RÉPUBLIQUE TUNISIENNE MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

Rapport de Projet Statistiques non paramétriques

Présenté par :

Salha MEDINI

Amna MEFTAH

Mayssa HEMDANA

Eya BESBES

Inscrites en 4éme année Cycle Ingénieur en Science des Données

---

Estimation non paramétrique de la densité

Supervisé par :

Dr Salah KHARDANI

Année universitaire 2022-2023

TABLE DES MATIÈRES

TABLE DES FIGURES

INTRODUCTION GÉNÉRALE

L'estimation de la densité non paramétrique est une technique couramment utilisée en statistique pour approximer la distribution d'une variable aléatoire continue.
Contrairement aux méthodes paramétriques, qui supposent une forme spécifique pour la distribution sous-jacente, les méthodes non paramétriques permettent une estimation plus flexible et adaptative de la densité, sans imposer de contraintes sur sa forme.
Ce rapport de projet se concentre sur l'utilisation de techniques non paramétriques pour estimer la densité f d'une variable aléatoire continue, ou comprendre la distribution des données observées est essentiel pour tirer des conclusions fiables et prendre des décisions considérables.

CHAPITRE 1

Introduction

Dans tout le chapitre, l'objectif sera d'estimer une densité f.
Pour cela, on s'appuiera sur un n-échantillon iid X $(X_1, X_2, \ldots, X_n)$, où chacune des variables X_i admet la densité f.

1.1 Quelques métriques

Dans cette section, nous allons définir quelques métriques que nous allons les utiliser dans la suite.

1.1.1 Les distances sur l'espace des fonctions

La distance L^∞ entre deux fonctions f et g est définie comme :

$$\|f-g\|_{L^\infty} \quad \sup_x |f(x)-g(x)|$$

La distance L^p entre deux fonctions f et g est définie comme :

$$\|f-g\|_{L^p} \quad \left(\int |f(x)-g(x)|^p \, dx\right)^{\frac{1}{p}}$$

La distance ponctuelle entre deux fonctions f et g au point x est définie comme :

$$d(f(x), g(x)) \quad |f(x)-g(x)|$$

1.1.2 La fonction de perte

La fonction de perte de $\mathbb{R}$ à $\mathbb{R}$+ est généralement définie comme suit :
w(x)=x² et elle est convexe.
Elle mesure la perte associée à l'estimation $\widetilde{f}(x)$ par rapport à la vraie fonction $f(x)$.

1.1.3 Le risque de l'estimateur

Les risques de l'estimateur $\widetilde{f}(x)$ obtenus par la méthode de noyau peuvent être définis comme suit :
Le risque quadratique est défini comme $R(\widetilde{f}) \quad \mathbb{E}[w(\widetilde{f}(x)-f(x))]$, telque w est la fonction de perte.

Il mesure l'erreur quadratique moyenne entre l'estimation $\widetilde{f}(x)$ et la vraie fonction $f(x)$ obtenue par la méthode de noyau.

1.1.4 Des définitions

1.1.4.1 Définition

Soit $(r_m)_n$ une suite et une constante C telles que

$$R(\widetilde{f}_n, F) \leq Cr_n$$

On dit que la suite d'estimateurs $(\widetilde{f}_n)_n$ atteint la vitesse (ou le taux) r_n sur la classe F (pour la distance d et la perte w).
Nous verrons que la vitesse sera d'autant plus grande que la classe F sera une classe de régularité élevée.

1.1.4.2 Définition

Si $B \in \mathbb{R}$, on note $[B]$ l'entier naturel qui soit le plus grand entier strictement inférieur à B.

1.1.4.3 Définition

Pour tout $B\ 0$ et tout $L\ 0$, on définit la classe de Hölder de régularité B et de rayon L par :

$$E(B,L)\ \{g : \mathbb{R} \to \mathbb{R} \mid g \text{ est } [B]\text{-fois dérivable et } \forall x,y \in \mathbb{R}, |g^{([B])}(x) - g^{([B])}(y)| \leq L|x-y|^{B-[B]}\}$$

Quand on intersecte $E(B,L)$ avec l'ensemble des densités, on note $E_x(B,L)$ cette intersection.

1.1.4.4 Les noyaux

Soit $K : \mathbb{R} \to \mathbb{R}$ intégrable et tel que $\int K(u)\,du\ 1$. Alors K est appelé noyau (kernel).

Exemples de noyaux :

- Noyau triangulaire : $K(u)\ (1-|u|)\mathbb{I}(-1\ u\ 1)$
- Noyau Biweight : $K(u)\ \frac{15}{16}(1-u^2)^2\mathbb{I}(-1\ u\ 1)$
- Noyau Gaussien : $K(u)\ \frac{1}{\sqrt{2\pi}}\exp(-\frac{u^2}{2})$

- Noyau d'Epanechnikov : $K(u)\ \ \frac{3}{4}(1-u^2)\mathbb{I}(-1\ \ u\ \ 1)$

1.2 Estimation non paramétrique de la densité

L'estimation classique de densité repose sur l'utilisation de modèles paramétriques, tels que les lois gaussiennes en dimension 1, et sur des connaissances a priori lorsque disponibles.
Cependant, cette approche présente des limitations en termes de représentation des données, d'absence de connaissances préalables et de risque d'interprétation erronée en cas de mauvais choix de modèle.
Les modèles non paramétriques offrent une alternative moins restrictive en ne faisant pas d'hypothèses spécifiques sur la forme de la densité. Cependant, si des connaissances fiables indiquent un modèle paramétrique approprié, celui-ci sera généralement préférable.
Le choix entre les deux dépend donc des connaissances a priori et de la proxiimité du modèle paramétrique avec la réalité.

1.2.1 Estimateur simple de la densité : l'histogramme

L'estimateur simple de la densité c'est l'histogramme. Il s'agit d'une méthode non paramétrique qui permet d'approximer la densité en discrétisant l'espace des valeurs.

L'idée principale derrière l'histogramme est de diviser l'intervalle des valeurs en plusieurs sous-intervalles de largeur égale, puis de compter le nombre d'observations se trouvant dans chaque sous-intervalle(partie de l'intervalle principale). La hauteur de chaque sous-intervalle représente une estimation de la densité dans cette partie de l'intervalle de l'espace des valeurs.

Supposons, pour simplifier, que nous soyons en dimension 1 et que les variables de l'échantillon soient des valeurs dans $(0,1]$, donc $f : (0,1] \to \mathbb{R}$. Nous nous donnons un découpage de $(0,1]$ en un certain nombre de classes $]a_r, a_{r1}), \ldots,]c_r, c_{rp})$. Pour simplifier encore, nous supposons que les classes ont la même longueur $a_{r1} - a_r\ \ h$. Cette longueur est notée h. Estimer f par la méthode de l'histogramme consiste simplement à estimer f par une fonction constante sur chaque classe, cette constante étant liée à la proportion de X_i

tombant dans cette classe. Plus précisément, on pose, pour tout $t \in]a_r, a_{r1})$:

$$\widetilde{f}(t) \quad \frac{1}{nh} \cdot \text{Card}\{i : X_i \in]a_j, a_{j1})\}$$

Pour voir d'où vient exactement cette formule, on a, si f est égale à une constante c sur $]a_r, a_{r1})$:

$$F(a_{j1}) - F(a_j) \quad \int_{a_j}^{a_{j1}} f(t)\,dt \quad c_j \cdot h$$

Ensuite, on approche la probabilité $F(a_{j1}) - F(a_{j1})$, qui correspond à la probabilité que $X_i \in]a_r, a_{r1})$, par la proportion de X_i se trouvant dans $]a_j, a_{j1}]$. On a alors :

$$\widetilde{C}_j \quad \frac{F(a_{j1}) - F(a_j)}{h} \quad \frac{\text{Card}\{i : X_i \in]a_j, a_{j1})\}}{nh}$$

La performance de cet estimateur dépend fortement du nombre de classes.

1.2.1.1 Code R et illustration graphique du choix du nombre de classes

```
hist(data, breaks = 10, main = "Histogramme avec 10 intervalles", freq = FALSE)
curve(dnorm(x, mean(data), sd(data)), add = TRUE, col = "blue", lwd = 2, lty = 2)
```

FIGURE 1.1 – Affichage des diagrammes et la courbe de densité

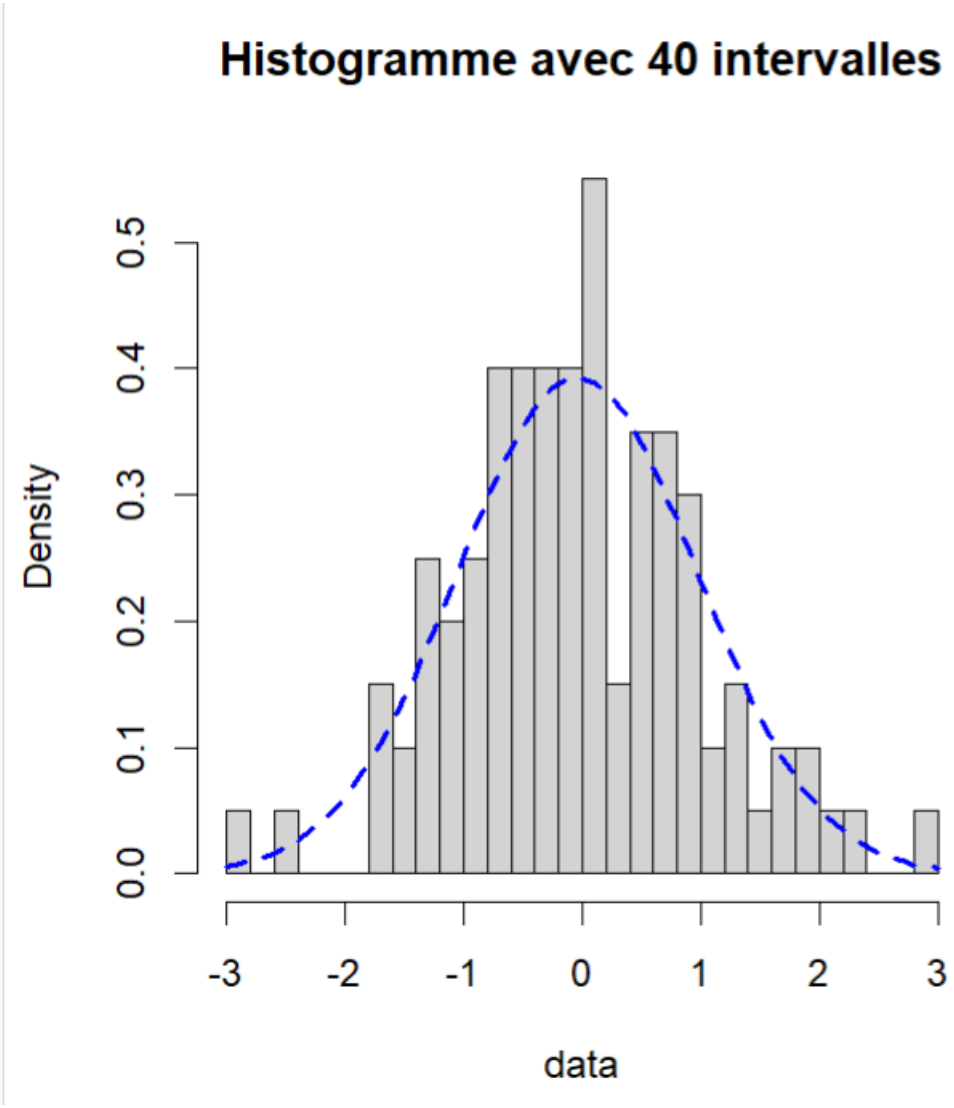

FIGURE 1.2 – Estimation de la densité avec 40 intervalles

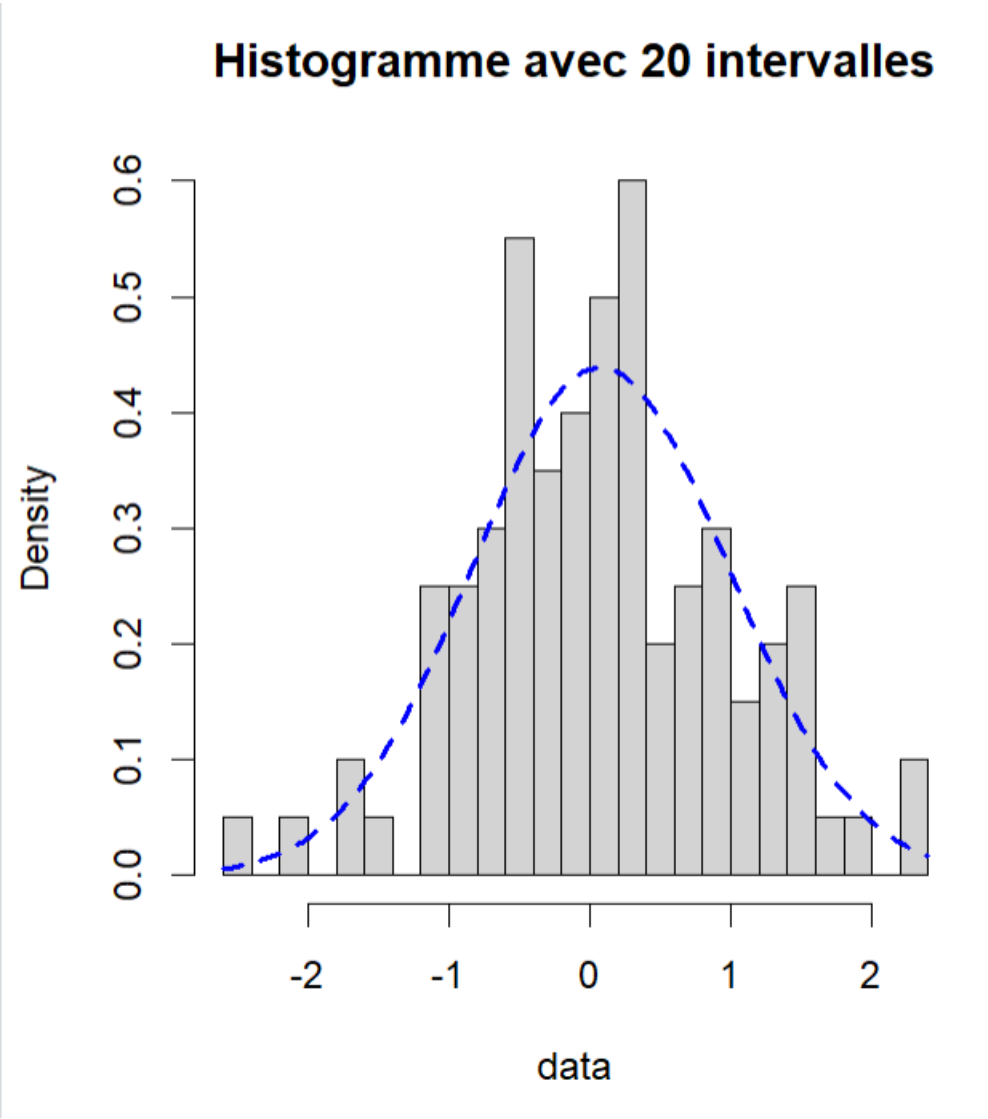

FIGURE 1.3 – Estimation de la densité avec 20 intervalles

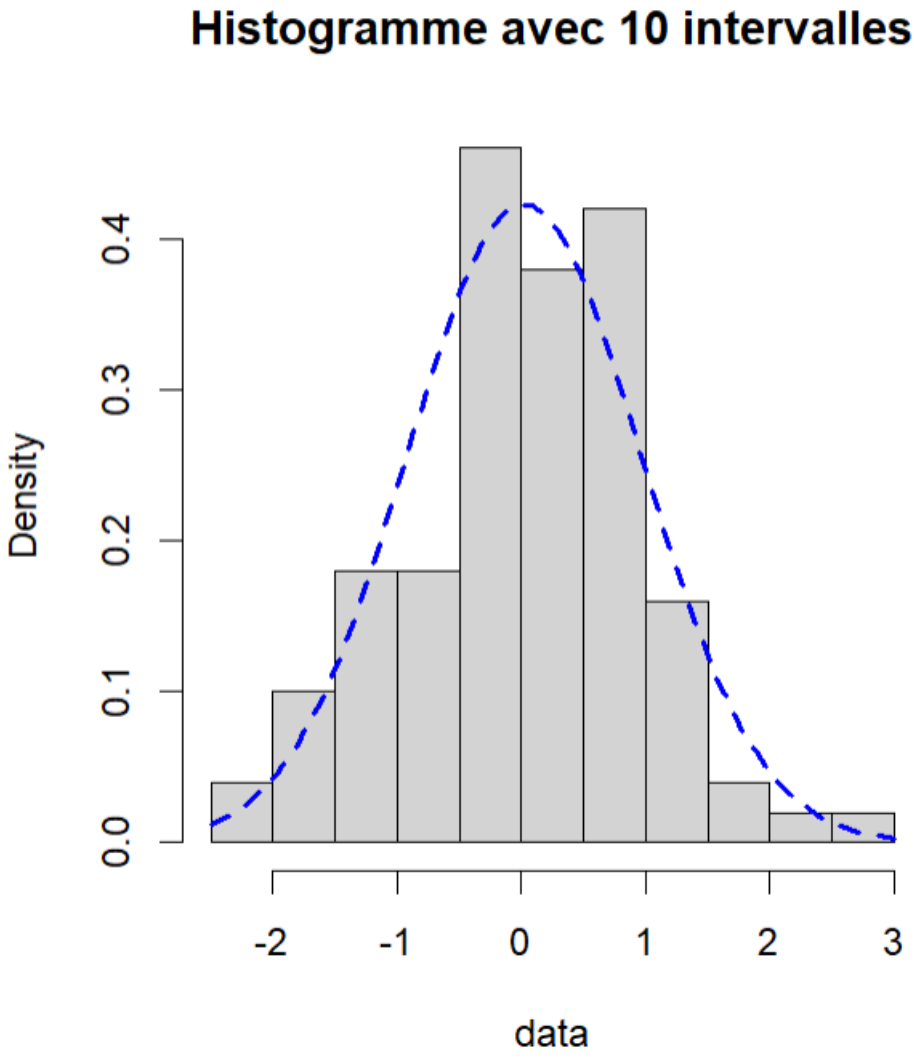

FIGURE 1.4 – Estimation de la densité avec 10 intervalles

1.2.1.2 Code Python et illustration graphique du choix du nombre de classes

Dans la suite nous allons essayer de construire un estimateur non paramétrique de la densité avec la méthode des histogrammes.

Les bibilthèques à utiliser

```
[128] from scipy.stats import norm
```

Génération de données aléatoires de taille 1000

```
[129] np.random.seed(0)
      data = np.random.randn(1000)
```

FIGURE 1.5 – Importation des bibilthèque et génération des données

Définition des paramètres de l'histogramme

```
density = True
```

Le nombre d'intervalles à tester

```
[131] num_bins_values = [10, 20, 30, 40]
```

Calcul de l'erreur de l'estimation

```
[137] def error__(density_real, histogram):
          interpolated_density_real = np.interp(histogram, np.linspace(0, 1, len(density_real)), density_real)
          return np.sum(np.abs(interpolated_density_real - histogram))
```

FIGURE 1.6 – Définition des paramétres et calcule de l'erreur

Boucler sur les intervalles

```
for num_bins in num_bins_values:
    # Creation de l'histogramme
    counts, bins, _ = plt.hist(data, bins=num_bins, density=density, edgecolor='black', alpha=0.7)

    # Calcul de la densité
    xmin, xmax = plt.xlim()
    x = np.linspace(xmin, xmax, 100)
    density_real = norm.pdf(x)

    # affichage de la courbe de densité
    plt.plot(x, density_real, 'r', label='Densité ')
    plt.xlabel('Valeurs')
    if density:
        plt.ylabel('Densité')
        plt.title(f'Estimation de densité par l\'histogramme (Densité) - {num_bins} intervalles')
    else:
        plt.ylabel('Nombre d\'observations')
        plt.title(f'Estimation de densité par l\'histogramme (Fréquence) - {num_bins} intervalles')

    # Calcul et affichage de l'erreur
    histogram = counts / np.sum(counts)
    error_ = error__(density_real, histogram)
    plt.text(0.05, 0.9, f'Erreur : {error_:.4f}', transform=plt.gca().transAxes)


    plt.legend()
    plt.show()
```

FIGURE 1.7 – Boucle sur les intervalles pour estimer la densité

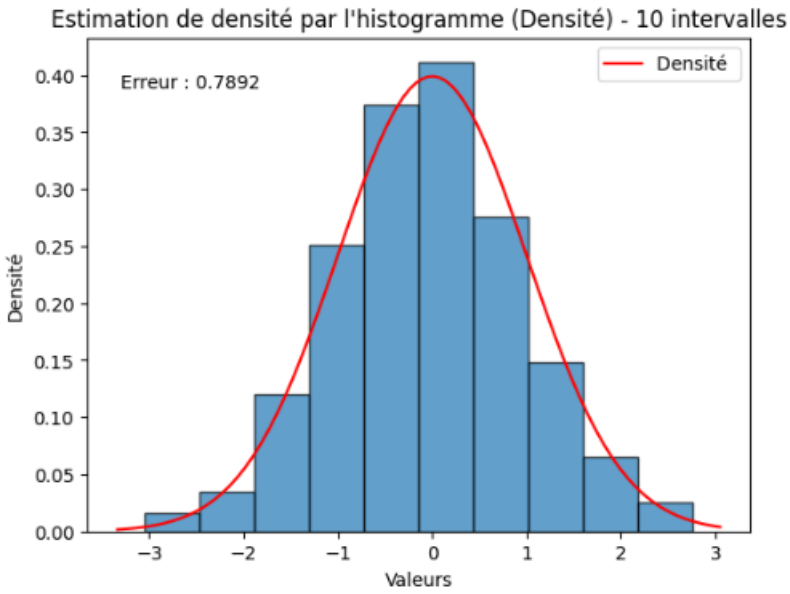

FIGURE 1.8 – Estimation de la densité avec 10 inetrvalles

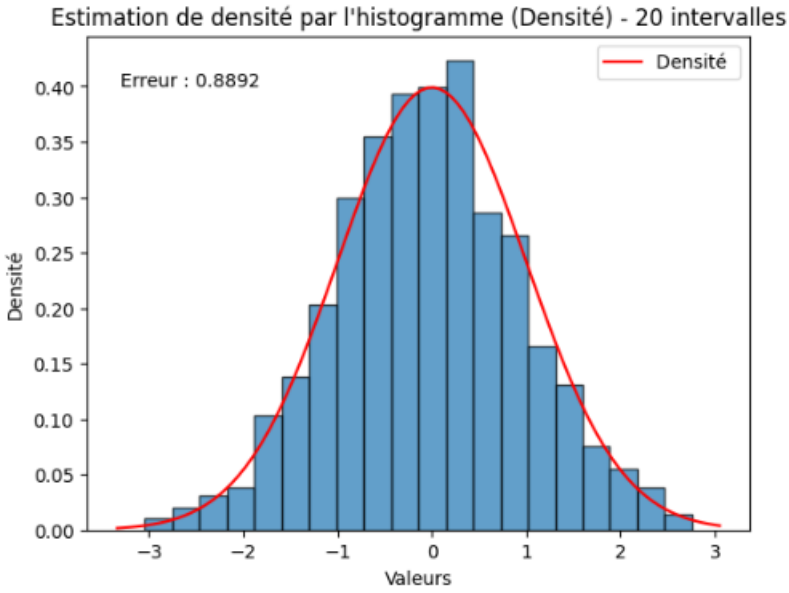

FIGURE 1.9 – Estimation de la densité avec 20 inetrvalles

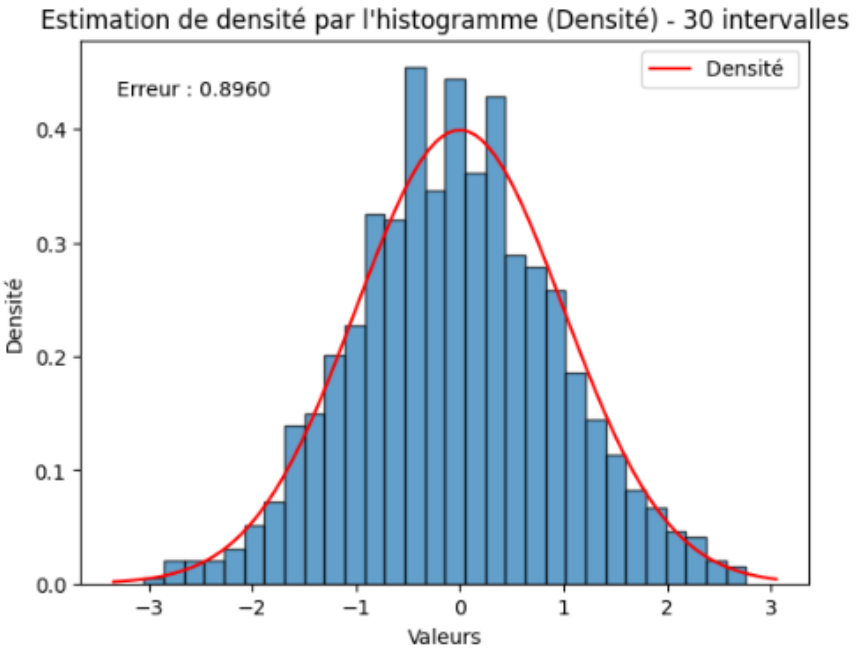

FIGURE 1.10 – Estimation de la densité avec 30 inetrvalles

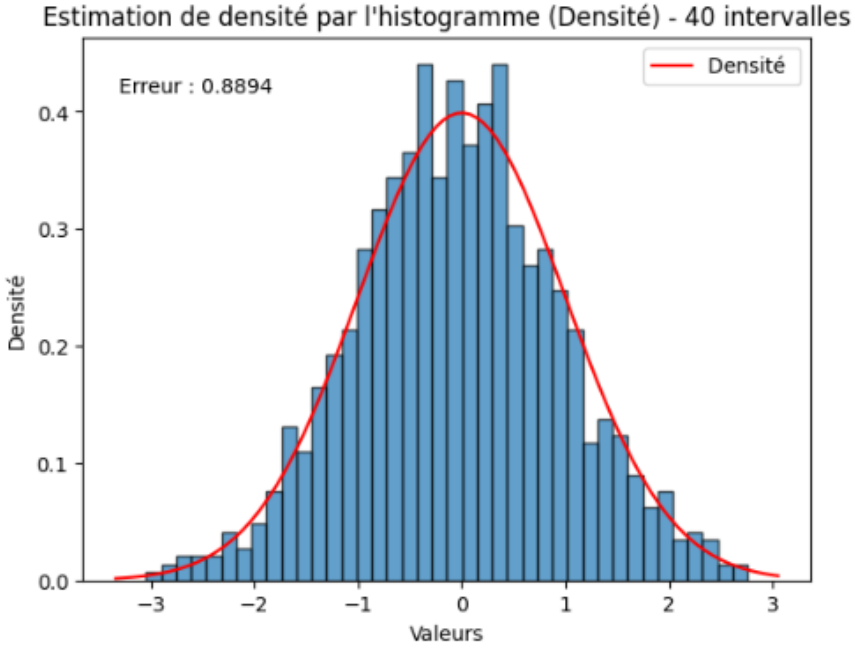

FIGURE 1.11 – Estimation de la densité avec 40 inetrvalles

D'apres les graphiques des figure ** , ** et ** des estimateurs à histogramme, l'effet du nombre d'intervalle peut etre observé de manière visuelle. Lorsque le nombre d'intervalle est faible, les barres de l'histogramme sont larges et il y a une perte de details dans la représentation de la distribution des données, cela résulte à une sous-estimation de la variation de la forme réelle de la distribution.

Les informations fines et les pics de la distribution peuvent etre lissés ou manquants, ce

qui peut donner une vision globale mais moins précise de la répartition des données.

D'un autre coté, lorsque le nombre d'intervalle est élevé, les barres de l'histogramme deviennent plus étroites et il y a une meilleure résolution dans la représentation de la distribution.
Ce la peut permettre de capturer les variations plus fines et de mieux visualiser les modes et les structures de la distribution. Cependant, un nombre d'intervalle trop élevé peut également entrainer un bruit excessif, résultant à une estimation moins lisse et moins stable de la distribution.

1.2.2 Estimateur à noyau

Un inconvénient de l'estimateur par histogramme précédent est que la fonction de densité résultante, notée $\widetilde{f}_n$, n'est pas régulière : il s'agit d'une fonction constante par morceau, qui a donc des sauts aux extrémités de chaque classe. En général, la densité à estimer est plus lisse, au moins continue.

L'estimation par noyau a pour but de répondre à cette lacune. Le principe est le suivant : si f est continue en x (ce qui est le cas pour les classes de fonctions que nous allons considérer), alors

$$\widetilde{f}(x) \;\; F'(x) \;\; \lim_{h\to 0} \frac{F(x\;h)-F(x-h)}{h} \;\; \lim_{h\to 0} \frac{F(x\;h)-F(x-h)}{2h}$$

L'idée est donc d'utiliser l'approximation suivante, pour h petit :

$$\widetilde{f}(x) \;\; \frac{F(x\;h)-F(x-h)}{2h}$$

Pour estimer la densité f, on peut donc passer par un estimateur de la fonction de répartition empirique F_n en choisissant un $h \;\; 0$ suffisamment petit pour que l'approximation ci-dessus soit valable.
On pose alors :

$$\widetilde{f}(x) \;\; \frac{F_n(x\;h)-F_n(x-h)}{2h} \;\; \frac{1}{2h}\sum_{i1}^{n} \mathbb{I}(X_i \in]x-h, x\;h[)$$

Si on pose $K_0(x) \;\; \frac{1}{2}\mathbb{I}x \in]-1,1[$, alors on a :

$$\widetilde{f}(x)\frac{1}{nh}\sum_{i1}^{n}K_0\left(\frac{x-X_i}{h}\right)$$

K_0 est appelé le noyau de Rosenblatt. Cet estimateur a le même inconvénient d'irrégularité que l'estimateur par histogramme. On a donc l'idée d'utiliser des noyaux plus réguliers, déja mentionnées précedement.

1.2.2.1 Code R et illustration graphique

Dans la suite nous allons exposer le code R et expliquer les étapes suivies.

```
'
KG=function(u){((1/sqrt(2*3.14))*(exp((-1/2)*x*x)))}
```

FIGURE 1.12 – Définition de la fonction de calcul de noyau gaussien

```
n=500 #Des valeurs aléatoires
X=rnorm(n) rnorm


x=seq(min(X), max(X), length=n)

h=n**(-1/5)

l=c()
for (i in 1:length(x) ){

  l[i]=fcharG2(X,x[i],h)

}
```

FIGURE 1.13 – Génération des données et calcule de la densité

```
plot(x,dnorm(x), type="l", lwd=2)
lines(x,l,col="red")

fcharE=function(X,xi,h){
  n=len(X)
  s=((1/n*h)*sum(KG((X-xi)/h)))
}
```

FIGURE 1.14 – Visualisation des données

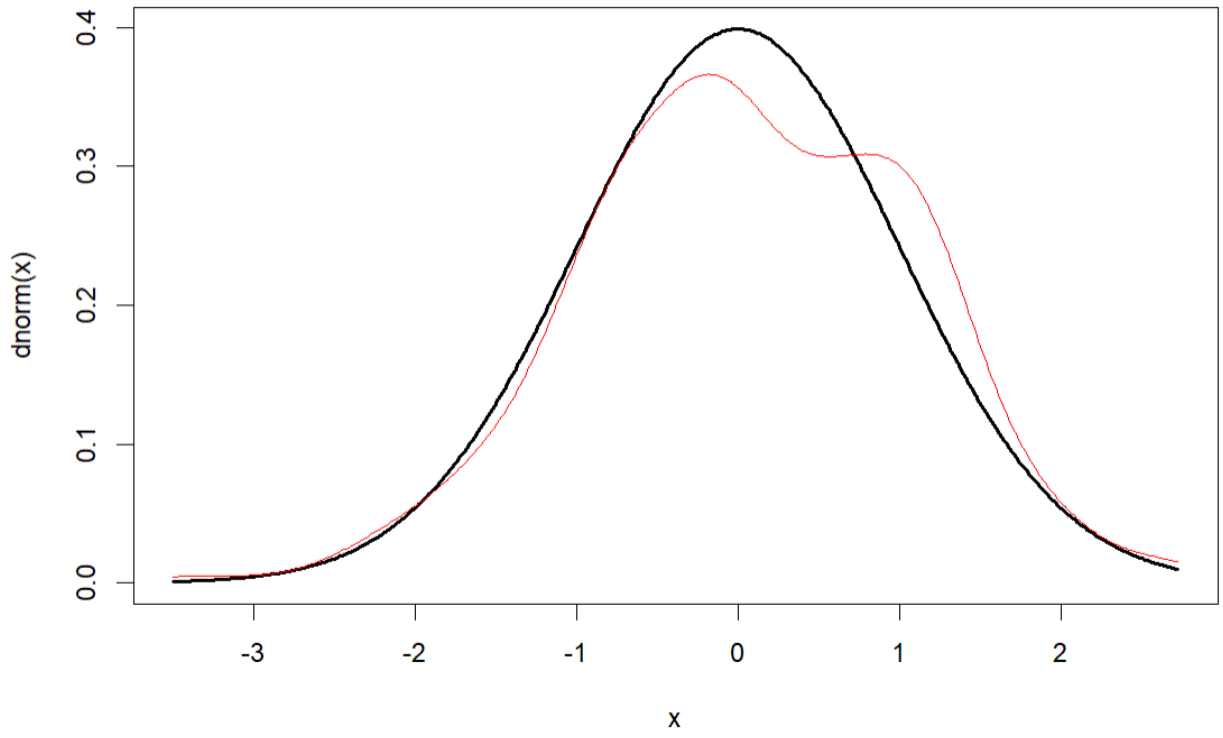

FIGURE 1.15 – Densité estimée et densité réelle avec noyau gaussien

Les figures montrent les étapes d'estimation de densité à noyau gaussien. Dans la figure ** il est illustré en rouge la courbe estimée et en noir la courbe réelle.

1.2.2.2 Code Python et illustration graphique

Dans la suite nous allons exposer le code Python et expliquer les étapes suivies, avec le noyau en premier lieu, puis avec le noyau d'epanichnikov.

Noyau gaussien

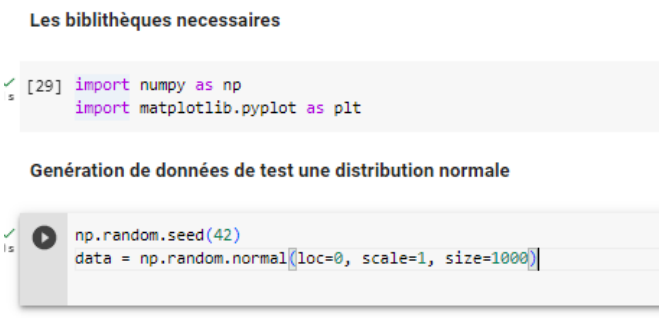

FIGURE 1.16 – Importation des bibilthèque et génération des données

La figure ** montre l'imporation des bibliothèque numpy pour le calcule numérique

et matplotlib pour la visualisation graphique. Pour la génération des données nous avons utilsé la graine du générateur de nombres aléatoires à 42 qui garantit que chaque fois se fait l'exécution du code, nous obtenons le meme ensemble de nombres aléatoires, puis nous avons généré 1000 nombres aléatoirement à partir d'une **distribution normale** avec **une moyenne de 0 et un écart type de 1**.

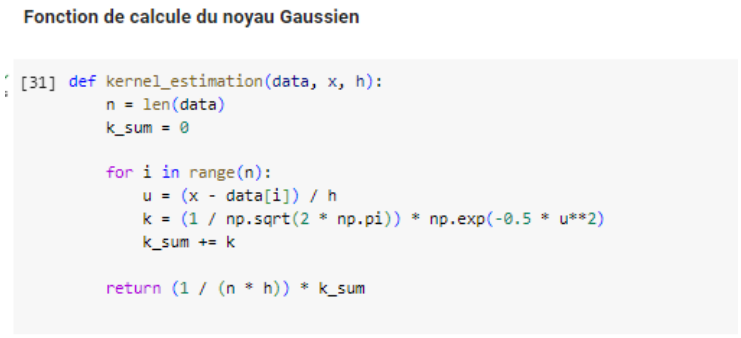

FIGURE 1.17 – Définition de la fonction du noyau gaussien

La figure ** montre la définition d'une fonction qui permet d'estimater le noyau gaussien pour une variable x donnée à partir d'un ensemble de données, et la fenetre h.

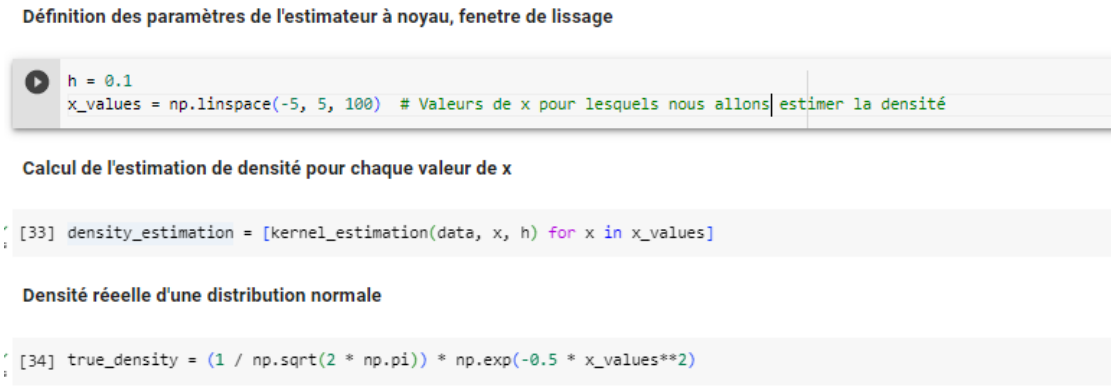

FIGURE 1.18 – Définition de la paramétre de lissage et calcule de densité

La figure ** montre l'estimation de la densité de probabilité pour chaque valeur de x dans le tableau x_values en utilisant l'estimation de noyau avec les données fournies et la fenetre h.

Affichage du résultat

```
plt.plot(x_values, density_estimation, label='Estimation')
plt.plot(x_values, true_density, label='Densité réelle')
plt.xlabel('x')
plt.ylabel('Densité')
plt.title("Estimateur à noyau vs Densité réelle")
plt.legend()
plt.show()
```

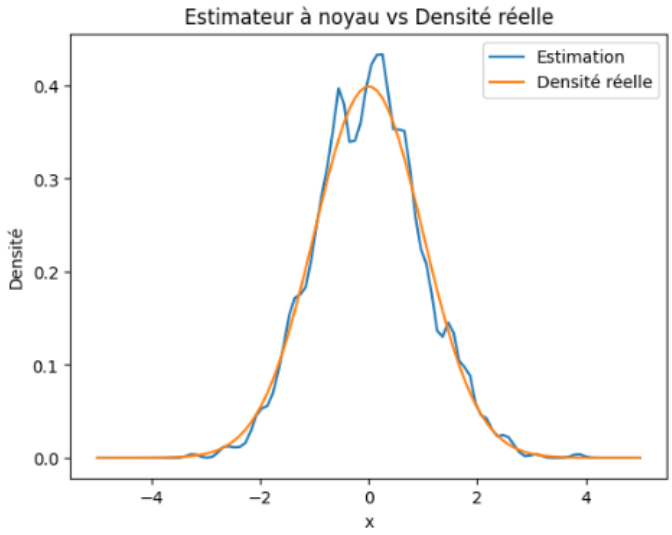

FIGURE 1.19 – Affichage de résultat de l'estimation de la densité

La prise d'écran dans la figure ** illustre le traçage des résultats de l'estimation de densité par noyau en le comparant avec la courbe de la densité de la distribution sous-jacente. On remarque que les deux courbes sont trés similaires graphiquement,on remarque une différence au voisinage de 0, sinon elles sont presque cofondus. Cela signifie que l'estimation de densité par noyau gaussien avec la fenetre h=0.1 est en bonne concordance avec la densité réelle dans notre cas.

Calcule de biais ²

```
def calculate_bias_squared(x, data):
    true_value = true_function(x)
    estimated_value = estimate_function(x, data)
    bias_squared = (true_value - estimated_value)**2
    return bias_squared
```

Calcule de la variance

```
def calculate_variance(x, data):
    estimated_values = [estimate_function(x, sample) for sample in data]
    mean_estimated_value = np.mean(estimated_values)
    variance = np.mean((estimated_values - mean_estimated_value)**2)
    return variance
```

FIGURE 1.20 – Fonctions de variance et de biais

La figure ** illustre les fonctions définies pour le calcule de la variance et le calcule de

biais au carré.

Point d'évaluation de la fonction

```
[57] x = 2
```

Affichage du résultat

```
[58] bias_squared = calculate_bias_squared(x, data)
     variance = calculate_variance(x, data)
     quadratic_risk = bias_squared + variance
     print("Biais au carré :", bias_squared)
     print("Variance :", variance)
     print("Risque quadratique ponctuel :", quadratic_risk)

     Biais au carré : 15.845717281803715
     Variance : 0.9579049897315173
     Risque quadratique ponctuel : 16.803622271535232
```

FIGURE 1.21 – Calcule de l'erreur quadratique ponctuelle

La figure ** montre le calcule de l'erreur quadratique, MSE= Variance + Biais ^{2}. Un biais au carré elevé peut indiquer un biais dans l'estimation, tandis qu'une variance élevée peut indiquer une grande variabilité dans les estimations. Le risque quadratique ponctuel combine ces deux aspects pour fournir une mesure globale de la qualité de l'estimation. Dans notre cas, le risque quadratique ponctuel est de 16.80 ce qui montre une certaine inexactitude de l'estimation.

L'impact de la fenetre de lissage sur les résultats

Les valeurs de la fenêtre de lissage à tester

```
[36] h_values = [0.05, 0.1, 0.2, 0.5]
```

```
[37] x_values = np.linspace(-5, 5, 100)
```

FIGURE 1.22 – Etudier l'impact de la vaiation du fenetre

Afiicher des résultats pour chaque fenêtre

```
[38] plt.figure(figsize=(10, 6))
     for h in h_values:
         density_estimation = [kernel_estimation(data, x, h) for x in x_values]
         true_density = (1 / np.sqrt(2 * np.pi)) * np.exp(-0.5 * x_values**2)
         plt.plot(x_values, density_estimation, label=f'h = {h}')
     plt.plot(x_values, true_density, label='Densité réelle', linestyle='--', color='black')
     plt.xlabel('x')
     plt.ylabel('Densité')
     plt.title("Estimateur à noyau pour différentes fenêtres")
     plt.legend()
     plt.show()
```

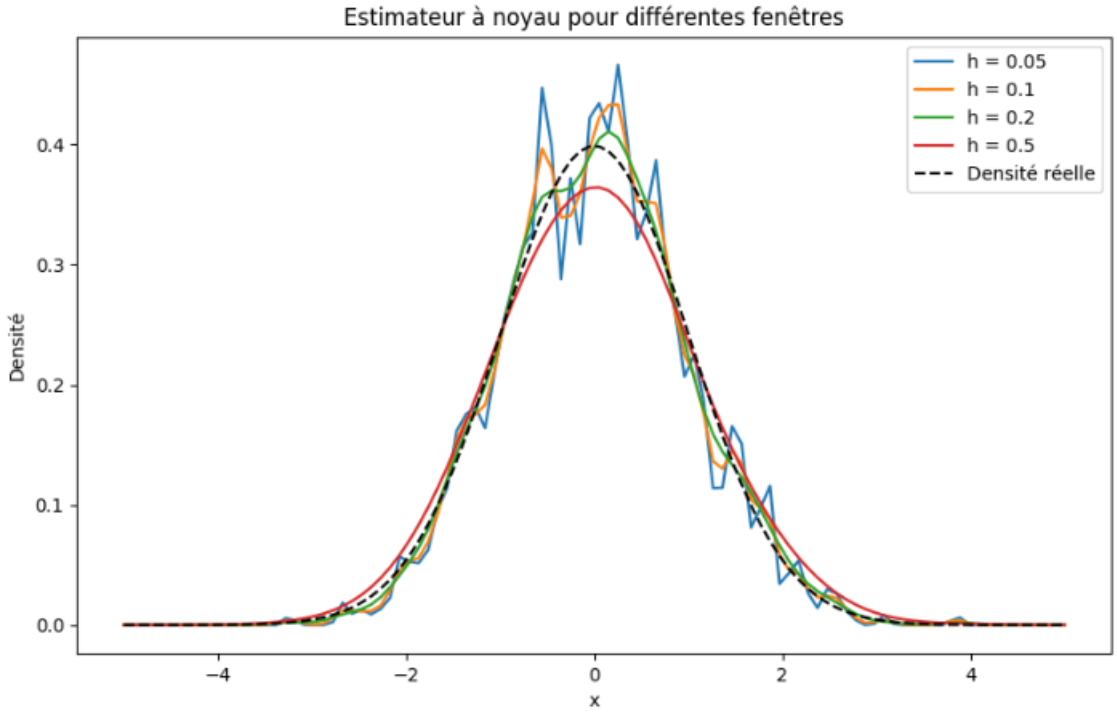

FIGURE 1.23 – Visualisation des densitées avec différentes fenetres

Les figures ** et ** montre l'effet de la variation de la fenetre, sur la courbe de la densité avec le noyau gaussien.

On remarque que toutes les courbes, sauf la courbe avec la fenetre 0.05 (en bleu) sont tres proches de la courbe d'origine si x n'apprtient pas au [-1,1], alors qu'il sont éloignées dans cette inetrvalle. Il est claire que la courbe estimée avec la fenetre égale à 0.5 (en rouge) ou la fenetre la plus grande est la plus proche de la courbe de densité d'origine, meme au voisinage de 0, alors que la courbe de la densité avec une fenetre égale à 0.05 (la plus petite fenetre) est la différente et éloignées à la courbe d'origine. Dans notre cas, la courbe la plus performante c'est avec la fenetre la plus elevée (0.5) et la courbe la moins performante c'est avec la fenetre 0.05. L'effet de lissage de h est trés claire dans cette figure.

Noyau d'Epanechnikov

Fonction de calcule du noyau d'Epanechnikov

```python
def kernel_estimation_epanichnikov(data, x, h):
    n = len(data)
    k_sum = 0

    for i in range(n):
        u = (x - data[i]) / h
        if np.abs(u) <= 1:
            k = 0.75 * (1 - u**2)
            k_sum += k

    return (1 / (n * h)) * k_sum
```

FIGURE 1.24 – Fonction de calcule de noyau d'Epanechnikov

La figure ** montre la fonction qui permet de calculer le noyau d'epanechnikov, avec les memes paramétres que la première fonction.

Noyau d'Epanechnikov

```python
plt.figure(figsize=(10, 6))
for h in h_values:
    density_estimation = [kernel_estimation_epanichnikov(data, x, h) for x in x_values]
    true_density = (1 / np.sqrt(2 * np.pi)) * np.exp(-0.5 * x_values**2)
    plt.plot(x_values, density_estimation, label=f'h = {h}')
plt.plot(x_values, true_density, label='Densité réelle', linestyle='--', color='black')
plt.xlabel('x')
plt.ylabel('Densité')
plt.title("Estimateur à noyau pour différentes fenêtres")
plt.legend()
plt.show()
```

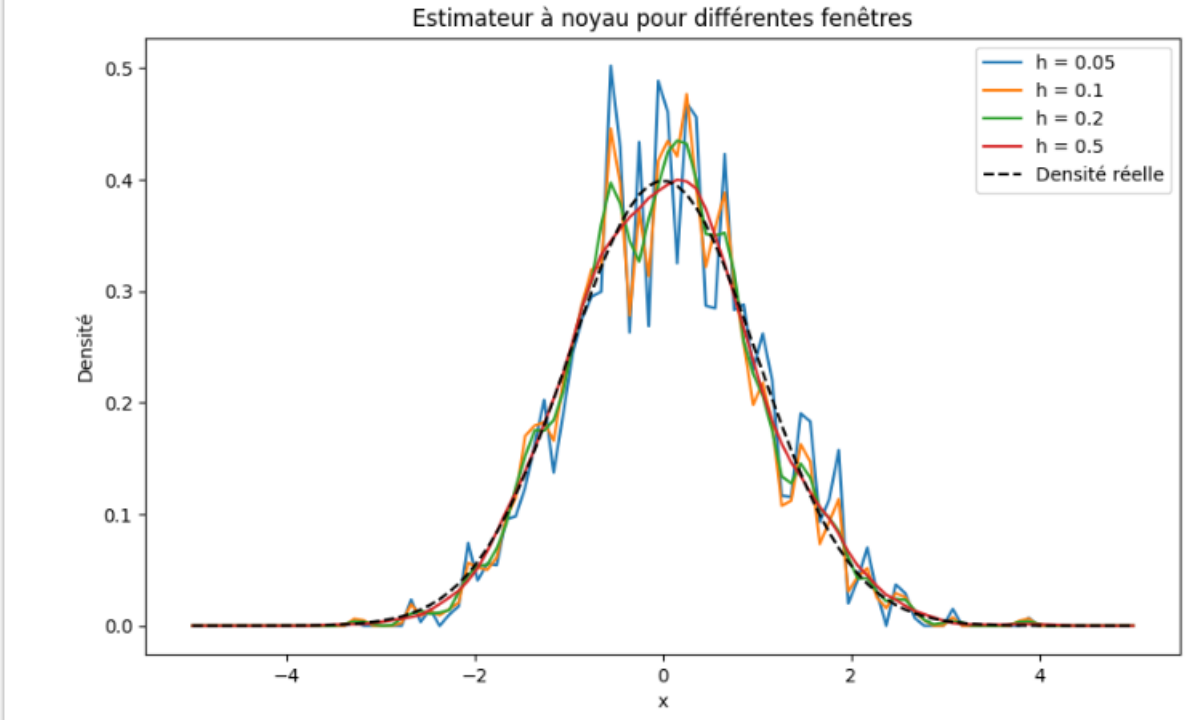

FIGURE 1.25 – Variation des fenetres avec le noyau d'Epanechnikov

La figure ** visualise la'impact de la variation du parametre de lissage h, sur la densité estimée. Les figures présentées illustrent l'effet de la variation de la fenetre sur la courbe de densité estimée à l'aide du noyau d'Epanechnikov. On observe que la plupart des courbes sont très proches de la courbe de densité d'origine, à l'exception de la courbe correspondant à une fenêtre de 0.05 (en bleu), qui differe considérablement dans l'intervalle [-1,1]. Cela met en évidence l'importance de la fenetre dans l'estimation à noyau.
Dans notre cas, la courbe la plus performante est celle avec la fenêtre la plus élevée (0.5), tandis que la courbe la moins performante est celle avec la fenetre de 0.05.
Cela démontre clairement l'effet de lissage de la fenêtre h sur l'estimation de la densité, ce qui est le meme cas avec le noyau gaussien.
En conclusion, l'ajustement de la fenetre dans l'estimateur à noyau est important pour obtenir une estimation précise de la densité.
Une fenetre trop petite peut conduire à une sous-estimation ou à une surestimation des variations locales, tandis qu'une fenetre trop grande peut entrainer une perte de détails fins. Le choix de la fenetre doit etre adapté à la structure des données et aux objectifs de l'analyse.

1.3 Risque quadratique ponctuel des estimateurs à noyaux sur la classe des espaces de Holder

Dans cette section, on s'intéresse au risque quadratique ponctuel de $\widetilde{f}_n$, étant donné $x_0 \in \mathbb{R}$, alors

$$R(\widetilde{f}_n, f) \;\; E\left[(\widetilde{f}_n(x_0) - f(x_0))^2\right]$$

Théorème

Soit $B \;\; 0$ et $L \;\; 0$ et K un noyau de carré intégrable et d'ordre $[B]$ tel que $\int |t|^B |K(t)| dt \leq \infty$.

Alors, en choisissant une fenêtre de la forme $h \;\; cn^{\frac{-1}{2B1}}$ avec une constante $c \geq 0$, on obtient :

$$\sup_f E\left[(\widetilde{f}_n(x_0) - f(x_0))^2\right] \leq cn^{\frac{-1}{2B1}} \quad \text{pour } x_0 \in \mathbb{R}$$

où C est une constante dépendant de L, B, c et K.

1.4 Construction de la noyau d'ordre l

La construction du noyau d'ordre l implique la définition d'une fonction $K_l(t)$ qui satisfait certains conditions.

Le noyau d'ordre l, noté $K_l(t)$, doit satisfaire les conditions suivantes :

1. $K_l(t)$ doit être une fonction positive pour tout t dans son domaine de définition.

2. L'intégrale du carré absolu de t élevé à la puissance l multiplié par $K_l(t)$ doit être finie :

$$\int |t|^l |K_l(t)| dt \leq \infty$$

3. $K_l(t)$ doit être symétrique par rapport à l'axe des ordonnées, c'est-à-dire que $K_l(t)$ $K_l(-t)$.

La construction d'un noyau d'ordre l peut varier en fonction des besoins spécifiques.

Construction du noyau d'ordre l par la méthode de Gram-Schmidt

La construction du noyau d'ordre l par la méthode de Gram-Schmidt implique l'orthonormalisation d'une base initiale de fonctions. Cette méthode permet de construire une base orthonormée à partir d'une base linéairement indépendante.

Supposons que nous disposons d'une base initiale de fonctions $K_0, K_1, K_2, \ldots$ que nous voulons utiliser pour construire le noyau d'ordre l. La méthode de Gram-Schmidt consiste à construire une nouvelle base orthonormée à partir de cette base initiale.

Le processus de Gram-Schmidt peut être résumé comme suit :

1. Initialisation : On définit le premier noyau d'ordre l comme étant $K_l^{(0)}$ K_0.

2. Pour m allant de 1 à l, on construit itérativement les noyaux d'ordre l en utilisant la relation de récurrence suivante :

$$K_l^{(m)}(t) \quad \frac{K_m(t) - \sum_{j0}^{m-1} \langle K_m, K_l^{(j)} \rangle K_l^{(j)}(t)}{\| K_m - \sum_{j0}^{m-1} \langle K_m, K_l^{(j)} \rangle K_l^{(j)} \|}$$

où $\langle \cdot, \cdot \rangle$ représente le produit scalaire et $\| \cdot \|$ la norme.

Après avoir appliqué la méthode de Gram-Schmidt, nous obtenons une nouvelle base orthonormée $K_l^{(0)}, K_l^{(1)}, K_l^{(2)}, \ldots$. Ces noyaux d'ordre l satisferont les conditions requises pour un noyau d'ordre l.

Construction du noyau d'ordre l à l'aide des polynômes de Legendre

La construction du noyau d'ordre l peut également être réalisée à l'aide des polynomes de Legendre. Les polynomes de Legendre sont une famille de polynomes orthogonaux définis sur l'intervalle $[-1, 1]$.

Pour construire le noyau d'ordre l, nous allons utiliser la formule suivante :

$$K_l(t) \quad \frac{1}{(2l\ 1)!!} \cdot \frac{d^l}{dt^l}\left[(t^2-1)^l\right]$$

où $(2l\ 1)!!$ représente la double factorielle et $\frac{d^l}{dt^l}$ est la dérivée l-ième.

Cette formule permet de calculer le noyau d'ordre l directement à partir des polynomes de Legendre. En évaluant cette formule pour différentes valeurs de l, on obtient les noyaux d'ordre l correspondants.

1.5 Choix de la fenetre h par validation croisée

Choix de la fenêtre h **par validation croisée**

Le choix de la fenetre h par validation croisée consiste à estimer la valeur optimale de h en utilisant la méthode de validation croisée.

La validation croisée est une méthode d'estimation de la performance d'un modèle en divisant l'ensemble de données en plusieurs sous-ensembles, et en utilisant une partie pour l'apprentissage et l'autre pour le test.

Nous supposons, la disposition d'un ensemble de données d'apprentissage $\{(x_i, y_i)\}_{i1}^{n}$, où x_i represente une observation et y_i la valeur cible correspondante.
Nous voulons estimer la fonction $\widetilde{f}_n(x)$ basé sur ces données.

La validation croisée par recherche de fenetre consiste à diviser l'ensemble de données en k plis ou sous-ensembles (*k-fold cross-validation*). Pour chaque pli k, on utilise les $k-1$ plis restants pour estimer la fonction $\widetilde{f}_{n,k-1}(x)$ en ajustant le modèle avec une fenetre h. Ensuite, on évalue la performance de la fonction estimée en utilisant le pli retenu.

Le choix de la fenetre optimale h_{opt} se fait en minimisant la moyenne des erreurs de validation croisée sur les k plis :

$$h_{\text{opt}} \quad \arg\min_h \frac{1}{k}\sum_{i1}^{k}\frac{1}{n_i}\sum_{j\in \text{Fold}_i}(y_j - \widetilde{f}_{n_i-1}(x_j))^2$$

où n_i est la taille du pli i, Fold_i est l'ensemble des indices des observations du pli i, et $\widetilde{f}_{n_i-1}(x_j)$ est la fonction estimée en utilisant les $k-1$ plis restants sans l'observation j.

1.6 Conclusion

A travers ce chapitre,nous avons mis en évidence l'importance de l'estimation non paramétrique de la densité et présente deux approches courantes : l'histogramme et l'estimateur à noyau. Nous avons aussi souligné l'impact du choix de la fenetre dans l'estimateur à noyau et le nombre d'intervalle dans la méthode d'histogramme, ainsi que la nécessité de prendre en compte les notions de biais et variance.

CONCLUSION GÉNÉRALE

En terme de conclusion, l'estimation à noyau et l'histogramme sont deux méthodes non paramétriques couramment utilisées pour estimer la densité d'une distribution à partir de données.

Chacune de ces méthodes présente ses propres avantages et inconvenients. Par exemple, le choix de la fenetre de lissage dans l'estimation à noyau et du nombre d'intervalles dans l'histogramme sont des décisions critiques pour obtenir des estimations précises de la densité.

Donc il faut explorer différentes valeurs de fenetre et de nombre d'intervalles pour trouver le bon compromis entre précision et lisibilité.

De plus, il est important de prendre en compte la structure sous-jacente des données pour choisir la méthode la plus appropriée.

NETOGRAPHIE

[1] Cours originale **Estimation de densités par estimateurs à noyau**, fourni par **Dr Salah KHARDANI**. Consulté le 23/05/2023.