 1.

Estimación de la matriz Hessiana

$$D^{\otimes 2}\hat{f}(\boldsymbol{x};\boldsymbol{H}) = \frac{1}{n}\sum_{i=1}^{n} D^{\otimes 2}K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i) = \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}(\boldsymbol{H}^{-\frac{1}{2}})^{\otimes 2}\sum_{i=1}^{n} D^{\otimes 2}K(\boldsymbol{H}^{-\frac{1}{2}}(\boldsymbol{x}-\boldsymbol{X}_i)) \tag{4.2}$$

La estimación de la matriz Hessiana es importante también desde el punto de vista del análisis exploratorio. Como dicen (Chacón y Duong, 2018), es importante para la identificación de regiones con muchos datos que se corresponden con modas locales de las funciones de densidad.

Estimación de derivadas de orden arbitrario

$$D^{\otimes r}\hat{f}(\boldsymbol{x};\boldsymbol{H}) = \frac{1}{n}\sum_{i=1}^{n} D^{\otimes r}K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{X}_i) = \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}(\boldsymbol{H}^{-\frac{1}{2}})^{\otimes r}\sum_{i=1}^{n} D^{\otimes r}K(\boldsymbol{H}^{-\frac{1}{2}}(\boldsymbol{x}-\boldsymbol{X}_i)) \tag{4.3}$$

Los resultados más relevantes sobre las propiedades asintóticas del estimador de derivadas de densidades de tipo kernel se pueden encontrar en (Chacón y Duong, 2018; Chacón y col., 2011). En (Chacón y col., 2011) demuestran que las propiedades sobre el error medio integrado asintótico (AMISE) para un Kernel arbitrario y también para el caso en el que K es el núcleo normal.

Dada la complejidad de los cálculos en este apartado, mostramos los resultados que conciernen a la estimación de derivadas de densidades que se mencionan en las dos referencias a las que hacemos alusión en este capítulo. Todas las demostraciones detalladas se pueden encontrar en las mismas.

En primer lugar se exploran la esperanza y la varianza del estimador,

$$\begin{aligned}
\mathbb{E}\left[D^{\otimes r}\hat{f}(\boldsymbol{x},\boldsymbol{H})\right] &= (K_{\boldsymbol{H}} * D^{\otimes r}f)(\boldsymbol{x}) \\
Var\left(D^{\otimes r}\hat{f}(\boldsymbol{x},\boldsymbol{H})\right) &= \frac{1}{n}\Bigg(\Big(((D^{\otimes r}K_{\boldsymbol{H}})(D^{\otimes r}K_{\boldsymbol{H}})^T) * f\Big)(\boldsymbol{x}) \\
&\qquad - (K_{\boldsymbol{H}} * D^{\otimes r}f)(\boldsymbol{x})(K_{\boldsymbol{H}} * D^{\otimes r}f)(\boldsymbol{x})^T\Bigg)
\end{aligned} \tag{4.4}$$

con esto, se obtiene

$$\begin{aligned}
MSE\left(D^{\otimes r}\hat{f}(\cdot,\boldsymbol{H})\right) = \frac{1}{n}\Big(\big(\|D^{\otimes r}K_{\boldsymbol{H}})\|^2 * f\big)(\boldsymbol{x}) - \|K_{\boldsymbol{H}} * D^{\otimes r}f)(\boldsymbol{x})\|^2\Big) + \\
+ \|(K_{\boldsymbol{H}} * D^{\otimes r}f)(\boldsymbol{x}) - D^{\otimes r}f)(\boldsymbol{x})\|^2
\end{aligned} \tag{4.5}$$

que integrando sobre $\boldsymbol{x}$,

$$\begin{aligned}
MISE\left\{D^{\otimes r}\hat{f}(\cdot,\boldsymbol{H})\right\} = tr\boldsymbol{R}(D^{\otimes r}f) + \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}tr((\boldsymbol{H}^{-1})^{\otimes r}\boldsymbol{R}(D^{\otimes r}K)) + \\
+ (1-\frac{1}{n})tr\boldsymbol{R}_{K*K,\boldsymbol{H},r}(f) - 2tr\boldsymbol{R}_{K,\boldsymbol{H},r}(f)
\end{aligned} \tag{4.6}$$

donde $\boldsymbol{R}_{K*K,\boldsymbol{H},r}(f) = \int K_{\boldsymbol{H}} * D^{otimesr}(\boldsymbol{x})K_{\boldsymbol{H}} * D^{otimesr}(\boldsymbol{x})^T d\boldsymbol{x}$.

En este caso, también se puede realizar un análisis asintótico de forma que bajo ciertas condiciones podamos obtener un ancho de banda que minimice el error cuadrático medio asintótico. De esta forma, (Chacón y Duong, 2018) muestran los resultados para la realización del análisis asintótico,

$$
\begin{aligned}
sesgo^2\left(D^{\otimes r}\hat{f}(\cdot,\boldsymbol{H})\right) &= \frac{1}{4}\mu_2^2(K)tr\left((\boldsymbol{I}_{d^r}\otimes vec(\boldsymbol{H})^T)\boldsymbol{R}(D^{\otimes(r+2)}f)(\boldsymbol{I}_{d^r}\otimes vec(\boldsymbol{H}))\right)+ \\
&\quad + o\left(\|vec(\boldsymbol{H})\|^2\right) \\
IV\left(D^{\otimes r}\hat{f}(\cdot,\boldsymbol{H})\right) &= \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}tr\left((\boldsymbol{H}^{-1})^{\otimes r}\boldsymbol{R}(D^{\otimes r}K)\right)+o\left(\frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}\|vec(\boldsymbol{H}^{-1})\|^r\right)
\end{aligned}
\tag{4.7}
$$

y con esto

$$
\begin{aligned}
AMISE\left\{D^{\otimes r}\hat{f}(\cdot,\boldsymbol{H})\right\} &= \frac{1}{n}|\boldsymbol{H}|^{-\frac{1}{2}}tr\left((\boldsymbol{H}^{-1})^{\otimes r}\boldsymbol{R}(D^{\otimes r}K)\right)+ \\
&\quad + \frac{1}{4}\mu_2^2(K)tr\left((\boldsymbol{I}_{d^r}\otimes vec(\boldsymbol{H})^T)\boldsymbol{R}(D^{\otimes(r+2)}f)(\boldsymbol{I}_{d^r}\otimes vec(\boldsymbol{H}))\right)
\end{aligned}
\tag{4.8}
$$

La matriz óptima que minimiza $AMISE\left\{D^{\otimes r}\hat{f}(\cdot,\boldsymbol{H})\right\}$ no tiene una formula cerrada.

Sin embargo, como muestran (Chacón y col., 2011) en el Teorema 6. Si f es una densidad normal con media $\boldsymbol{\mu}$ y varianza $\boldsymbol{\Sigma}$ y K es un kernel normal, $AMISE\left\{D^{\otimes r}\hat{f}(\cdot,\boldsymbol{H})\right\}$ tiene una fórmula más sencilla y el valor de la matriz de ancho de banda que minimiza dicha función es

$$
\boldsymbol{H}^{NS}_{AMISE,r} = \left(\frac{4}{d+2r+2}\right)^{\frac{2}{d+2r+2}}\boldsymbol{\Sigma}n^{-\frac{2}{d+2r+2}} \tag{4.9}
$$

De esta fórmula, podemos deducir para el caso $r=0$, la regla general o *rule-of-thumb*, Sin embargo, como hemos mencionado en (3.5), el cálculo de derivadas de ordenes elevados resulta complicada en la práctica. En el caso de la distribución normal, existe una fórmula cerrada para el cálculo de $D^{\otimes r}\phi_{\boldsymbol{H}}$ pero esta tiene una aplicación complicada. Por esto, (Chacón y Duong, 2014) muestran un procedimiento recursivo para calcular las derivadas de la función de densidad normal de forma eficiente. La programación de este algoritmo tampoco resulta inmediato para su aplicación.

5. Otros métodos de estimación: Estimación de los vecinos más próximos

El estimador de densidad de k-vecinos más próximos, fue inicialmente introducido por (Loftsgaarden y Quesenberry, 1965). En dicho trabajo se propone un estimador basado en la idea de que $P\left(X_i \in B_d(r, \boldsymbol{x})\right) = \int_{B_d(r,\boldsymbol{x})} f(y)dy$.

Es decir, si el número de vecinos k considerados es pequeño respecto de el número de observaciones en la muestra n, el radio de la bola cerrada en la que están los k vecinos más próximos será pequeño también y por tanto, se puede considerar que la densidad en esa bola será constante, es decir,

$$\int_{B_d(r,\boldsymbol{x})} f(y)dy = \int_{B_d(r,\boldsymbol{x})} f(x)dy = f(x)\int_{B_d(r,\boldsymbol{x})} dy = f(x)V_d(r,\boldsymbol{x})$$

y por tanto,

$$f(\boldsymbol{x}) = \lim_{r\to 0} \frac{P(\boldsymbol{X}_i \in B_d(r,\boldsymbol{x}))}{V_d(r,\boldsymbol{x})}$$

donde $B_d(r,\boldsymbol{x}) = \{\boldsymbol{z} \in \mathbb{R}^p \mid \|\boldsymbol{x}-\boldsymbol{z}\| \leq r\}$ es la bola euclidiana cerrada de dimensión p y $V_d(r,\boldsymbol{x}) = \frac{\pi^{\frac{p}{2}}}{\Gamma(\frac{p}{2}+1)} r^p = V_d(1,\boldsymbol{x})r^p$ es su volumen. Así se puede definir el estimador de k-vecinos más cercanos de Loftsgaarden y Quesenberry, $\hat{f}_{knn_L}(\boldsymbol{x})$, como[i]

$$\hat{f}_{knn_L}(\boldsymbol{x}) = \frac{k}{n}\frac{1}{V_d(r_K,\boldsymbol{x})} = \frac{1}{nr_k^p}\frac{k}{V_d(1,\boldsymbol{x})} \tag{5.1}$$

donde k es el número de vecinos más próximos a considerar y $r_K = \|\boldsymbol{X}_{(k)} - \boldsymbol{x}\|$ en la que empleamos la notación usual para estadísticos de orden. Podemos analizar el estimador (5.1) en mayor detalle. Así, observamos que lo que hace es dar un peso de $\frac{1}{V_d 1,\boldsymbol{x}}$ a las observaciones dentro de $B_d(r_k,\boldsymbol{x})$. Teniendo esto en cuenta, podemos reescribir el estimador como,

$$\begin{aligned}\hat{f}_{knn_L}(\boldsymbol{x}) &= \frac{1}{n}\sum_{i=1}^{n}\frac{1}{V_d(1,\boldsymbol{x})}\mathbf{1}_{\{\boldsymbol{X}_i \in B_d(\boldsymbol{x},r_k)\}}(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{V_d(1,\boldsymbol{x})}\mathbf{1}_{\{\|\boldsymbol{X}_i-\boldsymbol{x}\|\leq r_k\}}(\boldsymbol{x}) = \\ &= \frac{1}{nr_k^p}\sum_{i=1}^{n}\frac{1}{V_d(1,\boldsymbol{x})}\mathbf{1}_{\left\{\frac{\|\boldsymbol{X}_i-\boldsymbol{x}\|}{r_k}\leq 1\right\}}(\boldsymbol{x})\end{aligned} \tag{5.2}$$

Como puede apreciarse, estamos considerando una ponderación basada en las observaciones. En este caso, estamos ponderando de manera uniforme, esto es, con una función de peso discontinua en la frontera de la bola. En este caso, estamos considerando la función núcleo de la familia beta con $r = 0$ presentada en **??**.

Sin embargo, podríamos considerar una función de peso continua. La generalización natural consistiría en considerar funciones de tipo núcleo de la familia beta de órdenes superiores pero podríamos considerar otras funciones de tipo núcleo como la normal.

Con esto, (Moore y Yackel, 1977) generalizaron el estimador de k-vecinos más próximos considerando una función de densidad ω. (Mack y Rosenblatt, 1979) generalizaron el empleo de esta función

[i] Definen el estimador como $\frac{k-1}{n}\frac{1}{V(r_K,\boldsymbol{x})}$, por mantener la coherencia entre los estimadores se emplea k

de peso imponiendo únicamente que fuera una función acotada tal que $\int \omega(\boldsymbol{u})d\boldsymbol{u} = 1$ obteniendo la versión más general del estimador de densidad de k-vecinos más próximos,

$$\hat{f}_{knn}(\boldsymbol{x}) = \frac{1}{nr_k^p} \sum_{i=1}^{n} \omega\left(\frac{\boldsymbol{X}_i - \boldsymbol{x}}{r_k}\right) \tag{5.3}$$

donde asumimos que k depende del número de observaciones n, $k = k(n)$, de forma que $\lim_{n\to\infty} k(n) = \infty$ y $\lim_{n\to\infty} k(n)/n = 0$. Estas condiciones son análogas a aquellas que se establecen para el ancho de banda del estimador de densidad de tipo núcleo.

Si consideramos el estimador de tipo núcleo, tenemos en cuenta las observaciones que están a una cierta distancia del punto en el que estamos, mientras que el K-NN toma las observaciones entre el puntu y el k-ésimo vecino más cercano. Es por eso, que se puede entender el método K-NN como un estimador de tipo Kernel con ancho de banda dinámico.

El estimador de K-vecinos más próximos tiene una gran ventaja sobre el de tipo Kernel: Es muy fácil de calcular. Es por esto que para dimensiones elevadas, suele preferirse este al de tipo kernel.

6. Análsis de datos reales en Python

El objetivo de este capítulo radica en la implementación práctica del estimador de densidad de tipo núcleo a conjuntos de datos reales.

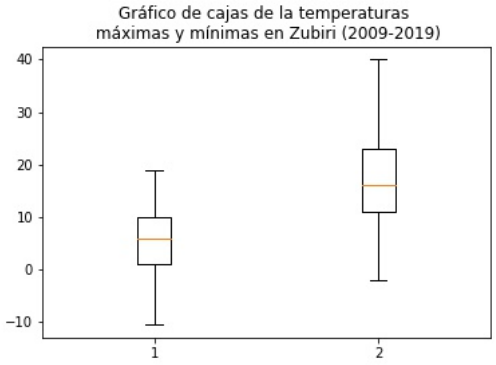
Nos centraremos en realizar un estudio de la temperatura de recogida entre 2009 y 2020 en la estación de Zubiri, un pequeño pueblo del pre-Pirineo Navarro situado a una altitud de 526 msnm. Los datos han sido tomados de la página de Meteorología y climatología de Navarra del Gobierno de Navarra.

Figura 16: Zubiri (Navarra)

El análisis con Python se ha intentado mantener lo más independiente posible. Es decir, se ha intentado que la dependencia del código desarrollado en cuanto a otros paquetes de Python sea mínima. En los Anexos II y III se puede encontrar parte de este código. Por otra parte, todo el código desarrollado esta disponible en GitHub y libremente accesible. Los paquetes que necesariamente hemos tenido que utilizar durante la elaboración del código son *NumPy* (Harris y col., 2020), *SciPy* (Virtanen y col., 2020) y *MatPlotLib* (Hunter, 2007). Son paquetes estándares en Python que sirven de base. Además, como se ha comentado anteriormente en el trabajo, debido a la dificultad de calcular derivadas de la función de densidad normal multivariante, se ha utilizado el paquete de cálculo simbólico *SymPy* (Meurer y col., 2017).

6.1. Conjunto de datos univariante. Temperaturas máximas y mínimas de Zubiri (Navarra) entre 2009 y 2019

En primer lugar, mostramos el gráfico de cajas para las temperaturas máximas y mínimas en Zubiri.



Podemos ver que es una zona en la que el calor no es especialmente importante durante el año.

Realizamos la estimación del ancho de banda para los diferentes métodos que hemos propuesto obteniendo los siguientes resultados,

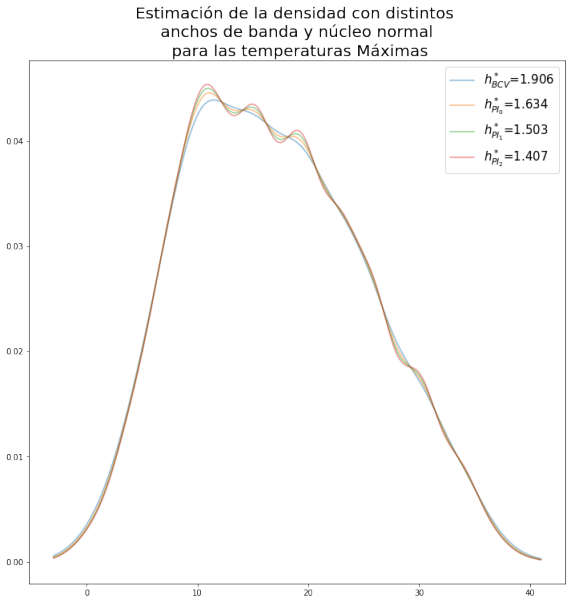


Figura 17: Estimación de la densidad de las temperaturas mínimas con el estimador de tipo núcleo. Se han usado los anchos de banda óptimos proporcionados por BCV y PI.

Podemos ver cómo los estimadores de *plug-in* se comportan de igual manera dando lugar a tres máximos entre los 10 y los 25 grados centígrados. El selector BCV no recoge estas pequeñas fluctuaciones y da un ancho de banda mayor, como era de esperar. Puede que esas fluctuaciones se deban a que los datos están tomados de forma ordinal. Quizás, al tratarse la estación de Zubiri de una estación manual, las medidas pueden tender a estar redondeadas por lo que vemos estos máximos locales.

Puede que sea esta también la razón por la que falla el selector de UCV que sobreajusta a los datos

de la muestra como se muestra en la siguiente imagen.

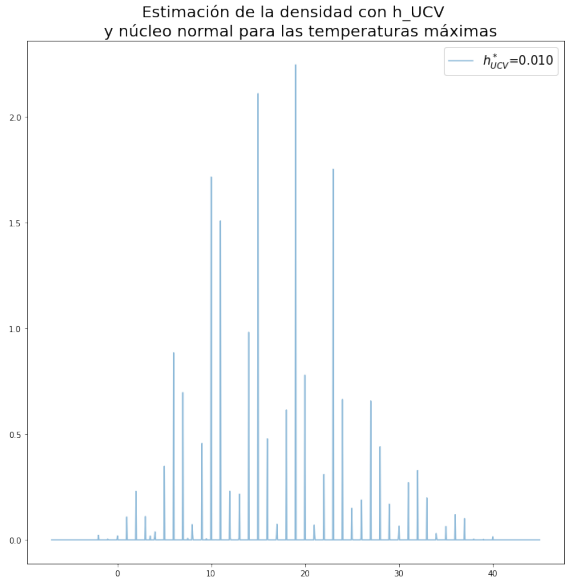


Figura 18: Estimación de la densidad de las temperaturas máximas con el estimador de tipo núcleo. Se ha usado el anchos de banda proporcionado por UCV. Se puede ver el sobreajuste claramente.

A simple vista, parece que en Zubiri es más probable tener días de máximas negativas que días con máximas que superen los 40 grados centígrados.

Realizamos el mismo análisis con las temperaturas mínimas,

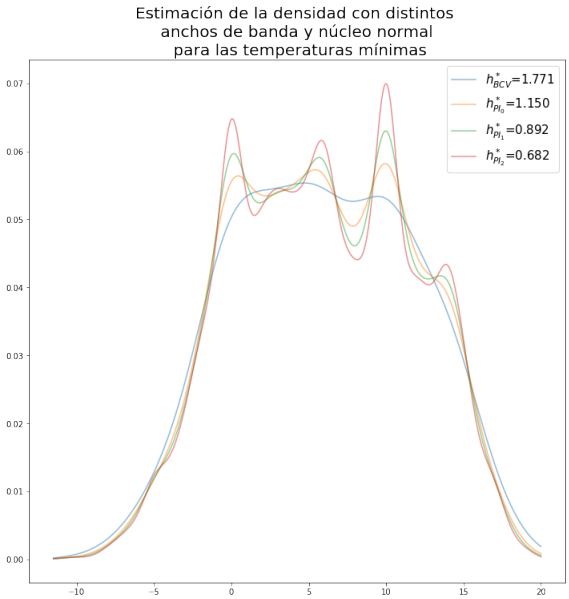


Figura 19: Estimación de la densidad de las temperaturas mínimas con el estimador de tipo núcleo. Se han usado los anchos de banda óptimos proporcionados por BCV y PI.

Observamos un fenómeno igual al que veíamos en en el caso de las temperaturas máximas. El selector de *plug-in* ciertos máximos locales que el BCV suaviza. En este caso también podría deberse al tipo de datos que estamos considerando y a las características de su recogida.

De este gráfico podríamos resaltar que es mucho más probable tener noches heladoras en Zubiri que una noche tórrida superando los 20 grados centígrados.

6.2. Conjunto de datos bivariante. Densidad Trimodal III

Debido a la complejidad de estimar en los datos de temperaturas máximas y mínimas de Zubiri, se realiza el análisis con una distribución que mencionan (Chacón y Duong, 2018; Wand y Jones, 1993) entre otros.

Generamos 100 muestras de esta función de densidad que viene descrita mediante la variable ateatoria, X, donde conocemos su función de densidad real,$f_X(x)$.

$$X \sim \frac{3}{7}N\left(\begin{pmatrix}-1\\0\end{pmatrix}, \frac{1}{25}\begin{pmatrix}9 & \frac{63}{10}\\ \frac{63}{10} & \frac{49}{4}\end{pmatrix}\right) + \frac{3}{7}N\left(\begin{pmatrix}0\\1\end{pmatrix}, \frac{1}{25}\begin{pmatrix}\frac{9}{2} & 0\\ \frac{}{\sqrt{3}} & \frac{49}{4}\end{pmatrix}\right) + \frac{1}{7}N\left(\begin{pmatrix}1\\ \frac{-2}{\sqrt{3}}\end{pmatrix}, \frac{1}{25}\begin{pmatrix}9 & 0\\ 0 & \frac{49}{4}\end{pmatrix}\right)$$

donde $Y \sim Bin(2, 0{,}24)$

En primer lugar vemos cómo

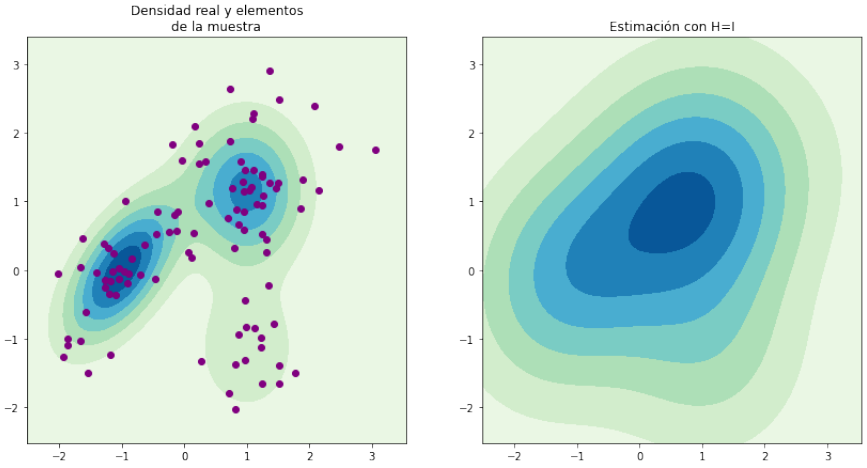


Figura 20: A la izquierda podemos ver las curvas de nivel de la función de densidad real generada de X. Además, se muestran sobre las curvas de nivel los elementos de la muestra generada aleatoriamente. A la derecha se puede visualizar una primera estimación de la densidad con el núcleo normal y la matriz de ancho de banda $\boldsymbol{I}_2$

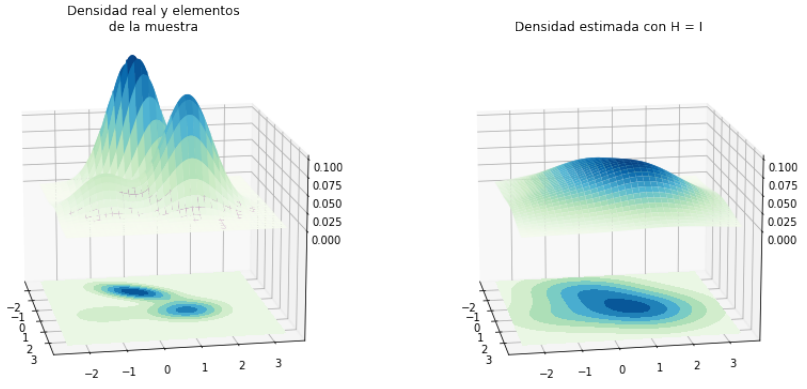


Figura 21: El gráfico similar al anterior nos muestra la superficie de la función de densidad.

Calculamos los anchos de banda óptimos para los métodos de validación cruzada insesgada y sesgada respectivamente, y para el estimador *plug-in* de 1 etapa.

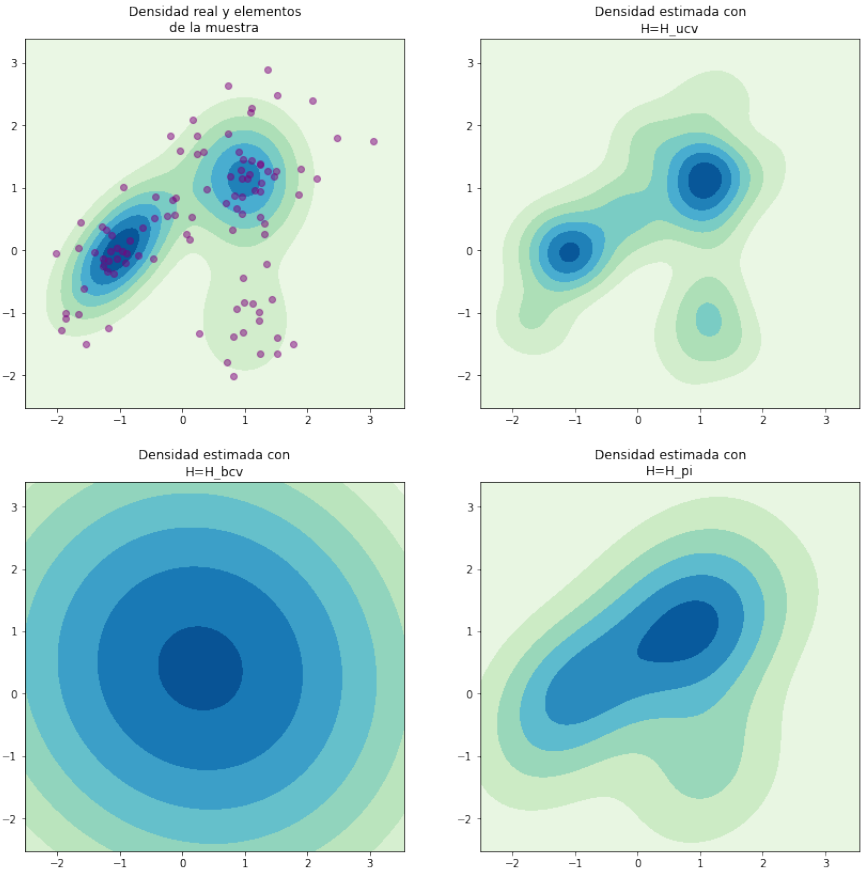


Figura 22: Podemos ver la estimación con los distintos anchos de banda óptimos

Vemos que el ancho de banda con el que se consigue una estimación mejor es el de validación

cruzada insesgada si bien sobreajusta ligeramente la densidad ya que se ven más marcadas las modas de la distribución. Por otro lado, aunque vemos que el desempeño del estimador de *plug-in*, también es muy satisfactorio si bien infra-ajusta un poco la densidad y la moda más pequeña no se aprecia muy bien. El selector de validación cruzada sesgada es el que peor rendimiento ofrece proporcionando una matriz de ancho de banda muy grande.

7. Conclusiones

El presente trabajo hace un repaso pormenorizado de la estimación de funciones de densidad mediante funciones de tipo núcleo. Todos los resultados expuestos en los capítulos relativos al estimador de densidades de tipo núcleo están autocontenidos y las pruebas están completas. Esto ya de por sí es un pequeño hito realizado ya que los resultados suelen estar muchas veces referenciados entre distintas publicaciones y las notaciones no suelen coincidir en todos los casos. Esto tiene importancia en un tema como el que nos ocupa ya que como se ha podido comprobar la notación no es del todo amigable especialmente en el caso multivariante.

Detallamos los principales métodos de selección de anchos de banda o de matrices de anchos de banda y mostramos algunos ejercicios sobre cómo realizar esta tarea computacionalmente.

Pero, principalmente se ha encaminado el objetivo principal que no era otro que el de construir una librería para Python en la que se pudiera realizar la estimación de las funciones de densidad a partir de una muestra dada permitiendo la máxima flexibilidad a la hora de elegir el ancho de banda. Si bien este objetivo no se ha resuelto totalmente debido a la complejidad de los cálculos que nos ocupan, sí que se ha avanzado en el desarrollo de funciones para finalizarlo próximamente.

Futuras investigaciones

Las lineas de expansión para el presente trabajo son tantas como las que puedan ser imaginables.

En primer lugar y por coherencia con lo mencionado anteriormente, el desarrollo de un paquete autocontenido en Python que permita realizar la estimación de funciones de densidad, no solo de tipo kernel sino también ampliándolo a la estimación mediante los vecinos más próximos o a la estimación mediante histogramas. El paquete debería contener además una cantidad importante de opciones para la selección de anchos de banda o de matrices de anchos de banda.

Por otra parte, en el presente trabajo hemos realizado un estudio de la estimación de densidades teniendo en cuanta la norma L_2 del error entre la estimación y la densidad. Siguiendo a (Devroye y Lugosi, 2012) podríamos realizar el mismo análisis, que en palabras de Devroye estudiar el problema en norma l1 tiene mas sentido. En este mismo sentido, (Biau y Devroye, 2015) estudia el estimador knn en un contexto de error l1 del estimador, (Zhao y Lai, 2020) estudia la convergencia l_1 y l_∞ del estimador de K-vecinos más próximos.

De la misma forma podríamos implementar este método para la regresión kernel para estimar la expectativa condicional de una variable aleatoria. Este punto está más allá del objetivo presentado, pero parece ser una extensión natural del presente trabajo.

También podríamos centrarnos en aplicaciones como Mean Shift Clustering que relata (Chacón

y Duong, 2018) en el que con el estimador de densidad y el estimador de la derivada se aplica un algoritmo de ascenso por gradiente para agrupar los puntos conforme a la moda que es más probable.

El abanico de posibilidades para seguir investigando por esta linea es tan extenso como apasionante.

Referencias

Aldershof, B., Marron, J., Park, B. & Wand, M. (1995). Facts about the Gaussian probability density function. *Applicable Analysis*, *59*(1-4), 289-306.

Bhattacharya, P. (1967). Estimation of a probability density function and its derivatives. *Sankhyā: The Indian Journal of Statistics, Series A*, 373-382.

Biau, G. & Devroye, L. (2015). *Lectures on the nearest neighbor method* (Vol. 246). Springer.

Casa, A., Chacón, J. E. & Menardi, G. (2019). Modal clustering asymptotics with applications to bandwidth selection.

Chacón, J. E. & Duong, T. (2010). Multivariate plug-in bandwidth selection with unconstrained pilot bandwidth matrices. *Test*, *19*(2), 375-398.

Chacón, J. E. & Duong, T. (2014). Efficient recursive algorithms for functionals based on higher order derivatives of the multivariate Gaussian density.

Chacón, J. E. & Duong, T. (2018). *Multivariate kernel smoothing and its applications*. Chapman; Hall/CRC. https://doi.org/10.1201/9780429485572

Chacón, J. E., Duong, T. & Wand, M. (2011). Asymptotics for general multivariate kernel density derivative estimators. *Statistica Sinica*, 807-840.

Devroye, L. & Lugosi, G. (2012). *Combinatorial methods in density estimation*. Springer Science & Business Media.

Duong, T. (2004). *Bandwidth selectors for multivariate kernel density estimation*. University of Western Australia.

Duong, T. y col. (2007). ks: Kernel density estimation and kernel discriminant analysis for multivariate data in R. *Journal of Statistical Software*, *21*(7), 1-16.

Duong, T. (2015). Spherically symmetric multivariate beta family kernels. *Statistics & Probability Letters*, *104*, 141-145.

Faraway, J. J. & Jhun, M. (1990). Bootstrap choice of bandwidth for density estimation. *Journal of the American Statistical Association*, *85*(412), 1119-1122.

Fourdrinier, D., Strawderman, W. E. & Wells, M. T. (2018). Spherically Symmetric Distributions. *Shrinkage Estimation* (pp. 127-150). Springer International Publishing. https://doi.org/10.1007/978-3-030-02185-6_4

García-Portugués, E. (2021). *Notes for Nonparametric Statistics* [Version 6.4.5. ISBN 978-84-09-29537-1]. https://bookdown.org/egarpor/NP-UC3M/

Hall, P. & Marron, J. S. (1991). Local minima in cross-validation functions. *Journal of the Royal Statistical Society: Series B (Methodological)*, *53*(1), 245-252.

Hall, P. & Marron, J. S. (1987). Estimation of integrated squared density derivatives. *Statistics & Probability Letters*, *6*(2), 109-115.

Hall, P., Marron, J. & Park, B. U. (1992). Smoothed cross-validation. *Probability theory and related fields*, *92*(1), 1-20.

Härdle, W. K., Müller, M., Sperlich, S. & Werwatz, A. (2004). *Nonparametric and semiparametric models*. Springer Science & Business Media.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., Fernández del Río, J., Wiebe, M., Peterson, P., ... Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*, 357-362. https://doi.org/10.1038/s41586-020-2649-2

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90-95. https://doi.org/10.1109/MCSE.2007.55

Jones, M. & Kappenman, R. (1992). On a class of kernel density estimate bandwidth selectors. *Scandinavian Journal of Statistics*, 337-349.

Langrené, N. & Warin, X. (2019). Fast and stable multivariate kernel density estimation by fast sum updating. *Journal of Computational and Graphical Statistics*, *28*(3), 596-608.

Loftsgaarden, D. & Quesenberry, C. (1965). A nonparametric estimate of a multivariate density function. *Annals of Mathematical Statistics*, *36*, 1049-1051.

Mack, Y. & Rosenblatt, M. (1979). Multivariate k-nearest neighbor density estimates. *Journal of Multivariate Analysis*, *9*(1), 1-15.

Mammen, E., Martínez Miranda, M. D., Nielsen, J. P. & Sperlich, S. (2011). Do-validation for kernel density estimation. *Journal of the American Statistical Association*, *106*(494), 651-660.

Meurer, A., Smith, C. P., Paprocki, M., Čertik, O., Kirpichev, S. B., Rocklin, M., Kumar, A., Ivanov, S., Moore, J. K., Singh, S., Rathnayake, T., Vig, S., Granger, B. E., Muller, R. P., Bonazzi, F., Gupta, H., Vats, S., Johansson, F., Pedregosa, F., ... Scopatz, A. (2017). SymPy: symbolic computing in Python. *PeerJ Computer Science*, *3*, e103. https://doi.org/10.7717/peerj-cs.103

Moore, D. S. & Yackel, J. W. (1977). Consistency properties of nearest neighbor density function estimators. *The Annals of Statistics*, 143-154.

Parzen, E. (1962). On estimation of a probability density function and mode. *The annals of mathematical statistics*, *33*(3), 1065-1076.

Rosenblatt, M. (1956). Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics*, *27*(3), 832-837. https://doi.org/10.1214/aoms/1177728190

Sain, S. R., Baggerly, K. A. & Scott, D. W. (1994). Cross-validation of multivariate densities. *Journal of the American Statistical Association*, *89*(427), 807-817.

Scott, D. W. (2015). *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons.

Scott, D. W., Tapia, R. A. & Thompson, J. R. (1977). Kernel density estimation revisited. *Nonlinear Analysis: Theory, Methods & Applications*, *1*(4), 339-372.

Scott, D. W. & Terrell, G. R. (1987). Biased and unbiased cross-validation in density estimation. *Journal of the american Statistical association*, *82*(400), 1131-1146.

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.

Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. Chapman; Hall.

Taylor, C. C. (1989). Bootstrap choice of the smoothing parameter in kernel density estimation. *Biometrika*, *76*(4), 705-712.

Terrell, G. R. (1990). The maximal smoothing principle in density estimation. *Journal of the American statistical association*, *85*(410), 470-477.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ... SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261-272. https://doi.org/10.1038/s41592-019-0686-2

Wand, M. P. & Jones, M. C. (1993). Comparison of smoothing parameterizations in bivariate kernel density estimation. *Journal of the American Statistical Association*, *88*(422), 520-528.

Wand, M. P. & Jones, M. C. (1994). *Kernel smoothing*. CRC press.

Zhao, P. & Lai, L. (2020). Analysis of KNN Density Estimation. *arXiv preprint arXiv:2010.00438*.

Anexo I. Demostración resultado en (2.4.1)

Lema 1. *Sea $K \in C^{p+2}(\mathbb{R})$ una función núcleo simétrica respecto al origen tal que $K(x) = o(t^{-p-2})$. Entonces,*

$$\int t^i K^{(p)}(t)dt = \begin{cases} 0 & \text{si} \quad i < p \text{ o } i+p \text{ es impar} \\ (-1)^p p! & \text{si} \quad i = p \\ (-1)^p \frac{(p+2)!\mu_2(K)}{2} & \text{si} \quad i = p+2 \end{cases}$$

Demostración. En primer lugar, notamos que al ser $K(x) = o(x^{-p-2})$, entonces $K^{(i)}(x) = o(t^{-p-2})$ $\forall i < p+2$. Además, al ser $K(x)$ una función par, $K^{(p)}$ es par cuando p es par e impar cuando p es impar.

Cuando $i+p$ es impar, i o p es impar. Si p es impar (par), i es par (impar) y por tanto, $K^{(p)}(x)$ y x^i son impar y par (par e impar) respectivamente. Por esto, su producto es impar y por tanto,

$$\int_{\mathbb{R}} x^i K^{(p)}(x)dx = 0$$

Para el resto de casos, realizamos la integración por partes.

$$\begin{aligned} \int x^i K^{(p)}(x)dx &= \lim_{x\to\infty} x^i K^{(p-1)}(x) - \lim_{x\to-\infty} x^i K^{(p-1)}(x) - i\int x^{i-1}K^{(p-1)}(x)dx = \\ &= -i\int x^{i-1}K^{(p-1)}(x)dx = \\ &= -i\lim_{x\to\infty} x^{i-1}K^{(p-2)}(x) - \lim_{x\to-\infty} x^{i-1}K^{(p-2)}(x) + (-1)^2 i(i-1)\int x^{i-2}K^{(p-2)}(x)dx = \\ &= (-1)^2 i(i-1)\int x^{i-2}K^{(p-2)}(x)dx = \cdots \\ &= (-1)^2 i!\int K^{(p-i)}(x)dx \end{aligned}$$

Entonces,

- $i < p$

$$\begin{aligned} \int x^i K^{(p)}(x)dx &= (-1)^i i!\int K^{(p-i)}(x)dx = \\ &= (-1)^i i!\left(\lim_{x\to\infty} K^{(p-i-1)}(x) - \lim_{x\to-\infty} K^{(p-i-1)}(x)\right) = 0 \end{aligned}$$

- $i = p$

$$\int x^i K^{(p)}(x)dx = (-1)^p p!\int K(x)dx = (-1)^2 p!$$

- $i = p+2$

$$\int x^i K^{(p)}(x)dx = (-1)^p p(p-1)\cdots(p-(p+3))\int x^2 K(x)dx = (-1)^p \frac{p!}{2}\mu_2(K)$$

□

Observación. *En el caso $p = 2$, tenemos que si $K \in C^4(\mathbb{R})$ una función núcleo simétrica respecto al origen tal que $K(x) = o(t^{-4})$, entonces,*

$$\int t^i K''(t)dt = \begin{cases} 0 & \text{si} \quad i < 2 \text{ o } i \text{ es impar} \\ 2 & \text{si} \quad i = 2 \\ 12\mu_2(K) & \text{si} \quad i = 4 \end{cases}$$

En este punto, nos disponemos a generalizar el resultado del Lema 3.2 en (Scott y Terrell, 1987). Estos consideran que la función núcleo es una función de densidad con soporte compacto. Sin embargo, es suficiente con que la tasa de decaimiento en las colas de la función de densidad sea lo suficientemente rápida.

Teorema 2 (Generalización del Lema 3.2 en (Scott y Terrell, 1987)). *Sean $f \in C^{p+2}(\mathbb{R})$ una función de densidad y $K \in C^{p+2}(\mathbb{R})$ una función núcleo simétrica respecto al origen tal que $K(x) = o(t^{-p-2})$. Entonces,*

$$\mathbb{E}\left[R(\hat{f}^{(p)})\right] = \frac{n-1}{n} R(f^{(p)}) + \frac{1}{nh^5} R(K^{(p)})$$

Demostración.

$$\mathbb{E}\left[R(\hat{f}^{(p)})\right] = \mathbb{E}\left[\int \hat{f}^{(p)}(x;h)^2 dx\right] = \mathbb{E}\left[\frac{1}{n^2}\int \left(\sum_{i=1}^{n} K_h^{(p)}(x - X_i)\right)^2 dx\right] =$$

$$= \mathbb{E}\left[\frac{1}{n^2}\int \sum_{i=1}^{n} K_h^{(p)}(x - X_i)^2 dx\right] + \mathbb{E}\left[\frac{1}{n^2}\int \sum_{i=1}^{n}\sum_{\substack{j=1 \\ j\neq i}}^{n} K_h^{(p)}(x - X_i) K_h^{(p)}(x - X_j) dx\right]$$

El primer sumando es determinista y puede reescribirse como

$$\frac{1}{n^2}\int \sum_{i=1}^{n} K_h^{(p)}(x - X_i)^2 dx \underset{x - X_i = hz}{=} \frac{1}{n^2}\int \sum_{i=1}^{n} K_h^{(p)}(hz)^2 h dz =$$

$$= \frac{1}{n^2 h^{2(p+1)}} nh \int K^{(p)}(z)^2 dz = \frac{1}{nh^{2p+1}} R(K^{(p)})$$

Por otro lado, el segundo sumando puede reescribirse como

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n}\mathbb{E}\left[\int K_h^{(p)}(x-X_i)K_h^{(p)}(x-X_j)dx\right] = \frac{n(n-1)}{n^2}\mathbb{E}\left[\int K_h^{(p)}(x-X_i)K_h^{(p)}(x-X_j)dx\right] =$$
$$= \frac{n(n-1)}{n^2}\int \mathbb{E}\left[K_h^{(p)}(x-X_i)K_h^{(p)}(x-X_j)\right]dx =$$
$$= \frac{n(n-1)}{n^2}\int \mathbb{E}\left[K_h^{(p)}(x-X_i)\right]\mathbb{E}\left[K_h^{(p)}(x-X_j)\right]dx =$$
$$= \frac{n(n-1)}{n^2}\int\left(\int K_h^{(p)}(x-y)f(y)\right)^2 dx \underset{x-y=hz}{=} \frac{n(n-1)}{n^2}\int\left(\int K_h^{(p)}(hz)f(x-hz)\right)^2 dx =$$
$$= \frac{n(n-1)}{n^2}\frac{h^2}{h^{2(p+1)}}\int\left(\int K^{(p)}(z)f(x-hz)\right)^2 dx =$$
$$= \frac{n(n-1)}{n^2h^{2p}}\int\left(\int K^{(p)}(z)\sum_{i=1}^{p+2}\frac{(-1)^i}{i!}(hz)^i f^{i)}(x)dz + o(h^{p+2})\right)^2 dx =$$
$$= \frac{n(n-1)}{n^2h^{2p}}\int\left(\sum_{i=1}^{p+2}\frac{(-1)^i h^i}{i!}f^{i)}(x)\int K^{(p)}(z)z^i dz + o(h^{p+2})\right)^2 dx =$$
$$= \frac{n(n-1)}{n^2}\int f^{(p)}(x)^2dx + o(h^2) = \frac{n(n-1)}{n^2}R(f'') + o(h^2)$$

en la última igualdad se ham empleado los resultados obtenidos en el lema anterior.

Así,

$$\mathbb{E}\left[R(\hat{f}'')\right] = \frac{(n-1)}{n}R(f'') + \frac{1}{nh^{2p+1}}R(K'') + o(h^2)$$

□

Hemos ampliado el resultado presentado por (Scott y Terrell, 1987) a núcleos que son funciones de densidad simétricas pero no necesariamente de soporte compacto. En el caso de tratarse de funciones de densidad con soporte compacto, notamos que la condición $K(x) \in C^{p+2}(\mathbb{R})$ que hemos impuesto es suficiente para que se garantizar la del Lema original que establece $K^{(i)}(\pm 1) = 0,\ 0 \leq i \leq p-1$. Si bien esta es más restrictiva, en el caso de funciones de densidad con soporte compacto podríamos relajarla a la original.

Por otra parte, la restriccion en la tasa de decaimiento de las colas de la distribución de K podría resultar restrictiva en caso de que quisieramos emplear como núcleo densidades asociadas a distribuciones leptocurticas.

Es ciertamente más restrictiva que

El caso concreto que se ha utilizado en (2.4.2) es,

Observación. *Sean $f \in C^4(\mathbb{R})$ una función de densidad y $K \in C^4(\mathbb{R})$ una función núcleo simétri-*

ca respecto al origen tal que $K(x) = o(t^{-4})$. *Entonces,*

$$\mathbb{E}\left[R(\hat{f}'')\right] = \frac{n-1}{n}R(f'') + \frac{1}{nh^5}R(K'')$$

Anexo II. Código Python para la estimación de densidades univariantes

El siguiente conjunto de funciones y comandos muestran el código desarrollado para este trabajo y muestran su uso.

```
- Author: Daniel Bacaicoa Barber (Sep 21)
```

Los siguientes paquetes serán necesarios de antemano

- NumPy
- SciPy
- SymPy

In [1]:
```
import numpy as np
import scipy
import matplotlib.pyplot as plt
import sympy

# Es necesario cargar ciertas funciones del paquete SciPy
from scipy import stats, integrate, signal, interpolate, misc, special
```

Funciones básicas necesarias

Desarrollaremos funciones que serán necesarias como apoyo para la estimación de densidades

Funciones Kernel soportadas

Se incluyen el Kernel normal y lo que provienen de la familia Beta (Duong, 2015)

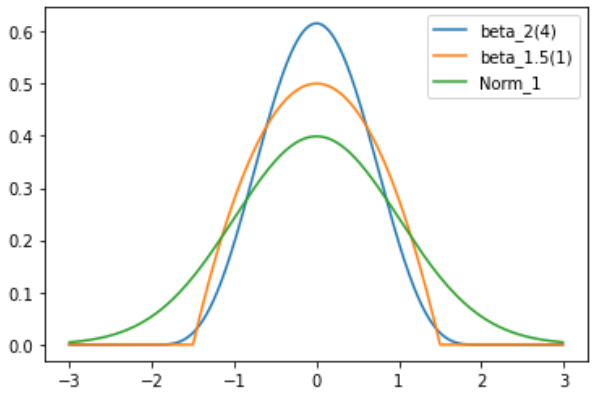
- $K_h(x) = \beta_h(x,r) = \frac{c_r}{h}\left(1-\left(\frac{x}{h}\right)^2\right)^r 1_{\{|x|\leq h\}}$ where $c_r = \frac{(2r+1)!}{2^{2r+1}(r!)^2}$
- $K_h(x) = \phi_h(x) = \frac{1}{h\sqrt{2\pi}} e^{-\frac{x^2}{2h^2}}$

Tenemos en cuenta que cualquiera de los dos es una función de densidad

1. $K(x) \geq 0$
2. $\int K(x)dx = 1$
3. $K(x) = k(-x)$

In [2]:
```
def beta(x,r,h,supp=[]):
    if supp==[]:
        supp=h
    c_r = np.math.factorial(2*r+1)/(2**(2*r+1)*np.math.factorial(r)**2)
    return c_r*(1/h)*(1-(x/h)**2)**r*(np.abs(x)<supp)
def norm(x,h,r = None):
    return stats.norm.pdf(x,0,h)
```

In [3]:
```
x = np.arange(-3,3,0.01)
f1 = beta(x, r = 4, h = 2)
f2 = beta(x, r = 1, h = 1.5)
f3 = norm(x, h = 1)
plt.plot(x,f1,x,f2,x,f3)
plt.legend(['beta_2(4)','beta_1.5(1)','Norm_1'])
plt.show()
```



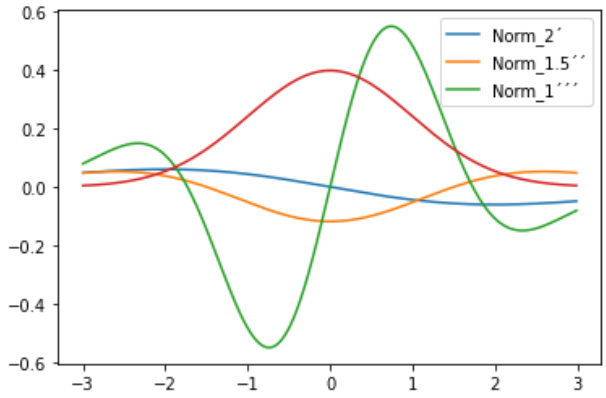
Funciones Auxiliares

Definimos algunas funciones que serán necesarias. Por ejemplo, la derivada de los núcleos

Nótese que utilizaremos el paquete de cálculo simbolico SymPy. Esto, no sería necesario en el entorno univariante ya que el cálculo numerico de las derivadas o la implementación de las derivadas de la función de densidad normal son inmediatas. Sin embargo, para mantener la coherencia con el entrono multivariante se hará de esta manera.

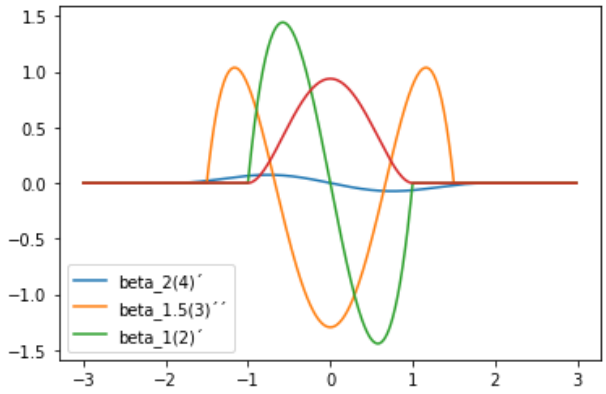
In [4]:
```
def derivate(x, K, h, r = None, order = 1):
    '''
    Cálculo simbólico de las derivadas.
    Input:
        x: puntos para evaluar la derivada
        k: Kernel
        h: ancho de banda
        r: el orden del kernel beta
        order: el orden de la derivada
    Output:
        f': la deriavada orden-ésima evaluada en x
    '''
    kern = K.__name__
    from sympy import lambdify
    sx = sympy.symbols('sx')
    if kern == 'norm':
        # for doing it without using symbolic calculus
        #(-1/h)**n * scipy.special.eval_hermitenorm(n, x/h) * stats.norm.pdf(x,0,h)
        expr = 1/(h*sympy.sqrt(2*sympy.pi))*sympy.exp(-(sx**2)/(2*h**2))
        DF = sympy.diff(expr,sx,order)
        s = (sx)
        DF_func =  lambdify(s, DF , modules='numpy')
        return DF_func(x)
    elif kern == 'beta':
        expr = (1-sx**2)**r
        DF = sympy.diff(expr,sx,order)
        s = (sx)
        DF_func =  lambdify(s, DF , modules='numpy')
        c_r = np.math.factorial(2*r+1)/(2**(2*r+1)*np.math.factorial(r)**2)
        return c_r * (1/h**(r+1)) * DF_func(x/h) * (np.abs(x)<h)
```

In [5]:
```
x = np.arange(-3,3,0.01)
f1 = derivate(x, norm, h = 2, order = 1)
f2 = derivate(x, norm, h = 1.5, order = 2)
f3 = derivate(x, norm, h = 1, order = 3)
f4 = derivate(x, norm, h = 1, order = 0)
plt.plot(x,f1,x,f2,x,f3,x,f4)
plt.legend(['Norm_2´','Norm_1.5´´','Norm_1´´´'])
plt.show()
```



In [6]:

```
x = np.arange(-3,3,0.01)
f1 = derivate(x, beta, r = 4, h = 2, order = 1)
f2 = derivate(x, beta, r = 3, h = 1.5, order = 2)
f3 = derivate(x, beta, r = 2, h = 1, order = 1)
f4 = derivate(x, beta, r = 2, h = 1, order = 0)
plt.plot(x,f1,x,f2,x,f3,x,f4)
plt.legend(['beta_2(4)´','beta_1.5(3)´´','beta_1(2)´'])
plt.show()
```



In [7]:

```
def R(K, h, r = None, order = 0):
    '''
    R(K) = \int_R K(x)^2 dx
    Input:
        k: Kernel
        h: Ancho de banda
        r: Orden del kernel beta
        order: orden de la derivada
    Output:
        R(K): El valor de la norma L2
    '''
    return integrate.quad(lambda x: derivate(x, K, h = h, r = r, order = order)**2,
                          -np.inf,np.inf)[0]

print(R(norm, h = 1, order = 0))
print(R(norm, h = 1, order = 2))
print(R(beta, h = 1, r = 1, order = 1))
print(R(beta, h = 1, r = 2, order = 1))
```

```
0.28209479177387786
0.2115710938304085
1.5
2.142857142857143
```

In [8]:

```
def mu(K, h, r = None, n = 1):
    '''
    Momento n-ésimo del Kernel
    Input:
```

```
        k: Kernel
        h: Ancho de banda
        r: Orden del kernel beta
        n: momento
    Output:
        mu_n(K): TMomento n-ésimo
    '''
    return integrate.quad(lambda x: x**n * derivate(x, K, h = h, r = r, order = 0),
                          -np.inf,np.inf)[0]

print(mu(norm, h = 1, n = 1))
print(mu(norm, h = 1, n = 2))
print(mu(beta, h = 1, r = 1, n = 1))
print(mu(beta, h = 1, r = 2, n = 2))
```

```
0.0
1.000000000000001
0.0
0.14285714285714285
```

In [9]:

```
def grid(sample):
    '''
    Genera una lista con (X_i-X_j) for i = 1,..n; j = i+1,...,n
    Input:
        sample: muestra
    Output
        Lista con los puntos
    '''
    n = len(sample)
    ordered_sample = np.sort(sample)
    symmetric_dif = np.subtract.outer(ordered_sample,ordered_sample)
    return symmetric_dif[np.tril_indices(n, k = -1)]
```

In [10]:

```
example_grid = grid(np.array([1.1,2.3,3.6]))
print(example_grid)
```

```
[1.2 2.5 1.3]
```

In [11]:

```
def sum_sum_conv(grid, K, h, r = None, order = 0):
    '''
    Sum Sum (K*K)(X_i-X_j)
    Input:
        sample: muestra
    Output
        Lista con los puntos
    '''
    bound = np.max([4*h,np.max(grid)+h])
    points = int(bound)*10000
    x = np.linspace(-bound,bound,points)
    delta = x[1]-x[0]
    convolved = signal.fftconvolve(derivate(x, K, h = h, r = r, order = order),
                                   derivate(x, K, h = h, r = r, order = order),
                                   mode='same') * delta

    f = interpolate.interp1d(x,convolved)
    return 2 * np.sum(f(grid)), f(grid)

def sum_sum_ker(grid, K, h, r = None, order = 0):
    '''
    Sum Sum K(X_i-X_j)
    Input:
        sample: muestra
    Output
        Lista con los puntos
    '''
    return 2 * np.sum(derivate(grid, K, h = h, r = r, order = order))
```

In [12]:

```
def psi(r,sigma):
```

```
    return (-1)**(r/2)*np.math.factorial(r)/((np.pi**(1/2))*((2*sigma)**(r+1))*np.math.factoria
```

In [13]:
```
print(sum_sum_conv(example_grid, norm, h = 1, order = 0))
print(sum_sum_ker(example_grid, norm, h = 1, order = 1))
print(sum_sum_conv(example_grid, beta, h = 1, r = 2, order = 1))
print(sum_sum_ker(example_grid, beta, h = 2, r = 2, order = 1))
```

```
(0.8817048076550167, array([0.19682237, 0.05913173, 0.18489831]))
-0.9992463737518531
(3.7209511390882466, array([ 9.96527046e-01, -1.07104774e-16,  8.63948523e-01]))
-0.7119140625
```

Funciones para la estimación de densidades de tipo kernel.

El estimador de densidades

In [14]:
```
def u_kde(sample, K, bw, r = None):
    '''
    Función para esimar la densidad
    Input:
        sample: muestra
        K: kernel
        bw: el ancho de banda
    Output:
        x: malla sobre la que se calcula
        kde: la estimación sobre la malla
    '''
    x = np.linspace(np.min(sample-1),np.max(sample+1),1001)
    n = len(sample)
    return x, np.sum([K(x-k, h = bw, r = r) for k in sample],0)/n
```

Definition of Bandwidth selection methods

Unbiased cross validation

In [15]:
```
def UCV(sample, K, r = None, h_min = 0.01, h_max = 3, npoints = 1001):
    n = len(sample)
    eval_grid = grid(sample)

    h_grid = np.linspace(0.01,3,1001) #h_min,h_max,npoints)


    UCV = []
    for h in h_grid:
        ucv_h = R(K, h = 1, r = r)/(n*h)
        ucv_h += sum_sum_conv(eval_grid, K, h, r = r, order = 0)[0]/(n*(n-1))
        ucv_h -= 2 * sum_sum_ker(eval_grid, K, h, r = r, order = 0)/(n*(n-1))
        UCV.append(ucv_h)

    return h_grid[np.argmin(UCV)], h_grid, UCV
```

Biased cross validation

In [16]:
```
def BCV(sample, K, r = None, h_min = 0.01, h_max = 3, npoints = 1001):
    n = len(sample)
    eval_grid = grid(sample)

    h_grid = np.linspace(h_min,h_max,npoints)

    BCV = []
    RK = R(K, h = 1, r = r)
    mu_2 =  mu(K, h = 1, r = r, n = 2)**2
    for h in h_grid:
        bcv_h = RK/(n*h)
        bcv_h += (1/4)* h**4 * mu_2 * sum_sum_conv(eval_grid, K, h, r = r, order = 2)[0]/(n*(n-
```

```
        BCV.append(bcv_h)

    return h_grid[np.argmin(BCV)], h_grid, BCV
```

Plug-in

In [17]:
```
def PI(sample, K, L, rK = None, rL = None, stages = 2):
    n = len(sample)
    hat_sigma = np.min([stats.tstd(sample),stats.iqr(sample)/(stats.norm.ppf(0.75)-stats.norm.p

    order = 4 + 2 * stages
    eval_psi = psi(order, hat_sigma)
    mu_2L = mu(L, h = 1, r = rL, n = 2)
    mu_2K = mu(K, h = 1, r = rK, n = 2)**2

    eval_grid = grid(sample)

    for i in range(stages):
        order = order - 2
        p_bw = (-2 * derivate(0, L, h = 1, r = rL, order = order)/(n*mu_2L*eval_psi))**(1/(orde
        eval_psi = derivate(0, L, h = p_bw, r = rL, order = order)/n
        eval_psi += sum_sum_ker(eval_grid, L, h = p_bw, r = rL, order = order)/(n**2)
    return (R(K, h = 1, r = rK)/(mu_2K * eval_psi * n))**(1/5)
```

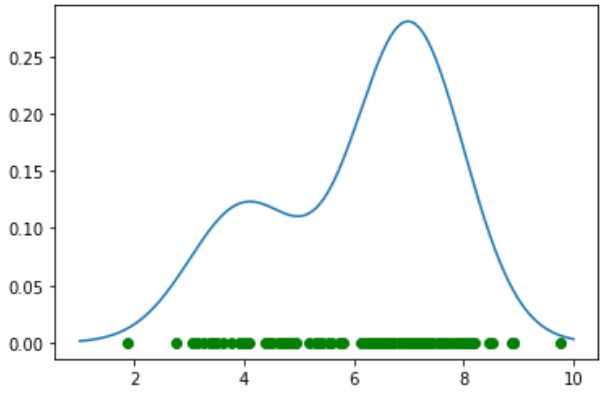
Experimento con datos sintéticos

Generamos una muestra de una mixtura de dos normales.

Generating the synthetic dataset

In [18]:
```
def normal_mixture(n, p, mu_1, sigma_1, mu_2, sigma_2):
    # The sample generated
    ber = np.random.binomial(1,p,n)
    N_1 = stats.norm.rvs(mu_1,sigma_1,size = n)
    N_2 = stats.norm.rvs(mu_2,sigma_2,size = n)
    sample = ber * N_1 + (1-ber) * N_2
    return sample
def real_normal_mixture(x, p, mu_1, sigma_1, mu_2, sigma_2, order = 0):
    # The real underlying density or its derivatives
    N_1 = derivate(x-mu_1, norm, h = sigma_1, order = 0)
    N_2 = derivate(x-mu_2, norm, h = sigma_2, order = 0)
    density = p * N_1 + (1-p) * N_2
    return density
```

In [19]:
```
n = 100
p = 0.3
mu_1 = 4; sigma_1 = 1
mu_2 = 7; sigma_2 = 1

np.random.seed(1234)
sample = normal_mixture(n,p, mu_1, sigma_1, mu_2, sigma_2)

x = np.arange(1,10,0.001)
plt.plot(x, real_normal_mixture(x, p, mu_1, sigma_1, mu_2, sigma_2))
plt.plot(sample,[0]*len(sample),'o',color = 'g')
plt.show()
```



Estimación con el Kernel Gaussiano

minimización de AMISE (f conocido)

In [20]:
```
R_f = integrate.quad(lambda x: real_normal_mixture(x, p, mu_1, sigma_1, mu_2, sigma_2, order =
                             -np.inf,np.inf)[0]
h_amise = (R(norm, h = 1)/(n*mu(norm, h = 1, n=2)**2 * R_f))**(1/5)
print('The AMISE optimal bandwidth is:%0.3f'%h_amise)
```

```
The AMISE optimal bandwidth is:0.437
```

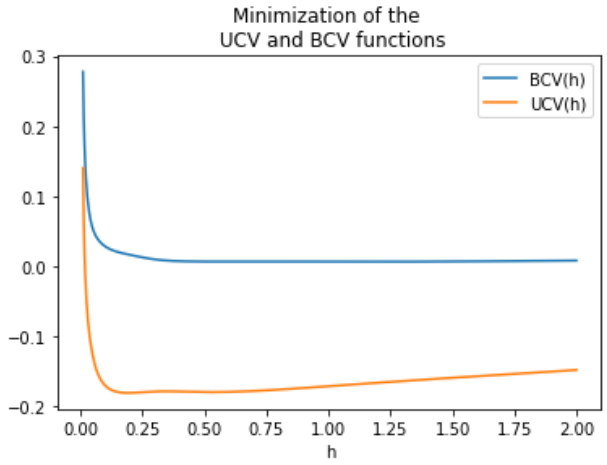
Y con el resto de métodos

In [21]:
```
# Unviased Cross Validation estimate of the Bandwidth
[h_ucv, h_grid_ucv, ucv ] = UCV(sample, norm)
print('The UCV optimal bandwidth is:%0.3f'%h_ucv)

# Calculamos el ancho de banda óptimo con BCV
[h_bcv, h_grid_bcv, bcv ] = BCV(sample, norm)
print('The UCV optimal bandwidth is:%0.3f'%h_bcv)

h_pi0 = PI(sample, K = norm, L = norm, stages = 0)
print('The optimal 0 stage PI bandwidth is:%0.3f'%h_pi0)
h_pi1 =  PI(sample, K = norm, L = norm, stages = 1)
print('The optimal 1 stage PI bandwidth is:%0.3f'%h_pi1)
h_pi2 =  PI(sample, K = norm, L = norm, stages = 2)
print('The optimal 0 stage PI bandwidth is:%0.3f'%h_pi2)
```

```
The UCV optimal bandwidth is:0.189
The UCV optimal bandwidth is:1.206
<ipython-input-12-d9d63576fb47>:2: DeprecationWarning: Using factorial() with floats is depreca
ted
  return (-1)**(r/2)*np.math.factorial(r)/((np.pi**(1/2))*((2*sigma)**(r+1))*np.math.factorial
(r/2))
The optimal 0 stage PI bandwidth is:0.697
The optimal 1 stage PI bandwidth is:0.565
The optimal 0 stage PI bandwidth is:0.498
```

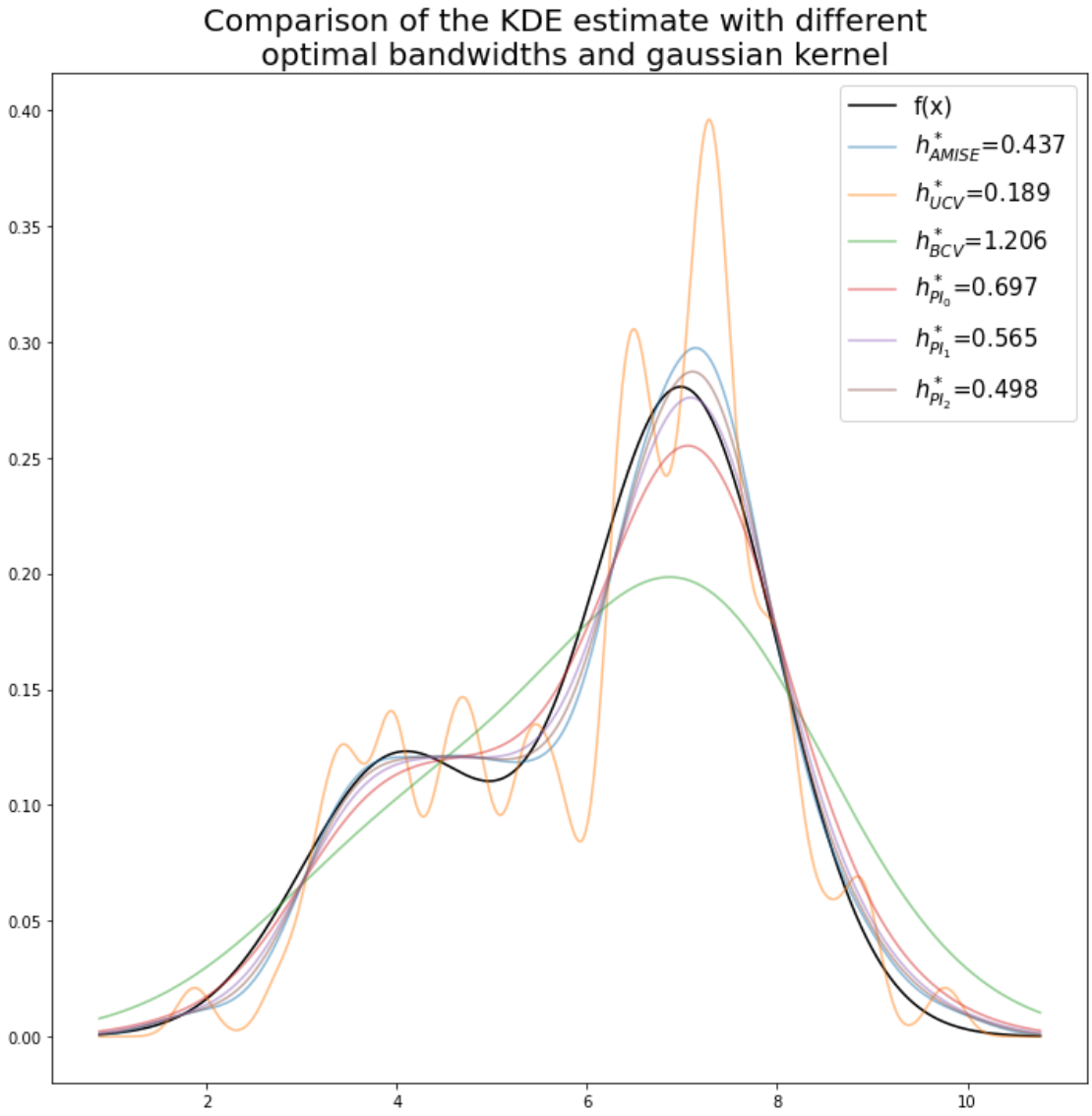
In [22]:
```
plt.plot(h_grid_bcv[0:666],bcv[0:666],label='BCV(h)')
plt.plot(h_grid_ucv[0:666],ucv[0:666],label='UCV(h)')
plt.xlabel('h')
plt.legend(loc='best')
plt.title('Minimization of the \n UCV and BCV functions')
```

Out[22]:
```
Text(0.5, 1.0, 'Minimization of the \n UCV and BCV functions')
```



In [23]:

```
t,famise = u_kde(sample, norm, h_amise)
ft = real_normal_mixture(t, p, mu_1, sigma_1, mu_2, sigma_2)
t,fucv = u_kde(sample,norm,h_ucv)
t,fbcv = u_kde(sample,norm,h_bcv)
t,fpi0 = u_kde(sample,norm,h_pi0)
t,fpi1 = u_kde(sample,norm,h_pi1)
t,fpi2 = u_kde(sample,norm,h_pi2)
plt.figure(figsize=[12,12])
plt.plot(t,ft,label='f(x)',color='black')
plt.plot(t,famise,label='$h_{AMISE}^*$=%0.3f'%h_amise,alpha=0.5)
plt.plot(t,fucv,label='$h_{UCV}^*$=%0.3f'%h_ucv,alpha=0.5)
plt.plot(t,fbcv,label='$h_{BCV}^*$=%0.3f'%h_bcv,alpha=0.5)
plt.plot(t,fpi0,label='$h_{PI_0}^*$=%0.3f'%h_pi0,alpha=0.5)
plt.plot(t,fpi1,label='$h_{PI_1}^*$=%0.3f'%h_pi1,alpha=0.5)
plt.plot(t,fpi2,label='$h_{PI_2}^*$=%0.3f'%h_pi2,alpha=0.5)

plt.legend(loc='best',fontsize=15)
#plt.title('Estimacion de la densidad con distintos \n anchos de banda y núcleo normal',fontsiz
plt.title('Comparison of the KDE estimate with different \n optimal bandwidths and gaussian ker
plt.show()
```



Estadísticas de los anchos de banda óptimos.

Ejecutaremos la estimación 10 veces para obtener información sobre los anchos de banda óptimos.

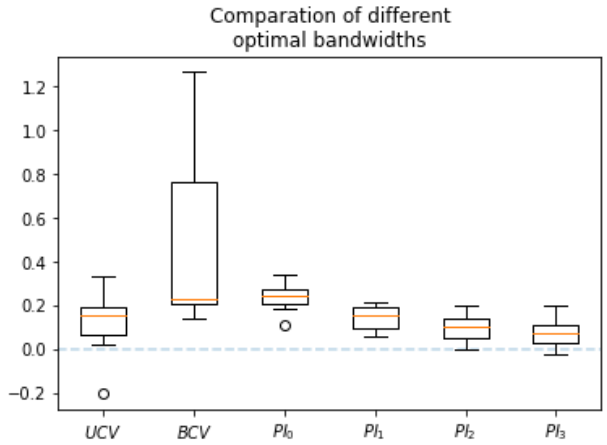
Tomaremos muestras de la misma distribución de la mezcla en cada iteración y calcularemos el ancho de banda óptimo en cada iteración.

In [24]:

```
H_UCV = []
H_BCV = []
H_PI0 = []
H_PI1 = []
H_PI2 = []
H_PI3 = []
for i in range(10):
    np.random.seed(i)
    sample = normal_mixture(n,p, mu_1, sigma_1, mu_2, sigma_2)
    H_UCV.append(UCV(sample, norm)[0])
    H_BCV.append(BCV(sample, norm)[0])
    H_PI0.append(PI(sample, K = norm, L = norm, stages = 0))
    H_PI1.append(PI(sample, K = norm, L = norm, stages = 1))
    H_PI2.append(PI(sample, K = norm, L = norm, stages = 2))
    H_PI3.append(PI(sample, K = norm, L = norm, stages = 3))
    print('%i iterations done'%(i+1)) if i % 10 == 9 else 0
results = np.array([np.array(H_UCV),np.array(H_BCV),np.array(H_PI0),np.array(H_PI1),np.array(H_
centered_results = results - h_amise
```

```
<ipython-input-12-d9d63576fb47>:2: DeprecationWarning: Using factorial() with floats is depreca
ted
  return (-1)**(r/2)*np.math.factorial(r)/((np.pi**(1/2))*((2*sigma)**(r+1))*np.math.factorial
(r/2))
10 iterations done
```

In [25]:
```
fig1, ax1 = plt.subplots()
#ax1.set_title('Comparación de la distribución\n de distintos selectores')
ax1.set_title('Comparation of different\noptimal bandwidths')
ax1.boxplot(centered_results)
ax1.hlines(0,0.5,6.5,linestyle='dashed',alpha=0.3)
plt.xticks([1,2,3,4,5,6],['$UCV$', '$BCV$', '$PI_0$', '$PI_1$', '$PI_2$', '$PI_3$'])
plt.show()
```



Usando el kernel beta (kernel Triweight, r=3)

Para el estimador PI, se utilizará un kernel piloto gaussiano

AMISE minimum when f is known

In [26]:
```
R_f = integrate.quad(lambda x: real_normal_mixture(x, p, mu_1, sigma_1, mu_2, sigma_2, order =
                             -np.inf,np.inf)[0]
h_amise = (R(norm, h = 1)/(n*mu(beta, r = 3, h = 1, n=2)**2 * R_f))**(1/5)
print('The AMISE optimal bandwidth is:%0.3f'%h_amise)
```

```
The AMISE optimal bandwidth is:1.053
```

Y con el resto de métodos

In [27]:
```
# Unviased Cross Validation estimate of the Bandwidth
[h_ucv, h_grid_ucv, ucv ] = UCV(sample, beta, r = 3)
print('The UCV optimal bandwidth is:%0.3f'%h_ucv)

# Calculamos el ancho de banda óptimo con BCV
[h_bcv, h_grid_bcv, bcv ] = BCV(sample, beta, r = 3, h_min = 0.1 , h_max = 3)
print('The UCV optimal bandwidth is:%0.3f'%h_bcv)

h_pi0 = PI(sample, K = beta, rK = 3, L = norm, stages = 0)
print('The optimal 0 stage PI bandwidth is:%0.3f'%h_pi0)
h_pi1 =  PI(sample, K = beta, rK = 3, L = norm, stages = 1)
print('The optimal 1 stage PI bandwidth is:%0.3f'%h_pi1)
h_pi2 =  PI(sample, K = norm, L = norm, stages = 2)
print('The optimal 0 stage PI bandwidth is:%0.3f'%h_pi2)
```

```
The UCV optimal bandwidth is:1.003
The UCV optimal bandwidth is:3.000
<ipython-input-12-d9d63576fb47>:2: DeprecationWarning: Using factorial() with floats is depreca
ted
  return (-1)**(r/2)*np.math.factorial(r)/((np.pi**(1/2))*((2*sigma)**(r+1))*np.math.factorial
(r/2))
```

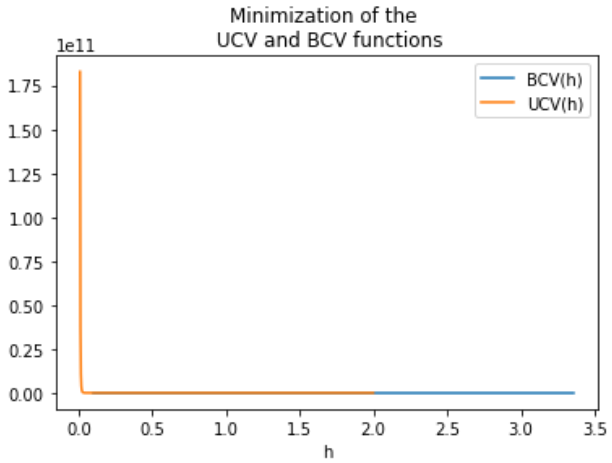
```
The optimal 0 stage PI bandwidth is:1.910
The optimal 1 stage PI bandwidth is:1.896
The optimal 0 stage PI bandwidth is:0.635
```

In [ ]:
```
[h_bcv, h_grid_bcv, bcv ] = BCV(sample, beta, r = 6, h_min = 0.1 , h_max = 5)
print('The UCV optimal bandwidth is:%0.3f'%h_bcv)
```

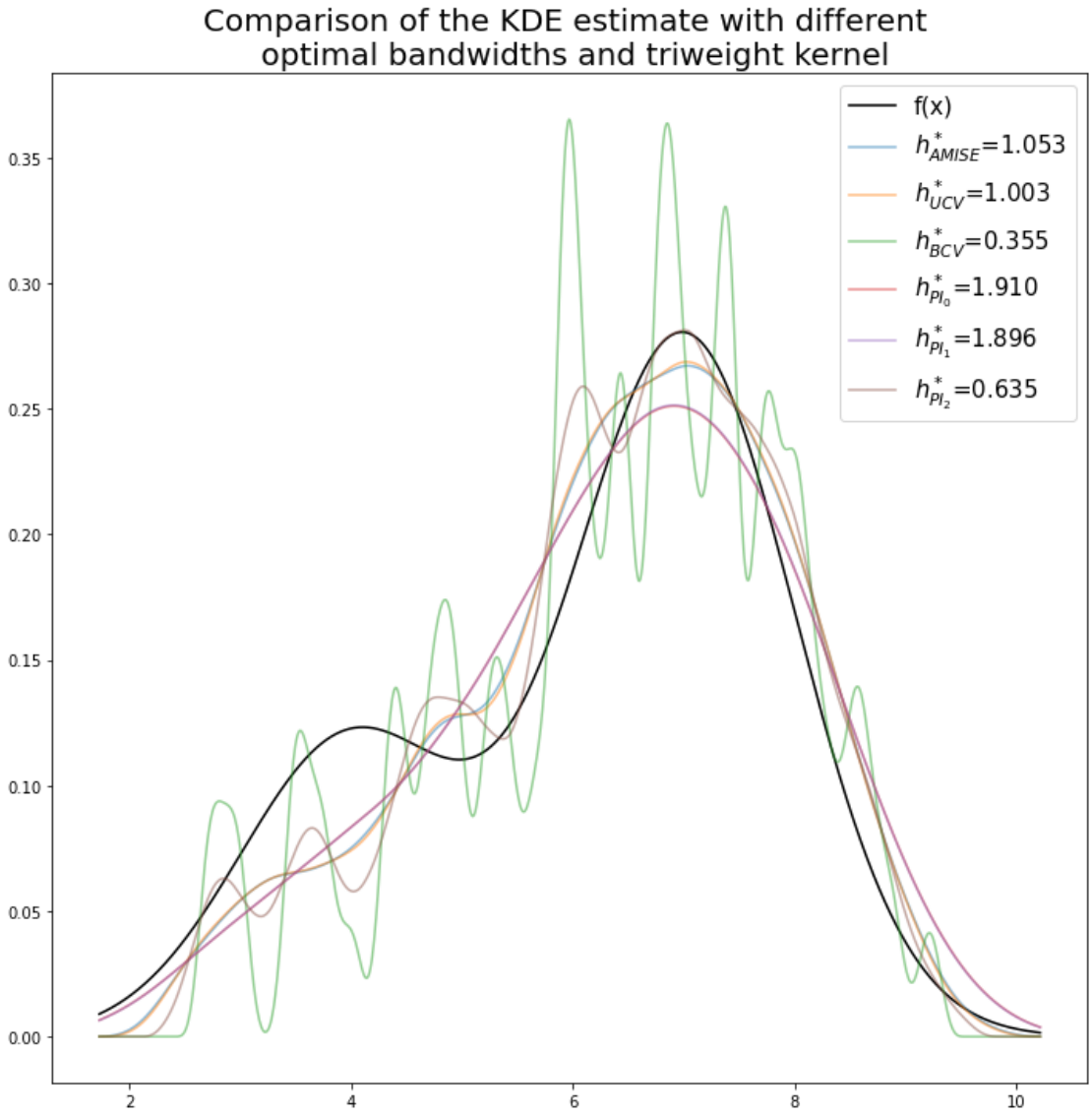
```
The UCV optimal bandwidth is:0.355
```

In [ ]:
```
plt.plot(h_grid_bcv[0:666],bcv[0:666],label='BCV(h)')
plt.plot(h_grid_ucv[0:666],ucv[0:666],label='UCV(h)')
plt.xlabel('h')
plt.legend(loc='best')
plt.title('Minimization of the \n UCV and BCV functions')
```

Out[ ]: Text(0.5, 1.0, 'Minimization of the \n UCV and BCV functions')



In [ ]:
```
t,famise = u_kde(sample, beta, r = 3, bw = h_amise)
ft = real_normal_mixture(t, p, mu_1, sigma_1, mu_2, sigma_2)
t,fucv = u_kde(sample,beta, r = 3, bw = h_ucv)
t,fbcv = u_kde(sample,beta, r = 6, bw = h_bcv)
t,fpi0 = u_kde(sample,beta, r = 3, bw = h_pi0)
t,fpi1 = u_kde(sample,beta, r = 3, bw = h_pi1)
t,fpi2 = u_kde(sample,beta, r = 3, bw = h_pi2)
plt.figure(figsize=[12,12])
plt.plot(t,ft,label='f(x)',color='black')
plt.plot(t,famise,label='$h_{AMISE}^*$=%0.3f'%h_amise,alpha=0.5)
plt.plot(t,fucv,label='$h_{UCV}^*$=%0.3f'%h_ucv,alpha=0.5)
plt.plot(t,fbcv,label='$h_{BCV}^*$=%0.3f'%h_bcv,alpha=0.5)
plt.plot(t,fpi0,label='$h_{PI_0}^*$=%0.3f'%h_pi0,alpha=0.5)
plt.plot(t,fpi1,label='$h_{PI_1}^*$=%0.3f'%h_pi1,alpha=0.5)
plt.plot(t,fpi2,label='$h_{PI_2}^*$=%0.3f'%h_pi2,alpha=0.5)

plt.legend(loc='best',fontsize=15)
#plt.title('Estimacion de la densidad con distintos \n anchos de banda y núcleo normal',fontsiz
plt.title('Comparison of the KDE estimate with different \n optimal bandwidths and triweight ke
plt.show()
```



Estadísticas de los anchos de banda óptimos.

Ejecutaremos la estimación 10 veces para obtener información sobre los anchos de banda óptimos.

Tomaremos muestras de la misma distribución de la mezcla en cada iteración y calcularemos el ancho de banda óptimo en cada iteración..

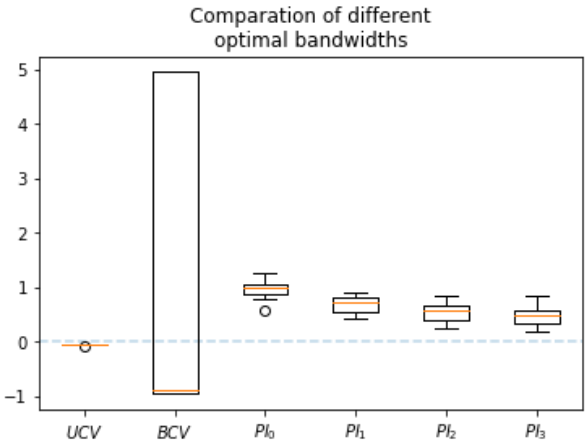
In [ ]:

```
H_UCV = []
H_BCV = []
H_PI0 = []
H_PI1 = []
H_PI2 = []
H_PI3 = []
for i in range(10):
    np.random.seed(i)
    sample = normal_mixture(n,p, mu_1, sigma_1, mu_2, sigma_2)
    H_UCV.append(UCV(sample, beta, r = 3)[0])
    H_BCV.append(BCV(sample, beta, r = 3, h_min = 0.1, h_max = 6)[0])
    H_PI0.append(PI(sample, K = beta, rK = 3, L = norm, stages = 0))
    H_PI1.append(PI(sample, K = beta, rK = 3, L = norm, stages = 1))
    H_PI2.append(PI(sample, K = beta, rK = 3, L = norm, stages = 2))
    H_PI3.append(PI(sample, K = beta, rK = 3, L = norm, stages = 3))
    print('%i iterations done'%(i+1)) if i % 10 == 9 else 0
results = np.array([np.array(H_UCV),np.array(H_BCV),np.array(H_PI0),np.array(H_PI1),np.array(H_
centered_results = results-h_amise
```

```
<ipython-input-12-d9d63576fb47>:2: DeprecationWarning: Using factorial() with floats is depreca
ted
  return (-1)**(r/2)*np.math.factorial(r)/((np.pi**(1/2))*((2*sigma)**(r+1))*np.math.factorial
(r/2))
10 iterations done
```

In [ ]:

```
fig1, ax1 = plt.subplots()
#ax1.set_title('Comparación de la distribución\n de distintos selectores')
ax1.set_title('Comparation of different\noptimal bandwidths')
ax1.boxplot(centered_results)
ax1.hlines(0,0.5,6.5,linestyle='dashed',alpha=0.3)
plt.xticks([1,2,3,4,5,6],['$UCV$', '$BCV$', '$PI_0$', '$PI_1$', '$PI_2$', '$PI_3$'])
plt.show()
```



Anexo III. Código Python para la estimación de densidades bivariante mediante un núcleo Gaussiano

El conjunto de funciones y comandos que se muestran en este documento están desarrollados para realizar la estimación de densidades de tipo Núcleo

```
- Autor: Daniel Bacaicoa Barber (Sep 21)
```

Son necesarios los siguientes paquetes

- NumPy
- SciPy
- SymPy
- MatplotLib

In [1]:
```
import numpy as np
import scipy
import matplotlib.pyplot as plt
import sympy

# Es necesario importar ciertos paquetes de SciPy
from scipy import stats, integrate, signal, interpolate, misc, special

# Paquetes para los gráficos
from matplotlib import cm
from mpl_toolkits.mplot3d import Axes3D
```

Funciones de tipo núcleo soportadas

Aunque programaremos también la familia beta para mostrarla, realizaremos los cálculos con el núcleo Gaussiano.

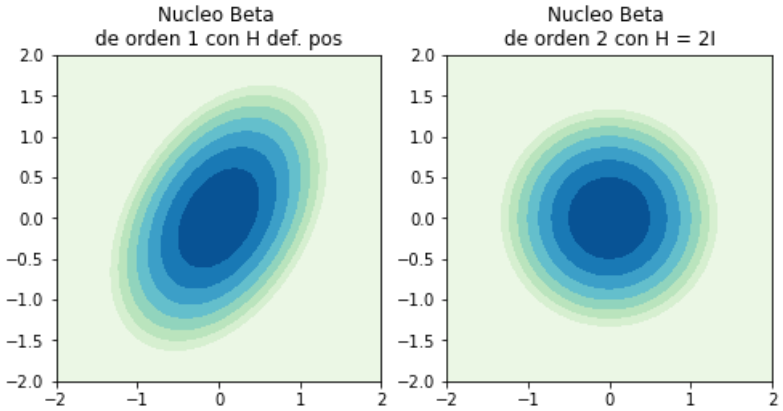
Familia beta (Duong, 2015)

- $K_H(x) = \beta_H(x,r) = \frac{c_r}{|H|^{\frac{1}{2}}}\left(1 - x^\intercal H^{-1}x\right)^r 1_{\{x^\intercal H^{-1}x \leq 1\}}$ where $c_r = \frac{2}{dv_d B(r+1,d/2)}$
- $K_H(x) = \phi_H(x) = \frac{1}{\sqrt{2\pi}}|H|^{-1/2}e^{-\frac{x^\intercal H^{-1}x}{2}}$

In [2]:
```
def Spherical_beta(p, r = 1, H = np.eye(2)):
    '''
    Función que evalua el kernel beta esférico
    Input:
        p: puntos donde evaluar la función
        r: orden del núcleo
        H: matriz de ancho de banda
    Output
        x: valor del núcleo en p
    '''
    if len(p.shape) == 1:
        # We have to convert the list (one-dim array) into a vector (two-dim array)
        p = p.reshape(-1,1)

    d = p.shape[1]
    p = p.T

    v_d = (np.pi**(d/2))/special.gamma(d/2 + 1)
    c_r = 2/(d*v_d*special.beta(r+1,d/2))
    detH = np.linalg.det(H)
    invH = np.linalg.inv(H)

    return  c_r * np.diag(1 - (p.T@invH@p))**r * (np.diag((p.T@invH@p))<=1)
```

In [3]:

```
n_grid = 100
x = np.linspace(-2,2,n_grid)
y = np.linspace(-2,2,n_grid)
X,Y = np.meshgrid(x,y)
Sigma = np.array([[2,1],[1,3]])

pos = np.array([X.flatten(),Y.flatten()]).T
Z1 = Spherical_beta(pos, r = 1, H = Sigma)
Z2 = Spherical_beta(pos, r = 1, H = 2*np.eye(2))



fig = plt.figure(figsize=[8,4])
ax1 = plt.subplot(1,2,1)
ax1.set_title('Nucleo Beta \nde orden 1 con H def. pos')
ax1.contourf(X,Y,Z1.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
ax1.set_aspect('equal', adjustable='box')


ax2 = plt.subplot(1,2,2)
ax2.set_title('Nucleo Beta \nde orden 2 con H = 2I')
ax2.contourf(X,Y,Z2.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
ax2.set_aspect('equal', adjustable='box')
```



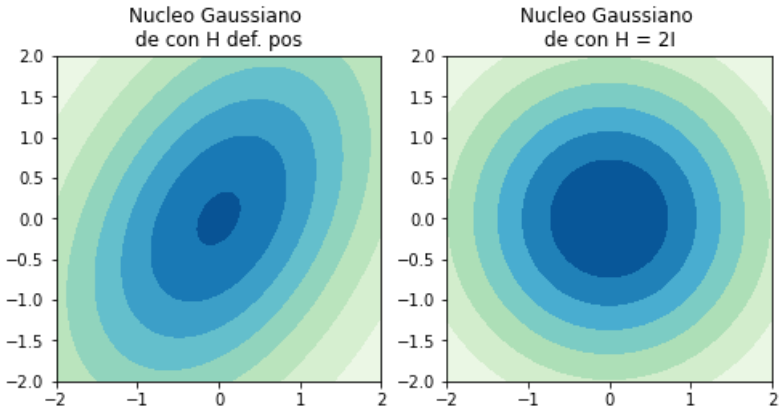
In [4]:

```
def Mv_norm(p, H = np.eye(2), r = None):
    '''
    Función que evalua el kernel Gaussiano
    Input:
        p: puntos donde evaluar la función
        H: matriz de ancho de banda
        r: Por si en alguna aplicación se le pasa r de beta, no tiene efecto
    Output
        x: valor del núcleo en p
    '''
    return stats.multivariate_normal.pdf(p, mean = np.zeros(2), cov = H)
```

In [5]:

```
n_grid = 100
x = np.linspace(-2,2,n_grid)
y = np.linspace(-2,2,n_grid)
X,Y = np.meshgrid(x,y)
Sigma = np.array([[2,1],[1,3]])

pos = np.array([X.flatten(),Y.flatten()]).T
Z1 = Mv_norm(pos, r = 1, H = Sigma)
Z2 = Mv_norm(pos, r = 1, H = 2*np.eye(2))



fig = plt.figure(figsize=[8,4])
ax1 = plt.subplot(1,2,1)
ax1.set_title('Nucleo Gaussiano \nde con H def. pos')
```

```
ax1.contourf(X,Y,Z1.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
ax1.set_aspect('equal', adjustable='box')


ax2 = plt.subplot(1,2,2)
ax2.set_title('Nucleo Gaussiano \nde con H = 2I')
ax2.contourf(X,Y,Z2.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
ax2.set_aspect('equal', adjustable='box')
```



Funciones Auxiliares

Definimos algunas funciones que serán necesarias. Por ejemplo, la derivada de los núcleos

Nótese que utilizaremos el paquete de cálculo simbolico SymPy.

In [6]:
```
def Kron_invsqrt(M, r = 2, inv_sqrt = True):
    '''
    Funcion que devuelve el prod. de kronecker r veces de una matriz elevada a (-1/2)
    Input:
        M: Matriz
        r: número de veces que se realiza el producto
        inv_sqrt: Si es False no se eleva la matriz a (-1/2)
    Output
        Mr: Matriz producto de tamaño 2^rx2^r
    '''
    if inv_sqrt == True:
        Mn = scipy.linalg.sqrtm(np.linalg.inv(M))
    else:
        Mn = M
    M_new = Mn
    for i in range(r-1):
        M_new = np.kron(M_new,Mn)
    return M_new
Z = np.array([[2,1],[1,3]])
print(Kron_invsqrt(Z,r=2))
print(Kron_invsqrt(np.eye(2),r=2))
```

```
[[ 0.57888544 -0.11055728 -0.11055728  0.02111456]
 [-0.11055728  0.46832816  0.02111456 -0.08944272]
 [-0.11055728  0.02111456  0.46832816 -0.08944272]
 [ 0.02111456 -0.08944272 -0.08944272  0.37888544]]
[[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]
```

In [7]:
```
def normal_derivative(p, H = np.eye(2), order = 1):
    '''
    Calcula la r-esima derivada de la función de densidad normal
    Input:
        p: Valores en los que evaluar la derivada
        H: Matriz de ancho de banda
        order: Drivada (hasta 4)
```

```
    Output
        D^{\otimes r}: vector con las derivadas por cada punto
    '''
    from sympy import lambdify
    sx, sy = sympy.symbols('sx sy')

    expr = 1/(2*sympy.pi) * sympy.exp(-1/2 * (sx**2+sy**2))
    if order == 1:
        DF = sympy.Matrix([[sympy.diff(expr,sx),
                            sympy.diff(expr,sy)]])
    elif order == 2:
        DF = sympy.Matrix([[sympy.diff(expr,sx,sx),
                            sympy.diff(expr,sx,sy),
                            sympy.diff(expr,sy,sx),
                            sympy.diff(expr,sy,sy)]])
    elif order == 3:
        DF = sympy.Matrix([[sympy.diff(expr,sx,sx,sx),
                            sympy.diff(expr,sx,sx,sy),
                            sympy.diff(expr,sx,sy,sx),
                            sympy.diff(expr,sx,sy,sy),
                            sympy.diff(expr,sy,sx,sx),
                            sympy.diff(expr,sy,sx,sy),
                            sympy.diff(expr,sy,sy,sx),
                            sympy.diff(expr,sy,sy,sy)]])
    elif order == 4:
        DF = sympy.Matrix([[sympy.diff(expr,sx,sx,sx,sx),
                            sympy.diff(expr,sx,sx,sx,sy),
                            sympy.diff(expr,sx,sx,sy,sx),
                            sympy.diff(expr,sx,sx,sy,sy),
                            sympy.diff(expr,sx,sy,sx,sx),
                            sympy.diff(expr,sx,sy,sx,sy),
                            sympy.diff(expr,sx,sy,sy,sx),
                            sympy.diff(expr,sx,sy,sy,sy),
                            sympy.diff(expr,sy,sx,sx,sx),
                            sympy.diff(expr,sy,sx,sx,sy),
                            sympy.diff(expr,sy,sx,sy,sx),
                            sympy.diff(expr,sy,sx,sy,sy),
                            sympy.diff(expr,sy,sy,sx,sx),
                            sympy.diff(expr,sy,sy,sx,sy),
                            sympy.diff(expr,sy,sy,sy,sx),
                            sympy.diff(expr,sy,sy,sy,sy)]])
    s = (sx, sy)
    DF_func = lambdify(s, DF , modules='numpy')

    det = np.linalg.det(H)**(-0.5)
    Hr = Kron_invsqrt(H,r = order)
    p = p @ Kron_invsqrt(H,r = 0)
    if len(p.shape) == 1:

        return (det*Hr@DF_func(p[0],p[1]).T).T
    else:
        return (det*Hr@np.array([DF_func(pos[0],pos[1]) for pos in p]).squeeze().T).T
normal_derivative(pos,order = 1)
```

Out[7]:
```
array([[ 0.00583005,  0.00583005],
       [ 0.00618798,  0.00631557],
       [ 0.00655438,  0.00683035],
       ...,
       [-0.00655438, -0.00683035],
       [-0.00618798, -0.00631557],
       [-0.00583005, -0.00583005]])
```

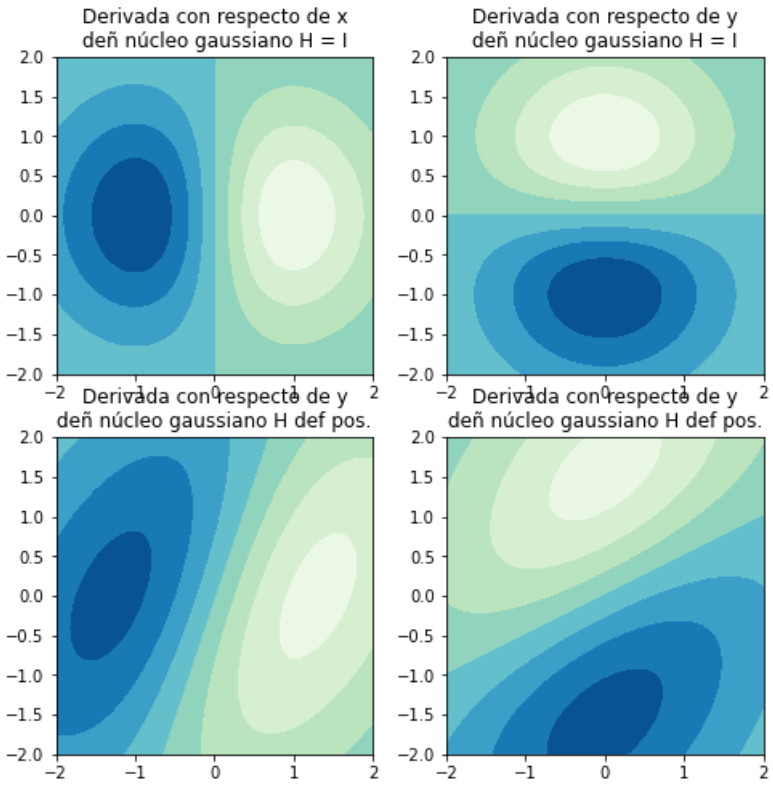
In [8]:
```
# Vemos que podemos reestructurar la salida D^2 en un tensor (100,100, 4)
# donde para cada punto tenemos vec(HK).T
normal_derivative(pos,order = 2).reshape(n_grid,n_grid,-1)[0,0,:]
```

Out[8]:
```
array([0.00874507, 0.0116601 , 0.0116601 , 0.00874507])
```

In [9]:
```
# Vemos que tambien podemos reestructurar la salida D^2 por ejemplo
# en un tensor (nº puntos, 2, 2) donde para cada punto tenemos la Hessiana
normal_derivative(pos,order = 2).reshape(10000,2,2)[0,:,:]
```

Out[9]:
```
array([[0.00874507, 0.0116601 ],
       [0.0116601 , 0.00874507]])
```

In [10]:
```
n_grid = 100
x = np.linspace(-2,2,n_grid)
y = np.linspace(-2,2,n_grid)
X,Y = np.meshgrid(x,y)
Sigma = np.array([[2,1],[1,3]])

pos = np.array([X.flatten(),Y.flatten()]).T
Z1 = normal_derivative(pos,order = 1, H=np.eye(2))[:,0]
Z2 = normal_derivative(pos,order = 1, H=np.eye(2))[:,1]
Z3 = normal_derivative(pos,order = 1, H=Sigma)[:,0]
Z4 = normal_derivative(pos,order = 1, H=Sigma)[:,1]

fig = plt.figure(figsize=[8,8])
ax1 = plt.subplot(2,2,1)
ax1.set_title('Derivada con respecto de x\ndeñ núcleo gaussiano H = I')
ax1.contourf(X,Y,Z1.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
ax1.set_aspect('equal', adjustable='box')

ax2 = plt.subplot(2,2,2)
ax2.set_title('Derivada con respecto de y\ndeñ núcleo gaussiano H = I')
ax2.contourf(X,Y,Z2.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
ax2.set_aspect('equal', adjustable='box')

ax3 = plt.subplot(2,2,3)
ax3.set_title('Derivada con respecto de y\ndeñ núcleo gaussiano H def pos.')
ax3.contourf(X,Y,Z3.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
ax3.set_aspect('equal', adjustable='box')

ax4 = plt.subplot(2,2,4)
ax4.set_title('Derivada con respecto de y\ndeñ núcleo gaussiano H def pos.')
ax4.contourf(X,Y,Z4.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
ax4.set_aspect('equal', adjustable='box')

plt.show()
```



Calculamos tambien

- $R(K) = R(\phi_I) = \int_{\mathbb{R}^2} \phi(x)^2 dx$
- $\mu_2(K) = \int_{\mathbb{R}^2} x_1^2 \phi(x)^2 dx$

In [11]:
```
# R(K) para K = N(0,I) y mu_2(K)
RN = integrate.dblquad(lambda y, x: Mv_norm([x,y])**2,-np.inf,np.inf, lambda x: -np.inf, lambda x
print(RN)
mu_2N = integrate.dblquad(lambda y, x: x**2*Mv_norm([x,y])**2,-np.inf,np.inf, lambda x: -np.inf,
print(mu_2N)
```

```
0.07957747154589243
0.039788735772780616
```

In [12]:
```
def grid(sample):
    '''
    Genera un vector con todas las restas del tipo (X_i-X_j) for i = 1,..n; j = 1,...,n, j!=i
    Input:
        muestra
    Output
        List con todas las diferencias salvo las de un punto consigo mismo
    '''
    n,d = sample.shape
    grid = np.array(n)
    for i in range(n):
        for j in range(n):
            if j!=i:
                if i==0 and j==1:
                    grid = np.array(sample[i,:]-sample[j,:])
                else:
                    grid = np.vstack((grid,np.array(sample[i,:]-sample[j,:])))
    return grid
```

In [13]:
```
example_grid = grid(np.array([[1,1],[1,0],[0,1]]))
print(example_grid.shape)
print(example_grid)
```

```
(6, 2)
[[ 0  1]
 [ 1  0]
 [ 0 -1]
 [ 1 -1]
 [-1  0]
 [-1  1]]
```

In [14]:
```
def sum_sum_conv(muestra, H, order = 4):
    '''
    Devuelve el kernel evaluado en todas las diferencias de los puntos de la muestra
    Tenemos en cuenta que si queremos la convilución simplemente sera K_{2H}
    Input:
        muestra
        H: La matriz de ancho de banda
        orden
    Output:
        el vectro suma de las derivadas
    '''
    n,_ = muestra.shape
    eval_grid = grid(muestra)
    if order == 0:
        return np.sum(Mv_norm(eval_grid, H = H),axis=0)/(n*(n-1))
    else:
        return np.sum(normal_derivative(eval_grid, order = order, H = H),axis=0)/(n*(n-1))
```

Función de estimación de la densidad

In [15]:
```
def mv_kde(sample, H):
    '''
```

```
    Función para calcular la densidad estimada
    Input:
        sample: la muestra
        H: La matriz ancho de banda
    Output:
        x: la malla de evaluación
        kde: el valor de la densidad
    '''
    n,_ = sample.shape

    n_grid = 100
    x = np.linspace(np.min(sample[:,0]-.5),np.max(sample[:,0]+.5),n_grid)
    y = np.linspace(np.min(sample[:,1]-.5),np.max(sample[:,1]+.5),n_grid)
    X,Y = np.meshgrid(x,y)


    pos = np.array([X.flatten(),Y.flatten()]).T
    return np.meshgrid(x,y), np.sum([Mv_norm(pos-k, H = H) for k in sample],0)/n
```

Métodos de selección de matriz de ancho de banda óptima

Unbiased cross validation

In [16]:

```
def Selector_UCV(sample):
    n,_ = sample.shape
    eval_grid = grid(sample)

    h_grid = np.linspace(0.1,3,11)
    rho_grid = np.linspace(-1,1,11)

    #Como punto inicial ponemos la identitad
    H_ast = np.eye(2)
    UCV = RN/n + sum_sum_conv(sample, H = 2*np.eye(2), order = 0)
    UCV -= 2*sum_sum_conv(sample, H = np.eye(2), order = 0)

    ucv_list = []

    for sigma_1 in h_grid:
        for sigma_2 in h_grid:
            for rho in rho_grid:
                A = np.array([[sigma_1, rho],
                              [rho, sigma_2]])
                H = A @ A.T
                det = np.linalg.det(H)**(-0.5)
                ucv_h = det*RN/n + sum_sum_conv(sample, H = 2*H, order = 0)
                ucv_h -= 2*sum_sum_conv(sample, H = H, order = 0)
                ucv_list.append(ucv_h)
                if ucv_h < UCV:
                    UCV = ucv_h
                    H_ast = H
    return H_ast, UCV, ucv_list
```

Biased cross validation

In [17]:

```
def Selector_BCV(sample):
    n,_ = sample.shape
    eval_grid = grid(sample)

    h_grid = np.linspace(0.1,3,11)
    rho_grid = np.linspace(-1,1,11)

    #Como punto inicial ponemos la identitad
    H_ast = np.eye(2)
    vec_H = H_ast.reshape((-1, 1), order="F")
    BCV = RN/n
    BCV += .25 * mu_2N**2 * vec_H.T @ sum_sum_conv(sample, H = np.eye(2), order = 4).reshape(4,4
```

```
    bcv_list = []

    for sigma_1 in h_grid:
```

```
        for sigma_2 in h_grid:
            for rho in rho_grid:
                A = np.array([[sigma_1, rho],
                              [rho, sigma_2]])
                H = A @ A.T
                det = np.linalg.det(H)**(-0.5)
                vec_H = H.reshape((-1, 1), order="F")

                bcv_h = det*RN/n
                bcv_h += .25 * mu_2N**2 * vec_H.T @ sum_sum_conv(sample, H = H, order = 4).resha
                bcv_list.append(bcv_h)
                if bcv_h < BCV:
                    BCV = bcv_h
                    H_ast = H
    return H_ast, BCV, bcv_list
```

Plug-in de una etapa

In [18]:

```
def Selector_PI(sample):
    n,_ = sample.shape
    # Para estimar la matriz de covarianzas
    Sigma = np.cov(sample.T)

    h_grid = np.linspace(0.1,3,11)
    rho_grid = np.linspace(-1,1,11)

    g_4 = (1/3)**(1/4)*2*Sigma*n**(-1/4)
    psi_4 = normal_derivative(np.zeros(2), H = g_4, order = 4).reshape(-1,1)
    psi_4 += sum_sum_conv(sample, H = g_4, order = 4).reshape(-1,1)


    H_NS = Sigma * n**(-0.5)
    vec_H_NS = H_NS.reshape((-1, 1), order="F")
    H_ast = H_NS

    PI = np.linalg.det(H_NS)**(-0.5) * RN/n
    PI += 0.25 * mu_2N**2 * psi_4.T @ np.kron(vec_H_NS,vec_H_NS)

    pi_list = []

    for sigma_1 in h_grid:
        for sigma_2 in h_grid:
            for rho in rho_grid:
                A = np.array([[sigma_1, rho],
                              [rho, sigma_2]])
                H = A @ A.T

                det = np.linalg.det(H)**(-0.5)
                vec_H = H.reshape((-1, 1), order="F")
                pi_h =  det * RN/n
                pi_h += 0.25 * mu_2N**2 * psi_4.T @ np.kron(vec_H,vec_H)

                pi_list.append(pi_h)
                if pi_h < PI:
                    PI = pi_h
                    H_ast = H
    return H_ast, PI, pi_list
```

Generamos un conjunto de datos sintético como mixtura de 3 normales

In [19]:

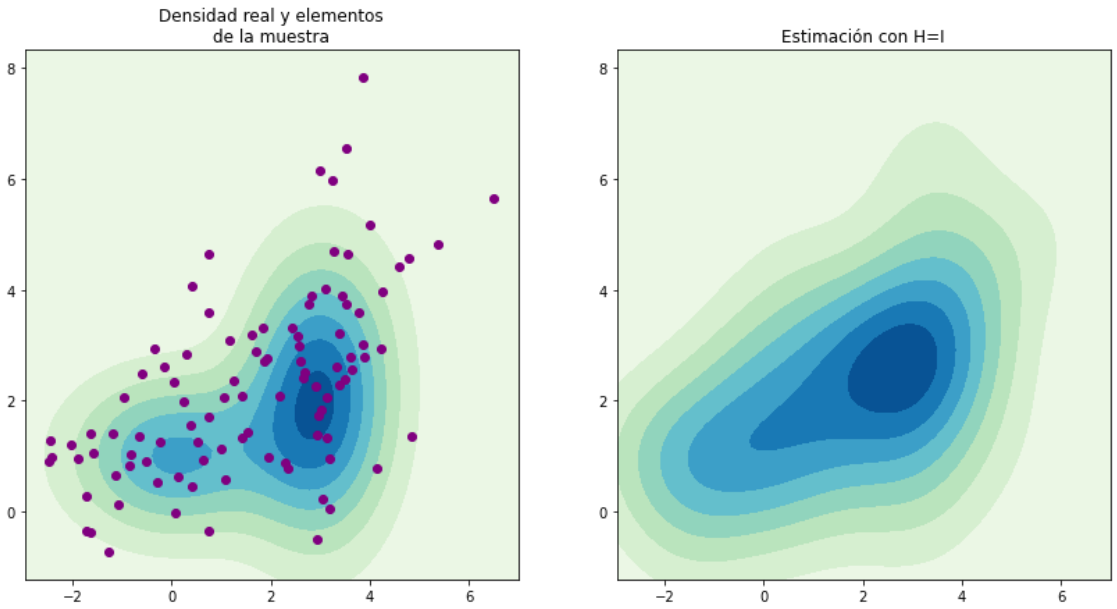
```
def normal_mixture(n, p, mu_1, sigma_1, mu_2, sigma_2, mu_3, sigma_3):
    # The sample generated
    binom = stats.binom.rvs(2, p, size=100)

    N_1 = stats.multivariate_normal.rvs(mu_1,sigma_1,size = n)
    N_2 = stats.multivariate_normal.rvs(mu_2,sigma_2,size = n)
    N_3 = stats.multivariate_normal.rvs(mu_3,sigma_3,size = n)

    sample = np.vstack((binom==0,binom==0)).T * N_1
    sample+= np.vstack((binom==1,binom==1)).T * N_2
```

```
    sample+= np.vstack((binom==2,binom==2)).T * N_3
    return sample

def real_normal_mixture(x, p, mu_1, sigma_1, mu_2, sigma_2, mu_3, sigma_3):
    # The real underlying density or its derivatives
    N_1 = stats.multivariate_normal.pdf(x, mu_1, sigma_1)
    N_2 = stats.multivariate_normal.pdf(x, mu_2, sigma_2)
    N_3 = stats.multivariate_normal.pdf(x, mu_3, sigma_3)
    density = p**2 * N_1 + 2*p*(1-p) * N_2 + (1-p)**2 * N_3
    return density
```

Estimación de la densidad con un Kernel Gaussiano

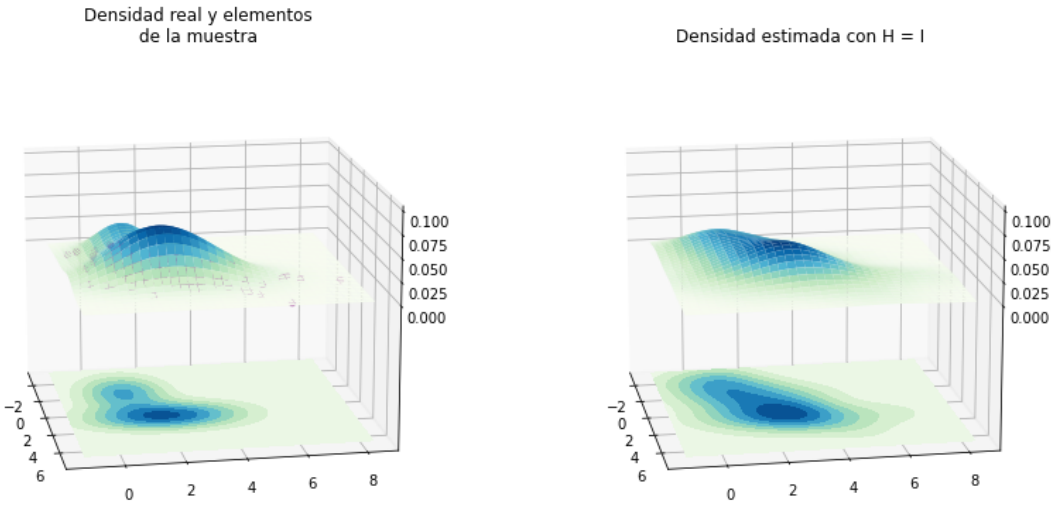
In [20]:

```
n = 100
p = 0.24

mu_1 = 3*np.ones(2); sigma_1 = np.array([[2,1],[1,3]])
mu_2 = np.array([0,1]); sigma_2 = np.diag([2,1])
mu_3 = np.array([3,2]); sigma_3 = np.diag([1,3])

np.random.seed(1234)
sample = normal_mixture(n,p, mu_1, sigma_1, mu_2, sigma_2, mu_3, sigma_3)

Q, kde = mv_kde(sample, H=np.eye(2))
X,Y = Q
n_grid = Q[0].shape[0]


pos = np.array([X.flatten(),Y.flatten()]).T

f = real_normal_mixture(pos, p, mu_1, sigma_1, mu_2, sigma_2, mu_3, sigma_3)

fig = plt.figure(figsize=(14,7))
ax1 = fig.add_subplot(1,2,1)
ax1.contourf(X,Y,f.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
for i in range(n):
    ax1.scatter(sample[i,0],sample[i,1],color = 'purple')
ax1.set_title('Densidad real y elementos\nde la muestra')


ax2 = plt.subplot(1,2,2)
ax2.contourf(X,Y,kde.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)

ax2.set_title('Estimación con H=I')
plt.show()
```



In [21]:

```
fig = plt.figure(figsize=[14,7])
ax1 = fig.add_subplot(1, 2, 1, projection='3d')
ax1.plot_surface(X, Y, f.reshape(n_grid,n_grid), rstride=3, cstride=3, linewidth=1, antialiased=
                 cmap=cm.GnBu)

cset = ax1.contourf(X, Y, f.reshape(n_grid,n_grid), zdir='z', offset=-0.15, cmap=cm.GnBu)
for i in range(n):
    ax1.scatter(sample[i,0],sample[i,1],color = 'purple', alpha=0.5)

ax1.set_zlim(-0.15,0.1)
ax1.set_zticks(np.linspace(0,0.1,5))
ax1.view_init(15, -10)
ax1.set_title('Densidad real y elementos\nde la muestra')

ax1 = fig.add_subplot(1, 2, 2, projection='3d')
ax1.plot_surface(X, Y, kde.reshape(n_grid,n_grid), rstride=3, cstride=3, linewidth=1, antialiase
                 cmap=cm.GnBu)

cset = ax1.contourf(X, Y, kde.reshape(n_grid,n_grid), zdir='z', offset=-0.15, cmap=cm.GnBu)

ax1.set_zlim(-0.15,0.1)
ax1.set_zticks(np.linspace(0,0.1,5))
ax1.view_init(15, -10)
ax1.set_title('Densidad estimada con H = I')

plt.show()
```



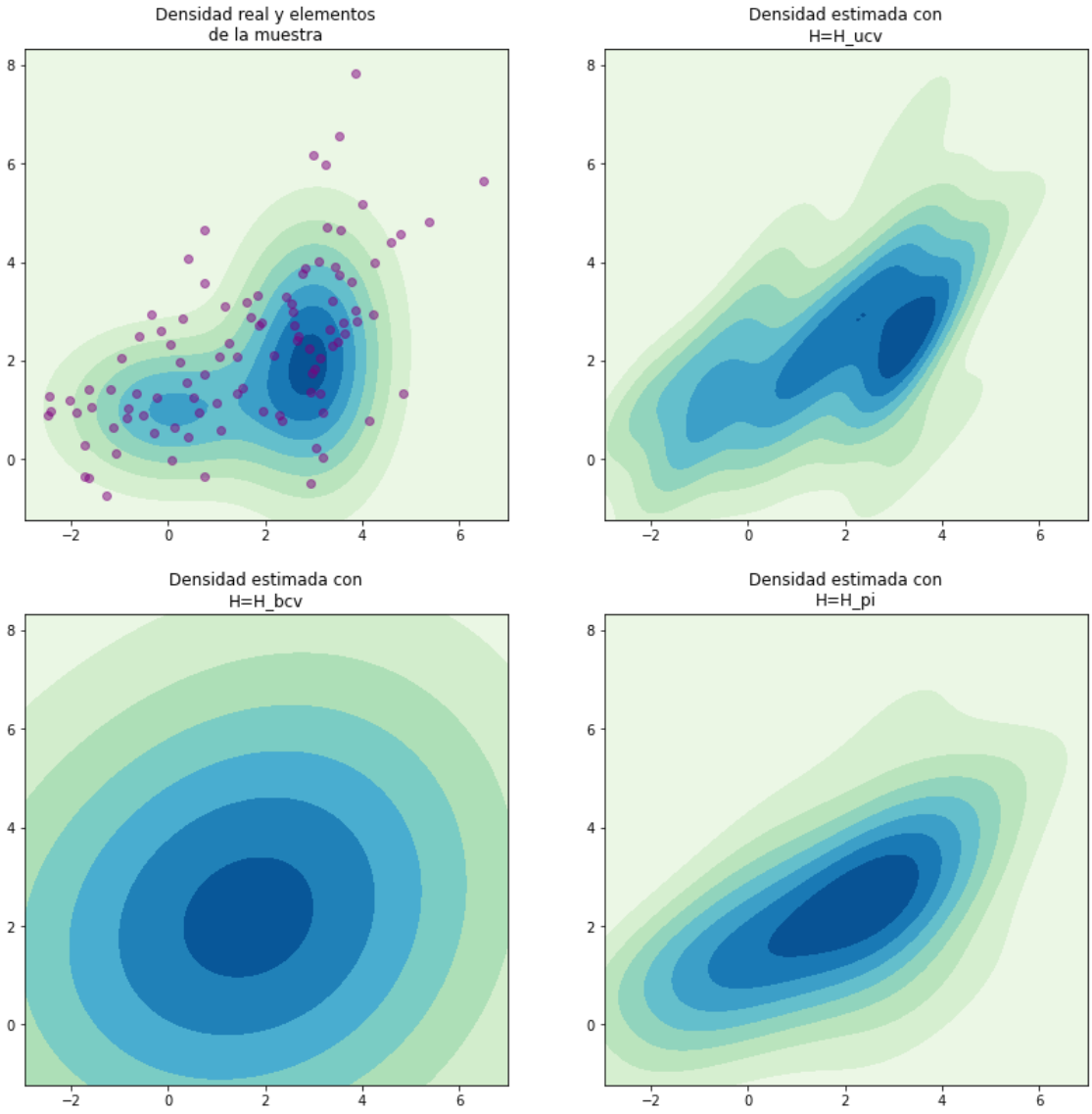
Y con los selectores presentados anteriormente

In [22]:

```
H_pi, _, _ = Selector_PI(sample)
print('The optimal bandwidth with PI is:\n',H_pi)
H_ucv, _, _ = Selector_UCV(sample)
print('\nThe optimal bandwidth with UCV is:\n',H_ucv)
H_bcv, _, _ = Selector_BCV(sample)
print('\nThe optimal bandwidth with BCV is:\n',H_bcv)
```

```
The optimal bandwidth with PI is:
 [[1.7476 0.892 ]
 [0.892  1.1009]]

The optimal bandwidth with UCV is:
 [[0.6224 0.66  ]
 [0.66   1.1009]]

The optimal bandwidth with BCV is:
 [[9. 0.]
 [0. 9.]]
```

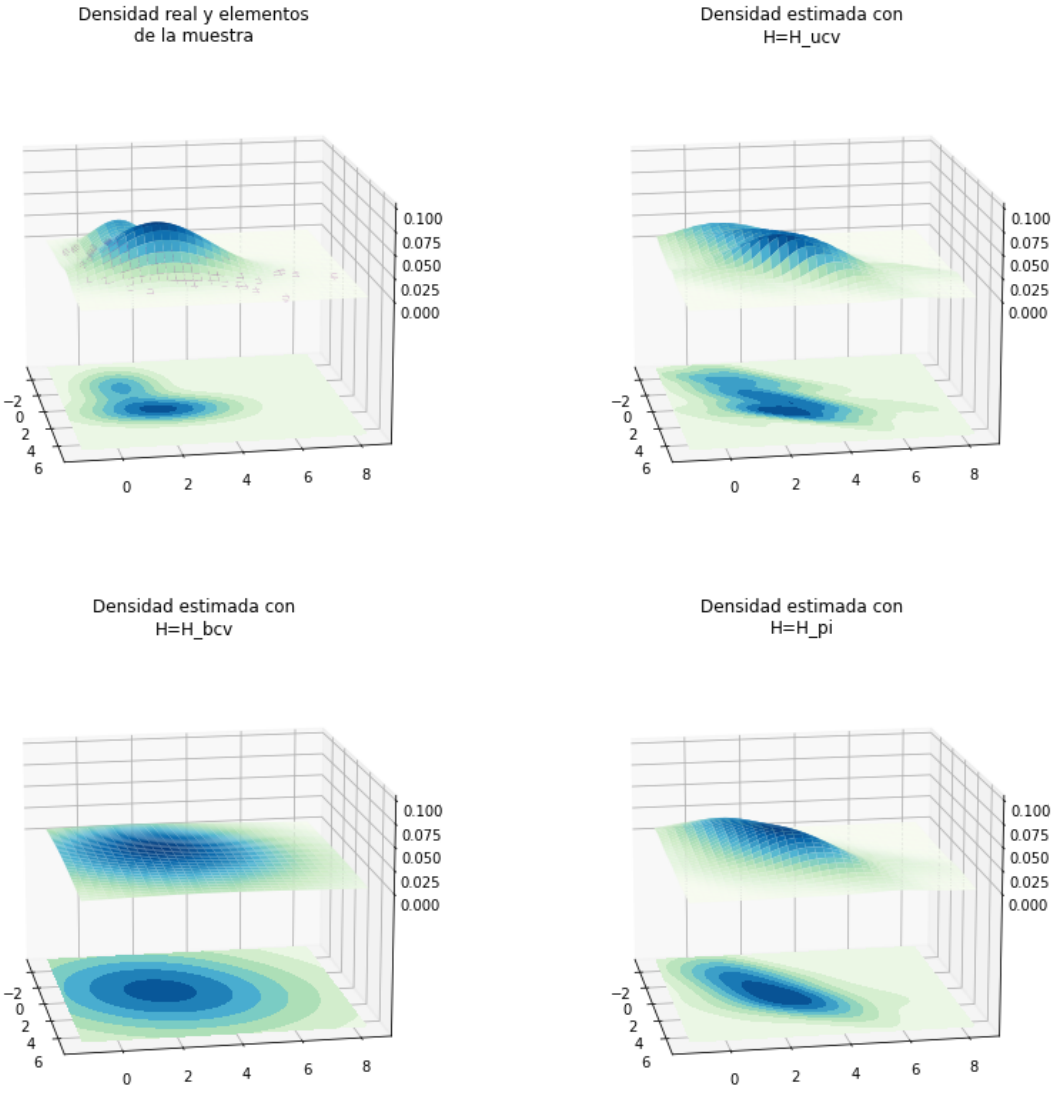
In [23]:
```
Q, kde_ucv = mv_kde(sample, H = H_ucv)
Q, kde_bcv = mv_kde(sample, H = H_bcv)
Q, kde_pi = mv_kde(sample, H = H_pi)
X,Y = Q
n_grid = Q[0].shape[0]
pos = np.array([X.flatten(),Y.flatten()]).T

f = real_normal_mixture(pos, p, mu_1, sigma_1, mu_2, sigma_2, mu_3, sigma_3)

fig = plt.figure(figsize=(14,14))
ax1 = fig.add_subplot(2,2,1)
ax1.contourf(X,Y,f.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
for i in range(n):
    ax1.scatter(sample[i,0],sample[i,1],color = 'purple',alpha=0.5)
ax1.set_title('Densidad real y elementos\nde la muestra')


ax2 = plt.subplot(2,2,2)
ax2.contourf(X,Y,kde_ucv.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
ax2.set_title('Densidad estimada con\nH=H_ucv')
ax3 = plt.subplot(2,2,3)
ax3.contourf(X,Y,kde_bcv.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
ax3.set_title('Densidad estimada con\nH=H_bcv')
ax4 = plt.subplot(2,2,4)
ax4.contourf(X,Y,kde_pi.reshape(n_grid,n_grid),cmap=plt.cm.GnBu)
ax4.set_title('Densidad estimada con\nH=H_pi')


plt.show()
```



In [24]:

```
Q, kde_ucv = mv_kde(sample, H = H_ucv)
Q, kde_bcv = mv_kde(sample, H = H_bcv)
Q, kde_pi = mv_kde(sample, H = H_pi)
X,Y = Q
n_grid = Q[0].shape[0]
pos = np.array([X.flatten(),Y.flatten()]).T

f = real_normal_mixture(pos, p, mu_1, sigma_1, mu_2, sigma_2, mu_3, sigma_3)

fig = plt.figure(figsize=(14,14))
ax1 = fig.add_subplot(2,2,1, projection='3d')
ax1.plot_surface(X, Y, f.reshape(n_grid,n_grid), rstride=3, cstride=3, linewidth=1, antialiased=T
                 cmap=cm.GnBu)
cset = ax1.contourf(X, Y, f.reshape(n_grid,n_grid), zdir='z', offset=-0.15, cmap=cm.GnBu)
for i in range(n):
    ax1.scatter(sample[i,0],sample[i,1],color = 'purple',alpha=0.5)
ax1.set_title('Densidad real y elementos\nde la muestra')


ax2 = plt.subplot(2,2,2, projection='3d')
ax2.plot_surface(X, Y, kde_ucv.reshape(n_grid,n_grid), rstride=3, cstride=3, linewidth=1, antiali
                 cmap=cm.GnBu)
cset = ax2.contourf(X, Y, kde_ucv.reshape(n_grid,n_grid), zdir='z', offset=-0.15, cmap=cm.GnBu)
ax2.set_title('Densidad estimada con\nH=H_ucv')
ax3 = plt.subplot(2,2,3, projection='3d')
ax3.plot_surface(X, Y, kde_bcv.reshape(n_grid,n_grid), rstride=3, cstride=3, linewidth=1, antiali
                 cmap=cm.GnBu)
```

```
cset = ax3.contourf(X, Y, kde_bcv.reshape(n_grid,n_grid), zdir='z', offset=-0.15, cmap=cm.GnBu)
ax3.set_title('Densidad estimada con\nH=H_bcv')
ax4 = plt.subplot(2,2,4, projection='3d')
ax4.plot_surface(X, Y, kde_pi.reshape(n_grid,n_grid), rstride=3, cstride=3, linewidth=1, antialia
                 cmap=cm.GnBu)
cset = ax4.contourf(X, Y, kde_pi.reshape(n_grid,n_grid), zdir='z', offset=-0.15, cmap=cm.GnBu)
ax4.set_title('Densidad estimada con\nH=H_pi')
ax1.set_zlim(-0.15,0.1)
ax1.set_zticks(np.linspace(0,0.1,5))
ax1.view_init(15, -10)
ax2.set_zlim(-0.15,0.1)
ax2.set_zticks(np.linspace(0,0.1,5))
ax2.view_init(15, -10)
ax3.set_zlim(-0.15,0.1)
ax3.set_zticks(np.linspace(0,0.1,5))
ax3.view_init(15, -10)
ax4.set_zlim(-0.15,0.1)
ax4.set_zticks(np.linspace(0,0.1,5))
ax4.view_init(15, -10)

plt.show()
```



In [ ]: